Destroy this report when it is no longer needed. Do not return to sender.

PLEASE NOTIFY THE DEFENSE NUCLEAR AGENCY, ATTN: STTI, WASHINGTON, D.C. 20305, IF YOUR ADDRESS IS INCORRECT, IF YOU WISH TO BE DELETED FROM THE DISTRIBUTION LIST, OR IF THE ADDRESSEE IS NO LONGER EMPLOYED BY YOUR ORGANIZATION.
This work sponsored by the Defense Nuclear Agency under RDT&E RMSS Code B325079464 V99QAXN106215 H2590D.
The purpose of this manual is to document the Modular Air Defense Model (MADEM) and its implementation. The manual discusses the software architecture, data structures and execution requirements in detail.

The manual is intended for use by programmers charged with maintaining or modifying MADEM. The MADEM Analyst Manual discusses the processes modeled, their structure and relationships, and the various assumptions made.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREFACE</td>
<td>1</td>
</tr>
<tr>
<td>LIST OF ILLUSTRATIONS</td>
<td>5</td>
</tr>
<tr>
<td>I INTRODUCTION</td>
<td>7</td>
</tr>
<tr>
<td>II SOFTWARE ARCHITECTURE</td>
<td>8</td>
</tr>
<tr>
<td>A. Introduction</td>
<td>8</td>
</tr>
<tr>
<td>B. Implementation Language</td>
<td>8</td>
</tr>
<tr>
<td>C. Data Structures</td>
<td>9</td>
</tr>
<tr>
<td>D. Simulation Control</td>
<td>9</td>
</tr>
<tr>
<td>1. Top Level Control</td>
<td>9</td>
</tr>
<tr>
<td>2. Initialization</td>
<td>13</td>
</tr>
<tr>
<td>3. Storage Management</td>
<td>13</td>
</tr>
<tr>
<td>4. Event Control</td>
<td>14</td>
</tr>
<tr>
<td>E. Diagnostics</td>
<td>17</td>
</tr>
<tr>
<td>1. Debug Routines</td>
<td>17</td>
</tr>
<tr>
<td>2. Data Structure Display Routines</td>
<td>19</td>
</tr>
<tr>
<td>F. Software Components</td>
<td>20</td>
</tr>
<tr>
<td>1. Processor Configuration</td>
<td>20</td>
</tr>
<tr>
<td>2. Preprocessor Functions and Subroutines</td>
<td>22</td>
</tr>
<tr>
<td>3. Main Processor Functions and Subroutines</td>
<td>26</td>
</tr>
<tr>
<td>4. Postprocessor Functions and Subroutines</td>
<td>34</td>
</tr>
<tr>
<td>III DATA STRUCTURE DOCUMENTATION</td>
<td>35</td>
</tr>
<tr>
<td>A. Introduction</td>
<td>35</td>
</tr>
<tr>
<td>B. Common Blocks</td>
<td>40</td>
</tr>
<tr>
<td>C. Data Base Structure-DATFILE</td>
<td>47</td>
</tr>
<tr>
<td>1. Data Block Index</td>
<td>47</td>
</tr>
<tr>
<td>2. Description</td>
<td>47</td>
</tr>
<tr>
<td>3. Structure Overview</td>
<td>48</td>
</tr>
<tr>
<td>4. Block Specifications</td>
<td>50</td>
</tr>
<tr>
<td>5. Linkages to Other Data Structures</td>
<td>59</td>
</tr>
<tr>
<td>6. Notes</td>
<td>60</td>
</tr>
</tbody>
</table>
### TABLE OF CONTENTS (CONTINUED)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>III D. Simulation Control Structures</td>
<td>61</td>
</tr>
<tr>
<td>1. Event Lists</td>
<td>62</td>
</tr>
<tr>
<td>2. Hex Address Tree and Related Lists</td>
<td>66</td>
</tr>
<tr>
<td>E. Command/Control Structures</td>
<td>71</td>
</tr>
<tr>
<td>1. Player Lists</td>
<td>72</td>
</tr>
<tr>
<td>2. C2 Trees</td>
<td>78</td>
</tr>
<tr>
<td>3. Passive Target List</td>
<td>86</td>
</tr>
<tr>
<td>F. RED Structures</td>
<td>89</td>
</tr>
<tr>
<td>1. Red Theater Commander</td>
<td>90</td>
</tr>
<tr>
<td>2. Potential Target List</td>
<td>98</td>
</tr>
<tr>
<td>3. Assigned Target List</td>
<td>103</td>
</tr>
<tr>
<td>4. Nonavailable Target List</td>
<td>109</td>
</tr>
<tr>
<td>5. Corridor Description List</td>
<td>114</td>
</tr>
<tr>
<td>6. Attack Requirements/Allocations Lists</td>
<td>121</td>
</tr>
<tr>
<td>7. Red Air Bases</td>
<td>130</td>
</tr>
<tr>
<td>8. Red Flights</td>
<td>137</td>
</tr>
<tr>
<td>G. Blue Structures</td>
<td>153</td>
</tr>
<tr>
<td>1. Allied Tactical Air Force (ATAF)</td>
<td>154</td>
</tr>
<tr>
<td>2. Sector Operations Centers (SOC)</td>
<td>159</td>
</tr>
<tr>
<td>3. Combat Reporting Centers (CRC)</td>
<td>164</td>
</tr>
<tr>
<td>4. Blue Airbases (B-AB)</td>
<td>175</td>
</tr>
<tr>
<td>5. Blue Flights (B-FLT)</td>
<td>182</td>
</tr>
<tr>
<td>6. Battalion Operations Centers (BOC)</td>
<td>198</td>
</tr>
<tr>
<td>7. Sam Battery (BTRY)</td>
<td>215</td>
</tr>
<tr>
<td>8. Passive Targets (PT)</td>
<td>231</td>
</tr>
<tr>
<td>H. Storage Space Management</td>
<td>235</td>
</tr>
<tr>
<td>1. Data Block Index</td>
<td>235</td>
</tr>
<tr>
<td>2. Description</td>
<td>235</td>
</tr>
<tr>
<td>3. Structure Overview</td>
<td>235</td>
</tr>
<tr>
<td>4. Block Specifications</td>
<td>238</td>
</tr>
<tr>
<td>5. Linkages to Other Data Structures</td>
<td>238</td>
</tr>
<tr>
<td>6. Notes</td>
<td>238</td>
</tr>
<tr>
<td>Appendices</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------</td>
<td>------</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>239</td>
</tr>
<tr>
<td>Executive Requirements</td>
<td></td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>263</td>
</tr>
<tr>
<td>Modular Information Data Access System (MIDAS)</td>
<td></td>
</tr>
<tr>
<td>APPENDIX C</td>
<td>281</td>
</tr>
<tr>
<td>The Program Design Language</td>
<td></td>
</tr>
<tr>
<td>APPENDIX D</td>
<td>291</td>
</tr>
<tr>
<td>MIDAS Tables</td>
<td></td>
</tr>
<tr>
<td>APPENDIX E</td>
<td>303</td>
</tr>
<tr>
<td>The Dynamic event scheduling Algorithm</td>
<td></td>
</tr>
<tr>
<td>APPENDIX F</td>
<td>311</td>
</tr>
<tr>
<td>MADEM Event Code Definitions</td>
<td></td>
</tr>
<tr>
<td>APPENDIX G</td>
<td>317</td>
</tr>
<tr>
<td>Random Number Generator Calls</td>
<td></td>
</tr>
<tr>
<td>APPENDIX H</td>
<td>319</td>
</tr>
<tr>
<td>MADEM Subroutine Reference Lists and Calling Hierarchies</td>
<td></td>
</tr>
<tr>
<td>APPENDIX I</td>
<td>469</td>
</tr>
<tr>
<td>MADEM Debug Routines</td>
<td></td>
</tr>
<tr>
<td>APPENDIX J</td>
<td>481</td>
</tr>
<tr>
<td>MADEM Data Structure Cross Reference</td>
<td></td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>II-1</td>
<td>Top Level Subroutine Calling Hierarchy</td>
</tr>
<tr>
<td>II-2</td>
<td>Event Control Subroutine Calling Hierarchies</td>
</tr>
<tr>
<td>II-3</td>
<td>Event Processing Overview</td>
</tr>
<tr>
<td>II-4</td>
<td>MADEM Processor Configuration</td>
</tr>
<tr>
<td>III-1</td>
<td>Data Base Structure Diagram</td>
</tr>
<tr>
<td>III-2</td>
<td>Discreet Event List Diagram</td>
</tr>
<tr>
<td>III-3</td>
<td>Hex Tree Structure Diagram</td>
</tr>
<tr>
<td>III-4</td>
<td>Player List Structure Diagram</td>
</tr>
<tr>
<td>III-5</td>
<td>C2 Tree Structure Diagram</td>
</tr>
<tr>
<td>III-6</td>
<td>Blue Command/Control Structure</td>
</tr>
<tr>
<td>III-7</td>
<td>Red Command/Control Structure</td>
</tr>
<tr>
<td>III-8</td>
<td>Passive Target List Structure Diagram</td>
</tr>
<tr>
<td>III-9</td>
<td>Red Theater Commander Structure Diagram</td>
</tr>
<tr>
<td>III-10</td>
<td>Potential Target List Structure Diagram</td>
</tr>
<tr>
<td>III-11</td>
<td>Potential/Assigned Target List Structures</td>
</tr>
<tr>
<td>III-12</td>
<td>Assigned Target List Structure Diagram (Red Commander Perception List)</td>
</tr>
<tr>
<td>III-13</td>
<td>Nonavailable Target List Structure Diagram</td>
</tr>
<tr>
<td>III-14</td>
<td>Attack Sectors</td>
</tr>
<tr>
<td>III-15</td>
<td>Corridor Description List Structure Diagram</td>
</tr>
<tr>
<td>III-16</td>
<td>Relationship of HEX LINK Lists to Corridor Boundaries</td>
</tr>
<tr>
<td>III-17</td>
<td>Attack Requirements/Allocations List Structure Diagram</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------------------------</td>
</tr>
<tr>
<td>III-18</td>
<td>Red Air Base Structure Diagram</td>
</tr>
<tr>
<td>III-19</td>
<td>Red Flight Structure Diagram</td>
</tr>
<tr>
<td>III-20</td>
<td>Flight Data Base Structure Diagram</td>
</tr>
<tr>
<td>III-21</td>
<td>Allied Tactical Air Force Structure Diagram</td>
</tr>
<tr>
<td>III-22</td>
<td>Sector Operations Center Structure Diagram</td>
</tr>
<tr>
<td>III-23</td>
<td>Combat Reporting Center Structure Diagram</td>
</tr>
<tr>
<td>III-24</td>
<td>Blue Airbase Structure Diagram</td>
</tr>
<tr>
<td>III-25</td>
<td>Blue Flight Structure Diagram</td>
</tr>
<tr>
<td>III-26</td>
<td>Flight Data Base Structure Diagram</td>
</tr>
<tr>
<td>III-27</td>
<td>Battalion Operations Center Structure Diagram</td>
</tr>
<tr>
<td>III-28</td>
<td>Digested Information List Configurations</td>
</tr>
<tr>
<td>III-29</td>
<td>Digested Information List and Perception List</td>
</tr>
<tr>
<td>III-30</td>
<td>SAM Battery Structure Diagram (HAWK and HERC)</td>
</tr>
<tr>
<td>III-31</td>
<td>SAM Battery Structure Diagram (PATRIOT)</td>
</tr>
<tr>
<td>III-32</td>
<td>BTRYDIL Configurations</td>
</tr>
<tr>
<td>III-33</td>
<td>Battery BTRYDIL Perception List Association</td>
</tr>
<tr>
<td>III-34</td>
<td>Passive Target List Structure Diagram</td>
</tr>
<tr>
<td>III-35</td>
<td>I-Space Array Configuration</td>
</tr>
<tr>
<td>III-36</td>
<td>Garbage Collection Matrix Structure Diagram</td>
</tr>
</tbody>
</table>
CHAPTER I
INTRODUCTION

The purpose of this manual is to document the MADEM Software Architecture. It is designed for use by programmers charged with maintaining or modifying MADEM. Those wishing to use MADEM in a study are referred to the MADEM ANALYST MANUAL.

Chapter II of this manual provides an overview of the MADEM Software Architecture. Its' primary objective is to associate particular subroutines with particular simulation control and modeling functions. A secondary objective is to provide a basic explanation of the data storage system used in MADEM. A general knowledge of the subroutines combined with an understanding of the data storage system provide a context within which the more detailed information in Chapter III, the Appendices and the Source Code may be used.

Chapter III of this manual contains detailed Data Structure Documentation. This information is crucial to an understanding of the MADEM Software. MADEM uses a complex list processing system to store and retrieve data. In this system the relationship among various blocks of data is as important to the functioning of the software as the contents of the blocks. Groups of related data blocks form data structures which, in effect, "drive" the software. Therefore, it is impossible to understand the Source Code without a clear picture of the data structures and their contents.
CHAPTER II
SOFTWARE ARCHITECTURE

A. INTRODUCTION

The primary objective of this chapter is to associate particular subroutines with the model functions they control. A secondary objective is to introduce the user to the data storage system used in MADEM. More detailed information on subroutines and data structures is contained in the Appendices, the Source Code and in Chapter III of this report.

B. IMPLEMENTATION LANGUAGE

MADEM was implemented in accordance with principles of toplodown structured programming. Prior to code generation, the MADEM design was specified in a BDM developed Program Design Language (PDL). The concepts and procedures involved in the use of PDL are discussed in Appendix C. In essence, the PDL for a given subroutine constitutes the equivalent of a logic flow diagram for the subroutine. The PDL for each MADEM subroutine appears in the source code. Information on the basic purpose, the inputs and outputs, and the calling interfaces for each subroutine is given in Appendix I. The overall model structure is reflected by the subroutine call diagrams in Appendix J.

The design specified by the MADEM PDL was implemented in FORTRAN. A special precompiler was used to allow convenient access to MADEM's many dynamically allocated data blocks. This precompiler is the BDM developed Modular Information Data Access system (MIDAS), which is described in Appendix B. MIDAS provides two important capabilities. The first allows automatic replacement of specified one line macro-instructions by corresponding sets of FORTRAN instructions. This feature is used to insure uniformity in the definition of named COMMONs from routine to routine. The second important MIDAS capability allows reference by name to elements of dynamically allocated data blocks. Thus, for example, MIDAS may permit a
reference of the form $P.$TYPE.$RATE$ to be used in place of a corresponding FORTRAN reference $\text{ITR(ITR(P+L)+5)}$, where $P$ has been declared a pointer to a data block of a type known to MIDAS.

C. DATA STRUCTURES

Most of the data employed by MADEM is stored internally in data blocks dynamically allocated from various storage arrays. Many types of such blocks are employed, each having a block name and most having a set of element names (and types) known to the MIDAS translator. Having established a pointer to such a block and having indicated the block type in a MIDAS "DECLARE" statement, a programmer may then reference any entry in the block by name. Chapter III presents descriptions for all of the MADEM data block types; the MIDAS name for each block is given, along with comments on the use of the block within the simulation. The name, type, and meaning (use) of each element of the block is also given. The structure and contents of each type of data block as implemented in MIDAS code is also indicated in Appendix E.

In addition to common storage arrays for dynamically allocated data blocks, MADEM employs a number of other named COMMON storage areas for holding simulation control information and temporary working data. Chapter III presents each of the COMMONs and indicates the meaning (use) of each element.

D. SIMULATION CONTROL

The purpose of this section is to discuss Simulation Control and Management Functions within MADEM and to associate particular subroutines with these functions.

1. Top Level Control

The Top Level Control routines for MADEM are shown in Figure II-1. MADEM is the main routine which invokes the four major control routines: OTHERDAT, LRKPRS, DELADD, and CONTROL. At the users option MADEM
NOTE: ONLY SUBROUTINES WHOSE PRIMARY FUNCTION IS CONTROL HAVE BEEN INCLUDED IN THIS DIAGRAM. FOR A COMPLETE CALLING HIERARCHY SEE APPENDIX J.

Figure II-1. Top Level Subroutine Calling Hierarchy
may also invoke ENTSTAT which provides a variety of subroutine diagnostics at various points in program execution.

OTHERDAT controls input to the (NON-UOIL) Data Base. The contents and structure of this data base are described in detail in Chapter III of this report. The data input procedure is documented in Chapter IV of the MADEM ANALYST MANUAL.

LRKPRS controls semantic processing of the User Oriented Input Language (UOIL) inputs. The data structures which are built from these inputs are discussed in Chapter III of this report. The Input Language and data requirements are documented in Chapter IV of the MADEM Analyst Manual. LRKPRS invokes two secondary control subroutines- SEMANT and NXTSYM. LRKPRS and NXTSYM parse the input sentences and convert them to "ICODES" which are passed to SEMANT. SEMANT converts these "ICODES" to data structures of various kinds. Users are cautioned to avoid modifications to LRKPRS, NXTSYM and SEMANT unless they are well versed in semantic processing techniques.

DELADD controls processing of discrete events in MADEM. It adds "events" to a Leftist EVENT TREE structure which is used to sort events scheduled by the various program modules. Events control is discussed in further detail in section D.4 and in Appendix E. of this manual.

CONTROL invokes three major control subroutines- LTREE, SELECT, and HLTPNT. LTREE removes "EVENTS" from the EVENT TREE constructed by DELADD. Events are removed and processed in order of their occurrence. These "EVENTS" are then processed by SELECT. Based on an internal coding system for EVENTS, (see Appendix F) SELECT invokes one of nine program modules. Each of these modules (represented in Figure D-1 by their main control routines) performs all of the functions required by the EVENTS. The functions of each module are summarized in Table II-1. Additional information on each module is provided in Section F.3, and Appendix I of this Manual as well as in the source code. If a Termination of Simulation event is found by LTREE, the simulation termination subroutine HLTPNT is invoked by CONTROL. HLTPNT controls printing of termination messages and output files to be stored for subsequent use by the main processor and post processor.
<table>
<thead>
<tr>
<th>MODULE CONTROL/ROUTINE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSIGN</td>
<td>COMBAT REPORTING CENTER MAKES OF INTERCEPTORS TO RED FLIGHTS</td>
</tr>
<tr>
<td>ATTACK</td>
<td>CARRY OUT GROUP ATTACK BY RED FLIGHTS</td>
</tr>
<tr>
<td>COMMIO</td>
<td>COMMUNICATIONS TRANSMISSION</td>
</tr>
<tr>
<td>DOGFITE</td>
<td>AIR TO AIR COMBAT PROCESSES</td>
</tr>
<tr>
<td>ENGAGE</td>
<td>SURFACE TO AIR MISSLE ENGAGEMENT DECISIONS</td>
</tr>
<tr>
<td>FLY</td>
<td>AIRCRAFT MOVEMENT</td>
</tr>
<tr>
<td>NEYBOR</td>
<td>DETERMINE NEARBY UNITS AND SCHEDULE ALL UNITS FOR A CHANCE TO &quot;SEE&quot; AN ACTION</td>
</tr>
<tr>
<td>PERCEPT</td>
<td>CONTROLS PERCEPTION OF OTHER UNITS</td>
</tr>
<tr>
<td>PLAN</td>
<td>RED THREAT PLANNING</td>
</tr>
<tr>
<td>PONDER</td>
<td>UNIT INFORMATION PROCESSING</td>
</tr>
<tr>
<td>TOWER</td>
<td>AIRBASE OPERATIONS</td>
</tr>
<tr>
<td>UMPIRE</td>
<td>SIMULTAINEOUS EVENT CONTROL</td>
</tr>
</tbody>
</table>
2. **Initialization**

The subroutines devoted solely to initialization are:

1. BDALT
2. BDLEX
3. BDLRK
4. BDPARS
5. BLKDAR
6. FETCH

All but one of these (BLKDAR) are used only by the Lexical Analyzer Routines. They contain parsing tables which are essential to the UOIL decoding process. BLKDAR is used to initialize the MADEM main routine. FETCH reads input data files for the main routine.

3. **Storage Management**

To provide maximum flexibility in the types of scenarios which can be handled by MADEM, a system of dynamic storage allocation is employed. The majority of data in MADEM is stored in a single array (ISPACE, or SPACE) from which storage space is dynamically allocated. Thus, deletion of data of one type frees space for data of other types. The subroutines exclusively devoted to storage allocation are:

1. GIMME
2. RELEASE
3. RELIST

Subroutines GIMME and RELEASE are the general purpose storage management routines employed in MADEM. To obtain a data block of length N (N=1-20) the statement:

```
Call GIMME (PN)
```

is used. The return value P is a pointer to the allocated block (address of the first word of the block in the ISPACE array). When the block is no longer needed, the statement:

```
Call RELEASE (P,N)
```

releases the block for subsequent reallocation. The released block is placed in a garbage collection matrix (see Chapter III.H) which consists of lists of blocks of various sizes. When a block of a given size is required, GIMME searches this list to find a release block of the desired size before it allocates virgin storage space.
Because very large blocks are seldom called for, released blocks over 20 words long are broken down into more commonly used four word blocks by the subroutine RELIST. Further details on storage space management are contained in Chapter III of this manual.

4. **Event Control**

Since MADEM is an event stepped simulation, the management of notices for pending events is an important component of simulation control. The following subroutines are devoted to this function:

1. DELADD
2. LTREE

DELADD is used to add events to a quisisorted LEFTIST EVENT TREE in which the nearest events in time are placed closest to the top of the tree. The top event block on the tree always contains the next event code to be processed. LTREE is used to extract a pending event from the EVENT TREE. The calling hierachies of DELADD and LTREE are illustrated in Figure II-2.

DELADD invokes three subroutines - GIMME, SNAP, AND LTRMRG. GIMME allocates storage space for the new event block, SNAP adds the newly created block to the event tree, and LTRMRG sorts the event tree to place all of the event blocks in the correct order with respect to the top of the tree.

LTREE removes the event block from the tree, passes the event code stored in the block to SELECT and releases the storage space formerly used by the event block by invoking the subroutine RELEASE.

An overview of event processing is shown in Figure II-3. Event blocks are added to the event tree as actions are taken by the various program modules invoked by SELECT. SELECT is in turn driven by event codes extracted from the event tree by LTREE. This process continues throughout the simulation until a termination event is found by LTREE. When this occurs, HLTPNT is called by CONTROL and the simulation is HALTED. Detailed documentation of this dynamic event scheduling algorithm as well as a listing of event codes may be found in Appendices E and F of this manual.
ADDS EVENTS TO THE EVENT TREE

GIMME
SNAP
LTRMRG

EXTRACTS EVENTS FROM THE EVENT TREE

RELEASE

Figure II-2. Event Control Subroutine Calling Hierarchies
Figure II-3. Event Processing Overview
E. **DIAGNOSTICS**

The purpose of this section is to document the diagnostic capabilities which have been built into the MADEM software. A working knowledge of these debug and data structure display routines is essential to anyone who must maintain or modify MADEM. The operation of all diagnosis subroutines is discussed in detail in Appendix J of this report.

1. **Debug Routines**

There are over 300 subroutines in the current version of MADEM. A program of this size cannot be developed and maintained without some resident debug and error recovery capability built into the software. In MADEM the following subroutines are devoted to this function:

- **ENTRYP**
- **EXITP**
- **ENTSTAT**
- **ICHECK**
- **RECOVR**
- **ITRAP**
- **RECCON**
- **RECER**
- **ROUTER**

MADEM's resident debug capabilities are based on the subroutine entry and exit tracking routines ENTRYP and EXITP. ENTRYP and EXITP are called at the beginning and end of nearly all routines in MADEM. Together they construct a circular list and pushdown stack of subroutine calls. They also construct a vector of subroutine entry counts and a corresponding vector of cumulative subroutine execution times. The circular list and pushdown stack allow the user to track the execution of all subroutines and to pinpoint fatal error locations. The vectors of entry counts and execution times provide useful information on reusage and efficiency of subroutines. At the user's option ENTRYP and EXITP will print out subroutine entry and exit messages for specified subroutines during normal program execution. Also at the user's option, ENTRYP and EXITP will call debug routines ICHECK and ITRAP (explained below) for specified subroutine calls to ENTRYP and EXITP.
A frequent list processing bug involves a zero or one digit ISPACE pointer. For this reason, the first ten words of ISPACE are reserved for nonuse, and should always be zero when none of these bugs occur. Debug routine ITRAP checks that area of ISPACE and stops the simulation by calling HALT whenever nonzero values are found there.

If it is desired to find out when other locations in ISPACE change value, debug routine ICHECK can be used. ICHECK holds pointers to up to ten ISPACE locations, as defined by input parameters on each run. Any time one of these ISPACE locations has a change in value, a one line message is printed indicating the old value, the new value, and the routine being executed when the change was noticed.

The vectors of subroutine entry counts, cumulative and average execution times, and flags indicating the ENTRYP and EXITP options specified are printed by ENTSTAT.

RECOR is a CDC System routine which is automatically activated within MADEM, but may be turned off at the user's option. It allows the program to regain control at the time that abnormal job termination would otherwise occur. It calls the MADEM subroutine RECCON in the event of catastrophic program failure.

RECCON can only be activated by RECOR. RECCON calls HALT, which stops the simulation and prints debug information.

ROUTER, which can be called from any routine, prints the subroutine name, the circular list subroutines called, and the pushdown stack containing the calling hierarchy.

RECER is the same as router except that RECER does not know the calling subroutine's name.

DBGREAD is the subroutine that reads, interprets, and processes all the debug input parameters that may be specified for each run. See Appendix J for further details.

HALT is the routine that is called whenever the simulation is to be stopped, whether it be abnormal or normal terminations. HALT saves the HOLD files and calls the appropriate subroutines to print valuable debug information. HALT prints the name of the 1st routine executed as well as
the reason for termination, the game time of the simulation, and the number of events executed in the volume. **HALT** then calls the following routines:

<table>
<thead>
<tr>
<th>Routine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRACE</td>
<td>prints current calling hierarchy</td>
</tr>
<tr>
<td>RECER</td>
<td>(see above)</td>
</tr>
<tr>
<td>ENTSTAT</td>
<td>(see above)</td>
</tr>
<tr>
<td>CLIST</td>
<td>prints key pointer values</td>
</tr>
<tr>
<td>ISDUMP</td>
<td>if selected, prints entire ISPACE array.</td>
</tr>
</tbody>
</table>

2. **Data Structure Display Routines**

Because of the complex nature of the data structures used in **MADEM** and their importance to the operation of **MADEM** subroutines, a series of data structure and common block display subroutines have been developed. These subroutines include the following:

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLIST</td>
<td>is used to print out the contents of all major common blocks used in <strong>MADEM</strong>. Key information contained in these blocks includes array dimensions and pointer values. <strong>ISDUMP</strong> is used to print out the contents of the dynamic storage array <strong>ISPACE</strong> within which all data structures reside. The outputs from <strong>CLIST</strong> and <strong>ISDUMP</strong> may be used to manually trace data structures in <strong>ISPACE</strong>. However, since manual data structure tracing rapidly becomes time consuming and error prone, a variety of automated data structure display subroutines have been constructed. The two foremost of these are <strong>DISPDAT</strong> and <strong>NIPULATOR</strong>. <strong>DISPDAT</strong> outputs a formatted display of the <strong>DATA BASE STRUCTURE</strong>, the <strong>EVENT TREE STRUCTURE</strong> and specified portions of <strong>ISPACE</strong>. In addition to the above display subroutines, a series of data block display subroutines has been developed which allows the user to build &quot;custom&quot; data structure display systems to suit special needs. Each data block has its own display subroutine which can display the block or all blocks in a linked list at the user's option. These block display subroutines can be placed under the control of special purpose control routines.</td>
</tr>
</tbody>
</table>
developed by the user. User supplied control subroutines combined with the block display subroutines constitute a "modular" data structure display system which can be reconfigured at the user's option. Further details are provided in Appendix J.

F. SOFTWARE COMPONENTS

The purpose of this section is to introduce the reader to the major components of the MADEM software system. Detailed operating instructions may be found in Appendix A of this manual and Chapter IV of the analyst manual.

1. Processor Configuration

MADEM has three major software components - a preprocessor, a main processor, and a postprocessor. The overall configuration of these processors and their input and output files is illustrated in Figure II-4.

The preprocessor (sometimes referred to as INITBIN) reads user inputs, which include the data base and red threat planning specifications, translates these inputs into appropriate data structures and carries out the red threat planning process. The resulting red attack plan is then saved on a "HOLD FILE" for subsequent use by the main processor also outputs a printed summary of the red attack plan and a 'HISTORY FILE'. The HISTORY FILE contains a record of unit creation events which can be used by the postprocessor to construct summary tables of the types of units created.

The main processor (sometimes referred to as RUNBIN) is used to simulate the actual combat processes which result from the red attack. The main processor is run in a cyclical fashion. The initial main processor run is made using the RED ATTACK PLAN HOLD FILE as input. Combat processes are then carried out for approximately one hour of game time after which execution is terminated and as HOLD FILE containing the status of all units at the end of the hour is output. This HOLD FILE then becomes the input for the next main processor cycle. Each of these main processor cycles is referred to as a volume. As many volumes can be run as one required to simulate the desired duration of conflict. However, the validity of the

20
Figure II-4. MADEM Processor Configuration
simulation probably declines rapidly after 8-11 hours of combat. The main processor also outputs an EVENT TRACE and a History File. The EVENT trace records all of combat events which occurred during the volume. The History File contains a record of key events which can be used by the postprocessor to summarize the battle.

The postprocessor (sometimes referred to as HISTBIN) reads History Files output by the preprocessor and main processor and outputs a variety of tabular summaries of the battle. These outputs include the following:

1. Red Aircraft acquired by Blue Defense Units
2. Red Aircraft engaged by Blue Defense Units
3. Red Aircraft damaged by Blue Defense Units
4. Blue Units damaged by Red Aircraft
5. Weapon System expenditures by Unit Type
6. Number of Red Aircraft to reach Targets
7. Number of Units created by Type

2. Preprocessor Functions And Subroutines

The following lists summarize the functions carried out by various preprocessor subroutines. Low level utilities have been deleted from these lists. For a complete calling hierarchy and detailed subroutine documentation see Appendices I and II, and the Source Code.

a. Top Level Control Routines

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MADEM</td>
<td>Main Routine</td>
</tr>
<tr>
<td>OTHERDAT</td>
<td>Constructs Simulation Data Base</td>
</tr>
<tr>
<td>LRKPRS</td>
<td>LR(K) Parser</td>
</tr>
<tr>
<td>SEMANT</td>
<td>Semantic Processing of UOIL</td>
</tr>
<tr>
<td>CONTROL</td>
<td>Event Control</td>
</tr>
<tr>
<td>SELECT</td>
<td>Selects Appropriate Event Module</td>
</tr>
<tr>
<td>PLAN</td>
<td>Red Threat Planning</td>
</tr>
<tr>
<td>HLTPNT</td>
<td>Halts Simulation</td>
</tr>
<tr>
<td>ENTSAT</td>
<td>Simulation software Use Statistics</td>
</tr>
</tbody>
</table>
b. Simulation Data Base Construction

OTHERDAT - Controls Construction of Data Base
ADDBLOK - Adds a Data Block to the Top of a Singly
           Linked List Structure
FINDBLOK - Finds a Data Block with a Specified Value in
           a Singly Linked List Structure

c. Semantic Processing Of UOIL

SEMANT - Controls Semantic Processing
CODE01 - Initialized Hex and C2 Trees
CODE03 - Constructs C2 Tree and Related Structures
SRCHPL - Searches the Players List
LOADPL - Enters a Unit onto the Players List
ABQUEVE - Sets up Airbase Data Structures
ADASASS - Constructs CRC, BOC and BTRY Structures
SCHTAB - Sets Hex Altitude From Altitude Data Base
CODE05 - Constructs Hex Tree and Related Lists
VOLLOAD - Places a unit on a Hex's Unit Occupancy List
TGTLIST - Constructs Target List
INITACQ - Initializes Acquisition Devices
CODE18 - Locates Targets in Hex Tree

d. Lexical Analyzer

LRKPRS - LR(K) Parser
ROCELL - Split Out Fields in Read State
NXTSYM - Get Next Symbol
LEXAN - Lexical Analyzer
CHRGEN - Character Generator
ADDCHR - Add Character to a String
LOOKUP - Lookup String in a Table
EXTSCN - Process Real and Integer Numbers
ERROR - Parser Error Recovery
APCEL 1 - Split out fields of apply state 1
APCEL 2 - Split Out Fields of Apply State 2
LACELL - Split Out Fields of Look-Ahead State
CARD - Read Card/Print Card
e. Red Threat Planning

PLAN - Controls Red Threat Planning
THTRPLN - Red Theater Planner
CORBOUN - Creates Corridor Boundary Structures
REVISE - Revises Red Force Allocations
ABVSCOR - Matches Airbases with Corridors
CLOSCOR - Finds Closest Corridor to an Airbase
CANDTGT - Considers Candidate Targets
JGESUIT - Determines Geographic Suitability of Targets
FORMTGT - Assigns Formations to Targets
TGTGONE - Changes Target Stands to Now-Available
AVAILBL - Changes Target Status to Available
SCHEDUL - Controls Flight Scheduling
RENDEVU - Calculates Formation Rendevous Points
ACFRAG - Sets-Up Flight Frag's
CRLFTRL - Creates Flight Data Structures
FLTGEOM - Determines Flight Mission Geometry
PLANOUT - Outputs Results of Planning Process
RLRAID - Releases Raidblok Structures
RLWAVE - Releases Wave Structures
RLTGTYP - Releases Target Type Structures
RLFMAKT - Releases formation Structures
RLCORD - Releases Corridor Structures
RLABDB - Releases Airbase Structures

f. Common Utilities

ADDBLOK - Adds a Data Block to the Top of a Linked List
CREATE - Creates Unit SB and C2 Data Structures
DROPBLOK - Drops a Block from a Singly Linked List
FINDBLOK - Finds a Block on a Singly Linked List
GETHEX - Finds or Creates a Hex Block
GIMME - Allocates Storage Space
HEXADD - Adds To Hex Numbers
HEXCHZ - Chooses Next Hex for a Flight to move to
HEXDIST - Calculates Distance Between Two Hexes
HEXINV - Calculates Inverse of a Hex Number
HEXMULT - Hex Multiplication by a Single Digit
HISTORY - Records an Event for Prost Processing
HXdGTS - Places a Hex Number in the IDIGITS Array
LINEX - Determines Intersection of Two Lines
LNPlot - Prints a Line on the Output File
MESAGE - Prints a Message on the Output File
OPTPTH - Finds Shortest Path Between Two Hexes
PAGE - Advances Page of Output
PELADD - Adds Target Blocks to the Potential Target Tree
PTREE - Extracts a Block From the Target Tree
PTRMRG - Sorts Target Tree
RELEASE - Releases Storage Space
RELIST - Subdivides Large Released Blocks
RITEI - Prints an Integer Value
RITEP - Prints a Pointer Value (octal)
RITER - Prints a Real Value
THH2PS - Translates Hex to Point Slope
THX2XY - Translates Hex to X,Y Coordinates
TLL2HX - Translates Lat-Long To Hex
TXY2HXL - Translates X,Y to Hex at Specified Level
PACK - Packs Two Halfword Fields (non-midas)
VNPACK - Unpacks Two Halfword Fields (non-midas)
NOWUCIT - Hex Peeper List Update
GETPIRS - Pointer Retrieval
KOMPARE - Compares Two Packed Words and Stores a New Value in the Specified Field.

Block Data Routines:
BDALT - Semantic Processing
BDLEX - Semantic Processing
BDLRK - Semantic Processing
3. Main Processor Functions And Subroutines

The following lists summarize the functions carried out by various main processor subroutines. For a complete calling hierarchy and detailed subroutine documentation see Appendices I and J and the Source Code. Although they are not repeated in this section, all of the Red Threat planning functions and Subroutines found in the preprocessor are also resident in the main processor.

a. Top Level Control Routines

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>MADEM</td>
<td>Main Routine</td>
</tr>
<tr>
<td>FETCH</td>
<td>Reads Hold Files</td>
</tr>
<tr>
<td>CONTROL</td>
<td>Event Control</td>
</tr>
<tr>
<td>SELECT</td>
<td>Selects Appropriate Event Module</td>
</tr>
<tr>
<td>ASSIGN</td>
<td>Controls CRC Assignment of Interceptions</td>
</tr>
<tr>
<td>ATTACK</td>
<td>Control Red Flight Ground Attacks</td>
</tr>
<tr>
<td>COMMO</td>
<td>Controls Communications</td>
</tr>
<tr>
<td>DOGFITE</td>
<td>Controls Air to Air Combat</td>
</tr>
<tr>
<td>ENGAGE</td>
<td>Controls Surface to Air Engagement</td>
</tr>
<tr>
<td>FLY</td>
<td>Controls Aircraft Movement</td>
</tr>
<tr>
<td>NAYBOR</td>
<td>Controls Determination of Unit Proximity</td>
</tr>
<tr>
<td>PERCEPT</td>
<td>Controls Perception of other Units</td>
</tr>
<tr>
<td>PLAN</td>
<td>Controls Red Threat Planning</td>
</tr>
<tr>
<td>PONDER</td>
<td>Controls Unit Information Processing</td>
</tr>
<tr>
<td>TOWER</td>
<td>Controls Airbase Operations</td>
</tr>
<tr>
<td>UMPIRE</td>
<td>Controls Simultaneous Event Processing</td>
</tr>
</tbody>
</table>

b. CRC Assignment Of Interceptions To Red Flights

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSIGN</td>
<td>Controls CRC Assignment of Red Flights</td>
</tr>
<tr>
<td>INTASIN</td>
<td>CRC Attempts Interceptor Assignment</td>
</tr>
<tr>
<td>TGTHEX</td>
<td>Calculates Interception Point for Two Flights</td>
</tr>
<tr>
<td>BTNASIN</td>
<td>CRC Attempts BOC Assignment</td>
</tr>
</tbody>
</table>
c. Red Flight Group Attack Process

ATTACK - Controls Red Flight Ground Attacks
SHRKILL - Shorad Engaged Flights
DESTROY - Actions Carried Out When Unit is Destroyed
TERMACQ - Terminates Acquisition Devices and Peeper list
KILFLIT - Actions Carried Out Flight Dies
FLTWYPE - Wipes Out Flight Action Order List
SAMWYPE - Actions Carried Out When A Sam Unit Dies
BYENOPS - Battery End Of Flight-Pass Up Procedures
SEEKP - Seek a Track in Perceptions List
CRCLOSS - CRC Actions On Loss Of Target Condition
BNLALLE - BOC Ponders Loss of All Engagement Possibilities
HANDZPT - Handles Zero Priority Tracks in Adil
BATCEAS - Battery Decisions To Cease Engagement
CANCALO - Cancel Allocations
READIL - Return Track to Adil
RELOCAT - Reposition Entry in Doubly Linked List'
SEEKTFV - Seek New Engagement For a Fire Unit
TRKCHEK - Air Defense Unit Ponders Trackability of TGT.
TOADIL - Refills Adil
ALLOFU - Allocate Fire Unit Against Red Flight
ALLOPAT - Allocate Patriot Against Red Flight
DILOUT - Eliminate Oil Entry
CNACTIK - Cancel Pending Actions for Track
CRCDIES - Actions Carried Out When CRC Dies
NUKBLND - Commo Block Out Of Units Near Nuke Blast

d. Communications

COMMO - Controls Communications Processes

e. Air To Air Combat

DOGFITE - Controls Air-To-Air Combat Processes
DESTROY - Actions Taken When Unit is Destroyed
f. **Surface To Air Engagement**
   - **ENGAGE** - Controls Surface-to-Air Engagement
   - **TRYSHOT** - Determine if a missile can be fired
   - **BATCEAS** - Battery Decision to Cease Engagement
   - **BNLALLE** - BOC Ponders Loss of all Engagement Possibilities
   - **FIRECHK** - Fire Missile Operations
   - **AMMOCHK** - Account for Missiles
   - **BYNOTRO** - Battery not ready for action
   - **DILOUT** - Eliminate Oil Entry

h. **Determination Of Unit Proximity**
   - **NAYBOR** - Controls Determination of Unit Proximity

i. **Unit Perception**
   - **PERCEPT** - Controls Unit Perception Process
   - **ABSEE** - Airbase Perception
   - **CFLYCRC** - CRC Perceptions Module Control
   - **CRCSEE** - CRC Message and Event Perception
   - **CRCEVNT** - CRC Direct Perceptions
   - **DETECT** - Detection Logic between Two Units
   - **LOSRADR** - Radar Line of Sight Determination
   - **CRCTRAK** - CRC Actions on Enemy Detection
   - **CRCKIL** - CRC Records Death Report
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BABMOVE</td>
<td>CRC Records Red Flight Movement</td>
</tr>
<tr>
<td>TGTHEX</td>
<td>Calculate Interception Point for Two Flights</td>
</tr>
<tr>
<td>NEWMOVE</td>
<td>CRC New Flight Perception Actions</td>
</tr>
<tr>
<td>FLYSEE</td>
<td>Flight Perception Module Control</td>
</tr>
<tr>
<td>ATKASES</td>
<td>Air Attack Damage Accessment on Ground Targets</td>
</tr>
<tr>
<td>CRCZINT</td>
<td>Interceptors Receive Messages from CRC</td>
</tr>
<tr>
<td>RONDSEE</td>
<td>Flight Perception of Rendezvous</td>
</tr>
<tr>
<td>GNDLOOK</td>
<td>Flight Perception and Attack of Ground Targets</td>
</tr>
<tr>
<td>SAMSEE</td>
<td>Sam Perceptions Module Control</td>
</tr>
<tr>
<td>SAMPRCM</td>
<td>Sam Perceives Aircraft Movement</td>
</tr>
<tr>
<td>BYTKCHK</td>
<td>Battery Tracking Operations</td>
</tr>
<tr>
<td>TOADIL</td>
<td>Try to Refill Adil</td>
</tr>
<tr>
<td>NEWPERC</td>
<td>Create Perceptions List Entry</td>
</tr>
<tr>
<td>BYPASUP</td>
<td>Battery Passes Track up to BOC</td>
</tr>
<tr>
<td>BYENOPS</td>
<td>Battery End of Flight-Pass Up Procedures</td>
</tr>
<tr>
<td>CRCLOSS</td>
<td>CRC Actions on Loss of Target Condition</td>
</tr>
<tr>
<td>DROPPS</td>
<td>Battery Considers Drop Possibility</td>
</tr>
<tr>
<td>SEEKENG</td>
<td>Seek New Engagement</td>
</tr>
<tr>
<td>DROPPS2</td>
<td>Battery Considers Drop Possibility 2</td>
</tr>
<tr>
<td>DLYACT</td>
<td>Add Action to Delayed Action Queue</td>
</tr>
<tr>
<td>ALLOBAT</td>
<td>Allocate BYRY against Red Flight</td>
</tr>
<tr>
<td>PRIORITY</td>
<td>Calculate Priority of Track</td>
</tr>
<tr>
<td>BNLALLE</td>
<td>BOC Ponders loss of All Engagement Possibilities</td>
</tr>
</tbody>
</table>

j. **Red Threat Planning**
   See Section (II.F.2)

k. **Unit Information Processing**
   PONDER - Controls Unit Information Processing
   TFYRCR - CRC/Flight Ponder Module Control
   CRCTHMK - CRC Decisions on Receipt of Information
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRCKIL</td>
<td>CRC Records Death Report</td>
</tr>
<tr>
<td>CRCTRAK</td>
<td>CRC Actions on Enemy Detection</td>
</tr>
<tr>
<td>CRCLOSS</td>
<td>CRC Actions on Loss of Target Condition</td>
</tr>
<tr>
<td>AB2CRC</td>
<td>CRC Interprets Message from Airbase</td>
</tr>
<tr>
<td>INT2CRC</td>
<td>CRC Interprests Message from Flight</td>
</tr>
<tr>
<td>DETECT</td>
<td>Detection Logic Between Two Units</td>
</tr>
<tr>
<td>BTN2CRL</td>
<td>CRC Interprets Message from BOC</td>
</tr>
<tr>
<td>AIRTTHNK</td>
<td>Ponder Decision to go into Air Combat</td>
</tr>
<tr>
<td>DOGTHNK</td>
<td>Ponder Next Actions After Air to Air Combat</td>
</tr>
<tr>
<td>GOTOAB</td>
<td>Actions on Return Flights to Airbase</td>
</tr>
<tr>
<td>FUELCHK</td>
<td>Check Flight Fuel Level</td>
</tr>
<tr>
<td>BOCINK</td>
<td>BOC Ponders Situation</td>
</tr>
<tr>
<td>BNPONSS</td>
<td>BOC Ponders Subordinate Status Message</td>
</tr>
<tr>
<td>BNPONBB</td>
<td>BOC Ponders BTRY Back In Action</td>
</tr>
<tr>
<td>FILERUP</td>
<td>Refuel Aircraft</td>
</tr>
<tr>
<td>INRANGE</td>
<td>Determine Time Period that Target is in Range</td>
</tr>
<tr>
<td>AZILIM</td>
<td>Impose Azimuth Limits on Engage Window</td>
</tr>
<tr>
<td>INSECT</td>
<td>Calculate Intersection with Sector Limits</td>
</tr>
<tr>
<td>SETASSN</td>
<td>Set Up for Possible ADV. Assignment</td>
</tr>
<tr>
<td>PRIORTY</td>
<td>Calculate Priority of Track</td>
</tr>
<tr>
<td>SEEKENG</td>
<td>Seek New Engagement</td>
</tr>
<tr>
<td>DLYACT</td>
<td>Add Action to Delayed Action Queue</td>
</tr>
<tr>
<td>BYUPDAT</td>
<td>Update Battery Status</td>
</tr>
<tr>
<td>BNNOTRD</td>
<td>BOC Not Ready For Action</td>
</tr>
<tr>
<td>DILOUT</td>
<td>Eliminate DIL Entry</td>
</tr>
<tr>
<td>BATTOOT</td>
<td>BTRY Decisions to Drop an Enemy Track</td>
</tr>
<tr>
<td>BNLALLE</td>
<td>BOC Ponders Loss of All Engagement Possibilities</td>
</tr>
<tr>
<td>CHKLAST</td>
<td>Determine Last Chance for Engagement and Set Priority</td>
</tr>
<tr>
<td>DROPOPS</td>
<td>BTRY Considers Drop Possibility</td>
</tr>
<tr>
<td>COVAPLY</td>
<td>Cover Threat Allocation Decisions</td>
</tr>
</tbody>
</table>
SEEKTAC - Seek to Assign Coverage
ALLOBAT - Allocate BTRY against Red Flight
READIL - Return Track to Adil
DECRALO - Decrease Allocations
BNPONEP - BOC Ponders Engagement Progress
SKSBTRK - Seek Reporting Subordinate and Track
BNCMDPR - BOC Command Decision Processes
ACCEPT - BTRY Accepts Assignments
BYALCOV - BTRYs Alter Coverage Level
CANCALO - Cancel Allocations
SEEKTFV - Seek New Engagement for a Fire Unit
PATDEC - Patriot Decrease Coverage on Track
BATTCOV - BTRY Decision to Allocate Coverage on Track
TRKCHEK - Ponder Trackability of Flight Target
ALLOFV - Allocate Fire Unit Against Red Flight
DLYACT - Add Action to Delayed Action Queue
BYHEDUP - BTRY Handle Engagement Data Update
SEEKP - Seek Track in Perceptions List
BNPONFD - BOC Ponders Flight Attrition
SOIGEST - Sam Process Information
BNNWTRK - BOC Handles Newly Visible Track
CHKCOV - Check for Coverage
BYNWTRK - BTRY Handles Newly Visible Track
PREPAFU - Prepare for Allocation of Fire Units
AUTOPRI - BTRY Decision of Target Priority
BNCONTC - BOC Considers Change in Track
BNCONHD - BOC Considers New Heading
BNRECOV - BOC Considers Reducing Coverage
BYCONTC - BTRY Considers Change in Track
BYCONLS - BTRY Considers Loss of Track
BATCEAS - BTRY Decision to Cease Engagement
BNPONDA - BOC Processes Delayed Actions
BNPONBD - BOC Ponders BTRY Death
SAMATON - BTRY Autonomous Actions
BTRYTNK - BTRY Ponders Situation
BYCMDPR - BTRY Command Decision Processes
BYPONTM - BTRY Ponders Track Movement
BYPONER - BTRY Ponders Engagement Results
PTPONER - Patriot Ponders Engagement Results
BYPONFO - BTRY Ponders Red Movement
BYPONRL - BTRY Ponders Reload
BYPONRS - BTRY Ponders Resupply
RESUPLY - BTRY Ponders Resupply of Fire Units

1. Airbase Operations
TOWER - Controls Airbase Operations
VOLLOAD - Places Unit on Hex Unit Occupancy List
INLTACQ - Initialize Acquisition Devices
REDEBREF - Process Flight After Landing
WIPEOUT - Invalidated a Node on Leftist Tree
DESTROY - Action When Unit is Destroyed
GOGETEM - Launch Interceptor Flights
CRFLTML - Create Flight Structures
PTRAND - Generate Uniform Random Hex
FLTGEOM - Calculate Flight Geometry

m. Simultaneous Event Control
UMPIRE - Controls Simultaneous Events
DESTROY - Action When Unit is Destroyed
NUKBLND - Commo Black out of Units Near NUKE Blast

Common Utilities
ADDBLOK - Adds a Data Block to the Top of Linked List
CREATE - Create Unit SB and C2 Data Structures
DELADD - Discrete Event Tree Adder
DGTSHX - Converts a Value to a Hex Number
DROPBLK - Drop a Block from a Singly Linked List
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FETCH</td>
<td>Red in Hold Files</td>
</tr>
<tr>
<td>FINDBLK</td>
<td>Find a Block in a Singly Linked List</td>
</tr>
<tr>
<td>FSDUMP</td>
<td>Print Out ISPACE Contents</td>
</tr>
<tr>
<td>FSINIT</td>
<td>Free Space Control Initialization</td>
</tr>
<tr>
<td>GETHEX</td>
<td>Find or Create a Hex Data Block</td>
</tr>
<tr>
<td>GIMME</td>
<td>Allocate Storage Space</td>
</tr>
<tr>
<td>HEXADD</td>
<td>Adds Two Hex Numbers</td>
</tr>
<tr>
<td>HEXCHZ</td>
<td>Chooses Next Hex for a Flight to move to</td>
</tr>
<tr>
<td>HEXDIST</td>
<td>Calculates Distance Between Two Hexes</td>
</tr>
<tr>
<td>HEXINV</td>
<td>Calculates inverse of a Hex Number</td>
</tr>
<tr>
<td>HEXMLT</td>
<td>Hex Multiplication</td>
</tr>
<tr>
<td>HEXMOVE</td>
<td>Hex Movement</td>
</tr>
<tr>
<td>HEXMULT</td>
<td>Hex Multiplication</td>
</tr>
<tr>
<td>HISTORY</td>
<td>Records an Event for the Postprocessor</td>
</tr>
<tr>
<td>HXDGT</td>
<td>Places a Hex Number in the IDIGTS Array</td>
</tr>
<tr>
<td>HXMLT2</td>
<td>Hex Multiplication</td>
</tr>
<tr>
<td>IJ2HX</td>
<td>Converts I,J Coordinates to Hex</td>
</tr>
<tr>
<td>JUGGLE</td>
<td>Ships Hex Number to Usable Order</td>
</tr>
<tr>
<td>UNEX</td>
<td>Determines Intersection of Two Lines</td>
</tr>
<tr>
<td>LPLOT</td>
<td>Prints a Line on the Output File</td>
</tr>
<tr>
<td>LOADPL</td>
<td>Loads the Players List</td>
</tr>
<tr>
<td>LTREE</td>
<td>Merges a Node on the Leftist Tree</td>
</tr>
<tr>
<td>LTRMRG</td>
<td>Merges a Node on the Leftist Tree</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>Prints a Message on the Output File</td>
</tr>
<tr>
<td>PACK</td>
<td>Packs Two Halfword Fields (non-midas)</td>
</tr>
<tr>
<td>PAGE</td>
<td>Adds Target Blocks to the Potential Target Tree</td>
</tr>
<tr>
<td>PELADD</td>
<td>Adds Target Blocks to the Potential Target Tree</td>
</tr>
<tr>
<td>PTREE</td>
<td>Extracts a Block from the Target Tree</td>
</tr>
<tr>
<td>PTRMRG</td>
<td>Sorts Target Tree</td>
</tr>
<tr>
<td>RITEI</td>
<td>Prints an Integer Value</td>
</tr>
</tbody>
</table>

33
RITEP - Prints a Pointer Value
RITER - Prints a Real Value
UNPACK - Unpacks Two Halfwords (non-midas)
UNLINK - Removes a Dead Unit from the C2 Tree
YANK - Removes an Entry from a Doubly Linked List
STICK - Inserts an Entry in a Doubly Linked List
NOWUCIT - Hex Peeper List Update
GETPTRS - Gets Required Pointers
MESBILD - Builds a Message Structure
UNSTAT - Unpacks an Aircraft Flight Status Board into a Common
STATPAK - Repacks an Aircraft Flight Status Board after it has been Updated
SSL - Sorts a Singly Linked List
INSERT - Inserts an Entry into an Ordered Linked List
RELOCAT - Reposition Entry in Doubly Linked List

4. Postprocessor Functions And Subroutines
The following list summaries functions carried out by various postprocessor subroutines. For a complete calling hierarchy and detailed subroutine documentation see Appendices I and J and the Source Code.

(1) RECORD - Main Control Routine
(2) INDEX - Finds Desired Index in a Storage Array
(3) MESSAGE - Prints a Message on the Output File
(4) PAGE - Advances Output Page
(5) RITEI - Prints an Integer Value
(6) RITER - Prints a Real Value
(7) TABOUT - Prints Contents of Storage Arrays
A. INTRODUCTION

MADEN uses a list processing system to store data. Information is stored in the dynamically allocated array ISPACE in a series of data structures. These data structures are composed of blocks linked in a variety of configurations (e.g., trees, lists, etc.) by pointers. Each block is made up of a specified number of words. In many cases words are divided into fields which contain discrete pieces of information. The pointers which connect blocks into data structures are actually addresses in the one dimensional array ISPACE. Since the number and configuration of blocks stored in ISPACE changes during model execution, the storage system is said to be dynamic. ISPACE is currently limited to 131,000 words of storage.

Sections B through G of this manual document the common blocks and data structures used by MADEM. Table III-1 provides a cross reference of data block types and the sections of this manual in which they are discussed. Since these blocks are defined and accessed using the Modular Information Data Access System (MIDAS), documentation of MIDAS and the tables used to define the data blocks have been included in Appendices B and D. In addition, Table III-2 provides an outline of the standard data structure documentation format used in this manual.
TABLE III-1. DATA BLOCK DOCUMENTATION CROSS REFERENCE

| Block | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
|       |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

- DATA BASE
- D1 EVENT LISTS
- D2 HEX TREE
- E1 PLAYER LISTS
- E2 GETS
- E3 PASSIVE TGT
- F1 RED CMDH
- F2 POTENTIAL TGT
- F3 ASSIGNED TGT
- F4 NONNAV TGT
- F5 CORTIHOL
- F6 AIK HEO
- F7 HED AB
- F8 RED FLT
- G1 TATF
- G2 SOC
- G3 CRC
- G4 BRED AB
- G5 BLUE FLT
- G6 BOC
- G7 BSHY
- GB PASSIVE TGT
- H STORAGE
### Table III-1. Data Block Documentation Cross Reference (Continued)

TABLE III-1. DATA BLOCK DOCUMENTATION CROSS REFERENCE (CONTINUED)

<table>
<thead>
<tr>
<th>C. DATA BASE</th>
<th>D.1 EVENT LISTS</th>
<th>D.2 HEX TREE</th>
<th>E.1 PLAYER LISTS</th>
<th>E.2 C2 TREES</th>
<th>E.3 PASSIVE TGT</th>
<th>F.1 RED CMDR</th>
<th>F.2 POTENTIAL TGT</th>
<th>F.3 ASSIGNED TGT</th>
<th>F.4 NONAVAIL TGT</th>
<th>F.5 CORRIDOR</th>
<th>F.6 ATTK REQ</th>
<th>F.7 RED AB</th>
<th>F.8 RED FLT</th>
<th>G.1 ATAF</th>
<th>G.2 SOC</th>
<th>G.3 CRC</th>
<th>G.4 BLUE AB</th>
<th>G.5 BLUE FLT</th>
<th>G.6 BOC</th>
<th>G.7 BTRY</th>
<th>G.8 PASSIVE TGT</th>
<th>H. STORAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROFILED BLOK</th>
<th>PR BUFFER</th>
<th>QUEUE</th>
<th>RAID BLOCK</th>
<th>READY QME</th>
<th>SB</th>
<th>SBR</th>
<th>SEER</th>
<th>SOURCE</th>
<th>STD BLOK</th>
<th>SUBLIST</th>
<th>SUB TYPE</th>
<th>TARGET BLOK</th>
<th>TARGET LIST BLOK</th>
<th>TTDBLOCK</th>
<th>WINGMAN</th>
<th>GARBAGE LIST BLOCKS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

38
TABLE III-2. STANDARD DATA STRUCTURE DOCUMENTATION FORMAT

* SECTION HEADING

A. DATA BLOCK INDEX
   A list of MIDAS data blocks discussed in the section.

B. DESCRIPTION
   A general description of the data structure and its purpose in MADEM. This description
   refers to the diagrams in subsections C and D. Other diagrams which show what the structure
   is attempting to represent in the real world (ex: C2 hierarchy) are also included if
   they are helpful to an understanding of the structure.

C. STRUCTURE OVERVIEW
   This subsection is intended to give the user a general view of the structure and its' component blocks.
   1) STRUCTURE DIAGRAM
      A diagram of the entire structure showing the overall configuration of data blocks.
   2) BLOCK DEFINITIONS
      Short definitions of all data blocks in the structure.

D. BLOCK SPECIFICATIONS
   This subsection is intended to give the user a detailed view of the data blocks in the
   structure and the exact definition of variables in each field.
   1) BLOCK DIAGRAMS
      Detailed diagrams of all data blocks including field names.
   2) FIELD DEFINITIONS
      Precise definitions for all fields in all blocks. These definitions are sub-divided by block and contain all relevant values.

E. LINKAGES TO OTHER DATA STRUCTURES
   Linkages to other data structures are noted along with the purpose of the linkages.

F. NOTES
   Any notes or questions relating to the data structure and its' contents.
B. COMMON BLOCKS

(1) COMMON/AAPK/ - air-to-Air PK Common
  AAPK - antiaircraft probability of kill

(2) COMMON/ACFRAG/ - AIRCRAFT FLIGHT Plan Common
  NSECTOR - Corridor Zone Number
  PTGTHEX -
  PADRCG -
  PHEXENT -
  TATREND -
  PHEXEXT -

(3) COMMON/AFM/ - Debugging common for ?TRACE? JINX -

(4) COMMON/AGPD/ - Air to Ground Probability of Detection AGPD -

(5) COMMON/AGPK/ - Air to Ground Probability of kill AGPK -

(6) COMMON/CLOOK/ - Common for dumping COMMONS
  ISTABL -
  NRSRV -
  ISTAB -

(7) COMMON/COMOUT/ -
  LBL -
  LBLCT -

(8) COMMON/CMPTR/ -
  PHXTOP - pointer to top of address tree
  PTRBLUED - pointer to top of blue C2 tree
  PTRRED - pointer to top of red C2 tree
  PBLTGT - pointer to top of list of red targets
  PTRC2 - pointer to active C2 tree
  PBLUPL - pointer to blue players list
  PREDPL - pointer to red players list
  PTRPL - pointer to active players list
  PLASTC - C2 pointer of last mentioned commander
  PLASTS - C2 pointer of last mentioned subordinate
  LSIDE - Active side (1 = Blue, 2 = red)
LHEX - last hex mentioned
PTRCDB - last mentioned corridor description pointer
PTRWDB - last mentioned wave description pointer
PTRRDB - last mentioned raid description pointer
PTRTTDB - last mentioned target type description pointer
BUFZN - buffer zone width value
IGMSRT - game hour start time, default = 1
IDYSRT - game day start time, default = 1
PTRDATA - pointer to data base
PTOPORD - pointer to red orders
PREDSEE - pointer to red commanders perception list
ISIZE - size of player list array, equal to maximum number of players on largest side

(9) COMMON/COMSCS/ -
LMASK - bit mask for left hand side of a ward
RMASK - bit mask for right hand side of a ward
INITCEL - initialize the cell (currently not used)
TREETOP - top of the discrete event list
ENDTIME - end of the game
INCTIME - increment of time for interim output
GTIME - game of current discrete event
CELMINT - increment time for cell event
LPTR - array of pointers to cell events

(10) COMMON/CRCCOM/ -
IEVENT - CRC's current event
PSB - pointer to the scoreboard
PADR - pointer to hex address
ITYPE - unit type
ISIDE - multiple definitions

(11) COMMON/CSTK/ - Language Processor
IPSTK - stack used by language processor
L -
LLMAX -
NSTK -
(12) COMMON/DATA/-
   PTRDAT -

(13) COMMON/DEBUG/-
   DBGA -
   DBGB -
   DBG -
   DBGD -

(14) COMMON/DFLAGS/-
   IDEBUG - a flag which turns on ENTRYP and EXITP. If equal to 2HON turns on entry and exit messages for all routines
   IDUMP - ISPACE dumping option flag, set as a parameter in MADEM if set equal to 2HON, ISPACE will be dumped
   IDATFLG - DATFILE display flag, set as input parameter in MADEM ZHON - will display DATFILE data structure only in INITBIN
   ISTOP - input parameter set in MADEM - if equal to 4HODAT stops after UOIL (can be used to get ISPACE dump before players list is released) - if equal 3HDEL stops INITBIN after DELADD and before CONTROL

(15) COMMON/XTRACE/-
   ITRPRTR - Pointer to circular list
   TRACIR - circular list of last 50 routine entries
   TRAPTR - pointer to routine pushdown stack
   TRAPDS - push down stack of routine entries
   ICOUNT - vector of routine entry counts

(16) COMMON/STEXT/-
   SEGTXT - vector of routine names in hollerith

42
(17) COMMON/FS/  
   PTRFS - pointer to first ward of free space  
   PTRRL - pointer to recovered storage list  
   MXSPCE - maximum length of ISPACE, set in MADEM  
   ID - last Assigned Player ID Number  
(18) COMMON/HALT/  
   IFSTOP - a flag, if equal to 1, MADEM stops in control  
   CPULIM - maximum number of CPU second set in MADEM  
   CELTIME -  
   LEVEL -  
(19) COMMON/INITPTR/  
   All major values from Player SB.SDB  
   PTRADR - PTR to current address  
   PTPRC2 - PTR to C2 node  
   PTRSDB - pointer to status display board  
   PTRFEL - pointer to future event list  
   PTRACQ - pointer to ACQDEVICE block  
   PTRCOM - pointer to command block  
   PTRSTAT - Pointer to a unit status block  
   PTRSEE - pointer to perceptions list  
   PTRSUB - pointer to a subordinate block  
   PTRORD - pointer to orders block  
   MYTYPE - Player unit type  
   MYSIDE - player side  
(20) COMMON/IODEV/  
   IN - device number of input file  
   NUOIL - device number of UOIL file  
   ND - device number of DATFILE  
(21) COMMON/JJOPT/  
   IOP - start or restart option flag  
   - if equal to 1 start at beginning (INITBIN)  
   - if equal to 2 restart (RUNBIN)
(22) COMMON/LIMITS/ - Max & Min limits on ISPACE
   LOWER - Extend core location of ISPACE (1)
   LUPPER - extended core location of last work of ISPACE
(23) COMMON/MASK/ -
   IL - masking constant
   ILM - masking constant
       = 77777777770000000000B
       = 10000000000B
(24) COMMON/MODVAR/ -
   INCDNT - incident code (event code)
   NEHMEN - receiver of the event
   LASSEN - schedules of an event
   TIME - game time of the scheduled event
   MSG - Pointer to the message block
   PTRGOD - PTR to a god like creature whose omnipotence
             awes all who meet
   RSEED - random seed
(25) COMMON/MXMIS/ -
    MXSUP -
(26) COMMON/PATH/ -
    LASTP -
    LREP -
    LASTR -
(27) COMMON/SAMPK/ -
    SAMPKA - SAM probability of kill
    SAMPKB - SAM probability of kill
(28) COMMON/SAMPTRS/-
    PMYDATA - pointer to data base associated with considered
               unit
    PDIL - pointer to element in digested information list
           which corresponds to a particular considered
           flight
PPINFOR - pointer to considered perception list entry for battery or BOC
POINFO - pointer to considered digested information block for battery or BOC
POINFO - pointer to old digested information block for battery or BOC
PBAT - pointer to SUBLIST block representing a particular subordinate BOC
PPAL - pointer to PAL entry for track
PFU - pointer to fire unit block

(29) COMMON/SEMINFO/-
IVALUE - an array which holds real number from the SEMANT sentences
IFLAG -

(30) COMMON/SPACE/-
BLANK - required!
ISPACE -

(31) COMMON/SPSTAT/-
ICIGIM - counts the number of blocks by block size that GIMME allocates
ICTREL - counts the number of blocks by block size that are released by RELEASE

(32) COMMON/STATBD/- AIRCRAFT STATUS Board
PFLTYP - pointer to Flight DATA BASE
PTRMUN - pointer to MUNITIONS list
PTRSTRT - pointer to start hex of leg
PTREND - pointer to end hex of leg
PTRNXT - pointer to next hex in flight path
PTRAB - pointer to SB of home j
NUMTGT - number of aircraft in air-to-air track
PTGTSB - pointer to air-to-air target SB
PGNDTGT - pointer to air-to-ground target SB
NUMAC - number of aircraft in flight
LEGSTA - type of leg currently being blown
INSTA - status of interceptor
ITALTCNG - climbing, diving level
NDXPROF - index into flight profile list
LNDSTA - landing status
IORBSTA - orbit status
IAIRCOM - air-to-air combat status
IGNDATK - air-to-ground combat status
JAMSTA - jammer status
FUEL - current fuel in of hexes
ALTUDE - current altitude
SPEED - current speed
DIRECT - current heading

(33) COMMON/THTRPLN/-
PFORSET - pointer to target list
PNOAVAL - pointer to non available target list

(34) COMMON/TRACK/-
PFADR -
ISID -

(35) COMMON/TYPES/-
IARRAY -

(36) COMMON/OUTDEV/-
OUTDEV - output device for data structure displays
C. DATA BASE STRUCTURE - DATFILE

1. Data Block Index
   - ACDB
   - ABQUEDB
   - ACRFTLIST
   - ACRFTONAB
   - ADSITEDB
   - AQDB
   - DATBLOK
   - DATBUF
   - FDBDBLOK
   - FLTDB
   - FMFLTDB
   - PAYBUF
   - PAYLOAD
   - PAYLDDBLOK
   - PROFILEDBLOK

2. Description
   The DATFILE data structure, built by subroutine OTHRDAT, contains only data from the MADEM input file 'DATFILE'. The DATFILE data is divided into 13 classes. Nine of these classes (classes 6026, 6001-6008) are stored in the DATFILE data structure. The other four classes are in common blocks.

   The DATFILE data structure is a matrix of linked lists that includes 15 distinct data blocks. Fourteen of those data blocks point to themselves and thus are singly linked lists. The only block that is not part of a linked list is the one word buffer (DATBUF) at the top of the DATFILE structure.

   The main linked list in DATFILE consists of nine occurrences of DATBLOK. Each occurrence is a two-word buffer block for one of the nine classes of DATFILE data that are stored in DATFILE. Each two-word buffer points to a data block specific to that class (6000 SERIES). Some of those data blocks point to even more linked lists. See the diagrams for a better picture of DATFILE.
3. Structure Overview
   a. Structure Diagram (See Figure III-1)
   b. Block Definitions

<table>
<thead>
<tr>
<th>Block</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATBUF</td>
<td>Data Base Buffer Block. Top of entire (DAFILE) structure. Pointed to by PTRDATA in common COMPTR.</td>
</tr>
<tr>
<td>DATBLOK</td>
<td>Data Class Block. Top of DATBLOK list pointed to by DATBUE block. One DATBLOK block for each data class (6000 SERIES).</td>
</tr>
<tr>
<td>ADSITEDB</td>
<td>Air Defense Site Data Base Block. Used to describe BOC and Battery characteristics. One block for each type of BOC and Battery (e.g., HAWK, HERC, PATRIOT).</td>
</tr>
<tr>
<td>EDBDBLOK</td>
<td>Formation Data Base Block. Description. One block for each formation type.</td>
</tr>
<tr>
<td>FMFLTDB</td>
<td>Formations Flight Data Base Block. Describes flights in the formation referred to in the EDBDBLOK blocks.</td>
</tr>
<tr>
<td>FLTDB</td>
<td>Flight Data Base Block. Contains basic flight characteristics. One block for each flight.</td>
</tr>
<tr>
<td>PAYLOAD</td>
<td>Payload Data Base Block. Contains basic information on payload capacity and range.</td>
</tr>
<tr>
<td>ACDR</td>
<td>Aircraft Data Base Block. Contains basic aircraft characteristics. One block for each aircraft type, Red and Blue.</td>
</tr>
<tr>
<td>PAYBUF</td>
<td>Payload Type Buffer Block. Buffer block for PAYLDBLOK blocks. One block for each payload type.</td>
</tr>
<tr>
<td>PAYLDBLOK</td>
<td>Payload Identification Block. Contains specific payload/weapon code. Represents a particular weapon of a given type.</td>
</tr>
</tbody>
</table>
Figure III-1. Data Base Structure Diagram
PROFILEDBLOK  Profile Specification Block. Describes basic mission profile in terms of altitude on various legs of the mission.

AQDB  Acquisition Data Base Block. Specifies acquisition range for a given device.

ABQUEDB  Air Base Queue Data Base Block. Service queue information for air bases. One for each air base.

ACRFTONAB  Initial Aircraft on Air Base Block. Specifies number of aircraft types on an air base. One for each air base, Red and Blue.

ACRFTLIST  Aircraft List Block. One for each aircraft type. Pointed to by ACRFTONAB. Contains aircraft type code.

4. Block Specifications
   a. Block Diagrams
      1) DATBUF
         PDATBLK NUMBLOK
      2) DATBLOK
         PNEXT CLASS

50
3) ADSITEDB

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>ADTYPE</td>
</tr>
<tr>
<td>MODVAL1</td>
<td>MAXNUMDIGEST</td>
</tr>
<tr>
<td>MAXTIMEDIGEST</td>
<td>MINTIMEDIGEST</td>
</tr>
<tr>
<td>LOSTIME</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>LASTCHANCE</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>ENGAGEWINDOW</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>MODVAL2</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>MODVAL3</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>COVONONE</td>
<td>ONE</td>
</tr>
<tr>
<td>COVONFEW</td>
<td>FEW</td>
</tr>
<tr>
<td>COVONMANY</td>
<td>MANY</td>
</tr>
<tr>
<td>TIMEFLIGHT</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>MISSILERANGE</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>MAXASSIGN</td>
<td>MODVAL4</td>
</tr>
<tr>
<td>MODVAL5</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>MAXTRACTRANGE</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>LOCKONTIME</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>MODVAL6</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>MODVAL7</td>
<td>(SPACE)</td>
</tr>
<tr>
<td>CONVLOAD</td>
<td></td>
</tr>
<tr>
<td>SNUKELOAD</td>
<td>LNUDELOAD</td>
</tr>
<tr>
<td>RESUPPLYCV</td>
<td>CVRESUPPLYFREQ</td>
</tr>
<tr>
<td>RESUPPLYSN</td>
<td>SNRESUPPLYFREQ</td>
</tr>
<tr>
<td>RESUPPLYSN</td>
<td>LNRESUPPLYFREQ</td>
</tr>
</tbody>
</table>

4) FDBDBLOK

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>NRFORM</td>
</tr>
<tr>
<td>PTRFLT</td>
<td>NOFLTL=3</td>
</tr>
<tr>
<td>SPFORMC</td>
<td></td>
</tr>
</tbody>
</table>

5) FMFLTDB

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>PNXFLDB</td>
</tr>
</tbody>
</table>

51
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6)</td>
<td>FLTDB</td>
</tr>
<tr>
<td>PNXFLDB</td>
<td>NRFLITE</td>
</tr>
<tr>
<td>PTYPLDS</td>
<td>NOPYLDS</td>
</tr>
<tr>
<td>PTYAQDB</td>
<td>PTACDB</td>
</tr>
<tr>
<td>MAXNOAC</td>
<td>MINNOAC</td>
</tr>
<tr>
<td>MULTAC</td>
<td>PROFILE</td>
</tr>
<tr>
<td>SPFLTC(SPACE)</td>
<td></td>
</tr>
<tr>
<td>DISTSEP(SPACE)</td>
<td></td>
</tr>
<tr>
<td>7)</td>
<td>PAYLOAD</td>
</tr>
<tr>
<td>PNXYPD</td>
<td>NRPDCLS</td>
</tr>
<tr>
<td>MAXAMT</td>
<td>MINAMT</td>
</tr>
<tr>
<td>MAXFIRERANGE</td>
<td>PAYLDDB</td>
</tr>
<tr>
<td>8)</td>
<td>ACDB</td>
</tr>
<tr>
<td>NEXT</td>
<td>NRCTYPE</td>
</tr>
<tr>
<td>MAXSPEED(SPACE)</td>
<td></td>
</tr>
<tr>
<td>CRUISESPEED(SPACE)</td>
<td></td>
</tr>
<tr>
<td>MAXALTITUDE(SPACE)</td>
<td></td>
</tr>
<tr>
<td>MINALTITUDE(SPACE)</td>
<td></td>
</tr>
<tr>
<td>MAXCLIMB(SPACE)</td>
<td></td>
</tr>
<tr>
<td>FUELCONSUME(SPACE)</td>
<td></td>
</tr>
<tr>
<td>ACQRANGE(SPACE)</td>
<td></td>
</tr>
<tr>
<td>RADARCS(SPACE)</td>
<td></td>
</tr>
<tr>
<td>ATTACKRADIUS(SPACE)</td>
<td></td>
</tr>
<tr>
<td>MAXFUEL(SPACE)</td>
<td></td>
</tr>
<tr>
<td>9)</td>
<td>PAYBUF</td>
</tr>
<tr>
<td>PNEXT</td>
<td>NRPDCLS</td>
</tr>
<tr>
<td>PAYLDDB</td>
<td>NUMBLOK</td>
</tr>
<tr>
<td>10)</td>
<td>PAYLDBLOK</td>
</tr>
<tr>
<td>NEXT</td>
<td>TYPEINDEX</td>
</tr>
<tr>
<td>11)</td>
<td>PROFILEBLOK</td>
</tr>
<tr>
<td>PNXPRDB</td>
<td>NRPROFL</td>
</tr>
<tr>
<td>ALTCREW(SPACE)</td>
<td></td>
</tr>
<tr>
<td>ALTOTTG(SPACE)</td>
<td></td>
</tr>
<tr>
<td>ALTDB(SPACE)</td>
<td></td>
</tr>
</tbody>
</table>
12) AQDB

<table>
<thead>
<tr>
<th>NEXT</th>
<th>NRAQTYP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANGE(SPACE)</td>
<td></td>
</tr>
<tr>
<td>NOUSE1</td>
<td></td>
</tr>
<tr>
<td>NOUSE2</td>
<td></td>
</tr>
</tbody>
</table>

13) ABQUEDB

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE1(SPACE)</td>
<td></td>
</tr>
<tr>
<td>VALUE2(SPACE)</td>
<td></td>
</tr>
<tr>
<td>VALUE3(SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

14) ACRFTONAB

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>ABID</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACRFTLIST</td>
<td>NUMBLOKS</td>
</tr>
</tbody>
</table>

15) ACRFTLIST

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>ACRFTID</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMACRF</td>
<td>FORMTYPE</td>
</tr>
</tbody>
</table>

b. Field Definitions

1) DATBUF Block

- **PDATBLK** Pointer to first DATBLOK block.
- **NUMBLOK** Number of DATBLOK blocks in list.

2) DATBLOK Block

- **PNEXT** Pointer to next DATBLOK block.
- **CLASS** DATFILE class (6000 SERIES).
  - 6026 = Air defense site data
  - 6001 = Formation specifications
  - 6002 = Flight specifications
  - 6003 = Aircraft specifications
  - 6004 = Payload specifications
  - 6005 = Flight profile specifications
  - 6006 = Acquisition device data
  - 6007 = Air base queue data
  - 6008 = Initial aircraft types on air bases
PCLASS Pointer to first data block for the class. The actual data block varies from class to class.

NUMBLOK Number of data blocks in list pointed to by PCLA.

3) ADSITEDB Block
PNEXT: Pointer to next ADSITEDB block.
ADTYPE: Unit type of this unit, must be a BOC or BTRY.
MODVAL1: Model Value = 1
MAXNUMDIGEST: Maximum number of flights on which a BOC or BTRY can be digesting info at one time.
MAXTIME DIGEST: Maximum time (in seconds) between consecutive digests of info (BOC & BTRY).
MINTIME DIGEST: Minimum time (in seconds) between consecutive digests of info (BOC & BTRY).
LOSTTIME: Time (in seconds) after which a track not seen is assumed permanently lost (BOC & BTRY).
LASTCHANCE: Time (in seconds) considered short for a subordinate to respond to a target. (Time from now until his last chance to shoot.) BOC only, for BTRY = 0.
ENGAGEWINDOW: Minimum length of subordinates engagement window for a significant engagement opportunity in seconds (BOC & BTRY).
MODVAL2: Model value = 0
MODVAL3: Model value = 0
COVONONE: Desired number of fire units coverage for one aircraft (BOC & BTRY).
ONE: Model value = 1

COVONFEW: Desired number of fire units coverage for few aircraft (BOC & BTRY).

FEW: Model value = 5, number of aircraft considered "few."

COVONMANY: Desired number of fire units coverage for many aircraft (BOC & BTRY).

MANY: Model value = 1000000.

TIMEFLIGHT: Maximum time (in seconds of flight for missile (BOC & BTRY).

MISSILERANGE: Maximum range for missiles in meters (BOC & BTRY).

MAXASSIGN: Maximum number of targets per ready fire unit to be assigned at one time. BOC only, BTRY = 0.

MODVAL4: Model value; for BOC = 8, for BTRY = 11.

MODVAL5: Model value 0.

MAXTRACKRANGE: Maximum tracking range in meters. BTRY only, BOC = 0.

LOCK ON TIME: Assumed time (in seconds) for BTRY to achieve lockon. BTRY only, BOC = 0.

MODVAL6: Model value = 0.

MODVAL7: Model value = 0.

CONVLOAD: Number of Conventional missiles.

SNUKELOAD: Number of large nukes.

LNUKELOAD: Number of large nukes.

RESUPPLYCV: Number of missiles per resupply of ammo. BTRY only, BOC = 0.

CVRESUPPLYFREQ: Time (in seconds) between resupply of conventional ammo. BTRY only, BOC = 0.
RESUPPLYSN: Number of missiles per resupply of small nukes. BTRY only, BOC = 0.
RESUPPLYLN: Number of missiles per resupply of large nukes. (BTRY only, BOC = 0.)
LNRESUPPLYFREQ: Time (in seconds) between resupply of large nukes. BTRY only, BOC = 0.

4). ADSITEBDB Block
PNEXT: Pointer to next FDBDBLOK.
NRFORM: Formation number, must be unique.
PTRFLT: Pointer to formations flight block (FMFLTDB).
NOFLTL: Number of flights in the formation.
SPFORMC: Formation cruise speed in meters/seconds.

5) FMFLTDB Block
PNEXT: Pointer to next formation flight block.
PNXFTDB: Pointer to flight data block (FLTDB).

6) FLTDB Block
PNXFLDB: Pointer to next FLTDB Clock.
NRFLITE: Unique flight specification number.
PTYPLDS: Pointer to payload data block (Payload Class 6002).
NOPYLDS: Number of payload data blocks.
PTYAQDB: Pointer to acquisition data block (AQDB, Class 6007).
PTACDB: Pointer to Aircraft Specification data block (ACDB, Class 6003).
MAXNOAC: Maximum number of aircraft in flight.
MINNOAC: Minimum number of aircraft in flight.
MULTAC: Multiples of aircraft required for flight.
PROFILE: Pointer to profile specification data block (PROFILEDBLOK, Class 6005).
SPFLTC: Flight cruising speed in meters/seconds.
DISTSEP: Flight separation distance in meters.

7) PAYLOAD Block
PNXTYPD: Pointer to next payload block.
NRPDCLS: Payload type, must be 3 or 4:
3 = air to ground
4 = air to air
MAXAMT: Maximum number of loads of this payload.
MINAMT: Minimum number of loads of this payload.
MAXFIRERANGE: Future use by an enhancement for maximum fire range for engagements greater than one hex.
PAYLDB: Pointer to payload ID DATA BLOCK (PAYLDBLOK, Class 6004).

8) ACDB Block
NEXT: Pointer to next ACDB block.
NRACTYPE: Aircraft type number.
MAXSPEED: Maximum speed in meters/seconds.
CRUISESPEED: Cruising speed in meters/seconds.
MAXALTITUDE: Maximum altitude in meters.
MINALTITUDE: Minimum altitude in meters.
MAXCLIMBDIVE: Maximum climb/dive rate in meters/seconds.
FUELCONSUME: Fuel consumption rate in hexes/seconds.
ACQRANGE: Acquisition range in meters.
RDRARCS: Radar cross section in hexes.
ATTACKRADIUS: Attack radius in meters.
MAXFUEL: Maximum fuel load in hexes.

9) PAYBUF Block
PNEXT: Pointer to next PAYBUF.
NRPDCLS: Type of payload, must be 3 or 4:
3 = Air to ground 4 = Air to air

PAYLDDDB: Pointer to ID Blocks (PAYLDDDBLOK,
Class 6004) for this payload type.
Number of ID Blocks for this payload type.

10) PAYLDDDBLOK Block
NEXT: Pointer to next ID block.
TYPEINDEX: Payload ID, unique within each payload type.

11) PROFILEDBLOK Block
PNXPRDB: Pointer to next PROFILEDBLOK.
NRPROFL: Profile Identification number, must be unique within the 6005 Class.
ALTCREN: Altitude of first leg in meters.
ALTOTGT: Altitude of second leg in meters.
ALTOAB: Altitude of third leg in meters.

12) AQDB Block
NEXT: Pointer to next AQDB block.
NRAQTYP: Unit type.
RANGE: Acquisition range in meters.
NOUSE1: Not used.
NOUSE2: Not used.

13) ABQUEDB Block
PNEXT: Pointer to next ABQUEDB block.
CLASS: Queue Class, currently model value = 1.
VALUE1: Model value = 0.005.
VALUE2: Model value = 0.
VALUE3: Model value = 0.2.

14) ACRFTONAB Block
PNEXT: Pointer to next ACRFTONAB block.
ABID: Air base ID.
ACRFTLIST: Pointer to aircraft list (ACRFLIST) for this air base.
NUMBLOCKS: Number of aircraft types on this air base. Limit 1 for Blue.

15) ACRFTLIST Block
PNEXT: Pointer to next ACRFTLIST block.
ACRFTID: Type of aircraft (400 Series).
NUMACRFT: Number of aircraft.
FORMTYPE: Formation type number.
* Required for Blue air bases.
* Equals zero for Red air bases.

5. Linkages to Other Data Structures
The DATFILE data structure is self-contained in the sense that none of its member data blocks point to blocks outside the DATFILE structure. The reason for this is that the DATFILE data structure is merely a representation of the input file DATFILE, and thus needs no external pointers. Other data structures and blocks, however, point to DATFILE as follows:

<table>
<thead>
<tr>
<th>Block</th>
<th>Points To</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMATIONBLOK</td>
<td>FDBDBLOK</td>
<td>To identify each Red formation.</td>
</tr>
<tr>
<td>FLTAKTBLOK</td>
<td>FLTDB</td>
<td>Once for each Red flight attacking a target.</td>
</tr>
<tr>
<td>ACRFTSTATUS</td>
<td>FLTDB</td>
<td>Once for each Red aircraft to identify its flight.</td>
</tr>
<tr>
<td>LOAD</td>
<td>PAYLOAD</td>
<td>To identify an aircraft's payload.</td>
</tr>
<tr>
<td>ABINFO</td>
<td>ACDB</td>
<td>Once for each aircraft type. So that an air base has access to aircraft specs.</td>
</tr>
<tr>
<td>QUEUES</td>
<td>ABQUEDB</td>
<td>Air base queue data block.</td>
</tr>
</tbody>
</table>

59
The top of the DATFILE structure (DATBUF) is pointed by by PTRDATA, a variable in common block COMPTR.

6. Notes

Three of DATFILE's blocks--DATBUF, DATBLOK, and PAYBUF--were not in the pre-1979 "full" MIDAS table. As of August 20, 1979, however, plans were to include these blocks in the MIDAS table.

The third word of block PAYLOAD is defined differently in the MIDAS table than it was used in the non-MIDASed OTIRDAT routine. It was a full word pointer to PAYLDBLOK but has been redefined so that the 2nd half word is that pointer and the first half word is the maximum firing range for PAYLOAD. This was done to allow for a future enhancement.
D. SIMULATION CONTROL STRUCTURES

The simulation control structures are used to control event processing and movement within the hexagonal coordinate system. Event processing structures include the discrete event list and future event lists. Hexagonal coordinate system related structures include the hex address tree, unit occupancy lists and peeper lists. These control structures are explained in detail in the following subsections.
1. **Event Lists**
   a. **Data Block Index**
      
      EVENT
      LEFTREE
      MESSAGE
   
   b. **Description**
      
      Event scheduling in MADEM is carried out using the three data block types listed above. These block types are used to build two event control data structures DEL and FEL.

      The DISCRETE EVENT LIST (DEL) is the most complex of these structures. It uses all three block types to build a leftist tree of the form shown in the structure diagram under the DEL label. The DEL structure is used to track all scheduled events with the next event to occur placed at the top of the tree. The leftist tree form is used to speed sorting of the large number of events scheduled by the model. For a complete explanation of leftist tree sorting algorithms see Appendix E. It should be noted that the DEL is actually a threaded tree in that the event blocks have pointer fields which allow them to be independently linked into doubly linked lists. This capability is shown in the structure diagram by the dotted lines.

      The future event list (FEL) is a doubly linked list formed within the DEL structure using the threading capability described above. The FEL is used to keep track of events scheduled by a particular player. This allows future events (as described by EVENT, LEFTREE, AND MESSAGE blocks) scheduled by a player to be deleted from the DEL with minimal effort if the player is destroyed before these events can occur. An example of a FEL is shown in the structure diagram.

c. **Structure Overview**

   1) **Structure Diagram** (Figure III-2)

   2) **Block Definitions**

      **LEFTREE** - LEFTIST EVENT TREE NODE. Contains time of event used for sorting and pointers which connect the node to the rest of the tree.
EVENT BLOCK. Contains a description of the event, its time of occurrence, pointers to the SB (Scoreboards) of the perpetrator and victim of an event. MESSAGE. Contains message codes associated with an event.

d. Block Specifications
1. Block Diagrams

a. EVENT

<table>
<thead>
<tr>
<th>NEHMEN</th>
<th>INCDNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTRUP</td>
<td>PTRDOWN</td>
</tr>
<tr>
<td>MSG</td>
<td>LASSEN TIME (SPACE)</td>
</tr>
</tbody>
</table>

b) LEFTREE

<table>
<thead>
<tr>
<th>TIME (SPACE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREVENT</td>
</tr>
<tr>
<td>DIST</td>
</tr>
<tr>
<td>PLEFT</td>
</tr>
<tr>
<td>PRITE</td>
</tr>
</tbody>
</table>

c) MESSAGE

<table>
<thead>
<tr>
<th>PTR</th>
<th>FREQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE 1</td>
<td>VALUE 2</td>
</tr>
<tr>
<td>+ VALUE 3</td>
<td>+ VALUE 4</td>
</tr>
</tbody>
</table>

Note: + indicates alternate field definition for preceding word.

2. Field Definitions

a) EVENT BLOCK

- NEHMEN: pointer to the player to be processed in the event. (Points to the player's scoreboard - SB)
- INCONT: Event code: (See Appendix F.)
- PTRUP: FEL pointer to preceding event block for the player scheduling the event.
- PTRDOWN: FEL pointer to following event block for the player scheduling the event.
- MSG: pointer to the MESSAGE block. Used only when a message event has been scheduled.
- TIME: Time of the event. Stored as a real variable.
b) **LEFTREE Node**

- **TIME** - Time of the event. Stored as a real variable.
- **PEVENT** - pointer to the EVENT block.
- **DIST** - Minimum distance (in nodes) from a node (LEFTREE) to a leaf of the leftist tree. (Note: leaves are vacuous)
- **PLEFT** - Pointer to LEFTREE node on the right of the DEL tree.

c) **MESSAGE Block**

- **PTR** -
- **FREQ** -
- **VALUE 1** -
- **VALUE 2** -
- **VALUE 3** -

e. **Linkages to other Data Structures**

The event blocks are linked to the unit scoreboards (SB) of both the player initiating the event and the player to be processed by the event. The DEL and FEL are also pointed to by the pointers TREETOR and PTRFEL, respectively. In addition, the EVENT blocks are pointed to be the following data blocks:

- **BOCSTAT**
- **DIL**
- **PAL**
- **BTRYDIL**
- **ENGAGE**
- **SB**
- **BTRystack**
- **FIREUNIT**

f. **Notes**

The MADEM event code definitions listed in Appendix F indicate the code number of each MADEM event and the subroutine in which event processing occurs.
2. Hex Address Tree and Related Lists
   
a. Data Block Index
   
   BUFFER
   HEX
   HEXELEV
   LINK

b. Description
   Terrain information and the position of units in the hexagonal coordinate system is stored in the HEX ADDRESS TREE and its associated PEEPER and UNIT OCCUPANCY lists.

   The HEX ADDRESS TREE is composed of HEX blocks arranged in a hierarchical tree structure. Each of the levels in the tree corresponds to a level in the hexagonal coordinate system. Table III-3 indicates the levels at which various types of information are stored in the HEX blocks terrain field as well as the levels at which PEPPER and UNIT OCCUPANCY lists are maintained.

   UNIT OCCUPANCY and PEEPERS lists may be attached to HEX blocks at various levels in the HEX ADDRESS TREE structure. These linked lists are composed of a BUFFER block and a chain of LINK blocks.

   The UNIT OCCUPANCY LIST (UOL) contains pointers to the scoreboards (SB) of units which occupy the HEX at the specified level.

   The PEEPER LIST (PL) contains pointers to the scoreboards (SB) of units which can SEE INTO the hex.

1) Structure Diagram (Figure III-3)
2) Block Definitions

   HEX - HEX BLOCK. Contains HEX number, tree pointers, unit occupancy list and peeper list pointers and terrain pointer.

   BUFFER - BUFFER BLOCK. Used by PEEPER and UNIT OCCUPANCY lists. Contains pointers to scoreboards (SB).
Figure III-3. Hex Tree Structure Diagram
<table>
<thead>
<tr>
<th>HEX LEVEL</th>
<th>NO. OF HEX DIGITS</th>
<th>HEX DIAMETER (KM)</th>
<th>HEX AREA $a^2$</th>
<th>STORAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>0</td>
<td>8575.</td>
<td>63,700,000</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>3240.</td>
<td>9,100,000</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>1225.</td>
<td>1,300,000</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>463.</td>
<td>185,600</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>175.</td>
<td>26,500</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>66.1</td>
<td>3,790</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>25.</td>
<td>541</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>9.45</td>
<td>77</td>
<td>3</td>
</tr>
</tbody>
</table>

1. Pointer to the PEEPER list is active at this level.
2. Pointer to elevation storage block active at this level.
   * Pointer to PEPPER list active
   * Pointer to Unit occupancy list active
3. Pointer to Unit occupancy list active.
d. Block Specifications

1). Block Diagrams

a) HEX

<table>
<thead>
<tr>
<th>HEXNUMBER</th>
<th>LEVEL</th>
<th>PUP</th>
<th>PDOWN</th>
<th>TERRAIN</th>
<th>PNEXT</th>
<th>PUOL</th>
<th>PEEPER</th>
</tr>
</thead>
</table>

b) HEXELEV

| ELEVAT | (SPACE) |

| NUMLINK | |

c) BUFFER

| PSTART | NUMLINK |

d) LINK

| PNEXT | PSB |

2) Field Definitions

a) HEX Block

- HEXNUMBER - HEX NUMBER (up to 7 Octal Digits)
- LEVEL - HEX LEVEL (13 - 6 used in MADEM)
- PUP - Pointer to parent. HEX block
- PDOWN - Pointer to daughter HEX block
- TERRAIN - At level 7 points to HEXELEV which contains altitude of the hex in meters
- PUOL - Pointer to unit occupancy list buffer block at Level 6
- PEEPER - Pointer to PEEPER LIST buffer block at Level 10.

b) HEXELEV Block

- ELEVAT - Elevation in meters. Stored as a real variable.
c) **Buffer Block**
   - PSTART: Pointer to first list block
   - NUMLINK: Number of links in list block chain.

d) **Link Block**
   - PNEXT: Pointer to next link block
   - PSB: Pointer to unit scoreboard

e. **Linkages to other Data Structures**
   Link blocks point to the unit scoreboards (SB) of units which occupy a hex in the UOL list and units which can be seen from a hex in the PEEPER LIST.

f. **Notes**
   - It appears that in the unMIDASized version of the model the BUFFER BLOCK is not used. However, it was included in this documentation because it exists in the MIDAS tables.
   - UOL and PL's may exist at all levels in the hex tree.
E. COMMAND/CONTROL STRUCTURES

The command/control structures are used to simulate the command and control functions associated with the NATO and PACT command hierarchies. The three structures in this category include the PLAYER's LISTS, C2 TREES, and the PASSIVE TARGET LIST. The players lists are used to access particular unit types. They are used extensively by the semantic processing routines to assemble the red and blue C2 TREES. The C2 TREES represent the hierarchies of red and blue players in the model. They are the core of the command/control simulation. The PASSIVE TARGET LIST is a list of blue units that are not part of the C2 simulation, but merely targets for red attacks. These structures are explained in detail in the following subsections.
1. **Player Lists**
   a. **Data Block Index**
      
      PLAYERBUFFER  
      PLYST  
      C2
   b. **Description**
      
      The red and blue PLAYER LISTS are used to access COMMAND/CONTROL blocks (C2) of specified types. This structure allows the semantic processing routines to access units without traversing the C2 TREE. This capability is used to construct the C2 TREE in response to VOIL input.
      
      The PLAYER LIST structures consist of a buffer word (PLAYERBUFFER) which points to an array (PLYST) within ISPACE which is dimensioned to the largest number of units in the scenario. Each word in the PLYST array contains a unit type code and a pointer to a C2 block of that type.
   c. **Structure Overview**
      
      1) **Structure Diagram** (Figure III-4)
      
      2) **Block Definitions**
         
         PLAYERBUFFER  Player List Buffer.  
         PLYST  Player List Array. Contains unit type codes and corresponding pointers to C2 blocks  
         C2  Command/Control Block. Contains pointers for C2 Tree. Along with unit number, side and type.
   d. **Block Specifications**
      
      1) **Block Diagrams**
         
         a) **PLAYERBUFFER**
            
            \[
            \begin{array}{c|c}
            PTRPL & VARWORD \\
            \end{array}
            \]
         
         b) **PLYST**
            
            \[
            \begin{array}{c|c}
            ILW & IRW \\
            ILW & IRW \\
            \end{array}
            \]
Figure III-4. Player List Structure Diagram
2) Field Definitions

a) PLAYERBUFFER Block
- PTRPL: Pointer to PLYST Player List
- VARWORD: Total number of players in the list

b) PLYST Array
- ILW: Unit type code (See subsection f)
- IRW: Pointer to the unit's C2 block.

c) C2 Block
- UNITNUMBER: Number of the unit. If negative the unit is a passive target.
- PUP: Pointer to the C2 Block of the unit's commander.
- PDOWN: Pointer to the C2 block of the unit's subordinate.
- PNEXT: Pointer to the C2 block of the unit's sibling.
- PSB: Pointer to the SB block of the unit.
- UNITTYPE: The unit's type code. (See subsection f)
- SIDE: Unit Affiliation.
  1 = Blue (NATO)
  2 = Red (PACT)

e. Linkages to Other Data Structures
The player lists are linked via the C2 blocks to the C2 Tree and Passive Target Lists.
f. Notes

The UNITTYPE Code Definitions used in MADEM are listed in Appendix G.
## MADEM UNITTYPE CODE DEFINITIONS

<table>
<thead>
<tr>
<th>Name</th>
<th>Decimal Number</th>
<th>Octal Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>HAWK BTRY</td>
<td>170</td>
<td>252</td>
</tr>
<tr>
<td>HGRC BTRY</td>
<td>180</td>
<td>264</td>
</tr>
<tr>
<td>PAT BTRY</td>
<td>175</td>
<td>257</td>
</tr>
<tr>
<td>HAWK BOC</td>
<td>150</td>
<td>226</td>
</tr>
<tr>
<td>HERC BOC</td>
<td>160</td>
<td>240</td>
</tr>
<tr>
<td>PAT BOC</td>
<td>155</td>
<td>233</td>
</tr>
<tr>
<td>CRC</td>
<td>130</td>
<td>202</td>
</tr>
<tr>
<td>AIR BASE</td>
<td>220</td>
<td>334</td>
</tr>
<tr>
<td>TAB</td>
<td>158</td>
<td>236</td>
</tr>
<tr>
<td>AWACS</td>
<td>132</td>
<td>204</td>
</tr>
<tr>
<td>LANCE</td>
<td>86</td>
<td>126</td>
</tr>
<tr>
<td>HJ</td>
<td>34</td>
<td>42</td>
</tr>
<tr>
<td>BRIDGE</td>
<td>39</td>
<td>47</td>
</tr>
<tr>
<td>DEPOS</td>
<td>40</td>
<td>50</td>
</tr>
<tr>
<td>PERSHING</td>
<td>83</td>
<td>123</td>
</tr>
<tr>
<td>POL</td>
<td>84</td>
<td>124</td>
</tr>
<tr>
<td>SASP</td>
<td>210</td>
<td>322</td>
</tr>
<tr>
<td>ASP</td>
<td>87</td>
<td>127</td>
</tr>
<tr>
<td>RESERVES</td>
<td>88</td>
<td>130</td>
</tr>
<tr>
<td>TRAINS</td>
<td>89</td>
<td>131</td>
</tr>
<tr>
<td>CLV BTRY</td>
<td>94</td>
<td>136</td>
</tr>
<tr>
<td>VII BTRY</td>
<td>90</td>
<td>132</td>
</tr>
<tr>
<td>CORP CP</td>
<td>95</td>
<td>137</td>
</tr>
<tr>
<td>DIV CP</td>
<td>92</td>
<td>134</td>
</tr>
<tr>
<td>SOC</td>
<td>153</td>
<td>231</td>
</tr>
<tr>
<td>ATAF</td>
<td>128</td>
<td>200</td>
</tr>
</tbody>
</table>
## AIRCRAFT UNITTYPE CODES (INTERNAL)

<table>
<thead>
<tr>
<th></th>
<th><strong>Decimal</strong></th>
<th><strong>Octal</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BLUE</strong></td>
<td>401 - 419</td>
<td>621 - 643</td>
</tr>
<tr>
<td>AWAC</td>
<td>499</td>
<td>763</td>
</tr>
<tr>
<td><strong>RED</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fighters</td>
<td>420 - 439</td>
<td>644 - 607</td>
</tr>
<tr>
<td>F/B</td>
<td>440 - 459</td>
<td>670 - 713</td>
</tr>
<tr>
<td>Bombers</td>
<td>460 - 478</td>
<td>714 - 737</td>
</tr>
<tr>
<td>Unassigned</td>
<td>480 - 498</td>
<td>704 - 762</td>
</tr>
</tbody>
</table>
2. C2 Trees
   a. Data Block Index
      C2
      PLBUFFER
      SB
      SDB
   b. Description
      Two tree data structures are used to represent the command
      and control hierarchies for the Blue (NATO) and Red (PACT) forces. Only
      player, as opposed to passive target, units are included in these trees.
      The overall structure of these trees is shown in the structure diagram below.
      The command and control hierarchies for red and blue forces used by the model
      are shown in subsection f.
   c. Structure Overview
      1) Structure Diagram (Figure III-5)
      2) Block Definitions
         PLBUFFER - C2 Tree Buffer Block. Contains pointer
to the tree ans the side of the tree.
         Command/Control Block. Contains tree
pointers, unit number, pointer to score-
d (SB), unit type code and side.
         SB - Scoreboard Block. Contains pointers to
C2 and HEX blocks, and the status display
board (SDB). Also contains pointers to
acquisition devices various unit status
blocks and the future event list. Use
varies with unit type.
         SDB - Status Display Board Block. Contains
subordinate, acquisition and order pointers.
Use varies with unit type.
Figure III-5. C2 Tree Structure Diagram
d. Block Specifications

1) Block Diagrams

a) PLBUFFER

<table>
<thead>
<tr>
<th>PTRPL</th>
<th>VARWORD</th>
</tr>
</thead>
</table>

b) C2

<table>
<thead>
<tr>
<th>UNIT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUP</td>
</tr>
<tr>
<td>PSB</td>
</tr>
<tr>
<td>UNITTYPE</td>
</tr>
</tbody>
</table>

Note: + Alternate Definition of Field

c) SB

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSDB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
<tr>
<td>DATABASE</td>
<td>PABSTATNS</td>
</tr>
</tbody>
</table>

Field Definitions:

a) PLBUFFER Block

<table>
<thead>
<tr>
<th>PTRPL</th>
<th>Pointer to C2 block</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARWORD</td>
<td>Side Indicator for tree:</td>
</tr>
<tr>
<td></td>
<td>= Blue (NATO)</td>
</tr>
<tr>
<td></td>
<td>= Red (PACT)</td>
</tr>
</tbody>
</table>
b) C2 Block
- UNITNUMBER - Number of the unit. If negative the unit is a passive target.
- PUP - Pointer to the C2 block of the unit's commander
- PDOWN - Pointer to the C2 block of the unit's subordinate.
- PNEXT - Pointer to the C2 block of the unit sibling.
- PSB - Pointer to the SB block of the unit.
- UNITTYPE - The unit's type code (see subsection f)
- SIDE - Unit Affiliation.
  1 = Blue (NATO)
  2 = Red (PACT)


c) SB Block
- ADDRESS - Pointer to HEX block of the HEX in which the unit is located
- PC2 - Pointer to C2 block of the units.
- PSDB - Pointer to the SB block of the unit.
- PFEL - Pointer to future event list event block.
- PACQ - Pointer to ACOBUF block. Used by CRC's for acquisition devices.
- ID - Identification number
- DATABASE - If BOC or BTRY points to ADSITEDG block
- PABSTATUS - Points to ABSTATUS block if the unit is an air base
- +PARCFTSTAT - Points to ARCTSTATUS block if the unit is a flight of aircraft.
+PBOCSTAT  - Points to BOCSTAT block if the unit is a battalion operations center.
+PBTRYSTAT  - Points to BTRYSTAT block if the units is an antiaircraft battery.
+STATUS     - Alternative Field Definition.

d) SDB Block
PSB         - Pointer to the unit's SB block.
PSEEBUF     - Pointer to SEEBUF block which is used by aircraft flight units to record damage levels. This field definition is used only by flights.

+PSEE        - Pointer to CRCEES block which is used by CRC units to record the blue and red flights it sees, and by the TRC as a pointer to its assigned targets list TARGETLISTBLOK.

SUBORDINATE - Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs. field use combinations are as follows:

<table>
<thead>
<tr>
<th>Unit Type</th>
<th>Subordinate Points to</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SUBLIST</td>
</tr>
<tr>
<td>BTRY</td>
<td>Fireunit</td>
</tr>
</tbody>
</table>

ORD         - Points to Orders block if the unit is a flight.
+PRAID - Points to the RAIDBLOK if the unit is the red theater commander.

e. **Linkages to other Data Structures**
The C2 TREES are linked to various unit status structures depending upon the type of unit represented by the C2 blocks.

f. **Notes**
Figures III-6, and III-7 show the hierarchies of blue and red player units. These hierarchies are representative of model processes only and are not direct models of the real world.
*NOTE. TEMPORARILY LAUNCHED FROM A.B. EVENTUALLY RETURNS TO A.B.

Figure III-6. Blue Command/Control Structure
Figure III-7. Red Command/Control Structure
3. **Passive Target List**

a. **Data Block Index**

C2  
SB (Modified)

b. **Description**

The PASSIVE TARGET LIST is made up of C2 and associated SB blocks which are in a linked list. This list is separate from the C2 TREE and is used to represent Blue units which are non-players. These Blue units are non-players in the sense that they merely act as objectives for red attacks.

Passive target units are characterized by negative unit numbers in their C2 blocks and a modified SB block which has three rather than the usual four words. The configuration of the passive target list is shown in the structure diagram.

c. **Structure Overview**

1) **Structure Diagram** (III-8)

2) **Block Definitions**

| C2 | Command/Control Block. Contains list pointers, a negative unit number, pointer to SB, unit type code and side. |
| SB (Modified) | Scoreboard Block. Three rather than the usual four words. Contains pointers to C2 and HEX blocks. |

d. **Block Specifications**

1) **Block Diagrams**

   a) **C2**

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
<th>PUP</th>
<th>PDOWN</th>
<th>PSB</th>
<th>PNEXT</th>
<th>UNITTYPE</th>
<th>SIDE</th>
</tr>
</thead>
</table>

86
Figure III-8. Passive Target List Structure Diagram
2) Field Definitions
   a) C2 Block
      UNITNUMBER - Number of the Unit. If negative, the unit is a passive target.
      PUP - Pointer to the C2 block of the unit's commander.
      PDOWN - Pointer to the C2 block of the unit's sibling.
      PSB - Pointer to the SB block of the unit.
      UNITTYPE - The unit's type code (see subsection f)
      SIDE - Unit Affiliation.
         1 = Blue (NATO)
         2 = Red (PACT)
      PACQ - Pointer to ACQBUF block. Used by CRC's for acquisition devices (not used)
      ID - Identification Number

e. Linkages to Other Data Structures
   The PASSIVE TARGET LIST is used exclusively for targeting by the red planning module. Its only external unrange is to the HEX block in which the unit is located.

f. Notes
F. **RED STRUCTURES**

The Red Structures are used to simulate the Red Threat Planning and attack processes. The Red Theater Commander (RTC) is the focal point of all other Red Structures. The RED THEATER COMMANDER structures together with its associated CORRIDOR DESCRIPTION lists and ATTACK REQUIREMENTS/ALLOCATIONS lists controls the assignment of flights to specific targets. Potential targets are obtained from the POTENTIAL TARGET LIST and placed on either the ASSIGNED TARGET list or NONAVAILABLE TARGET list by the RTC. Orders are then generated for each flight which determine the mission profile for the COMMINS attack. Once flights have been launched many of the planning structures mentioned above are no longer required. They are therefore released for other uses.
1. **Red Theater Commander**
   a. **Data Block Index**
      
      C^2
      RAIDBLOK
      SB
      SOB
      STDBLOK
      TARGETLISTBLOK
      TTDBLOK
      WAVEBLOK
   b. **Description**
      The Red Theater Commander is the core of the Red Attack Planning Process. It controls execution of raids and waves through its RAIDBLOK/WAVEBLOK list. Similarly, choice of targets is controlled by the commander's PERCEPTIONS LIST (also referred to as the ASSIGNED TARGET LIST). The Red Theater Commander also controls the allocation of flights to targets on its PERCEPTIONS LIST through pointers from its WAVEBLOK blocks to the REQUIREMENTS/ALLOCATIONS LISTS. In addition, pointers from its RAIDBLOK blocks to CORRIDOR DESCRIPTION LISTS allow the Red Commander to assign attacks to various corridors.
   c. **Structure Overview** (Figure III-9)
      1) **Structure Diagram**
      2) **Block Definitions**

      | Block   | Description                                      |
      |---------|--------------------------------------------------|
      | C^2     | COMMAND/CONTROL BLOCK. Contains unit no, type, side and C^2PTRS. |
      | SB      | SCOREBOARD BLOCK. Contains HEX PTR, Unit ID, PTR to FEL. |
      | SOB     | STATUS DISPLAY BLOCK. Contains PTRS to assigned tgt list (perceptions list) and RAIDBLOK initialization. |
      | RAIDBLOK| RAID DESCRIPTION BLOCK. Contains basic parameters for a raid including no. of waves and corridors. |
WAVEBLOK - WAVE DESCRIPTION BLOCK. Contains basic parameters for a wave including no. of tgt types, start time and duration.

TARGETLISTBLOK - TARGET LIST BUFFER. Contains number of tgt type (TTDBLOK) blocks in the list.

TTDBLOK - TARGET TYPE BLOCK. Contains TGT type code and PTR to next TTDBLOK type, also contains PTR to a list of specific tgts of the same type (STDBLOK) and a count of the no. of specific targets.

STDBLOK - SPECIFIC TARGET DESCRIPTION BLOCK. Contains PTRs to tft SB and HEX and to its TGPTTREE. Also contains perceived damage level and a PTR to the next STDBLOK.

d. Block Specifications
1) Block Diagrams
a) \( C^2 \)

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNITTYPE</td>
</tr>
<tr>
<td>SIDE</td>
</tr>
<tr>
<td>PSB</td>
</tr>
<tr>
<td>PNEXT</td>
</tr>
<tr>
<td>PDOWN</td>
</tr>
<tr>
<td>PUP</td>
</tr>
</tbody>
</table>

b) SB

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSDB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
<tr>
<td>DATABASE</td>
<td>PABSTANS</td>
</tr>
</tbody>
</table>

| + PARCPSTAT |
| + PBOCSTAT  |
| + PBTRYSTAT |
| + BTSTATUS  |
### c) SDB

<table>
<thead>
<tr>
<th>PSB</th>
<th>PSEEBUF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSEE</td>
</tr>
<tr>
<td>SUBORDINATE</td>
<td>ORD</td>
</tr>
<tr>
<td></td>
<td>PRAID</td>
</tr>
</tbody>
</table>

### d) TARGET LIST BLOK

| PTRTYTL | NOTYTPL |

### d) TTDBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>NRTGTYP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTGTPL</td>
<td>NOTGTPL</td>
</tr>
</tbody>
</table>

### f) STDBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>PTGTSB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PTGTLTR</td>
</tr>
<tr>
<td></td>
<td>DAMAGPER (SPACE)</td>
</tr>
<tr>
<td></td>
<td>PADRPER</td>
</tr>
</tbody>
</table>

### g) RAIDBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>NRRAID</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTRWAVE</td>
<td>NOWAVES</td>
</tr>
<tr>
<td>PTALORD</td>
<td>NOCORDS</td>
</tr>
</tbody>
</table>

### h) WAVEBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>NRWAVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTGTYPE</td>
<td>NOTGTYP</td>
</tr>
<tr>
<td></td>
<td>STARTIME (SPACE)</td>
</tr>
<tr>
<td></td>
<td>DURATION (SPACE)</td>
</tr>
</tbody>
</table>
2) **Field Definitions**

a) **C² BLOCK**

- **UNITNUMBER** - number of the unit.
- **PUP** - Pointer to the C² block of the unit's commander.
- **PDOWN** - Pointer to the C² block of the unit's subordinate.
- **PNEXT** - Pointer to the C² block of the unit's sibling.
- **PSB** - Pointer to the SB block of the unit.
- **UNITTYPE** - The Unit's type code.
- **SIDE** - Unit affiliation.
  
  \[1 = \text{BLUE}\]
  \[2 = \text{RED (PACT)}\]

b) **SB Block**

- **ADDRESS** - Pointer to HEX block of the hex in which the unit is located.
- **PL2** - Pointer to C² block of the unit.
- **PSDB** - Pointer to the SDB block of the unit.
- **PFEL** - Pointer to future event list EVENT block.
- **PACQ** - Pointer to ACQBUF block. Used by CRC's for acquisition devices.
- **ID** - Identification number.
- **DATABASE** - If BOC or BTRY points to ADSITEDB block.
- **PBSTATUS** - Points to ABSTATUS block if the unit is an airbase.
+ PARCFTSTAT - Points to AFCFTSTATUS block if the unit is a flight of aircraft.
+ PBOCSTAT - Points to BOCSTAT block if the unit is a battalion operations center.
+ PBTRYSTAT - Points to BTRYSTAT block if the unit is an antiaircraft battery.
+ STATUS - Alternative Field Definition.

**c) SDB Block**
PSB - Pointer to the Unit's SB block.
PSEEBUF - Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights.

+ PSEE - Pointer to CRCEES block which is used by CRC units to record the blue and red flights it sees. Also used by the RTC to point to its assigned targets list TARGETLISTBLOK.

SUBORDINATE - Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs. field use combinations are as follows:
<table>
<thead>
<tr>
<th>UNIT TYPE</th>
<th>SUBORDINATE POINTS TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SUBLIST</td>
</tr>
<tr>
<td>BTRY</td>
<td>FIREUNIT</td>
</tr>
</tbody>
</table>

| ORD       | Points to ORDERS block if the unit is a flight. |
| + PRAID   | Points to the RAIDBLOK if the unit is the red theater commander. |

**d) TARGETLISTBLOK BLOCK**

| PTRTYTL   | Pointer to TTDBLOK block |
| NOTYTPL   | Number of TTDBLOK blocks (target types) in the list. |

**e) TTDBLOK BLOCK**

| PNEXT     | Pointer to next TTDBLOK block in the list |
| NRTGTYP   | Pointer to the STDBLOK block list |
| PTGTPL    | Pointer to the STDBLOK block list |
| NGTGTPL   | Number of STDBLOK blocks in the list |

**f) STDBLOK BLOCK**

<p>| PNEXT     | Pointer to the next STDBLOK block in the ASSIGNED TARGET LIST |
| PTGTSB    | Pointer to SB block of the specific target |
| PTGTLTR   | Pointer to TGPTTREE node |
| DAMAGPER  | Perceived damage to target. Stored as a real variable. Initialized to 2. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADRPER</td>
<td>Pointer to HEX block in which target unit is located.</td>
</tr>
<tr>
<td>g) RAIDBL BLOCK</td>
<td>Pointer to the next RAIDBLOK block in the RAIDBLOK list.</td>
</tr>
<tr>
<td>PNEXT</td>
<td>Raid number.</td>
</tr>
<tr>
<td>NRRAID</td>
<td>Pointer to WAVEBLOK BLOCK.</td>
</tr>
<tr>
<td>PTRWAVE</td>
<td>First in list.</td>
</tr>
<tr>
<td>NOWAVES</td>
<td>Number of waves in the list.</td>
</tr>
<tr>
<td>PTRCORD</td>
<td>Pointer to first CORRIDORBLOK in the CORRIDOR DESCRIPTION LIST.</td>
</tr>
<tr>
<td>NOCORDS</td>
<td>Number of corridors.</td>
</tr>
<tr>
<td>h) WAVEBLOK BLOCK</td>
<td>Pointer to the next WAVEBLOK block in the list.</td>
</tr>
<tr>
<td>PNEXT</td>
<td>Wave number</td>
</tr>
<tr>
<td>NRWAVE</td>
<td>Pointer to first TARGETBLOK block in the list of target types to be attacked in the wave.</td>
</tr>
<tr>
<td>NOTGTYT</td>
<td>Number of target types in the wave.</td>
</tr>
<tr>
<td>STARTIME</td>
<td>Start time. Real variable</td>
</tr>
<tr>
<td>DURATION</td>
<td>Length of wave. Real variable</td>
</tr>
</tbody>
</table>

e. Linkages to Other Data Structures
The Red Theater Commander shares its STDBLOK blocks with the POTENTIAL TARGET LIST and the NONAVAILABLE TARGET LIST. In addition, its RAIDBLOK blocks point to the CORRIDOR DESCRIPTION STRUCTURES and its WAVEBLOK blocks point to the attack REQUIREMENTS/ALLOCATION structures.
f. Notes
2. Potential Target List
   a. Data Block Index
      FOREST
      FORTGTBUFFER
      STDBLOK
      TGTPTREE
   b. Description

   The POTENTIAL TARGET LIST is used to keep track of potential targets by the Red Theater Planning Module. It consists of a linker list of FOREST blocks. Each of which corresponds to a particular target type found in the Blue C^2 TREE and PASSIVE TARGET LIST (See 0.2 and 0.3). Each FOREST block points to a leftist tree composed of linked TGTPTREE and STDBLOK blocks. These trees contain location, damage and command/control information for specific targets of the type found in the origin FOREST blocks. These specific target trees are sometimes referred to as PROBABALISTIC EVENT TREES.

   Within each tree the TGTPTREE blocks act as the nodes of the tree while the STDBLOK blocks act as repositories of key information on the target. The nodes of the tree are sorted by damage level with the least damaged target residing at the top of the tree. For a complete explanation of the leftist tree sorting algorithm see APPENDIX E. Perceived damage is stored in STDBLOK while actual damage is stored in TGTPTREE block.

   STDBLOK blocks may be strung into linked lists to form the assigned target list discussed in E.2. The overall configuration of the POTENTIAL TARGET LIST is shown in the structure diagram.

c. Structure Overview
   1) Structure Diagram (Figure III-10)
   2) Block Definitions
      FORTGTBUFFER - FOREST LIST BUFFER BLOCK. Contains number of forest blocks.
      FOREST - TARGET TYPE BLOCK. Contains TGTTYPE code and a PTR to a tree of specific targets of that type.
Figure III-10. Potential Target List Structure Diagram
TGTPTREE - TARGET TREE NODE. Contains PTAS to left and right nodes of leftist tree used to sort specific TGTs by damage level. Also contains PTR to its corresponding STDBLOCK. TGTPTREE damage level is the actual damage level.

STDBLOK - SPECIFIC TARGET DESCRIPTION BLOCK. Contains PTRS to TGT SB and HEX and to its corresponding TFTP TREE. Also contains the perceived damage level for the TGT. STDBLOCKS have a PNEXT field which allows them to be threaded into a linked list of assigned targets.

d. Block Specifications

1) Block Diagrams

a) FORTGTBUFFER

```
| PFOREST | VARWORD |
```

b) FOREST

```
| PNEXT   | NRTYPE |
| PTREE   |       |
```

c) TGTPTREE

```
| DAMAGE  |         |
| PSTDBLOK| DIST   |
|         | PLEPT  |
|         | PRITE  |
```

d) STDBLOK

```
| PNEXT | PTGTSB |
|       |       |
|       | PTGTLTR|
|       |        |
|       | DAMAGPER|
|       | PADRPER |
|       | F-9     |
```
2) **Field Definitions**

a) **FORTFTBUFFER BLOCK**
   - **PFOREST** - Pointer to FOREST block.
   - **VARWORD** - Not used

b) **FOREST BLOCK**
   - **PNEXT** - Pointer to next FOREST block.
   - **PTREE** - Pointer to top TGTPTREE block in probabilistic event tree.

c) **TGTPTREE NODE**
   - **DAMAGE** - Actual damage to the target. Stored as a real variable. Initialized to 2.0
   - **PSTDBLOK** - Pointer to STDBLOK block.
   - **DIST** - Distance leaf of the tree in nodes.
   - **PLEFT** - Pointer to left TGTPTREE node in the tree.
   - **PRITE** - Pointer to right TGTPTREE node in the tree.

d) **STDBLOK BLOCK**
   - **PNEXT** - Pointer to the next STDBLOK block in the ASSIGNED TARGET LIST.
   - **PTGTNSB** - Pointer to SB block of the specific target.
   - **PTGLTTR** - Pointer to TGTpTREE node
   - **DAMAGPER** - Perceived damage to target. Stored as a real variable. Initialized to 2.
   - **PAORPER** - Pointer to HEX block in which target unit is located.

e. **Linkages to other Data Structures**
   - The POTENTIAL TARGET LIST shares its STDBLOK blocks with the ASSIGNED TARGET LIST and the NONAVAILABLE TARGET LIST.

f. **Notes**
   - The relationship of the potential and assigned target lists is illustrated in Figure III-11.
Figure III-11. Potential/Assigned Target List Structures
3. **Assigned Target List**  
   (Red Commander Perceptions List)  
   a. **Data Block Index**  
      ATTACKBLOK  
      SDB  
      STDBLOK  
      TARGETLISTBLOK  
      TTDBLOK  
   b. **Description**  
      The ASSIGNED TARGET LIST is used by the Red Theater Planning module to keep track of targets designated for attack in the current raid. It is actually a series of nested lists with lists of specific target description blocks (STDBLOK) stratified by target type. The STDBLOK blocks are obtained from the POTENTIAL TARGET LIST. The desired STDBLOK blocks in the potential target list are pointed to be specific target resource allocation blocks (ATTACKBLOK) created by the Red Theater Planner when resources are matched to attack requirements. (See E.4 for further details). The overall configuration of the ASSIGNED TARGET LIST is shown in the structure diagram. The assigned target list is attached to the Red Theater Commander's SDB block.  
   c. **Structure Overview**  
      1) **Structure Diagram** (Figure III-12)  
      2) **Block Definitions**  
         TARGETLISTBLOK - **TARGET LIST BUFFER.** Contains number of TGT TYPE (TTDBLOK) blocks in the list.  
         TTDBLOK - **TARGET TYPE BLOCK.** Contains TGT type code and PTR to next TTDBLOK type, also contains PTR to 4 list of specific tgts. of the same type (STDBLOK) and a count of the no. of specific targets.
Figure III-12. Assigned Target List Structure Diagram
(Official Commander Perception List)
STDBLOK - SPECIFIC TARGET DESCRIPTION BLOCK. Contains PTRS to tgt. SB and HEX and to its TGPTREE. Also contains perceived damage level and a PTR to the next STDBLOK.

SDB - STATUS DISPLAY BOARD of Red Commander
SB - SCOREBOARD of Red Commander
C - COMMAND CONTROL BLOCK of Red Commander
ATTACKBLOK - SPECIFIC TARGET ATTACK RESOURCE ALLOCATION BLOCK for target.

d. Block Specifications
1. Block Diagrams
   a) SDB
      - PSB | PSEEBUF
      - + | PSEE
      - SUBORDINATE | ORD
      - + | PRAID
   b) TARGETLISTBLOK
      - PTRTYTL | NOTYTPL
   c) TTDBLOK
      - PNEXT | NRTGTYP
      - PTFTPL | NOTGTP
   d) STDBLOK
      - PNEXT | PTGTSB
      - | PTGTLTR
      - DAMAGPER (SPACE)
      - | PADRPER
   e) ATTACKBLOK
      - PNEXT | PNXTGIL
      - PFMAKTC | NOFMAKT
      - ISECTOR | PNXTCRD
2) Field Definitions

a) SBD BLOCK
   PSB - Pointer to the Unit's SB block.
   PSEEBUF - Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights.
   +PSEE - Pointer to CRCEES block which is used by CRC units to record the Blue and Red Flights it sees. Also used by the RTC as a pointer to its assigned targets list.

SUBORDINATE - Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs. field use combinations are as follows:

<table>
<thead>
<tr>
<th>UNIT TYPE</th>
<th>SUBORDINATE POINTS TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SUBLIST</td>
</tr>
<tr>
<td>BTRY</td>
<td>FIREUNIT</td>
</tr>
</tbody>
</table>

ORD - Points to ORDERS block if the unit is a flight.
+ PRAID - Points to the RAIDBLOK if the unit is the Red Theater commander.

b) TARGETLISTBLOK BLOCK
   PTRTYTL - Pointer to TTDBLOK block
   NOTYTPL - Number of TTDBLOK blocks (target types) in the list.
c) TTDBLOK BLOCK
PNEXT - Pointer to next TTDBLK block in the list
NRTGTYP - Target type code.
PTGTPN - Pointer to the STDBLOK block list
NOTGTPN - Number of STDBLOK blocks in the list

d) STDBLOK BLOCK
PNEXT - Pointer to the next STDBLOK block in the ASSIGNED TARGET LIST.
PTGTSB - Pointer to SB block of the specific target.
PTGTLTR - Pointer to TFTPTREE node.
DAMAGPER - Perceived damage to target.
            Stored as a real variable.
            Initialized to 2.
PADRPER - Pointer to HEX block in which target unit is located.

e) ATTACK BLOCK
PNEXT - Pointer to next ATTACK BLOK in list.
PNXTGTL - Pointer to STDBLOK block.
PFMAKTG - Pointer to FAKTGBLOK block.
NOFMAKT - Number of FAKTGBLOK blocks (formation attacking target) in the list.
ISECTOR - Sector of the attack relative to the attack corridor. (see corridor)
P NXTCRD - Pointer to the CORRIDOR block for the attack.

e. Linkages to Other Data Structures
The ASSIGNED TARGET LIST shares its STDBLOK blocks with the POTENTIAL TARGET LIST and the NONAVAILABLE TARGET LIST. In addition its STDBLOK blocks are pointed to by ATTACKBLOK blocks which reside in the attack REQUIREMENTS/ALLOCATIONS structure.
f. Notes

Figure III-11 illustrates the relationship of the potential and assigned target lists. This relationship is crucial to an understanding of the target assignment process.
4. Nonavailable Target List
   a. Data Block Index
      NOAVAILBLOK
      STDBLOK
   b. Description
      The NONAVAILABLE TARGET LIST is used by the Red Theater Planning Module to keep track of targets which cannot be attacked in the current raid. Targets are placed on the NONAVAILABLE TARGET LIST when they are not in the allowable attack shown in Subsection F and when attack resources are not sufficient to match mission requirements. The overall configuration of the nonavailable target list is shown in the structure diagram.
   c. Structure Overview
      1) Structure Diagram (Figure III-13)
      2) Block Definition
         NOAVAILBLOK - NON-AVAILABLE TARGET BLOCK. Created by TGTGONE routine and used by other targets which are not available for attack due to lack of resources or geographic unsuitability. Contains PTR to STDBLOCK and PTR to next NOAVAILBLOK. Also contains perceived damage level. (Projected damage level).
         STDBLOK - SPECIFIC TARGET DESCRIPTION BLOCK. Contains target SB and HEX PTRS as well as a PTR to the TGTPTREE. Also contains the perceived damage level for the target and a PNEXT field which allows them to be threaded into a linked list of assigned targets.
Figure III-13. Nonavailable Target List Structure Diagram
d. **Block Specifications**

1) **Block Diagrams**

a) **NOAVAILBLOK**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Pointer to NOAVAILBLOK block</td>
</tr>
<tr>
<td>PJDAMAGE</td>
<td>Projected damage level</td>
</tr>
<tr>
<td>PSTDBLOK</td>
<td>Pointer to STDBLOK block on the POTENTIAL TARGET LIST.</td>
</tr>
</tbody>
</table>

b) **STDBLOK**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Pointer to the next STDBLOK block in the ASSIGNED TARGET LIST.</td>
</tr>
<tr>
<td>PTGTSB</td>
<td>Pointer to SB block of the specific target.</td>
</tr>
<tr>
<td>PTGTLTR</td>
<td>Pointer to TGTPTREE node</td>
</tr>
<tr>
<td>DAMAGPER</td>
<td>Perceived damage to target. Stored as a real variable. Initialized to 2.</td>
</tr>
<tr>
<td>PADRPER</td>
<td>Pointer to HEX block in which target unit is located.</td>
</tr>
</tbody>
</table>

e. **Linkages to Other Data Structures**

The nonavailable target list shares its' STDBLOK blocks with the POTENTIAL TARGET LIST and the ASSIGNED TARGET LIST.
f. Notes

Figure III-14 illustrates the attack sectors relative to an attack corridor.
Figure III-14. Attack Sectors

ACCEPTABLE
TARGET
AREAS
ATTACK ZONE = 4
CORRIDOR CENTER = 1
RIGHT BUFFER = 3
LEFT BUFFER = 2
UNACCEPTABLE = Ø
5. **Corridor Description Lists**
   a. **Data Block Index**
      - ABVCR
      - ACTAB
      - CONSTBLOK
      - CORRIDORBLOK
      - HEXBLOK
      - HEXLINK
   b. **Description**

      The CORRIDOR DESCRIPTION LISTS are used by the Red Theater Planning Module to specify the boundaries of the attack corridors input by the user. They are also used to keep track of the assignment of airbases to corridors within aircraft range and the types of aircraft available on the bases. The general configuration of attack corridors is illustrated in Subsection F.

   c. **Structure Overview**
      1) **Structure Diagram** (Figure III-15)
      2) **Block Definitions**
         - CORRIDORBLOK - CORRIDOR DESCRIPTION BLOCK. Contains basic corridor parameters input by user.
         - CONSTBLOK - CORRIDOR CONSTANTS BLOCK. Contains values calculated from corridorblok contents which describe the corridor boundaries.
         - HEXBLOK - HEXLINK BUFFER BLOCK. Contains part to five HEXLINK lists which specify the corridor in terms of hexes.
         - HEXLINK - HEXLINK BLOCK. Contains ptrs. to next HEXLINK and PTR to HEX.
Figure III-15. Corridor Description List Structure Diagram
d. Block Specifications

1) Block Diagrams

a) CORRIDORBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>NRCORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>PABUSCOR</td>
<td>NOABVCR</td>
</tr>
<tr>
<td>PLHEX</td>
<td>PRHEX</td>
</tr>
<tr>
<td>PCHEX</td>
<td>NHWIDTH</td>
</tr>
<tr>
<td>PHLIST</td>
<td>PBDCNST</td>
</tr>
<tr>
<td>DEPTHLR (SPACE)</td>
<td></td>
</tr>
<tr>
<td>ANGCORD (SPACE)</td>
<td></td>
</tr>
<tr>
<td>ANGSPRD (SPACE)</td>
<td></td>
</tr>
<tr>
<td>BUFRWDH (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

b) CONSTBLOK

- CORDSCOPE
- YINTLBUF
- YINTLCOR
- YINTRCOR
- YINTRBUF
- XSPREAD
- YSPREAD
- ENTRYSLOPE
- YLINENTRY
- YENDCORR
2) Field Definitions

a) CORRIDORBLOK BLOCK

PNEXT  - Pointer to next CORRIDORBLOK block
NRCORD - Corridor Number.
PLHEX  - Pointer to left corridor hex.
PRHEX  - Pointer to right corridor hex.
PCHEX  - Pointer to centerline corridor hex.
PBCNST  - Pointer to CONSTBLOK block.
PABVSCOR  - Pointer to ABVCR block list.
NOABVCT  - Number of ABVER blocks in the list.
PHLIST  - Pointer to HEXBLOK block.
DEPTHCR  - Depth of corridor, real variable.
ANGCORD  - Angle of corridor, real azimuth
ANGSPRD  - Spread angle of corridor exit. real
BUFRWDH  - Sam buffer zone width. real

b) CONSTBLOK BLOK

CORDSCLOPE  - Corridor slope
YINTLBUF  - Y intercept of buffer zone boundary-left.
YINTRCOR - Y intercept of corridor boundary-right.
YINTRBUF - Y intercept of buffer zone boundary - right
XSPREAD - X coordinate of exit spread lines.
YSPREAD - Y coordinate of exit spread lines.
ENTRYSLOPE - Slope of corridor center line.
YLINENTRY - Y intercept of corridor center line.
YENDCORD - Y intercept of corridor exit.

HEXBLOK BLOCK

ITOTAL -
PHEXY - Pointer to HEXLINK list (corridor left to corridor center)
NLIST1 - Number of (PHEX4) HEXLINK blocks in the list
PHEX1 - Pointer to HEXLINK list (buffer left to corridor left)
NLIST2 - Number of (PHEX1) HEXLINK blocks in the list.
PHEX3 - Pointer to HEXLINK list (corridor right to buffer right)
NLIST3 - Number of (PHEX3) HEXLINK blocks in the list.
PHEX6 - Pointer to HEXLINK list (corridor center to corridor right)
NLIST4 - Number of (PHEX6) HEXLINK blocks in the list.
PHEXUNK - Number of HEXLINK list hexes between corridor ends
NLISTUNK - Number of (PHEXUNK) HEXLINK blocks in the list.
d) **HEXLINK BLOCK**
- **PNEXT** - Pointer to next HEXLINK block in the list.
- **HEX** - Pointer to HEX block (Note: could also be a nex number)

e) **ABVCR BLOCK**
- **PNEXT** - Pointer to next ABVCR block in the list
- **PTRABSB** - Pointer to airbase SB block
- **PACTAB** - Pointer to ACTAB block
- **NOACTAB** - Number of ACTAB blocks in the list

f) **ACTAB BLOCK**
- **PNEXT** - Pointer to next ACTAB in the list
- **NRACTYP** - Aircraft type code
- **NOACOH** - Number of aircraft on hand
- **NOACASN** - Number of aircraft assigned

e. **Linkages to other Data Structures**
The CORRIDOR DESCRIPTION LISTS are pointed to by a RAIDBLOK block which describes the raid in which the corridor is used.

f. **Notes**
Figure III-16 illustrates the relationship of HEXLINK lists to corridor boundaries.
Figure III-16. Relationship of HEX LINK Lists To Corridor Boundaries
6. **Attack Requirements/Allocations Lists**
   
a. **Data Block Index**
   
   ATTACKBLOK
   ACTAB
   COMMAND
   FAKTGBLOK
   FDBDBLOK
   FLTAKTBLOK
   FMFLTDB
   FORMATIONBLOK

b. **Description**

   The REQUIREMENTS LIST is used by the Red Theater Planner to specify the number and composition of formations required to attack targets of various types. Each TARGETBLOK block corresponds to a generic target type (e.g., airbase, boc, btry). FORMATIONBLOK blocks correspond to the formation types required for a target type. Each FORMATIONBLOK points to a FDBDBLOK block (formation data base) which in turn points to a list of FMFLTDB blocks (formation flight description) which point to FLTDB blocks (flight data base) for various types of flights. Each flight is homogeneous with respect to aircraft type. The aircraft for a flight must be assembled on a single air base. Partial flights cannot be launched. However, formations can be assembled from flights originating at multiple air bases. The formations required for each target type must be specified by the user for each raid via the UOL.

   The ALLOCATIONS LIST parallels the REQUIREMENTS LIST. It corresponds to the actual allocation of aircraft resources to specific targets of the types specified in the TARGETBLOK blocks. The structure diagram illustrates this parallel structure. FKTGBLOK blocks (formation attacking target) in the ALLOCATIONS LIST correspond to the FORMATIONBLOK blocks in the REQUIREMENTS list. Similarly, FLTAKTBLOK blocks (flight attacking target) correspond to FMFLTDB blocks. It should be noted that all blocks indicated by an asterisk (*) are released after planning, the
only blocks which remain after planning are the database and command blocks required for subsequent operation of FLIGHTS created by the planning module.

c. Structure Overview

1) Structure Diagram (Figure III-17)

2) Block Definitions

<table>
<thead>
<tr>
<th>REQUIREMENTS LIST DEFINITIONS</th>
<th>resource requirements for target types.</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAVEBLOK</td>
<td><strong>WAVE DESCRIPTION BLOCK.</strong> Contains PTRS to next wave and TARGETBLOK list along with the wave number and the number of TARGETBLOK types in the list also contains the start time and duration of the wave.</td>
</tr>
<tr>
<td>TARGETBLOK</td>
<td><strong>TARGET TYPE RESOURCE REQUIREMENTS BLOCK.</strong> Contains target type code, PTRS to formation and attackblok lists and the number of formations and attackbloks in each list. Also contains the maximum aircraft allocation for the TGT type, the number allocated, and the minimum and maximum attack radius.</td>
</tr>
<tr>
<td>FORMATIONBLOK</td>
<td><strong>FORMATION DESCRIPTION BLOCK.</strong> Contains PTRS to next FORMATION BLOCK and to FDBDBLOK as well as the number of formations required and assigned</td>
</tr>
<tr>
<td>FDBDBLOK</td>
<td><strong>FORMATION DATA BASE BLOCK.</strong> Contains basic formation type specifications including pointer to flight data base list (FMFLTDB).</td>
</tr>
<tr>
<td>FMFLTDB</td>
<td><strong>FORMATION COMPONENT FLIGHT DATA BLOCK.</strong> Contains PTR to FLTDB flight data base block for a flight attached to the formation.</td>
</tr>
</tbody>
</table>
Figure III-17. Attack Requirements/Allocations List Structure Diagram

123
ALLOCATIONS LIST DEFINITIONS - Resource Allocations for specific targets.

ATTACKBLOK - SPECIFIC TARGET RESOURCE ALLOCATION BLOCK. Contains PTRS to specific target description block (STDBLOK), formations attacking targetBlist (FAKTGBLOK) and the CORRIDOR block for the corridor through which the attack will take place. Also contains attack sector.

FAKTGBLOK - FORMATIONS ATTACKING TARGET BLOCK. Contains description of a formation assigned to attack the target. Contains PTRS to the FDBDBLOK described above, the next FAKTGBLOK attacking the TGT and a list of FLTAKTBLOK (flights attacking the TGT) which make up the formation. Also contains the number of FLTAKTBLOKS.

FLTAKBLOK - FLIGHTS ATTACKING TARGET BLOCK. Basic description of the flight, its composition, home base and orders.

ACTAB - AIRCRAFT ON AIRBASE BLOCK. Number of aircraft of the type specified in FLTAKTBLOK on hand and assigned at FLT's home base.

COMMAND - COMMAND FOR ENTITY. A list of up to six COMMAND blocks describes all of the actions which must be taken by the Red Flight in the course of its mission. Contains the action to be taken, a PTR to the hex address, and optionally the time at which the action is to occur.
d. Block Specifications

1) Block Diagrams

a) WAVEBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>NRWAVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTGTYPE</td>
<td>NOTGTYP</td>
</tr>
<tr>
<td>STARTIME (SPACE)</td>
<td></td>
</tr>
<tr>
<td>DURATION (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

b) TARGETBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>NRTGTYP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTRFORM</td>
<td>NOFORM</td>
</tr>
<tr>
<td>PTGTATK</td>
<td>NOTGTAK</td>
</tr>
<tr>
<td>MAXACAL</td>
<td>NOACALC</td>
</tr>
<tr>
<td>MAXRHEX</td>
<td>NONRHEX</td>
</tr>
</tbody>
</table>

c) FORMATIONBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>PNXFRDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOFRMRQ</td>
<td>NOFRMAL</td>
</tr>
</tbody>
</table>

d) FDBDBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>NRFORM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTRFLT</td>
<td>NOFLT</td>
</tr>
<tr>
<td>SPFORMC</td>
<td></td>
</tr>
</tbody>
</table>

e) FMFLTDB

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>PNXFLODB</th>
</tr>
</thead>
</table>

f) ATTACKBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>PNXTGTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFMAKTC</td>
<td>NOFMAKT</td>
</tr>
<tr>
<td>ISECTOR</td>
<td>PNXTCRD</td>
</tr>
</tbody>
</table>

g) FAKTGBLOK

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>PNXFRDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFLTAKT</td>
<td>NOFLAKT</td>
</tr>
</tbody>
</table>
h) FLTAKTBLK

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Pointer to next WAVE block</td>
</tr>
<tr>
<td>PNXFDB</td>
<td>Number of waves</td>
</tr>
<tr>
<td>PFLABSB</td>
<td>Pointer to TARGETBLOK</td>
</tr>
<tr>
<td>PTRFRAG</td>
<td>Number of target types</td>
</tr>
<tr>
<td>PFLTSB</td>
<td>Start time for wave</td>
</tr>
<tr>
<td>NOACFLT</td>
<td>Duration of wave</td>
</tr>
<tr>
<td>PNXACAB</td>
<td></td>
</tr>
<tr>
<td>PNXFLDB</td>
<td></td>
</tr>
<tr>
<td>PFLTSB</td>
<td></td>
</tr>
</tbody>
</table>

i) ACTAB

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Number of acts</td>
</tr>
<tr>
<td>NRACHTYP</td>
<td>Pointer to ATTACKBLOK</td>
</tr>
<tr>
<td>NOACOH</td>
<td>Number of formations</td>
</tr>
<tr>
<td>NOACASN</td>
<td></td>
</tr>
</tbody>
</table>

j) COMMAND

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Pointer to next COMMAND block</td>
</tr>
<tr>
<td>NUMACTS</td>
<td>Number of acts</td>
</tr>
<tr>
<td>TMFLG</td>
<td>Address of time</td>
</tr>
<tr>
<td>TIME</td>
<td>(space) Start time for wave</td>
</tr>
<tr>
<td>ACTION</td>
<td>Duration of wave</td>
</tr>
</tbody>
</table>

k) STDBLOK

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Pointer to next WAVE block</td>
</tr>
<tr>
<td>PTGTMB</td>
<td>Number of target types</td>
</tr>
<tr>
<td>PTRFORM</td>
<td>Pointer to FORMATIONBLOK</td>
</tr>
<tr>
<td>NOFORM</td>
<td>Number of formations</td>
</tr>
<tr>
<td>PADRPER</td>
<td></td>
</tr>
</tbody>
</table>

2) Field Definitions

a) WAVE Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Pointer to next WAVE block</td>
</tr>
<tr>
<td>NWRWAVE</td>
<td>Number of waves</td>
</tr>
<tr>
<td>PTGTYP</td>
<td>Pointer to TARGETBLOK</td>
</tr>
<tr>
<td>NｔGTYP</td>
<td>Number of target types</td>
</tr>
<tr>
<td>STARTTIME</td>
<td>Start time for wave</td>
</tr>
<tr>
<td>DURATION</td>
<td>Duration of wave</td>
</tr>
</tbody>
</table>

b) TARGET Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>Pointer to next target block</td>
</tr>
<tr>
<td>NRTGTYP</td>
<td>Number of target types</td>
</tr>
<tr>
<td>PTRFORM</td>
<td>Pointer to FORMATIONBLOK</td>
</tr>
<tr>
<td>NOFORM</td>
<td>Number of formations</td>
</tr>
<tr>
<td>PRGTATK</td>
<td>Pointer to ATTACKBLOK</td>
</tr>
</tbody>
</table>
NOTGTAK - number of attack blocks
MAXACAL - maximum AC allocations
NOACALC - number of AC allocations
MAXRHEX - maximum range in hexes at which the target can be attacked
MINRHEX - minimum range in hexes at which the target can be attacked

c) FORMATIONBLOK
PNEXT - pointer to next FORMATIONBLOK
NOFRMRQ - number of formation req.
NOFRMAL - actual number of formations

d) FDBDBLOK
PNEXT - pointer to next FDBDBLOK
NRFORM - formation number, must be unique
PTRFLT - pointer to formation flight data block (FMFLTDB)
NOFLTL - number of flights in the formation
SPFORMC - formation cruise speed in meters/seconds

e) FMFLTDB
PNEXT - pointer to next formation flight block (FMFLTDB)
PNXFLDB - pointer to flight data block (FLTDB)
f) ATTACKBLOK
PNEXT - pointer to next ATTACKBLOK
PNXTGTL - pointer to STDBLOK
PFMAKTG - pointer to formation attacking target block (FAKTGBLOK)
NOFMAKT - number of formation attack blocks
ISECTOR - number of sector to be attacked, relative to attack corridor
PNXTCRD - pointer to corridor block (CORRIDORBLOK)
g) **FAKTGBLOK**
- **PNEXT**: pointer to next formation attacking target block (FAKTGBLOK)
- **PNXFRDB**: pointer to formation data base block (FDBDBLOK)
- **PFLTAKT**: pointer to flight attacking target block (FLTAKTBLOK)
- **NOFLAKT**: number of flights attacking target blocks

h) **FLTAKTBLOK**
- **PNEXT**: pointer to next FLTAKTBLOK
- **PNXFLDB**: pointer to FLTDB
- **NOACFLT**: number of actual flights
- **PFLABSB**: pointer to air base scoreboard
- **PNXACAB**: pointer to aircraft on air base block (ACTAB)
- **PTRFRAG**: pointer to COMMAND block
- **PFLTSB**: pointer to flight scoreboard

i) **ACTAB**
- **PNEXT**: pointer to next ACTAB
- **NRACTYP**: aircraft type number
- **NOACOH**: number of aircraft on hand
- **NOACASN**: number of aircraft assigned

j) **COMMAND**
- **PNEXT**: pointer to next COMMAND block
- **NUMACTS**: number of the command
- **TMFLG**: time flag, if 1 a time is associated with the command
- **ADDRESS**: pointer to hex block for the command
- **TIME**: time command is to be performed
- **ACTION**: command or action code
k) **STD BLOK**
   
   PNEXT  pointer to next STDBLOK
   PTGTSB pointer to target scoreboard
   PTGTLTR pointer to TGTPTREE
   DAMGPER perceived damage level
   PADRPER pointer to hex address

   e. **Linkages to Other Data Structures**

   The REQUIREMENTS LIST is pointed to by the WAVEBLOK block which is associated with the RED THEATER COMMANDER. The ATTACKBLOK block in the ALLOCATIONS LIST points to an STDBLOK block (specific target description block) in the Red theater commander's ASSIGNED TARGET LIST (also known as the Red theater commander perceptions list). Since these STDBLOK blocks are also in the POTENTIAL AND UNAVAILABLE TARGET LISTS, the allocations list is also connected to both the POTENTIAL AND UNAVAILABLE TARGETS LISTS.

   f. **Notes**
7. RED AIR BASES
   a. Data Block Index
      ABINFO
      ABQUEDB
      ABSTATUS
      ACDB
      C2
      QUEUES
      QUESTAT
      SB
   b. Description
      RED AIR BASE STRUCTURES are designed to keep track of the
      number and type of aircraft on the AIRBASE as well as the status of each
      aircraft type in terms of launch capability. The overall configuration of
      this structure is shown in the structure diagram.
   c. Structure Overview
      1) Structure Diagram (See Figure III-18)
      2) Block Definitions
         
         C2 - COMMAND CONTROL BLOCK, contains unit NO, type and side, and C2 PTRS.
         SB - SCOREBOARD BLOCK, contains Hex address, MR, unit ID, PTR to future event list for the unit, and ABSTATUS PTR.
         SDB - STATUS DISPLAY BLOCK. Not used by AIRBASE.
         ABSTATUS - AIR BASE STATUS BLOCK, contains PTRS to the AIRBASE information list (ABINFO) and the QUEUES list for each aircraft type on base. Also contains no of A/C on base and no of types of A/C on base.
         ABINFO - AIRCRAFT ON BASE INFORMATION BLOCK. Contains aircraft type, no on hand and no in each service queue, and a PTR to A/C type's ACDB.
Figure III-18. Red Air Base Structure Diagram
**d. Block Specification**

1) **Block Diagrams**

<table>
<thead>
<tr>
<th>a) C2</th>
<th>UNITIONBER</th>
<th>PUP</th>
<th>PDOWN</th>
<th>PSB</th>
<th>PNEXT</th>
<th>UNITTYPE</th>
<th>SIDE</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>b) SB</th>
<th>ADDRESS</th>
<th>PC2</th>
<th>PSDB</th>
<th>PFEL</th>
<th>PACQ</th>
<th>DATABASE</th>
<th>PABSTATUS</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>c) ABSTATUS</th>
<th>PACTAB</th>
<th>NOACTAB</th>
<th>PTR2QUES</th>
<th>NOACONAB</th>
<th>ABDAMAGE (SPACE)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>d) QUEUES</th>
<th>NEXT</th>
<th>QUENUM</th>
<th>PTR</th>
<th>NUMBER</th>
<th>PQDB</th>
<th>PQUESTAT</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>e) QUESTAT</th>
<th>VALUE (SPACE)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>f) ABQUEDB</th>
<th>PNEXT</th>
<th>CLASS</th>
<th>VALUE1 (SPACE)</th>
<th>VALUE2 (SPACE)</th>
<th>VALUE3 (SPACE)</th>
</tr>
</thead>
</table>

**QUEUES** - AIRCRAFT SERVICE QUEUE BLOCK.

**ABQUEDB** - AIRBASE QUEUE DATA BASE BLOCK.

**QUESTAT** - QUEUE STATUS BLOCK.

**ACDB** - AIRCRAFT DATA BASE BLOCK. Operational parameters of A/C type.
2) Field Specifications

a) C2 Block

UNITNUMBER  unit number
PUP        pointer to C2 block of unit's commander
PDOWN     pointer to C2 block of unit's subordinate
PSB           pointer unit's SB block
PNEXT      pointer to unit's sibling. C2 block
UNITTYPE  unit type code (220)
SIDE      unit affiliation 2 = Red (Pact)
b) **SB Block**
- ADDRESS: pointer to HEX block in which unit is located
- PC2: pointer to C2 block of the unit
- PSDB: pointer to SDB block (inactive)
- PFEL: pointer to future event list for the unit
- PACQ: pointer to acquisition devices (inactive)
- ID: unit ED number
- DATABASE: pointer to data base block (inactive)

c) **ABSTATUS Block**
- PACTAB: pointer to ABINFO block
- NOACTAB: number of ABINFO blocks in the list. Corresponds to number of aircraft on the base.
- PTR2QUES: pointer to QUEUES block
- NOACONAB: number of QUEUES blocks in the list. Corresponds to number of aircraft types on the base.
- ABDAMAGE: damage level of base. Real variable.

d) **QUEUES Block**
- NEXT: pointer to next QUEUES block in the list
- QUENUM: QUEUE number. (2 = ready queue)
- PTR NUMBER
- PQDB: pointer to ABQUEDB block
- PQUESTAT: pointer to QUESTAT block

e) **QUESTAT Block**
- VALUE: unknown. Real variable.
f) **ABQUEDB Block**
   - PNEXT: pointer to next ABQUEDB block
   - CLASS: aircraft class
   - VALUE1
   - VALUE2
   - VALUE3

g) **ABINFO Block**
   - NEXT: pointer to next ABINFO block
   - NKACTYP: aircraft type code
   - NOACOH: number of aircraft of type (NRACTYP) on hand on the base
   - PTRACDB: pointer to ACDB for the aircraft type (NRACTYP)
   - NORMRQ: number in repair queue
   - NOREARMQ: number in rearm queue
   - NOREFUELQ: number in refuel queue
   - NOLAUNCHQ: number in launch queue

h) **ACDB Block**
   - NEXT: pointer to next ACDB block in data base (not used in this context)
   - NRACTYPE: aircraft type code
   - MAXSPEED: maximum speed. Real variable
   - CRUISESPEED: cruising speed. Real variable
   - MAXALTITUDE: maximum altitude. Real variable
   - MAXCLIMBDIVE: maximum rate of altitude change. Real variable
   - FUELCONSUME: fuel consumption rate. Real variable
   - ACQRANGE: acquisition range. Real variable
   - RADARCS: radar cross-section. Real variable
   - ATTACKRADIUS: maximum attack range of aircraft. Real variable
   - MAXFUEL: maximum fuel capacity. Real variable
e. Linkages to Other Data Structures

f. Notes

QUEUE blocks and their related lists are not used at present.
8. Red Flights
   a. Data Block Index
      ACDB
      AQDB
      ARCFSAW
      ARCFTSTATUS
      C2
      COMMAND
      FLTDB
      FORMATION
      LOAD
      MUN
      ORDERES
      PAYDDBLOK
      PAYLOAD
      SB
      SDB
      SEEBUF
      WINGMAN
   b. Description
      The Red Flight Data Structures control the actions of Red Flights. In addition to the three command/control blocks C2, SB and SDB, Red Flights also use three lists. These lists include: the PERCEPTIONS LIST, the ORDERS LIST and the FLIGHT STATUS LIST.

      The PERCEPTIONS LIST is composed of a buffer and a singly-linked list of ARCFSAW blocks. These ARCFSAW blocks contain information on Blue Targets perceived by the Red Flight. This information includes the location of the Blue Targets and its perceived damage level. When the Red Flight returns to its' airbase the perceived damage to Blue Target is transferred to the Red Theater Planner's STDBLOK blocks. Thus damage perception by Red Flights are transmitted to the Red Commander for subsequent planning activities.
The ORDERS LIST is composed of two buffered lists. The first is made-up of up to six COMMAND blocks which specify the actions to be taken by the flight at various points in its mission. These COMMAND blocks determine the flight geometry and mission profile for the flight. The second list is made up of WINGMAN blocks which contain pointers to the SR blocks of other flights in the other Red Flights in the formation.

The FLIGHT STATUS LIST consists of an ARCFTSTATUS block which tracks flight status and a set of two MUNITIONS LISTS which keep track of air-to-air and air-to-ground ordnance carried by the Red flight. The ARCFTSTATUS block points to an FLTOB block which is the core of a FLIGHT DATA BASE STRUCTURE. This structure is used as a template for construction of flights of specified types. It provides the basic aircraft characteristics and initial payload levels used to create and operate the FLIGHT.

Both the FLIGHT DATA STRUCTURES and the FLIGHT DATA BASE STRUCTURES are shown in the structure diagrams.

c. Structure Overview
1) Structure Diagrams (Figure III-19 & III-20)
2) Block Definitions
   a) COMMAND/CONTROL BLOCKS:
      \[ C^2 \] - COMMAND/CONTROL BLOCK. Contains unit no, type, side, and \[ C^2 \] PTRS.
      \[ SB \] - SCOREBOARD BLOCK. Contains hex address, unit id, PTR to FEL and STAT.
      \[ SOB \] - STATUS DISPLAY BLOCK. Contains PTRS to perceptions list (SEEBUF) and orders list (ORDERS). Subordinate PTR not used by Red FLTS.
Figure III-19. Red Flight Structure Diagram
Figure III-20. Flight Data Base Structure Diagram
b) **FLIGHT STATUS LIST**

**ARCFTSTATUS** - FLIGHT STATUS BLOCK. Contains basic flight status information includes PTRS to starting, ending and next hex address in current move. Also includes PTRS to home base and TGT SB's, a status word, and current fuel, altitude, speed, and direction.

**MUN** - MUNITIONS LISTS BUFFER BLOCK. Buffer for air-to-air and air-to-ground munitions lists

**PAYLOAD** - PAYLOAD CLASS DESCRIPTION BLOCK. Basic payload parameters for class of ordinance. Contains class type, max and min amounts and max fire range. Also includes PTR to a list of attached ordnance types of the same class.

**PAYDDBLOCK** - PAYLOAD TYPE DATA BASE BLOCK. Contains PTR to next PAYDDBLOK and type of ordnance.

**LOAD** - PAYLOAD LIST BUFFER. Used to break payloads into types and keep track of ammunition load weight.

c) **PERCEPTIONS LIST**

**SEEBUF** - PERCEPTIONS LIST BUFFER BLOCK

**ARCFSAW** - AIRCRAFT PERCEPTION BLOCK. Contains information on entity perceived by a flight. Includes PTR to hex and SB of entity.
3) **FLIGHT DATA BASE**

- **FLTDB** - FLIGHT DATA BASE BLOCK.
  Contains basic flight description including No. of payloads, maximum no of A/C, minimum no. of A/C, and Multac, SPFLTC (SPACE, DISTSER(SPACE))

- **PAYLOAD** - PAYLOAD DESCRIPTION BLOCK.
  Contains payload class (NRPDGLS) max and min amount of payload, and max fire range

- **PAYLDBLOK** - PAYLOAD TYPE BLOCK. Contains payload type index

- **AQDB** - ACQUISITION DEVICE DATA BASE BLOCK.
  Contains type index and range of acquisition device.

- **ACDB** - AIRCRAFT DATA BASE BLOCK.
  Contains A/C characteristics such as speed, max range, etc.
**PROFILEBLOK** - MISSION PROFILE BLOCK. Contains flight altitude levels for three phases of mission - Alt. to corridor entrance, alt to tgt, and alt from tgt to air base.

d. **Block Specifications**

1) **Block Diagrams**

   a) $c^2$

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
<th>PUP</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSB</td>
<td>PNEXT</td>
</tr>
<tr>
<td>UNITTYPE</td>
<td>SIDE</td>
<td></td>
</tr>
</tbody>
</table>

   b) $SB$

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC</th>
<th>$c^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PACQ</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>DATABASE</td>
<td>PABSTATUS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>+ PARCFTSTAT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>+ PBOCSTAT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>+ STATUS</td>
<td></td>
</tr>
</tbody>
</table>

   c) $SDB$

<table>
<thead>
<tr>
<th>PSB</th>
<th>PSEEBUF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+ PSEE</td>
</tr>
<tr>
<td>SUBORDINATE</td>
<td>ORD</td>
</tr>
<tr>
<td></td>
<td>+ PRAID</td>
</tr>
</tbody>
</table>

   d) $SEEBUF$

   | PTRSEE | NUNITS |

   e) $ARCFSAW$

   | PNEXT | PSB |
   | ADDRESS | TYPE |
   | DAMAGE (SPACE) |

   f) $ORDERS$

   | PTRFORMS | PRACT |
### g) COMMAND
- **PNEXT**: NUMACTS
- **TMFLG**: ADDRESS
- **TIME (SPACE)**: ACTION

### h) FORMATION
- **PFORM**: NUMFLTS

### i) WINGMAN
- **PNEXT**: PSB

### j) AFCFTSTATUS
- **PFLTDB**: PNUMITIONS
- **PSTRTHX**: PENDHX
- **PNXTHX**: PAIRBASE
- **CNTRLMODE**: PAIRTGT
- **PGNOTDG**: NUMAIRCRAFT
- **NUMAG**: DUMMY

- **JAMSTAT**
- **GRNDATK**
- **AIRCOMBAT**
- **ORBITSTAT**
- **LANDING**
- **PROFILENOX**
- **ALTUDECHNG**
- **INTERCEPTSTATUS**

- **FLITELEG**
- **FUEL (SPACE)**
- **ALTITUDE (SPACE)**
- **SPEED (SPACE)**
- **DIRECTION (SPACE)**

### k) MUN
- **PAG**: NUMAG
- **PAA**: NUMAA
<table>
<thead>
<tr>
<th></th>
<th>LOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NEXT</td>
</tr>
<tr>
<td></td>
<td>AMOUNT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>FLTDB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PNXFLDB</td>
</tr>
<tr>
<td></td>
<td>PTYPLOD</td>
</tr>
<tr>
<td></td>
<td>PTYACDB</td>
</tr>
<tr>
<td></td>
<td>MAXNOAC</td>
</tr>
<tr>
<td></td>
<td>MULTAC</td>
</tr>
<tr>
<td></td>
<td>SPFLTC (SPACE)</td>
</tr>
<tr>
<td></td>
<td>DISTSEP (SPACE)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>PAYLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PNXTYPO</td>
</tr>
<tr>
<td></td>
<td>MAXAMT</td>
</tr>
<tr>
<td></td>
<td>MAXFIRE</td>
</tr>
<tr>
<td></td>
<td>MAXFIRE RANGE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>PAYLDBLOK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NEXT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>AQDB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NEXT</td>
</tr>
<tr>
<td></td>
<td>RANGE (SPACE)</td>
</tr>
<tr>
<td></td>
<td>NOUSE 1</td>
</tr>
<tr>
<td></td>
<td>NOUSE 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>PROFILESBLOK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PNXPRDB</td>
</tr>
<tr>
<td></td>
<td>ALTCREN (SPACE)</td>
</tr>
<tr>
<td></td>
<td>ALTOAB (SPACE)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>ACDB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NEXT</td>
</tr>
<tr>
<td></td>
<td>MAXSPEED (SPACE)</td>
</tr>
<tr>
<td></td>
<td>CRUISESPEED (SPACE)</td>
</tr>
<tr>
<td></td>
<td>MAXALTITUDE (SPACE)</td>
</tr>
<tr>
<td></td>
<td>MINALTITUDE (SPACE)</td>
</tr>
</tbody>
</table>

145
2) Field Definitions

a) $C^2$ Block

- UNITNUMBER - Number of the unit.
- PUP - Pointer to the $C^2$ block of the unit's commander.
- PDOWN - Pointer to the $C^2$ block of the unit's subordinate.
- PNEXT - Pointer to the $C^2$ block of the unit's sibling.
- PSB - Pointer to the SB block of the unit.
- UNITTYPE - The unit's type code (128)
- SIDE - Unit Affiliation. 2 - Red (PACT)

b) SB Block

- ADDRESS - Pointer to HEX block of the HEX in which the unit is located.
- PL2 - Pointer to $C^2$ block of the unit.
- PSDB - Pointer to the SDB block of the unit.
- PFEL - Pointer to future event list EVENT block.
- PACQ - Pointer to ACQBUF block. Used by CRC's for acquisition devices.
ID - Identification number

DATABASE - If BOC or BTRY points to ADSITEDB block.

PABSTATUS - Points to ABSTATUS block if the unit is an air base.

+PARCFTSTAT - Points to ARCFSTATUS block if the unit is a flight of aircraft.

+PODOCSTAT - Points to BOCSTAT block if the unit is a battalion operations center.

+PBTRYSTAT - Points to BTRYSTAT block if the unit is an antiaircraft battery.

+STATUS - Alternative field definition.

SOB Block

PSB - Pointer to the unit's SB block.

PSEEBUF - Pointer to SEEBUF block which is used only by flights.

+PSEE - Pointer to CRCEES block which is used by CRC units to record the blue and red flights it sees. Also used by the BIC as a pointer to its assigned target block. TARGETLISTBLOK.

SUBORDINATE - Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs. field use combination are as follows:

<table>
<thead>
<tr>
<th>UNIT TYPE</th>
<th>SUBORDINATE POINTS TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SUBLIST</td>
</tr>
<tr>
<td>BTRY</td>
<td>FIREUNIT</td>
</tr>
<tr>
<td>ORD</td>
<td>Points to ORDERS block if the unit is a flight.</td>
</tr>
</tbody>
</table>
+PRAID - Points to the RAIDBLOK if the unit is the Red Theater Commander.

d) SEEBUF BLOCK
PTRSEE - Pointer to ARCFSAW block. (first in list)
NUNITS - Number of ARCFSAW blocks in the list. Corresponds to number of targets perceived.

e) ARCFSAW BLOCK
PNEXT - Pointer to next ARCFSAW block in the list.
PSB - Pointer to SB block of target.
ADDRESS - Pointer to HEX block in which target is located.
TYPE - Unit type code of target.
DAMAGE - Perceived damage level of the target. Real variable.

f) ORDERS BLOCK
PTRFORMS - Pointer to FORMATION block.
PTRACT - Pointer to COMMAND block.

g) COMMAND BLOCK
PNEXT - Pointer to next command block
NUMACTS - Number of the command in the list.
TMFLG - Time flag. If 1 at time is associated with the command.
ADDRESS - Pointer to HEX block in which the command is to be carried out.
TIME - Time the command is to be performed.
ACTION - Command or action code.

h) FORMATION BLOCK
PFORM - Pointer to first WINGMAN block in the list.
NUMFLTS - Number of WINGMAN blocks in the list. Corresponds to number of flights in the formation.

i) WINGMAN BLOCK
PNEXT - Pointer to next WINGMAN block in the list.
PSB - Pointer to the SB block of the flight.

j) ARCFTSTATUS BLOCK
PFLTDB - Pointer to FLTDB block
PNUMITIONS - Pointer to NUM block
PSTRTHX - Pointer to HEX block in which current move begins
PENDHX - Pointer to HEX block in which current move ends
PNXTHX - Pointer to next HEX block
PAIRBASE - Pointer to SB block of flight's home air base.
CNTRLMODE -
PAIRTGT - Pointer to SB block of airborne target.
PGNDTGT - Pointer to SB block of ground target.
NVMAIRCRAFT - Number of aircraft in the flight.
FLITELEG
INTERCEPTSTATUS
ALTUDCHNG
PROFILENDX
LANDING
ORBITSTAT
AIRCOMBAT
JAMSTAT
FUEL - Current fuel level. Real variable
ALTITUDE - Altitude in meters. Real variable
SPEED - Speed. Real variable
DIRECTION - Direction. Real variable

k) MUM BLOCK
PAG - Pointer to ground attack LOAD block
NUMAG - Number of LOAD blocks in ground attack munitions list
PAA - Pointer to air attack LOAD block
NUMAA - Number of LOAD blocks in air attack munitions list

l) LOAD BLOCK
PNEXT - Pointer to next LOAD block in the list
TYPE - Munitions class
AMOUNT - Amount of munitions in tons
PORDB - Pointer to PAYLOAD block

m) FLTDB BLOCK
PBXFLDB - Pointer to next FLTDB block
NRFLITE - Unique flight specification number
PTYPLDS - Pointer to PAYLOAD data block
Payload class 6002
NOPYLDS - Number of PAYLOAD data blocks
PTYAQDB - Pointer to ACQUISITION data block (AQDB), Class 6007)
PTACDB - Pointer to aircraft specification data block (ACDB, CLASS 6003)
MAXNOAC - Maximum number of aircraft in flight
MINNOAC - Minimum number of aircraft in flight
MULTAC - Multiples of aircraft required for flight
PROFILE - Pointer to profile specification data block (Profiled blok, class 6005)
SPFLTC(SPACE) Flight crusing speed in meters/second (real)
QISTSEP(SPACE) Flight separation distance in meters (real)
n) PAYLOAD BLOCK
   PNXTYPD - Pointer to next payload block
   NRPOCLS - Payload type, must be 3 or 4
               3 = air-to-ground
               4 = air-to-air
   MAXAMT - Maximum number of loads of this payload
   MINAMT - Minimum number of loads of this payload
   MAXFIRERANGE Future use by an enhancement for maximum fire range for engagements greater than one hex
   PAYLDBB - Pointer to payload ID data block (PAYLDBBLOK)
o) PAYLDBBLOK BLOCK
   NEXT - Pointer to next ID block
   TYPEINDEX - Payload ID, unique within each payload type
p) AQDB BLOCK
   NEXT - Pointer to next AQDB block
   NRAQTYP - Unit type
   RANGE(SPACE) Acquisition range in meters (real)
   NOUSE1 - Not used
   NOUSE2 - Not used
q) PROFILEDBLOK BLOCK
   PNXPROB - Pointer to next PROFILEDBLOK
   NRPROFL - Profile identification number, must be unique within the 6005 class
ALTCREN - Altitude of first leg in meters (real)
ALTOTGT - Altitude of second leg in meters (real)
ALTOAB - Altitude of third leg in meters (real)

ACDB BLOCK
NEXT - Pointer to next ACDB block
NRACTYPE - Aircraft type number
MAXSPEED - Maximum speed in meters/second (real)
CRUISESPEED - Cruising speed in meters/second (real)
MAXALITUDE - Maximum altitude in meters (real)
MINALITUDE - Minimum altitude in meters (real)
MAXCLIMBDIVE - Maximum climb/dive rate in meters/seconds (real)
FUELCONSUME - Fuel consumption rate in hexes/second (real)
ACQRANGE - Acquisition range in meters (real)
RADARCS - Radar cross sections in hexes (real)
ATTACKRADIUS - Attack radius in meters (real)
MAXFUEL - Maximum fuel load in hexes (real)

e. Linkages to Other Data Structures
f. Notes
G. **BLUE STRUCTURES**

In contrast to Red Structures which are dominated by threat planning related structures, Blue Structures are all related to some type of combat entity or player. Seven of those player types are connected to the BLUE C² tree, while the eight is connected to the passive target list.
1. ALLIED TACTICAL AIR FORCE (ATAF)
   a. DATA BLOCK INDEX
      C2
      SB
      SND
   b. DESCRIPTION
      The ALLIED TACTICAL AIR FORCE structure resides at the top of the blue C2 TREE (see E.2.f). It is currently used only to maintain the consistency of the blue command/control structure and does not initiate actions in the course of the simulation.
   c. STRUCTURE OVERVIEW
      1) STRUCTURE DIAGRAM (Figure III-21)

      ![Structure Diagram]

      Figure III-21. Allied Tactical Air Force Structure Diagram

      2) BLOCK DEFINITIONS
         PLBUFFER - C2 TREE BUFFER BLOCK. Contains pointer to the tree and the side of the tree
         C2 - COMMAND/CONTROL BLOCK. Contains tree pointers, unit number, pointer to scoreboard (SB), unit type code and side
         SB - SCOREBOARD BLOCK. Contains pointers to C2 and HEX blocks, and the status display board (SDB). Also contains pointers
to acquisition devices, various unit status blocks and the future event list. Use varied with unit type.

**SDB**

**STATUS DISPLAY BOARD BLOCK.** Contains subordinate, acquisition and order pointers. Use varies with unit type.

d. **BLOCK SPECIFICATIONS**

1) **PLBUFFER**

<table>
<thead>
<tr>
<th>PTRPL</th>
<th>VARIWORD</th>
</tr>
</thead>
</table>

2) **C2**

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUP</td>
</tr>
<tr>
<td>PSB</td>
</tr>
<tr>
<td>UNITYPE</td>
</tr>
</tbody>
</table>

**NOTE:** + ALTERNATE DEFINITION OF FIELD

3) **SB**

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSDB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
<tr>
<td>DATABASE</td>
<td>PABSTATUS</td>
</tr>
</tbody>
</table>

+ PARCFSTAT
+ PBOCSTAT
+ PBTRYSTAT
+ STATUS

4) **SDB**

<table>
<thead>
<tr>
<th>PSB</th>
<th>PSEEBUF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSEE</td>
</tr>
<tr>
<td>SUBORDINATE</td>
<td>ORD</td>
</tr>
</tbody>
</table>

+ PRAID
e. FIELD DEFINITIONS

1) PLBUFFER BLOCK
   PTRPL - Pointer to C2 Block
   VARWORD - Side indicator for tree
   1 = BLUE (NATO)
   2 = RED (PACT)

2) C2 BLOCK
   - Number of the unit. If negative, the unit is a passive target
   PUP - Pointer to the C2 block of the unit's commander
   PDOWN - Pointer to the C2 block of the unit's sibling
   PSB - Pointer to the SB block of the unit
   UNITTYPE - The unit's type code (see subsection 6).
   SIDE - Unit affiliation
   1 = BLUE (NATO)
   2 = RED (PACT)

3) SB BLOCK
   ADDRESS - Pointer to HEX block of the hex in which the unit is located
   PC2 - Pointer to C2 block of the unit
   PSDB - Pointer to the SDB block of the unit
   PFEL - Pointer to future event list EVENT block
   PACQ - Pointer to ACQBUF block. Used by CRC's for acquisition devices
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Identification number</td>
</tr>
<tr>
<td>DATABASE</td>
<td>If BOC or BTRY points to ADSITEDB block</td>
</tr>
<tr>
<td>PABSTATUS</td>
<td>Points to ASTATUS block if the unit is an airbase</td>
</tr>
<tr>
<td>+PARCFTSTAT</td>
<td>Points to ARCFTSTATUS block if the unit is a flight or aircraft.</td>
</tr>
<tr>
<td>+PBOCSTAT</td>
<td>Points to BOCSTAT block if the unit is a battalion operations center</td>
</tr>
<tr>
<td>TPBTRYSTAT</td>
<td>Points to BTRYSTAT block if the unit is an anti-aircraft battery</td>
</tr>
<tr>
<td>+STATUS</td>
<td>alternative field definition</td>
</tr>
<tr>
<td>4) SDB BLOCK</td>
<td>Pointer to the unit's SB block</td>
</tr>
<tr>
<td>PSB</td>
<td>Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights</td>
</tr>
<tr>
<td>PSEEBUF</td>
<td>Pointer to CRCEES block which is used by CRC units to record the blue and red flights it sees. Also used by the RTC to point to its assigned target list.</td>
</tr>
<tr>
<td>+PSEE</td>
<td>TARGET_LISTBOOK</td>
</tr>
</tbody>
</table>
SUBORDINATE - Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs field use combinations are as follows:

<table>
<thead>
<tr>
<th>UNIT_TYPE</th>
<th>SUBORDINATE POINTS TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SVBUST</td>
</tr>
<tr>
<td>BTRY</td>
<td>FIREUNIT</td>
</tr>
</tbody>
</table>

ORD - Points to ORDERS block if the unit is a flight

+PRAID - Points to the RAIDBLOK if the unit is the Red theater commander
2. SECTOR OPERATIONS CENTERS (SOC)
   a. DATA BLOCK INDEX
      C2
      SB
      SDB
   b. DESCRIPTION
      The SECTOR OPERATIONS CENTER structures reside on the Blue C2
tree on the Blue C2 tree on the level below the ATAF. Like the ATAF, they
are currently used only to maintain the consistency of the Blue command/control
structure. SOC's do not initiate actions in the course of the simulation.
   c. STRUCTURE OVERVIEW
      1) STRUCTURE DIAGRAM (Figure III-22)

      Figure III-22. Sector Operations Center Structure Diagram

      2) BLOCK DEFINITIONS
         PLBUFFER - C2 Tree Buffer Block. Contains
         pointer to the tree and to the
         side of the tree
         C2 - Command/Control Block. Contains
         tree pointers, unit number, pointer
         to scoreboard (SB). unit type code
         and side
         SB - Scoreboard Block. Contains pointers
         to C2 and HEX blocks, and the status
         display board (SDB). Also contains
         pointers to acquisition devices,
         various unit status blocks and the
         future EVENT list. Use varies with
**unit type**

SDR

*Status Display Board Block.* Contains subordinate, acquisition and order pointers. Use varies with unit type.

d. **BLOCK SPECIFICATIONS**

1) **BLOCK DIAGRAMS**

a) **C2**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNITNUMBER</td>
<td></td>
</tr>
<tr>
<td>PUP</td>
<td>PDOWN</td>
</tr>
<tr>
<td>PSB</td>
<td>PNEXT</td>
</tr>
<tr>
<td>UNITYYPE</td>
<td>SIDE</td>
</tr>
</tbody>
</table>

**NOTE:** + ALTERNATE DEFINITION OF FIELD

b) **SB**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>PC2</td>
</tr>
<tr>
<td>PSDB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
<tr>
<td>DATABASE</td>
<td>PABSTATUS</td>
</tr>
<tr>
<td></td>
<td>+ PARCFTSTAT</td>
</tr>
<tr>
<td></td>
<td>+ PBOCSTAT</td>
</tr>
<tr>
<td></td>
<td>+ PBTRYSTAT</td>
</tr>
<tr>
<td></td>
<td>+ STATUS</td>
</tr>
</tbody>
</table>

c) **SDB**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB</td>
<td>PSEEBUF</td>
</tr>
<tr>
<td></td>
<td>+ PSEE</td>
</tr>
<tr>
<td>SUBORDINATE</td>
<td>ORD</td>
</tr>
<tr>
<td></td>
<td>+ PRAID</td>
</tr>
</tbody>
</table>
2) **FIELD DEFINITIONS**

a) **C2 BLOCK**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNITNUMBER</td>
<td>Number of the unit. If negative, the unit is a passive target</td>
</tr>
<tr>
<td>PVP</td>
<td>Pointer to the C2 block of the unit's commander</td>
</tr>
<tr>
<td>PDOWN</td>
<td>Pointer to the C2 block of the unit's subordinate</td>
</tr>
<tr>
<td>PNEXT</td>
<td>Pointer to the C2 block of the unit's sibling</td>
</tr>
<tr>
<td>PSB</td>
<td>Pointer to the SB block of the unit</td>
</tr>
<tr>
<td>UNITYPE</td>
<td>The unit's type code (see subsection 5))</td>
</tr>
<tr>
<td>SIDE</td>
<td>Unit affiliation&lt;br&gt;1 = BLUE (NATO)&lt;br&gt;2 = RED (PACT)</td>
</tr>
</tbody>
</table>

b) **SB BLOCK**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>Pointer to HEX block of the hex in which the unit is located</td>
</tr>
<tr>
<td>PC2</td>
<td>Pointer to C2 block of the unit</td>
</tr>
<tr>
<td>PSDB</td>
<td>Pointer to the SDB block of the unit</td>
</tr>
<tr>
<td>PFEL</td>
<td>Pointer to future event list EVENT block</td>
</tr>
<tr>
<td>PACQ</td>
<td>Pointer to ACQBUF block. Used by SRC's for acquisition devices.</td>
</tr>
<tr>
<td>ID</td>
<td>Identification number</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DATABASE</td>
<td>If BOC or BTRY points to ADSITEDB block</td>
</tr>
<tr>
<td>PABSTATUS</td>
<td>Points to ABSTATUS block if the unit is an airbase</td>
</tr>
<tr>
<td>+PARCFTSTAT</td>
<td>Points to ARCFTSTATUS block if the unit is a flight of aircraft</td>
</tr>
<tr>
<td>+PBOCSTAT</td>
<td>Points to BOCSTAT block if the unit is a battalion operation center</td>
</tr>
<tr>
<td>+PBTRYS</td>
<td>Points to BTRYS block if the unit is an antiaircraft battery</td>
</tr>
<tr>
<td>+STATUS</td>
<td>Alternative field definition</td>
</tr>
<tr>
<td>c) SDB BLOCK</td>
<td></td>
</tr>
<tr>
<td>PSB</td>
<td>Pointer to the unit's SB block</td>
</tr>
<tr>
<td>PSEEBUF</td>
<td>Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights.</td>
</tr>
<tr>
<td>+PSEE</td>
<td>Pointer to CRCEES block which is used by CRC units to record the blue and red flights it sees. Also used by the RTC to point to its assigned target list.</td>
</tr>
</tbody>
</table>
SUBORDINATE - Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs yield use combinations are as follows:

<table>
<thead>
<tr>
<th>UNIT TYPE</th>
<th>SUBORDINATE POINTS TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SUBLIST</td>
</tr>
<tr>
<td>BTRY</td>
<td>FIREUNIT</td>
</tr>
</tbody>
</table>

ORD - Points to ODERS block if the unit is a flight

+ PRAID - Points to the RAIDBLOK if the unit is the Red theater commander

e. LINEAGES TO OTHER DATA STRUCTURES
   Pointer to HEX block in which unit is located

f. NOTES
3. COMBAT REPORTING CENTERS (CRC)
   a. DATA BLOCK INDEX
      ACQBUF
      ACQDEVICE
      C2
      CACSEEBLUE
      CRCSEERED
      CRCSEES
      CRCSUBORD
      SB
      SDB
      SEER
      SUB
      SUBTYPE
   b. Description
      The COMBAT REPORTING CENTERS are the highest ranking active players on the Blue side. In addition to the basic command control structures (C², SB and SDB), CRC's possess three unique lists - the PERCEPTIONS LIST, the SUBORDINATE LIST, and the ACQUISITION LIST.
      The ACQUISITION LIST is used to keep track of acquisition devices belonging to the CRC. The SUBORDINATE LIST is used to keep track of the CRC's subordinate blue units (including FLIGHTS). These lists are stratified by device and subordinate type respectively.
      The PERCEPTION LIST consists of two branches. One keeps track of Red Units and the Blue Units perceiving them. The other keeps track of Blue Units perceived by the CRC directly. It is important to note that the CRC may perceive enemy units through its subordinates. The overall configuration of the CRC structure is shown in the structure diagram.
   c. Structure Overview
      1) Structure Diagram (Figure III-23)
Figure III-23. Combat Reporting Center Structure Diagram
2) **Block Definitions**

- **C**\(^2\)** - COMMAND/CONTROL BLOCK. Contains tree pointers, unit number, pointer to scoreboard (SB), unit type code and side.

- **SB** - SCOREBOARD BLOCK. Contains pointers to C\(^2\) and HEX blocks, and the status display board (SDB). Also contains pointers to acquisition devices various unit status blocks and the future event list. Use varies with unit type.

- **SDB** - STATUS DISPLAY BOARD BLOCK. Contains subordinate, acquisition and order pointers. Use varies with unit type.

- **ACQBUF** - ACQUISITION DEVICE LIST BUFFER. Contains a pointer to the device list and the number of devices (ACQDEVICE BLOCKS) in the list.

- **ACQDEVICE** - ACQUISITION DEVICE DESCRIPTION BLOCK. Contains device code, operating status and a pointer to the next ACQDEVICE block in the list.

- **AQDB** - ACQUISITION DEVICE DATA BASE BLOCK. Basic data on specific acquisition device.

- **SUB** - SUBORDINATE LIST BUFFER. Contains a pointer to the subordinate list and the number of subordinate types (SUBTYPE BLOCKS) in the list.
### SUBTYPE
- **Subordinate Type Block.** Contains a subordinate type code, the number of subordinates of that type, a pointer to the next SUBTYPE and a pointer to a list of specific subordinate unit blocks (CRCSUBORD).

### CRCSUBORD
- **CRC Subordinate Block.** Contains unit ID of the subordinate, a pointer to its SB block and HEX block, and a pointer to the next CRCSUBORD block in the list.

### CRCSEES
- **CRC Perceptions List Buffer Block.** Contains pointers to Red and Blue branches and the number of units (CRCSEERED and CRCSEEBLUE BLOCKS) in each branch.

### CRCSEERED
- **Red Unit Perceived Block.** Contains a description of Red Unit perceived. Includes pointers to its' SB and HEX blocks as well as its' direction of movement. Also contains a pointer to a list of Blue units which can see it.

### SEER
- **Blue Perception Unit Block.** Contains unit ID of the Blue Unit perceiving a Red Unit.

### CRCSEEBLUE
- **Blue Unit Perceived Block.** Contains the unit ID, SB and HEX block pointers of a Blue Unit perceived by the CRC.
### d. Block Specifications

1) Block Diagrams

#### a) \( C^2 \)

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
<th>PUP</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB</td>
<td>PNEXT</td>
<td></td>
</tr>
<tr>
<td>UNITTYPE</td>
<td>SIDE</td>
<td></td>
</tr>
</tbody>
</table>

Note: + Alternate Definition of field.

#### b) SB

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSDB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
</tbody>
</table>

**DATABASE**

- + PARCFSTAT
- + PBOCSTAT
- + PBTRYSTAT
- + STATUS

#### c) SDB

<table>
<thead>
<tr>
<th>PSB</th>
<th>PSEEBUF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+ PSEE</td>
</tr>
</tbody>
</table>

**SUBORDINATE**

- + PRAID

#### d) ACQBUF

| PTRACQ | NUMDEV |

#### e) ACQDEVICE

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKING</td>
<td>PACQDB</td>
</tr>
<tr>
<td>JAM</td>
<td>LEVEL</td>
</tr>
</tbody>
</table>

#### f) AQDB

<table>
<thead>
<tr>
<th>NEXT</th>
<th>NRAQTYP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANGE(SPACE)</td>
<td></td>
</tr>
<tr>
<td>NOUSE1</td>
<td></td>
</tr>
<tr>
<td>NOUSE2</td>
<td></td>
</tr>
</tbody>
</table>

168
2) FIELD DEFINITIONS

a) C² BLOCK

| UNITNUMBER | - | Number of the unit. If negative the unit is a passing target.
| PVP        | - | Pointer to the C² block of the unit's commander.
| PDOWN      | - | Pointer to the C² block of the unit's subordinate. |
PNEXT - Pointer to the $C^2$ block of the unit's commander.

PSB - Pointer to the SB block of the unit.

UNITTYPE - The unit's type code (see subsection f).

SIDE - Unit affiliation.
1 = Blue (NATO)
2 = Red (PACT)

b) SB BLOCK
ADDRESS - Pointer to HEX block of the HEX in which the unit is located.
PC2 - Pointer to $C^2$ block of the unit.
PSDB - Pointer to the SDB block of the unit.
PFEL - Pointer to future event list EVENT BLOCK.
PACQ - Pointer to ACQBUF BLOCK. Used by CRC's for acquisition devices.
ID - Identification number
DATABASE - If BOC or BTRY points to ADSITEDB block.
PABSTATUS - Points to ABSTATUS block if the unit is an airbase.
+PARCFTSTAT - Points to ARCFSTATUS block if the unit is a flight of aircraft.
+PBOCSTAT - Points to BOCSTAT block if the unit is a battalion operations center.
+PBTRYSTAT - Points to BTRYSTAT block if the unit is an anti-aircraft battery.
+STATUS - Alternative field definition
+PBTRYSTAT - Points to BOCSTAT block if the unit is an anti-aircraft
+STATUS - Alternative field definition.

c) **SOB BLOCK**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB</td>
<td>Pointer to the unit's SB block.</td>
</tr>
<tr>
<td>PSEEBUF</td>
<td>Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights.</td>
</tr>
<tr>
<td>PSEE</td>
<td>Pointer to CRCEES block which is used by CRC units to record the Blue and Red flights it sees. Also used by RTC to point to its' assigned targets list</td>
</tr>
</tbody>
</table>

**SUBORDINATE** - Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs. field use combinations are as follows:

<table>
<thead>
<tr>
<th>UNIT TYPE</th>
<th>SUBORDINATE POINTS TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SUBLIST</td>
</tr>
<tr>
<td>BTRY</td>
<td>FIREUNIT</td>
</tr>
<tr>
<td>ORD</td>
<td>Points to ORDERS block if the unit is a flight</td>
</tr>
<tr>
<td>+PRAID</td>
<td>Points to RAIDBLOK if the unit is the Red Theater Commander.</td>
</tr>
</tbody>
</table>
d) **ACQBUF BLOCK**
   - **PTRACQ**: Pointer to ACQDEVICE block.
   - **NUMDEV**: Number of ACQDEVICE blocks in the list.

e) **ACQDEVICE BLOCK**
   - **PNEXT**: Pointer to the next ACQDEVICE block in the list.
   - **TYPE**: Acquisition device type code
   - **WORKING**: 
   - **PACQDB**: Pointer to ACDB block.
   - **JAM**: 
   - **LEVEL**: 

f) **AQDB BLOCK**
   - **NEXT**: Pointer to next AQDB block in data base
   - **NRAQTYP**: Acquisition device type code
   - **RANGE**: Range of device. Real variable
   - **NOUSE1**: 
   - **NOUSE2**: 

g) **SUB BLOCK**
   - **PSUB**: Pointer to SUBTYPE block
   - **NUMBER**: Number of SUBTYPE blocks in the list

h) **SUBTYPE BLOCK**
   - **PNEXT**: Pointer to next SUBTYPE block
   - **TYPE**: Subordinate type code
   - **PTRSUB**: Pointer to CRCSUBORD block
   - **NUMBER**: Number of CRCSUBORD blocks in the list

i) **CRCSUBORD BLOCK**
   - **PNEXT**: Pointer to next CRCSUBORD block
   - **ID**: Unit ID
PSB - Pointer to Unit's SB block
ADDRESS - Pointer to Unit's HEX block.
WORD1 -
WORD2 -

j) CRCEES BLOCK
REDSEE - Pointer to CRCSEERED block
NUMRED - Number of CRCSEERED blocks in the list.
BLUESSEE - Pointer to CRCSEEBLUE blocks
NUMBLUE - Number of CRCSEEBLUE blocks in the list.

k) CRCSEERED BLOCK
PNEXT - Pointer to next CRCSEERED block
ID - Unit ID.
PSB - Pointer to Red Unit's SB block
ADDRESS - Pointer to Red Unit's HEX block
RPT -
HUNTER -
DIRECTION - Direction of travel, real variable
PNX - Pointer to SEER block
NUMSEE - Number of SEER blocks in the list

l) SEER BLOCK
PNEXT - Pointer to next SEER block
ID - Unit ID of Blue perceiver

m) CRCSEEBLUE BLOCK
PNEXT - Pointer to next CRCSEEBLUE block
ID - Unit ID
PSB - Pointer to Unit's SB block
ADDRESS - Pointer to Unit's HEX block
e. LINKAGES TO OTHER DATA STRUCTURES

The CRC STRUCTURE is linked to both the HEX ADDRESS TREE and the $C^2$ TREES of both sides.

f. NOTES
4. BLUE AIRBASES (B-AB)

a. Data Block Index

ABINFO
ABQUEDB
ABSTATUS
ACDB
C²
QUEUES
QUESTAT
SB

b. Description

BLUE AIRBASE STRUCTURES are designed to keep track of the number and type of aircraft on the AIRBASE as well as the status of each aircraft type in terms launch capability. The overall configuration of these structure is shown in the structure diagram.

c. Structure Overview

1) Structure Diagram (Figure III-24)

2) Block Definitions

C² - COMMAND CONTROL BLOCK. Contains unit no., type and side, and C² PTRS.

SB - SCOREBOARD BLOCK. Contains HEX address PTR, Unit ID, PTR to future event list for the unit, and ABSTATUS PTR.

SDB - STATUS DISPLAY BLOCK. Not used by airbase.

ABSTATUS - AIRBASE STATUS BLOCK. Contains PTRS to the airbase information list (ABINFO) and the QUEUES list for each aircraft type on base.

ABINFO - AIRCRAFT ON BASE INFORMATION BLOCK. Contains aircraft type, no on hand and no. in each service queue, and a PTR to A/C type's ACDB.
Figure III-24. Blue Airbase Structure Diagram
d. Block Specification

1) Block Diagrams

a) C2

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
<th>PUP</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSB</td>
<td>PNEXT</td>
</tr>
<tr>
<td></td>
<td>UNITTYPE</td>
<td>SIDE</td>
</tr>
</tbody>
</table>

b) SB

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSDB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
<tr>
<td>DATABASE</td>
<td>PABSTATUS</td>
</tr>
</tbody>
</table>

c) ABSTATUS

<table>
<thead>
<tr>
<th>PACTAB</th>
<th>NOACTAB</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTR2QUES</td>
<td>NOACONAB</td>
</tr>
<tr>
<td>ABDAMAGE(SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

d) QUEUES

<table>
<thead>
<tr>
<th>NEXT</th>
<th>QUENUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTR</td>
<td>NUMBER</td>
</tr>
<tr>
<td>PQDB</td>
<td>PQQUESTAT</td>
</tr>
</tbody>
</table>

e) QUESTAT

| VALUE(SPACE) |

f) ABQUEDB

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE1 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>VALUE2 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>VALUE3 (APCE)</td>
<td></td>
</tr>
</tbody>
</table>
2) Field Specifications

a) $C^2$ BLOCK

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNITNUMBER</td>
<td>Unit number</td>
</tr>
<tr>
<td>PUP</td>
<td>Pointer to $C^2$ block of unit's commander</td>
</tr>
<tr>
<td>PDOWN</td>
<td>Pointer to $C^2$ block of unit's subordinate</td>
</tr>
<tr>
<td>PSB</td>
<td>Pointer Unit's SB block</td>
</tr>
<tr>
<td>PNEXT</td>
<td>Pointer to Unit's sibling. $C^2$ block</td>
</tr>
<tr>
<td>UNITTYPE</td>
<td>Unit type code (220)</td>
</tr>
<tr>
<td>SIDE</td>
<td>Unit affiliation</td>
</tr>
</tbody>
</table>

$1 = \text{Blue (NATO)}$

$2 = \text{Red (PACT)}$
b) **SB BLOCK**

- **ADDRESS**
  - Pointer to HEX block in which unit is located.
- **PC2**
  - Pointer to C² block of the unit.
- **PSOB**
  - Pointer to SOB block (Inactive)
- **PFEL**
  - Pointer to future event list for the unit.
- **PACQ**
  - Pointer to acquisition devices (inactive)
- **ID**
  - Unit ID number
- **DATABASE**
  - Pointer to database block (inactive)


c) **ABSTATUS BLOCK**

- **PACTAB**
  - Pointer to ABINFO block.
- **NOACTAB**
  - Number of ABINFO blocks in the list corresponds to number of aircraft on the base.
- **PTR2QUES**
  - Pointer to QUEUES BLOCK
- **NOACONAB**
  - Number of QUEUES blocks in the list. Corresponds to number of aircraft types on the base.
- **ABDAMAGE**
  - Damage level of base. Real variable.

d) **QUEUES BLOCK**

- **NEXT**
  - Pointer to next QUEUES block in the list.
- **QUENUM**
  - QUEUE number (2 = ready queue)
- **PTR**
  - NUMBER
- **PQDB**
  - Pointer to ABQUEDB block
- **QPESTAT**
  - Pointer to QUESTAT block
e) **QUESTAT BLOCK**
   VALUE - Unknown, real variable

f) **ABQUEDB BLOCK**
   PNEXT - Pointer to next ABQUEDB block
   CLASS - Aircraft class
   VALUE1 -
   VALUE2 -
   VALUE3 -

g) **ABINFO BLOCK**
   NEXT - Pointer to next ABINFO block
   NRACTYP - Aircraft type code
   NOACOH - Number of aircraft of type (NRACTYP) on hand on the base.
   PTRACDB - Pointer to ACDB for the aircraft type (NRACTYP)
   NORMRQ - Number in repair queue
   NOREARMQ - Number in rearm queue
   NOLAUNCHQ - Number in launch queue

h) **ACDB BLOCK**
   NEXT - Pointer to next ACDB block in data base (not used in this context)
   NRACTYPE - Aircraft type code
   MAXSPEED - Maximum speed. Real variable
   CRUISESPEED - Cruising speed. Real variable
   MAXALTITUDE - Maximum altitude. Real variable
   MAXCLIMBDIVE - Maximum rate of altitude change. Real variable.
   FUELCONSUME - Fuel consumption rate. Real variable.
   ACQRANGE - Acquisition range. Real variable.
RADARCS - Radar cross-section. Real variable.
ATTACKRADIUS - Maximum attack range of aircraft. Real variable.
MAXFUEL - Maximum fuel capacity. Real variable.

e. Linkages to Other Data Structures
f. Notes
   QUEUES blocks and their related lists are not used at present.
5. **Blue Flights (B-FLT)**

a. **Data Block Index**
   - ACDB
   - AQDB
   - ARCFSAW
   - ARCFSTATUS
   - C²
   - COMMAND
   - FLTDB
   - FORMATION
   - LOAD
   - MUN
   - ORDERES
   - PAYDDBLOK
   - PAYLOAD
   - SB
   - SEEBUF
   - WINGMAN

b. **Description**

   The BLUE FLIGHT DATA STRUCTURES control the actions of blue flights. In addition to the three command/control blocks C², SB and SDB, BLUE FLIGHTS also use three lists. These lists include: the PERCEPTIONS LIST, the ORDERS LIST and the FLIGHT STATUS LIST.

   The PERCEPTIONS LIST is composed of a buffer and a singly-linked list of ARCFSAW blocks. These ARCFSAW blocks contain information on Red flights perceived by the Blue FLIGHT. This information includes the location of the Red targets. When the Blue FLIGHT returns to its' airbase the perceived damage to Red Flights is transferred to the Blue CRC.

   The ORDERS LIST is composed of two buffered lists. The first is made-up of up to six COMMAND blocks which specify the actions to be taken by the flight at various points in its' mission. These COMMAND blocks determine the flight geometry and mission profile for the flight. The second list is made up to WINGMAN blocks which contain pointers. To the SB blocks of other flights in the other Blue FLIGHTS in the formation.
The FLIGHT STATUS LIST consists of an ARCFTSTATUS block which tracks flight status and a set of two NUMITIONS LISTS which keep track of air-to-air and air-to-ground ordinance carried by the Red Flight. The ARCFTSTATUS block points to an FLTOB block which is the core of a FLIGHT DATA BASE STRUCTURE. This structure is used as a template for construction of flights of specified types. It provides the basic aircraft characteristics and initial payload levels used to create and operate the flight.

Both the FLIGHT DATA STRUCTURES and the FLIGHT DATA BASE STRUCTURES are shown in the structure diagrams.

c. Structure Overview
   1) Structure Diagrams (Figures III-25 & III-26)
Figure III-25. Blue Flight Structure Diagram
Figure III-26. Flight Data Base Structure Diagram
2) **Block Definitions**

**COMMAND/CONTROL BLOCKS**

- **C**\(^2\) COMMAND/CONTROL BLOCK. Contains Unit No., type, side, and C\(^2\) PTRS.
- **SB** SCOREBOARD BLOCK. Contains HEX address, unit ID, PTR to FEL and STAT
- **SDB** STATUS DISPLAY BLOCK. Contains PTRS to perceptions list (SEEBUF) and orders list (ORDERs). Subordinate PTR not used by Red FLTS.

**FLIGHT STATUS LIST**

- **ARCFCTSTATUS** FLIGHT STATUS BLOCK. Contains basic flight status information includes PTRS to startin, ending and next HEX address in current move. Also includes PTRS to homebase and TGT SB's a status word, and current fuel, altitude, speed, and direction.
- **MUN** MUNITIONS LIST BUFFER BLOCK. Buffer for air to air and air to ground munitions lists.
- **PAYLOAD** PAYLOAD CLASS DESCRIPTION BLOCKS. Basic payload parameters for class of ordinance. Contains class type, max and min amounts and max fire range. Also includes PTR to a list of attached ordinance types of the same class.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAYDDBLOK</td>
<td>PAYLOAD TYPE DATA BASE BLOCK. Contains PTR to next PAYDDBLOK and type of ordinance.</td>
</tr>
<tr>
<td>LOAD</td>
<td>PAYLOAD LIST BUFFER. Used to break payloads into types and keep track of ammunition load weight.</td>
</tr>
<tr>
<td>PERCEPTIONS LIST</td>
<td>PERCEPTIONS LIST BUFFER BLOCK</td>
</tr>
<tr>
<td>SEEBUF</td>
<td>AIRCRAFT PERCEPTION BLOCK. Contains information on entity perceived by a flight. Includes PTR to HEX and SB of entity.</td>
</tr>
<tr>
<td>ARCFSAW</td>
<td></td>
</tr>
<tr>
<td>ORDERS LIST</td>
<td>ORDEMRS LIST BUFFER BLOCK. Contains no. of orders remaining.</td>
</tr>
<tr>
<td>ORDERS</td>
<td></td>
</tr>
<tr>
<td>COMMAND</td>
<td>COMMAND DESCRIPTION BLOCK. Describes command to be followed by flight at specified address. Up to six in the list.</td>
</tr>
<tr>
<td>FORMATION</td>
<td>FORMATION BUFFER BLOCK. Contains number of flts in the formation to which the flight belongs.</td>
</tr>
<tr>
<td>WINGMAN</td>
<td>WINGMAN LIST BLOCK. Contains PTR to other flts in the formation.</td>
</tr>
</tbody>
</table>
FLIGHT DATA BASE

FLTDB - FLIGHT DATA BASE BLOCK. Contains basic flight description including no. of payloads, maximum no. of A/C, minimum No. of A/C, and Multic SPFLTC(SPACE, DISTSER(SPACE)

PAYLOAD - PAYLOAD DESCRIPTION BLOCK. Contains payload class (NRPDCLS) max and min amount of payload, and max fire range

PAYLDBLOK - PAYLOAD TYPE BLOCK. Contains payload type index

AQDB - ACQUISITION DEVICE DATA BASE BLOCK. Contains type index and range of acquisition device.

ACDB - AIRCRAFT DATA BASE BLOCK. Contains A/C characteristics such as speed, max range etc.

PROFILEDBLOK - MISSION PROFILE BLOCK. Contains flight altitude levels for three phases of mission - ALT to corridor entrance, ALT to TGT, and ALT from TGT to Airbase.

1) Block Diagrams
a) C

| UNITNUMBER | C
|------------|----
| PUP        | PDOWN
| PSB        | PNEXT
| UNITTYPE   | SIDE

188
### b) SB

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>PC2</td>
</tr>
<tr>
<td>PSDB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
<tr>
<td>DATABASE</td>
<td>PABSTATUS</td>
</tr>
<tr>
<td>+</td>
<td>PARCFTSTAT</td>
</tr>
<tr>
<td>+</td>
<td>PBOCSTAT</td>
</tr>
<tr>
<td>+</td>
<td>PBTRYSTAT</td>
</tr>
<tr>
<td>+</td>
<td>STATUS</td>
</tr>
</tbody>
</table>

### c) SDB

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB</td>
<td>PSEEBUF</td>
</tr>
<tr>
<td>+</td>
<td>PSEE</td>
</tr>
<tr>
<td>SUBORDINATE</td>
<td>ORD</td>
</tr>
<tr>
<td>+</td>
<td>PRAID</td>
</tr>
</tbody>
</table>

### d) SEEBUF

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTRSEE</td>
<td>NUNITS</td>
</tr>
</tbody>
</table>

### e) ARCSAW

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>PSB</td>
</tr>
<tr>
<td>ADDRESS</td>
<td>TYPE</td>
</tr>
<tr>
<td>DAMAGE (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

### f) ORDERS

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTRFORMS</td>
<td>PTRACT</td>
</tr>
</tbody>
</table>

### g) COMMAND

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>NUMACTS</td>
</tr>
<tr>
<td>TMFLG</td>
<td>ADDRESS</td>
</tr>
<tr>
<td>TIME (SPACE)</td>
<td></td>
</tr>
<tr>
<td>ACTION</td>
<td></td>
</tr>
</tbody>
</table>

### h) FORMATION

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFORM</td>
<td>NUMFLTS</td>
</tr>
</tbody>
</table>

### i) WINGMAN

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNEXT</td>
<td>PSB</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------</td>
</tr>
<tr>
<td>ARCFTSTATUS</td>
<td></td>
</tr>
<tr>
<td>PFLTDB</td>
<td>NUMAIRCRAFT DUMMY</td>
</tr>
<tr>
<td>PSTRTHX</td>
<td>PENDHX JAMSTAT</td>
</tr>
<tr>
<td>PNXTHX</td>
<td>PAIRBASE GRNDATK</td>
</tr>
<tr>
<td>CNTRLMODE</td>
<td>AIRCOMBAT ORBITSTAT</td>
</tr>
<tr>
<td>PGNOTGT</td>
<td>LANDING PROFILENDX</td>
</tr>
<tr>
<td></td>
<td>ALTUECHNG INTERCEPTSTATUS</td>
</tr>
<tr>
<td>FLITELEG</td>
<td>FUEL (SPACE)</td>
</tr>
<tr>
<td></td>
<td>ALTITUDE (SPACE)</td>
</tr>
<tr>
<td></td>
<td>SPEED (SPACE)</td>
</tr>
<tr>
<td></td>
<td>DIRECTION (SPACE)</td>
</tr>
<tr>
<td>k) MUN</td>
<td></td>
</tr>
<tr>
<td>PAG</td>
<td>NUMAG</td>
</tr>
<tr>
<td>PAA</td>
<td>NUMAA</td>
</tr>
<tr>
<td>l) LOAD</td>
<td></td>
</tr>
<tr>
<td>PNEXT</td>
<td>TYPE PORDDB</td>
</tr>
<tr>
<td>AMOUNT</td>
<td></td>
</tr>
<tr>
<td>1) FLTDB</td>
<td></td>
</tr>
<tr>
<td>PNXFLDB</td>
<td>NRFLITE</td>
</tr>
<tr>
<td>PTYPLDS</td>
<td>NOPYLDS</td>
</tr>
<tr>
<td>PTYAQDB</td>
<td>PTACDB</td>
</tr>
<tr>
<td>MAXNOAC</td>
<td>MINNOAC</td>
</tr>
<tr>
<td>MULTAC</td>
<td>PROFILE</td>
</tr>
<tr>
<td>SPFLTC (SPACE)</td>
<td></td>
</tr>
<tr>
<td>DISTSEP (SPACE)</td>
<td>190</td>
</tr>
</tbody>
</table>
n) PAYLOAD

<table>
<thead>
<tr>
<th>PAYLOAD</th>
<th>PAYLODBLK</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNXTYPD</td>
<td>NEXT</td>
</tr>
<tr>
<td>NRPDCLS</td>
<td>TYPEINDEX</td>
</tr>
<tr>
<td>MAXAMT</td>
<td>MINAMT</td>
</tr>
<tr>
<td>MAXFIRERANGE</td>
<td>PAYLODB</td>
</tr>
</tbody>
</table>

o) PAYLDDBLOK

<table>
<thead>
<tr>
<th>PAYLDDBLOK</th>
<th>AQDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEXT</td>
<td>TYPEINDEX</td>
</tr>
<tr>
<td>NRAQTYP</td>
<td>RANGE</td>
</tr>
<tr>
<td>NOUSE1</td>
<td>NOUSE2</td>
</tr>
</tbody>
</table>

p) AQDB

<table>
<thead>
<tr>
<th>AQDB</th>
<th>PROFILEDBLOK</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEXT</td>
<td>PROFILEDBLOK</td>
</tr>
<tr>
<td>NRAQTYP</td>
<td>PNXPROB</td>
</tr>
<tr>
<td>RANGE (SPACE)</td>
<td>NRPROFL</td>
</tr>
<tr>
<td>NOUSE1</td>
<td>ALTCREN (SPACE)</td>
</tr>
<tr>
<td>NOUSE2</td>
<td>ALTOTGT (SPACE)</td>
</tr>
<tr>
<td>ALTOAB (SPACE)</td>
<td>MAXSPEED (SPACE)</td>
</tr>
</tbody>
</table>

q) PROFILEDBLOK

<table>
<thead>
<tr>
<th>PROFILEDBLOK</th>
<th>MAXALTITUDE (SPACE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNXPROB</td>
<td>CRUISESPEED (SPACE)</td>
</tr>
<tr>
<td>NRPROFL</td>
<td>MAXALTITUDE (SPACE)</td>
</tr>
<tr>
<td>ALTCREN (SPACE)</td>
<td>MAXCLIMB Divine (SPACE)</td>
</tr>
<tr>
<td>ALTOTGT (SPACE)</td>
<td>FUELCONSUME (SPACE)</td>
</tr>
<tr>
<td>ALTOAB (SPACE)</td>
<td>ACQRANGE (SPACE)</td>
</tr>
<tr>
<td>MAXSPEED (SPACE)</td>
<td>RADARCS (SPACE)</td>
</tr>
<tr>
<td>CRUISESPEED (SPACE)</td>
<td>ATTACKRADIUS (SPACE)</td>
</tr>
<tr>
<td>MAXALTITUDE (SPACE)</td>
<td>MAXFUEL (SPACE)</td>
</tr>
</tbody>
</table>

r) ACDB

<table>
<thead>
<tr>
<th>ACDB</th>
<th>ACDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEXT</td>
<td>NRACTYPE</td>
</tr>
<tr>
<td>NRAQTYP</td>
<td>MAXSPEED (SPACE)</td>
</tr>
<tr>
<td>RANGE (SPACE)</td>
<td>CRUISESPEED (SPACE)</td>
</tr>
<tr>
<td>NOUSE1</td>
<td>MAXALTITUDE (SPACE)</td>
</tr>
<tr>
<td>NOUSE2</td>
<td>MINALTITUDE (SPACE)</td>
</tr>
<tr>
<td>ALTOAB (SPACE)</td>
<td>MAXCLIMB Divine (SPACE)</td>
</tr>
<tr>
<td>MAXSPEED (SPACE)</td>
<td>FUELCONSUME (SPACE)</td>
</tr>
<tr>
<td>CRUISESPEED (SPACE)</td>
<td>ACQRANGE (SPACE)</td>
</tr>
<tr>
<td>MAXALTITUDE (SPACE)</td>
<td>RADARCS (SPACE)</td>
</tr>
<tr>
<td>MINALTITUDE (SPACE)</td>
<td>ATTACKRADIUS (SPACE)</td>
</tr>
<tr>
<td>MAXCLIMB Divine (SPACE)</td>
<td>MAXFUEL (SPACE)</td>
</tr>
</tbody>
</table>

2) Field Definitions

a) C² Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNITNUMBER</td>
<td>Number of the unit.</td>
</tr>
<tr>
<td>PUP</td>
<td>Pointer to the C² block of the unit's commander.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PDOWN</td>
<td>Pointer to the $C^2$ block of the unit's sibling.</td>
</tr>
<tr>
<td>PSB</td>
<td>Pointer to the SB block of the unit.</td>
</tr>
<tr>
<td>UNITTYPE</td>
<td>The unit's type code (128)</td>
</tr>
<tr>
<td>SIDE</td>
<td>Unit affiliation</td>
</tr>
<tr>
<td></td>
<td>$2 = \text{Red (PACT)}$</td>
</tr>
<tr>
<td>PL2</td>
<td>Pointer to $C^2$ block of the unit.</td>
</tr>
<tr>
<td>PSDB</td>
<td>Pointer to the SDB block of the unit.</td>
</tr>
<tr>
<td>PFEL</td>
<td>Pointer to future event list EVENT block.</td>
</tr>
<tr>
<td>PACQ</td>
<td>Pointer to ACQBUF block. Used by CRC's for acquisition devices.</td>
</tr>
<tr>
<td>ID</td>
<td>Identification number.</td>
</tr>
<tr>
<td>DATABASE</td>
<td>If BOC or BTRY points to ADSITEDB block</td>
</tr>
<tr>
<td>PABSTATUS</td>
<td>Points to ABSTATUS block if the unit is an airbase.</td>
</tr>
<tr>
<td>+PARCFTSTAT</td>
<td>Points to ARCFTSTATUS block if the unit is a flight of aircraft</td>
</tr>
<tr>
<td>+PBOCSTAT</td>
<td>Points to BOCSTAT block if the unit is a battalion operations center.</td>
</tr>
<tr>
<td>+PBTRYSTAT</td>
<td>Points to BTRSTAT block if the unit is an anti-aircraft battery</td>
</tr>
<tr>
<td>+STATUS</td>
<td>Alternative field definition.</td>
</tr>
</tbody>
</table>

192
c) **SDB BLOCK**

- **PSB**  
  Pointer to the unit's SB block

- **PSEEBUF**  
  Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights.

- **+PSEE**  
  Pointer to CRCEES block which is used by CRC units to record the blue and red flights it sees. Also by the RTC as a pointer to it's assigned target list.

**SUBORDINATE**

Points to different types of subordinate description or target description blocks depending upon the unit type. Possible unit type vs. field use combination are as follows:

<table>
<thead>
<tr>
<th>UNIT TYPE</th>
<th>SUBORDINATE POINTS TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC</td>
<td>SUB</td>
</tr>
<tr>
<td>BOC</td>
<td>SUBLIST</td>
</tr>
<tr>
<td>BTRY</td>
<td>FIREUNIT</td>
</tr>
</tbody>
</table>

- **ORD**  
  Points to ORDERS block if the unit is a flight.

- **+PRAID**  
  Points to the RAIDBLOK if the unit is the Red Theater Commander.

d) **SEEBUF BLOCK**

- **PTRSEE**  
  Pointer to ARCFSAW block (first in list)
NUNITS - Number of ARCFSAW blocks in the list. Corresponds to number of targets perceived.

e) ARCFSAW BLOCK
PNEXT - Pointer to next ARCFSAW block in the list.
PSB - Pointer to SB block of target.
ADDRESS - Pointer to HEX block in which target is located.
TYPE - Unit type code of target
DAMAGE - Perceived damage level of the target. Real variable.

f) ORDERS BLOCK
PTRFORMS - Pointer to FORMATION block
PTRACT - Pointer to COMMAND block.
PNEXT - Pointer to next command block.
NUMACTS - Number of the command in the list.
TMFLG - Time flag. If 1 at time is associated with the command.
ADDRESS - Pointer to HEX block in which the command is to be carried out.
TIME - Time the command is to be performed.
ACTION - Command or action code

h) FORMATION BLOCK
PFORM - Pointer to first WINGMAN block in the list.
NUMFLTS - Number of WINGMAN blocks in the list. Corresponds to number of flights in the formation.
i) **WINGMAN BLOCK**

- **PNEXT** - Pointer to next WINGMAN block in the list
- **PSB** - Pointer to the SB block of the flight.

j) **ARCFSTATUS BLOCK**

- **PFLTDB** - Pointer to FLTDB block
- **PMUNITIONS** - Pointer to MUN block
- **PSTRTHX** - Pointer to HEX block in which current move begins
- **PENDHX** - Pointer to HEX block in which current move ends
- **PNXTHX** - Pointer to next HEX block
- **PAIRBASE** - Pointer to SB block of flight's home airbase.
- **CNTRLMODE** - 
- **PAIRTGT** - Pointer to SB block of airborne target.
- **PGNDTGT** - Pointer to SB block of ground target
- **NUMAIRCRAFT** - Number of aircraft in the flight
- **FLITELEG**
- **INTERCEPTSTATUS**
- **ALTUDECHNG**
- **PROFILENDX**
- **LANDING**
- **ORBITSTAT**
- **AIRCOMBAT**
- **BRNDATK**
- **FUEL** - Current fuel level. Real variable
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTITUDE</td>
<td>Altitude in meters. Real variable</td>
</tr>
<tr>
<td>SPEED</td>
<td>Speed. Real variable</td>
</tr>
<tr>
<td>DIRECTION</td>
<td>Direction. Real variable.</td>
</tr>
</tbody>
</table>

**k) MUN BLOCK**
- PAG: Pointer to ground attack LOAD block
- NUMAG: Number of LOAD blocks in ground attack munitions list.
- PAA: Pointer to air attack LOAD block
- NUMDA: Number of LOAD blocks in air attack munitions list.

**1) LOAD BLOCK**
- PNEXT: Pointer to next LOAD block in the list
- TYPE: Munitions class
- AMOUNT: Amount of munitions in tons
- PORDDB: Pointer to PAYLOAD block

**m) FLTDB BLOCK**
- PNXFLDB: Pointer to next FLTDB block
- NRFLITE: Unique flight specification number
- PTYPLODS: Pointer to payload data block payload class 6002
- NOPYLDS: Number of payload data blocks
- PTYAQDB: Pointer to acquisition data block (AQDB, Class 6007)
- PTACDB: Pointer to aircraft specification data block (ACDB, Class 6003)
- MAXNOAC: Maximum number of aircraft in flight
MINNOAC - Minimum number of aircraft in flight
MULTAC - Multiples of aircraft required for flight
PROFILE - Pointer to profile specification data block (PROFILEDBLOK), Class 6005
SPFLTC(SPACE) - Flight cruising speed in meters/second (real)
DISTSEP(SPACE) - Flight separation distance in meters (real)
n) PAYLOAD BLOCK
PNXTYPD - Pointer to next PAYLOAD block
NRPOCLS - Payload type, must be 3 or 4
3 = air to ground
4 = air to air
MAXAMT - Maximum number of loads of this payload
MINAMT - Minimum number of loads of this payload
MAXFIRERANGE - Future use by an enhancement for maximum fire range for engagements greater than one hex
PAYLDDDB - Pointer to PAYLOAD ID data block (PAYLDDDBLOK)
o) PAYLDDDBLOK BLOCK
NEXT - Pointer to next ID block
TYPEINDEX - Payload ID, unique within each payload type
p) AQDB BLOCK
NEXT - Pointer to next AQDB block
NRAQTYD - Unit type
6. BATTALION OPERATIONS CENTERS (BOC)

a. Data Block Index
   ABSITEDB
   BOCSTAT
   C2
   DIB
   DIL
   PAL
   PERLIST
   SB
   SDB
   SOURCE
   SUBLIST

b. Description
   The air defense information for the Battalion Operations Center (BOC) in MADEM is stored in the Digested Information List (DIL). The DIL is integrated with the subordinates of the BOC to create the Threat-Defense Analysis Structure for each BOC.

   The DIL represents the perceived threat to the BOC and is pointed to from the BOC's Status Board (BOCSTAT). The list of subordinates to the BOC are pointed to from the BOC Status Display Board (SDB). The BOC links its available subordinates to the targets producing the Threat-Defense Analysis Structure, made up of Possibilities and Allocations Lists (PAL) blocks. This structure of PAL blocks is in the form of a sparse matrix. Conceptually, the rows of the matrix shows what targets (DIL's) are associated with each subordinate (SUBLIST) so the subordinate can be maximally allocated. The columns are each target, DIL, and the subordinates that are currently associated with it. The elaborate linking allows easy addition or deletion from the matrix as well as ease of searching for the information needed.

   The DIL's are divided up into three different configurations: (1) Active DIL list, (2) Passive DIL list, (3) Force out queue. This allows
the threats, DIL's, to be handled in a systematic way. All the DIL configurations are pointed to from the BOCSTATUS.

The Active DIL (ADIL) is a doubly linked circular list of all the threats actively being considered by the BOC. The DIL's in the ADIL are also doubly threaded creating a list in sequence of priorities. Each DIL in the ADIL is considered one at a time in the list in increments controlled by the Scheduled Digest event. The Air Defense Data Base controls the maximum number of DIL's a BOC can have in its ADIL.

When a BOC fully assigns the responsibility of a threat, DIL, its subordinates, the DIL is taken out of the ADIL and put into the Passive DIL list (PDIL). The PDIL is a doubly linked list of DIL blocks with a pointer to the head of the list.

If a DIL in the ADIL is forced out because the need to bring in a DIL of a higher priority, it is put in the Force Out Queue (FOQ). The FOQ is a double linked list with a head and tail pointer to the list. This is an overflow list and the DIL will eventually be brought back into the ADIL when there is an opening. The head and tail pointers allow access from either end of the list to get the best DIL out of the FOQ.

Each DIL has an associated Digested Information Block (DIB) with it. This holds the current information that the BOC has on the target. When the DIL is in the Active DIL List a new DIB is created and compared to the old one each time the DIL (target) is examined. The DIL's are also associated with the Perceptions List, a PERLIST block from the Perceptions List and a DIL are linked up. The information in the PERLIST is from the Perception event and is used with the DIL to represent the information on a target.

c. Structure Overview
   1) Structure Diagrams (Figure III-27, III-28, III-29)
   2) Block Definitions

   COMMAND/CONTROL BLOCK. Contains unit no., type, side and C² PTRS.
Figure III-27. Battalion Operations Center Structure Diagram
Figure III-28. Digested Information List Configurations

Figure III-29. Digested Information List and Perception List
SB - SCOREBOARD BLOCK. Contains HEX address; unit ID, PTR to BOCSTAT

SDB - STATUS DISPLAY BLOCK. Contains PTRS to subordinate list and perceptions list.

ADSITEDB - AIR DEFENSE SIGHTED DATA BLOCK. Contains information on the sighted aircraft.

BOCSTAT - BOC STATUS BOARD. Contains PTR to the three DIL configurations, ADIL, TRACKED, FOQ, along with general info on the BOC.

DIL - DIGESTED INFORMATION LIST. Contains information on a target or possible target on coverage, engagement windows, and other info, contains PTRS to associated DIL and perception block.

SUBLIST - SUBORDINATE LIST BLOCK. Subordinate to the BOC, can be associated with one or more targets.

PAL - POSSIBILITIES AND ALLOCATIONS LIST. Created when a subordinate and a target are associated together.

PERLIST - PERCEPTIONS LIST BLOCK. Forms a list, contains information on the seeing status of the target.
DIB - DIGESTED INFORMATION BLOCK.
Contains information on the target as heading, velocity, and position

d. Block Specifications
1) Block Diagrams
a) C2

| UNITNUMBER | PUP | PDOWN |
| PSB | PNEXT |
| UNITYPE | SLIDE |

b) SB

| ADDRESS | PC2 |
| PSDB | PFEL |
| PACQ | ID |
| DATABASE | PBOCSTAT |

c) SOB

| PSB | PSEEBUF |
| SUBORDINATE | ORD |

d) SUBLIST

| BTRYSB | PNEXT |
| AUTO | NUMFU |
| RAMMO | LOAD |
| POAQUE | NOUSE |
| PPAL | ADDRESS |

e) BOCSTAT

| AUTOFLG | READY |
| PSDIG | PSDELY |
| PADIL | NUMADIL |
| PHFOQ | PTFOQ |
| PHNDIL | PTNDIL |
| HPRIOR | TPRIOR |
| PPDIL | NUMROY |
| PHADAQ | PTDAQ |
### f) DIL

<table>
<thead>
<tr>
<th>PUP</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPERCL</td>
<td>PDIB</td>
</tr>
<tr>
<td>LDC</td>
<td>PRIORITY</td>
</tr>
<tr>
<td>PRIUP</td>
<td>PRIDN</td>
</tr>
<tr>
<td>DCOV</td>
<td>SHORT</td>
</tr>
<tr>
<td>PDELAY</td>
<td>PENCEV</td>
</tr>
<tr>
<td>PPAL</td>
<td>PSUB</td>
</tr>
</tbody>
</table>

### g) PAL

<table>
<thead>
<tr>
<th>PUPT</th>
<th>PDOWNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUPB</td>
<td>PDOWNB</td>
</tr>
<tr>
<td>START (SPACE)</td>
<td></td>
</tr>
<tr>
<td>END (SPACE)</td>
<td></td>
</tr>
<tr>
<td>COVER</td>
<td>POIL</td>
</tr>
<tr>
<td>PEVENT</td>
<td>PSUB</td>
</tr>
</tbody>
</table>

### h) ADSITEDB

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>ADTYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODVAL1</td>
<td>MAXNUMDIGEST</td>
</tr>
<tr>
<td>MAXTIMEDIGEST</td>
<td>MINTIMEDIGEST</td>
</tr>
<tr>
<td>LOSTTIME (SPACE)</td>
<td></td>
</tr>
<tr>
<td>LASTCHANGE (SPACE)</td>
<td></td>
</tr>
<tr>
<td>ENGAGEWINDOW(SPACE)</td>
<td></td>
</tr>
<tr>
<td>MODVAL2 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MODVAL3 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>LOVONONE</td>
<td>ONE</td>
</tr>
<tr>
<td>COVONFEW</td>
<td>FEW</td>
</tr>
<tr>
<td>COVONMANY</td>
<td>MANY</td>
</tr>
<tr>
<td>TIMEFLIGHT (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MISSILERANGE (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MAXASSIGN</td>
<td>MODVAL4</td>
</tr>
<tr>
<td>MODVAL5 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MAXTRACKRANGE (SPACE)</td>
<td></td>
</tr>
<tr>
<td>LOCKONTIME (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>
h) ADSITEDB (Continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODVAL6 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MODVAL7 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>CONVLOAD</td>
<td></td>
</tr>
<tr>
<td>SNUKELOAD</td>
<td>LNUKELOAD</td>
</tr>
<tr>
<td>RESUPPLYCV</td>
<td>CVRESUPPLYFREQ</td>
</tr>
<tr>
<td>RESUPPLYSN</td>
<td>SNRESUPPLYFREQ</td>
</tr>
<tr>
<td>RESUPPLYLN</td>
<td>LNRESUPPLYFREQ</td>
</tr>
</tbody>
</table>

i) DIB

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME (SPACE)</td>
<td></td>
</tr>
<tr>
<td>SIDE</td>
<td>NUMAL</td>
</tr>
<tr>
<td>LOST</td>
<td>POSITION</td>
</tr>
<tr>
<td>HEADING (SPACE)</td>
<td></td>
</tr>
<tr>
<td>VELOCITY (SPACE)</td>
<td></td>
</tr>
<tr>
<td>ALTITUDE (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

j) PERLIST

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUP</td>
<td>PDOWN</td>
</tr>
<tr>
<td>PSB</td>
<td>PDIL</td>
</tr>
<tr>
<td>SEEN</td>
<td>PSS</td>
</tr>
<tr>
<td>PUPCHN</td>
<td>PDNCHN</td>
</tr>
<tr>
<td>TIME</td>
<td></td>
</tr>
</tbody>
</table>

k) SOURCE

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB</td>
<td>PNEXT</td>
</tr>
</tbody>
</table>

2) Field Definitions

a) \( C^2 \) Block

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNITNUMBER</td>
<td>Number of the unit. If negative, the unit is a passing target.</td>
</tr>
<tr>
<td>PUP</td>
<td>Pointer to the ( C^2 ) block of the unit's commander.</td>
</tr>
<tr>
<td>PDOWN</td>
<td>Pointer to the ( C^2 ) block of the unit's subordinate.</td>
</tr>
</tbody>
</table>
b) **SB Block**

- **ADDRESS** - Pointer to HEX block of the hex in which the unit is located.
- **PC2** - Pointer to C2 block of the unit.
- **PSDB** - Pointer to the SDB block of the unit.
- **PFEL** - Pointer to future event list EVENT block.
- **PACQ** - Pointer to ACQBUF block. Used by CRC's for acquisition devices.
- **ID** - Identification number.
- **DATABASE** - If BOC or BTRY points to ADSITEDB block.
- **PBOCSTAT** - Points to BOCSTAT block if the unit is a Battalion Operations Center.

c) **SDB Block**

- **PSB** - Pointer to the unit's SB block.
- **PSEEBUF** - Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights.
d) **SUBLIST**
- **BTRYSB** - Pointer to the **SB** block of the **BOC**.
- **PNEXT** - Pointer to the **SUBLIST** block of the subordinate's sibling.
- **AUTO** -
- **NUMFU** -
- **RAMMO** -
- **LOAD** -
- **PDAQUE** - Pointer to the **DAQUE** block.
- **NOUSE** - Pointer to the possibilities and allocations list block.
- **PPAL** -
- **ADDRESS** - Pointer to the **HEX** block.

e) **BOCSTAT**
- **AUTOFLG** -
- **READY** -
- **PSDIG** - Pointer to the **EVENT** block which is the next scheduled digest event.
- **PSDELY** - Pointer to the **EVENT** block.
- **PADIL** - Pointer to the **DIL** block which is the next threat (DIL) to look at in the active DIL list.
- **NUMADIL** - Number of DILS in the active DIL list.
- **PHFOQ** - Pointer to the **DIL** block which is the head of the Force Out Queue.
PTFOQ - Pointer to the DIL block which is the tail of the Force Out Queue.

PHNDIL - Pointer to the PERLIST block. This pointer is not actively used in MADEM at present.

PTNDIL - Pointer to the PERLIST block. This pointer is not actively used in MADEM at present.

HPRIOR - Pointer to the DIL block which is the head of the priority chain within the active DIL list.

TPRIOR - Pointer to the DIL block which is the tail of the priority chain within the active DIL list.

PPDIL - Pointer to the DIL block which is the head of the passive DIL list.

NUMROY -

PHDAQ -

PTDAQ -

f) DIL

PUP - Pointer to the DIL block which is the pointer up in DIL chain.

PDOWN - Pointer to the DIL block which is the pointer next in DIL chain.

PPERCL - Pointer to the PERLIST block which is the pointer to the perceptions list.
PDIB - Pointer to the DIB block which is the digested information block. It holds the information about the flight's track.

LOC - The code to tell which DIL configuration this OIL is located. Code: 0 - Active DIL List 1 - Force Out Queue 2 - Passive DIL List

PRIORITY - Priority of

PRIUP - Pointer to the DIL block which is the up pointer in the priority chain in the active DIL list.

PDOWN - Pointer to the DIL block which is the down pointer in the priority chain in the active DIL list.

DCOV - Desired coverage of this target flight.

SHORT - Shortfall from desired coverage, i.e., Desired-Allocated = Shortfall.

PDELAY - Pointer to the DAQE block which points to the delayed action queue, a chain of delayed actions regarding this target.

PENGEV - Pointer to the EVENT block.

PPAL - Pointer to the PAL block which is the possibilities and allocations list block, a part of the threat-defense analysis.

PSUB - Pointer to the SUBLIST block.
g) **PAL**

- **PUPT** - Pointer to the **PAL** block.
- **PDOWNT** - Pointer to the **PAL** block.
- **PUPB** - Pointer to the **PAL** block.
- **PDOWNB** - Pointer to the **PAL** block.
- **START** - Beginning time that the target is active for this BOC.
- **END** - Ending time that the target is active for this BOC.
- **COVER** - Amount of coverage the subordinate is giving this target.
- **PDIL** - Pointer to the **PAL** block which is the target (DIL) associated with this PAL.
- **PEVENT** - Pointer to the **EVENT** block.
- **PSUB** - Pointer to the **SUBLIST** block which is the subordinate associated with this PAL.

h) **ADSITEDB** Block

- **PNEXT** - Pointer to next **ADSITEDB** block.
- **AOTYPE** - Unit type of this unit. Must be a BOC or BTRY.
- **MODVAL 1** - Model Value - 1.
- **MAXIMUM DIGEST** - Maximum number of flights on which a BOC or BTRY can be digesting info at one time.
- **MAX TIME DIGEST** - Maximum time (in seconds) between consecutive digests of info (BOC and BTRY).
- **MIN TIME DIGEST** - Minimum time (in seconds) between consecutive digests of info (BOC and BTRY).
LOST TIME - Time (in seconds) after which a track not seen is assumed permanently lost (BOC and BTRY).

LAST CHANCE - Time (in seconds) considered short for a subordinate to respond to a target. (Time from now until his last chance to shoot.) BOC only, for BTRY = 0.

ENGAGE WINDOW - Minimum length of subordinates engagement window for a significant engagement opportunity in seconds (BOC and BTRY).

MODVAL 2 - Model Value = 0

MODVAL 3 - Model Value = 0

COVONONE - Desired number of fire units coverage for one aircraft (BOC and BTRY).

ONE - Model Value = 1

COVONFEW - Desired number of fire units coverage for few aircraft (BOC and BTRY).

FEW - Model Value = 5, number of aircraft considered "few."

COVONMANY - Desired number of fire units coverage for many aircraft (BOC and BTRY).

MANY - Model Value = 1000000.

TIMEFLIGHT - Maximum time (in seconds) of flight for missile (BOC and BTRY).

MISSILE RANGE - Maximum range for missiles in meters (BOC and BTRY).
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXASSIGN</td>
<td>Maximum number of targets per ready fire unit to be assigned at one time.</td>
</tr>
<tr>
<td></td>
<td>BOC only, for BTRY = 0.</td>
</tr>
<tr>
<td>MODVAL 4</td>
<td>Model Value; for BOC = 8, BTRY = 11.</td>
</tr>
<tr>
<td>MODVAL 5</td>
<td>Model Value = 0</td>
</tr>
<tr>
<td>MAX TRACK RANGE</td>
<td>Maximum tracking range in meters.</td>
</tr>
<tr>
<td></td>
<td>BTRY only, for BOC = 0.</td>
</tr>
<tr>
<td>LOCK ON TIME</td>
<td>Assumed time (in seconds) for BTRY achieve lock on. BTRY only, for BOC = 0.</td>
</tr>
<tr>
<td>MODVAL 6</td>
<td>Model Value = 0.</td>
</tr>
<tr>
<td>MODVAL 7</td>
<td>Model Value = 0.</td>
</tr>
<tr>
<td>CONVLOAD</td>
<td>Number of conventional missiles.</td>
</tr>
<tr>
<td>SNUKELOAD</td>
<td>Number of small nukes.</td>
</tr>
<tr>
<td>LNUKELOAD</td>
<td>Number of large nukes.</td>
</tr>
<tr>
<td>RESUPPLYCV</td>
<td>Number of missiles per resupply of ammo. BTRY only, for BOC = 0.</td>
</tr>
<tr>
<td>CVRESUPPLYFREQ</td>
<td>Time (in seconds) between resupply of conventional ammo. BTRY only, for BOC</td>
</tr>
<tr>
<td>RESUPPLYSN</td>
<td>Number of missiles per resupply of small nukes. BTRY only, BOC = 0.</td>
</tr>
<tr>
<td>SNRESUPPLYFREQ</td>
<td>Time (in seconds) between resupply of small nukes. BTRY only, BOC = 0.</td>
</tr>
<tr>
<td>RESUPPLYLN</td>
<td>Number of missiles per resupply of large nukes. (BTRY only, BOC = 0).</td>
</tr>
</tbody>
</table>
LNRESUPPLYFREQ - Time (in seconds) between resupply of large nukes. BTRY only, BOC = 0.

i) DIB
TIME - Side of target (DIL).
NUMAL -
LOST - Code when target is lost.
POSITION - Position of target.
HEADING - Heading of target (DIL).
VELOCITY - Velocity of target.
ALTITUDE - Altitude of target.

j) PERLIST
PUP - Pointer to the PERLIST block which is an up pointer in the perceptions list.
PDOWN - Pointer to the PERLIST block which is a next pointer in the perceptions list.
PSB - Pointer to the BOC's scoreboard.
PDIL - Pointer to the DIL block which is associated with it.
SEEN - Code to show seeing status of the target.
  0 - Can not see and has not been assigned.
  1 - Early warning, has been assigned by superior.
  2 - Can see target.
PSS - Pointer to the SOURCE block which is a list of subordinates that can see the target.
PUPCHN -
PDNCHN -
TIME - Time that the target was last seen.

k) SOURCE
PSB - Pointer to the SCOREBOARD block which is the scoreboard of the subordinate that can see the target.
PNEXT - Pointer to the SOURCE block which is the next subordinate in the list that can see the target.

e. **Linkages to Other Data Structures**
The BOC structures are linked to the command and control tree by the C2 block of the BOC.

f. **Notes**
7. **SAM BATTERY (BTRY)**
   a. **Data Block Index**
      ABSITEDB
      ALLOCATE
      BTRYDIL
      BTRYSTAT
      C^2
      DIB
      FIREUNT
      PATENGAGE
      PERLIST
      SB
      SDB
   b. **Description**
      The air defense information for the SAM Battery (BTRY) in MADEM is stored in the Battery Digested Information List (BTRYDIL). The fireunits of a Battery are allocated to the BTRYDILS to create the Threat-Defense Allocation Structure for each Battery.

      The BTRYDIL represents the perceived threat to the Battery and is pointed to from the Battery's Status Board (BTRYSTAT). The list of fireunits of the Battery are pointed to from the Battery Status Display Board (SDB). The fireunits are the Battery's subordinates and the number of fireunits are limited depending on the type of Battery it is. A Hawk battery can have at most two fireunits where a Herc battery can have only one. When a fireunit is allocated to a threat, BTRYDIL, an ALLOCATE block is created to link the two together. Note that at most there can only be one allocation block for any one fireunit, but a threat, BTRYDIL, could have multiple allocation blocks. This does not hold for the last Battery type, the PATRIOT. A PATRIOT does not have any fireunits as subordinates, it has a known engagement capacity. The number of engagements can be allocated to one or multiple threats, BTRYDILS. When a PATRIOT engagement is allocated to a threat a PATENGAGE block is created.
The BTRYDILS are divided up into three different DIL configurations; (1) Active DIL list, (2) TRACKED DIL list, (3) Force out queue. This allows the threats, DILS, to be handled in a systematic way. All the DIL configurations are pointed to from the BTRYSTAT.

The Active DIL (ADIL) is a doubly linked circular list of all the threats actively being considered by the Battery. The DIL's in the ADIL are considered one at a time in the list, in increments controlled by the Scheduled Digest event. The Air Defense Data Base controls the maximum number of DIL's a Battery can have in it's ADIL.

When a Battery fully assigns the responsibility of a threat DIL to it's subordinates, the DIL is taken out of the ADIL and put into the TRACKED DIL list. The Tracked List is a doubly linked list of DIL blocks with a pointer to the head of the list.

If a DIL in the ADIL is forced out because the need to bring in a DIL of a higher priority, it is put in the Force Out Queue (FOQ). The FOQ is a doubly linked list with a head and tail pointer to the list. This is an overflow list and the DIL will eventually be brought back into the ADIL when there is an opening. The head and tail pointers allow access from either end of the list to get the best DIL out of the FOQ.

Each DIL has an associated Digested Information Block (DIB) with it. This holds the current information that the BOC has on the target. When the DIL is in the Active DIL List a new DIB is created and compared to the old one each time the DIL target is examined. The DIL's are also associated with the Perceptions List, a PERLIST block from the Perceptions List and a DIL are linked up. The information in the PERLIST is from the Perception event and is used with the DIL to represent the information on a target.

c. Structure Overview
   1. Structure Diagram (Figure III-30, III-31, III-32, III-33)
Figure III-30. SAM Battery Structure Diagram (HAWK and HERC)

Figure III-31. SAM Battery Structure Diagram (PATRIOT)
Figure III-32. BTRYDIL Configurations
Figure III-33. Battery BTRYDIL Perception List Association
d. **Block Specifications**

1) **Block Diagrams**

a) \(c^2\)

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
<th>IDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUP</td>
<td>PDOWN</td>
</tr>
<tr>
<td>PSB</td>
<td>PNEXT</td>
</tr>
<tr>
<td>UNITTYPE</td>
<td>SIDE</td>
</tr>
</tbody>
</table>

b) **SB**

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSOB</td>
<td>PFEL</td>
</tr>
<tr>
<td>PACQ</td>
<td>IO</td>
</tr>
<tr>
<td>DATABASE</td>
<td>PBOSTAT</td>
</tr>
</tbody>
</table>

c) **SDB**

<table>
<thead>
<tr>
<th>PSB</th>
<th>PSEEBUF</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBORDINATE</td>
<td>ORD</td>
</tr>
</tbody>
</table>

d) **FIREUNIT**

<table>
<thead>
<tr>
<th>AMMO</th>
<th>PNEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>POIL</td>
<td>STAGE</td>
</tr>
<tr>
<td>CEASE</td>
<td>PENGEV</td>
</tr>
<tr>
<td>CAMMO</td>
<td>BUSV</td>
</tr>
<tr>
<td>NIAMMO</td>
<td>N2AMMO</td>
</tr>
</tbody>
</table>

e) **BTRYSTAT**

<table>
<thead>
<tr>
<th>AUTO</th>
<th>NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSDIG</td>
<td>PSDELY</td>
</tr>
<tr>
<td>PADIL</td>
<td>NADIL</td>
</tr>
<tr>
<td>PHFOQ</td>
<td>PTFOQ</td>
</tr>
<tr>
<td>PHNDIL</td>
<td>PTNDIL</td>
</tr>
<tr>
<td>HPRIOR</td>
<td>IPRIOR</td>
</tr>
<tr>
<td>TRACTED</td>
<td>IDLE</td>
</tr>
<tr>
<td>PHDAQ</td>
<td>PTDAQ</td>
</tr>
<tr>
<td>PTL</td>
<td>A1</td>
</tr>
<tr>
<td>RESUPPLY</td>
<td>A2</td>
</tr>
<tr>
<td>AMTDT</td>
<td>NUMENG</td>
</tr>
<tr>
<td>NUCMO</td>
<td>A3</td>
</tr>
</tbody>
</table>
f) **BTRYDIL**

<table>
<thead>
<tr>
<th>PUP</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPERC</td>
<td>PDIB</td>
</tr>
<tr>
<td>LOC</td>
<td>PRIORITY</td>
</tr>
<tr>
<td>PRIUP</td>
<td>PRION</td>
</tr>
<tr>
<td>DCOV</td>
<td>SHORT</td>
</tr>
<tr>
<td>PDELAY</td>
<td>PENGIV</td>
</tr>
<tr>
<td>PAL</td>
<td>NUKE</td>
</tr>
<tr>
<td>CEASE</td>
<td>ASSIGN</td>
</tr>
<tr>
<td>TRACK</td>
<td>FIRE</td>
</tr>
<tr>
<td>START (SPACE)</td>
<td></td>
</tr>
<tr>
<td>END (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

g) **ALLOCATE**

<table>
<thead>
<tr>
<th>PUP</th>
<th>PPDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMMOTYPE</td>
<td>PFU</td>
</tr>
</tbody>
</table>

h) **ADSITEDB**

<table>
<thead>
<tr>
<th>PNEXZ</th>
<th>ADTYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODVAL1*</td>
<td>MAXNUMDIGEST</td>
</tr>
<tr>
<td>MAXTIMEDIGEST</td>
<td>MINTIMEDIGEST</td>
</tr>
<tr>
<td>LOSTTIME (SPACE)</td>
<td></td>
</tr>
<tr>
<td>ENGAGEWINDOW (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MODVAL2 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MODVAL3 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>COVONONE</td>
<td>ONE</td>
</tr>
<tr>
<td>COVONFEW</td>
<td>FEW</td>
</tr>
<tr>
<td>COVONMANY</td>
<td>MANY</td>
</tr>
<tr>
<td>TIMEFLIGHT (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MISSILE RANGE (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MAXASSIGN MODVAL4</td>
<td></td>
</tr>
<tr>
<td>MODVAL5 (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MAXTRACKRANGE (SPACE)</td>
<td></td>
</tr>
<tr>
<td>LOCKONTIME (SPACE)</td>
<td></td>
</tr>
<tr>
<td>MODVAL6 (SPACE)</td>
<td>MODVAL7 (SPACE)</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>CONVLOAD</td>
<td></td>
</tr>
<tr>
<td>SNUKLOAD</td>
<td>LNUKELOAD</td>
</tr>
<tr>
<td>RESUPPLYCV</td>
<td>CVRESUPPLYFREQ</td>
</tr>
<tr>
<td>RESUPPLYSN</td>
<td>SNRESUPPLYFREQ</td>
</tr>
<tr>
<td>RESUPLLYLN</td>
<td>LNRESUPPLYFREQ</td>
</tr>
</tbody>
</table>

### i) PATENGAGE

<table>
<thead>
<tr>
<th>PUP</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDIL</td>
<td>STAGE</td>
</tr>
<tr>
<td>CEASE</td>
<td>PENGEV</td>
</tr>
</tbody>
</table>

### j) DIB

<table>
<thead>
<tr>
<th>TIME (SPACE)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SIDE</td>
<td>NUMAL</td>
</tr>
<tr>
<td>LOST</td>
<td>POSITION</td>
</tr>
<tr>
<td>HEADING (SPACE)</td>
<td></td>
</tr>
<tr>
<td>VELOCITY (SPACE)</td>
<td></td>
</tr>
<tr>
<td>ALTITUDE (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

### k) PERLIST

<table>
<thead>
<tr>
<th>PUP</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB</td>
<td>PDIL</td>
</tr>
<tr>
<td>SEEN</td>
<td>PSS</td>
</tr>
<tr>
<td>PULPCHN</td>
<td>PDNCHN</td>
</tr>
<tr>
<td>TIME (SPACE)</td>
<td></td>
</tr>
</tbody>
</table>

### 2) Field Definitions

#### a) C2 Block

- **UNITNUMBER** - Number of the unit. If negative the unit is a passing target.
- **PUP** - Pointer to the C² block of the unit's commander.
- **PDOWN** - Pointer to the C² block of the unit's subordinate.
PNEXT - Pointer to the $C^2$ block of the unit's sibling.

PSB - Pointer to the SB block of the unit.

UNITTYPE - The unit's type code (see subsection f)

SIDE - Unit affiliation.
1 = Blue (NATO)
2 = Red (PACT)

b) SB BLOCK
ADDRESS - Pointer to HEX block of the HEX in which the unit is located.
PC2 - Pointer to $C^2$ block of the unit.
PSDB - Pointer to the SDB block of the unit.
PFEL - Pointer to future event list EVENT block.
PACQ - Pointer to ACQBUF block. Used by CRC's for acquisition devices.
ID - Identification number
DATABASE - If BOC or BTRY points to ADSITEDB block.
PBOCSTAT - Points to BOCSTAT block if the unit is a battalion operations center.

c) SDB BLOCK
PSB - Pointer to the Unit's SB block.
PSEEBUF - Pointer to SEEBUF block which is used by aircraft flight units to record targets seen and their perceived damage levels. This field definition is used only by flights.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBORDINATE</td>
<td>Points to FIREUNIT block which is a list of subordinate fire units.</td>
</tr>
<tr>
<td>ORD</td>
<td>Points to ORDERS block if the unit is a flight.</td>
</tr>
<tr>
<td>d) AMMO</td>
<td>Pointer to the FIREUNIT block of the subordinate fireunits sibling.</td>
</tr>
<tr>
<td>PNEXT</td>
<td>Pointer to the FIREUNIT block of the subordinate fireunits sibling.</td>
</tr>
<tr>
<td>PDIL</td>
<td>Pointer to the BTRVDIL block associate with this battery fireunit.</td>
</tr>
<tr>
<td>STAGE</td>
<td></td>
</tr>
<tr>
<td>CEASE</td>
<td></td>
</tr>
<tr>
<td>PENGEV</td>
<td></td>
</tr>
<tr>
<td>CAMMO</td>
<td></td>
</tr>
<tr>
<td>BUSY</td>
<td></td>
</tr>
<tr>
<td>NIAMMO</td>
<td></td>
</tr>
<tr>
<td>N2AMMO</td>
<td></td>
</tr>
<tr>
<td>e) BTRYSTAT</td>
<td></td>
</tr>
<tr>
<td>f) BTRYDIL</td>
<td></td>
</tr>
<tr>
<td>PUP</td>
<td>Pointer to the BTRYDIL block which is the previous entry in the DIL chain.</td>
</tr>
<tr>
<td>PDOWN</td>
<td>Pointer to the BTRYDIL block which is next entry in the DIL chain.</td>
</tr>
<tr>
<td>PPERC</td>
<td>Pointer to the PERLIST block which is the entry in the perceptions list for this DIL.</td>
</tr>
</tbody>
</table>
**PDIB** - Pointer to the DIB block which is the digested information block holding information about the target.

**LOC** - The code to tell which DIL configuration this DIL is located in

- Code: 0 - Active DIL list
- 1 - Force out queue
- 2 - Tracked list

**PRIORITY** - Priority of this flight as a target.

**PRIUP** - Pointer to the BTRYDIL block which is the head of the priority chain within the active DIL list.

**PRIDN** - Pointer to the BTRYDIL block which is the tail of the priority chain within the active DIL list.

**DCOV** - Desired coverage on this flight.

**SHORT** - Shortfall from desired coverage:
- short - DCOV (DESIRED COVERAGE) - Allocated.

**PDELAY** - Pointer to the DAQE block which is the delayed action queue, a list of delayed action for this track.

**PENGEV** - Pointer to the EVENT block

**PAL** - Pointer to the ALLOCATE block which is the allocations list for this tracked target, holds the list of all fire units allocated to a target.
<table>
<thead>
<tr>
<th>NUKE</th>
<th>Flags target NUKE/NoNUKE.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEASE</td>
<td>'cease' marker</td>
</tr>
<tr>
<td>ASSIGN</td>
<td></td>
</tr>
<tr>
<td>TRACK</td>
<td>Flag: 1 - waiting to track</td>
</tr>
<tr>
<td></td>
<td>2 - untrackable</td>
</tr>
<tr>
<td></td>
<td>0 -</td>
</tr>
<tr>
<td>FIRE</td>
<td>Waiting to fire flag</td>
</tr>
<tr>
<td>START</td>
<td>Starting window</td>
</tr>
<tr>
<td>END</td>
<td>End window</td>
</tr>
</tbody>
</table>

**g) ALLOCATE**

<table>
<thead>
<tr>
<th>PUP</th>
<th>Pointer to the ALLOCATE block which is the previous entry in the list of allocations.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDOWN</td>
<td>Pointer to the ALLOCATE block which is the next entry in the list of allocations.</td>
</tr>
<tr>
<td>AMMOTYPE</td>
<td></td>
</tr>
<tr>
<td>PFU</td>
<td>Pointer to the FIREUNIT block which is the fire unit associated with this allocation.</td>
</tr>
</tbody>
</table>

**h) ADSITEDB BLOCK**

<table>
<thead>
<tr>
<th>PNEXT</th>
<th>Pointer to next ADSITEDB block.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADTYPE</td>
<td>Unit type of this unit, must be a BOC or BTRY.</td>
</tr>
<tr>
<td>MODVAL 1</td>
<td>Model value = 1</td>
</tr>
<tr>
<td>MAXNUM DIGEST</td>
<td>Maximum number of flights on which a BOC or BTRY can be digesting info at one time.</td>
</tr>
<tr>
<td>MAXTIMEDIGEST</td>
<td>Maximum time (in seconds) between consecutive digests of into (BOC &amp; BTRY).</td>
</tr>
</tbody>
</table>
MINTIMEDIGEST - Minimum time (in seconds) between consecutive digests of into (BOC & BTRY)

LOSTTIME: - Time (in seconds) after which a track not seen is assumed permanently lost (BOC & BTRY).

LASTCHANCE - Time (in seconds) considered short for subordinate to respond to a target. (time from now until his last chance to shoot). BOC only, for BTRY = 0.

ENGAGE WINDOW - Minimum length of subordinates engagement window for a significant engagement opportunity in seconds (BOC & BTRY).

MODVAL 2 - Model Value = 0

MODVAL 3 - Model value = 0

COVONONE - Desired number of fire units coverage for one aircraft (BOC & BTRY)

ONE - Model value = 1

COVONFEW - Desired number of fire units coverage for few aircraft (BOC & BTRY)

FEW - Model value = 5, number of aircraft considered "few".

COVONMANY - Desired number of fire units coverage for many aircraft (BOC & BTRY).

MANY - Model value = 1000000.

TIMEFLIGHT - Maximum time (in seconds) of flight for missile (BOC & BTRY)
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MISSILE RANGE</td>
<td>Maximum range for missiles in meters (BOC &amp; BTRY).</td>
</tr>
<tr>
<td>MAXASSIGN</td>
<td>Maximum number of targets per ready fire unit to be assigned at one time. BOC only, for BTRY = 0.</td>
</tr>
<tr>
<td>MODVAL 4</td>
<td>Model value; for BOC = 8, for BTRY = 11.</td>
</tr>
<tr>
<td>MODVAL 5</td>
<td>Model value = 0.</td>
</tr>
<tr>
<td>MAX TRACK RANGE</td>
<td>Maximum tracking range in meters. BTRY only, for BOC = 0.</td>
</tr>
<tr>
<td>LOCK ON TIME</td>
<td>Assume time (in seconds) for BTRY to achieve lockon. BTRY only, for BOC = 0.</td>
</tr>
<tr>
<td>MODVAL 6</td>
<td>Model value = 0.</td>
</tr>
<tr>
<td>MODVAL 7</td>
<td>Model Value = 0.</td>
</tr>
<tr>
<td>CONVLOAD</td>
<td>Number of conventional missiles.</td>
</tr>
<tr>
<td>SNUKELOAD</td>
<td>Number of large nukes.</td>
</tr>
<tr>
<td>LNUKELOAD</td>
<td>Number of large nukes.</td>
</tr>
<tr>
<td>RESUPPLYCV</td>
<td>Number of missiles per resupply of ammo. BTRY only, for BOC = 0.</td>
</tr>
<tr>
<td>CVRESUPPLYFREQ</td>
<td>Time (in seconds) between resupply of conventional ammo. BTRY only, for BOC = 0.</td>
</tr>
<tr>
<td>RESUPPLYSN</td>
<td>Number of missiles per resupply of small nukes. BTRY only, BOC = 0.</td>
</tr>
<tr>
<td>SNRESUPPLYFREQ</td>
<td>Time (in seconds) between resupply of small nukes. BTRY only, BOC = 0.</td>
</tr>
<tr>
<td>RESUPPLYLN</td>
<td>Number of missiles per resupply of large nukes. (BTRY only, BOC = 0.</td>
</tr>
</tbody>
</table>
LNRESUPPLYFREQ - Time (in seconds) between resupply of large nukes. BTRY only, BOC = 0.

i) PATENGAGE
   PUP - Pointer to the PATENGAGE block which is the previous entry in the list of PATENGAGES below the BTRYDIL.
   PDOWN - Pointer to the PATENGAGE block which is the next entry in the list of PATENGAGES below the BTRYDIL.
   PDIL - Pointer to the BTRYDIL block associated with this PATENGAGE.
   STAGE -
   CEASE -
   PENGEV - Pointer to the EVENT block.

j) DIB
   TIME -
   SIDE - Side of target (DIL)
   NUMAL -
   LOST - Code when target is lost.
   POSITION - Position of target
   HEADING - Heading of target (DIL)
   VELOCITY - Velocity of target
   ALTITUDE - Altitude of target

k) PERLIST
   PUP - Pointer to the PERLIST block which is an up pointer in the perceptions list.
   PDOWN - Pointer to the PERLIST block which is a next pointer in the perceptions list.
PSB - Pointer to the DIL block which is associated with it.
SEEN - Code to show seeing status of the target
   0 - can not see, and has not been assigned
   1 - early warning, has been assigned by superior
   2 - can see target
PSS - Not used with battery, only BOC
PUPCHN -
PDNCHN -
TIME - Time that the target was last seen

e. Linkages to other Data Structures
   The battery structures are linked to the command and control tree by the C² block of the battery.

f. Notes
8. PASSIVE TARGETS (PT)
   a. Data Block Index
      \[ C_2 \]
      SB (modified)
   b. Description
      The PASSIVE TARGET LIST is made up of \( C_2 \) and associated CB blocks which are in a linked list. This list is separate from the \( C_2 \) TREE and is used to represent blue units which are non-players. These blue target units are non-players in the sense that they merely act as objectives for red attacks.
      Passive target units are characterized by negative unit numbers in their \( C_2 \) blocks and a modified SB block which has three rather than the usual four words. The configuration of the passive target list is shown in the structure diagram.
   c. Structure Overview
      1) Structure Diagram Figure III-34)
Figure III-34. Passive Target List Structure Diagram
2) **Block Definitions**

- **COMMAND/CONTROL BLOCK.** Contains list pointers, a negative unit number, pointer to SB, unit type code and side. The usual four words. Contains pointers to $C^2$ and HEX blocks.

### d. Block Specifications

1) **Block Diagrams**

a) $C^2$

<table>
<thead>
<tr>
<th>UNITNUMBER</th>
<th>PDOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUP</td>
<td>PSB</td>
</tr>
<tr>
<td>PSB</td>
<td>UNITTYPE</td>
</tr>
<tr>
<td>UNITTYPE</td>
<td>SIDE</td>
</tr>
</tbody>
</table>

b) **SB (MODIFIED)**

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>PC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUMLTA IV DAMAGE</td>
<td>PACQ</td>
</tr>
<tr>
<td>PACQ</td>
<td>ID</td>
</tr>
</tbody>
</table>

2) **Field Definitions**

a) $C^2$ **BLOCK**

- **UNITNUMBER** - Number of the unit. If negative the unit is a passive target.
- **PUP** - Pointer to the $C^2$ block of the unit's commander.
- **PDOWN** - Pointer to the $C^2$ block of the unit's subordinate.
- **PNEXT** - Pointer to the $C^2$ block of the unit's sibling.
- **PSB** - Pointer to the SB block of the unit.
- **UNITTYPE** - The unit's type code (see subsection f)
b) **SB BLOCK**

- **ADDRESS** - Pointer to HEX block of the hex in which the unit is located.
- **PC2** - Pointer to C² block of the unit.
- **CUMLTIVDAMAGE** - Cumulative damage to target.
- **PACQ** - Pointer to ACQBUF block. Used by CRC's for acquisition devices. (not used)

- **ID** - Identification number.

### e. Linkages to Other Data Structures

The PASSIVE TARGET LIST is used exclusively for targeting by the red planning module. Its only external linkage is to the HEX block in which the unit is located.

### f. Notes
H. STORAGE SPACE MANAGEMENT

1. Data Block Index
   Assorted unnamed unMIDASized blocks released from active use for data storage.

2. Description
   Storage space in the dynamically allocated array ISPACE (131000) is carried out using the routines GIMME and RELEASE. GIMME is used to allocate storage blocks in ISPACE. Given the desired block length in words, GIMME first searches the Garbage Collection Matrix structure (illustrated in the structure diagrams) for a previously allocated but now released block of the desired length. If no block is found in the matrix, GIMME accesses NEW SPACE (Free Space) using the ESPTR pointer and allocates the desired block. If there is no space available, GIMME stops the simulation and issues a STORAGE SPACE OVERFLOW message.

   RELEASE is used to construct the Garbage Collection Matrix. When a block is no longer required, GIMME is called with the length of the block to be released. RELEASE then adds the used block to the appropriate list in the matrix. The array ICTGIM(20) is used to count the number of released blocks in each size list. Blocks greater than or equal to 20 words in length are counted in element 20 of ICTGIM.

3. Structure Overview
   a. Structure Diagrams (See Figures III-35 and III-36)
   b. Block Definitions
      BLKLENGTH BLOCK SIZE LIST BUFFER BLOCK. Contains a block length field, a pointer to the list of BLK blocks of the specified length and a pointer to the next BLKLENGTH block for the n next size SIZE LIST.

      BLK RELEASED BLOCK. Storage block which has been released for further use. Contains a pointer to the next BLK block in its SIZE LIST. All remaining words initialized to zero (0).
ISPACE/SPACE

* PREVIOUSLY ALLOCATED BUT NOW RELEASED FOR REUSE.

NEW SPACE (FREE SPACE)

USED SPACE

RSLPTR - POINTER TO (ADDRESS OF) GARBAGE COLLECTION MATRIX
FSPTR - POINTER TO (ADDRESS OF) FREE SPACE

Figure III-35. I-Space Array Configuration
Figure III-36. Garbage Collection Matrix Structure Diagram
4. **Block Specifications**

a. **Block Diagrams**

1) **BLKLENGTH**
   - BLOCKLENGTH
   - PTRBLOCK
   - PTRNEXT

2) **BLK**
   - PTRNEXT
   - 0
   - 0
   - 0
   - 0
   - 0
   - 0
   - 0
   - 0
   - VARIABLE LENGTH

b. **Field Definitions**

1) **BLKLENGTH Block**
   - BLOCKLENGTH Length of BLK blocks on list (in words).
   - PTRBLOCK Pointer to first BLK block in the list.
   - PTRNEXT Pointer to next BLKLENGTH block.

2) **BLK Block**
   - PTRNEXT Pointer to next BLK block in the size list.

5. **Linkages to Other Data Structures**

All other data structures are built using storage blocks from either
the GARBAGE MATRIX or FREESPACE.

6. **Notes**
APPENDIX A
EXECUTION REQUIREMENTS

MADEM ON THE AFWL SYSTEM

MADEM uses either of 2 computers at AFWL, the Y mainframe (MFY) or the X mainframe (MFX). Both are CDC cyber 176 machines. We generally run on MFY since our data 100 has a direct line there, but we can run on either. It is possible to use a dial up terminal for either batch or interactive service using the following phone numbers:

<table>
<thead>
<tr>
<th>Baud Rate</th>
<th>MFY</th>
<th>MFX</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>505-264-2082 (3)</td>
<td>505-842-5162 (17)</td>
</tr>
<tr>
<td>300</td>
<td>--</td>
<td>505-842-9980 (10)</td>
</tr>
<tr>
<td>300</td>
<td>--</td>
<td>505-264-5875 (3)</td>
</tr>
<tr>
<td>300</td>
<td>--</td>
<td>505-265-9861 (3)</td>
</tr>
<tr>
<td>1200</td>
<td>505-264-5840 (3)</td>
<td>505-264-5705</td>
</tr>
<tr>
<td>1200</td>
<td>505-264-7812 (3)</td>
<td>--</td>
</tr>
<tr>
<td>4800</td>
<td>505-842-6392 (2)</td>
<td>505-842-6391 (4)</td>
</tr>
<tr>
<td>4800</td>
<td>--</td>
<td>505-842-5711 (6)</td>
</tr>
</tbody>
</table>

The number in parenthesis is the number of lines on that rotary.

USING THE DATA 100

The Data 100 (JB-SC) remote batch terminal has a direct line going to AFWL's MFY.

To Bring up the Data 100:

1. Load Emulator - Red in "Data 100" cards by pressing "HALT", then "LOAD". After cards have been read press "RUN".
2. Press Xmit button - Xmit light on is for D29 keypunch.
3. Wait for "Data Link" Light
4. Before entering each command, press control-A.
5. Type "LOGIN, SGCBDM, WDNA14V6, SUP (or L5 for V6)
7. Type "C".

To Enter Cards:

1. Load Cards
2. Press Start
3. Type "R" When Reader stops.

To Turn on Line Printer Type "ON,LP"

AFWL JOB CARD (1st Card in Deck)

Example: WBDMSBM, ST176, TA10, J037, P66.

WBDMSBM - Can be any name, 1st 5 CHARS used as 1st 5 CHARS of 7 CHAR job name.
ST176 - Tells it to run on either CYBER176 (MFY or MFX). Can also use STMFX or STMFY to run on a particular machine.

T40 - CPU time limit in OCTAL seconds.

I037 - IO time limit in OCTAL seconds.

P66 - Request for 66 priority. The highest priority allowed is dependent on IO+CPU time requested.

<table>
<thead>
<tr>
<th>P66</th>
<th>IO+CPU</th>
<th>100 OCTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>P60</td>
<td>IO+CPU</td>
<td>400 OCTAL</td>
</tr>
<tr>
<td>P50</td>
<td>IO-CPU</td>
<td>1,000 OCTAL</td>
</tr>
</tbody>
</table>

AFWL ACCOUNT CARD (2nd Card in Deck)

Example: Account BSMBDM, WDNA14V6-SGC, BDM, 703-821-4223.

Account - Card Identifier
BSMBDM - Means Nothing
WDNA14V6 - Account
SGC - Password
BDM - Not needed, but it identifies us
703-821-4223 - Not needed, but is phone number of user in case the need to call user.
<table>
<thead>
<tr>
<th>PERM FILE NAME</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MADEM INITBIN</td>
<td>CURRENT INITBIN BINARY FILE</td>
</tr>
<tr>
<td>MADEM RUNBIN</td>
<td>CURRENT RUNBIN BINARY FILE</td>
</tr>
<tr>
<td>MADEM HISTBIN</td>
<td>CURRENT HISTBIN BINARY FILE</td>
</tr>
<tr>
<td>DATFILE AAA</td>
<td>DATFILE INPUT FOR AAA RUN</td>
</tr>
<tr>
<td>DATFILE TEST 1</td>
<td>DATFILE INPUT FOR TEST RUN</td>
</tr>
<tr>
<td>UOIL AAA</td>
<td>UOIL INPUT FOR AAA RUN</td>
</tr>
<tr>
<td>UOIL TEST 1</td>
<td>UOIL INPUT FOR TEST RUN</td>
</tr>
<tr>
<td>MIDASTABLE SOURCE</td>
<td>SOURCE FOR FULL MIOAS TABLE</td>
</tr>
<tr>
<td>MADEM IDAS TABLES</td>
<td>GENERATED MIDAS TABLES</td>
</tr>
<tr>
<td>ICOMPBIN</td>
<td>CURRENT BINARY FOR COMPARE 21SPACES</td>
</tr>
<tr>
<td>NIPULSTOR</td>
<td>BINARY FOR NIPULSTOR-DEBUG ROUTINE</td>
</tr>
<tr>
<td>KOMMONBIN</td>
<td>BINARY FOR COMMON ANALYZER</td>
</tr>
<tr>
<td>MADEM ONE PLAN 1 AAA</td>
<td>HOLD FILE LAST MADEM I INITBIN AAA RUN</td>
</tr>
<tr>
<td>MADEM ONE PLAN 2 AAA</td>
<td>HOLD FILE FROM CAST MADEM F AAA INITBIN RUN</td>
</tr>
<tr>
<td>MADEMONE INITBIN</td>
<td>MADEM I INITBIN BINARY, (UP TO M1)</td>
</tr>
<tr>
<td>MADEM ONE RUNBIN</td>
<td>MADEM I RUNBIN BINARY, (UP TO M1)</td>
</tr>
<tr>
<td>MADEM ONE HISTBIN</td>
<td>MADEM I RUNBIN BINARY, (UP TO M1)</td>
</tr>
<tr>
<td>MADEM TWO NORTIBIN</td>
<td>MADEM II INITBIN BINARY</td>
</tr>
<tr>
<td>MADEM TWO RUNBIN</td>
<td>MADEM II RUNBIN BINARY</td>
</tr>
</tbody>
</table>
LARGE PERM FILES AT AFWL

Any perm file larger than 35 RB's (1960 PRU's) will be routinely purged at AFWL.

1 PRU = 64 Words
1 RB = 56 PRU's = 3,584 Words
35 RB's = 125,440 Words

Theoretically, one can have a large perm file saved at AFWL if it is approved by Airman Vickers.

To have a perm file over 35 RB's saved write to:

Airman Richard Vickers
AFWL LADPO
Kirtland AFB
New Mexico 87117
(505) 264-7984

With the following information:
1. Justification for the large file
2. How long the file is to be saved
3. The name of the file
4. Your account number
5. The cycle numbers of that file to be protected.

To give a better probability to your request being approved send a cc to:

Pat Smari She works in the software consultine office and has
AFWL/ADSD promised to help us to get approval. Kirtland AFB
New Mexico 87117
(505) 264-0831

REGISTERING TAPES AT AFWL

AFWL Owned Tapes

To rent an AFWL owned tape simply call the AFWL Tape Librarian at:
(505) 264-0225

Have prepared the same information as is needed when sending a BDM tape (see below).

BDM Owned Tapes

If you don't have a tape, get one from Phuoy and send it to:

AFWL/ADPO
Kirtland AFB
New Mexico 87117
Attention: EXPEDITOR
Include the following information:

1. A 10-character tape name beginning with BDMV, C.I.E. BDMUMADEM 6)
2. The tape density
3. Number of tracks on tape
4. Your phone number (they will call with the USN)
<table>
<thead>
<tr>
<th>VSN</th>
<th>TAPE_NAME</th>
<th>TRACK</th>
<th>DENS</th>
<th>OWNED BY</th>
<th>LOCATION</th>
<th>DATE ACQUIRED</th>
<th>CONTAINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>JH66</td>
<td>BDMUMADEM 1</td>
<td>NT</td>
<td>HD</td>
<td>BDM</td>
<td>BDM</td>
<td></td>
<td>MADEM I SOURCE</td>
</tr>
<tr>
<td>JH67</td>
<td>BDMUMADEM 2</td>
<td>NT</td>
<td>HD</td>
<td>BDM</td>
<td>BDM</td>
<td></td>
<td>MADEM I SOURCE</td>
</tr>
<tr>
<td>JH68</td>
<td>BDMUMADEM 3</td>
<td>NT</td>
<td>HD</td>
<td>BDM</td>
<td>BDM</td>
<td></td>
<td>MADEM I PL</td>
</tr>
<tr>
<td>BC59</td>
<td></td>
<td>MT</td>
<td>HY</td>
<td>AFWL</td>
<td>AFWL</td>
<td></td>
<td>MADEM II PL</td>
</tr>
<tr>
<td>BC87</td>
<td></td>
<td>MT</td>
<td>HY</td>
<td>AFWL</td>
<td>AFWL</td>
<td></td>
<td>MADEL II PL</td>
</tr>
<tr>
<td>AB31</td>
<td>BDMUMADEM 1</td>
<td>MT</td>
<td>HY</td>
<td>AFWL</td>
<td>AFWL</td>
<td>7/79</td>
<td>MADEM III PL-Backup</td>
</tr>
<tr>
<td>AB33</td>
<td>BDMUMADEM 2</td>
<td>MT</td>
<td>HY</td>
<td>AFWL</td>
<td>AFWL</td>
<td>7/79</td>
<td>MADEM III PL-Backup</td>
</tr>
<tr>
<td>AD78</td>
<td>BDMUMADEM 3</td>
<td>MT</td>
<td>HY</td>
<td>BDM (24090)</td>
<td>AFWL</td>
<td>7/79</td>
<td>MADEM I PL-Most Recent</td>
</tr>
<tr>
<td>AD87</td>
<td>BDMUMADEM 4</td>
<td>MT</td>
<td>HY</td>
<td>BDM (24091)</td>
<td>AFWL</td>
<td>7/79</td>
<td>MADEM III PL-Backup</td>
</tr>
<tr>
<td>AI36</td>
<td>BDMUMADEM 5</td>
<td>MT</td>
<td>HY</td>
<td>AFWL</td>
<td>AFWL</td>
<td>8/79</td>
<td>MADEM III PL-Current</td>
</tr>
<tr>
<td>A176</td>
<td>BDMUMADEM 6</td>
<td>MT</td>
<td>HY</td>
<td>AFWL</td>
<td>AFWL</td>
<td>9/79</td>
<td>NOT USED</td>
</tr>
</tbody>
</table>

NT = Nine Track  HD = 800 CPI  PL = Program Library
MT = Seven Track  HY = 800 BPI
HRLY RATES

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>CONNECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFB</td>
<td>170/HR $5/HR, 7 A.M.-7 P.M.</td>
</tr>
<tr>
<td>MFX</td>
<td>700/HR $10/HR, 7 A.M.-7 P.M.</td>
</tr>
</tbody>
</table>

BILLING BASED ON SYSTEM RECORDS
PERMFILE STORAGE FREE

BILLING QUESTIONS

Mr. Elsberrid (505) 264-0831
(CALL OR WRITE FOR DAY FILES)

AFWL/ADS
Attention: Consulting Office
Kirtland AFB
New Mexico 87117

In particular call on accounting program errors for monthly bills call Dan Thornburg at (505) 264-0208.

HOW TO RUN MADEM

MADEM is currently set up to run on either the X or Y mainframe at AFWL.

To run MADEM you need on perm files:

1. UOIL (input data)
2. DATFILE (input data)
3. INITBIN (pre-processor binary file)
4. RUNBIN (main binary file)
5. HISTRIN (post-processor binary file)

MADEM is run in 3 stages; the middle stage is generally run in 6 to 12 steps, or volumes.

The 3 stages must be run in the following order: INITBIN, then RUNBIN once for each volume, then HISTBIN.

The complete run will result in 2 printouts plus one printout (the event trace) for each volume.

See Figure 1 for a run diagram, and the JCL listings for examples of JCL desks used for a run.
INITBIN

INITBIN is MADEM's pre-processor. It processes the DATFILE and UOIL, and then plans the red raid, all the while initializing 1 space and the common blocks. The 2 hold files save 1 space and the common blocks for subsequent volume runs. INITBIN also outputs a history file for the post-processor.

The Binary file INITBIN is stored on a perm file under the name 'MADEM-INITBIN' for MADEM III.

INITBIN INPUT:

1. DATFILE - Tape 8 - Perm File name is 'DATFILEAAA'
2. UOIL - Tape 7 - Perm File name is 'UOILAAA'
3. Input deck with parameters:
   Card 1: (Unformatted, Mandatory)
   Parm 1 - must be INTEGER 1 for INITBIN
   Parm 2 - INTEGER, size of 1 space (max) during INITBIN
   Parm 3 - Max number of players on either side
   Parm 4 - Not used in INITBIN, max CPU time (real)
   Card 2: To the last card:
   The second set of options are all optional. Each of these parms must begin on column 10x+l, where x=(07). There may be from 1 to 8 RARMS per card, with as many cards as are necessary
  Parms:
   'DEBUG=ON' - Turns on printing or subroutine trace messages.
   'DUMP=ON' - Will dump 1 space at end of run.
   'DATFILE=ON' - Turns on display of DATFILE DATA structure
   'STOP=ODAT' - Stop INITBIN after DATFILE
   'STOP=UOIL' - Stops INITBIN after semant (UOIL)
   'STOP=DEL' - Stop INITBIN after DELADD, before executing plan event.

INITBIN OUTPUT:

1. 2 Hold files with 1 space, common blocks (Tape 10, Tape 11)
2. 1 File for history processing (Tape 4)
3. Printout, which includes:
   1) Plan of red raid
   2) Datfile structure pointers
   3) UOIL ECHOS:
      • commander-subordinate relationships
      • hex number for each unit, RED and BLUE
      • primary target line (PTL) values for each applicable BLUE unit
      • description of corridor block
      • hex and C^ pointers for each unit, RED and BLUE
4) Common block dump
5) Other data, according to parameters, which may include:
   • subroutine trace messages
   • 1 space dump
   • data structure dumps

Perm Files Names for INITBIN output:

TAPE 10 - PLAN 1 AAA
TAPE 11 - PLAN 2 AAA
TAPE 4  - HSTPLAN AAA
WBDBSMBM,ST176,T1177,10177,P60,EC200. MADEM INITBIN RUN (AAA)
ACCOUNT BSMBDM,WDNA14V6-SGC,BDM,703-821-4223. B. MACALEER
COMMENT. ****************************************************
COMMENT. * MADEM INITBIN RUN: *
COMMENT. * READS DATFILE, UOIL INPUT *
COMMENT. * GENERATES ISPACE *
COMMENT. * PLANS RED RAID *
COMMENT. * SAVES ISPACE ON HOLD FILES *
COMMENT. * SAVES COMMON BLOCKS ON HOLD FILES *
COMMENT. *
COMMENT. * FILES:
COMMENT. * TAPE4 - HISTORY FILE *
COMMENT. * TAPE6 - PRINTED OUTPUT *
COMMENT. * TAPE7 - UOIL INPUT *
COMMENT. * TAPE10 - FIRST HOLD FILE *
COMMENT. * TAPE11 - SECOND HOLD FILE *
COMMENT. * TAPE14 - PRINTED OUTPUT *
COMMENT. *
COMMENT. ****************************************************
REQUEST,TAPE4,*PF.
REQUEST,TAPE10,*PF.
REQUEST,TAPE11,*PF.
ATTACH,TAPE8,CDAT78A,ID=WDNA14V6.
ATTACH,XBIN,MADEMINITBIN,ID=WDNA14V6,MR=1.
LDSET,PRESET=ZERO,MAP=SBEX.
LOAD,XBIN.
EXECUTE.
DMP,100,7200.
REWIND,TAPE4.
REWIND,TAPE6.
REWIND,TAPE10.
REWIND,TAPE11.
REWIND,TAPE14.
COMMENT. ****************************************************
COMMENT. * SAVE HOLD FILES, HISTORY FILE *
COMMENT. ****************************************************
CATALOG,TAPE10,PLAN1AAA,ID=WDNA14V6,RP=999.
CATALOG,TAPE11,PLAN2AAA,ID=WDNA14V6,RP=999.
CATALOG,TAPE4,HISTPLANAAA,ID=WDNA14V6,RP=999.
COMMENT. COPYBF,TAPE14,OUTPUT.
COPEBF,TAPE6,OUTPUT.
AUDIT,ID=WDNA14V6.
COMMENT. ****************************************************
COMMENT. * IF WE BOMBED, GET OUTPUT ANYWAY *
COMMENT. ****************************************************
EXIT.
DMP,100,7200.
REWIND,TAPE14.
COPYBF,TAPE14,OUTPUT.
REWIND,TAPE6.
COPYBF,TAPE6,OUTPUT.
COMMENT.
AUDIT,ID=WDNA14V6.
COMMENT. ********************************************************************
COMMENT.*
COMMENT.* FIRST INPUT CARD IS MANDATORY AND *
COMMENT.* HOLDS 4 PARAMETERS:*
COMMENT.* 1. IOP - MUST BE 1 FOR INITBIN *
COMMENT.* 2. MSPCE - SIZE OF ISPACE *
COMMENT.* 3. MAX NO. PLAYERS ON ONE SIDE *
COMMENT.* 4. MAX CPU TIME OF THIS RUN *
COMMENT.*
COMMENT.* THE SECOND INPUT CARDS ARE OPTIONAL *
COMMENT.* AND CAN HOLD THESE PARAMETERS: *
COMMENT.* DEBUT=ON - SUBROUTINE TRACE *
COMMENT.* MESSAGES ARE PRINTED *
COMMENT.* DUMP=ON - WILL DUMP ISPACE *
COMMENT.* DATFILE=ON - WILL DISPLAY DATFILE *
COMMENT.* STOP=ODAT - STOPS INITBIN AFTER *
COMMENT.* PROCESSING DATFILE, BEFORE UOIL *
COMMENT.* STOP=UOIL - STOPS AFTER UOIL *
COMMENT.* STOP=DEL - STOPS AFTER DELADD *
COMMENT.*
COMMENT. ********************************************************************
&
EDR
1,50000,600,90.
DEBUG=ON
DUMP=ON
DATFILE=ON
STOP=ODAT
#
EO1
RUNBIN

A full production run is generally accomplished in volumes, but may be done in one run if desired. The length of a volume is controlled by the fourth input parameter, which tells MADEM how much CPU time to use before stopping. Using 90 seconds, most MADEM runs take six to twelve volumes. The hold files between each volume are saved so that the user can rerun any given volume. This is done by changing the 'ATTACK' and 'CATALOG' cards (in the JCL deck) after each volume run. The hold files are numbered within the Perm File name to identify the volume that created them. For example, files AAAV10, AAAV11, AAAV12, and AAAV13 are hold files created by volume 1. The 'AAA' stands for run type AAA. There are currently input files on cards for 5 runs, called Runs AAA, BBB, CCC, DDD, and EEE.

Perm File Name for RUNBIN: 'MADEMRUNBIN'

RUNBIN INPUT:

1. 4 Hold files - TAPE15, TAPE16, TAPE17, TAPE18.
   (For a Volume 1 Run use only TAPE15 and TAPE16).
2. Input deck with parameters:
   Card 1:
   PARM 1 - Must be INTEGER '2' for RUNBIN
   PARM 2 - INTEGER, MAX size of ISPACE
   PARM 3 - INTEGER, MAX number of players on either side.
   PARM 4 - Real, MAX CPU time of volume (in seconds)
   Card 2 to the last card:
   Same as INITBIN, but only DUMP=ON and DEBUG=ON are effective parameters.

RUNBIN OUTPUT:

1. 4 Hold files - TAPE10, TAPE11, TAPE12, TAPE13
2. TAPE4 - for the post processor
3. Event trace listing (printed)
4. Common block dump (printed)
5. DEBUG messages (if chosen)
6. ISPACE 'DUMP' (if chosen)

250
WBOMBMS, T176, T177, I0177, P60, EC400.  MADEM PRODUCTION RUN (AAA)
ACCOUNT BSBDM, WDNA14V6-5GC, BDM, 703-821-4223.  B. MACALEER

COMMENT.  **************************************************************
COMMENT. *
COMMENT. * MADEM PRODUCTION RUN *
COMMENT. *
COMMENT.  **************************************************************
COMMENT. * FILES: *
COMMENT. *
COMMENT. * TAPE4 - HISTORY FILE *
COMMENT. *
COMMENT. * TAPE6 - PRINTED OUTPUT *
COMMENT. *
COMMENT. * TAPE10 - FIRST HOLD FILE *
COMMENT. *
COMMENT. * TAPE11 - SECOND HOLD FILE *
COMMENT. *
COMMENT. * TAPE12 - THIRD HOLD FILE *
COMMENT. *
COMMENT. * TAPE13 - FOURTH HOLD FILE *
COMMENT. *
COMMENT. * TAPE14 - PRINTED OUTPUT *
COMMENT. *
COMMENT.  **************************************************************
REQUEST, TAPE4*PF.
REQUEST, TAPE10,*PF.
REQUEST, TAPE11,*PF.
REQUEST, TAPE12,*PF.
REQUEST, TAPE13,*PF.

COMMENT. ATTACK,BIN, MADEMRUNBIN, ID=WDNA14V6, MR=1.
LDSET, PRESET=ZERO.
LOAD,BIN.
EXECUTE, PL=20000.
REWIND, TAPE4.
REWIND, TAPE6.
REWIND, TAPE10.
REWIND, TAPE11.
REWIND, TAPE12.
REWIND, TAPE13.
REWIND, TAPE14.

COMMENT.  **************************************************************
COMMENT. *SAVE ISPACE, COMMONS *
COMMENT. *SAVE TAPE4 FOR HISTORY PROCESSING *
COMMENT.  **************************************************************
CATALOG, TAPE10, AAAV10, ID=WDNA14V6, RP=999.
CATALOG, TAPE11, AAAV11, ID=WDNA14V6, RP=999.
CATALOG, TAPE12, AAAV12, ID=WDNA14V6, RP=999.
CATALOG, TAPE13, AAAV13, ID=WDNA14V6, RP=999.
CATALOG, TAPE4, HS1AAAV1, ID=WDNA14V6, RP=999.

COMMENT. DMP,100,7200.
COMMENT.  **************************************************************
COMMENT. *TAPE14 USUALLY NULL *
COMMENT.  **************************************************************
COPYBF, TAPE14, OUTPUT.
COMMENT. 251
COMMENT. ********************************************
COMMENT. * TAPE6 HAS EVENTS, DUMPS (IF ANY) *
COMMENT. ********************************************
COPYBF,TAPE6,OUTPUT.
COMMENT.
AUDIT,ID=WDNA14V6.
COMMENT. ********************************************
COMMENT. * IF WE BOMB, PRINT OUTPUT ANYWAY *
COMMENT. ********************************************
EXIT.
DMP,100,7200.
REWIND,TAPE14.
COPYBF,TAPE14,OUTPUT.
REWIND,TAPE6.
COPYBE,TAPE6,OUTPUT.
AUDIT,ID=WDNA14V6.
COMMENT. ********************************************
COMMENT. * FIRST INPUT CARD IS MANDATORY AND *
COMMENT. * HOLDS 4 PARAMETERS: *
COMMENT. * 1. IOP - MUST BE 2 FOR RUNBIN *
COMMENT. * 2. MSPCE - SIZE OF ISPACE *
COMMENT. * 3. MAX NO. PLAYERS ON ONE SIDE *
COMMENT. * 4. MAC CPU TIME OF THIS RUN *
COMMENT. *
COMMENT. * THE SECOND INPUT CARDS ARE OPTIONAL *
COMMENT. * AND CAN HOLD THESE PARAMETERS: *
COMMENT. * DEBUG=ON - SUBROUTINE TRACE *
COMMENT. * MESSAGES ARE PRINTED *
COMMENT. * DUMP=ON - WILL DUMP ISPACE *
COMMENT. *
COMMENT. ********************************************
& EOR
2,131000,600,90.
DEBUG=ON
DUMP=ON
# E01
CARDS FOR VOLUMES:
COMMENT. 
COMMENT. *********************************
CINEBT, * VOLUME 2, RUN TYPE AAA *
COMMENT. *********************************
COMMENT. 
ATTACH, TAPE15, AAV10, ID=WDNA14V6.
ATTACH, TAPE16, AAV11, ID=WDNA14V6.
ATTACH, TAPE17, AAV12, ID=WDNA14V6.
ATTACH, TAPE18, AAV13, ID=WDNA14V6.
CATALOG, TAPE10, AAV20, ID=WDNA14V6, RP=999.
CATALOG, TAPE11, AAV21, ID=WDNA14V6, RP=999.
CATALOG, TAPE12, AAV22, ID=WDNA14V6, RP=999.
CATALOG, TAPE13, AAV23, ID=WDNA14V6, RP=999.
CATALOG, TAPE4, HSTAAAV2, ID=WNDA14V6, RP=999.
COMMENT.
COMMENT. *********************************
COMMENT. *VOLUME 3, RUN TYPE AAA *
COMMENT. *********************************
COMMENT. 
ATTACH, TAPE15, AAV20, ID=WNDA14V6.
ATTACH, TAPE16, AAV21, ID=WNDA14V6.
ATTACH, TAPE17, AAV22, ID=WNDA14V6.
ATTACH, TAPE18, AAV23, ID=WNDA14V6.
CATALOG, TAPE10, AAV30, ID=WNDA14V6, RP=999.
CATALOG, TAPE11, AAV31, ID=WNDA14V6, RP=999.
CATALOG, TAPE12, AAV32, ID=WNDA14V6, RP=999.
CATALOG, TAPE13, AAV33, ID=WNDA14V6, RP=999.
CATALOG, TAPE4, HSTAAAV3, ID=WNDA14V6, RP=999.
COMMENT.
COMMENT. *********************************
COMMENT. *VOLUME 4, RUN TYPE AAA *
COMMENT. *********************************
COMMENT. 
ATTACH, TAPE15, AAV30, ID=WNDA14V6.
ATTACH, TAPE16, AAV31, ID=WNDA14V6.
ATTACH, TAPE17, AAV32, ID=WNDA14V6.
ATTACH, TAPE18, AAV33, ID=WNDA14V6.
CATALOG, TAPE10, AAV40, ID=WNDA14V6, RP=939.
CATALOG, TAPE11, AAV41, ID=WNDA14V6, RP=999.
CATALOG, TAPE12, AAV42, ID=WNDA14V6, RP=999.
CATALOG, TAPE13, AAV43, ID=WNDA14V6, RP=999.
CATALOG, TAPE4, HSTAAAV4, ID=WNDA14V6, RP=999.
COMMENT.
COMMENT. *********************************
COMMENT. *VOLUME 5, RUN TYPE AAA *
COMMENT. *********************************
COMMENT. 
ATTACH, TAPE15, AAV40, ID=WNDA14V6.
ATTACH, TAPE16, AAV41, ID=WDNA14V6.
ATTACH, TAPE17, AAV42, ID=WDNA14V6.
ATTACH, TAPE18, AAV43, ID=WDNA14V6.
CATALOG, TAPE10, AAV50, ID=WDNA14V6, RP=999.
CATALOG, TAPE11, AAV51, ID=WDNA14V6, RP=999.
CATALOG, TAPE12, AAV52, ID=WDNA14V6, RP=999.
CATALOG, TAPE13, AAV53, ID=WDNA14V6, RP=999.
CATALOG, TAPE4, HSTAAV5, ID=WDNA14V6, RP=999.
COMMENT.
COMMENT. ********************************************
COMMENT. * VOLUME 6, RUN TYPE AAA           *
COMMENT. ********************************************
COMMENT.
ATTACH. TAPE15, AAV50, ID=WDNA14V6
HISTBIN

After all volumes have been run, the post processor can be run to summarize the outcome of events.

PERM FILE name of BINARY FILE: HISTBIN

INPUT:

(1) TAPE4 - A concatenation of all post-processor (history) files which includes exactly one file for each volume plus one file from INITBIN

(2) Two card input deck of PARMS, as in JCL example.

OUTPUT:

Printed summary of the MADEM RUN.

NOTE: Little is known about HISTBIN AT THIS TIME.
WBDMBSM, ST176, T177, 10177, P60. MADEM HISTORY PROCESSING ACCOUNT BSBMOM, WDNA14V6-SCG, BDM, 703-821-4223. B. MACALEER
ATTACH, LGO, MADEMHISTBIN, ID=WDNA14V6, MR=1.
ATTACH, PLAN, HSTPLANAAA, ID=WDNA14V6.
ATTACH, V1, HSTAAAV1, ID=WDNA14V6.
ATTACH, V2, HSTAAAV2, ID=WDNA14V6.
ATTACH, V4, HSTAAAV4, ID=WDNA14V6.
ATTACH, V5, HSTAAAV5, ID=WDNA14V6.
ATTACH, V6, HSTAAAV6, ID=WDNA14V6.
ATTACH, V7, HSTAAAV7, ID=WDNA14V6.
ATTACH, V8, HSTAAAV8, ID=WDNA14V6.
ATTACH, V9, HSTAAAV9, ID=WDNA14V6.
COPYBR, PLAN, TAPE4.
COPYBR, V1, TAPE4.
COPYBR, V2, TAPE4.
COPYBR, V3, TAPE4.
COPYBR, V4, TAPE4.
COPYBR, V5, TAPE4.
COPYBR, V6, TAPE4.
COPYBR, V7, TAPE4.
COPYBR, V8, TAPE4.
COPYBR, V9, TAPE4.
REWIND, TAPE4.
LDSET, PRESET=ZERO.
LOAD, LGO.
EXECUTE.
REWIND, TAPE4.
COPYSBF, TAPE4, OUTPUT.
AUDIT, ID=WDNA14V6.
EXIT.
AUDIT, ID=WDNA14V6.
& NOR
CONVENTIONAL 1978 AAA LEVEL 2 PK
39000. .999999999. .999999999. .999999999. .999999999. .999999999.
# EOI
RUNBIN

A full production run is generally accomplished in volumes, but may be
done in one run if desired. The length of a volume is controlled by the
fourth input parameter, which tells MADEM how much CPU time to use before
stopping. Using 90 seconds, most MADEM runs take six to twelve volumes.
The hold files between each volume are saved so that the user can rerun any
given volume. This is done by changing the 'ATTACK' and 'CATALOG' cards
(in the JCL deck) after each volume run. The hold files are numbered
within the Perm File name to identify the volume that created them. For
example, files AAV10, AAV11, AAV12, and AAV13 are hold files created by
volume 1. The 'AAA' stands for run type AAA. There are currently input
files on cards for 5 runs, called Runs AAA, BBB, CCC, DDD, and EEE.

Perm File Name for RUNBIN: 'MADEMRUNBIN'

RUNBIN INPUT:

1. 4 Hold files - TAPE15, TAPE16, TAPE17, TAPE18.
   (For a Volume 1 Run use only TAPE15 and TAPE16).
2. Input deck with parameters:
   Card 1:
   PARM 1 - Must be INTEGER '2' for RUNBIN
   PARM 2 - INTEGER, MAX size of ISPACE
   PARM 3 - INTEGER, MAX number of players on either side.
   PARM 4 - Real, MAX CPU time of volume (in seconds)
   Card 2 to the last card:
   Same as INITBIN, but only DUMP=ON and DEBUG=ON are effective
   parameters.

RUNBIN OUTPUT:

1. 4 Hold files - TAPE10, TAPE11, TAPE12, TAPE13
2. TAPE4 - for the post processor
3. Event trace listing (printed)
4. Common block dump (printed)
5. DEBUG messages (if chosen)
6. ISPACE 'DUMP' (if chosen)
WBDMBSM,ST176,T177,I0177,P60,EC400. MADEM PRODUCTION RUN (AAA)
ACCOUNT BSBDM,WDNA14V6-SGC,BDM,703-821-4223. B.MACALEER
AUDIT, ID=WDNA14V6.
COMMENT.
COMMENT. *****************************************************************************
COMMENT. * PURGE INITBIN'S OUTPUT FILES *
COMMENT. *****************************************************************************
COMMENT.
PURGE, PLAN1, PLAN1AAA, ID=WDNA14V6, LC=1.
PURGE, PLAN2, PLAN2AAA, ID=WDNA14V6, LC=1.
PURGE, HSTPLANAAA, ID=WDNA14V6, LC=1.
COMMENT. *****************************************************************************
COMMENT. * PURGE VOLUME 1 OUTPUT FILES *
COMMENT. *****************************************************************************
PURGE, AAVV10, ID=WDNA14V6, LC=1.
PURGE, AAVV11, ID=WDNA14V6, LC=1.
PURGE, AAVV12, ID=WDNA14V6, LC=1.
PURGE, AAVV13, ID=WDNA14V6, LC=1.
PURGE, HSTV1, HSTAAAV1, ID=WDNA14V6, LC=1.
COMMENT. *****************************************************************************
COMMENT. * PURGE VOLUME 2 OUTPUT FILES *
COMMENT. *****************************************************************************
PURGE, AAVV20, ID=WDNA14V6, LC=1.
PURGE, AAVV21, ID=WDNA14V6, LC=1.
PURGE, AAVV22, ID=WDNA14V6, LC=1.
PURGE, AAVV23, ID=WDNA14V6, LC=1.
PURGE, HSTV2, HSTAAAV2, ID=WDNA14V6, LC=1.
COMMENT. *****************************************************************************
COMMENT. * PURGE VOLUME 3 OUTPUT FILES *
COMMENT. *****************************************************************************
PURGE, AAVV30, ID=WDNA14V6, LC=1.
PURGE, AAVV31, ID=WDNA14V6, LC=1.
PURGE, AAVV32, ID=WDNA14V6, LC=1.
PURGE, AAVV33, ID=WDNA14V6, LC=1.
PURGE, HSTV3, HSTAAAV3, ID=WDNA14V6, LC=1.
COMMENT. *****************************************************************************
COMMENT. * PURGE VOLUME 4 OUTPUT FILES *
COMMENT. *****************************************************************************
PURGE, AAVV40, ID=WDNA14V6, LC=1.
PURGE, AAVV41, ID=WDNA14V6, LC=1.
PURGE, AAVV42, ID=WDNA14V6, LC=1.
PURGE, AAVV43, ID=WDNA14V6, LC=1.
PURGE, HSTV4, HSTAAAV4, ID=WDNA14V6, LC=1.
COMMENT. *****************************************************************************
COMMENT. * PURGE VOLUME 5 OUTPUT FILES *
COMMENT. *****************************************************************************
PURGE, AAVV50, ID=WDNA14V6, LC=1.
PURGE, AAVV51, ID=WDNA14V6, LC=1.
PURGE, AAVV52, ID=WDNA14V6, LC=1.
PURGE,AAA53,ID=WDNA14V6,LC=1.
Purge,HSTV5,HSTAAAV5,ID=WDNA14V6,LC=1.
COMMENT. **************************************************************************
COMMENT. * PURGE VOLUME 6 OUTPUT FILES *
Purge,AAA560,ID=WDNA14V6,LC=1.
Purge,AAA561,ID=WDNA14V6,LC=1.
Purge,AAA562,ID=WDNA14V6,LC=1.
Purge,AAA563,ID=WDNA14V6,LC=1.
Purge,HASTV6,HSTAAAV6,ID=WDNA14V6,LC=1/
AUDIT,ID=WDNA14V6.
& EOR
# EOI
The MIDAS BIN code is called "MIDAS' on account BDMAFM at AFWL.

**JCL**: MIDAS, COMPIL, XXX. this run a file called 'COMPIL'
OPTION: PL=nnn (generally the output from UPDATE) 
where nnn=line limit of through MIDAS and puts the compilable 
FTNOUT FILE. Default: nnn=5,000. default for XXX is FTNOUT.

**MADEM MIDAS TABLES**

(1) "4" Data Structures
(2) All Data Structures

**PROGRAM LIBRARY DECK NAME CHANGES**

The following subroutines were listed in decks on the program library whose
deck name did not match the subroutine name. The deck names have been
changed to match the subroutine names. In some MADEM I and MADEM II
listings. However, these routines are filed under the old deck name.

<table>
<thead>
<tr>
<th>SUBROUTINE</th>
<th>OLD DECK NAME</th>
<th>NEW DECK NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB2CRC</td>
<td>ABMSG</td>
<td>AB2CRC</td>
</tr>
<tr>
<td>BNCONTIC</td>
<td>NBNCONT</td>
<td>BNCONTIC</td>
</tr>
<tr>
<td>BNLALLE</td>
<td>NBNLALL</td>
<td>BNLALLE</td>
</tr>
<tr>
<td>BTN2CRC</td>
<td>BTNMSG</td>
<td>BTN2CRC</td>
</tr>
<tr>
<td>BYALCOV</td>
<td>PROCCLU</td>
<td>BYALCOV</td>
</tr>
<tr>
<td>BYCONTIC</td>
<td>NBYCONT</td>
<td>BYCONTIC</td>
</tr>
<tr>
<td>BYTKCHK</td>
<td>NBYTKCH</td>
<td>BYTKCHK</td>
</tr>
<tr>
<td>CRC2INT</td>
<td>INTASIN</td>
<td>CRC2INT</td>
</tr>
<tr>
<td>HANDZPT</td>
<td>NHANDZP</td>
<td>HANDZPT</td>
</tr>
<tr>
<td>IN RANGE</td>
<td>NINRANG</td>
<td>IN RANGE</td>
</tr>
<tr>
<td>INTASIN</td>
<td>BADASIN</td>
<td>INTASIN</td>
</tr>
<tr>
<td>INT2CRC</td>
<td>INTMSG</td>
<td>INT2CRC</td>
</tr>
<tr>
<td>RELOCAT</td>
<td>NRELOC</td>
<td>RELOCAT</td>
</tr>
<tr>
<td>SAMPRCM</td>
<td>NSAMPRC</td>
<td>SAMPRCM</td>
</tr>
<tr>
<td>TRKCHEK</td>
<td>NTRKCHE</td>
<td>TRKCHEK</td>
</tr>
</tbody>
</table>

The only remaining subroutines that do not occupy a deck of the same name
are the VOIL routines in decks ULSUB2 and UOILSUB.
STRAY SUBROUTINES

The following subroutines reside in the noted deck on the program library. These routines are standard UOIL routines, and need not be altered for MADEM. These routines are represented on our master list by deck name, and may be filed as such in the books.

<table>
<thead>
<tr>
<th>DECK UOILSUB</th>
<th>DECK ULSUB2</th>
</tr>
</thead>
<tbody>
<tr>
<td>APCEL1</td>
<td>ADDCHR</td>
</tr>
<tr>
<td>APCEL2</td>
<td>ERROR</td>
</tr>
<tr>
<td>CARD</td>
<td>EXTSCN</td>
</tr>
<tr>
<td>CHRGEN</td>
<td>LEXAN</td>
</tr>
<tr>
<td>LACELL</td>
<td>LOOKUP</td>
</tr>
<tr>
<td>NXTSYM</td>
<td>LRKPRS</td>
</tr>
</tbody>
</table>

261
APPENDIX B
MODULAR INFORMATION DATA ACCESS SYSTEM (MIDAS)

A. INTRODUCTION

MIDAS consists of two parts: the MIDAS language and the MIDAS Translator. The MIDAS language allows one to write programs using only the logical aspects of data structures. The MIDAS Translator reads data structure definitions and programs written in the MIDAS language, realizes the physical implementation of the logical data structures, and generates a FORTRAN program or subprogram as output.

The advantages of using MIDAS are twofold. First, the programmer is no longer concerned with the details of data structure implementation, and is free to use natural names for elements of the data structures. Second, a program written in MIDAS is easier to convert to another type of computer since the logical definition of the data structures do not change, only the physical implementation which is now completely mechanized.

The MIDAS Translator is controlled by a set of tables which define the logical data structures to be translated and their specific physical implementation. Thus, each programming project will utilize a different set of tables, corresponding to the data structures peculiar to that project.

The MIDAS Translator constructs these tables automatically using information supplied by the user. This definition information is expressed using a Data Structure Definition Language (DSDL) which defines the logical data structures and their physical implementation details. This language can also be used to uniquely specify and document the logical design of data structures during the preliminary and detailed design phases of a project.

B. THE DATA STRUCTURE DEFINITION LANGUAGE

Data structures are closely related to sets; therefore, we will use concepts and notations adapted from set theory for describing the logical aspects of data structures. A data structure contains members, in the
same way that a set has members (we are excluding the null set in this
discussion since a data structure with no members, and hence no informa-
tion, is not of practical value). We will assign unique names to all data
structures. We may assign arbitrary names to the data structure members,
provided we do not use any of the data structure names. Thus, for example,
we can write:

\[
A = (A_1, A_2, A_3, A_4) \\
B = (B_1, B_2, B_3, B_4, B_5) \\
C = (DOG)
\]

Here we have defined three data structures, A, B, and C, with members as
shown.

We impose the restriction that a data structure member may not be a
data structure. This means that the following definition is illegal:

\[
A = (B, C) \\
B = (D, E)
\]

This restriction is imposed by the current capability of the MIDAS Transla-
tor and is expected to be removed at a later date.

We also note that the logical definition of a data structure does not
require the members to be ordered. For example, the following two defini-
tions of data structure A are logically equivalent:

\[
A = (B, C) \\
A = (C, B)
\]

However, physical implementation of data structures using the MIDAS Trans-
lator will require that the members be ordered. Therefore, we will assume
that the members are ordered for this reason. The order is that in which
the members are listed in the data structure definition. Since there is
only one definition for each data structure, there is no possibility of a
conflict in order arising.

A member may have an attribute associated with it. The only attribute
which will be permitted under the initial DSDL will be the pointer attri-
bute. The pointer attribute specifies that the member is a pointer to a
specified data structure. A member with the pointer attribute is denoted
by the form \(\text{A} = \text{e} \text{B}\), where \(\text{A}\) is the member name and \(\text{B}\) is the data structure which is pointed to by \(\text{A}\). For example:

\[
\begin{align*}
\text{A} &= (E = \text{e}B, F, \text{PTRC} = \text{e}C) \\
\text{B} &= (\text{PTRUP} = \text{e}C, 81) \\
\text{C} &= (D, \text{PTRNEXT} = \text{e}A)
\end{align*}
\]

These definitions are equivalent to the following schematic representation:

Members which are data structure pointers always point to the data structure itself, not to any members of the data structure. This distinction is important to understand, since the entire MIDAS system is based on this convention. Thus, in the above example, member PTRC of data structure \(\text{A}\) points to data structure \(\text{C}\), but not to member \(D\) of data structure \(\text{C}\).

The above discussion treats only the logical aspects of data structure definition. In order for MIDAS to implement a data structure it is also necessary to specify the physical properties of the data structure. This is done for each member of the structure.

Each structure will be implemented in one or more words of storage. A member may occupy an entire word, or it may occupy a bit field within a word. In addition, the data structure itself is embedded within a FORTRAN structure such as an array or COMMON block. A means must be provided for specifying all of these physical properties.

DSOL will utilize a parenthesized notation to specify the physical properties of data structures. The FORTRAN structure which contains a logical data structure will be denoted as in the following example:

\[
\text{A(ISPACE)} = (B, C, D)
\]

This indicates that data structure \(\text{A}\) is physically contained within an array named ISPACE. Only onedimensional arrays may be used for FORTRAN structures with the current version of the MIDAS Translator.
Members of data structures are stored as one member per word, in the
same order as the members are listed in the data structure definition,
unless otherwise indicated. Thus, in the above example, data structure A
requires three words of storage space, one each for B, C, and D.

It may be desirable to pack several members into the same word, with a
specific bit field allocated for each member within the word. All of the
members which occupy the same physical word are enclosed by parentheses.
For example:

A(ISPACE) = (B, (C, D, E), F)

Here data structure A requires three words of storage, but members C, D,
and E are packed into the second word. We have shown how two or more
members are packed, but we have not indicated the bit fields which contain
them. Bit field designation is accomplished with an integer which speci-
fies the number of consecutive bits in the field. This integer is enclosed
in parentheses and immediately follows the member name (if the member name
has an attribute, the bit field information follows the attribute informa-
tion). For a given computer word, the bit fields are assigned in the order
that members are specified, starting with the leftmost bit and proceeding
to the right. It is not necessary that the bit space of the entire word be
allocated. For example:

A(ISPACE) = (B, (C(12), D(28), E=*P(20)), F)

Assuming a 60 bit word, this data structure definition can be represented
schematically as:

The Data Structure A is contained in array ISPACE. Data structure P, which
is pointed to by member E of A may or may not be in ISPACE.

All data structure members which are bit packed are assumed to be
accessed (in the FORTRAN sense) in the array name specified on the left
hand side of the data structure definition statement. Normally, this
"default" array will be of integer type, since it is unlikely that a packed word will contain floating point data. On the other hand, other members of the same data structure which occupy entire words may be of a different number type, such as real, and therefore require access using a different FORTRAN array name.

The default array name can be overridden for a particular member by following the member name with the array name required by that member; this member array name is enclosed by parentheses and applies only to that specific member. This feature is applicable only to members which occupy entire words; all members which are bit packed automatically use the default array name. For example:

\[ A(ISPACE) = (B(SPACE), (C(12), D(28), E^P(20)), F) \]

C. TABLE GENERATION FOR THE MIDAS TRANSLATOR

Before the MIDAS Translator can be used to translate source programs written in the MIDAS language, it must be told how to interpret logical data structure references and implement them in FORTRAN. This is done by an input language which includes statements written in the Data Structure Definition Language along with other information. Using this input, the MIDAS Translator builds the translation tables from scratch, or, alternatively, augments a table which already exists.

Table construction or augmentation is known as Phase I operation of the MIDAS Translator. This phase is optional in a given MIDAS run, but must have been done at least once before midas source language programs can be translated.

Phase I operation is initiated by the appearance of a special input card which has one of two forms. If the translation tables are being constructed from scratch the initial card has the form:

/MIDAS dialect

Where dialect is one of the key words CDC, IBM, MULTICS, or UNIVAC.
Thus the dialect specifies a particular computer system for which the tables will be constructed. On the other hand, if additional data structure definitions are being added to previously generated tables, MIDAS already knows which computer system is required, and the initial card has the form:

/MIDAS

The MIDAS card is followed by information which defines the number type for all arrays which are being introduced for the first time in subsequent data structure definition statements; if an array has previously been introduced in an earlier Phase 1 run, MIDAS already knows its number type and respecification of number type for that array will not be allowed. The specification of array type is needed when tables are being built for either IBM or MULTICS dialect. For CDC and UNIVAC dialect the array type information is superfluous and need not be included.

The array type information is introduced with the card:

/TYP
/DEFINE

which is followed by one or more cards containing data structure definitions, using the Data Structure Definition Language described in the previous section.

Following the data structure definition block the user may include an optional macro definition block. This block is introduced by the card:

/MACRO

which is followed by one or more macro definitions. Each macro definition is introduced by a card containing the macro name enclosed by $ characters. This is then followed by one or more card images which contain the actual macro text. The macro text may be arbitrary except that the first character on any macro text card can not be a $ or /. For example:

/MACRO
$SUES
THIS IS A MACRO
$MARY$
MARY HAD
A
LITTLE
MACRO

In this example two macros, $SUES$ and $MARY$, are defined.

The end of the input for Phase I operation is the card:

/END

Note that the only required input cards for Phase I operation are /MIDAS and /END. All other input is optional, however, at least one definition block must be included, as introduced by /TYPE, /DEFINE, or /MACRO.
If a previously generated table is being augmented, only additions are permitted to the table. No previously defined information may be changed or redefined.

D. THE MIDAS LANGUAGE

In order to illustrate the MIDAS language we define a set of data structures which will be used in conjunction with an extended example. The definitions are written using the input language for Phase I operation of the MIDAS Translator:

/MIDAS CDC
/DEFINE
SB(ABC) = (PAT=*P, (TIM(30), JOHN=*CC(IO), DON(20)), JOE=*Q)
CC(XYZ) = (TIM, (DON(30), DICK(30)))
Q(4AA) = ((TIM(20), PAT(30))
P(BB) = (TIM)
/MACRO
$SUES$
   INTEGER ABC
   DATA IDEBUG/O/
/END

These data structures have the following schematic representation:
Next we present a MIDAS subroutine which exhibits all of the features of the MIDAS language:

**SUBROUTINE PDQ (B,C,JIM,BILL,BOB)**

1

**COMMON/ARRAYS/ABC(1000),XYZ9500),AAA(200),BBB(700)**

2

**DECLARE JIM=SB, K**

3

**DECLARE J, BILL = CC**

4

**DECLARE BOB = SB**

5

$SUES$

6

$J=JIM$

7

$K=BOB.PATS$

8

100 A=B*C+J*K

9

B=$J.JOE.PATS$

10

200 $K.TIMS=3+SK.TIMS*5/SHILL.DONS$

11

RETURN

12

END

13

This subroutine is not intended to represent meaningful computation but only to serve as an example of the use of the MIDAS language. The lines of the subroutine are numbered to permit easy referencing in the discussion which follows.

The arrays which contain the data structure SB, CC, Q, and P must be available within the subroutine since they will be referenced by the FORTRAN subroutine generated by the MIDAS Translator. This is accomplished through the COMMON statement in line 2. The arrays could also have been established through the parameter list or with DIMENSION statements.

The data structures SB, CC, P, and Q are actually prototypes or templates. In practice, there will probably be many copies of a given type of data structure in use. The individual copies are distinguished from each other by the use of different pointers for each copy. Thus, we need a means of associating a particular pointer name with the type of data structure which it is pointing to. This is done with the DECLARE statement.
In line 3 we declare the FORTRAN variable JIM to be a pointer to a data structure of type SB. Similarly, in line 5 we declare BOB to also be a pointer to a data structure of type SB. JIM and BOB may point to the same instance of SB or to different instances. In line 4 we declare BILL to be a pointer to a data structure of type CC. It is necessary to establish values for JIM, BOB, and BILL before each of them is first used in a MIDAS data structure reference. In this example, these values are passed through the parameter list.

Given the name of a pointer to a data structure, we can now locate and reference the value of any member within that data structure. This is done by constructing a compound name, enclosed in $ symbols, using the (.) period symbol to separate elements of the compound name. For example, we can reference the member JOHN in data structure SB, using JIM as a pointer to SB, by the notation $JIM.JOHN$. This provides us with the actual value of JOHN. Since JOHN is a pointer to a data structure of type CC, we can then refer to member DICK of CC by the notation $JIM;JOHN;DICK$.

It is useful to have a means of working with partial compound names by establishing intermediate pointer values. This avoids having to unravel a long compound name each time the name is used, with the corresponding space and time penalties in the executing program. This can be accomplished using a pointer macro. A pointer macro establishes a MIDAS name for a pointer value which has been derived by traversing a pointer chain through the data structures. Once defined, the pointer macro can then be used in place of the compound name which it represents.

Examples of the specification of pointer macros are shown in lines 7 and 8. As can be seen, the pointer macro definition consists of a simple replacement statement enclosed by $ symbols. The left side of the replacement statement is an integer FORTRAN variable; the right side is a simple or compound name representing a data structure pointer. On the right side the simple name or the first element of the compound name may be either a data structure pointer or a previously defined pointer macro.

Once defined, a pointer macro may not be redefined. The definition remains to the end of the program or subprogram. Also, a pointer macro

272
must be defined before it is used. This requires that the definition card physically must precede any cards which use the pointer macro in the MIDAS source deck, and that the pointer macro definition statement must be executed before any statements which reference the pointer macro in the actual program execution.

In line 8 we define the pointer macro K as representing the string BOB.PAT; this will actually generate a FORTRAN variable K which contains the value of $BOB.PAT$, so that K may subsequently be used in the normal FORTRAN sense. However, since K also represents the string BOB.PAT, we note that the reference $K.TIM$ is identical to the reference $BOB.PAT.TIM$, as shown in line 11.

Similarly, we are substituting the MIDAS name J for JIM (and consequently the value of FORTRAN variable J is set to the value of FORTRAN variable JIM) as shown in line 7. This permits us to write the MIDAS name SJ.JOE.PATS in line 10, which is equivalent to SJIM.JOE.PATS.

Whenever a name is to be used as a pointer macro, this fact must be noted in a DECLARE statement. This is done by simply including the name in the list for a DECLARE statement such as is done in lines 3 and 4 for K and J, respectively. Individual items in a DECLARE statement list are separated by commas as shown.

MIDAS also permits the user to employ card macro definitions. A reference to a card macro is shown in line 6 and consists of the macro name (e.g., SUE) enclosed in $ symbols. The macro reference must not start in columns 1 through 6. Whenever a card macro is encountered, MIDAS will replace the macro with a set of one or more 80 column card images which correspond to the macro name. These card images may not contain any non-FORTRAN (i.e., "IDAS") text, since the card images are not interpreted by the MIDAS translator. The END card must not be included within the macro card set. The card macro may occur anywhere before the END card, there may be as many different card macros as desired, and the same card macro may appear several times.

There are some restrictions on ordering of MIDAS information. As discussed earlier, pointer macro definitions must precede any use of that
pointer macro both in card sequence and execution sequence. Also, the
DECLARE statement must precede the use of any item specified in that
DECLARE statement.

Whenever an END card is encountered, signifying the end of a subpro-
gram, all definitions local to that subprogram as expressed on DECLARE
cards and through pointer macro definition statements are lost. Other
subprograms may follow and will be processed in sequence, but they must
establish their own local definitions.

A word about number types: Names chosen for pointers to data struc-
tures and macro pointers (as specified in DECLARE statements) will have an
identical FORTRAN name. Since these names represent pointers, they
would be chosen so that they start with letters I through N. Otherwise, it will
be necessary to type them as INTEGER using a TYPE statement.

E. USING THE MIDAS TRANSLATOR

The MIDAS Translator is a cross-translator, i.e., it is capable of
generating output for several different computing systems including the
system it operates on. Furthermore, MIDAS is designed to operate on all of
the computing systems for which it is capable of providing translation
output. These systems are CDC 6000/7000/Cyber series with FTN FORTRAN, IBM
360/370 series with FORTRAN Level G and H, Univac 1108 with FORTRAN V, and
Honeywell MULTICS with FORTRAN.

The operating details depend in part upon the computing system on
which the MIDAS Translator is operating. However, certain aspects of
operation are common to all systems and will be discussed first.

1. General Operating Details

MIDAS Translator operation involves two phases. Phase 1
generates the MIDAS translation tables using the input language described
in Section C. For a particular programming project and target computer a
Phase 1 operation must be executed at least once. The tables which are
generated may be saved as files and used for immediate or later translation
of programs written in the MIDAS language. Furthermore, the tables may be
augmented at any later time by additional Phase 1 operations.
Whenever a Phase I operation occurs, the tables which are generated as output are called New MIDAS Tables. Whenever the MIDAS Translator is run for strictly translation purposes or for augmenting existing MIDAS tables, an existing set of tables must be provided as input; these existing tables are called Old MIDAS Tables. Thus, every MIDAS Translator run except the initial run which builds the first set of tables for a particular project and/or target computer will require a set of Old MIDAS Tables.

The old and new tables are input and output as binary files. The file names are dependent upon the computer system on which MIDAS is running. In addition, there are three other files: card image input, printed output, and translated output.

The card image input file contains two optional blocks of information, of which at least one of the blocks must be present. The first block is the data structure definition using the language described in Section C. The other block contains the programs to be translated, written in MIDAS language. Each program to be translated must terminate with a normal FORTRAN END card, and as many programs as desired may sequentially appear in the input. If both data structure definitions and source programs appear, the data structure definitions must come first.

Listed output from the MIDAS Translator includes all input information and diagnostics in the event of error. If a source program contains a MIDAS detectable error, an error statement will be inserted into the translated output for that program which will produce a FORTRAN error when the program is compiled. Subsequent source programs will be translated correctly.

The translated output file contains card images of the translated source program in the FORTRAN dialect appropriate to the target computing system.

2. CDC Operating Details

The following file names are used by MIDAS:

2.275
INPUT    Input card images
OUTPUT   Listed output
OLDMT    Old MIDAS Tables
NEWTMT   New MIDAS Tables
FTNOUT   Translated source programs

The program operates in a field length of 55008 words. MIDAS can be called by the single control card:

MIDAS.

If the input card images are on a file other than INPUT, say, file SOURCE, then the control card should read:

MIDAS,SOURCE.

The MIDAS Translator on CDC systems provides complete translation capability to CDC, IBM, UNIVAC, and MULTICS target systems.

3. MULTICS Operating Details

The following file names are used by MIDAS:

FILE04  Input card images
FILE06  Listed output
FILE07  Translated source programs
FILE08  Old MIDAS Tables
FILE09  New MIDAS Tables

The MIDAS Translator on MULTICS provides complete translation capability to IBM, UNIVAC, and MULTICS target systems. Data structure definitions involving bit packed words are not permitted for translation to CDC target systems.
4. **IBM Operating Details**
The IBM version of MIDAS is under development at the present time.

5. **UNIVAC Operating Details**
The UNIVAC version of MIDAS is under development at the present time.
TO: MIDAS Users
FROM: A. F. Malmberg, Chief Scientist (5200)
SUBJECT: Multiple Definition Capability in MIDAS

MIDAS has been extended to permit the user to make multiple definitions of data structures and data structure members. These definitions may be made at the data structure, word, or byte level. This feature is supported by UPDATE level AM78131A and above of MIDAS.

The + symbol is used to separate the multiple definitions at a particular level. For example, two definitions of the data structure $S$ can be made as follows:

$$S(AAA) = (J, K) + (L, M, N)$$

Note that each definition of the data structure may have a different number of words.

Multiple definitions at the word level can be accomplished as shown in this example:

$$S(AAA) = (J + P, K + B + C)$$

Multiple definitions at the byte level may also be made:

$$S(AAA) = (J, (D(3) + E, F(10), G(30) + H + W))$$

When making multiple definitions for a particular byte, the first definition listed must specify the byte size in bits; subsequent definitions of that byte must not specify the byte size.

Each data structure definition statement must contain unique names. Thus, the following multiple definition is not allowed:

$$S(AAA) = (J, K) + (L, J)$$
Other than common byte size for a particular byte, there are no restrictions imposed on the nature of the members in multiple definition. Thus, each member may be independently assigned pointer attributes and alternate FORTRAN array names. For example:

\[ S(AAA) = (J) + (K, (F=*B(3)+C, D(5)) + M=*S(R)) \]

Data structure definition statements may be continued from one card image to the next if the last non-blank character on the card to be continued is either \( , \) or \( . \)
APPENDIX C

THE PROGRAM DESIGN LANGUAGE

A. INTRODUCTION

The algorithms and program segments for MADEM have been developed using a program design language (PDL). The use of PDL permits design to be expressed without the necessity of using an implementation in a standard programming language to specify the design detail.

Traditionally, narrative descriptions, decision tables, and flowcharts have been used to describe the design of a software system. These techniques are now being challenged by program design languages such as that used for the MADEM design. These PDLs provide: (1) a vehicle to translate functional modeling concepts into program design; (2) a replacement for design logic flowcharts; and (3) a means for communication between technical and nontechnical personnel, designers and developers.

PDL also has the advantage of having a closer relationship to programming languages than traditional methods of expressing design, thereby permitting a more direct mapping of design specification into code.

PDL is English-like in its means of expression and follows certain semantic and syntactic conventions. The concepts of structured programming are applied in the form of basic control structures for logic flow and indentation. Top down programming is implemented by specifying in PDL the top level portion of the program and evolving the PDL into succeeding levels of detail. There is considerable latitude in the selection of predicate and function descriptions which may be in English, in a computer language, or some combination of both.

The PDL is used for both the actual design framework of the program as well as for algorithms appearing only in hard copy form. The use of PDL eliminates the need for all flow charts and provides a self-documenting capability for the program itself. Realization of an implementation consists
simply of adding the necessary coding to the logical design statements. Thus, the design language and the implementation language coexist in the final source code.

B. CONTROL STRUCTURES

Four types of control structures are used: sequential composition, DO WHILE, IF-THEN-ELSE, and CASE. By suitable combination of these four types, programs and algorithms of any complexity may be expressed. These basic structures are combined with an indentation notation which is used to delineate the bounds of each structure in the program or algorithm.

The keywords, DO WHILE, IF-THEN-ELSE, CASE, etc., are written as structured comments. This permits the predicates for DO WHILE, IF-THEN, and CASE to be written in more natural terms since they do not have to be intelligible to the compiler. In implementation they are followed immediately by the associated programming language code statements. Thus, the programmer has the ability to write programs in true structured style using language elements of his own choosing, along with the implementation of the structure using a standard programming language. In this way, the coding and design documentation are carried along in one-to-one correspondence.

In the structured comment format, each comment is begun with an asterisk to visually delineate comment lines from coding lines. A comment may be continued on the next line. However, the continuation line does not have the asterisk header and starts one column farther to the right than the initial line of comment. Thus, if, for a given indentation level, the comment asterisk appears in column n, the first character of the comment continuation will start in column n+1. All implementation statements also start in column n+1; if an implementation statement is continued, all such continuation lines start at column n+2. Thus, the structured skeleton of the program design in PDL is easily followed by the identifying asterisks which introduce the structured comments. Each subsequent indentation level is started three columns farther to the right from the previous level to indicate nested structures.
Sequential compositions are shown diagrammatically in Figure C-1 and are written on the same indentation level, as in this example:

*INITIALIZE CONTROL VARIABLES
*READ INPUT DATA
*BUILD DATA STRUCTURE
*TRAVERSE DATA STRUCTURE, GENERATING OUTPUT

The IF-THEN-ELSE structure shown in Figure C-2 is illustrated by:

*IF(predicate)THEN
  *BLOCK 1
*ELSE
  *BLOCK 2
*BLOCK 3

If the predicate is satisfied, then BLOCK 1 is executed. BLOCK 1 is automatically terminated by the appearance of ELSE on the same level as the IF-THEN. Thus, following completion of BLOCK 1, control is passed to BLOCK 3 (which is actually a sequential block following the IF-THEN-ELSE block, appearing here only for purposes of illustration). If the predicate is not satisfied, then BLOCK 2 is executed, followed by BLOCK 3.

An alternate form of the IF-THEN-ELSE structure is shown in Figure C-3 and occurs when there is no ELSE part:

*IF(predicate)THEN
  *BLOCK 1
*BLOCK 3

If the predicate is satisfied, then BLOCK 1 is executed, followed by BLOCK 3. If the predicate is not satisfied, BLOCK 1 is not executed, and control passes immediately to the execution of BLOCK 3.
Figure C-1. Sequential Composition

Figure C-2. If-Then-Else Structure
IF(P) THEN
  BLOCK 1
  ...
  ...

Figure C-3. If-Then Structure

DO WHILE(P)
  BLOCK 1
  ...
  ...

Figure C-4. Do While Structure
The DO WHILE structure is shown in Figure C-4 and has the form:

```
*DO WHILE(predicate)
   *BLOCK 1
*ENDDO
*BLOCK 2
```

BLOCK 1 is executed until the predicate becomes FALSE. The use of the ENDDO statement is optional.

A variant on the DO WHILE is the DO, sometimes a more natural means of expressing the conditions for executing the subordinate block. Its action is identical to the DO WHILE:

```
*DO(predicate)
   *BLOCK 1
*BLOCK 2
```

Another type of control structure is the CASE structure shown in Figure C-5. This is used to select one of several possible blocks, depending upon the value of an expression. Its structure can be seen by the following example:

```
*CASE(expression)
   *Expression value = 1
      *BLOCK 1
   *Expression value = 2
      *BLOCK 2
   ...
   ...
   *Expression value = n
      *BLOCK n
*BLOCK A
```
Figure C-5. Case Structure

CASE(E)
  E = 1
  BLOCK 1
  E = 2
  BLOCK 2
  ...
  E = n
  BLOCK N
  ...
  ...

E

1

2

N
The expression is evaluated and control is passed to a particular block, depending upon the expression value. Following completion of the specified block, control is passed out of the CASE structure to the next sequential block, BLOCK A.

C. **SEGMENTATION**

Programs are subdivided into segments, both for purposes of multiple references and to aid intellectual manageability. Each segment is labeled with a name to identify it. The name may be any English phrase of one or more words which serve to identify the segment, usually chosen to describe its function. The segment identification occurs as the first structure comment, using the keyword SEGMENT for identification:

*SEGMENT(GENERATE OUTPUT DISPLAY)*

The segment identification may then be followed with arbitrary comments which provide further information on the purpose of the segment and its data and control interface, as needed. These explanatory comments do not use the asterisk header so that there will be no accidental confusion with the actual control structures which follow.

Segments may be referenced by using the keyword INCLUDE as in this example:

*INCLUDE(GENERATE OUTPUT DISPLAY)*

The close of a segment is always indicated by the statement:

END

D. **PROGRAM DESIGN AND IMPLEMENTATION**

The program design process involves a complete logical specification of the program in PDL. All logical steps are expressed in the language
so that another person can fully comprehend the logical process being specified.

Combining these ideas permits the writing of algorithms and programs of any complexity. As an example, consider the following segment which specifies an algorithm for traversing an n-ary tree, where each node of the tree is represented by a linked list of sibling cells, each of which has a pointer to a descendant node. Thus, each sibling cell has two pointers, one for the sibling cell and one for the descendant node.

*SEGMENT(TRAVERSE N-ARY TREE)
*ENTER ROOT NODE
*TRAVERSE=TRUE
*BACKUP=FALSE
*DO WHILE(TRAVERSE)
  *IF(NODE CELL HAS DESCENDANT AND BACKUP=FALSE)THEN
    *DESCEND TO DESCENDANT CELL
  *ELSE
    *BACKUP TO PARENT CELL
  *IF(ROOT NODE)THEN
    *TRAVERSE=FALSE
    *BACKUP=TRUE
*END

When the PDL structure is implemented, the programming language statements are inserted immediately following the corresponding PDL statement. In many cases, the implementation of a simple PDL structure will contain additional microstructure. Such microstructure should also follow the same rules for indentation of its logical elements.

As an example of the implementation of PDL into FORTRAN, consider the implementation of the above example for traversal of n-ary trees. The tree cells are stored in a FORTRAN array IQ with a pointer LTREE for the root cell of the tree. Each cell contains two pointers, with the first
cell word containing the sibling pointer and the second cell word containing
the descendant pointer. A pointer value of zero indicates a null pointer
which terminates the pointer chain. The implemented segment can thus be
written as:

```
C *SEGMENT(TRAVERSE N-ARY TREE)
SUBROUTINE TRAVERSE(LTREE)
C A PUSHDOWN STACK ISTACK WITH POINTER J IS USED TO BACKTRACK
C TOWARDS ROOT NODE OF TREE
COMMON/TREE/IQ(1000)
DIMENSION ISTACK(100)
C *ENTER ROOT NODE
L=LTREE
J=1
C *TRAVERSE=TRUE
ITRAV=1
C *BACKUP=FALSE
IBACK=0
C *DO WHILE(TRAVERSE)
1000 IF(ITRAV.EQ.0) GO TO 2000
C IF(NODE CELL HAS DESCENDANT AND BACKUP=FALSE)THEN
   IF(IQ(L+I).EQ.0.OR.IBACK.NE.0) GO TO 1100
   ISTACK(J)=L
   J=J+1
   L=IQ(L+I)
   GO TO 1500
C ELSE
C IF(NODE CELL HAS SIBLING Cell)THEN
1100 IF(IQ(L).EQ.0) GO TO 1200
   L=IQ(L)
   C *BACKUP=FALSE
   IBACK=0
   GO TO 1500
C ELSE
C *BACKUP TO PARENT CELL
1200 J=J-1
   L=ISTACK(J)
C IF(ROOT NODE)THEN
1300 IF(L.NE.LTREE) GO TO 1300
C *TRAVERSE=FALSE
   ITRAV=0
C *BACKUP=TRUE
   IBACK=1
1500 GO TO 1000
2000 RETURN
END
```
APPENDIX D

MIDAS TABLES

/MIDAS CDC
/DEFINE
ABINFO(ISPACE) = (**ABINFO(30), NRACTYP(30)),
(NOACOH(30), PTRACDB==ACDB(30)),
NORMGR,
NOREARMO,
NOREFUEL,
NOAIRING,
NOUSE)
ABQUEDS(ISPACE) = (**ABQUEDS(30), CLASS(30)),
VALUE1(ISPACE),
VALUE2(ISPACE),
VALUE3(ISPACE)
ABSTATUS(ISPACE) = (**PACTAB==ABINFO(30), NOACTAB(30)),
(PTR2QUEUES==QUEUES(30), NOACONAD(30)),
ABDAMAGE(ISPACE)
ABVCR(ISPACE) = (**PACTAB==ABVCR(30), NRACTYPE(30)),
(PACTAB==ACTAB(30), NOACTAB(30))
ACDB(ISPACE) = (**ACDB(30), NRACTYPE(30)),
MAXSPEED(ISPACE),
CRUISESPEED(ISPACE),
MAXALTITUDE(ISPACE),
MINALITITUDE(ISPACE),
MAXCLIMBDEVS(ISPACE),
FUELCONSUME(ISPACE),
ACORANGE(ISPACE),
RADARCS(ISPACE),
ATTACKRADuish(ISPACE),
MAXFUEL(ISPACE)
ACDBUF(ISPACE) = (**PTRACO==ACODEVICE(30), NUMDEV(30))
ACODEVICE(ISPACE) = (**PNEXT==ACODEVICE(30), TYPE(30)),
(WORKING(30), PACODE==ACDB(30)),
(JAN(30), LEVEL(30))
ACRFTLIST(ISPACE) = (**PNEXT==ACRFTLIST(30), ACRFTID(30)),
NUMACRFT,
FORCTYPE
ACRFTONAB(ISPACE) = (**PNEXT==ACRFTONAB(30), ABD(30)),
(ACRFTLIST(30), NUMEL(30))
ACTAB(ISPACE) = (**PNEXT==ACTAB(30), NRACTYPE(30)),
(NOACOFH(30), NOAONDB(30))
ADCCLINK(ISPACE) = (**PNEXT==ADCLINK(30), ID(30))
ADCSITED(ISPACE) = (**PNEXT==ADCSITED(30), ADTYPE(30)),
(MODVAL1(30)=INSTANT, MAXINDEX(30)),
(MAXCLEDIDEST(30), MINTIMEINDEX(30)),
OSTTIME(ISPACE),
LASTCHANCE(ISPACE), LOWINDEX(ISPACE),
ENGAGGLISH(binary),
MODVAL2(ISPACE),
MODVAL3(ISPACE),
(COONONE(30), ONE(30)),
(COONONEFEW(30), FEW(30)),

291
ALLOCATE (1SPACE) = (FLU=ALLOCATE (30), PDOWN=ALLOCATE (30)),
{AMTYPE (30), PRI=FIRSTUNIT (30))}
ANYDIL (1SPACE) = (FLU=ANYDIL (30), PDOWN=ANYDIL (30)),
{FFER=FIRSTLIST (20), PDID=DISD (20)),
{LOC (30), PRIORITY (20),
PRIUP=ANYDIL (30), PRID=ANYDIL (30),
DCOV (30), SHORT (30),
PEDELAY=FADE (20), PENDEV=EVENT (20),
{PAL (20), NULL (20)]]}
ADD (1SPACE) = (NEXT=ADD (20), NAGTYPE (20)),
{RANGE (SPACE),
YOUSEL, NOUSE2}
ARCTSAW (1SPACE) = (FNEXT=ARCTSAW (30), FS=SB (30)),
{ADDRESS=HEX (30), TYPE (30),
DAMAGE (SPACE)}
ARCTSTATUS (1SPACE) = (PFTL=FFLTDS (30), FPMUNIT=MPH (30),
{PSTRTHX=HEX (30), PENDH=HEX (30),
PNXTHX=HEX (30), PAIRBASE=SB (30),
NULL1 (20), PAIRGOT=SB (30),
{PPNOTG=SBD (30), NUMMAIRCRAFT (20),
FLITELICO (4), INTERCEPTSTATUS (4), ALTUDECHNG (4),
PROFILEX (4), LANDING (4), ORBITSTAT (4), AIRCOMERAT (4),
NULL2 (4), JMSTAT (4), DUMMY (24))}
{FUEL, ALTITUDE (SPACE),
SPEED (SPACE),
DIRECTION (SPACE)}
ATTACKSLCA (1SPACE) = (FNEXT=ATTACKSLCA (30), FNXTSAGL=STOEL (30),
{FPMARK=FFMARKSLCA (30), NPOMARK (30))
{ISECTOR (30), FNXTCSAG=CORRIDORSAG (30))}
SLO-HEID (1SPACE) = (ISTATUS (24), TMSPEC (17), ALLOCATOR (10), TYPE (20),
BODDIL (1SPACE) = (FLU=BOCDIL (30), PDOWN=BOCDIL (30),
{FFER=FIRSTLIST (20), PDID=DIS (20),
LOC (30), PRIORITY (20),
PRIUP=BOCDIL (30), PRID=BOCDIL (30),
DCOV (30), SHORT (30),
PEDELAY=FADE (20), PENDEV=EVEN (20),
{PAL=PAL (30), PUDS=SUBLIST (20)]]}
BOCDILNOCHANGE (1SPACE) = (NULL1,
NULL2, NULL3)
DOCSSTAT(1 SPACE) = ((AUTO(30), READY(30)),
  (PSD10 = EVENT(30), PSEDELY = EVENT(30)),
  (PADIL = BOCIDL(30), NADIL(30)),
  (PHFO0 = BOCIDL(30), PTOO0 = BOCIDL(30)),
  (FNDIL = FERLIST(30), FNNDIL = FERLIST(30)),
  (HPRIOR = BOCIDL(30), TRPRIOR = BOCIDL(30)),
  (PDIL = BOCIDL(30), PDILA = BAC(30)),
  [VION(30), PTGTS(30), PDIAS(30)])

PCTGTLOSTM50(1 SPACE) = NULL1,
  NULL2,
  NULL3.

DOBSSTAT(1 SPACE) = ((AUTO(20), READY(20)),
  (PSD10 = EVENT(30), PSEDELY = EVENT(30)),
  (PADIL = AMYDIL(30), NADIL(30)),
  (PHFO0 = AMYDIL(30), PTOO0 = AMYDIL(30)),
  (FNDIL = AMYDIL(30), FNNDIL = AMYDIL(30)),
  (HPRIOR = AMYDIL(30), TRPRIOR = AMYDIL(30)),
  (ODIL = AMYDIL(30), CAPACITY(30)),
  (PDA0 = DATE(30), FTA0 = DATE(30))

STRYD1(1 SPACE) = ((PUP = STRYD1(30), PDWAW = STRYD1(30)),
  (PFCM = FERLIST(30), PDI = BAC(20)),
  (LOC1(30), PRIORITY(30)),
  (PR = STRYD1(30), PRIN = STRYD1(30)),
  (DOD1(30), SHORT(30)),
  (FSDELY = DATE(30), FENDELY = EVENT(30)),
  (PFA0 = ALLOCATE(30), NKE(30)),
  (CEASE(30) = DISASSN = DID, ASNFRIOR(30)),
  (WATTITRACK(30), WAIETFIRE(30)),
  START( SPACE ),
  END( SPACE )

STYFIREM50(1 SPACE) = (NULL1,
  NULL2,
  NULL3.

STYSTAT(1 SPACE) = ((AUTO(30), NUM3(30)),
  (PSD10 = EVENT(30), PSEDELY = EVENT(30)),
  (PADIL = STRYD1(30), NADIL(30)),
  (PHFO0 = STRYD1(30), PTOO0 = STRYD1(30)),
  (FNDIL = FERLIST(30), FNNDIL = FERLIST(30)),
  (HPRIOR = STRYD1(30), TRPRIOR = STRYD1(30)),
  (TRACEN = STRYD1(30), IDE(30)),
  (PDA0 = DATE(30), FTA0 = DATE(30)),
  (PFTL(30), A1(30) = CAMMO),
  (RESUPPLY(30), A2(30) = LAMMO + TSTAMMO),
  (NUNO(30), A3(30) = N2AMMO),
  (AMT0(30), NUME(30))

BYCEASEM50(1 SPACE) = (NULL1,
  NULL2,
  NULL3.

BYEND(1 SPACE) = (PTGTS(30), NULL4(30))

BYNOFROOMM50(1 SPACE) = (NULL1,
  NULL2,
(PTOTS(30)+PREADS+PROVERB+POSTSE+FINTSB),
(TGTYPE(30)+DEADTYPE+ADDRESS+NULL+NUMACFT)
CRC3EESBLUE(I$PACE)=((PNEXT=CRC3EESBLUE(30), ID(30)),
(PSB=SB(30), ADDRES$=HEX(30)))
CRC3EERED(I$PACE)=((PNEXT=CRC3EERED(30), ID(30)),
(PSB=SB(30), ADDRESS=HEX(30)),
(RPT(30), HUNTER(30)),
(DIRECTION(SPACE),
(PNXTSER=SEER(30), NUMSEE(30)))
CRC3EES(I$PACE)=((REDSEE=CRC3EEREDE(30), NUMRED(30)),
(BLUESEE=CRC3EESBLUE(30), NUMBLUE(30)))
CRC3SUBORD(I$PACE)=((PNEXT=CRC3SUBORD(30), ID(30)),
(PSB=SB(30), ADDRESS=HEX(30)),
(ASSORDMFLU(30), BNA$BONT(30))
ACRFLASZONE+
NTRCPTRASNM=NUMACONAB)
DACE(I$PACE)=((PPREV=DACE(30), PNEXT=DACE(30)),
(LACT(30), PTR(30)),
(ARG1(30), PND=DACE(30)),
(ARG2(30), ARG3(30)))
DATBLOK(I$PACE)=((PNEXT=DATBLOK(30), CLASS(30)),
(PADSITE=ADSITE(30)+
ENClOAtM=ADVART(30)+
PACONAB=ACRFLASZONE+
CLASS, NUMLOC=(30)))
DATBLOK(I$PACE)=((PDATBLOK=DATBLOK(30), NUMBLOK(30)))
DBCLASSBLOK(I$PACE)=((PNEXT=DBCLASSBLOK(30), CLASS(30)),
(PTR(30), NUMBER(30)))
DEATHBLOK(I$PACE)=((PSB=SB(30), PRD=HEX(30)),
(UNITTYPE(30), SIDE(30)))
DIB(I$PACE) =(TIME(SPACE),
(SIDE(30), NUMAC(30)),
(LOST(30), POSITION(30)),
(UNITTYPE(30), SIDE(30)))
DUMMYBLOK(I$PACE)=((PNEXT(30), NEWKEY(30)))
ENDDAYSIMSG(I$PACE)=(NULL1,
NULL2,
NULL3,
(PTOTS=SB(30), COVER(30)),
STARTOP(SPACE),
ENDOP(SPACE),
(PTDS=DB(30), PRIORITY(30)))
ENGRSCUT(I$PACE)=((FKILL(30), PFU(30)))
EVENT(I$PACE)=((NEPEN=SB(30), INC(30)),
(PTUP=EVENT(30), PTRADDW=EVENT(30)),
(MSG(30), LASSEN=SB(30)),
...)
(AMOUNT(20), PREDICT-PAYLOAD(20))
LOSTCOVER(I SPACE) = (P R E V = P R E V ⇀ L O S T C O V E R (20), PDIL = BOCDIL(20)),
(NULL(30), PRIORITY(20))
LOSTSIGHTLOCK(I SPACE) = (NULL(30), NOSEER(30)),
(null(30), ADDRESS(30)))
MESSAGE(I SPACE) = ((PTR(20), FREQ(30)),
(VALUE1(30), VALUE2(30)) = (VALUES(30)),
(TYPE(30), LENGTH(30))))
MISSILEFIRMING(I SPACE) = ((AMMOTYPE(30), FU(30)),
TIMEINTERCEPT(I SPACE))
MUN(I SPACE) = ((PAGE=LOAD(30), NUMAR(30)),
(PAAL=LOAD(30), NUMAA(30))
NOAVAILLOCK(I SPACE) = (P R E V = NOAVAILLOCK,
P DAMAGE(I SPACE),
PTSTDELOCK = STDELOCK)
ORDERS(I SPACE) = ((PTRFORMS=FORMATION(20), PTRACT=COMMAND(30))
PAL(I SPACE) = ((FUPF=PAL(10), PDOWN=PAL(20)),
(PFPS=PAL(30), PDOWN=PAL(30)),
START(I SPACE),
END(I SPACE),
(COVER(20), PDIL=BOCDIL(30)),
(PREV=EVENT(30), PSUB=ISUBLIST(30))
PATENDAGE(I SPACE) = ((FUPF=PATENDAGE(30), PDOWN=PATENDAGE(30)),
(PDIL=ETRYDIL(30), STAGE(30)),
(CEASE(20), FENCE==EVENT(30))
PAYBUF(I SPACE) = ((PEXT=PAYBUF(30), NRPOCLS(30)),
(PAYLDDLE=PAYLDDLE(30), NUMBLOC(30))
PAYLDDELOCK(I SPACE) = ((PEXT=PAYLDDLE(30), TYPEINDEX(30))
PAYLOAD(I SPACE) = ((PNTAX=PAYLOAD(30), NRPOCLS(30)),
(MAXAMT(30), MINAMT(30))
(MAIFPERANSE(30), PAYLDDLE=PAYLDDLE(30))
PERLIST(I SPACE) = ((PNTAX=PERLIST(30), PDOWN=PERLIST(30)),
(PBP=30), PDIL=ANVIL(30)),
(SEEN(20), PSS=ISOURCE(30)),
(PUCNCH=PERLIST(30), PUNCHN=PERLIST(30)),
TIME(I SPACE))
PLAYERUFFER(I SPACE) = ((PTRPL=PLAYERUFFER(30), VARWORD(30)))
PLUFFER(I SPACE) = ((PTRPL=PLUFFER(30), VARWORD(30)))
PLYLIST(I SPACE) = ((LILW(30), [RUN=PLUFFER(30)])
POSSCOVER(I SPACE) = ((PNTAX=POSSCOVER(30), PDIL=20CDIL(30)
PSTAT=ISUBLIST),
(PFAL=PAL(20), PRIORITY(20))
PROFILESER(I SPACE) = ((FNXPRED=PROFILESER(20), NRSPRED(20))
ALTGEN(20),
ALTST(20),
ALTDAIG(20))
PUUFFER(I SPACE) = ((PSTART=LINK(30), NUMLINK(30))
QUESTAT(I SPACE) = (REALNUMBER(I SPACE))
QUES(20) = ((.NULL=QUES(20), QUENUM(30)),
(PTR(30), NUMBER(30)),
(PODE=ADDQS(20), POQUESTAT(30)))
RAIDLOCK(I SPACE) = ((PNTAX=RAIDLOCK(30), NRRAID(30)),
(PTRRAVE=RAVELOCK(30), NOAVES(30)),
(PTRRCORD=SSRCORD(30), SSORDER(30))
REAYQUE(I SPACE) = ((PNTAX=REAYQUE(30), NAAYTYP(30))
...
RELLIST(ISPACE) = ((PNEXT = RELLIST(ISPACE), PPU = FIREUNIT(ISPACE)))
SATELLISTIC(ISPACE) = (NULLI, NULLI)
SB(I) = (PNEXT = SB(I), TGTIL(I) = NCOV),
STARTOP(ISPACE),
ENDOPP(ISPACE),
(PSD = DIR(I), TGTFRCLR(I)),
(EMPTY(I), DCON(I))
SUB(I) = (ADDRESS = HEX(I), PC = -Z(I))
(PSDB = Dir(I), PFEL = EVENT(I)) * MULTIDAMAGE,
(PACS = ACOBUF(I), ID(I))
(DATABASE(I))
ADJFORMTPH, PAGSTATUS = ASTATUS(I)
PARCFSTAT = AROCFSTATUS,
FDCQSTAT = AOCQSTAT,
PDTRYSTAT = ARTRYSTAT,
REDARCFUPCINT,
STATUS)
SUB(I) = ((PSDB = EB(I), PSEEDUF = SSEEUF(I), PSEE = P3SEE(E)),
(SUBORDINATE = SUB(I), CRD = ORDER(S(I)))
(NAI = RAID(I)))
SEEDUF(ISPACE) = ((PTRSEE = AROFTSAW(I), NUMT(I))
SEER(I) = ((PNEXT = SEER(I), PSEEREB(I)),
SRC = (PSD = S(E(I), PNEXT = SRC(E(I)))
STDBLOK(ISPACE) = ((PNEXT = STDBLOK(I), PTGTSD = SB(I)),
PTGTLTR = TGTPTREE,
DAMPER(ISPACE),
PABRPER = HEX)
SUB(I) = ((PSDB = SUBTYPE(I), NUMBER(I)))
SUBLIST(I) = ((PSDB = SB(I), PNEX = SUBLIST(I)),
(AUTO(I) = DEATHMARK, NUMF(I))
(RAMMD(I), LOAD(I))
(PCRED = DAME(I), NOUSE(I))
(PPAL = PAL(I), ADDRESS = HEX(I))
SUBTYPE(I) = ((PNEXT = SUBTYPE(I), TYPE(I))
(PTRSUB = ORGORD(I), NUMBER(I))
TARGETLOK(ISPACE) = ((PNEXT = TARGETBLOK(I), NRTGTYP(I)),
(PTRFORM = FORMATIONEBL(I), NOFORM(I)),
(PTOTAT = ATTACKBLOK(I), NOTOTAT(I)),
(MAXCALG(I), NOCALC(I)),
(MAXRHEX(I), MINRHEX(I))
TARGETLLOK(ISPACE) = ((PNEXT = C(1), VARPORD(I)))
TARGETTLOK(ISPACE) = ((PNEXT = TTDBLOK(I), NOTYTPL(I)))
TGTPTREE(ISPACE) = (DAMAGE(ISPACE),
(INVALID(I), PSTDBLOK = STDLOK(1), DIST(4)),
PLEFT = TGTPTREE(17), PRITE = TGTPTREE(17))
TRACKINGBLK(ISPACE) = ((NULL(I), PSEERSB(I)),
(PHOPVPSD(I), ADDRESS(I))
TTDBLOK(ISPACE) = ((PNEXT = TTDBLOK(I), NRTGTYP(I))
(PGTPPL = STCBLOK(I), NOTGTPPL(I))
UPDAT(ISPACE) = ((MISSELS(I), AMMOTYPE(I), ENUNITYTYPE(I)),
(PGTSBD = SB(I), PP(I))
WAVELOCK(ISPACE) = ((PNEXT = WAVELOCK(I)), NAWAVE(I))

298
+ PTRSTAT, PTRSEE, PTRSUB, PTRORD, MYTYPE, MYSIDE
*COMMONINTMSG*
  COMMON/INTMSG/MESSAGE, PGTRED, ITYPE, PINTLIST, PINTID, PTRDL:
*COMMONIODEV*
  COMMON/IODEV/IN, NOIL, ND, IOUT, IFETCH(2), TFETCH, IFETCH, NOILS
*COMMONLUP*
  COMMON/LUP/TP, ITYPE, PINTLIST, TFETCH, ITYPE
*COMMONMAX*
  COMMON/MAX/MAX(20, IFLAG)
*COMMONPERGAME*
  COMMON/PERGAME/XSEER, YSEER, XSEEN, YSEEN, RANGE, B机器, MAPO
*COMMONPAT*
  COMMON/PAT/SPAT(20), ICOUNT, ICOUNT(20), ICTREL(20), ICTGIM(20)
*COMMONSTATBCS*
  COMMON/STAT/STATCD/PFLINTYP(1), PTRMUN, PTRSTAT, PTRSTRT, PTRTEND, PTRNXT, PTRAB,
  + NUMTGT, PGTGSB, PGTGTO, NUMACT, NACTNG, NACTNCRF,
  + NXTSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NUMSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
  + NORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA, IORDSTA,
*NOSEG*

/END

NOSEG = 320

."
APPENDIX E
THE DYNAMIC EVENT SCHEDULING ALGORITHM

An event stepped simulation, such as TAC REPELLER, is controlled by one or more time sorted lists of event notices, where each event notice represents the occurrence of some event. At the start of the simulation, all the notices in the event lists correspond to exogenous events, i.e., events which are externally generated and act as a driving function for the simulation. As the simulation progresses, time is incremented as each event notice in the event lists becomes current. An event notice initiates a computational process which may generate additional event notices. In this way the simulation continues. The simulation stops whenever the event lists are exhausted or some other specified termination condition occurs.

The passage of time between events is generally very irregular. In fact, consecutive events may occur at the same instant of time, or there may be a very large time interval between them. Event stepped simulations are computationally efficient because of the ability of the event scheduling mechanism to initiate computation only at those times at which something is going on in the simulation.

The fundamental problem in an event stepped simulation is to devise an efficient way of locating the next event notice in time sequence. This can be done by keeping an event list in time sorted order so that the next event notice is always on the top of the list.

The algorithm for maintaining an event list must be chosen so that it is computationally efficient to 1) remove the event notice with the earliest time from the event list and 2) insert a new event notice into the list.

In TAC REPELLER the event notices for dynamically scheduled events are maintained in a quasi-sorted order using a bifurcated arborescence called a leftist tree (so called because it leans to the left, i.e., there are predominantly more links to the left than to the right). The algorithm for manipulating such a tree is such that the top cell of the tree (the root) is always guaranteed to be the cell with the earliest scheduled event time,
even though all other cells are only in quasi-sorted order. Despite some algorithmic complexity, the leftist tree software is very efficient.

Each event notice cell contains the scheduled event time, a path distance value, a pointer to a left subtree, and a pointer to a right subtree. The space requirements are $O(n)$, where $n$ is the number of event notices in the event list.

Since the scheduled event time is the quantity upon which sorting is based, it will be referred to as the KEY in the following discussion. The path distance $D$ is the minimum path length from the node to a leaf of the tree; because of the way in which the tree is constructed, this minimum length path will always be a rightmost path. The pointer to the left subtree will be denoted by $LP$, and the pointer to the right subtree will be denoted by $RP$. The pointer to a leaf in the tree will be assigned a value of $0$. The leaf does not actually contain any explicit information and therefore leaves are not actually represented in memory. We may refer to the KEY and $D$ quantities for the cell comprising the roots of the left and right subtrees by $KEY(LP)$, $D(LP)$, $KEY(RP)$, and $D(RP)$. We adopt the convention that $KEY(0) = \infty$, and $D(0) = 0$.

The leftist tree may be defined by listing the properties of the KEY and $D$ fields for each cell $P$:

1. $KEY(P) \leq KEY(LP(P))$
2. $KEY(P) \leq KEY(RP(P))$
3. $D(P) = D(RP(P)) + 1$
4. $D(LP(P)) \geq D(RP(P))$

These properties ensure that any path from the root to a leaf traverses the event notices in ascending time order. Thus, the root always contains the next scheduled event notice.

Removal of the root of the leftist tree (i.e., removal of the next scheduled event notice) requires a constant time. However, it must be followed by a merging of the two subtrees below the root before any other operations are executed on the tree. Merging of the subtrees (both of which are themselves leftist trees) is the most expensive operation. In the worst
case this requires \( m \) merge steps followed by \( m \) interchange steps, where

\[
m = \text{least integer not less than } \log^2(n)
\]

\( n \) = number of nodes (event notices) in the tree.

The subtree merging is not required if the right subtree is vacuous, and is trivial if the left subtree is vacuous. Thus, merging of leftist trees requires time of \( O(\log n) \) in the worst case. Insertion of a new cell into the tree also requires \( O(\log n) \) in the worst case. In the best case, merging and insertion require a constant time. Furthermore, the software for merging and insertion is identical so that only a single routine is required for all tree manipulations. The overhead for small \( n \) is reasonable and the method is very efficient for large \( n \).

The algorithm for merging two leftist subtrees \( P \) and \( Q \) utilizes a stack for saving the nodes which are visited during the tree traversal.

The algorithm is:

1. IF (KEY(SUBTREE \( P \)) IS GREATER THAN KEY(SUBTREE \( Q \))) THEN
2. *INTERCHANGE SUBTREES \( P \) AND \( Q \)
3. *IF (SUBTREE \( P \) IS VACUOUS) THEN
4. *SUBTREE \( P \) = SUBTREE \( Q \)
5. ELSE
6. *MERGE SUBTREE \( Q \) INTO SUBTREE \( P \)
7. *SUBTREE \( X \) = \( P \)
8. *DO WHILE(SUBTREE \( Q \) EXISTS)
9. *OKEY = KEY(Q)
10. *TRaverse SUBTREE \( X \) ALONG THE RIGHTMOST LINKS, COMPARING KEY AT EACH NODE OF \( X \) WITH OKEY, SO AS TO LOCATE THE INSERTION POINT FOR \( Q \) ACCORDING TO ASCENDING KEY VALUE. SAVE THE NODES OF \( X \) WHICH WERE TRAVERESED ON THE STACK.
11. *BREAK SUBTREE \( X \) AT THE INSERTION POINT, SAVING THE SUBTREE OF \( X \) BELOW THE INSERTION POINT AS SUBTREE \( T \).
12. *APPEND SUBTREE \( Q \) TO SUBTREE \( X \) AT THE INSERTION POINT, USING THE LEFT LINK IF IT IS NOT ALREADY IN USE: OTHERWISE USE THE RIGHT LINK.
13. *SUBTREE \( X \) = \( Q \)
14. *SUBTREE \( Q \) = \( T \)
15. *DO FOR ALL NODES SAVED ON STACK DURING TRAVERSE, STARTING AT LAST INSERTION POINT AND ENDING AT THE ROOT)
16. *IF(DISTANCE TO LEAF FROM LEFT SUBTREE SHORTER THAN FOR RIGHT SUBTREE) THEN
17. *INTERCHANGE SUBTREES
18. *CALCULATE DISTANCE FOR PARENT NODE OF SUBTREES
An example of event sorting using the leftist tree algorithm will now be given. We start with the leftist tree shown in Figure E-1(a). The event notices are numbered with the integers, where the integer value is the scheduled time for the event notice.

The next scheduled event 1 is removed. This splits the original tree into two subtrees as shown in Figure E-1(b). We must merge subtree 3 into subtree 2. This is done by traversing subtree 2 along the rightmost path, until an insertion point is found for subtree 3. This insertion point will be between nodes 2 and 5. A subtree with root 5 is detached and the subtree 3 is appended to subtree 2 at the insertion point. This produces the configuration shown in Figure E-1(c).

Now subtree 5 must be merged into subtree 2. Subtree 2 is traversed along the rightmost path until an insertion point for subtree 5 is found. This insertion point will be between nodes 3 and 8 in Figure E-1(c). A subtree with root node 8 is detached and subtree 5 is appended to subtree 2 at the insertion point. This produces the configuration shown in Figure E-1(d).

Next subtree 8 must be merged into subtree 2. Subtree 2 is traversed along the rightmost path until an insertion point for subtree 8 is found. This insertion point is after node 5 in Figure E-1(d). Since no right subtree exists below node 5 at this time, no subtree can be detached, and subtree 8 is simply appended as the right subtree for node 5. This produces the configuration shown in Figure E-1(e).

Subtree merging has now been completed. Now the nodes traversed during the merging process must be traversed in reverse order and subtree interchanges made as necessary to produce a leftist tree (note that the tree in Figure E-1(e) is a rightist tree). The subtree with root node 5 is already in leftist form so no action is required.

The subtree with root node 3 is not a leftist tree, therefore interchange the left and right subtrees below node 3. This produces the leftist subtree with root node 3 shown in Figure E-1(f).
Figure E-1. Example of Merging of Leftist Trees
MERGE 8 INTO 2.

INTERCHANGE 4 AND 5

INTERCHANGE 7 AND 3

ADD 1

Figure E-1. Example of Merging of Leftist Trees (Continued)
The subtree with root node 2 is not a leftist tree, therefore interchange the left and right subtrees below node 2. This produces the leftist subtree with root node 2 shown in Figure E-1(g). Since node 2 is the root of the entire tree, the entire tree is now in leftist form. This completes the merging process.

Now let us add node 1 back into the tree of Figure E-1(g). Since node 1 corresponds to an earlier time than any node of the leftist tree into which it is being merged, we see that node 1 becomes the root node of a new tree. This new leftist tree is shown in Figure E-1(h). Note that its form is radically different from the original tree in Figure E-1(a).

This example represents the worst case situation for merging two subtrees.
# APPENDIX F
## MADEM EVENT CODE DEFINITIONS

<table>
<thead>
<tr>
<th>EVENT CODE</th>
<th>MESSAGE CODE</th>
<th>SCHEDULING SUBROUTINE</th>
<th>EVENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>12691111</td>
<td>BADMOVE</td>
<td>Schedule CRC Assignment</td>
<td></td>
</tr>
<tr>
<td>12691111</td>
<td>CRCLOSS</td>
<td>Schedule Assignment</td>
<td></td>
</tr>
<tr>
<td>12691111</td>
<td>CRCKIL</td>
<td>CRC Assignment</td>
<td></td>
</tr>
<tr>
<td>12691111</td>
<td>INT2CRC</td>
<td>Schedule CRC Assignment</td>
<td></td>
</tr>
<tr>
<td>12691111</td>
<td>NEWMOVE</td>
<td>Schedule CRC Assignment</td>
<td></td>
</tr>
<tr>
<td>14651250</td>
<td>GNDLOOK</td>
<td>Target found, Schedule Attack</td>
<td></td>
</tr>
<tr>
<td>25120001</td>
<td>1312</td>
<td>INTASIN</td>
<td></td>
</tr>
<tr>
<td>25120001</td>
<td>1510</td>
<td>INTASIN</td>
<td></td>
</tr>
<tr>
<td>25120001</td>
<td>2731</td>
<td>BTNASIN</td>
<td></td>
</tr>
<tr>
<td>25340001</td>
<td>2751</td>
<td>AMMOCHK</td>
<td></td>
</tr>
<tr>
<td>25390001</td>
<td>1310</td>
<td>INTRFLY</td>
<td></td>
</tr>
<tr>
<td>25390001</td>
<td>1340</td>
<td>FUELCHK</td>
<td></td>
</tr>
<tr>
<td>25390001</td>
<td>1984</td>
<td>COMMAND</td>
<td></td>
</tr>
<tr>
<td>25390001</td>
<td>1984</td>
<td>INTRFLY</td>
<td></td>
</tr>
<tr>
<td>25620001</td>
<td>1330</td>
<td>INTFIND</td>
<td></td>
</tr>
<tr>
<td>25650001</td>
<td>1300</td>
<td>FLYSEE</td>
<td></td>
</tr>
<tr>
<td>2565001</td>
<td>1312</td>
<td>CRCTRACK</td>
<td></td>
</tr>
<tr>
<td>2565001</td>
<td>1320</td>
<td>AIRTHNK</td>
<td></td>
</tr>
<tr>
<td>2565001</td>
<td>1330</td>
<td>CRC2INT</td>
<td></td>
</tr>
<tr>
<td>25690001</td>
<td>2</td>
<td>BNCMDPR</td>
<td></td>
</tr>
<tr>
<td>25690001</td>
<td>2</td>
<td>BNNOTRD</td>
<td></td>
</tr>
</tbody>
</table>

311
<table>
<thead>
<tr>
<th>EVENT CODE</th>
<th>MESSAGE CODE</th>
<th>SCHEDULING SUBROUTINE</th>
<th>EVENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>25690001</td>
<td>3</td>
<td>BNPONBB</td>
<td>Schedule commo CRC, ready</td>
</tr>
<tr>
<td>25690001</td>
<td>1310</td>
<td>CRC2INT</td>
<td>Schedule interceptor available message to CRC</td>
</tr>
<tr>
<td>25690001</td>
<td>1320</td>
<td>AIRTHNK</td>
<td>Schedule msg to CRC of new target</td>
</tr>
<tr>
<td>25690001</td>
<td>1340</td>
<td>DOGTHNK</td>
<td>Message to CRC of unavailability</td>
</tr>
<tr>
<td>25690001</td>
<td>1500</td>
<td>BADMOVE</td>
<td>Schedule break off message to interceptor</td>
</tr>
<tr>
<td>25690001</td>
<td>1520</td>
<td>BADMOVE</td>
<td>Schedule msg to interceptor of new vector</td>
</tr>
<tr>
<td>25690001</td>
<td>1520</td>
<td>INT2CRC</td>
<td>Schedule message to interceptor to fly towards CRC (Return to)</td>
</tr>
<tr>
<td>25690001</td>
<td>1600</td>
<td>BNCONLS</td>
<td>Schedule commo CRC target lost</td>
</tr>
<tr>
<td>25690001</td>
<td>1600</td>
<td>BNCMDPR</td>
<td>Commo CRC loss of sight of target</td>
</tr>
<tr>
<td>25690001</td>
<td>1600</td>
<td>BOCTINK</td>
<td>Schedule commo CRC target unseen</td>
</tr>
<tr>
<td>25690001</td>
<td>1620</td>
<td>BNCONTC</td>
<td>Schedule commo CRC consider track change</td>
</tr>
<tr>
<td>25690001</td>
<td>2752</td>
<td>BYPONRL</td>
<td>Schedule commo BTN reload</td>
</tr>
<tr>
<td>25690001</td>
<td>2761</td>
<td>BYHEDUP</td>
<td>Schedule commo BTN no chance</td>
</tr>
<tr>
<td>25690001</td>
<td>2761</td>
<td>BNLALLE</td>
<td>Schedule commo superior, no chance</td>
</tr>
<tr>
<td>25690001</td>
<td>2762</td>
<td>BYCONLS</td>
<td>Track loss for BTN</td>
</tr>
<tr>
<td>25690001</td>
<td>2762</td>
<td>BYNWTRK</td>
<td>Schedule commo BTN track change</td>
</tr>
<tr>
<td>25690001</td>
<td>2762</td>
<td>BYCONTC</td>
<td>Schedule commo BTN track change</td>
</tr>
<tr>
<td>25690001</td>
<td>2762</td>
<td>BTRYTNK</td>
<td>Schedule commo BTN target not sighted</td>
</tr>
<tr>
<td>25690001</td>
<td>2780</td>
<td>BYPONER</td>
<td>Schedule engagement results for BTN</td>
</tr>
<tr>
<td>25690001</td>
<td>2790</td>
<td>BYPONFD</td>
<td>Schedule commo BTN flight dead</td>
</tr>
<tr>
<td>25690001</td>
<td>2831</td>
<td>ALLOBAT</td>
<td>Btry engage command</td>
</tr>
<tr>
<td>EVENT CODE</td>
<td>MESSAGE CODE</td>
<td>SUBROUTINE</td>
<td>EVENT</td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------</td>
</tr>
<tr>
<td>25690001</td>
<td>2832</td>
<td>BNPONEP</td>
<td>Schedule commo Btry cease fire</td>
</tr>
<tr>
<td>25690001</td>
<td>2832</td>
<td>DECRALO</td>
<td>Cease fire msg for battery</td>
</tr>
<tr>
<td>25690001</td>
<td>2832</td>
<td>DROPPOS</td>
<td>Schedule commo btry ceasefire</td>
</tr>
<tr>
<td>25690001</td>
<td>2832</td>
<td>DROPPS2</td>
<td>Schedule commo btry ceasefire</td>
</tr>
<tr>
<td>25690001</td>
<td>2834</td>
<td>ALLOBAT</td>
<td>Btry coverage update</td>
</tr>
<tr>
<td>25690001</td>
<td>2834</td>
<td>DECRALO</td>
<td>Coverage update message</td>
</tr>
<tr>
<td>25690001</td>
<td>2835</td>
<td>BNCONHD</td>
<td>Schedule commo Btry engagement data update</td>
</tr>
<tr>
<td>25930001</td>
<td>1700</td>
<td>GOGETEM</td>
<td>Call CRC to notify of takeoff</td>
</tr>
<tr>
<td>31691000</td>
<td></td>
<td>AIRTHNK</td>
<td>Schedule dogfite</td>
</tr>
<tr>
<td>34342950</td>
<td></td>
<td>ENGANZE</td>
<td>Schedule engage Btry fire</td>
</tr>
<tr>
<td>34342951</td>
<td></td>
<td>ENGANGE</td>
<td>Schedule engage Btry fire</td>
</tr>
<tr>
<td>34692900</td>
<td></td>
<td>ALLOPAT</td>
<td>Schedule engage Btry (Lockon)</td>
</tr>
<tr>
<td>34692900</td>
<td></td>
<td>ALLOFU</td>
<td>Schedule engage Btry (Lockon)</td>
</tr>
<tr>
<td>34692900</td>
<td></td>
<td>BYTKCHK</td>
<td>Schedule engage Btry (Lockon)</td>
</tr>
<tr>
<td>34692901</td>
<td></td>
<td>BYPONER</td>
<td>Schedule engage to fire again</td>
</tr>
<tr>
<td>34692901</td>
<td></td>
<td>PTPONER</td>
<td>Schedule engage to fire again</td>
</tr>
<tr>
<td>34692950</td>
<td></td>
<td>BYPONTM</td>
<td>Schedule engage Btry fire</td>
</tr>
<tr>
<td>39391313</td>
<td></td>
<td>FLITE</td>
<td>Schedule next fly</td>
</tr>
<tr>
<td>39931313</td>
<td></td>
<td>GOGETEM</td>
<td>Schedule launch of interceptor</td>
</tr>
<tr>
<td>39931313</td>
<td></td>
<td>TOWER</td>
<td>Schedule fly flight take off</td>
</tr>
<tr>
<td>62141100</td>
<td></td>
<td>ATKASES</td>
<td>Schedule death perception</td>
</tr>
<tr>
<td>62311100</td>
<td></td>
<td>DOGFITE</td>
<td>Schedule death perception</td>
</tr>
<tr>
<td>62391100</td>
<td></td>
<td>SHRKILL</td>
<td>Schedule ptrgod to perceive death</td>
</tr>
<tr>
<td>62391313</td>
<td></td>
<td>FLITE</td>
<td>Schedule Naybor flying plane</td>
</tr>
<tr>
<td>62391380</td>
<td></td>
<td>COMMAND</td>
<td>Schedule Naybor ECM on</td>
</tr>
<tr>
<td>62391390</td>
<td></td>
<td>COMMAND</td>
<td>Schedule Naybor ECM OFF</td>
</tr>
</tbody>
</table>
## MADEM EVENT CODE DEFINITIONS

<table>
<thead>
<tr>
<th>EVENT CODE</th>
<th>SCHEDULING SUBROUTINE</th>
<th>EVENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>62931100</td>
<td>TOWER</td>
<td>Schedule naybor others notice landing</td>
</tr>
<tr>
<td>62951100</td>
<td>UMPIRE</td>
<td>Schedule naybor flight death</td>
</tr>
<tr>
<td>65141100</td>
<td>SAMWYPE</td>
<td>Percept superior subordinate dead</td>
</tr>
<tr>
<td>65141120</td>
<td>ATTACK</td>
<td>Schedule perception of attack results</td>
</tr>
<tr>
<td>65250000</td>
<td>COMMO</td>
<td>Schedule percent reciever (msg)</td>
</tr>
<tr>
<td>65311001</td>
<td>DOGFITE</td>
<td>Schedule target to perceive attack</td>
</tr>
<tr>
<td>65391350</td>
<td>COMMAND</td>
<td>Schedule rendezvous perception of all flights in formation</td>
</tr>
<tr>
<td>65391400</td>
<td>COMMAND</td>
<td>Schedule ground target perception</td>
</tr>
<tr>
<td>65620000</td>
<td>NAYBOR</td>
<td>Schedule perception of unit.</td>
</tr>
<tr>
<td>65621100</td>
<td>DESTROY</td>
<td>Notify CRC of Btn death</td>
</tr>
<tr>
<td>65651100</td>
<td>KILFLIT</td>
<td>Notify CRC of subordinate death</td>
</tr>
<tr>
<td>65691400</td>
<td>FLYSEE</td>
<td>Resume ground attack</td>
</tr>
<tr>
<td>67671234</td>
<td>MADEM</td>
<td>Schedule red commander to begin plan of Raid.</td>
</tr>
<tr>
<td>67674141</td>
<td>THTRPLN</td>
<td>Plan next raid</td>
</tr>
<tr>
<td>69142,91</td>
<td>SAMWYPE</td>
<td>Ponder superior subordinate dead</td>
</tr>
<tr>
<td>69142793</td>
<td>SAMWYPE</td>
<td>Ponder Btry superior dead</td>
</tr>
<tr>
<td>69311905</td>
<td>DOGFITE</td>
<td>Schedule dogfite outcome ponder</td>
</tr>
<tr>
<td>69342890</td>
<td>RELOAD</td>
<td>Schedule ponder Btry reload</td>
</tr>
<tr>
<td>69391375</td>
<td>FLY</td>
<td>Schedule ponder air combat</td>
</tr>
<tr>
<td>6965001</td>
<td>SAMSEE</td>
<td>Schedule ponder message</td>
</tr>
<tr>
<td>69651175</td>
<td>FLYSEE</td>
<td>Schedule interceptor message ponder</td>
</tr>
<tr>
<td>69651375</td>
<td>FLYSEE</td>
<td>Schedule Air combat ponder</td>
</tr>
<tr>
<td>EVENT CODE</td>
<td>SCHEDULING SUBROUTINE</td>
<td>EVENT</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------</td>
<td>-------</td>
</tr>
<tr>
<td>69651400</td>
<td>CRCEVNT</td>
<td>Schedule CRC ponder</td>
</tr>
<tr>
<td>69651410</td>
<td>CRCEVENT</td>
<td>Movement event code</td>
</tr>
<tr>
<td>69651420</td>
<td>CRCEVNT</td>
<td>Loss of sight</td>
</tr>
<tr>
<td>69651460</td>
<td>CRCSEE</td>
<td>Message from Airbase</td>
</tr>
<tr>
<td>69651470</td>
<td>CRCSEE</td>
<td>Message from interceptor</td>
</tr>
<tr>
<td>69651480</td>
<td>CRCSEE</td>
<td>Message from BTN</td>
</tr>
<tr>
<td>69652790</td>
<td>SAMSEE</td>
<td>Schedule ponder flight death</td>
</tr>
<tr>
<td>69652800</td>
<td>BYTKCHK</td>
<td>Ponder track move</td>
</tr>
<tr>
<td>69652805</td>
<td>BYTKCHK</td>
<td>Ponder track lost</td>
</tr>
<tr>
<td>69692700</td>
<td>ACCEPT</td>
<td>Ponder CE track info digest</td>
</tr>
<tr>
<td>69692700</td>
<td>BNLALLE</td>
<td>Schedule digest event</td>
</tr>
<tr>
<td>69692700</td>
<td>FILERUP</td>
<td>Schedule ponder digest</td>
</tr>
<tr>
<td>69692700</td>
<td>NEWPERC</td>
<td>Schedule ponder CE (track info digest)</td>
</tr>
<tr>
<td>69692700</td>
<td>READIL</td>
<td>Schedule ponder Btry digest</td>
</tr>
<tr>
<td>69692700</td>
<td>SAMATON</td>
<td>Schedule ponder Sam (Track info digest)</td>
</tr>
<tr>
<td>69692700</td>
<td>SDIGEST</td>
<td>Ponder CE Digest track info</td>
</tr>
<tr>
<td>69692705</td>
<td>CHKCOV</td>
<td>Ponder BTN delayed action</td>
</tr>
<tr>
<td>69692705</td>
<td>SEEKENG</td>
<td>Ponder BTN delayed action</td>
</tr>
<tr>
<td>69692705</td>
<td>SEEKTAC</td>
<td>Ponder BTN delayed action</td>
</tr>
<tr>
<td>69692710</td>
<td>BYNWTRK</td>
<td>Ponder Btry last chance</td>
</tr>
<tr>
<td>69692710</td>
<td>BTHEDUP</td>
<td>Schedule new last chance event</td>
</tr>
<tr>
<td>69692710</td>
<td>BYCONHD</td>
<td>Schedule ponder Btry last chance</td>
</tr>
<tr>
<td>69692710</td>
<td>BTRYTNK</td>
<td>Last chance event</td>
</tr>
<tr>
<td>69692710</td>
<td>BNPONBB</td>
<td>Schedule ponder BTN last chance event</td>
</tr>
<tr>
<td>69692710</td>
<td>CHKLAST</td>
<td>Ponder BTN last chance</td>
</tr>
<tr>
<td>EVENT CODE</td>
<td>SCHEDULING SUBROUTINE</td>
<td>EVENT</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------</td>
<td>-------</td>
</tr>
<tr>
<td>69692710</td>
<td>PREPAFU</td>
<td>Schedule ponder Btry last chance</td>
</tr>
<tr>
<td>69692710</td>
<td>SETASSN</td>
<td>Ponder BTN chance gone</td>
</tr>
<tr>
<td>69692715</td>
<td>ACCEPT</td>
<td>Schedule expected sighting event</td>
</tr>
<tr>
<td>69692715</td>
<td>BTRYTNK</td>
<td>Reschedule expected sighting event</td>
</tr>
<tr>
<td>69692720</td>
<td>CHKCOV</td>
<td>Schedule opportunity knocks event</td>
</tr>
<tr>
<td>69692720</td>
<td>BNPONBB</td>
<td>Schedule opportunity knocks event</td>
</tr>
<tr>
<td>69692890</td>
<td>ADASASS</td>
<td>Schedule 1st reload/resupply for each fire unit</td>
</tr>
<tr>
<td>69692890</td>
<td>BYPONRL</td>
<td>Schedule next reload</td>
</tr>
<tr>
<td>69692895</td>
<td>RESUPLY</td>
<td>Schedule ponder Btry resupply</td>
</tr>
<tr>
<td>69952790</td>
<td>UMPIRE</td>
<td>Schedule ponder Btry flight death</td>
</tr>
<tr>
<td>69952792</td>
<td>NUKBLND</td>
<td>Schedule ponder BTN (subordinate) cut off</td>
</tr>
<tr>
<td>69952794</td>
<td>NUKBLND</td>
<td>Schedule ponder Btry (superior cut off)</td>
</tr>
<tr>
<td>69952880</td>
<td>UMPIRE</td>
<td>Schedule ponder Btry engagement results</td>
</tr>
<tr>
<td>69992793</td>
<td>CRCDIES</td>
<td>Schedule superior death event for subordinate</td>
</tr>
<tr>
<td>93651312</td>
<td>ABSEE</td>
<td>No launch event scheduled. Schedule launch event</td>
</tr>
<tr>
<td>93651984</td>
<td>ABSEE</td>
<td>Schedule tower to land flight</td>
</tr>
<tr>
<td>93671313</td>
<td>ACFRAG</td>
<td>Schedule takeoff for flight</td>
</tr>
<tr>
<td>93931312</td>
<td>GOGETEM</td>
<td>Schedule another flight for tower to launch</td>
</tr>
<tr>
<td>93931312</td>
<td>TOWER</td>
<td>Notify commander of flight landing</td>
</tr>
<tr>
<td>95343000</td>
<td>ENGAGE</td>
<td>Schedule UMPIRE</td>
</tr>
</tbody>
</table>
APPENDIX G
RANDOM NUMBER GENERATOR CALLS

MADEM uses the random number generator RANF in all of its' MONTE-CARLO actions. The seed for this uniform random number generator is set using the routine RSEED. Both RANE and RSEEDE are CDC supplied routines.
## ALL USES OF RANF AND RSEED IN MADEM

<table>
<thead>
<tr>
<th>SUBROUTINE</th>
<th>RANF LINE NUMBERS</th>
<th>RSEED LINE NUMBER</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIRTANK</td>
<td>70</td>
<td>70</td>
<td>R</td>
</tr>
<tr>
<td>ATTACK</td>
<td>81</td>
<td>61</td>
<td>R</td>
</tr>
<tr>
<td>CANOTGT</td>
<td>75</td>
<td>75</td>
<td>R</td>
</tr>
<tr>
<td>CLIST</td>
<td>--</td>
<td>43</td>
<td>R</td>
</tr>
<tr>
<td>CRCEVENT</td>
<td>98</td>
<td>98</td>
<td>R</td>
</tr>
<tr>
<td>CRCTRACK</td>
<td>83</td>
<td>83</td>
<td>R</td>
</tr>
<tr>
<td>DETECT</td>
<td>89</td>
<td>89</td>
<td>R</td>
</tr>
<tr>
<td>DOGFITE</td>
<td>72</td>
<td>72</td>
<td>R</td>
</tr>
<tr>
<td>FILERUP</td>
<td>28</td>
<td>28</td>
<td>R</td>
</tr>
<tr>
<td>GNDLOOK</td>
<td>35</td>
<td>35</td>
<td>R</td>
</tr>
<tr>
<td>NEWMOVE</td>
<td>29</td>
<td>29</td>
<td>R Problem of Error Increases Over Time</td>
</tr>
<tr>
<td>OUTPTRS</td>
<td>--</td>
<td>27</td>
<td>R</td>
</tr>
<tr>
<td>PTRAND</td>
<td>16, 33, 38</td>
<td>16, 33, 38</td>
<td>R</td>
</tr>
<tr>
<td>RELEASE</td>
<td>--</td>
<td>22, 34, 45</td>
<td>(Writes) R</td>
</tr>
<tr>
<td>SCHEDUL</td>
<td>38, 50</td>
<td>38, 50</td>
<td>R</td>
</tr>
<tr>
<td>SDIGEST</td>
<td>58</td>
<td>58</td>
<td>R</td>
</tr>
<tr>
<td>SHRKILL</td>
<td>30</td>
<td>30</td>
<td>R</td>
</tr>
<tr>
<td>VMPIRE</td>
<td>37</td>
<td>37, 37, 39</td>
<td>RSEED=RANCRSEED (SET)</td>
</tr>
</tbody>
</table>

R = reference only  
SET = RSEED set in the subroutine
APPENDIX H
MADEM SUBROUTINE REFERENCE LISTS
AND CALLING HIERARCHIES

1. Preprocessor

LIST OF SUBROUTINES - MADEW

1. A8QUEUE
2. A8VSCOR
3. A8FRAG
4. A8ASSS
5. A8BLOK
6. A8OCR
7. A8DUMP
8. A8CEL1
9. A8CEL2
10. A8SIGN
11. A8TACK
12. A8VLBL
13. A8ALT
14. A8LEX
15. A8LXK
16. A8PAS
17. A8KDAT
18. A8DGT
19. A8ARD
20. A8CENTER
21. A8GEN
22. A8LIST
23. A8LIST2
24. A8LSCOR
25. A8DE01
26. A8DE03
27. A8DE05
28. A8DE18
29. A8MDW
30. A8NTROL
31. A8BON
32. A8CREATE
33. A8FTRML
34. A8READ
35. A8CD
36. A8LADD
37. A8TS+X
38. A8SPAQ
39. A8SPACD
40. A8SPACL
41. A8SPACR
42. A8SPADE
43. A8SPAQD
44. A8SPLAT
45. A8SPDAB
46. A8SPFAT
47. A8SPMT
48. A8SPAR
49. A8SPRAY
50. A8SPRO
51. A8SPRYB
52. A85DEC
LIST OF FORTRAN LIBRARY ROUTINES

1. ALOG.
2. ASIN.
3. ATAN2.
4. ATAN.
5. COS.
6. DEGEN.
7. ENDPI.
8. END.
9. GOTOER.
10. INPBI.
11. IMPC1.
12. IMPCK.
13. IMPFI.
14. ITOJ.
15. OUTBI.
16. OUTC1.
17. OUTCR.
18. QINTRY.
19. RANDOM.
20. REWIND.
21. SIN.
22. STOP.
23. TAN.
24. TAPE.
25. ATOI.
<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Calls</th>
<th>Called By</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABQUEUE</td>
<td>ENTRYP GIMME ADDBLOK FINOBLK EXITP</td>
<td>CODE05 CODE03</td>
</tr>
<tr>
<td>ABVSCOR</td>
<td>ENTRYP CLOS COR GIMME ADDBLOK EXITP</td>
<td>REVISE</td>
</tr>
<tr>
<td>ACFRAQ</td>
<td>ENTRYP GIMME CRFLTML ADDBLOK OFP14 RELEASES FLTGEOM HEXOIST DELADD EXITP</td>
<td>SCHEDUL</td>
</tr>
<tr>
<td>A04ASSS</td>
<td>ENTRYP GIMME FINOBLK GOTOER. DELADD ADDBLOK EXITP</td>
<td>CODE03</td>
</tr>
<tr>
<td>ADDBLOK</td>
<td>ENTRYP EXITP</td>
<td>UOLLOAD T3TLIST SEVANT OTHROAT N0WUCIT INITACQ FINOFILT CRFLTML C &amp; D OTST A04ASSS ACFRAQ A04ASSS ACFRAQ ABVSCOR ABQUEUE</td>
</tr>
<tr>
<td>ADDCVR</td>
<td>CALLS</td>
<td>CALLED BY</td>
</tr>
</tbody>
</table>
7. ADUMP
   CALLS:
   CALLED BY:
   LEXAN
   ISHIFT
   TAPE6#
   OUTCI,
   CALLED BY:
   L3KPR5
   OUTCR.

8. APCEL1
   CALLS:
   CALLED BY:
   L3KPR5
   ISHIFT
   TAPE6#
   OUTCI,
   CALLED BY:
   L3KPR5
   OUTCR.

9. APCEL2
   CALLS:
   CALLED BY:
   L3KPR5
   ISHIFT
   TAPE6#
   OUTCI.

10. ASSIGN
    CALLS:
    CALLED BY:
    SELECT
    CALLED BY:
    SELECT

11. ATTACK
    CALLS:
    CALLED BY:
    THTRPLN
    ENTRYP
    PELADD
    RELEASE
    EXITP

12. AVAIL9L
    CALLS:
    CALLED BY:
    THTRPLN

13. 3DALT
14. 80LEX
15. 80LAK
16. 80PARS
17. 8LKDAT

18. CANDT6T
    CALLS:
    CALLED BY:
    THTRPLN
    ENTRYP
    FINOBLK
    CLOGCOR
    JGESUIT
    FORMSTG
    GIMME
    ADDBLOK
    TGTGONE
    PTREE
    RANDOM.
    PELADD
    EXITP

19. CARD
    CALLS:
    CALLED BY:

325
20. CENTER
CALLS: ENTRYP EXITP
CALLED BY: TXY2HXL TLL2MX

21. CHARGE
CALLS: CARD TAPE6# OUTCI.
CALLED BY: LEXAN EXTSCN

22. CLIST
CALLS: PAGE LNPLST MESSAE RITEP RITEI TAPE6# OUTCI. OUTCR. CLIST
CALLED BY: HALT HEXCHZ FSDUMP FINDBLK

23. CLIST2
CALLS: PAGE LNPLST MESSAE RITEI RITEP TAPE6# OUTCI. OUTCR. RITEP
CALLED BY: CLIST

24. CLOSOR
CALLS: ENTRYP HEXDIST EXITP
CALLED BY: CANOTGT A9VSCOR

25. CODE01
CALLS: ENTRYP GIMME RITEP EXITP
CALLED BY: SEVANT

26. CODE03
CALLS: ENTRYP SRCNPL CREATE LOADPL RITEP LNPLST
CALLED BY: SEVANT
27. CODE05 CALLS:
ENRTYP
SRCHPL
FINDBLK
GETHEK
RITEI
RITEP
LOAD
LOAD
UOLLOAD
ABQUEUE
TOTALIST
INITACQ
HISTORY
EXITP

CALLED BY:
SEMANT

28. CODE18 CALLS:
GETHEK
GINME
PACK
UNPACK
LOAD
RITEP
RITEI
LOAD
LOAD
TOTALIST
HISTORY

CALLED BY:
SEMANT

29. COMMON

CALLED BY:
SELECT

30. CONTROL CALLS:
ENRTYP
LTREE
SELECT
SNAP
UNSNAP
RELEASE
SECOND
HLPLOT
EXITP

CALLED BY:
META

31. CORBON CALLS:
ENRTYP
GIME
THWPS
TWX2HXL
INDEXIST
SIN
COS
LINE
OPTPATH

CALLED BY:
TNTRPLY
32. CREATE CALLS:
ENTRYP
GIMME
EXITP
CALLED BY:
SRCMHPL
CRFTML
CODE03

33. CRFTML CALLS:
ENTRYP
CREATE
GIMME
ADDBLOK
HISTORY
EXITP
CALLED BY:
ACFRAG

34. OGREAD CALLS:
OUTCI.
INPC1.
EDF
DEC001.
CALLED BY:
WOEDM

35. DECOMS CALLS:
ENTRYP
MESSAGE
TRACE
RECE
GIMME
SNAP
LTRANR
EXITP
CALLED BY:
TWTRPL
A0ASAS
ACFRAG
WOEDM

36. VALADD CALLS:
ENTRYP
MESAGE
TRACE
RECE
GIMME
SNAP
LTRANR
EXITP
CALLED BY:
TWTRPL
A0ASAS
ACFRAG
WOEDM

37. DGTSIX CALLS:
ITOJ.
CALLED BY:

38. DISPABQ CALLS:
OUTCI.
CALLED BY:
DISPDAT

39. DISPACD CALLS:
OUTCI.
CALLED BY:
DISPFILT
DISPDAT

40. DISPACL CALLS:
OUTCI.
CALLED BY:
DISPACR

41. DSPACR CALLS:
OUTCI.
DISPACL
CALLED BY:
DISPACR

42. DISPADO CALLS:
OUTCI.
CALLED BY:
DISPDAT

43. DISPADQ CALLS:
OUTCI.
CALLED BY:
DISPFILT
44. DISPDAT CALLS:
OUTCI.,
DISPADC
DISPDB
DISPFHL
DISPACD
DISPRAF
DISPRAO
DISPAPD
DISPACH

45. DISPFOB CALLS:
OUTCI.,
DISPFHF

46. DISPFHL CALLS:
OUTCI.,
DISPACD
DISPRAO
DISPAPD

47. DISPFHF CALLS:
OUTCI.,
DISPFHL

48. DISPRAF CALLS:
OUTCI.,
DISPFHL

49. DISPFRY CALLS:
OUTCI.,
DISPFHL

50. DISPDFR CALLS:
OUTCI.

51. DISPFRO CALLS:
OUTCI.

52. DM5DEC CALLS:
ENTRYP
EXIT

53. DQDFTE CALLS:
ENTRYP
RELEASE

54. DROPBLK CALLS:
ENTRYP
RELEASF
55. ENGAGE
56. ENTRYP

CALLS:

MESSAG
RITEI
RECEI
TAPES
OUTCI.
ROUTER
ITRAP
SECOND

CALLED BY:

SELECT

CALLED BY:

LTMNG
UNPACK
UNSNAP
TXY2HXL
TXY2MX
TIME
TL2MX
TM2XY
THTRPLV
TTHPSP
TGLIST
TGTGONE
SRCHPL
SNAP
SENANT
SELECT
SCMTAB
SCHEDUL
GLAVE
GLTGTYP
GLTATAK
LCH877
LFTMAI
LCORD
RLABDB
REVISE
REVDEVU
RELST
RELEASE
PTREE
PLANOUT
PLAN
PELADD
PACK
OUTA
OPDRAOT
OPTPDM
NOWUCIT
LTREE
L3ADPL
L1NEX
KOMPARE
JUGGLE
JTJ
JUGESUIT
INITAICQ
INIT
I12MX
57. ENTSTAT CALLS TAPE6# OUTCI.
   CALLED BY: MALT FSDUMP
58. EOF
   CALLED BY: OBBREAD CARD
59. ERROR CALLS TAPE6# OUTCI.
   CALLED BY: LMKPSR
60. EXIT CALLS SECOND MESSAGE
   CALLED BY: LTSWAP UNPACK
HEXADD
GINHE
GETPTRS
GETHEX
FSIMET
FSHIM
FORMTGT
FLTGEOM
FINDFLT
FINDBLK
FETCH
DROPBLK
DSWDEC
DELADE
CRFLTML
CREATE
CORB0UN
CONTROL
CODEO6
CODEO3
CODEO1
CLOSEDE
CENTER
CANDOTGT
4AVAILBL
4DBBLK
4DASASS
4CFRAG
4BVSDEC
4BQUEUE
MADEM

61. EXTSCN CALLS: CALLED BY:
CALLS: CALLED BY:
CHRGEN LEXAN
IT0J.
62. FELDEL CALLS: CALLED BY:
CALLED BY:
63. FETCH CALLS: CALLED BY:
ENTRP SELECT
INPBI.
EXITP MADEM
64. FINDBLK CALLS: CALLED BY:
ENTRP T4TRPLY
ROUTER T3LIST
MESSAGE SEWANT
RETEI OTHERDAT
CLIST NOWUC1T
EXITP FORMTGT
CALLED BY:
FINDFLT FORMTGT
CODEO5
CANDOTGT
4DASASS
4DASASS
<table>
<thead>
<tr>
<th>Page 16</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>65. FINDFLT</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>66. FLTDEOM</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>67. FLY</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>68. FORMGT</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>69. F5DUMP</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>70. FSINIT</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>71. GETHEX</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
72. GETPTRS
CALLS:
ENTRYP
EXITP
CALLED BY:
PLAN

73. GIMME
CALLS:
ENTRYP
HALT
MESSAGE
RITEI
RITEP
EXITP
CALLED BY:
ULOAD
TGLIST
TSTGONE
SEMAN
REVIEW
RELEASE
PELLADE
OTTHROAT
OPTPTH
NOWUCIT
INITACO
INIT
GETHEX
FORMTG
FINDTFLY
DELADO
CREFLTM
CREATE
CORBOUN
CODE18
CODE1
CODCTG
ADASASS
ACFRAG
ABIVSCOR
ABQUEUE

74. HALT
CALLS:
HOLD
OUTCI.
RITEI.
RITEI
TRACE
RECR
ENTSTAT
CLIST
PAGE
ADDUM
ISDUMP
ENDFIL.
STOP.
CALLED BY:
UPACK
ITRAP
RECCON
SELECT
RELEASE
PACK
OTTHROAT
MLTPNT
SEHME
MENSE

75. MEXADD
CALLS:
ENTRYP
ETOJ.
EXITP
CALLED BY:
SCHEDULE
NOWUCIT
MEXMLT
MEXCHZ

76. MEXCHZ
CALLS:
CALLED BY:

<table>
<thead>
<tr>
<th>Page 18</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRYP</td>
</tr>
<tr>
<td>77. HEXDIST</td>
</tr>
<tr>
<td>79. HEXMUX</td>
</tr>
<tr>
<td>80. HEXMULT</td>
</tr>
<tr>
<td>81. HISTORY</td>
</tr>
<tr>
<td>82. HLTPNT</td>
</tr>
<tr>
<td>83. HOLD</td>
</tr>
</tbody>
</table>
CALLS: ENTRYP EXITP

TH2HX
TH4XXY
HXMLT
GETHXX

CALLS: TH2HX
I2HXX

CALLS: OUTC1.

CALLS: HXMLT
ENTRYP
I2OJ
EXITP

CALLS: ENTRYP
XTOI.
GIMME
RITEP
SNAP
EXITP

CALLS: ENTRYP
GIMME
ADDBLOK
ALOG.
MESSAGE
RITEI
XTOI.
NOWCIT
EXITP

CALLS: LCMLOC
IS1HFT
TAPE6
OUTC1.

CALLS: OUTC1.
OUTCR.

CALLS: OUTCI.

CALLS: MALT
UNPACK
PACK

CALLS: APCEL2
LACELL
APCELL
ROCELL
93. ITRAP

CALLS:
- HALT

CALLED BY:
- EXITP
- ENTRYP
- TMTRPLY

94. JGESUIT

CALLS:
- ENTRYP
- TMX2XY
- ATAN2
- EXITP

CALLED BY:
- CANOTOT

95. JTJ

CALLS:
- ENTRYP
- MESSAGE
- RITEI
- ISHIFT
- TAPE6#
- OUTCI,
- FSODUMP
- STOP
- EXITP

96. JUGGLE

CALLS:
- ENTRYP
- EXITP

CALLED BY:
- TMX2XY

97. KOMPARE

CALLS:
- ENTRYP
- UNPACK
- PACK
- EXITP

CALLED BY:
- TMTRPLY
- FINDFLT

98. LACELI

CALLS:
- ISHIFT
- TAPE6#
- OUTCI,

CALLED BY:
- LRKPRS

99. LCMLOC

CALLS:
- UNPACK
- RITEP
- PACK
- IPUL
- FSODUMP
- MADEM

100. LEXAN

CALLS:
- CHRGEM

CALLED BY:
- NRTSYM
ISHIFT
GOTOER.
ADDCHR
LOOKUP
EXTSCN
TAPE6#
OUTCI.

101. LINEX
CALLS:
ENTRYP
EXITP

102. LUPLT
CALLS:
TAPE6#
OUTCI.

103. LOADPL
CALLS:
ENTRYP
EXITP

104. LOOKUP
CALLS:
ISHIFT
TAPE6#
OUTCI.

105. LAXDRS
CALLS:
SEMANT
TAPE6#
OUTCI.
ROCELL
NATSYN
ERROR
ISHIFT
APCELL
APCEL2
LACELL

106. LTREE
CALLS:
ENTRYP
RELEASE
LTMRG
EXITP

107. LTMRG
CALLS:
ENTRYP
EXITP

108. MADEM
CALLS:
QENTRY.

CALLED BY:
OPTPTH
CORBourn

CALLED BY:
CLIST2
SEMANT
OUTPTRS
PROM
CODE18
CODE05
CODE03
CLIST

CALLED BY:
CODE03

CALLED BY:
LEXAN

CALLED BY:
MADEM

CALLED BY:
CONTROL

CALLED BY:
LTREE
DELADO

339
109. WASKER

110. MESSAGE

111. WARYB

112. WOUUC

RECON
ENTRYP
INPF1.
DBGREAD
RECOVR
GOTOER.
FETCH
PAGE
LCMLOC
FSINIT
OTHERDAT
DISPOAT
MALT
INIT
LKPRS
RELIST
DELETE
CONTROL
EXITP
END.

109. WASKER

110. MESSAGE

111. WARYB

112. WOUUC

CALLED BY: HISTORY

CALLED BY: CLIST2
ENTRYP
SELECT
RELEASE
PACK
OUTPTRS
JTO
INITAC3
HISTORY
MEXMLT
MEXCMZ
GIMME
GETHEX
FINDBLK
DELETE
CLIST

CALLED BY: SELECT

CALLED BY: INITAC3
<table>
<thead>
<tr>
<th>Page</th>
<th>CALLS1</th>
<th>CALLED BY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>113. NAISY</td>
<td>LEXAN TAPE6# OUTCI.</td>
<td>L9KPRS</td>
</tr>
<tr>
<td>114. DATPTM</td>
<td>ENTRYP GIMME THM2PS MEXCH2 LINEX EXITP</td>
<td>CORBON ACFRAO</td>
</tr>
<tr>
<td>115. OHROAT</td>
<td>ENTRYP GIMME INPCL TAPE6# OUTCI RITEP ADDBLK INPFI FINOBLK MALT EXITP</td>
<td>NADEM</td>
</tr>
<tr>
<td>116. OUTA</td>
<td>ENTRYP TAPE6# OUTCI. EXITP</td>
<td></td>
</tr>
<tr>
<td>117. OUTP65</td>
<td>LNPLCT MESSAGE RITEI RITEP RITEP UNPACK</td>
<td></td>
</tr>
<tr>
<td>118. PACK</td>
<td>ENTRYP LCMLOC PAGE MESSAGE RITEI TRACE ROUTER RITEP ISOUMP MALT EXITP</td>
<td>SEMANT KMPARE CODE10</td>
</tr>
<tr>
<td>Page</td>
<td>Calls</td>
<td>Called By</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>119</td>
<td>CALLS: TAPE6# OUTCI.</td>
<td>CALLED BY: CLIST2, HALT, PACK, CLIST, W40EM</td>
</tr>
<tr>
<td>120</td>
<td>CALLS: ENTRYP GIMME PTRMRG EXITP</td>
<td>CALLED BY: TSTLIST, CANDTGT, AVAILBL</td>
</tr>
<tr>
<td>121</td>
<td>CALLS: ENTRYP GETPRPS TMTRPLN EXITP</td>
<td>CALLED BY: SELECT</td>
</tr>
<tr>
<td>122</td>
<td>CALLS: ENTRYP GETPRPS TMTRPLN EXITP</td>
<td>CALLED BY: SELECT</td>
</tr>
<tr>
<td>123</td>
<td>CALLS: ENTRYP OUTCI. EXITP</td>
<td>CALLED BY: TMTRPLN</td>
</tr>
<tr>
<td>124</td>
<td>CALLS: ENTRYP RELEASE ISHIFT EXITP</td>
<td>CALLED BY: SELECT</td>
</tr>
<tr>
<td>125</td>
<td>CALLS: ENTRYP RELEASE ISHIFT EXITP</td>
<td>CALLED BY: TSTGONE, CANDTGT</td>
</tr>
<tr>
<td>126</td>
<td>CALLS: ENTRYP RELEASE ISHIFT EXITP</td>
<td>CALLED BY: PTRMRG</td>
</tr>
<tr>
<td>127</td>
<td>CALLS: ENTRYP RELEASE ISHIFT EXITP</td>
<td>CALLED BY: PELADD</td>
</tr>
<tr>
<td>128</td>
<td>CALLS: ENTRYP RELEASE ISHIFT EXITP</td>
<td>CALLED BY: L9KPRS</td>
</tr>
<tr>
<td>129</td>
<td>CALLS: ENTRYP RELEASE ISHIFT EXITP</td>
<td>CALLED BY: MADEM</td>
</tr>
<tr>
<td>130</td>
<td>CALLS: ENTRYP RELEASE ISHIFT EXITP</td>
<td>CALLED BY: HAMLE, EXITP, ENTRYP</td>
</tr>
</tbody>
</table>

342
130. RECOVQ

131. RELEASE

CALLS:
ENTRYP
MESSAGE
RITEI
RITEP
HALT
GIMME
EXITP

CALLED BY:
MADEN

132. RELIST

CALLS:
ENTRYP
RELEASE
EXITP

CALLED BY:
MADEN

133. RENDEVU

CALLS:
ENTRYP
THAXXY
TAX2HAL
EXITP

CALLED BY:
SCHSUL

134. REVISE

CALLS:
ENTRYP
GIMME
ABV4COR
RELEASE
EXITP

CALLED BY:
THTRPLY

135. RITEI

CALLS:
TAPE5B
OUTCI.

CALLED BY:
CLISTZ
HALT
EXITP
ENTRYP
SEMANT
SELECT
RELEASE
PACK
OUTPTRS
JTJ
INITACQ
HEXMLT
136. RITEP  CALLS:
   LCMLOC
   TAPE#
   OUTCI.

   CALLED BY:
   CLIST2
   UNPACK
   SEMANT
   RELEASE
   PACK
   OUTPUTS
   OTHRDAT
   INIT
   HEXCHZ
   GIME
   CODE18
   CODE05
   CODE03
   CODE01
   CLIST

137. RITER  CALLS:
   TAPE#
   OUTCI.

   CALLED BY:
   CLIST2
   MALT
   SEMANT
   OUTPUTS
   FSDUMP
   CLIST

138. RLAB03  CALLS:
   ENTRY
   RELEASE
   EXITP

   CALLED BY:
   RLCORD

139. RLCORD  CALLS:
   ENTRY
   RLAB08
   RELEASE
   EXITP

   CALLED BY:
   RLRAID

140. RLFMAK1  CALLS:
   ENTRY
   RELEASE
   EXITP

   CALLED BY:
   RLTGTAK

141. RLRAID  CALLS:
   ENTRY
   RLVAVE
   RLCORD
   RELEASE
   EXITP

   CALLED BY:
   TMTRPL4
142. $	ext{RLTGTK}$

**Calls:**
- Entry
- Release
- Exit

**Called by:**
- RLTGTK

143. $	ext{RLTGTK}$

**Calls:**
- Entry
- Release
- RLTGTK
- Exit

**Called by:**
- RWTAX

144. $	ext{RLWAVE}$

**Calls:**
- Entry
- RLTGTK
- Release
- Exit

**Called by:**
- RLRAI

145. $	ext{ROUTER}$

**Calls:**
- Tapex
- Outci.
- Outcr.

**Called by:**
- ENTRY
- PACK
- FINOBLK

146. $	ext{SCHEDUL}$

**Calls:**
- Entry
- Unpack
- Random
- Render
- Hexadd
- Gethex
- Hexdist
- Acfadd
- Exit

**Called by:**
- THTRPLY

147. $	ext{SCHTAB}$

**Calls:**
- Entry
- Exit

**Called by:**
- GETHEX

148. $	ext{SECOND}$

**Called by:**
- EXIT
- ENTRY
- MLTPNT
- CONTROL

149. $	ext{SELECT}$

**Calls:**
- Entry
- GCTDER
- Message
- RITEI
- MALT
- Feldel
- ASSIGN
- ATTACK
- COMM

**Called by:**
- CONTROL
150. SEMANT

CALLS:
ENTRY
GOTOEP
CODE01
GIMME
PACK
RITE1
RITEP
LPLOT
CODE03
GETHEX
CODE05
DMSCDC
TLL2MX
INPC1.
INPCR.
TAPE5#
OUTCI.
OUTCR.
ADDBLOK
TTIME
RITEN
FINDBLK
CODE18
XT01.
SRCHPL
EXIT

CALLED BY:
L9KPR5

PAGE 28

151. SHUFFLE

CALLS:
ENTRY
EXIT

CALLED BY:
T42MX

152. SNAP

CALLS:
ENTRY
EXIT

CALLED BY:
117
       DELADD
       CONTROl

153. SRCHPL

CALLS:
ENTRY
CREATE
EXIT

CALLED BY:
SEMANT
       C03E05
       C03E03
<table>
<thead>
<tr>
<th>Page 29</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>154. TG7G04E</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>155. TG7LIST</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>156. TMT2PS</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>157. TMTAP4N</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>158. TH2XY</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>159. TH2MX</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>160. TLL2MX</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>161. TLL2XY</strong></td>
</tr>
</tbody>
</table>

347
162. TOWER

163. TRACE

164. TRPRMT

165. TRPRNT

166. TRPRRT

167. TTIME

168. TV2MH

169. TV2MKL

170. TV2L:

171. UNPIRE

172. UNPACK

SIN.  
TAN.  
CALLED BY:

SELECT

CALLED BY:

MALT
PACK
HECHZ
PSDUMP
DELA00

CALLS:
TRPRRT
TRPRNT
TRPRNT

CALLS:
TAPES#
OUTCL.

CALLS:
ENTRYP
EXITP

CALLS:
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP
ENTRYP

CALLS:
MALT
PACK
HECHZ
PSDUMP
DELA00

CALLS:
SELECT

CALLS:
TRACE

CALLS:
TRACE

CALLS:
TRACE

CALLS:
TRACE

CALLS:
SEMAST

CALLS:
TLL2MH

CALLS:
RENOEVU
CRABOON

CALLS:
ASIN.
ATAN.

CALLS:
SCHEDUL
OUTPTRS
KOMPARE
CODEE18

348
173. UNSNAQ

CALLS:
ENTRYP
EXITP

CALLED BY:
CONTROL

174. UOLLOAD

CALLS:
ENTRYP
GIMME
400BLOK
EXITP

CALLED BY:
CODE18
CODE05

175. WIREDUT

176. XSHIFT
<table>
<thead>
<tr>
<th>Function</th>
<th>Called By</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALOG</td>
<td>INITACQ</td>
</tr>
<tr>
<td>ASIN</td>
<td>TXY2LL</td>
</tr>
<tr>
<td>ATAN2</td>
<td>JGESUIT, FLTGEOM</td>
</tr>
<tr>
<td>ATAN</td>
<td>TXY2LL</td>
</tr>
<tr>
<td>COS</td>
<td>TXY2ML, CORBOUN</td>
</tr>
<tr>
<td>DECOD1</td>
<td>DBGREAD</td>
</tr>
<tr>
<td>ENDPL1</td>
<td>MALT</td>
</tr>
<tr>
<td>END</td>
<td>MADEM</td>
</tr>
<tr>
<td>30TOER</td>
<td>LEXAN, SEMANT, SELECT, ADASS, MADEM</td>
</tr>
<tr>
<td>INRB1</td>
<td>FETCH</td>
</tr>
<tr>
<td>INPCI</td>
<td>DBGREAD, CARD, SEMANT, OTHRDAOT</td>
</tr>
<tr>
<td>INPCR</td>
<td>SEMANT</td>
</tr>
<tr>
<td>INDRI</td>
<td>OTHRDAOT, MADEM</td>
</tr>
<tr>
<td>IT0J</td>
<td>EXTSCN</td>
</tr>
</tbody>
</table>
15. OUTBI.

16. OUTCI.

CALLED BY:

HOLD

CALLED BY:

CLIST2
ISDUMP
HALT
ADUMP
DBREAD
ICHECK
DISPPHY
DISPPRO
DISPPAY
DISPPAF
DISPPHF
DISPFLT
DISPDBB
DISPQOD
DISPADOS
DISPACR
DISPACL
DISPACO
DISPABQ
DISPADAT
EXIT
ENTRYP
ENTSTAT
ROUTER
REER
RECON
NXTSYM
APCELZ
LACELL
APCEL1
RDCELL
RACD
CHRGEN
ERROR
ANDCHR
LOOKUP
LXPRM
LXAN
TRPRNT
SEWANT
SELECT
RITER
RITEP
RITEL
PLANOUT
17. OUTCR.

CALLED BY:
- CLIST2
- ISDUMP
- ADUMP
- ROUTER
- RECER
- SEWANT
- SELECT
- FS DUMP
- CLIST

18. 21NTRY.

CALLED BY:
- WDEM

19. RANDOM.

CALLED BY:
- SCHEDULE
- CANDOTG

20. REWIND.

CALLED BY:
- HOLD

21. SIM.

CALLED BY:
- TX2MXL
- LL2XY
- LL2MX
- CORBON

22. STOP.

CALLED BY:
- HALT
- JTJ

23. TAN.

CALLED BY:
- LL2XY
- LL2MX

24. TAPE64

CALLED BY:
- CLIST2
- EXITP
- ENTRYP
- ENTRSTAT
- ROUTER
- RECER
- RECCON
- NXTSYN
PAGE 38

00109 3. ENTRYP (SEE LINE 00011)
00109 3. EXITP (SEE LINE 00033)
00110 2. OTHROAT
00111 3. ENTRYP (SEE LINE 00011)
00112 3. EXITP (SEE LINE 00011)
00113 3. ENTRYP (SEE LINE 00011)
00114 4. HALT (SEE LINE 00009)
00115 4. MESSGE (SEE LINE 00012)
00116 4. RITEI (SEE LINE 00015)
00117 4. RITEP (SEE LINE 00066)
00118 4. EXITP (SEE LINE 00033)
00119 3. INPCT.
00120 3. TAPE64
00121 3. OUTCI.
00122 3. RITEP (SEE LINE 00066)
00123 3. ADDBLK
00124 4. ENTRYP (SEE LINE 00011)
00125 4. EXITP (SEE LINE 00033)
00126 3. INPFI.
00127 3. FINDBLK
00128 4. ENTRYP (SEE LINE 00011)
00129 4. ROUTER (SEE LINE 00024)
00130 4. MESSGE (SEE LINE 00012)
00131 4. RITEI (SEE LINE 00015)
00132 4. CLIST (SEE LINE 00058)
00133 4. EXITP (SEE LINE 00033)
00134 3. HALT (SEE LINE 00009)
00135 3. EXITP (SEE LINE 00033)
00136 2. DISPDAT
00137 3. OUTCI.
00138 3. OUTPOS
00139 4. OUTCI.
00140 3. DISPFDB
00141 3. OUTCI.
00142 4. DISPRAW
00143 5. OUTCI.
00144 5. DISPFIT
00145 6. OUTCI.
00146 6. DISPLAY
00147 7. OUTCI.
00148 7. DISPLAY
00149 8. OUTCI.
00150 6. DISPLAY
00151 7. OUTCI.
00152 6. DISPLAY
00153 7. OUTCI.
00154 6. DISPLAY
00155 7. OUTCI.
00156 3. DISPFIT (SEE LINE 00144)
00157 3. DISPFACD (SEE LINE 00152)
00158 3. DISPFIT
00159 4. OUTCI.
00160 4. DISPLAY (SEE LINE 00148)
00161 3. DISPLAY (SEE LINE 00154)
00162 3. DISPLAY (SEE LINE 00150)
3. DISPAGY
4. OUTCI.
3. DISPCRA
4. OUTCI.
4. DISPCAL
5. OUTCI.
2. MALT
(SEE LINE 00009)

2. INIT

3. ENTRY.
(SEE LINE 00011)

3. SND.

3. GIMME
(SEE LINE 00112)

3. RITEP
(SEE LINE 00066)

3. SNAP

2. EXIT.
(SEE LINE 00033)

3. EXITP.
(SEE LINE 00066)

3. SMAO

4. EXITP.
(SEE LINE 00033)

4. GOTOEq.

4. C300E01

S.

314E

2. EXIT.

3. SE4&T.

4. ENTRYP
(SEE LINE 00011)

5. EXIT.
(SEE LINE 00033)

2. EXIT.

5. 314E
(SEE LINE 00112)

4. ZACK

3. ENTRYP
(SEE LINE 00011)

5. LVL.

5. 94GE
(SEE LINE 00059)

6. 4ESAGE
(SEE LINE 00012)

6. RITEP
(SEE LINE 00066)

7. LUT.

5. EXIT.
(SEE LINE 00033)

5. EXIT.
(SEE LINE 00033)

5. CREATE
(SEE LINE 00208)

6. ENTRYP
(SEE LINE 00011)

7. EXIT.
(SEE LINE 00033)

6. EXIT.
(SEE LINE 00033)

6. EXIT.
(SEE LINE 00033)
00383  3. ROCELL
00384  4. ISHIFT
00385  4. TAPE6#
00386  4. OUTCI.
00387  3. NXTSYN
00388  4. LEXAN
00389  5. CHRGEN
00390  6. CARD
00391  7. INPUT.
00392  7. EOF
00393  7. TAPE6#
00394  7. OUTCI.
00395  6. TAPE6#
00396  6. OUTCI.
00397  5. ISHIFT
00398  5. GOTOER.
00399  5. ADDCHR
00400  6. ISHIFT
00401  6. TAPE6#
00402  6. OUTCI.
00403  5. LOOKUP
00404  6. ISHIFT
00405  6. TAPE6#
00406  6. OUTCI.
00407  5. EXTSYN
00408  6. CHRGEN
00409  6. ITOJ.
00410  6. TAPE6#
00411  5. OUTCI.
00412  4. TAPE6#
00413  4. OUTCI.
00414  3. ERROR
00415  4. TAPE6#
00416  4. OUTCI.
00417  3. ISHIFT
00418  3. ARCCEL
00419  4. ISHIFT
00420  4. TAPE6#
00421  4. OUTCI.
00422  3. ARCCEL2
00423  4. ISHIFT
00424  4. TAPE6#
00425  4. OUTCI.
00426  3. LACELL
00427  4. ISHIFT
00428  4. TAPE6#
00429  4. OUTCI.
00430  2. RELIST
00431  3. ENTRYP
00432  3. RELEASE
00433  3. EXIT
00434  2. DELAED
00435  2. CONTROL
00436  3. ENTRYP
00437  3. LTAPE
00603 9. HECMTZ (SEE LINE 00496)
00604 9. TINHPS (SEE LINE 00467)
00605 9. ATAYZC (SEE LINE 00033)
00606 8. EXITX (SEE LINE 00303)
00607 8. HECMTST (SEE LINE 00230)
00608 7. ENTRYP (SEE LINE 00033)
00609 7. DELEADO (SEE LINE 00230)
00610 6. PLANOUT (SEE LINE 00011)
00611 6. EXITP (SEE LINE 00033)
00612 6. DELADD (SEE LINE 00230)
00613 6. EXITP (SEE LINE 00033)
00614 5. EXITP (SEE LINE 00033)
00615 5. EXITP (SEE LINE 00033)
00616 5. EXITP (SEE LINE 00033)
00617 4. EXITP (SEE LINE 00033)
00618 4. EXITP (SEE LINE 00033)
00619 4. EXITP (SEE LINE 00033)
00620 4. EXITP (SEE LINE 00033)
00621 4. EXITP (SEE LINE 00033)
00622 4. EXITP (SEE LINE 00033)
00623 4. EXITP (SEE LINE 00033)
00624 4. EXITP (SEE LINE 00033)
00625 4. EXITP (SEE LINE 00033)
00626 4. EXITP (SEE LINE 00033)
00627 4. EXITP (SEE LINE 00033)
00628 4. EXITP (SEE LINE 00033)
00629 4. EXITP (SEE LINE 00033)
00630 4. EXITP (SEE LINE 00033)
00631 4. EXITP (SEE LINE 00033)
00632 4. EXITP (SEE LINE 00033)
00633 4. EXITP (SEE LINE 00033)
00634 4. EXITP (SEE LINE 00033)
00635 4. EXITP (SEE LINE 00033)
00636 4. EXITP (SEE LINE 00033)
00637 4. EXITP (SEE LINE 00033)
00638 4. EXITP (SEE LINE 00033)
00639 4. EXITP (SEE LINE 00033)
00640 4. EXITP (SEE LINE 00033)
00641 4. EXITP (SEE LINE 00033)
00642 4. EXITP (SEE LINE 00033)
00643 4. EXITP (SEE LINE 00033)
00644 4. EXITP (SEE LINE 00033)
00645 5. EXITP (SEE LINE 00033)
00646 4. POVER (SEE LINE 00175)
00647 4. TOWER (SEE LINE 00175)
00648 4. UNAPR (SEE LINE 00175)
00649 4. TAPE60 (SEE LINE 00175)
00650 4. OUTCI (SEE LINE 00175)
00651 4. OUTCR (SEE LINE 00175)
00652 3. SNAP (SEE LINE 00033)
00653 3. SNAP (SEE LINE 00033)
00654 3. UNSNAP (SEE LINE 00033)
00655 4. ENTRYP (SEE LINE 00033)
00656 4. EXITP (SEE LINE 00033)
00657 3. RELEASE (SEE LINE 00033)

365
3. SECOND
4. ENTRY (SEE LINE 00011)
4. SECOND (SEE LINE 00009)
4. EXITO (SEE LINE 00033)
3. EXITO (SEE LINE 00033)
2. EXITO (SEE LINE 00033)
2. END.
# 2. Main Processor

**LIST OF SUBROUTINES - PAGE 1**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ABSEE</td>
</tr>
<tr>
<td>2.</td>
<td>ABYSCOR</td>
</tr>
<tr>
<td>3.</td>
<td>ABZCSC</td>
</tr>
<tr>
<td>4.</td>
<td>ACCEPT</td>
</tr>
<tr>
<td>5.</td>
<td>ACFRAG</td>
</tr>
<tr>
<td>6.</td>
<td>ADDBLOK</td>
</tr>
<tr>
<td>7.</td>
<td>ADUMP</td>
</tr>
<tr>
<td>8.</td>
<td>AIRITWNK</td>
</tr>
<tr>
<td>9.</td>
<td>ALLOBAT</td>
</tr>
<tr>
<td>10.</td>
<td>ALLOFU</td>
</tr>
<tr>
<td>11.</td>
<td>ALLOBAT</td>
</tr>
<tr>
<td>12.</td>
<td>AMMOCMK</td>
</tr>
<tr>
<td>13.</td>
<td>ASSSIN</td>
</tr>
<tr>
<td>14.</td>
<td>ATKASES</td>
</tr>
<tr>
<td>15.</td>
<td>ATTACK</td>
</tr>
<tr>
<td>16.</td>
<td>AUTOPRI</td>
</tr>
<tr>
<td>17.</td>
<td>AVAILBL</td>
</tr>
<tr>
<td>18.</td>
<td>AZILIN</td>
</tr>
<tr>
<td>19.</td>
<td>BADMOVE</td>
</tr>
<tr>
<td>20.</td>
<td>BATCEAS</td>
</tr>
<tr>
<td>21.</td>
<td>BATTCOV</td>
</tr>
<tr>
<td>22.</td>
<td>BATTOUT</td>
</tr>
<tr>
<td>23.</td>
<td>BLKDAT</td>
</tr>
<tr>
<td>24.</td>
<td>BNCHDPR</td>
</tr>
<tr>
<td>25.</td>
<td>BNCHSHD</td>
</tr>
<tr>
<td>26.</td>
<td>BNCHWLS</td>
</tr>
<tr>
<td>27.</td>
<td>BNCHWTC</td>
</tr>
<tr>
<td>28.</td>
<td>BNLLALLE</td>
</tr>
<tr>
<td>29.</td>
<td>BNNOTRD</td>
</tr>
<tr>
<td>30.</td>
<td>BNNOTRK</td>
</tr>
<tr>
<td>31.</td>
<td>BNPROBB</td>
</tr>
<tr>
<td>32.</td>
<td>BNPROBD</td>
</tr>
<tr>
<td>33.</td>
<td>BNPRODA</td>
</tr>
<tr>
<td>34.</td>
<td>BNPROEP</td>
</tr>
<tr>
<td>35.</td>
<td>BNPROFA</td>
</tr>
<tr>
<td>36.</td>
<td>BNPROFD</td>
</tr>
<tr>
<td>37.</td>
<td>BNPROSS</td>
</tr>
<tr>
<td>38.</td>
<td>BNRCOV</td>
</tr>
<tr>
<td>39.</td>
<td>BOCTINK</td>
</tr>
<tr>
<td>40.</td>
<td>BTMASC</td>
</tr>
<tr>
<td>41.</td>
<td>BTNZCRC</td>
</tr>
<tr>
<td>42.</td>
<td>BTRYPTNK</td>
</tr>
<tr>
<td>43.</td>
<td>BTLICOV</td>
</tr>
<tr>
<td>44.</td>
<td>BYMCPDR</td>
</tr>
<tr>
<td>45.</td>
<td>BYMCHMO</td>
</tr>
<tr>
<td>46.</td>
<td>BYCOWLS</td>
</tr>
<tr>
<td>47.</td>
<td>BYCOWTC</td>
</tr>
<tr>
<td>48.</td>
<td>BYENOPS</td>
</tr>
<tr>
<td>49.</td>
<td>BYMEDUP</td>
</tr>
<tr>
<td>50.</td>
<td>BYNOTRD</td>
</tr>
<tr>
<td>51.</td>
<td>BYNOTRK</td>
</tr>
<tr>
<td>52.</td>
<td>BYPASUP</td>
</tr>
</tbody>
</table>
108. DOGTWNK
109. DROPALK
110. DROPP0S
111. DROPPS2
112. ENGAGE
113. ENTRYP
114. ENTRSTAT
115. EOF
116. EXITP
117. FELDEL
118. FETCH
119. FILEUP
120. FINDALK
121. FINDFLT
122. FINDIT
123. FIRECHK
124. FLITE
125. FLTGEOM
126. FLTYPE
127. FLY
128. FLYSEE
129. FORMTST
130. FS0UP
131. FSINIT
132. FUELCHK
133. GETMEX
134. GETPTRS
135. GIMME
136. GNOLJ0K
137. GOGETEM
138. GOTOAB
139. HALT
140. HANDZPT
141. HEXAND0
142. HEXCHZ
143. HEX0IST
144. HEXLW
145. HEXMVT
146. HEXMOV
147. HEXMUL
148. HISTORY
149. HLTINT
150. HOLD
151. HX0GTS
152. HX0LZ
153. ICHECK
154. IJRMX
155. INIT
156. INITACQ
157. INRANGE
158. INSECT
159. INSERT
160. INTASIN
161. INTFINO
162. INTFLY
163. INT2CRC
164. IPJL
165. ISDUMP
166. ISHIFT
167. ITRAP
168. JGESJIT
169. JFJ
170. JUGGLE
171. KILFLIT
172. KOMPARE
173. LCMLOC
174. LINEX
175. LINPLOT
176. LOADPL
177. LOSRAOR
178. LDRPS
179. LTREE
180. LTRMAG
181. MADE
182. MASKER
183. MESSAGE
184. MESSBILD
185. MAYBOR
186. NEWMOVE
187. NEWPERC
188. NOWUCIT
189. NUMBIND
190. OPTPTH
191. OTTHROT
192. OUTA
193. OUTPYRS
194. PACK
195. PAGE
196. P4TDEC
197. RELADO
198. PERCEPT
199. PLAN
200. PLANOUT
201. RONDER
202. PEPAFU
203. PRIORITY
204. FTPOWER
205. PTRAN
206. PTREE
207. PTRMAG
208. READIL
209. RECCON
210. RED
211. RECOVR
212. REDEREF
213. RELEASE
214. RELIST
215. RELOAD
216. RELOCAT
217. RELSTILL
273. UNLINK
274. UNPACK
275. UNSNAP
276. UNSTAT
277. UNLOAD
278. WIPEDOUT
279. WITHDRAW
280. RPAA
281. XPO
282. XPK
283. XSHIFT
284. XYZMIX
285. TANK
LIST OF FORTRAN LIBRARY ROUTINES - PAGE 7

1. ALOG
2. ASIN
3. ATAN
4. COS
5. DECOD
6. ENDFIL
7. END
8. GOTOER
9. INPB1
10. INPC
11. INPFI
12. ITOJ
13. OUTB1
14. OUTC
15. OUTCQ
16. QINTRY
17. RANDOM
18. REWIND
19. SIN
20. SORT
21. STOP
22. TAN
23. TAPER
24. XTOI
<table>
<thead>
<tr>
<th>Page 8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBROUTINE REFERENCE LIST - MADE</strong></td>
</tr>
</tbody>
</table>
| **1. ABSEE** | **CALLS**: | **UNPACK**
**DELADD**
**RELEASF**
**TAPES**
**OUTC** |
**CALLED BY**: | **PERCEPT** |
| **2. ABVSCOR** | **CALLS**: | **ENTRYP**
**CLOSCOR**
**GIMME**
**ADDBLOK**
**EXITP** |
**CALLED BY**: | **REVISE** |
| **3. A92CRC** | **CALLS**: | **UNPACK**
**RELEASF**
**FINDBLK**
**GIMME**
**ADDBLOK**
**TAPES**
**OUTC**
**DROPBLK** |
**CALLED BY**: | **CRCTHNK** |
| **4. ACCEPT** | **CALLS**: | **UNPACK**
**SEEK**
**GIMME**
**STOP**
**STICK**
**DETECT**
**DELADD**
**PACK**
**TAPES**
**OUTC**
**BYALCOV**
**RELEASE**
**BYMEDUP**
**OUTA** |
**CALLED BY**: | **SCHMOPR**
**SYCMOPR** |
| **5. ACFRAG** | **CALLS**: | **GIMME**
**UNPACK**
**CRFLTH**
**ADDBLOK**
**OPTPTH**
**RELEASE**
**PACK**
**FLTGEOM**
**MEADIST**
**DELADD** |
**CALLED BY**: | **SCHMUP** |
6. ADDBLOK  CALLS:
ENTRYP
EXITO
CALLED BY:
UOLLLOAD
REDEBRP
NOWUCIT
NEWMOVE
IVT2CRC
IVTACQ
GOGET4
FLYSEE
FINDIT
FINDFLT
CRFLTM
CANDST
BADMOVE
ATKASES
APFRAG
AVSCOR
AR2CRC

7. ADUMP  CALLS:
OUTCI,
OUTCR,
CALLED BY:
HALT
RECON

8. AIRTHNNK  CALLS:
HEADCST
MESBILD
DELA00
TGMEX
TAPE6#
OUTCI,
RANDOM
CALLED BY:
TFLYCRC

9. ALLOBAT  CALLS:
PRIORITY
MESBILD
GIMME
PACK
TAPE6#
OUTCI,
OUTA
DELA00
YANK
STICK
TOMOIL
CALLED BY:
SEEKTAC
SEEKENG
B0CTINK

10. ALLOFU  CALLS:
GIMME
STICK
DELA00
TAPE6#
OUTCI,
CALLED BY:
SEEKTFU
BACKUP

11. ALLOPAT  CALLS:
GIMME
CALLED BY:
SEEKTFU
12. AMMOCHK

CALLS:
MESBILD
DELAAD
GOTOER
BYNOTRD
TAPE6#
OUTCI.

CALLED BY:
ENGAGE

13. ASSIGN

CALLS:
GETPTRS
INTASIN

CALLED BY:
SELECT

14. ATKASES

CALLS:
GIMME
DELAAD
DESTROY
TAPE6#
OUTCI.
ADDDBL0K
THM2PS
ATANZ.

CALLED BY:
CFLYCRC

15. ATTACK

CALLS:
GETPTRS
UMSTAT
SMRKILI
FINDBLK
XPK
RANDOM.
NUMBLNO
DROPBLK
HISTORY
DELAAD
STAPAK

CALLED BY:
SELECT

16. AUTOPQI

CALLED BY:
SAMATON
PREPAFY
BYCONNO

17. AVAILBL

CALLS:
RELAAD
RELEASE

CALLED BY:
TMTPRLY

18. AZILIM

CALLS:
SIN.
COS.
INSECT
SORT.

CALLED BY:
INRANGE
26. BNCONLS

Calls:
DROPPROS
DILOUT
MNSBILD
DELADD

Called By:
SOIGEST

27. BNCNCTC

Calls:
CHKCOV
DROPPROS
BNLALLE
SETASSN
SEEKTAC
GIMME
DLYACT
CHKLST
BNCONMD
BNRECVR
TAPE6#
OUTCI.
YANK
STICK
TOAOIL
DILOUT

Called By:
SOIGEST
BYPONOA

28. BNLALLE

Calls:
MNSBILD
DELADD
YANK
HANDZTP
DILOUT
STICK

Called By:
TRYSHOT
SOIGEST
TRKCHCK
SMPRCH
BYCONC
BNCONC
BYWTRK
BYENDPS
BYCONLS
BYTYNk
BCTINK
BNPONE
BNPONOA
BYWTRK
BATTOUT

29. BNNOTDO

Calls:
TAPE6#

 Called By:
BYUPDAT
OUTC1.
RELEASE
DILOUT

30. BNXWTRK
CALLS:
CHKCOV
BNLALLE
SETASSN
SEEKTAC
GIMME
DLYACT

31. BNPONAB
CALLS:
FILEUP
UNPACK
INRANGE
GIMME
DELOAD
SETASSN
SEEKENG
PRIORITY
RELOCATE
STICK
GOTOER.
DLYACT
TAPE64
OUTC1.

32. BNPONAD
CALLS:
TAPE64
OUTC1.
BNNOTRO
BATTOUT
COVARLY
RELEASE

33. BNPONDA
CALLS:
YANK
UNPACK
GOTOER.
SETASSN
SEEKTAC
CHKCOV
BNLALLE
GIMME
DLYACT
BNCOVTC
SEEKENG
RELEASE

34. BNPONED
CALLS:
GOTOER,
SKSBRTRK
DROPOSS
BNLALLE
UNPACK
CHKLAST
SEEKTAC
GIMME
PACK
OLYACT
SEEKENG
MESISBILD
DELAOD
YANK
STICK
READIL

35. BNPONFA
CALLS:
SEEKP
TAPE#
OUTCI.
BNRECOV

36. BNPONFO
CALLS:
SEEKP
TAPE#
OUTCI.
RELEASE
YANK
DROPOUT
MESISBILD
DELAOD

37. BNPONSS
CALLS:
UNPACK
GOTOER.
BNPONFB
SEEKENG
GIMME
OLYACT
BYUPDAT

38. BNRECOV
CALLS:
UNPACK
WITHDRAW
PACK

39. BOCTINK
CALLS:
BNPONSE
BNPONEP
BNCMOPR
BNPONFA
BNPONFO
RELEASE
SDIGEST
BNPONDA
TAPE#
OUTCI.

PAGE 14
40. BTNASIN

CALLS:
UNPACK
FINDBLK
PACK
HEXDIST
RELEASE
DROPBLK
MBSBILD
DELADD
TAPE#64
OUTCI.

41. BTN2CRC

CALLS:
UNPACK
RELEASE
TAPE#64
OUTCI.
CRCLOSE
CRCHEL
CRCITK
CRCTRK

42. BTYTMK

CALLS:
BYCMDPR
SDEGEST
SEEKP
BYPONTM
BYPOWER
TAPE#64
OUTCI.
YANK
BATGEAS
BNLALLE
MBSBILD
DELADD
BYPONFD
SAMATON
BYPONAL
BYPONAR

43. BYALCOV

CALLS:
UNPACK
PACK
GIME
SLL
CANCEALO
SEEKTFU

44. BTYTMK

CALLS:
BYCMDPR
SDEGEST
SEEKP
BYPONTM
BYPOWER
TAPE#64
OUTCI.
YANK
BATGEAS
BNLALLE
MBSBILD
DELADD
BYPONFD
SAMATON
BYPONAL
BYPONAR

45. BYALCOV

CALLS:
UNPACK
PACK
GIME
SLL
CANCEALO
SEEKTFU

46. BTYTMK

CALLS:
BYCMDPR
SDEGEST
SEEKP
BYPONTM
BYPOWER
TAPE#64
OUTCI.
YANK
BATGEAS
BNLALLE
MBSBILD
DELADD
BYPONFD
SAMATON
BYPONAL
BYPONAR

47. BYALCOV

CALLS:
UNPACK
PACK
GIME
SLL
CANCEALO
SEEKTFU

48. BTYTMK

CALLS:
BYCMDPR
SDEGEST
SEEKP
BYPONTM
BYPOWER
TAPE#64
OUTCI.
YANK
BATGEAS
BNLALLE
MBSBILD
DELADD
BYPONFD
SAMATON
BYPONAL
BYPONAR

49. BYALCOV

CALLS:
UNPACK
PACK
GIME
SLL
CANCEALO
SEEKTFU

50. BTYTMK

CALLS:
BYCMDPR
SDEGEST
SEEKP
BYPONTM
BYPOWER
TAPE#64
OUTCI.
YANK
BATGEAS
BNLALLE
MBSBILD
DELADD
BYPONFD
SAMATON
BYPONAL
BYPONAR

51. BYALCOV

CALLS:
UNPACK
PACK
GIME
SLL
CANCEALO
SEEKTFU

52. BTYTMK

CALLS:
BYCMDPR
SDEGEST
SEEKP
BYPONTM
BYPOWER
TAPE#64
OUTCI.
YANK
BATGEAS
BNLALLE
MBSBILD
DELADD
BYPONFD
SAMATON
BYPONAL
BYPONAR

53. BYALCOV

CALLS:
UNPACK
PACK
GIME
SLL
CANCEALO
SEEKTFU
RELEASE
PATDEC
BATTCOV
DLYACT
TAPE6#
OUTCI.

44. BYCHDR CALLS: ACCEPT SEEK TAPE6# OUTCI. BATCEAS DILOUT BYALCOV BYMEOP RELEASE
CALLED BY: BTRYTNK

45. BYCINV CALLS: TAPE6# OUTCI. DELADD AUTOPRI STICK RELOCAT BATTCOV GIMME DLYACT
CALLED BY: BYCINV

46. BYCINV CALLS: BATCEAS BNLAILE MESBILD DELADD DILOUT
CALLED BY: SDIGEST

47. BYCIV CALLS: MESBILD DELADD INRANGE BATCEAS BNLAILE PREPAFU BATTCOV GIMME DLYACT BYCINV MO UNPACK BYALCOV STICK RELOCAT YANK TOADIL
CALLED BY: SDIGEST BYPCOMP

48. BYENDS CALLS:
CALLED BY: 

382
49. BYMEDUP

CALLS:
SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:

49. BYMEDUP

SEEKP
UNPACK
TANK
RELEASE
GETPTRS
CRCLOSS
BNLALLE

CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
CALLED BY:
<table>
<thead>
<tr>
<th>Page 19</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>54. BYPONFO</strong></td>
</tr>
<tr>
<td><strong>55. BYPONRL</strong></td>
</tr>
<tr>
<td><strong>56. BYPONRS</strong></td>
</tr>
<tr>
<td><strong>57. BYPONTH</strong></td>
</tr>
<tr>
<td><strong>58. BYTKCCHK</strong></td>
</tr>
<tr>
<td><strong>59. BYTKDAT</strong></td>
</tr>
</tbody>
</table>
60. CANCALO CALLS:

61. CANDATG CALLS:

62. CENTER CALLS:

63. CFLYCC CALLS:

64. CHKDOC CALLS:

65. CHKLAST CALLS:

CALLS: CALLED BY:

YANK RELEASE TAPE# OUTCL. REAOL

ENTRYP FINDBLK CLOSOC JGESUIT FORTGT GIMME ADDBLOK TOTBONE PTREE RANDOM PELADO EXITP

ENTRYP EXITP

CRCSEE UNSTAT FLYSEE ATKASE CRC2INT RONDSEF QNLOOK STATPAX

UNPACK INRANGF GIMME TAPE60 OUTCI. OUTA DELADD

DROPPED

CALLED BY:

PTPONER BYALCOV PATDEC BYPONER BATCEAS

CALLED BY:

CALLED BY:

CALLED BY:

CALLED BY:

CALLED BY:

CALLED BY:

CALLED BY:

CALLED BY:
<table>
<thead>
<tr>
<th>CALLS</th>
<th>PAGE</th>
<th>LNPLOT</th>
<th>MESSAGE</th>
<th>RITEP</th>
<th>RITEI</th>
<th>RITE</th>
<th>TAPE6#</th>
<th>OUTCI</th>
<th>OUTCR</th>
<th>CLIST2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLED BY</td>
<td>HALT</td>
<td>UNSTAT</td>
<td>TGTHX</td>
<td>TRKCHEK</td>
<td>SAPRCN</td>
<td>MAYBOR</td>
<td>HEMOVE</td>
<td>HEMCHZ</td>
<td>FSQUMP</td>
<td>FLTTYPE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CALLS</th>
<th>PAGE</th>
<th>LNPLOT</th>
<th>MESSAGE</th>
<th>RITEP</th>
<th>RITEI</th>
<th>RITE</th>
<th>TAPE6#</th>
<th>OUTCI</th>
<th>OUTCR</th>
<th>RITEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLED BY</td>
<td>CLIST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CALLS</th>
<th>ENTRYP</th>
<th>HEXDIST</th>
<th>EXITP</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLED BY</td>
<td>CANDOTST</td>
<td>ABVSCOR</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CALLS</th>
<th>YANK</th>
<th>RELEASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLED BY</td>
<td>DILOUT</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CALLS</th>
<th>UNPACK</th>
<th>GOTOER</th>
<th>DELADD</th>
<th>TAPE6#</th>
<th>OUTCI</th>
<th>MESBILD</th>
<th>FLTTYPE</th>
<th>THNP2PS</th>
<th>ATANZ</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLED BY</td>
<td>FLITE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CALLS</th>
<th>UNPACK</th>
<th>DELADD</th>
<th>RELEASE</th>
<th>PACK</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLED BY</td>
<td>SELECT</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

386
<table>
<thead>
<tr>
<th>Page 21</th>
</tr>
</thead>
<tbody>
<tr>
<td>72. CONTROL CALLS:</td>
</tr>
<tr>
<td>ENTRYP</td>
</tr>
<tr>
<td>LTREE</td>
</tr>
<tr>
<td>SELECT</td>
</tr>
<tr>
<td>SNAP</td>
</tr>
<tr>
<td>UNSNAP</td>
</tr>
<tr>
<td>RELEASE</td>
</tr>
<tr>
<td>SECOND</td>
</tr>
<tr>
<td>HLTPNT</td>
</tr>
<tr>
<td>EXITP</td>
</tr>
<tr>
<td>73. CORBON CALLS:</td>
</tr>
<tr>
<td>ENTRYP</td>
</tr>
<tr>
<td>GIMME</td>
</tr>
<tr>
<td>TM2PPS</td>
</tr>
<tr>
<td>TXY2ML</td>
</tr>
<tr>
<td>HEXDIST</td>
</tr>
<tr>
<td>SIN.</td>
</tr>
<tr>
<td>COS.</td>
</tr>
<tr>
<td>LEXNX</td>
</tr>
<tr>
<td>OPTOTH</td>
</tr>
<tr>
<td>EXITP</td>
</tr>
<tr>
<td>74. COVARPLT CALLS:</td>
</tr>
<tr>
<td>SEEKAT</td>
</tr>
<tr>
<td>GIMME</td>
</tr>
<tr>
<td>DLYACT</td>
</tr>
<tr>
<td>RELEASE</td>
</tr>
<tr>
<td>75. CRDCIES CALLS:</td>
</tr>
<tr>
<td>UNPACK</td>
</tr>
<tr>
<td>RELEASE</td>
</tr>
<tr>
<td>DELADD</td>
</tr>
<tr>
<td>FINDBLK</td>
</tr>
<tr>
<td>TRACE</td>
</tr>
<tr>
<td>MESSAGE</td>
</tr>
<tr>
<td>RITEI</td>
</tr>
<tr>
<td>CLIST</td>
</tr>
<tr>
<td>DROPBLK</td>
</tr>
<tr>
<td>76. CRCEYNT CALLS:</td>
</tr>
<tr>
<td>GIMME</td>
</tr>
<tr>
<td>TAPE6#</td>
</tr>
<tr>
<td>OUTCI.</td>
</tr>
<tr>
<td>DELADD</td>
</tr>
<tr>
<td>RELEASE</td>
</tr>
<tr>
<td>FINDBLK</td>
</tr>
<tr>
<td>DETECT</td>
</tr>
<tr>
<td>UNPACK</td>
</tr>
<tr>
<td>HEXDIST</td>
</tr>
<tr>
<td>LOSRADDOR</td>
</tr>
<tr>
<td>COS.</td>
</tr>
<tr>
<td>RANDOM.</td>
</tr>
<tr>
<td>CRCTRAK</td>
</tr>
<tr>
<td>Page 22</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td><strong>77. CRCKIL</strong></td>
</tr>
<tr>
<td>FINOBLK</td>
</tr>
<tr>
<td><strong>78. CRCLOSS</strong></td>
</tr>
<tr>
<td>FINOBLK</td>
</tr>
<tr>
<td><strong>79. CRCSEE</strong></td>
</tr>
<tr>
<td>RELEASE</td>
</tr>
<tr>
<td><strong>80. CRCTHNK</strong></td>
</tr>
<tr>
<td>UNPACK</td>
</tr>
<tr>
<td><strong>81. CRCTR4K</strong></td>
</tr>
<tr>
<td>FINOBLK</td>
</tr>
<tr>
<td><strong>82. CRC2INT</strong></td>
</tr>
<tr>
<td>UNPACK</td>
</tr>
</tbody>
</table>
88. DESTROY

Calls: TAPE# OUTCI. GETPTRS TERMINO KILFLIT GIMME DELADD SAMWYPE CRODIES UNLINK RELEASE UNSTAT

Called by: UNPIRE TOWER SHRKILL REDEBRF FLY DOSFITE ATKASES

89. DETECT

Calls: HEXDIST UNPACK THH2PS ATAN2 LOSRADR COS RANMOD TAPE# OUTCI.

Called by: SAMPRCW INTZCRC FLYSEE CRECVNT ACCEPT

90. DGTSRX

Calls: ITOJ.

91. DILOUT

Calls: CNACTTK RELEASE UNPACK YANK TOADIL

Called by: SAMATON PTPONER HANDOPT B4LALLE B4NCONTC BYDONRD BYPONER BYNORD BYCONLS BYCMOPR B0CTING B4PONFD B4NORD B4NCONLS B4CMOPR

92. DISPAGB

Calls: OUTCI.

Called by: DISPDAT

93. DISPACO

Calls: OUTCI.

Called by: DISPFILT DISPDAT

94. DISPACL

Calls: OUTCI.

Called by: DISPACR
<table>
<thead>
<tr>
<th>Page 26</th>
</tr>
</thead>
<tbody>
<tr>
<td>95. DISPACR</td>
</tr>
<tr>
<td>96. DISPAOS</td>
</tr>
<tr>
<td>97. DISPAOQ</td>
</tr>
<tr>
<td>98. DISPDAQ</td>
</tr>
<tr>
<td>99. DISPFLB</td>
</tr>
<tr>
<td>100. DISPFLT</td>
</tr>
<tr>
<td>101. DISPFMB</td>
</tr>
<tr>
<td>102. DISPPAF</td>
</tr>
<tr>
<td>103. DISPPAY</td>
</tr>
<tr>
<td>104. DISPPRO</td>
</tr>
<tr>
<td>105. DISPPYR</td>
</tr>
</tbody>
</table>
PAGE 27

106. DLYACT

CALLS:
UNPACK
PACK
STICK

CALLED BY:
WINDRAW
SOIGEST
BYALCOV
BYCONTG
BYCONTC
DROPPOS
DROPPSS
COVAPLY
BYUPDAT
BYWTRX
BYCONMD
BYWONSS
BYWONEP
BYWOND5
BYWON69
BYWON72
BYWON7X
BYWON80
BYWON8D

107. DOGFITE

CALLS:
GETPTRS
UNSTAT
TAPEDT
OUTCI.
HISTORY
FINDBLK
DROPBLK
DELADD
XPAA
RANDOM.
BINME
DESTROY

CALLED BY:
SELECT

108. DOGTHWK

CALLS:
WESBILD
DELADD
TAPEDT
OUTCI.
GOTOAB
FUELCHK

CALLED BY:
TFLYCRC

109. DROPBLK

CALLS:
ENTRYP
RELEASE
EXITP

CALLED BY:
SAMWYPE
NUKBLNO
WUWCT
KLLFLIT
INTCRUC
HEXMOVE
FLTRYPE
DOGFITE
CRCTRX
CRCLOSE
CRCKL
CRCDKI
CRCDKIES
<table>
<thead>
<tr>
<th>Page 28</th>
</tr>
</thead>
<tbody>
<tr>
<td>110. DROPPS1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

| 110. DROPPS2     | CALLS1 | CALLED BY1: |
|                  | YANK   | SEEKENG     |
|                  | RELEASE | WESBILO    |
|                  | DELADD | SEEKENG    |
|                  | SEEKENG| GIMME      |
|                  | DLYACT |            |

| 112. ENGAGE      | CALLS1 | CALLED BY1: |
|                  | GETPTRS| SELECT      |
|                  | TRYSHOT|            |
|                  | TAPE6# |            |
|                  | OUTCI. |            |
|                  | GIMME  |            |
|                  | DELADD |            |
|                  | UNPACK |            |
|                  | PACK   |            |
|                  | AMMOCHK|            |
|                  | RELEASE|            |
|                  | HISTORY|            |

| 113. ENTRY         | CALLS1 | CALLED BY1: |
|                   | MESSAGE| LT3MRB      |
|                   | RITEI  | U4RACK      |
|                   | RECR   | U4LOAD      |
|                   | TAPE6# | U4SNAP      |
|                   | OUTCI. | TX2MHL      |
|                   | ROUTER | TX2MX       |
|                   | TRAP   | TX2XY       |
|                   | SECOND | T4TRPL4     |
|                   |        | T4W2PS      |
|                   |        | T4T4ONE     |
|                   |        | SNAP        |
|                   |        | SELECT      |
|                   |        | SCHTAB      |
|                   |        | RLWAVE      |
|                   |        | ALTSTYP     |
OUTCI.

115. EOF

116. EXITP

CALLS:

SECOND
MESSAGE
RITE!
RECEI
TAPE&
OUTCI.
ITRAP
ICHECK

called by:

DBREAD

called by:

LTMRG
UNPACK
UNLOAD
UNSNAP
TXY2MXL
TXY2MX
TMXZXY
TMXPILY
TMX2PS
T3T3GONE
SNAP
SELECT
SCHTAB
RLWAVE
RLTGSTYP
RLTGSTAK
RLRAID
RLFMAKT
RLCORD
RLABDB
REVISE
RELIST
RELEASE
PITREE
PLANOUT
PLAN
PELADD
PACK
OUT
NOWUCIT
LTREE
LOADPL
LINES
KOMPARE
JUGGLE
JTJ
JGETSUIT
INITACQ
IJ2MX
MA3GTS
MOLD
MHTPNT
HISTORY
HEXMULT
HEXMLT
HEXINV
HEXCHZ
HEXADD
S1CHME
117. FELDEL
CALLS: UNPACK, UNSNAP, RELEASE, TAPE6#, OUTCI.
CALLED BY: SELECT

118. FETCH
CALLS: ENTRYP, INPB1, EXITp
CALLED BY: MADEM

119. FILERUP
CALLS: RANDOM, TAPE6#, OUTCI, GIMME, YANK, STICK, DEL#00
CALLED BY: BYPONRL, BYPON88

120. FINOBK
CALLS: ENTRYP, ROUTER, MESSAGE, RITE1, CLIST, EXITp
CALLED BY: TOWER, TTRAPL, RNAMEP, REDEBRF, NDNL4ND, N3WCLT, N3M#0VE, KLFIFIT, INT2CRC, N3M#0VE, G30GET14, GNLOOK, FORMTGT, FLT#0RE
121. FINDFLT CALLS: ENTRYP HEXOIST FINDBLK KOMPARE GIMME ADDBLOK EXITD
   CALLED BY: FORMTGT
122. FINDIT CALLS: FINDBLK GIMME ADDBLOK
   CALLED BY: INTFINO
123. FIRECHK CALLS: THX2XY SIN. COS. SRT. ATAN2. ASIN.
   CALLED BY: TRYSHOT
124. FLITE CALLS: INTRFLY UNPACK COMMAND HEXCH2 OPTOTH RELEASEF SIN. DELADD
   CALLED BY: FLY
125. FLTGEOM CALLS: PACK HEXCH2 THM2PS ATAN2.
   CALLED BY: GOGETEM ACFRAG
126. FLTWYDE CALLS: UNPACK
   CALLED BY: KILFLIT
RELEASF  
FINDBLK  
TRACE  
MESSAGE  
RITEI  
CLIST  
FSDUMP  
DROPBLK  
PACK

127. FLY  

CALLS:  
GETPTRS  
UNSTAT  
MEXHOVF  
FUELCHK  
SHRKTLL  
FLITE  
DELADO  
STATEF  
GIMME  
PACT  
TAPE$#  
OUTC!  
DESTROY

128. FLYSEP  

CALLS:  
DELADO  
GIMME  
ADDDBLK  
TAPE$#  
OUTC!  
MESTBLD  
RELEASE  
MEXDIST  
DETCT  
HISTORY

129. FORNTAT  

CALLS:  
ENTRYP  
GIMME  
FINDBLK  
FINDFLY  
RELEASE  
EXIT$

130. FSUMP  

CALLS:  
ENTRYP  
TRACE  
ENTSTAT  
CLIST  
RITER  
LMPLLOT  
LCMLOC  
TAPE$#  
OUTC!  

PAGE 33
131. FSINIT
OUTCR, EXITP

132. FUELCHK
CALLS:
UNPACK
EXIST
GOTOAB
TAPE6#
OUTCI,
WESBILDO
DELADD

CALLED BY:
UNPACK
FLY
DOSTMNK

133. GETHEX
CALLS:
ENTRYP
HDGTS
MESSAGE
RITEI
GIMME
SCHTAB
EXITP

CALLED BY:
ENTRYP
RXVZXL
SCHEDUL
PTRAND
NUKBLND
NOWUCIT
HEXCHZ

134. GETPTRS
CALLS:
ENTRYP
EXITP

CALLED BY:
ENTRYP
TOWER
RONDSEE
POWDER
PLAN
PERCEPT
SAMPBCY
FLY
ENGAGE
DOGFITE
DESTROY
BYPASUP
BYENOPS
ATTACK
ASSIGN

135. GIMME
CALLS:
ENTRYP
HALT
MESSAGE
RITEI
RITEP
EXITP

CALLED BY:
ENTRYP
WHOMRAH
ULLOAD
UNPIRE
TOWER
TO40IL
TSTGOME
SHRKILL
SEEKTAC
SEEKENG
S010EST
SAMPBCY
REVISE
RESUPLY
RELOAD

PAGE 38
136. GNDLOOK

CALLS:
- FINDBLK
- XPD
- RANKDO
- DELADD
- HISTORY
- TAPE6#
- QUTCI+
- UNPACK
- THM2PS
- ATAN2.

CALLED BY:
- CFLYCRC

137. GOGETEN

CALLS:
- UNPACK
- FINOBLK
- CRPLTML
- TAPE6#
- QUTCI+
- GIMME
- ADDBLOK
- PTRAND
- PACK
- FLTGEOM
- ULOAD
- INITACO
- DELADD
- WESBILD

CALLED BY:
- TOWER

138. GOTOAB

CALLS:
- UNPACK
- FLTWTYPE
- OPTPRTM
- RELEASE
- THM2PS
- ATAN2.

CALLED BY:
- FUELCHK
- DOGTMNK

139. HALT

CALLS:
- HOLD
- OUTCI+
- RITER
- RITEI
- TRACE
- RECENT
- ENSTAT
- CLIST
- PAGE
- ADUMP
- ISDUMP
- ENDFIL.

CALLED BY:
- UNPACK
- ITRAP
- RECCON
- SELECT
- RELEASE
- PACK
- HLTPRTM
- GIMME
- WADEM
STOP.

140. NANDZPT
CALLS:
BATCEAS
YANK
STICK
TOADIL
RELOCAT
DILOUT

141. HEXADD
CALLS:
ENTRYP
ITOL
EXITP

142. HEXCHZ
CALLS:
ENTRYP
TRACE
MESSAGE
RITEP
CLIST
HEXADD
HEXINV
GETHEX
EXITP

143. MEXOIST
CALLS:
ENTRYP
TRACE
MESSAG
RITEP
CLIST
HEXADD
HEXINV
GETHEX
EXITP

144. HEXINV
CALLS:
ENTRYP
ITOL
EXITP

145. HEXMLT
CALLS:
HEXADD
ENTRYP
MASTTS

CALLED BY:
BNLALLE

142. HEXCHZ

143. MEXOIST

SCHEDUL
NATMOR
INTERFLY
FUELCHK
FLYSEE
FINDFLY
DETECT
CRCEVENT
CORRBOUND
CLOSCOR
FINASIN
BADMOVE
INTASIN
AIRTHNK
ACFRAQ

SCHEDUL
NATMOR
INTERFLY
FUELCHK
FLYSEE
FINDFLY
DETECT
CRCEVENT
CORRBOUND
CLOSCOR
FINASIN
BADMOVE
INTASIN
AIRTHNK
ACFRAQ

HEXCHZ

I12M1X
146. HEXMOVE CALLS:
MESSAGE
RITEI
EXITP
CALLED BY:
TRACE
MESSAGE
RITEI
CLIST
UNPACK
XTOI.
NOWUCIT
FINOLBK
DROPBLK
RELEASE
PACK
UOLOAD
CALLED BY:
FLY

147. HEXMULT CALLS:
ENTRYP
ITOJ.
EXITP
CALLED BY:
PTAND
NOWUCIT

148. HISTORY CALLS:
ENTRYP
MESSAGE
MASKER
TAPE6K
OUTCI.
EXITP
CALLED BY:
UMPIRE
SHKILL
NEWPERC
NEWMOVE
GNDLOOK
FLYSEE
ENGAGE
OOGFITE
CRFLML
BADMOVE
ATTACK

149. HLTONT CALLS:
ENTRYP
SECOND
MALT
EXITP
CALLED BY:
CONTROL

150. MOLD CALLS:
ENTRYP
OUTBI.
REWIND.
EXITP
CALLED BY:
MALT

151. MXDGETS CALLS:
ENTRYP
EXITP
CALLED BY:
TH2MX
TH2XY
HEXMLT
GETHER

152. MXMLT2 CALLS:
<table>
<thead>
<tr>
<th>Line</th>
<th>Function</th>
<th>Calls</th>
<th>Called By</th>
<th>Called By Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>153</td>
<td>ICHECK</td>
<td>CALLS: OUTCP</td>
<td>EXITP</td>
<td></td>
</tr>
<tr>
<td>154</td>
<td>IJ2HX</td>
<td>CALLS: HXMLT, ENTRYP, ITOJ, EXITP</td>
<td>CALLS: EXITP</td>
<td>CALLS: IJ2HX, IXYZNL, HXMLT2</td>
</tr>
<tr>
<td>155</td>
<td>INIT</td>
<td>CALLS: EXITP</td>
<td>CALLS:</td>
<td></td>
</tr>
<tr>
<td>156</td>
<td>INITACQ</td>
<td>CALLS: ENTRYP, GIMME, ADDOBLOK, ALG, MEGASGE, RITEI, XTOI, NOWUCIT, EXITP</td>
<td>CALLS: EXITP</td>
<td>CALLS: TOWER, GDGETE4</td>
</tr>
<tr>
<td>157</td>
<td>INRANGE</td>
<td>CALLS: TH2PS, ATAN2, SQRT, COS, TAPES#, OUTCP, AZILIM</td>
<td>CALLS: EXITP</td>
<td>CALLS: BYCONTC, CMKCOV, BYPONRL, BYNVTRK, BNPONNB</td>
</tr>
<tr>
<td>158</td>
<td>INSECT</td>
<td>CALLS: TAN, ATAN2</td>
<td>CALLS: EXITP</td>
<td>CALLS: AZILIM</td>
</tr>
<tr>
<td>159</td>
<td>INSERT</td>
<td>CALLS: UNPACK, XSHIFT</td>
<td>CALLS: EXITP</td>
<td>CALLS: SSLL</td>
</tr>
<tr>
<td>160</td>
<td>INTASIN</td>
<td>CALLS: FINDBLK, TRACE, MEGASGE, RITEI, CLIST, UNPACK, MEGAOIST, TOSHHEX, MESSBILD, PACK, DELADD</td>
<td>CALLS: EXITP</td>
<td>CALLS: ASSIGN</td>
</tr>
</tbody>
</table>
161. INTFINO

CALLS:
- UNPACK
- TAPE6#
- OUTCI.1
- FINDIT
- PACK
- MESSBILD
- DELADD

162. INTFLY

CALLS:
- TOTHEX
- MESSBILD
- MESSBILD
- DELADD
- TAPE6#
- OUTCI.1
- UNPACK
- THM2PS
- ATAN2.
- OPTRTM
- RELEASE

163. INT2CRC

CALLS:
- UNPACK
- FINDBLK
- RELEASE
- CRCKIL
- TAPE6#
- OUTCI.1
- DETECT
- MESSBILD
- DELADD
- DROPBLK
- GIMME
- ADDBLOK

164. IPJL

CALLS:
- LCMLOC
- ISHIFT
- TAPE6#
- OUTCI.1

165. ISDUMP

CALLS:
- OUTCI.1
- OUTCR.

166. ISHIFT

CALLS:
- OUTCI.1
- OUTCR.

CALLED BY:
- FLITE
- CRCTHMK
- MALT
- UNPACK
- PACK
- SSSL
- PTREE
- LOSRADR
167. ITREP
CALLS:
MALT

168. JGESUIT
CALLS:
ENTRYP
THX2XY
ATAN2
EXITP

169. JTI
CALLS:
ENTRYP
MESSAGE
RITEI
SHIFT
TAPE6
OUTC1
FSOUMP
STOP
EXITP

170. JUGGLE
CALLS:
ENTRYP
EXITP

171. KILFLIT
CALLS:
GIMME
DELAY
STOP
UNPACK
RELEASE
UNSTAT
FLTYPE
FINDBLK
DROPBLK

172. KOMPARE
CALLS:
ENTRYP
UNPACK
PACK
EXITP

173. LCMLOC
CALLS:
UNPACK
RITEP
PACK
TPJL
FSOUMP
MADEN

174. LINEX
CALLS:

CALLED BY:
ENTRYP
THTRPL4

CALLED BY:
EXITP
THTRPL4

CALLED BY:
ENTRYP
CANDGT

CALLED BY:
THX2XY

CALLED BY:
DESTROY

CALLED BY:
THTRPL4
FINOFIT

CALLED BY:
UNPACK
RITEP
PACK
TPJL
FSOUMP
MADEN

CALLED BY:

175. LNPLT
    CALLS:
    TAPES, OUTCT.
    CALLED BY:
    TAPE60, OUTPTRS, FS_DUMP, CLIST

176. LOADPL
    CALLS:
    ENTRYP, EXITP

177. LOSRAG
    CALLS:
    ISHIFT, OPTPTP, TMHP2P, SORT, COS, UNPACK, RELEASE
    CALLED BY:
    BYTKCH, DETECT, CRK_EVT

178. LRKPRS
    CALLS:
    CALLED BY:
    MADEM

179. LTREE
    CALLS:
    ENTRYP, RELEASE, LTRMRG, EXITP
    CALLED BY:
    CONTROL

180. LTRMRG
    CALLS:
    ENTRYP, EXITP
    CALLED BY:
    LTREE, DELADD

181. MADEM
    CALLS:
    QENTRY, RECON, ENTRYP, INPFL, OBGRADD, RECOVER, GOTOER, FETCH, PAGE, LCMLOC, FSINIT, OTHRDATA, DISPOAT, HALT, INIT, LRKPRS, RELIST, DELADD, CONTROL
182. MESSAGE

EXITP
END

183. MESSAGE

called by:
HISTORY

184. MESSAGE

called by:
HISTORY

CALLS:

182. MESSAGE

TAPES#
OUTCI.

183. MESSAGE

OUTPTRS
TRKCHEX
SAMPRCM
NAYBOR
JTJ
INITACQ
HISTORY
HEXMOVE
HEXMLT
HEXCHZ
GIMME
GETHEX
FLTWTPE
FINDBLK
DELOAD
CRC4CIES
CLIST
INTASIN

184. MESSAGE

CALLS:

GIMME
PACK

called by:

TOWER
BYCONTG
BNLALLE
INTFALY
INTZCRC
INTZFIND
CRC2ZINT
GODETEH
FUELCHK
FLYSEE
DROPPS2
DROPPS3
DOGTMNK
DECRALD
CRCTRAK
COMMAND
SYIPRAL
SYIFNFO
SYIFNER
SYNWTTRK
185. WAYBOQ

CALLS:
UNPACK
MESSAGE
RITEI
TRACE
CLIST
HEXOIST
GIMME
DELAOD
RELEASE

CALLED BY:
SELECT

186. NEWMOVE

CALLS:
FINDBLK
RANDOM
TAPE6
OUTCI-
GIMME
ADDBLK
HISTORY
DELAOD

CALLED BY:
CRCTRAK

187. NEWPERC

CALLS:
HISTORY
GIMME
STICK
DELAOD

CALLED BY:
SAMP RC4
BYPASUP

188. NOWUCIT

CALLS:
ENTRYP
HEXADD
HEXMULT
GETHEM
GIMME
ADDBLK
FINDBLK
DROPBLK
RELEASE
EXITP

CALLED BY:
TERMAC2
IVITAC2
HEXMOVE
Pei3E
45

189. NUKBLYO CALLS:
UNPACK
GIMME
NEXADD
GETHEX
TAPE6#
OUTCL.
DELADD
SAMWYPE
TERMACO
FINDBLK
RELEASE
DROPBLK
UNLINK

190. OORTPH CALLS:
GIMME
THM2PS
HECMZ
LINEX
PACK

191. OTHROAT CALLS:
ENTRYP
TAPE6#
OUTCL.
EXITP

192. OUTA CALLS:
ENTRYP
TAPE6#
OUTCL.
EXITP

193. OUTPTRS CALLS:
ENTRYP
LCMLOC
PAGE
MESSAGE
RTE1
RTER
RTER
UNPACK

194. PACK CALLS:
ENTRYP
LCMLOC
PAGE
MESSAGE
RTE1
TRACE
ROUTER
RTER
ISDUMP
HALT
EXITP

PAGE 45

411
195. PAGE

CALLS:
- TAPE60
- OUTC1.

CALLED BY:
- CLIST2
- HALT
- PACK
- CLIST
- MADAEN

196. PATDEC

CALLS:
- CANCALO
- SEEKTFU

CALLED BY:
- BYAICOV

197. PELADD

CALLS:
- ENTRYP
- GIMME
- PTRNOS
- EXITP

CALLED BY:
- REDEBRF
- CANDTGT
- AVAILBL

198. PERCEPT

CALLS:
- GETPRAS
- ABSEE
- CFLYCRC
- SAMSEE

CALLED BY:
- SELECT

199. PLAN

CALLS:
- ENTRYP
- GETPRAS
- THTQPLN
- EXITP

CALLED BY:
- SELECT
<table>
<thead>
<tr>
<th>Function</th>
<th>Calls</th>
<th>Called By</th>
</tr>
</thead>
<tbody>
<tr>
<td>200. PLANOUT</td>
<td>CALLS1: ENTRYP OUTCP EXITP</td>
<td>CALLED BY1: TMTTRPLY</td>
</tr>
<tr>
<td>201. PONDER</td>
<td>CALLS1: GETPRTS TFLYCRC BOCTINK BTRYTINX</td>
<td>CALLED BY1: SELECT</td>
</tr>
<tr>
<td>202. PREPAFU</td>
<td>CALLS1: UNPACK PACK AUTOPRI STICK RELOCAT DELADD</td>
<td>CALLED BY1: BYCOMTC BYPONRL BYNWTRK</td>
</tr>
<tr>
<td>203. PRIORTY</td>
<td>CALLS1: UNPACK PACK CANCALO SEEKTPU DILOUT DELADD</td>
<td>CALLED BY1: SETASSN CHKLAST BNPONBB 8NCONHD ALLOBAT</td>
</tr>
<tr>
<td>204. PTPONER</td>
<td>CALLS1: UNPACK PACK CANCALO SEEKTPU DILOUT DELADD</td>
<td>CALLED BY1: BTPONER</td>
</tr>
<tr>
<td>205. PTRAND</td>
<td>CALLS1: UNPACK RANODM. HEXADD MEXMULT GETHER</td>
<td>CALLED BY1: GOSETE4</td>
</tr>
<tr>
<td>206. PTREE</td>
<td>CALLS1: ENTRYP RELEASE ISMFT EXITP</td>
<td>CALLED BY1: TSTGONE CANOTGT</td>
</tr>
<tr>
<td>207. PTMRNG</td>
<td>CALLS1: UNPACK YANK STICK</td>
<td>CALLED BY1: PELADD</td>
</tr>
<tr>
<td>208. READIL</td>
<td>CALLS1: UNPACK YANK STICK</td>
<td>CALLED BY1: SEEKTPU CANCALO BNPONER</td>
</tr>
<tr>
<td>Page 50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>214. RELIST</td>
<td>CALLED BY: MADEN</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>ENTRYP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELEASE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EXITP</td>
<td></td>
</tr>
<tr>
<td>215. RELOAD</td>
<td>CALLED BY: BYPONRS</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>GIMME</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GOTOER.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DELADD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RESUPLY</td>
<td></td>
</tr>
<tr>
<td>216. RELOCAT</td>
<td>CALLED BY: TOADIL</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>YANK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STICK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TOADIL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SETASSN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAMATON</td>
<td></td>
</tr>
<tr>
<td></td>
<td>READIL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PREPAFU</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HANDZLOT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BYCONTC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CHKLAST</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYNWRK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BYHEDEUP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BYCONHO</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BYPONRS</td>
<td></td>
</tr>
<tr>
<td>217. RELSILL</td>
<td>CALLED BY: SCHEDUL</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>RELEASE</td>
<td></td>
</tr>
<tr>
<td>218. RENDEVU</td>
<td>CALLED BY: SCHEDUL</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>UNPACK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TMAZEXY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TATZMHL</td>
<td></td>
</tr>
<tr>
<td>219. RESUPLY</td>
<td>CALLED BY: RELOAD</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>GIMME</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DELADD</td>
<td></td>
</tr>
<tr>
<td>220. REVISE</td>
<td>CALLED BY: TMRPLY</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>ENTRYP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GIMME</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ABVSCOR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELEASE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EXITP</td>
<td></td>
</tr>
<tr>
<td>221. RITEI</td>
<td>CALLED BY: CLIST2</td>
<td></td>
</tr>
<tr>
<td>CALLS:</td>
<td>TAPE#</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OUTCI.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CLIST2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HALT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EXITP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ENTRYP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TSTHEX</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SELECT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELEASE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PACK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OUTPTRS</td>
<td></td>
</tr>
</tbody>
</table>

416
222. RITER

Calls:
- LCMLOC
- TAPE6#
- OUTCI.

Called by:
- CLIST2
- UNPACK
- RELEASE
- PACK
- OUTPUTS
- SAMPROC
- HEXMX
- GIMME
- CLIST
- INTASIN

223. RITER

Calls:
- TAPE6#
- OUTCI.

Called by:
- CLIST2
- WAIT
- OUTPUTS
- F5DUMP
- CLIST

224. RL480B

Calls:
- ENTRYP
- RELEASE
- EXITP

Called by:
- RLCORD

225. RLCORD

Calls:
- ENTRYP
- RL480B
- RELEASE
- EXITP

Called by:
- RLRAID

226. RL4MAK

Calls:
- ENTRYP
- RELEASE
- EXITP

Called by:
- RL4TAK

227. RLRAID

Calls:
- ENTRYP
- RELEASE
- EXITP

Called by:
- SYMPONER
- BATCEAS

245. SELECT

Calls:
- ENTRYP
- RELEASE
- EXITP

Called by:
- ENTRYP
- RELEASE
- EXITP
ENTRYP
GOTDER,
MESSGE
RITEI
HALT
FELDEL
ASSIGN
ATTACK
COMNO
DOGSITE
ENGAGE
FLY
NEBOR
PERCEPT
PLAN
PONDER
TOWER
UMPRED
TAPE60
OUTCI.
OUTCR.
EXITP

246. SETASSN CALLS1 CALLED BY1:
UNPACK
PACK
STICK
PRIORITY
RELOCAT
DELADD

247. SHKFILL CALLS1 CALLED BY1:
ALOG.
RANDOM.
HISTORY
UNPACK
TAPE60
OUTCI.
GIMME
PACK
DELADD
DESTROY

248. SHUFFLE CALLS1 CALLED BY1:
TH2MX

249. SKSB TK CALLS1 CALLED BY1:
UNPACK
SEEKP

250. SNAP CALLS1 CALLED BY1:
ENTRYP
EXITP

251. SSSL CALLS1 CALLED BY1:
252. STATRAK

UNPACK
ISSHIFT
INSERT

CALLED BY:
TFLYCRC
REALSEE
FLY
TFLYCRC
ATTACK

253. STICK

CALLSI:
UNPACK
PACK

CALLED BY:
TOADIL
SETASSW
DIGEST
READIL
PREPAFU
TAKCHEK
RELOCAT
HANDZPT
WMPERC
BYKCHK
BYCONTC
BYLALLE
BYCONTC
FILERUP
OLYACT
BYNWTRK
BYCONHO
BNPOWER
BNPOWER
BNCONHO
B Tateas
ALLOPAT
ALLOFU
ALLOBAT
ACCEPT

254. TERMACO

CALLSI:
UNPACK
XTO1.
NOMICIT
RELEASE

CALLED BY:
NUKBLND
DESTROY

255. TFLYCRC

CALLSI:
CRCFHNK
UNSTAT
AIRFHNK
DOGFHNK
STATRAK

CALLED BY:
POWDER

256. TSTGONE

CALLSI

CALLED BY:

419
<table>
<thead>
<tr>
<th>257. T0THEX</th>
<th>CALLS:</th>
<th>CALLED BY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRYP</td>
<td>TRACE</td>
<td>INTRFLY</td>
</tr>
<tr>
<td>T0THEX</td>
<td>MESA B</td>
<td>GTHMOVE</td>
</tr>
<tr>
<td>GIMME</td>
<td>RITE</td>
<td>INTASIN</td>
</tr>
<tr>
<td>EXITP</td>
<td>CLIST</td>
<td>A1RTMK</td>
</tr>
<tr>
<td></td>
<td>UNPACK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TH2PS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AT4MZ</td>
<td></td>
</tr>
<tr>
<td></td>
<td>COS.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SORT.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SIN.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>T4TAS14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CLIST</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>258. TH2PS</th>
<th>CALLS:</th>
<th>CALLED BY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRYP</td>
<td>ENTRYP</td>
<td>UMPIRE</td>
</tr>
<tr>
<td>TMX2XY</td>
<td>TMX2XY</td>
<td>T0THEX</td>
</tr>
<tr>
<td>EXITP</td>
<td>EXITP</td>
<td>BONDSEE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OPTPTH</td>
</tr>
<tr>
<td></td>
<td></td>
<td>T4ACHE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INRANGE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOSRABS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INTRFLY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CRCZINT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GOTOAB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>440LOOK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FLTGEOM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DETECT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CORBOUN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMMAND</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATKASES</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>259. TMRPLN</th>
<th>CALLS:</th>
<th>CALLED BY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRYP</td>
<td>ENTRYP</td>
<td>PLAN</td>
</tr>
<tr>
<td>CORBOUN</td>
<td>CORBOUN</td>
<td></td>
</tr>
<tr>
<td>REVISe</td>
<td>REVISe</td>
<td></td>
</tr>
<tr>
<td>FINDBLK</td>
<td>FINDBLK</td>
<td></td>
</tr>
<tr>
<td>RCMPARE</td>
<td>RCMPARE</td>
<td></td>
</tr>
<tr>
<td>CANDTGT</td>
<td>CANDTGT</td>
<td></td>
</tr>
<tr>
<td>ITRAP</td>
<td>ITRAP</td>
<td></td>
</tr>
<tr>
<td>AVAILBL</td>
<td>AVAILBL</td>
<td></td>
</tr>
<tr>
<td>SCHEDUL</td>
<td>SCHEDUL</td>
<td></td>
</tr>
<tr>
<td>DEL400</td>
<td>DEL400</td>
<td></td>
</tr>
<tr>
<td>PLANOUT</td>
<td>PLANOUT</td>
<td></td>
</tr>
<tr>
<td>RLRAID</td>
<td>RLRAID</td>
<td></td>
</tr>
<tr>
<td>EXITP</td>
<td>EXITP</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>260. TMX2XY</th>
<th>CALLS:</th>
<th>CALLED BY:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRYP</td>
<td>ENTRYP</td>
<td>TMX2PS</td>
</tr>
<tr>
<td>HADTS</td>
<td>HADTS</td>
<td>RENDEVU</td>
</tr>
<tr>
<td>JUGGLE</td>
<td>JUGGLE</td>
<td>JGESUIT</td>
</tr>
</tbody>
</table>
EXITP

261. TH2MH CALLS: 
HXXGTS
SHUFFLE

262. TOA2L CALLS: 
TAPES
OUTCL.
YANK
STICK
RELOCAT
GIMME

263. TOWER CALLS: 
GETPTRS
TAPES
UNPACK
UULLOAD
INITACQ
DELADD
PACK
FINDBLK
REDEBF
GIMME
GOGETEM
MESBILD
DESTROY

264. TRACE CALLS: 
TRAPRT
TRAPMT
TRAPNT

265. TRKCHEK CALLS: 
TH2P5
SORT.
ATAN2.
YANK
MESSAGE

FIRECHK CALLS: 
CALLED BY: 
HXXLT2

262. TOA2L CALLS: 
CALLED BY: 
S01GSF
SAMA10N
TRKCHEK
MANOZPT
BYTKCHK
BYCONTC
BYCONTC
DILOUT
ALLOBAT

263. TOWER CALLS: 
CALLED BY: 
SELECT

264. TRACE CALLS: 
CALLED BY: 
HALT
YANK
UNSTAT
TOTMEX
PACK
SAMPROM
YBOLR
HEXMOVE
HEXCHZ
FSQUMP
FLTYPE
DELADD
COCOIES
INTASIV

265. TRKCHEK CALLS: 
CALLED BY: 
SEEKTFU
BATTCOV
RITEI
CLIST
FSOUMP
STICK
TOADIL
BNLALLE

266. TRPRMT
CALLS:
TAPES#
OUTCI

267. TRPRMT
CALLS:
TAPES#
OUTCI

268. TRPRMT
CALLS:
GOTOSER,
TAPES#
OUTCI,
YANK
BATCEAS
BNLALLE
FIRECHK

269. TRYSHOT
CALLS:
ENTRYP
EXITP

270. TXY2MH
CALLS:
ENTRYP
EXITP

271. TXY2MHXL
CALLS:
ENTRYP
COS.
SIN.
XTOI.
CENTER
L2MH
GETHEX
EXITP

272. UNPIRE
CALLS:
UNPACK
THM2PS
SORT.
XTOI.
HISTORY
RANDOM,
TAPES#
OUTCI.
GIMME
PACK
DELADD
DESTROY
RELEASE
NUKBLNO

Called by:
TRACE
TRACE
TRACE
ENGAGE
BYPONT4

PAGE 59
| Page 50 |
|---|---|---|---|

### 273. UNLINK

**Called By:**
- DISK

**Calls:**
- UNPACK

| Page 50 |
|---|---|---|---|

### 274. UNPACK

**Called By:**
- DISK

**Calls:**
- ZAP

**Called By:**
- INITIAL
- DISK
275. UNSNAP

CALLS:

ENTRY
EXIT

CALLED BY:

FELDEL
CONTROL

276. UNSTAT

CALLS:

TRACE
MESSAGE
CLIST
FSKUMP
STOP.
UNPACK

CALLED BY:

FLYCRC
RODSEE
KILFLIT
FLY
DOGITE
DESTROY
CFLYCRC
ATTACK

277. UOLLOAD

CALLS:

ENTRY
GIMME
ADDGBK

CALLED BY:

TOWER
HEXMOVE
GOGETEW
EXITP

278. WIPEOUT

279. TMODR

CALS1

UNPACK
GIMME
PACK
SSL
DECLALO
SEEKENG
OLYACT
RELEASE

280. XPA

281. XDO

282. XPK

283. XSHIFT

284. XY2HX

CALS1

CENTER
XY2HX

285. YANK

CALS1

UNPACK
TAPE6#
OUTCI.
TRACE
OUTA
FSDUMP
STOP.
PACK

CALS1

TRYSHOT
TOADL
SEEKAC
SUBPRE
READL
TACCHEK
S&PRCN
RELLOCAT
HAND2PT
BYTKCHK
BYCONT
BYLALL
BYCONT
FILERUP
DROPPS2
DROPPDS
DILOUT
CNACTT
CN&CALD
BYPONFD
BYENDOR
BTRYNKL
BYPONFD

CALLED BY:
REDEBRF
CALLED BY:
SNRECOV

CALLED BY:
MOIFEST
CALLED BY:
SNLOOK
CALLED BY:
ATTACK
CALLED BY:
INSERT

CALLED BY:

PAGE 52

425
1. ALOG.
   CALLED BY:
   SHKILL
   INITACS

2. ASIN.
   CALLED BY:
   FIRECHK

3. ATAN2.
   CALLED BY:
   TGTHEX
   RONGSEE
   TRMCHEK
   INVANGE
   BYTKCHK
   JGESUIT
   TYPFLY
   CRC2INT
   INSECT
   GOTOAB
   NOLOOK
   FLTGEOM
   FIRECHK
   DETECT
   COMMAND
   ATKASES

4. COS.
   CALLED BY:
   TTY2MHXL
   TGTHEX
   INVANGE
   INSRAD
   FIRECHK
   DETECT
   CRCCEVT
   CORBON
   RIZILIM

5. DECODI.
   CALLED BY:
   DBOREAD

6. ENDFIL.
   CALLED BY:
   HALT

7. ENO.
   CALLED BY:
   MAEM

8. GOTOER.
   CALLED BY:
   TRYSHOT
   SELECT
   SANYPE
   RELOAD
   COMMAND
9. INPBI.

10. INPCI.

11. INPFI.

12. ITOJ.

13. OUTBI.

14. OUTCI.

Called by:

- FETCH
- DBREAD
- MADEM
- EU2MX
- HEXMULT
- HEXINV
- HEXADD
- DGTSHA
- HOLD
- CLIST2
- ISDUMP
- MALT
- ADUMP
- DBREAD
- ICCHECK
- DISPPYR
- DISPPRO
- DISPPAY
- DISPPAF
- DISPFMFM
- DISPFLT
- DISPFDB
- DISPAQQ
- DISPAOS
- DISPAACR
- DISPACL
- DISPACD
- DISPABQ
- DISPADT
- EXIT
- ENTRYP
- ENTSTAT
- ROUTER
- RECER
- RECON
YANK
UMPIRE
TRYSHOT
TAPENT
TOWER
TOADIL
SHRKILL
SELECT
SOIGEST
SAMATION
RONDSEE
RITER
RITEP
RITEI
READIL
BYALCOV
PLANOUT
PAGE
OUTA
NUKBLNO
INRANGE
NEWMOVE
&CONTC
MESSAGE
LWPLT
JTJ
IPJL
INTRFLY
INTCRRC
INTFIND
CRZINT
HISTORY
GOGETEN
GNLLOOK
FUELCHK
FSDUMP
FLYSEE
FLY
FLERUP
FEDEL
ENGAGE
DOSTHNK
DOGITE
DETECT
DESTROY
DECDALO
CCTRAK
COKIL
CCEVNT
COMMAND
CLIST
CHKCOV
CANCALO
BYPRONTM
BYPONAR
18. REWIND.
   CALLED BY:
   HOLD

19. S1N.
   CALLED BY:
   TXY2MXL
   TSTMX
   FLITE
   FIRECHK
   CORBOUN
   AZILIM

20. SORT.
   CALLED BY:
   UMPIRE
   TSTMX
   TKCHER
   INRANGE
   BYTKCHK
   LSRADN
   FIRECHK
   AZILIM

21. STOP.
   CALLED BY:
   MALT
   YANK
   UVSTAT
   KILFI IT
   JT
   ACCEPT

22. TAN.
   CALLED BY:
   INSECT

23. TAPE
   CALLED BY:
   CLIST2
   EXITP
   ENTRYP
   ENSTAT
   ROUTER
   RECER
   RECON
   YANK
   UMPIRE
   PRESHT
   TAPRNT
   TOWER
   TOADIL
   SHNKILL
   SELECT
   SOIGEST
   SARATON
   RONEE
   RITER
   RITER
   RITE1
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00053</td>
<td>6. OUTCI.</td>
</tr>
<tr>
<td>00054</td>
<td>4. PECER (SEE LINE 00018)</td>
</tr>
<tr>
<td>00055</td>
<td>4. ENSTAT</td>
</tr>
<tr>
<td>00056</td>
<td>5. TAPE6#</td>
</tr>
<tr>
<td>00057</td>
<td>5. OUTCI.</td>
</tr>
<tr>
<td>00058</td>
<td>4. CLIST</td>
</tr>
<tr>
<td>00059</td>
<td>5. PAGE</td>
</tr>
<tr>
<td>00060</td>
<td>6. TAPE6#</td>
</tr>
<tr>
<td>00061</td>
<td>6. OUTCI.</td>
</tr>
<tr>
<td>00062</td>
<td>5. LNPLCT</td>
</tr>
<tr>
<td>00063</td>
<td>6. TAPE6#</td>
</tr>
<tr>
<td>00064</td>
<td>6. OUTCI.</td>
</tr>
<tr>
<td>00065</td>
<td>5. MESSAG (SEE LINE 00012)</td>
</tr>
<tr>
<td>00066</td>
<td>5. RITEP</td>
</tr>
<tr>
<td>00067</td>
<td>6. LCMLOC</td>
</tr>
<tr>
<td>00068</td>
<td>6. TAPE6#</td>
</tr>
<tr>
<td>00070</td>
<td>5. OUTCI.</td>
</tr>
<tr>
<td>00071</td>
<td>5. TAPE6#</td>
</tr>
<tr>
<td>00072</td>
<td>5. RITEP (SEE LINE 00044)</td>
</tr>
<tr>
<td>00073</td>
<td>5. RITEI (SEE LINE 00015)</td>
</tr>
<tr>
<td>00074</td>
<td>5. OUTCI.</td>
</tr>
<tr>
<td>00075</td>
<td>5. CLIST2</td>
</tr>
<tr>
<td>00076</td>
<td>5. PAGE (SEE LINE 00059)</td>
</tr>
<tr>
<td>00077</td>
<td>6. LNPLCT (SEE LINE 00062)</td>
</tr>
<tr>
<td>00078</td>
<td>6. MESSAG (SEE LINE 00012)</td>
</tr>
<tr>
<td>00079</td>
<td>6. RITEI</td>
</tr>
<tr>
<td>00080</td>
<td>6. RITE6 (SEE LINE 00015)</td>
</tr>
<tr>
<td>00081</td>
<td>6. TAPE6#</td>
</tr>
<tr>
<td>00082</td>
<td>6. OUTCI.</td>
</tr>
<tr>
<td>00083</td>
<td>5. OUTCI.</td>
</tr>
<tr>
<td>00084</td>
<td>5. OUTCR</td>
</tr>
<tr>
<td>00085</td>
<td>6. RITEP (SEE LINE 00066)</td>
</tr>
<tr>
<td>00086</td>
<td>4. PAGE (SEE LINE 00059)</td>
</tr>
<tr>
<td>00087</td>
<td>4. ADUMO (SEE LINE 00006)</td>
</tr>
<tr>
<td>00088</td>
<td>4. ISSUMP</td>
</tr>
<tr>
<td>00089</td>
<td>5. OUTCI.</td>
</tr>
<tr>
<td>00090</td>
<td>5. OUTCR</td>
</tr>
<tr>
<td>00091</td>
<td>4. ENDFIL</td>
</tr>
<tr>
<td>00092</td>
<td>4. STOP</td>
</tr>
<tr>
<td>00093</td>
<td>2. ENTRYP (SEE LINE 00011)</td>
</tr>
<tr>
<td>00094</td>
<td>2. INPFI</td>
</tr>
<tr>
<td>00095</td>
<td>2. DBREAD</td>
</tr>
<tr>
<td>00096</td>
<td>3. OUTCI.</td>
</tr>
<tr>
<td>00097</td>
<td>3. INPFI</td>
</tr>
<tr>
<td>00098</td>
<td>3. EOF</td>
</tr>
<tr>
<td>00099</td>
<td>3. DECODI</td>
</tr>
<tr>
<td>00100</td>
<td>2. RECOV</td>
</tr>
<tr>
<td>00101</td>
<td>2. JOTER</td>
</tr>
<tr>
<td>00102</td>
<td>3. FETCH</td>
</tr>
<tr>
<td>00103</td>
<td>3. ENTRYP (SEE LINE 00011)</td>
</tr>
<tr>
<td>00104</td>
<td>3. INPFI</td>
</tr>
<tr>
<td>00105</td>
<td>3. EXITP (SEE LINE 00033)</td>
</tr>
<tr>
<td>00106</td>
<td>2. PAGE (SEE LINE 00059)</td>
</tr>
<tr>
<td>00107</td>
<td>2. LCMLOC</td>
</tr>
<tr>
<td>00108</td>
<td>2. FSINIT</td>
</tr>
</tbody>
</table>

435
<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>00108</td>
<td>2. INPUT</td>
</tr>
<tr>
<td>00109</td>
<td>2. OUTPUT</td>
</tr>
<tr>
<td>00110</td>
<td>3. OUTCI.</td>
</tr>
<tr>
<td>00111</td>
<td>3. OUTPUT</td>
</tr>
<tr>
<td>00112</td>
<td>4. OUTFI.</td>
</tr>
<tr>
<td>00113</td>
<td>4. OUTPUT</td>
</tr>
<tr>
<td>00114</td>
<td>5. OUTFI.</td>
</tr>
<tr>
<td>00115</td>
<td>5. OUTPUT</td>
</tr>
<tr>
<td>00116</td>
<td>5. DISPFLT</td>
</tr>
<tr>
<td>00117</td>
<td>6. OUTCI.</td>
</tr>
<tr>
<td>00118</td>
<td>6. OUTPUT</td>
</tr>
<tr>
<td>00119</td>
<td>6. DISPFLY</td>
</tr>
<tr>
<td>00120</td>
<td>7. OUTCI.</td>
</tr>
<tr>
<td>00121</td>
<td>7. OUTPUT</td>
</tr>
<tr>
<td>00122</td>
<td>8. OUTFI.</td>
</tr>
<tr>
<td>00123</td>
<td>8. OUTPUT</td>
</tr>
<tr>
<td>00124</td>
<td>9. OUTFI.</td>
</tr>
<tr>
<td>00125</td>
<td>9. OUTPUT</td>
</tr>
<tr>
<td>00126</td>
<td>10. OUTFI.</td>
</tr>
<tr>
<td>00127</td>
<td>10. OUTPUT</td>
</tr>
<tr>
<td>00128</td>
<td>11. OUTFI.</td>
</tr>
<tr>
<td>00129</td>
<td>11. OUTPUT</td>
</tr>
<tr>
<td>00130</td>
<td>3. DISPFLT (SEE LINE 00117)</td>
</tr>
<tr>
<td>00131</td>
<td>3. DISPFLY (SEE LINE 00112)</td>
</tr>
<tr>
<td>00132</td>
<td>3. OUTCI.</td>
</tr>
<tr>
<td>00133</td>
<td>3. OUTPUT</td>
</tr>
<tr>
<td>00134</td>
<td>4. OUTFI.</td>
</tr>
<tr>
<td>00135</td>
<td>4. OUTPUT</td>
</tr>
<tr>
<td>00136</td>
<td>5. OUTFI.</td>
</tr>
<tr>
<td>00137</td>
<td>5. OUTPUT</td>
</tr>
<tr>
<td>00138</td>
<td>6. OUTFI.</td>
</tr>
<tr>
<td>00139</td>
<td>6. OUTPUT</td>
</tr>
<tr>
<td>00140</td>
<td>7. OUTFI.</td>
</tr>
<tr>
<td>00141</td>
<td>7. OUTPUT</td>
</tr>
<tr>
<td>00142</td>
<td>2. HALT (SEE LINE 00009)</td>
</tr>
<tr>
<td>00143</td>
<td>2. INPUT</td>
</tr>
<tr>
<td>00144</td>
<td>2. OUTPUT</td>
</tr>
<tr>
<td>00145</td>
<td>2. RELIST</td>
</tr>
<tr>
<td>00146</td>
<td>3. ENTRYP</td>
</tr>
<tr>
<td>00147</td>
<td>3. RELEASE</td>
</tr>
<tr>
<td>00148</td>
<td>4. ENTRYP (SEE LINE 00011)</td>
</tr>
<tr>
<td>00149</td>
<td>4. MESSAGES (SEE LINE 00012)</td>
</tr>
<tr>
<td>00150</td>
<td>4. RITEI (SEE LINE 00015)</td>
</tr>
<tr>
<td>00151</td>
<td>4. RITEP (SEE LINE 00066)</td>
</tr>
<tr>
<td>00152</td>
<td>4. HALT (SEE LINE 00009)</td>
</tr>
<tr>
<td>00153</td>
<td>4. GUNME</td>
</tr>
<tr>
<td>00154</td>
<td>5. ENTRYP (SEE LINE 00011)</td>
</tr>
<tr>
<td>00155</td>
<td>5. HALT (SEE LINE 00009)</td>
</tr>
<tr>
<td>00156</td>
<td>5. MESSAGES (SEE LINE 00012)</td>
</tr>
<tr>
<td>00157</td>
<td>5. RITEI (SEE LINE 00015)</td>
</tr>
<tr>
<td>00158</td>
<td>5. RITEP (SEE LINE 00066)</td>
</tr>
<tr>
<td>00159</td>
<td>5. EXITP (SEE LINE 00033)</td>
</tr>
<tr>
<td>00160</td>
<td>4. EXITP (SEE LINE 00033)</td>
</tr>
<tr>
<td>00161</td>
<td>3. EXIT</td>
</tr>
<tr>
<td>00162</td>
<td>2. DELADD (SEE LINE 00033)</td>
</tr>
<tr>
<td>Line</td>
<td>Instruction</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>00163</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00165</td>
<td>MESSAGE</td>
</tr>
<tr>
<td>00166</td>
<td>TRACE</td>
</tr>
<tr>
<td>00164</td>
<td>RECER</td>
</tr>
<tr>
<td>00168</td>
<td>GIVE</td>
</tr>
<tr>
<td>00169</td>
<td>SNAP</td>
</tr>
<tr>
<td>00170</td>
<td>EXIT</td>
</tr>
<tr>
<td>00171</td>
<td>LTWAG</td>
</tr>
<tr>
<td>00172</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00173</td>
<td>EXIT</td>
</tr>
<tr>
<td>00174</td>
<td>EXIT</td>
</tr>
<tr>
<td>00175</td>
<td>CONTROL</td>
</tr>
<tr>
<td>00176</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00177</td>
<td>LTREE</td>
</tr>
<tr>
<td>00178</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00179</td>
<td>RELEASE</td>
</tr>
<tr>
<td>00180</td>
<td>LTWAG</td>
</tr>
<tr>
<td>00181</td>
<td>EXIT</td>
</tr>
<tr>
<td>00182</td>
<td>SELECT</td>
</tr>
<tr>
<td>00183</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00184</td>
<td>GORDER</td>
</tr>
<tr>
<td>00185</td>
<td>MESSAGE</td>
</tr>
<tr>
<td>00186</td>
<td>RITEI</td>
</tr>
<tr>
<td>00187</td>
<td>HALT</td>
</tr>
<tr>
<td>00188</td>
<td>FELDOL</td>
</tr>
<tr>
<td>00189</td>
<td>UNPACK</td>
</tr>
<tr>
<td>00190</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00191</td>
<td>LCMLOC</td>
</tr>
<tr>
<td>00192</td>
<td>RITE</td>
</tr>
<tr>
<td>00193</td>
<td>ISDUMP</td>
</tr>
<tr>
<td>00194</td>
<td>MALT</td>
</tr>
<tr>
<td>00195</td>
<td>EXIT</td>
</tr>
<tr>
<td>00196</td>
<td>USNAP</td>
</tr>
<tr>
<td>00197</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00198</td>
<td>EXIT</td>
</tr>
<tr>
<td>00199</td>
<td>RELEASE</td>
</tr>
<tr>
<td>00200</td>
<td>TPEW</td>
</tr>
<tr>
<td>00201</td>
<td>DUTCI</td>
</tr>
<tr>
<td>00202</td>
<td>ASSION</td>
</tr>
<tr>
<td>00203</td>
<td>SETPTAS</td>
</tr>
<tr>
<td>00204</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00205</td>
<td>EXIT</td>
</tr>
<tr>
<td>00206</td>
<td>IVFASIN</td>
</tr>
<tr>
<td>00207</td>
<td>FIMDBLK</td>
</tr>
<tr>
<td>00208</td>
<td>ENTRY</td>
</tr>
<tr>
<td>00209</td>
<td>ROUTE</td>
</tr>
<tr>
<td>00210</td>
<td>MESSAGE</td>
</tr>
<tr>
<td>00211</td>
<td>RITEI</td>
</tr>
<tr>
<td>00212</td>
<td>CLIST</td>
</tr>
<tr>
<td>00213</td>
<td>EXIT</td>
</tr>
<tr>
<td>00214</td>
<td>TRACE</td>
</tr>
<tr>
<td>00215</td>
<td>MESSAGE</td>
</tr>
<tr>
<td>00216</td>
<td>RITEI</td>
</tr>
<tr>
<td>00217</td>
<td>CLIST</td>
</tr>
</tbody>
</table>
00218 6. UNPACK (SEE LINE 00189)
6. HEXOIST
6. TGMHEX
00219 7. TRACE (SEE LINE 00048)
00220 7. MESSAGE (SEE LINE 00012)
00221 7. RITEI (SEE LINE 00015)
00222 7. CLIST (SEE LINE 00058)
00223 7. UNPACK (SEE LINE 00189)
00224 7. THM2PS
00225 8. ENTRYD (SEE LINE 00011)
00226 8. HEXOIST
00227 8. TGTHEX
00228 8. ENTRYD (SEE LINE 00011)
00229 9. ENTRYD (SEE LINE 00011)
00230 9. MESSAGE (SEE LINE 00012)
00231 9. RITEI (SEE LINE 00015)
00232 9. EXITP (SEE LINE 00033)
00233 9. EXITP (SEE LINE 00033)
00234 7. ATAN2
7. COS
7. SORT
7. SINE
00235 7. EXITP (SEE LINE 00033)
00236 7. EXITP (SEE LINE 00033)
00237 7. CENTER
00238 9. ENTRYD (SEE LINE 00011)
00239 9. EXITP (SEE LINE 00033)
00240 9. EXITP (SEE LINE 00033)
00241 8. ENTRYD (SEE LINE 00011)
00242 8. ENTRYD (SEE LINE 00011)
00243 8. ENTRYD (SEE LINE 00011)
00244 8. ENTRYD (SEE LINE 00011)
00245 8. ENTRYD (SEE LINE 00011)
00246 9. ENTRYD (SEE LINE 00011)
00247 9. EXITP (SEE LINE 00033)
00248 9. EXITP (SEE LINE 00033)
00249 9. EXITP (SEE LINE 00033)
00250 9. EXITP (SEE LINE 00033)
00251 9. EXITP (SEE LINE 00033)
00252 9. EXITP (SEE LINE 00033)
00253 9. EXITP (SEE LINE 00033)
00254 9. EXITP (SEE LINE 00033)
00255 9. EXITP (SEE LINE 00033)
00256 9. EXITP (SEE LINE 00033)
00257 9. EXITP (SEE LINE 00033)
00258 9. EXITP (SEE LINE 00033)
00259 9. EXITP (SEE LINE 00033)
00260 9. EXITP (SEE LINE 00033)
00261 9. EXITP (SEE LINE 00033)
00262 9. EXITP (SEE LINE 00033)
00263 9. EXITP (SEE LINE 00033)
00264 9. EXITP (SEE LINE 00033)
00265 9. EXITP (SEE LINE 00033)
00266 9. EXITP (SEE LINE 00033)
00267 9. EXITP (SEE LINE 00033)
00268 9. EXITP (SEE LINE 00033)
00269 9. EXITP (SEE LINE 00033)
00270 9. EXITP (SEE LINE 00033)
00271 9. EXITP (SEE LINE 00033)
00272 9. EXITP (SEE LINE 00033)
<table>
<thead>
<tr>
<th>Line</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00274</td>
<td>EXIT</td>
<td>(See Line 00033)</td>
</tr>
<tr>
<td>00275</td>
<td>EXIT</td>
<td>(See Line 00033)</td>
</tr>
<tr>
<td>00277</td>
<td>MBSRDL</td>
<td></td>
</tr>
<tr>
<td>00278</td>
<td>DIMM</td>
<td>(See Line 00153)</td>
</tr>
<tr>
<td>00279</td>
<td>PACK</td>
<td></td>
</tr>
<tr>
<td>00280</td>
<td>ENTRYD</td>
<td>(See Line 00011)</td>
</tr>
<tr>
<td>00281</td>
<td>LCM3C</td>
<td></td>
</tr>
<tr>
<td>00282</td>
<td>PAGE</td>
<td>(See Line 00059)</td>
</tr>
<tr>
<td>00283</td>
<td>MESSAGE</td>
<td>(See Line 00012)</td>
</tr>
<tr>
<td>00284</td>
<td>RITE</td>
<td>(See Line 00015)</td>
</tr>
<tr>
<td>00285</td>
<td>Trace</td>
<td>(See Line 00048)</td>
</tr>
<tr>
<td>00286</td>
<td>ROUTER</td>
<td>(See Line 00024)</td>
</tr>
<tr>
<td>00287</td>
<td>RITE</td>
<td>(See Line 00056)</td>
</tr>
<tr>
<td>00288</td>
<td>SOUNI</td>
<td>(See Line 00087)</td>
</tr>
<tr>
<td>00289</td>
<td>HALT</td>
<td>(See Line 00009)</td>
</tr>
<tr>
<td>00290</td>
<td>EXIT</td>
<td>(See Line 00033)</td>
</tr>
<tr>
<td>00291</td>
<td>PACK</td>
<td>(See Line 00277)</td>
</tr>
<tr>
<td>00292</td>
<td>DELADD</td>
<td>(See Line 00162)</td>
</tr>
<tr>
<td>00293</td>
<td>TAPE</td>
<td></td>
</tr>
<tr>
<td>00294</td>
<td>OUTC</td>
<td></td>
</tr>
<tr>
<td>00295</td>
<td>BTFM</td>
<td></td>
</tr>
<tr>
<td>00296</td>
<td>UNPACK</td>
<td>(See Line 00189)</td>
</tr>
<tr>
<td>00297</td>
<td>PIN3LKL</td>
<td>(See Line 00207)</td>
</tr>
<tr>
<td>00298</td>
<td>PACK</td>
<td>(See Line 00277)</td>
</tr>
<tr>
<td>00299</td>
<td>HERDST</td>
<td>(See Line 00147)</td>
</tr>
<tr>
<td>00300</td>
<td>DROPATK</td>
<td></td>
</tr>
<tr>
<td>00301</td>
<td>ENTRYD</td>
<td>(See Line 00011)</td>
</tr>
<tr>
<td>00302</td>
<td>RELEASE</td>
<td>(See Line 00147)</td>
</tr>
<tr>
<td>00303</td>
<td>EXIT</td>
<td>(See Line 00033)</td>
</tr>
<tr>
<td>00304</td>
<td>MBSRDL</td>
<td>(See Line 00275)</td>
</tr>
<tr>
<td>00305</td>
<td>DELADD</td>
<td>(See Line 00162)</td>
</tr>
<tr>
<td>00306</td>
<td>TAPE</td>
<td></td>
</tr>
<tr>
<td>00307</td>
<td>OUTC</td>
<td></td>
</tr>
<tr>
<td>00308</td>
<td>ATTACK</td>
<td></td>
</tr>
<tr>
<td>00309</td>
<td>GETPR</td>
<td>(See Line 00203)</td>
</tr>
<tr>
<td>00310</td>
<td>UNSTAT</td>
<td></td>
</tr>
<tr>
<td>00311</td>
<td>TRACE</td>
<td>(See Line 00048)</td>
</tr>
<tr>
<td>00312</td>
<td>MESSAGE</td>
<td>(See Line 00012)</td>
</tr>
<tr>
<td>00313</td>
<td>CLIST</td>
<td>(See Line 00059)</td>
</tr>
<tr>
<td>00314</td>
<td>FSONDL</td>
<td></td>
</tr>
<tr>
<td>00315</td>
<td>ENTRYD</td>
<td>(See Line 00011)</td>
</tr>
<tr>
<td>00316</td>
<td>TRACE</td>
<td>(See Line 00048)</td>
</tr>
<tr>
<td>00317</td>
<td>ENSTAT</td>
<td>(See Line 00055)</td>
</tr>
<tr>
<td>00318</td>
<td>CLIST</td>
<td>(See Line 00059)</td>
</tr>
<tr>
<td>00319</td>
<td>RITE</td>
<td>(See Line 00044)</td>
</tr>
<tr>
<td>00320</td>
<td>LCM3C</td>
<td>(See Line 00062)</td>
</tr>
<tr>
<td>00321</td>
<td>TAPE</td>
<td></td>
</tr>
<tr>
<td>00322</td>
<td>OUTC</td>
<td></td>
</tr>
</tbody>
</table>
6. ALOG.
6. RANDOM.
6. HISTORY
7. ENTRYP
7. MESSAGE
7. WASKER
7. TAPES#
7. OUTCI.
7. EXITP
6. UNPACK
6. TAPES#
6. OUTCI.
6. GIMME
5. PACK
5. DELADD
6. DESTROY
7. TAPES#
7. OUTCI.
7. GETPTES
7. TERMAC
8. UNPACK
8. EXIT.
8. NMUCIT
9. ENTRYP
9. HEXADD
9. HEXMULT
10. ENTRYP
10. ITOJ.
10. EXITP
9. GETHFX
9. GIMME
9. ADBLCK
9. ENTRYP
10. ENTRYP
10. EXITP
9. FINBLK
9. DROPBLK
9. RELEASE
9. EXITP
8. RELEASE
7. KILPIT
8. GIMME
8. DELADD
7. STOP.
7. UNPACK
7. RELEASE
7. UNSTAT
7. FLPYF
7. UNPACK
7. RELEASE
7. FINBLK
7. TRACE
7. MESSAGE
7. RITEI
7. CLIST
7. FS0JMP
6. ALOI.
6. RANDOM.
6. HISTORY
7. ENTRYP
7. MESSAGE
7. WASKER
7. TAPES#
7. OUTCI.
7. EXITP
6. UNPACK
6. TAPES#
6. OUTCI.
6. GIMME
5. PACK
5. DELADD
6. DESTROY
7. TAPES#
7. OUTCI.
7. GETPTES
7. TERMAC
8. UNPACK
8. EXIT.
8. NMUCIT
9. ENTRYP
9. HEXADD
9. HEXMULT
10. ENTRYP
10. ITOJ.
10. EXITP
9. GETHFX
9. GIMME
9. ADBLCK
9. ENTRYP
10. ENTRYP
10. EXITP
9. FINBLK
9. DROPBLK
9. RELEASE
9. EXITP
8. RELEASE
7. KILPIT
8. GIMME
8. DELADD
7. STOP.
7. UNPACK
7. RELEASE
7. UNSTAT
7. FLPYF
7. UNPACK
7. RELEASE
7. FINBLK
7. TRACE
7. MESSAGE
7. RITEI
7. CLIST
7. FS0JMP
6. ALOI.
6. RANDOM.
6. HISTORY
7. ENTRYP
7. MESSAGE
7. WASKER
7. TAPES#
7. OUTCI.
7. EXITP
6. UNPACK
6. TAPES#
6. OUTCI.
6. GIMME
5. PACK
5. DELADD
6. DESTROY
7. TAPES#
7. OUTCI.
7. GETPTES
7. TERMAC
8. UNPACK
8. EXIT.
8. NMUCIT
9. ENTRYP
9. HEXADD
9. HEXMULT
10. ENTRYP
10. ITOJ.
10. EXITP
9. GETHFX
9. GIMME
9. ADBLCK
9. ENTRYP
10. ENTRYP
10. EXITP
9. FINBLK
9. DROPBLK
9. RELEASE
9. EXITP
8. RELEASE
7. KILPIT
8. GIMME
8. DELADD
7. STOP.
7. UNPACK
7. RELEASE
7. UNSTAT
7. FLPYF
7. UNPACK
7. RELEASE
7. FINBLK
7. TRACE
7. MESSAGE
7. RITEI
7. CLIST
7. FS0JMP
6. ALOI.
6. RANDOM.
6. HISTORY
7. ENTRYP
7. MESSAGE
7. WASKER
7. TAPES#
7. OUTCI.
7. EXITP
6. UNPACK
6. TAPES#
6. OUTCI.
6. GIMME
5. PACK
5. DELADD
6. DESTROY
7. TAPES#
7. OUTCI.
7. GETPTES
7. TERMAC
8. UNPACK
8. EXIT.
8. NMUCIT
9. ENTRYP
9. HEXADD
9. HEXMULT
10. ENTRYP
10. ITOJ.
10. EXITP
9. GETHFX
9. GIMME
9. ADBLCK
9. ENTRYP
10. ENTRYP
10. EXITP
9. FINBLK
9. DROPBLK
9. RELEASE
9. EXITP
8. RELEASE
7. KILPIT
8. GIMME
8. DELADD
7. STOP.
7. UNPACK
7. RELEASE
7. UNSTAT
7. FLPYF
7. UNPACK
7. RELEASE
7. FINBLK
7. TRACE
7. MESSAGE
7. RITEI
7. CLIST
7. FS0JMP
6. ALOI.
6. RANDOM.
6. HISTORY
7. ENTRYP
7. MESSAGE
7. WASKER
7. TAPES#
7. OUTCI.
7. EXITP
6. UNPACK
6. TAPES#
6. OUTCI.
6. GIMME
5. PACK
5. DELADD
6. DESTROY
7. TAPES#
7. OUTCI.
7. GETPTES
7. TERMAC
8. UNPACK
8. EXIT.
8. NMUCIT
9. ENTRYP
9. HEXADD
9. HEXMULT
10. ENTRYP
10. ITOJ.
10. EXITP
9. GETHFX
9. GIMME
9. ADBLCK
9. ENTRYP
10. ENTRYP
10. EXITP
9. FINBLK
9. DROPBLK
9. RELEASE
9. EXITP
8. RELEASE
7. KILPIT
8. GIMME
8. DELADD
7. STOP.
7. UNPACK
7. RELEASE
7. UNSTAT
7. FLPYF
7. UNPACK
7. RELEASE
7. FINBLK
7. TRACE
7. MESSAGE
7. RITEI
7. CLIST
7. FS0JMP
6. ALOI.
6. RANDOM.
6. HISTORY
7. ENTRYP
7. MESSAGE
7. WASKER
7. TAPES#
7. OUTCI.
7. EXITP
6. UNPACK
6. TAPES#
6. OUTCI.
6. GIMME
5. PACK
5. DELADD
6. DESTROY
7. TAPES#
7. OUTCI.
7. GETPTES
7. TERMAC
8. UNPACK
8. EXIT.
8. NMUCIT
9. ENTRYP
9. HEXADD
9. HEXMULT
10. ENTRYP
10. ITOJ.
10. EXITP
9. GETHFX
9. GIMME
9. ADBLCK
9. ENTRYP
10. ENTRYP
10. EXITP
9. FINBLK
9. DROPBLK
9. RELEASE
9. EXITP
8. RELEASE
7. KILPIT
8. GIMME
8. DELADD
7. STOP.
7. UNPACK
7. RELEASE
7. UNSTAT
7. FLPYF
7. UNPACK
7. RELEASE
7. FINBLK
7. TRACE
7. MESSAGE
7. RITEI
7. CLIST
7. FS0JMP
6. ALOI.
6. RANDOM.
9. DROPBLK (SEE LINE 00299)
9. PACK (SEE LINE 00277)
8. FINDBLK (SEE LINE 00207)
8. DROPBLK (SEE LINE 00299)
7. GINME (SEE LINE 00153)
7. DELADD (SEE LINE 00162)
7. SAMTYPE
6. UNPACK (SEE LINE 00199)
6. GINME (SEE LINE 00153)
6. DELADD (SEE LINE 00162)
5. GOTOER.
5. RELEASE (SEE LINE 00147)
5. BYEVP8S
5. SEEKP
5. UNPACK (SEE LINE 00189)
5. YANK
4. JNORM (SEE LINE 00169)
4. TAPESM
4. OUTCI.
4. TRACE (SEE LINE 00048)
4. OUTA
4. ENTRYP (SEE LINE 0011)
4. TAPESM
4. OUTCI.
4. EXTRP (SEE LINE 00033)
4. FSDUMP (SEE LINE 00313)
4. STOP
4. PACK (SEE LINE 00277)
3. RLESE (SEE LINE 00147)
3. GETPTRS (SEE LINE 00203)
3. CLOSSL
3. FINDBLK (SEE LINE 00207)
3. UNPACK (SEE LINE 00189)
3. DROPBLK (SEE LINE 00299)
3. DELADD (SEE LINE 00162)
2. BWASLLE
2. WESBLL3 (SEE LINE 00275)
2. DELADD (SEE LINE 00162)
2. YANK
2. REMOVT
1. UNPACK (SEE LINE 00399)
1. YANK
1. CANCALO
1. YANK (SEE LINE 00398)
1. RELEASE (SEE LINE 00147)
1. TAPESM
1. OUTCI.
1. READIL
1. UNPACK (SEE LINE 00168)
1. YANK (SEE LINE 0039)
1. STICK
1. UNPACK (SEE LINE 00168)
1. PACK (SEE LINE 00168)
1. TAPESM
1. OUTCI.
1. RELCAT
<table>
<thead>
<tr>
<th>Page 30</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Line</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>00493</td>
<td>8. <strong>UNPACK</strong> (SEE LINE 00189)</td>
</tr>
<tr>
<td>00494</td>
<td>9. <strong>RELEASE</strong> (SEE LINE 00147)</td>
</tr>
<tr>
<td>00495</td>
<td>9. <strong>DECLADD</strong> (SEE LINE 00152)</td>
</tr>
<tr>
<td>00496</td>
<td>9. <strong>FINDBLK</strong> (SEE LINE 00207)</td>
</tr>
<tr>
<td>00497</td>
<td>9. <strong>TRACE</strong> (SEE LINE 00048)</td>
</tr>
<tr>
<td>00498</td>
<td>9. <strong>MESSAGE</strong> (SEE LINE 00012)</td>
</tr>
<tr>
<td>00499</td>
<td>9. <strong>RITEI</strong> (SEE LINE 00015)</td>
</tr>
<tr>
<td>00500</td>
<td>9. <strong>CLIST</strong> (SEE LINE 00058)</td>
</tr>
<tr>
<td>00501</td>
<td>9. <strong>DROPBLK</strong> (SEE LINE 00299)</td>
</tr>
<tr>
<td>00502</td>
<td>7. <strong>UNLINK</strong></td>
</tr>
<tr>
<td>00503</td>
<td>8. <strong>UNPACK</strong> (SEE LINE 00189)</td>
</tr>
<tr>
<td>00504</td>
<td>8. <strong>RELEASE</strong> (SEE LINE 00147)</td>
</tr>
<tr>
<td>00505</td>
<td>7. <strong>UNSTAT</strong> (SEE LINE 00309)</td>
</tr>
<tr>
<td>00506</td>
<td>5. <strong>FINDBLK</strong> (SEE LINE 00207)</td>
</tr>
<tr>
<td>00507</td>
<td>5. <strong>IRK</strong></td>
</tr>
<tr>
<td>00508</td>
<td>5. <strong>RANDOM</strong></td>
</tr>
<tr>
<td>00509</td>
<td>5. <strong>YMBIND</strong></td>
</tr>
<tr>
<td>00510</td>
<td>6. <strong>UNPACK</strong> (SEE LINE 00189)</td>
</tr>
<tr>
<td>00511</td>
<td>6. <strong>GINWE</strong> (SEE LINE 00153)</td>
</tr>
<tr>
<td>00512</td>
<td>6. <strong>HEXADD</strong> (SEE LINE 00252)</td>
</tr>
<tr>
<td>00513</td>
<td>6. <strong>GETHEX</strong> (SEE LINE 00264)</td>
</tr>
<tr>
<td>00514</td>
<td>6. <strong>TAPEEN</strong></td>
</tr>
<tr>
<td>00515</td>
<td>6. <strong>OUTCI</strong></td>
</tr>
<tr>
<td>00516</td>
<td>6. <strong>DECLADD</strong> (SEE LINE 00162)</td>
</tr>
<tr>
<td>00517</td>
<td>6. <strong>SAMXYDE</strong> (SEE LINE 00389)</td>
</tr>
<tr>
<td>00518</td>
<td>6. <strong>TERMACQ</strong> (SEE LINE 00347)</td>
</tr>
<tr>
<td>00519</td>
<td>6. <strong>FINDBLK</strong> (SEE LINE 00207)</td>
</tr>
<tr>
<td>00520</td>
<td>6. <strong>RELEASE</strong> (SEE LINE 00147)</td>
</tr>
<tr>
<td>00521</td>
<td>6. <strong>DROPBLK</strong> (SEE LINE 00299)</td>
</tr>
<tr>
<td>00522</td>
<td>8. <strong>UNLINK</strong> (SEE LINE 00502)</td>
</tr>
<tr>
<td>00523</td>
<td>5. <strong>DROPBLK</strong> (SEE LINE 00299)</td>
</tr>
<tr>
<td>00524</td>
<td>5. <strong>HISTORY</strong> (SEE LINE 00310)</td>
</tr>
<tr>
<td>00525</td>
<td>5. <strong>DELADD</strong> (SEE LINE 00162)</td>
</tr>
<tr>
<td>00526</td>
<td>5. <strong>STATPAK</strong></td>
</tr>
<tr>
<td>00527</td>
<td>4. <strong>COMMON</strong></td>
</tr>
<tr>
<td>00528</td>
<td>4. <strong>UNPACK</strong> (SEE LINE 00189)</td>
</tr>
<tr>
<td>00529</td>
<td>4. <strong>DELADD</strong> (SEE LINE 00162)</td>
</tr>
<tr>
<td>00530</td>
<td>4. <strong>RELEASE</strong> (SEE LINE 00147)</td>
</tr>
<tr>
<td>00531</td>
<td>4. <strong>PACK</strong> (SEE LINE 00277)</td>
</tr>
<tr>
<td>00532</td>
<td>4. <strong>DOPTFTE</strong></td>
</tr>
<tr>
<td>00533</td>
<td>9. <strong>GETPTRS</strong> (SEE LINE 00203)</td>
</tr>
<tr>
<td>00534</td>
<td>9. <strong>UNSTAT</strong> (SEE LINE 00309)</td>
</tr>
<tr>
<td>00535</td>
<td>9. <strong>TAPEEN</strong></td>
</tr>
<tr>
<td>00536</td>
<td>9. <strong>OUTCI</strong></td>
</tr>
<tr>
<td>00537</td>
<td>9. <strong>HISTORY</strong> (SEE LINE 00310)</td>
</tr>
<tr>
<td>00538</td>
<td>9. <strong>FINDBLK</strong> (SEE LINE 00207)</td>
</tr>
<tr>
<td>00539</td>
<td>9. <strong>DROPBLK</strong> (SEE LINE 00299)</td>
</tr>
<tr>
<td>00540</td>
<td>9. <strong>DELADD</strong> (SEE LINE 00162)</td>
</tr>
<tr>
<td>00541</td>
<td>9. <strong>XPAA</strong></td>
</tr>
<tr>
<td>00542</td>
<td>9. <strong>RANDOM</strong></td>
</tr>
<tr>
<td>00543</td>
<td>9. <strong>GINWE</strong> (SEE LINE 00153)</td>
</tr>
<tr>
<td>00544</td>
<td>9. <strong>DESTROY</strong> (SEE LINE 00343)</td>
</tr>
<tr>
<td>00545</td>
<td>4. <strong>ENBASE</strong></td>
</tr>
<tr>
<td>00546</td>
<td>5. <strong>GETPTRS</strong> (SEE LINE 00203)</td>
</tr>
<tr>
<td>00547</td>
<td>5. <strong>TRYSHOT</strong></td>
</tr>
</tbody>
</table>
00659 7. ATAN2. (SEE LINE 00612)
00659 6. HEXCNZ (SEE LINE 00607)
00660 6. OPTPTH (SEE LINE 00147)
00662 6. DELADO (SEE LINE 00162)
00664 5. DELADD (SEE LINE 00162)
00665 5. STATPAK (SEE LINE 00153)
00666 5. GIMME (SEE LINE 00277)
00669 5. TAPE6#
00670 5. OUTCI.
00671 5. DESTROY (SEE LINE 00343)
00672 5. MAYBOR (SEE LINE 00189)
00673 5. WESAGE (SEE LINE 00012)
00674 5. RITEI (SEE LINE 00015)
00675 5. TRACE (SEE LINE 00048)
00676 5. CLIST (SEE LINE 00058)
00677 5. MEXDIST (SEE LINE 00153)
00679 5. OELADO (SEE LINE 00162)
00680 5. RELEASE (SEE LINE 00147)
00681 5. PERCEPT
00682 5. GETPTRS (SEE LINE 00203)
00683 5. ABSEE
00684 6. UNPACK (SEE LINE 00189)
00685 6. DELADO (SEE LINE 00147)
00686 6. RELEASE (SEE LINE 00147)
00687 6. TAPE6#
00688 6. OUTCI.
00689 5. CFLYCRC
00690 6. CRCSEE
00691 7. RELEASE (SEE LINE 00147)
00692 7. GIMME (SEE LINE 00153)
00692 7. UNPACK (SEE LINE 00189)
00694 7. DELADO (SEE LINE 00162)
00695 7. CRCVEVT
00696 8. GIMME (SEE LINE 00153)
00697 8. TAPE6#
00698 8. OUTCI.
00699 8. DELADO (SEE LINE 00162)
00700 8. RELEASE (SEE LINE 00147)
00701 8. FINDBLK (SEE LINE 00267)
00702 8. DETECT
00703 9. MEXDIST
00704 9. UNPACK (SEE LINE 00189)
00705 9. TME2PS (SEE LINE 00226)
00706 9. ATAN2.
00707 9. LORAD#
00708 10. ISHIFT
00709 10. OPTPTH (SEE LINE 00607)
00710 10. THM2PS (SEE LINE 00226)
00711 10. SORT.
00712 10. COS.
7. DETECT (SEE LINE 00702)
7. BYTKCMK
8. SRT.
9. ATNL5
10. CJSRAQ (SEE LINE 00707)
11. DELAO (SEE LINE 00182)
12. YANK (SEE LINE 00398)
13. STICK (SEE LINE 00432)
14. T3A0TL (SEE LINE 00454)
7. NEWPERC
8. HISTORY (SEE LINE 00330)
9. GIMME (SEE LINE 00153)
10. STICK (SEE LINE 00432)
11. DELAO (SEE LINE 00152)
7. GETPTFS (SEE LINE 00203)
7. CRCTRAK (SEE LINE 00724)
7. BMRUP
8. SEEK:
9. UNPACK (SEE LINE 00189)
10. GIMME (SEE LINE 00153)
11. PACK (SEE LINE 00277)
12. NEWPERC (SEE LINE 00322)
13. GETPTFS (SEE LINE 00203)
14. CRCTRAK (SEE LINE 00724)
7. BRYENPS (SEE LINE 00395)
7. CRCLOJS (SEE LINE 00413)
7. DROPOS
8. YANK (SEE LINE 00398)
9. RELEASE (SEE LINE 00147)
10. WESBILO (SEE LINE 00275)
11. DELAO (SEE LINE 00162)
8. SEEKING
9. DROPSZ
10. YANK (SEE LINE 00398)
11. RELEASE (SEE LINE 00147)
12. WESBILO (SEE LINE 00275)
13. DELAO (SEE LINE 00162)
14. SEEKENG (SEE LINE 00454)
15. GIMME (SEE LINE 00153)
16. BLYACT
17. UNPACK (SEE LINE 00189)
18. PACK (SEE LINE 00277)
19. STICK (SEE LINE 00432)
20. GIMME (SEE LINE 00153)
21. DELAO (SEE LINE 00162)
22. WESBILO (SEE LINE 00275)
23. UNPACK (SEE LINE 00189)
24. PACK (SEE LINE 00277)
25. STICK (SEE LINE 00432)
26. GIMME (SEE LINE 00153)
27. ALLOBAT
28. PRIOR
29. WESBILO (SEE LINE 00275)
30. GIMME (SEE LINE 00153)
7. CLOSCOR  (SEE LINE 00915)
7. JGECIT   (SEE LINE 00011)
7. T3J2XY   (SEE LINE 00229)
7. ATAYZ   (SEE LINE 00033)
7. EXIT3   (SEE LINE 00011)
7. FORMAT3 (SEE LINE 00011)
7. JGECIT   (SEE LINE 00153)
7. FINDBLK  (SEE LINE 00207)
7. FINDFLT  (SEE LINE 00011)
7. HEXDIST  (SEE LINE 00207)
7. L3KARE   (SEE LINE 00925)
7. G4ME     (SEE LINE 00153)
7. 4303L04  (SEE LINE 00359)
7. EXITP    (SEE LINE 00033)
7. PELASE   (SEE LINE 00147)
7. EXIT3    (SEE LINE 00033)
7. G4ME     (SEE LINE 00153)
7. EXIT3    (SEE LINE 00033)
7. TGTG34E  (SEE LINE 00011)
7. E4TRYP   (SEE LINE 00011)
7. EXIT3    (SEE LINE 00033)
7. PT3E     (SEE LINE 00957)
7. RAND34   (SEE LINE 00957)
7. PELADJ   (SEE LINE 00957)
7. EXIT3    (SEE LINE 00033)
7. EXIT3    (SEE LINE 00033)
7. IT3RAP   (SEE LINE 00028)
7. 4VAIL3   (SEE LINE 00028)
7. RELA3D   (SEE LINE 00966)
7. RELEASE  (SEE LINE 00147)
7. SCHEDUL  (SEE LINE 00189)
7. UNPACK   (SEE LINE 00189)
7. RANDOY3 (SEE LINE 00189)
7. RENDE4U  (SEE LINE 00189)
7. UNPACK   (SEE LINE 00229)
7. 4TX2XY   (SEE LINE 00229)
7. 4TX2XY   (SEE LINE 00242)
7. MEXAD3   (SEE LINE 00252)
7. G3TH3X   (SEE LINE 00264)
7. HEXDIST  (SEE LINE 00264)
7. AEFAS3   (SEE LINE 00264)
7. G4ME     (SEE LINE 00153)
7. G4ME     (SEE LINE 00153)
01153 9. DJTCI
01154 9. RELEASE (SEE LINE 00147)
01155 9. DILOUT (SEE LINE 00480)
01156 8. BATTOUT
01157 9. YANK (SEE LINE 00398)
01158 9. BVLALLE (SEE LINE 00418)
01159 9. UNPACK (SEE LINE 00189)
01160 9. CHKLAST
01161 10. DROPPS (SEE LINE 00489)
01162 10. PRIORITY
01163 10. RELD0T (SEE LINE 00437)
01164 10. DELADD (SEE LINE 00162)
01165 9. GIWME (SEE LINE 00153)
01166 9. RELEASE (SEE LINE 00147)
01167 9. S3LL (SEE LINE 00868)
01168 8. D0VARY
01169 9. SEEKTAC
01170 10. DROPPS (SEE LINE 00849)
01171 10. UNPACK (SEE LINE 00189)
01172 10. GIWME (SEE LINE 00153)
01173 10. PACK (SEE LINE 00277)
01174 10. RELEASE (SEE LINE 00147)
01175 10. SSL (SEE LINE 00868)
01176 10. DELADD (SEE LINE 00162)
01177 10. ALLOBAT (SEE LINE 00874)
01178 10. YANK (SEE LINE 00399)
01179 10. READIL
01180 9. GIWME (SEE LINE 00153)
01181 9. DLYAFT (SEE LINE 00862)
01182 9. RELEASE (SEE LINE 00147)
01183 8. GIWME (SEE LINE 00153)
01184 8. PACK (SEE LINE 00277)
01185 8. SSL (SEE LINE 00868)
01186 8. OEERALO
01187 9. WESBILD (SEE LINE 00275)
01188 9. DELADD (SEE LINE 00162)
01189 9. TAPIEO
01190 9. DJTCI
01191 8. SEEKTAC (SEE LINE 00169)
01192 8. DLYACT (SEE LINE 00862)
01193 8. RELEASE (SEE LINE 00147)
01194 6. I1MPOWER
01195 7. G0TDER
01196 7. SKEPBAX
01197 8. UNPACK
01198 8. SEEK
01199 7. DROPPS (SEE LINE 00849)
01200 7. BVLALLE (SEE LINE 00418)
01201 7. UNPACK (SEE LINE 00189)
01202 7. CHKLAST (SEE LINE 00160)
01203 7. SEEKTAC (SEE LINE 00169)
01204 7. GIWME (SEE LINE 00153)
01205 7. PACK (SEE LINE 00277)
01206 7. DLYACT (SEE LINE 00862)
01207 7. SEEKENG (SEE LINE 00854)
7. BNRREV
8. UNPACK (SEE LINE 00189)
8. UNPACK
9. UNPACK (SEE LINE 00189)
9. GIMME (SEE LINE 00153)
9. PACK (SEE LINE 00277)
9. SELL (SEE LINE 00868)
9. DECRALD (SEE LINE 00186)
9. SEEKEND (SEE LINE 00854)
9. OLYACT (SEE LINE 00862)
9. RELEASE (SEE LINE 00187)
8. PACK (SEE LINE 00277)
7. SEEKP
7. TAPESK
7. OUTCI.
7. RELEASE (SEE LINE 00147)
7. YANK (SEE LINE 00398)
7. DROPOSS (SEE LINE 00849)
7. DILJU (SEE LINE 00860)
7. MESSBLO (SEE LINE 00275)
7. DELADD (SEE LINE 00162)
6. RELEASE (SEE LINE 00147)
6. SOBTEST
5. UNPACK (SEE LINE 00189)
5. GIMME (SEE LINE 00153)
5. PACK (SEE LINE 00277)
5. TAPESK
5. OUTCI.
5. RANDOM
5. SYNTRK
4. CRYCDW
2. UNPACK (SEE LINE 00189)
2. INVANGE (SEE LINE 00115)
2. GIMME (SEE LINE 00153)
2. TAPESK
2. OUTCI.
1. OUTA (SEE LINE 00403)
1. DELADD (SEE LINE 00162)
1. AVLaLE (SEE LINE 00418)
1. SETASSV (SEE LINE 00131)
1. SEEFTAC (SEE LINE 00169)
1. GIMME (SEE LINE 00153)
1. OLYACT (SEE LINE 00862)
7. SYNTRK
6. MESSBLO (SEE LINE 00275)
5. DELADD (SEE LINE 00182)
4. AVLaLE (SEE LINE 00418)
3. INVANGE (SEE LINE 00115)
2. PREPAF
1. UNPACK (SEE LINE 00189)
1. PACK (SEE LINE 00277)
1. AJHJARI
1. STICK (SEE LINE 00432)
9. RELOCAT (SEE LINE 00437)
01538  6. WIPEOUT (SEE LINE 00153)
01539  6. GIMME (SEE LINE 00153)
01540  6. ADDBLOK (SEE LINE 00359)
01541  6. PELADD (SEE LINE 00966)
01542  6. DESTROY (SEE LINE 00343)
01543  6. GIMME (SEE LINE 00153)
01544  6. GOGETM (SEE LINE 00189)
01545  6. UNPACK (SEE LINE 00189)
01546  6. FINDBLK (SEE LINE 00207)
01547  6. CRFLTM (SEE LINE 00989)
01548  6. TAPE6# (SEE LINE 00207)
01549  6. OUTCI. (SEE LINE 00207)
01550  6. GIMME (SEE LINE 00153)
01551  6. ADDBLOK (SEE LINE 00359)
01552  6. DESTROY (SEE LINE 00343)
01553  7. UNPACK (SEE LINE 00189)
01554  7. RAND34 (SEE LINE 00252)
01555  7. HEXA3D (SEE LINE 00353)
01556  7. HEXMLT (SEE LINE 00353)
01557  7. GETHEX (SEE LINE 00264)
01558  7. PACK (SEE LINE 00277)
01559  7. UNPACK (SEE LINE 00277)
01560  7. ULOAD (SEE LINE 00277)
01561  7. INITACQ (SEE LINE 00162)
01562  7. DELADD (SEE LINE 00162)
01563  7. MEBILDO (SEE LINE 00275)
01564  7. MEBILDO (SEE LINE 00275)
01565  7. DESTROY (SEE LINE 00343)
01566  7. UNPACK (SEE LINE 00189)
01567  7. UNPACK (SEE LINE 00189)
01568  7. UNPACK (SEE LINE 00189)
01569  7. UNPACK (SEE LINE 00189)
01570  7. UNPACK (SEE LINE 00189)
01571  7. UNPACK (SEE LINE 00189)
01572  7. UNPACK (SEE LINE 00189)
01573  7. UNPACK (SEE LINE 00189)
01574  7. UNPACK (SEE LINE 00189)
01575  7. UNPACK (SEE LINE 00189)
01576  7. UNPACK (SEE LINE 00189)
01577  7. UNPACK (SEE LINE 00189)
01578  7. UNPACK (SEE LINE 00189)
01579  7. UNPACK (SEE LINE 00189)
01580  7. UNPACK (SEE LINE 00189)
01581  7. UNPACK (SEE LINE 00189)
01582  7. UNPACK (SEE LINE 00189)
01583  7. UNPACK (SEE LINE 00189)
01584  7. UNPACK (SEE LINE 00189)
01585  7. UNPACK (SEE LINE 00189)
01586  7. UNPACK (SEE LINE 00189)
01587  7. UNPACK (SEE LINE 00189)
01588  7. UNPACK (SEE LINE 00189)
01589  7. UNPACK (SEE LINE 00189)
01590  7. UNPACK (SEE LINE 00189)
01591  7. UNPACK (SEE LINE 00189)
01592  7. UNPACK (SEE LINE 00189)

01538  6. WIPEOUT (SEE LINE 00153)
01539  6. GIMME (SEE LINE 00153)
01540  6. ADDBLOK (SEE LINE 00359)
01541  6. PELADD (SEE LINE 00966)
01542  6. DESTROY (SEE LINE 00343)
01543  6. GIMME (SEE LINE 00153)
01544  6. GOGETM (SEE LINE 00189)
01545  6. UNPACK (SEE LINE 00189)
01546  6. FINDBLK (SEE LINE 00207)
01547  6. CRFLTM (SEE LINE 00989)
01548  6. TAPE6# (SEE LINE 00207)
01549  6. OUTCI. (SEE LINE 00207)
01550  6. GIMME (SEE LINE 00153)
01551  6. ADDBLOK (SEE LINE 00359)
01552  6. DESTROY (SEE LINE 00343)
01553  7. UNPACK (SEE LINE 00189)
01554  7. RAND34 (SEE LINE 00252)
01555  7. HEXA3D (SEE LINE 00353)
01556  7. HEXMLT (SEE LINE 00353)
01557  7. GETHEX (SEE LINE 00264)
01558  7. PACK (SEE LINE 00277)
01559  7. UNPACK (SEE LINE 00277)
01560  7. ULOAD (SEE LINE 00277)
01561  7. INITACQ (SEE LINE 00162)
01562  7. DELADD (SEE LINE 00162)
01563  7. MEBILDO (SEE LINE 00275)
01564  7. MEBILDO (SEE LINE 00275)
01565  7. DESTROY (SEE LINE 00343)
01566  7. UNPACK (SEE LINE 00189)
01567  7. UNPACK (SEE LINE 00189)
01568  7. UNPACK (SEE LINE 00189)
01569  7. UNPACK (SEE LINE 00189)
01570  7. UNPACK (SEE LINE 00189)
01571  7. UNPACK (SEE LINE 00189)
01572  7. UNPACK (SEE LINE 00189)
01573  7. UNPACK (SEE LINE 00189)
01574  7. UNPACK (SEE LINE 00189)
01575  7. UNPACK (SEE LINE 00189)
01576  7. UNPACK (SEE LINE 00189)
01577  7. UNPACK (SEE LINE 00189)
01578  7. UNPACK (SEE LINE 00189)
01579  7. UNPACK (SEE LINE 00189)
01580  7. UNPACK (SEE LINE 00189)
01581  7. UNPACK (SEE LINE 00189)
01582  7. UNPACK (SEE LINE 00189)
01583  7. UNPACK (SEE LINE 00189)
01584  7. UNPACK (SEE LINE 00189)
01585  7. UNPACK (SEE LINE 00189)
01586  7. UNPACK (SEE LINE 00189)
01587  7. UNPACK (SEE LINE 00189)
01588  7. UNPACK (SEE LINE 00189)
01589  7. UNPACK (SEE LINE 00189)
01590  7. UNPACK (SEE LINE 00189)
01591  7. UNPACK (SEE LINE 00189)
01592  7. UNPACK (SEE LINE 00189)
3. Post Processor

LIST OF SUBROUTINES - RECORD

1. EOF
2. INDEX
3. MESSAGE
4. PAGE
5. RECORD
6. RITEI
7. RITER
8. TABOUT
LIST OF FORTRAN LIBRARY ROUTINES - RECORD

1. INPCI.
2. INPFI.
3. OUTCI.
4. OUTCI2.
5. QINTER.
6. STOP.
7. TAPE6.
SUBROUTINE REFERENCE LIST - RECORD

1. EOF
   CALLED BY: RECORD

2. INDEX
   CALLS:
      TAPE#
      OUTCI.
      CALLED BY: RECORD

3. MESSAGE
   CALLS:
      TAPE#
      OUTCI.
      CALLED BY: MESSAGE, OUTCI.

4. PAGE
   CALLS:
      TAPE#
      OUTCI.
      CALLED BY: PAGE

5. RECORD
   CALLS:
      QENTRY.
      INPCLI.
      INPFI.
      PAGE
      EOF
      INDEX
      MESSAGE
      RITEI
      RITER
      TABOUT
      STOP.
      CALLED BY: RECORD

6. RITEI
   CALLS:
      TAPE#
      OUTCI.
      CALLED BY: RECORD

7. RITER
   CALLS:
      TAPE#
      OUTCI.
      CALLED BY: RECORD

8. TABOUT
   CALLS:
      PAGE
      MESSAGE
      TAPE#
      OUTCI.
      OUTCR.
      CALLED BY: RECORD
<table>
<thead>
<tr>
<th>FORTRAN LIBRARY REFERENCE LIST - RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INPCI.</td>
</tr>
<tr>
<td>2. INPFI.</td>
</tr>
<tr>
<td>3. OUTCI.</td>
</tr>
<tr>
<td>4. OUTCR.</td>
</tr>
<tr>
<td>5. JNTRY.</td>
</tr>
<tr>
<td>6. STOP.</td>
</tr>
<tr>
<td>7. TAPE6#</td>
</tr>
</tbody>
</table>

467
RECORD SUBROUTINE CALLING HIERARCHY -

00001  1. RECORD
00002    2. INPUT.
00003    2. INPUT.
00004    2. INPUT.
00005    2. PAGE
00006      3. TAPE5.
00007      3. OUTCI.
00008      2. OUTCI.
00009      2. EOF
00010      2. INDEX
00011      3. TAPE5.
00012      3. OUTCI.
00013      2. MESSAGE
00014        3. TAPE5.
00015        3. OUTCI.
00016        2. RITE1
00017        3. TAPE5.
00018        3. OUTCI.
00019        2. RITE2
00020        3. TAPE5.
00021        3. OUTCI.
00022        2. TABOUT
00023        3. PAGE       (SEP LINE 00005)
00024        3. MESSAGE     (SEP LINE 00013)
00025        3. TAPE5.
00026        3. OUTCI.
00027        3. OUTCI.
00028        2. STOP.
APPENDIX I

MADEM DEBUG ROUTINES

This appendix contains an alphabetical list and description of the debug routines available in MADEM. Following this list is a more detailed description of how to implement certain of these capabilities.

ADUMP - Dumps to the printer any array in octal.

Parameters:
- IARRAY - array to be dumped
- ISTART - offset of first word to be dumped
- LENGTH - number of words to be dumped
- INAME - hollerith name of the array

ANALYZE - A separate program that analyzes a binary file and gives the following information:
- Cross reference of all routine calls
- Complete calling hierarchy

BLOCK DATA ROUTINES - 41 routines, each representing a data block, that can be manipulated to create data structure display routines. Each block routine is capable of printing data blocks or a list of data blocks, and of calling other block routines to print subordinate lists.

Parameters:
- POINTER - pointer to first data block
- LEVEL - number of hierarchical types of blocks to print in the data structure
- MAXBLKS - number of blocks in the main list to be printed.

CLIST (and CLIST2) - Common block dump routine. Prints key common pointer values.
DBGREAD - Debug parameter read routine. Reads all input parameters used for debug purposes. These parameters are interpreted and the necessary flags are set to activate these parameters. Invalid parameters are ignored.

DISPLAT - Display DATFILE data structure. Uses 13 of the block data routines. Activated by a debug parameter.

ENTRYP - Entry debug routine. Called at the beginning of most MADEM subroutines. Keeps track of the calling hierarchy in the pushdown stack and of the last 50 routines called in the circular list. Counts the number of times each routine calls ENTRYD and, along with EXITP, times the execution of these routines. Optionally, ENTRYP can call debug routines ITRAP and ICHEC. Also, can optionally print specified routine trace messages. These options are set through debug parameters.

Parameter:

    SEGNUM - segment number of calling routine

ENTSTAT - Prints vector of routine entry counts, execution times, trace message flags, and debug call flags.

EXITP - Same as ENTRYP, but called at the end of a routine rather than at the beginning.

Parameter:

    SEGNUM - segment number of calling routine

HALT - Used whenever the simulation is to be stopped. Performs the following functions, mostly through subroutine calls:

    • creates hold files for restarts
    • prints name of calling subroutine
    • prints reason for termination.
    • prints pushdown stack
    • prints names of last 50 routines called
    • calls ENTRYP (see ENTRYP)
• calls CLIST (see CLIST)
• optionally (controlled by debug parameter) prints ISPACE.
• Stops the simulation.

Parameters:
  SEGNUM - segment number of calling routine
  MSGHALT - forty character message

ICHECK - Checks specified locations in ISPACE, and prints a message when the value of that location changes. The message indicates the old value, the new value, and the last non-debug routine called. The ISPACE locations are selected through debug parameters read by DBGREAD. DEBUG must be set to "ON" for ICHECK to be called from any given routine (see DBGREAD).

Parameter:
  SEGNUM - segment number of calling routine

ICOMP - A separate program that compares two sets of hold files and indicates when there are differences in the ISPACEs. Used when midasizing, to insure the midasized version runs the same as the unmidasized version.

Parameters:
  ISTART - first word of ISPACE to be compared
  IMAX - last word of ISPACE to be compared
  MAXDIFF - maximum number of mismatches before stopping
  IEXTRA - maximum number of extra words printed if one ISPACE is larger than the other.
  HSIZE1 - number of ISPACE words in each hold file for second ISPACE.
ISDUMP - Dumps to printer selected portions or all of the ISPACE array. Will always print at least the first ten words.

Parameters:

ISTART - first word to be dumped
LENGTH - number of words to be dumped

ITRAP - Checks ISPACE of zero and ISPACE locations one through eight for proper values. ISPACE of zero should always be equal to 99999999.0, and other eight locations should always be zero. If any of these locations have improper values, then HALT is called to stop the simulation and print debug information. DEBUG must be set to "ON" for ITRAP to be called from ENTRYP and EXITP (see DBGRAD).

Parameter:

SEGNUM - segment number of calling routine, or of routine that called ENTRYP/EXIT.

LOCATE - A batch text search program that can be used as a MIDAS cross reference program.

NIPULSTOR - A post run debugging facility that can:
1. Print selected areas of ISPACE.
2. Dump the C2 data structures.
3. Dump the EVENT tree.
4. Dump structures of a given unit.
5. Call CLIST.

See the detailed description at the end of this appendix.

RECCON - Activated only by RECUR in the event of abnormal job termination and subsequent recovery. Calls HALT to print debug information and stop the simulation.

RECER - Prints current calling hierarchy (pushdown stack) and the names of the last 50 subroutines called (circular list).
RECOUR - Allows the MADEM program to regain control of execution at the time that abnormal job termination would otherwise occur. RECOUR calls RECCON in the event of catastrophic failure. RECOUR is automatically initialized at the beginning of the MADEM Program. RECOUR may be turned off by using debug parameter "RECOUR=OFF."

Parameters:

NAME - name of the routine to be executed if flagged conditions occur (RECCON)

FLAGS - octal value of error conditions that trip RECOUR (077)

CHECKSUM - No checksum desired (0)

ROUTER - Same as RECER, but with SEGNUM as a parameter, so the calling routine name can be printed.

MADEM DEBUG PARAMETERS

The INPUT file to MADEM holds various parameters that affect only the particular volume that is being run. The first input card holds seven numbers described under MADEM Operations. This card is mandatory. Debug parameters follow this first card, and are entirely optional. There are a variety of debug options which may be turned on or off by using debug parameters. These options and their corresponding parameters are listed below. Parameters, except where noted otherwise, must begin in column one. All parameters are actually ten characters long, with either leading or trailing blanks implied.

The debug options:

1. Debug Status
   For each routine that calls ENTRYP and EXITP, the debug status is set to either "ON" or "OFF". When debug is "ON" for a routine, then ENTRYP and EXITP have the addition of calling ITRAP and ICHECK when processing
that routine. This slows down execution considerably when many routines
are "ON," but is a valuable debug tool.

DEFAULT: Debug is "OFF" for all routines.

PARAMETERS:

"DEBUG=ON" - turns delay to "ON" for all routines
"DXXXXXXX" - where XXXXXXX is a routine name.
Changes the debug status of that routine only. If it was "ON," it is set to
"OFF," and vice-versa. By using "DEBUG-
ON" followed by a few occurrences of
this parameter, all routines but a few
can be set to "ON." Likewise, by only
using this parameter, only a few rou-
tines can be set to "ON."

2. Trace Status

The same as debug status, except that what is being turned "ON"
and "OFF" is the printing of subroutine call trace messages.

DEFAULT: Trace is "OFF" for all routines.

PARAMETERS:

"TRACE=ON" - turns on trace messages for all
routines.
"TXXXXXXX" - same as "DXXXXXXX" but for the trace
status.

Never use "TRACE=ON" by itself. The resulting output will be thousands of
pages of trace messages.

3. Recovery Status

Controls the initializing of system recover routine RECUR. When
recover is "ON," then RECUR will be initiated upon abnormal termination.
When Recover is "OFF," no calls will be made to RECUR.

DEFAULT: Recover is set to "ON"

PARAMETER:

"RECUR=OFF" - turns off recovery routine.
4. **Icheck Status**

Controls ISPACE locations checked by debug routine ICHECK. To have ISPACE locations checked, the word "CHECK" must start in column 1, followed on the same card by up to seven decimal ISPACE pointer right justified ending in columns 10x, where x = 2 thru 8. All check cards together may not have more than ten ISPACE locations.

DEFAULT: no ISPACE locations are checked.

5. **Release Status**

The allocation of blocks in ISPACE is controlled by two routines: GIMME and RELEASE. Release un-allocates previously used storage for future use. Sometimes it is advantageous to turn off release so that all new blocks will be allocated at the end of "used" ISPACE (free space). This is done by turning the release status off. It also must be indicated after which event within the volume that release is turned off. To turn release off, put "RELEASEXXX" beginning in column one, followed on the same card by a decimal number ending in column 20. This number is the event after which release is turned off.

DEFAULT: Release is never turned off.

6. **Stop Status** (Pre-processor only)

The Stop Status controls how far the preprocessor runs before stopping. There is no restart capability for the preprocessor; the early stops are for debugging purposes only.

DEFAULT: Pre-processor runs to normal completion.

PARAMETERS:

"STOP=ODATH" - stops after reading DATFILE.
"STOP=UOILW" - stops after reading UOIL.
"STOP=DEL" - stops after planning first event.

7. **Datfile Display** (Pre-processor only)

To get DISPDAT to print the DATFILE data structure, use parameter "DATFILE=ONW".

475
DEFAULT: Datfile is not displayed
PARAMETER:
"DATFLE=ONV"

8. **Ispace Dump**

DEFAULT: only first ten words of ISPACE are dumped
PARAMETER:
"DUMP=ONV" - all of ISPACE dumped.

**NIPUL8TOR**

NIPUL8TOR is a fortran program that was written to use as a debugging tool for MADEM. NIPUL8TOR can dump selected areas of ISPACE, selected data structures, or MADEM's common blocks.

At the end of each MADEM run (or volume of a run), MADEM saves ISPACE and all other common blocks in a series of files. NIPUL8TOR gets ISPACE and the common blocks from this series of files. To maintain compatibility, NIPUL8TOR uses MADEM's fetch routine to retrieve the data from these files. NIPUL8TOR also uses MADEM's CLIST subroutine, as well as the subroutines that CLIST calls.

**USING NIPUL8TOR**

To use NIPUL8TOR, you need:
1) The binary file "NIPUL8TOR"
2) The series of MADEM files that holds ISPACE and the common blocks
3) The correct set of NIPUL8TOR directives that tell NIPUL8TOR exactly which dumps you want.

The sample JCL deck on the next page shows the input that will give you an end of Volume 3 dump that exercises all the NIPUL8TOR options. The first four attach commands access the MADEM files that were dumped by a volume 3 run. The NIPUL8TOR directives (commands) and directive parameters are also shown.

476
WBDMBSM,ST176,T200,10177,P60,EC400. MADEM NIPUL8TOR RUN ACCOUNT BSMBUM, WDNA14V6-SGC,BDM,703-821-4223. B MACALLER
COMMENT.
COMMENT. ***********************************************
COMMENT. * VOLUME 1, RUN TYPE AAA *
COMMENT. ***********************************************
ATTACH,TAPE15,PLAN1AAA,ID=WDNA14V6.
ATTACH,TAPE16,PLAN2AAA,ID=WDNA14V6.
ATTACH,LGO,NIPUL8TOR,ID=WDNA14V6.
LDSET,PRESET=ZERO.
LOAD,LGO.
EXECUTE,,PL=50000.
AUDIT,ID=WDNA14V6.
EXIT.
AUDIT,ID=WDA14V6.
& EOR
1 DUMP SELECTED AREA OF ISPACE
1000,2000
2 DUMP AROUND A WORD OF ISAPCE
1500,500
3 DUMP A C2 STRUCTURE
PTR=002196
3 DUMP THE BLUE C2 TREE
BLUE
3 DUMP THE RED C2 TREE
RED
4 DUMP LEFTIST TREE AND EVENT NODE INFO
PTR=012345
4 DUMP THE DEL AND EVENT INFO
DEL
6 DUMP STRUCTURES OF A GIVEN UNIT
14225
7 DUMP COMMON BLOCKS
7 # E0I 477
DIRECTIVES FOR NIPUL8TOR

Directives, submitted through an input deck, tell NIPUL8TOR exactly what to dump. Depending on the directive, there may be zero, one, or two parameters on the input card following the directive. The directive appears alone on a card. The directive is a digit (1-7), and directive parameters are either decimal integers or commands, as indicated below. There is no limit to the number of directives used in the SIPUL8TOR run, or to the number of repetitions of any one directive. The directives are not order dependent.

**DIRECTIVE 1:** Dump a selected area of ISPACE. This directive will dump a chunk of ISPACE, as defined by the two parameters.

**PARAMETERS**
1) First word of ISPACE to be dumped.
2) Last word of ISPACE to be dumped.

Obviously, the first PARM must be less than or equal to the second PARM.

**DIRECTIVE 2:** Dump around a word of ISAPCE. This directive will also dump a chunk of ISPACE as defined by the two parameters, but the PARMS have different meanings.

**PARAMETERS**
1) Middle word of ISPACE area to be dumped.
2) Number of words on either side of the middle word, to be dumped.

**DIRECTIVE 3:** Dump the C2 structure CTREED. This directive will display a C2 tree as defined by the parameter. It may be used to dump the red or blue C2 trees, or any subset of a C2 tree. For each unit in the C2 tree, the SB, SDB, C2, and unit status board blocks will be displayed.
PARAMETERS

3 options that define the C2 tree:
1) RED - dumps the red C2 tree
2) BLUE - dumps the blue C2 tree
3) PTR=IIIIII, where IIIIIII = pointer to buffer at top of C2 tree. (IIIIII is a right justified 6 digit integer).

To dump a subset of a tree, use PTR=XXXXXX, where XXXXXX = PTRC2 + 1, where PTRC2 points to unit above the top unit in the subset. (i.e, PJRC2 + 1 simulates the buffer).

DIRECTIVE 4: Dump leftist tree and event node info. This directive will dump the leftist tree and corresponding event blocks.
PARAMETERS
One PARM, defines leftist tree.
2 options:
1) DEL - dumps discrete event list
2) PTR = IIIIIII, where IIIIIII is a right justified 6 digit integer that points to the top node in the tree.

DIRECTIVE 5: Not used. Future plans call for a hex structure dump.

DIRECTIVE 6: Dump structures of a given unit. This directive can be used to dump the blocks of a given unit, where the unit is displayed as in the C2 dump.
PARAMETER
Decimal integer pointer to the C2 unit to be dumped.

DIRECTIVE 7: Dump the common blocks. This directive will display most of MADEM's common blocks.
No PARAMETERS, do not use a second card.
APPENDIX J
MADEM DATA STRUCTURE CROSS REFERENCE

<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
<th>Column 5</th>
<th>Column 6</th>
<th>Column 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHART</td>
<td>TREE</td>
<td>TRAP</td>
<td>TRAP</td>
<td>TRAP</td>
<td>TRAP</td>
<td>TRAP</td>
</tr>
<tr>
<td>HTML</td>
<td>HTML</td>
<td>HTML</td>
<td>HTML</td>
<td>HTML</td>
<td>HTML</td>
<td>HTML</td>
</tr>
<tr>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
</tr>
<tr>
<td>SQL</td>
<td>SQL</td>
<td>SQL</td>
<td>SQL</td>
<td>SQL</td>
<td>SQL</td>
<td>SQL</td>
</tr>
<tr>
<td>XML</td>
<td>XML</td>
<td>XML</td>
<td>XML</td>
<td>XML</td>
<td>XML</td>
<td>XML</td>
</tr>
<tr>
<td>JSON</td>
<td>JSON</td>
<td>JSON</td>
<td>JSON</td>
<td>JSON</td>
<td>JSON</td>
<td>JSON</td>
</tr>
<tr>
<td>CSS</td>
<td>CSS</td>
<td>CSS</td>
<td>CSS</td>
<td>CSS</td>
<td>CSS</td>
<td>CSS</td>
</tr>
<tr>
<td>SVG</td>
<td>SVG</td>
<td>SVG</td>
<td>SVG</td>
<td>SVG</td>
<td>SVG</td>
<td>SVG</td>
</tr>
<tr>
<td>HTML5</td>
<td>HTML5</td>
<td>HTML5</td>
<td>HTML5</td>
<td>HTML5</td>
<td>HTML5</td>
<td>HTML5</td>
</tr>
<tr>
<td>CSS3</td>
<td>CSS3</td>
<td>CSS3</td>
<td>CSS3</td>
<td>CSS3</td>
<td>CSS3</td>
<td>CSS3</td>
</tr>
<tr>
<td>XML1</td>
<td>XML1</td>
<td>XML1</td>
<td>XML1</td>
<td>XML1</td>
<td>XML1</td>
<td>XML1</td>
</tr>
<tr>
<td>JSON1</td>
<td>JSON1</td>
<td>JSON1</td>
<td>JSON1</td>
<td>JSON1</td>
<td>JSON1</td>
<td>JSON1</td>
</tr>
<tr>
<td>CSS2</td>
<td>CSS2</td>
<td>CSS2</td>
<td>CSS2</td>
<td>CSS2</td>
<td>CSS2</td>
<td>CSS2</td>
</tr>
<tr>
<td>SVG1</td>
<td>SVG1</td>
<td>SVG1</td>
<td>SVG1</td>
<td>SVG1</td>
<td>SVG1</td>
<td>SVG1</td>
</tr>
<tr>
<td>HTML6</td>
<td>HTML6</td>
<td>HTML6</td>
<td>HTML6</td>
<td>HTML6</td>
<td>HTML6</td>
<td>HTML6</td>
</tr>
<tr>
<td>CSS4</td>
<td>CSS4</td>
<td>CSS4</td>
<td>CSS4</td>
<td>CSS4</td>
<td>CSS4</td>
<td>CSS4</td>
</tr>
<tr>
<td>XML2</td>
<td>XML2</td>
<td>XML2</td>
<td>XML2</td>
<td>XML2</td>
<td>XML2</td>
<td>XML2</td>
</tr>
<tr>
<td>JSON2</td>
<td>JSON2</td>
<td>JSON2</td>
<td>JSON2</td>
<td>JSON2</td>
<td>JSON2</td>
<td>JSON2</td>
</tr>
</tbody>
</table>

(continued on next page)
<table>
<thead>
<tr>
<th></th>
<th>ANX</th>
<th>ASPRO</th>
<th>ARGQ</th>
<th>CDRK</th>
<th>COUT</th>
<th>CPMIR</th>
<th>COMSES</th>
<th>CIRC80</th>
<th>CIRK80</th>
<th>CONTR</th>
<th>CRTAD</th>
<th>F87</th>
<th>F87</th>
<th>F87</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
<th>GAT1</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYAICOV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYCMOPR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYCONRD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYCONIS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYCONIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYENDPS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYHEDOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYNORMD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYNTRK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYPASUP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYPONER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYPONEO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYPONRI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYPONRS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYPONIM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYKCIK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYKOA1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAGNOI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAGNOI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CILESIE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHRISTI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
|          | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBUS | ADBU
<table>
<thead>
<tr>
<th>CLOS COR</th>
<th>CNACT1K</th>
<th>CODE01</th>
<th>CODE03</th>
<th>CODE05</th>
<th>CODE18</th>
<th>COMMAND</th>
<th>COMM0</th>
<th>CONTROL</th>
<th>CORBOUN</th>
<th>COVAPLY</th>
<th>CRCDIES</th>
<th>CRCEVNT</th>
<th>CRCKIL</th>
<th>CRCL0SS</th>
<th>CRCSEE</th>
<th>CRCTHNK</th>
<th>CRCTRAK</th>
<th>CRC2INT</th>
<th>CREATE</th>
<th>CRHLIML</th>
<th>DECDMS</th>
<th>DECRALO</th>
<th>DLIADD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*NO DATA STRUCTURES*
<table>
<thead>
<tr>
<th>DESTROY</th>
<th>DETECT</th>
<th>DGTSIX</th>
<th>DILOUT</th>
<th>DLYACT</th>
<th>DMSEDC</th>
<th>DOGFITE</th>
<th>DOGTHNK</th>
<th>DROPBLK</th>
<th>DROPPS</th>
<th>DROPPS2</th>
<th>ENGAGE</th>
<th>INTRYP</th>
<th>INSTAT</th>
<th>EXITP</th>
<th>HEDEL</th>
<th>FETCH</th>
<th>FILERUP</th>
<th>FILTER</th>
<th>FINDBLK</th>
<th>FINDFL</th>
<th>FINDIT</th>
<th>FIRECHK</th>
<th>FLITE</th>
</tr>
</thead>
</table>

'NO DATA STRUCTURES
<table>
<thead>
<tr>
<th>RLGTYP</th>
<th>RLWAVE</th>
<th>RONDSEE</th>
<th>ROUTER</th>
<th>SAMATON</th>
<th>SAMPRCM</th>
<th>SAMSEE</th>
<th>SAMWYPE</th>
<th>SCHEDUL</th>
<th>SCHTAB</th>
<th>SDIGEST</th>
<th>SLEKENG</th>
<th>SEEKP</th>
<th>SEEKAC</th>
<th>SEEKFU</th>
<th>SELECT</th>
<th>SEMANT</th>
<th>SETASSN</th>
<th>SHRILL</th>
<th>SHUFFLE</th>
<th>SBSRTRK</th>
<th>SNAP</th>
<th>Srichpl</th>
<th>SSEE</th>
</tr>
</thead>
</table>

'NO DATA STRUCTURES'
<table>
<thead>
<tr>
<th>STATPAK</th>
<th>STICK</th>
<th>TABOUT</th>
<th>TABOUT2</th>
<th>TERTMACG</th>
<th>TFLYCRC</th>
<th>TGTGONE</th>
<th>TGTHEX</th>
<th>TGTLIST</th>
<th>THH2PS</th>
<th>THTHPIN</th>
<th>THX2XY</th>
<th>TH2NX</th>
<th>TLL2NX</th>
<th>TLL2XY</th>
<th>TOADIL</th>
<th>TOWER</th>
<th>TRACE</th>
<th>TRKCHEK</th>
<th>FRYSHOT</th>
<th>TTIME</th>
<th>TXY2NX</th>
<th>TXY2HXL</th>
<th>TXY2XL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*NO DATA STRUCTURES*
<table>
<thead>
<tr>
<th>ULSUB2</th>
<th>UMPIRE</th>
<th>UNLINK</th>
<th>UNSNAP</th>
<th>UNSTAT</th>
<th>UDLSUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>UOLOAD</td>
<td>WIPEOUT</td>
<td>WITHDRAW</td>
<td>XPAA</td>
<td>XPD</td>
<td>XPK</td>
</tr>
<tr>
<td>XSHIFT*</td>
<td>XY2HX</td>
<td>YANK</td>
<td>LTRMRG</td>
<td>UNPACK*</td>
<td>DBGREAD*</td>
</tr>
<tr>
<td>ICHECK*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*NO DATA STRUCTURES*
<table>
<thead>
<tr>
<th>Column Block</th>
<th>Column Block</th>
<th>Column Block</th>
<th>Column Block</th>
<th>Column Block</th>
<th>Column Block</th>
<th>Column Block</th>
<th>Column Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
</tr>
<tr>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
</tr>
<tr>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
</tr>
<tr>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
</tr>
<tr>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
</tr>
<tr>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
</tr>
<tr>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
<td>Column Block</td>
</tr>
</tbody>
</table>

506
<table>
<thead>
<tr>
<th></th>
<th>ON</th>
<th>DUMMY BLOCK</th>
<th>ENGAGE</th>
<th>EVENT</th>
<th>PAYLOAD</th>
<th>FIRE</th>
<th>FIRE UNIT</th>
<th>FORMATION</th>
<th>FORMATION BLOCK</th>
<th>FORMATION HEAD BLOCK</th>
<th>HEX</th>
<th>HEX BLOCK</th>
<th>HEX LINK</th>
<th>LEFT FREE</th>
<th>LOAD</th>
<th>MESSAGE</th>
<th>NEXT</th>
<th>NOAVAL BLOCK</th>
<th>ORDER</th>
<th>PAYLOAD</th>
<th>PAYLOAD LIST</th>
<th>PAYLOAD BUFFER</th>
<th>PL BUFFER</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLTGEOM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLTWYPE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLYSEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORMTGT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSAMP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSINIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FUELCHK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETHEX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GSHIPRS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GIMME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GNDLOCK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GOATEM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GOTOAB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HAND/PT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEXADD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEXCH2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEXINV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEXMLT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEXXMOV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEXMULT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HISTORY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTPNT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOLD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BDALT</td>
<td>BDLEX</td>
<td>BDLRK</td>
<td>BDPAKS</td>
<td>BLKDAT</td>
<td>BNCMDPR</td>
<td>BNCONGD</td>
<td>BNCOUIS</td>
<td>BNCONTIC</td>
<td>BNCUALL</td>
<td>BNNODRO</td>
<td>BNNWTRK</td>
<td>BNPON8B</td>
<td>BNPON8D</td>
<td>BNPONDA</td>
<td>BNPONEP</td>
<td>BNPONFA</td>
<td>BNPONFD</td>
<td>BNPONSS</td>
<td>BNRCOV</td>
<td>BNOCTNK</td>
<td>BNPASIN</td>
<td>BNP2CRC</td>
<td>BRTYTNK</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>-------</td>
<td>--------</td>
<td>--------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PROGDED BLOCK</td>
<td>CALL BUFFER</td>
<td>QUEUE CTRL</td>
<td>RAID CTRK</td>
<td>READY LCL</td>
<td>SQB</td>
<td>SEE BUF</td>
<td>SER</td>
<td>SOURCE</td>
<td>SUB</td>
<td>SUBLST</td>
<td>TARGET BLOK</td>
<td>TARGET LIST BLOK</td>
<td>TID BLOK</td>
<td>TID TREE</td>
<td>WAVE BLOK</td>
<td>WINGMAN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>---------------</td>
<td>-------------</td>
<td>------------</td>
<td>----------</td>
<td>-----------</td>
<td>-----</td>
<td>---------</td>
<td>-----</td>
<td>---------</td>
<td>-----</td>
<td>--------</td>
<td>-------------</td>
<td>-------------------</td>
<td>---------</td>
<td>---------</td>
<td>----------</td>
<td>---------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLOSCOR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNACTTTX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CODE01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CODE03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CODE05</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CODE1B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMMAND</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMMNO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CORBOWN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COVAPLY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRCDIES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRCEVNT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRCKIL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRCLOSS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRCSEE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRCTHNK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRCTRAK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRC2INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRFTML</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECDMS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECRALO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DELADD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DISTRIBUTION LIST

DEPARTMENT OF DEFENSE

Armed Forces Staff College
ATTN: Reference & Technical Svcs Br

Assistant Secretary of Defense
International Security Affairs
ATTN: European & NATO Affairs
ATTN: Policy Plans & NSC Affairs

Assistant Secretary of Defense
Comm. Cnd, Cont & Intell
ATTN: Surveillance & Warning Sys

Assistant to the Secretary of Defense
Atomic Energy
ATTN: Executive Assistant

Commander-in-Chief, Atlantic
ATTN: N-22

Commander-in-Chief, Pacific
ATTN: J-634
ATTN: J-54
ATTN: J-32

Defense Intelligence Agency
ATTN: DIA/VPA-2, Fed Res Div
ATTN: DB-6
5 cy ATTN: DB-4

Defense Nuclear Agency
ATTN: SINA
ATTN: NATA
ATTN: NATD
4 cy ATTN: TITL

Defense Technical Information Center
12 cy ATTN: DD

Field Command
Defense Nuclear Agency
ATTN: FCPR

Field Command
Defense Nuclear Agency
Livermore Branch
ATTN: FCPRL

Intelligence Center, Pacific
ATTN: COMIPAC

Interservice Nuclear Weapons School
ATTN: TTV

Joint Chiefs of Staff
ATTN: J-3
ATTN: J-5, Strategy Division, W. McClain
ATTN: J-5, Nuclear Division
ATTN: J-3, Strategic Operations Div
ATTN: SAGA/SSD
ATTN: SAGA/SFD

National Defense University
ATTN: MACLIB-CR
ATTN: ICAP, Tech Lib

DEPARTMENT OF DEFENSE (Continued)

Joint Strategic Planning Staff
ATTN: JL
ATTN: JLA
ATTN: JP
2 cy ATTN: JLTW-2

Director
NET Assessment
Office of the Secretary of Defense
ATTN: Military Assistants

U.S. European Command
ATTN: J-2
ATTN: J-3
ATTN: J-LW
ATTN: J-SSRG
ATTN: J-6
ATTN: J-2-1TO

U.S. National Military Representative SHAPE
ATTN: U.S. Documents Officer

Under Secretary of Defense for Rsch & Engrg
ATTN: Strategic & Space Sys (OS)
ATTN: Tactical Warfare Programs

DEPARTMENT OF THE ARMY

Deputy Chief of Staff for Ops & Plans
Department of the Army
ATTN: DAMO-2XK
ATTN: DAMO-NC

Harry Diamond Laboratories
Department of the Army
ATTN: DRL-D-N-P

U.S. Army Cnd & General Staff College
ATTN: ACQ, Library Div
Army Concept Analysis Agency
ATTN: CCA-AADL

Commander-in-Chief
U.S. Army Europe and Seventh Army
ATTN: AEACG
ATTN: AEAGE

U.S. Army Nuclear & Chemical Agency
ATTN: Library

U.S. Army TRADOC Sys Analysis Actvy
ATTN: ATAA-TAC

U.S. Army War College
ATTN: Library

U.S. Military Academy
Department of the Army
ATTN: Document Library
DEPARTMENT OF THE NAVY

Naval Postgraduate School
ATTN: Code 1424, Library

Naval Research Laboratory
ATTN: Code 1240

Naval Sea Systems Command
ATTN: SEA-09533
ATTN: SEA-643

Naval Surface Weapons Center
ATTN: Code F31

Naval War College
ATTN: Document Control

Nuclear Weapons Tng Group, Pacific, Dept of Navy
ATTN: Document Control

Nuclear Weapons Tng Group, Atlantic, Dept of Navy
ATTN: Document Control

Office of Naval Research
ATTN: Technical Info Services

DEPARTMENT OF THE AIR FORCE

Air Force Systems Command
ATTN: DL
ATTN: XR
ATTN: SD

Air Force Weapons Laboratory
Air Force Systems Command
ATTN: SUL

Air University Library
Department of the Air Force
ATTN: AUL-LSE

Assistant Chief of Staff
Intelligence
Department of the Air Force
ATTN: INA

Deputy Chief of Staff
Research, Development, & Acq
Department of the Air Force
ATTN: AFRDQI

Foreign Technology Division
Air Force Systems Command
ATTN: CCN
ATTN: TOTM
ATTN: SDN

Strategic Air Command
Department of the Air Force
ATTN: NR
ATTN: ADWN
ATTN: XP
ATTN: NRI, STINFO Library
ATTN: DO

U.S. Air Force, Pacific
ATTN: XP

Commander-in-Chief
U.S. Air Forces in Europe
ATTN: USAF/DEX

DEPARTMENT OF THE AIR FORCE (Continued)

Commander-in-Chief
U.S. Air Forces in Europe
ATTN: USAF/DEX

Commander-in-Chief
U.S. Air Force in Europe
ATTN: USAF/INA

Commander-in-Chief
U.S. Air Forces in Europe
ATTN: USAF/INT

Commander-in-Chief
U.S. Air Forces in Europe
ATTN: USAF/XPX

USAF Tactical Fighter Weapons Center
ATTN: Commander

DEPARTMENT OF ENERGY

Department of Energy
Albuquerque Operations Office
ATTN: CTID

OTHER GOVERNMENT AGENCY

Central Intelligence Agency
ATTN: OSWR/NED

DEPARTMENT OF ENERGY CONTRACTORS

Lawrence Livermore National Lab
ATTN: Tech Info Dept, Library

Los Alamos National Lab
ATTN: MS 364
ATTN: M/5634, T. Dowler

DEPARTMENT OF DEFENSE CONTRACTORS

Advanced International Studies Institute
ATTN: M. Harvey

Aerospace Corp
ATTN: Library

BDM Corp
ATTN: J. Braddock
4 cy ATTN: M. Filteau
4 cy ATTN: T. Hawkins
4 cy ATTN: L. Eliese
4 cy ATTN: B. Macaleer
4 cy ATTN: J. Phillips

JAYCOR
ATTN: R. Sullivan

Kaman Sciences Corp
ATTN: F. Shelton

Kaman Sciences Corp
ATTN: E. Daugs

Kaman Tempo
ATTN: DASIAC

Pacific-Sierra Research Corp
ATTN: G. Lang
ATTN: H. Brode
Pacific-Sierra Research Corp
ATTN: D. Gormley

R & D Associates
ATTN: R. Port
ATTN: P. Haas
2 cy ATTN: Document Control
2 cy ATTN: S. Borjon

R & D Associates
ATTN: J. Thompson

Science Applications, Inc
ATTN: Document Control
ATTN: J. Warner
ATTN: M. Drake

Science Applications, Inc
ATTN: W. Layson
ATTN: Document Control

Science Applications, Inc
ATTN: D. Kaul
ATTN: L. Goure

System Planning Corp
ATTN: J. Douglas

Tetra Tech, Inc
ATTN: F. Bothwell

TRW Defense & Space Sys Group
ATTN: N. Lipner

TRW Defense & Space Sys Group
ATTN: P. Dai