AMMUNITION RESUPPLY MODEL PROGRAMMERS MANUAL
VOLUME II

TECHNICAL REPORT TR 2-80

UNITED STATES ARMY
COMBINED ARMS CENTER

COMBINED ARMS
STUDIES AND ANALYSIS ACTIVITY

APPROVED FOR PUBLIC RELEASE DISTRIBUTION UNLIMITED

STUDIES AND ANALYSIS DIRECTORATE

UNCLASSIFIED
AMMUNITION RESUPPLY MODEL
VOLUME II
PROGRAMMERS MANUAL

by
Mr. Donald J. Remen
MAJ Robert B. Clarke
and
Mr. James Fox
ACN 36801

80-CACDA-2271

This document has been approved for public release and sale; its distribution is unlimited.
Ammunition Resupply Model: Volume II - Programmers Manual

Mr. Donald J. Remen
MAJ Robert B. Clarke
Mr. James Fox

Ammunition Resupply Model (ARM) was designed to simulate those activities associated with ammunition resupply-demand, reload, resupply-in-parallel with the play of the Jiffy war game in the evaluation of a division size force. The purpose of ARM is to assess the capability of a given TOE structure to respond to logistical demands placed upon it by various numbers of ammunition-expending weaponry. The other volume of the report is the Methodology and Users Manual which a discussion of model methodology, data base development, interface requirements with the war game, and the operators guide.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
   - War Game
   - Jiffy Game
   - Ammunition Resupply
   - Force Structure Trade-off
   - Division 86

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
   This report is one of two produced to document the Ammunition Resupply Model (ARM). The model was designed to simulate those activities associated with ammunition resupply-demand, reload, resupply-in-parallel with the play of the Jiffy war game in the evaluation of a division size force. The purpose of ARM is to assess the capability of a given TOE structure to respond to logistical demands placed upon it by various numbers of ammunition-expending weaponry. The other volume of the report is the Methodology and Users Manual which a discussion of model methodology, data base development, interface requirements with the war game, and the operators guide.
ABSTRACT

ARM (Ammunition Resupply Model) is an interactive/batch representation of Class V flow from the corps support area (CSA) to the weapon. The model receives a file consisting of a record of ammunition usage by unit by ammunition type from a combat simulation (presently, the simulation being used is METRO). The ARM using a preloaded data base of ammunition handling procedures and capabilities represents the flow of ammunition as it would have occurred within the battle. The unit status over time is indicative of the capability of the system to supply ammunition to the weapons and the reasonableness of the firing doctrine used within the attrition simulation given the ammunition resupply system carrying capabilities.

ARM is written in FORTRAN IV and is machine independent with the exception of the subroutine that reads the file created by the attrition model.
FOREWORD

In general, existing logistics models tend to address resupply requirements in aggregated terms, such as tons per man per day or rounds per tube per day. Although this approach has considerable merit for evaluating large force structures engaged in sustained combat, it is inadequate for addressing the impact of logistics on organizations engaged in short, intense conflict scenarios.

Ammunition expenditures emerging from high level (as opposed to high resolution) war games have traditionally been either unconstrained or based on a percentage of an "anticipated" daily resupply capability. Because of this, support analyses have not been the product of a concurrent logistics simulation utilizing the same scenario, but have been based on evaluations made after game completion. This method can paint a false picture of a combat unit's effectiveness. The logistics system, especially its ability to resupply critical commodities such as ammunition and fuel, must be evaluated during the course of the simulated battle.

The study directive for the Division-86 study called for a Force Structure Trade-off Analysis (FSTA) of various division alternatives. The tool for this FSTA effort was the Jiffy war game. To derive meaningful insights into the effects of the ammunition resupply assets contained in the different force structures and their impact on the combat effectiveness of the various units within the division, ammunition resupply had to be evaluated in some detail. Such an evaluation must include simulating the time-consuming resupply process that places ammunition on individual weapon systems, as well as the movement of the different units' transportation assets to secure additional ammunition. It is this concept that provides the basis for the Ammunition Resupply Model (ARM), a concept that reflects the real-world factors that affect ammunition resupply. ARM was, therefore, developed to work in parallel with Jiffy in conducting a total FSTA of the Division-86 alternatives.

The concept for ARM was developed in Oct-Nov 1978, with the methodology and logic flow charts being completed in Dec 1978. The actual coding of the model was accomplished from Dec 1978 through Feb 1979, and the model was operational in May 1979. This report provides the documentation for the program description and the FORTRAN code listing.

The authors of this report wish to acknowledge Harry Jones of the Model Design, Development and Validation Branch of COA for his assistance in programming several of the operating routines. Our thanks also to Mr. Ken Pickett, Dr. Dave Bash, and Mr. Harvey Taylor of Methodology and Quality Assurance Branch for their help in providing some initial file structure organization and programming logic flow charts. Special thanks are given to Mrs. Elizabeth Etheridge, who served as Technical Editor for this report, and the girls in the Word Processing Center East, who typed the report.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>i</td>
</tr>
<tr>
<td>FOREWORD</td>
<td>ii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS.</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vi</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>GENERAL INFORMATION.</td>
<td>1</td>
</tr>
<tr>
<td>Summary of ARM Operations</td>
<td>1</td>
</tr>
<tr>
<td>Program Developer</td>
<td>1</td>
</tr>
<tr>
<td>Program Specifications</td>
<td>3</td>
</tr>
<tr>
<td>Operating Environment</td>
<td>3</td>
</tr>
<tr>
<td>COMMON BLOCKS IN ARM</td>
<td>4</td>
</tr>
<tr>
<td>Composition of Common Blocks</td>
<td>4</td>
</tr>
<tr>
<td>Common Block Usage in ARM</td>
<td>4</td>
</tr>
<tr>
<td>PROGRAM STRUCTURE - MACRO VIEW</td>
<td>15</td>
</tr>
<tr>
<td>Purpose of This Report Section</td>
<td>15</td>
</tr>
<tr>
<td>Main Program Operations</td>
<td>15</td>
</tr>
<tr>
<td>ARM Subroutine Structure</td>
<td>15</td>
</tr>
<tr>
<td>DESCRIPTIONS OF THE INDIVIDUAL SUBROUTINES</td>
<td>20</td>
</tr>
<tr>
<td>a. PROGRAM ARM</td>
<td>21</td>
</tr>
<tr>
<td>b. SUBROUTINE GETEV.T.</td>
<td>25</td>
</tr>
<tr>
<td>c. SUBROUTINE NEXTEV</td>
<td>27</td>
</tr>
<tr>
<td>d. SUBROUTINE PUTEVT</td>
<td>29</td>
</tr>
<tr>
<td>Subroutine Name</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SUBROUTINE ASP</td>
<td>33</td>
</tr>
<tr>
<td>SUBROUTINE ASPAR1</td>
<td>37</td>
</tr>
<tr>
<td>SUBROUTINE ASPARV</td>
<td>40</td>
</tr>
<tr>
<td>SUBROUTINE ATP</td>
<td>43</td>
</tr>
<tr>
<td>SUBROUTINE ATPARV</td>
<td>49</td>
</tr>
<tr>
<td>SUBROUTINE ATPAR1</td>
<td>53</td>
</tr>
<tr>
<td>SUBROUTINE ATPAR2</td>
<td>56</td>
</tr>
<tr>
<td>SUBROUTINE CONTRL</td>
<td>59</td>
</tr>
<tr>
<td>SUBROUTINE CSAARV</td>
<td>61</td>
</tr>
<tr>
<td>SUBROUTINE DEMAND</td>
<td>64</td>
</tr>
<tr>
<td>SUBROUTINE ENDSIM</td>
<td>68</td>
</tr>
<tr>
<td>SUBROUTINE HASPAR</td>
<td>70</td>
</tr>
<tr>
<td>SUBROUTINE HELARV</td>
<td>72</td>
</tr>
<tr>
<td>SUBROUTINE RELoad</td>
<td>75</td>
</tr>
<tr>
<td>SUBROUTINE REPORT</td>
<td>80</td>
</tr>
<tr>
<td>SUBROUTINE UNTARV</td>
<td>86</td>
</tr>
<tr>
<td>SUBROUTINE UNTDEP</td>
<td>89</td>
</tr>
<tr>
<td>SUBROUTINE CREEVT</td>
<td>93</td>
</tr>
<tr>
<td>SUBROUTINE EDIT</td>
<td>96</td>
</tr>
<tr>
<td>SUBROUTINE EVINIT</td>
<td>101</td>
</tr>
<tr>
<td>SUBROUTINE EVSTOP</td>
<td>103</td>
</tr>
<tr>
<td>SUBROUTINE FINTK</td>
<td>105</td>
</tr>
<tr>
<td>SUBROUTINE GETQUE</td>
<td>108</td>
</tr>
<tr>
<td>SUBROUTINE INIT</td>
<td>110</td>
</tr>
</tbody>
</table>
cc. SUBROUTINE INTROK. ........................................... 112
dd. SUBROUTINE IQ. ............................................... 115
ee. SUBROUTINE LDPWDR. .......................................... 117
ff. SUBROUTINE LOOKEV. .......................................... 120
gg. SUBROUTINE NXTQUE. .......................................... 122
hh. SUBROUTINE OPERA ............................................ 124
ii. SUBROUTINE PUTQUE. .......................................... 127
jj. SUBROUTINE QINIT ............................................. 129
kk. SUBROUTINE RDIEEXO ......................................... 131
ll. SUBROUTINE RDJIFF ........................................... 134
mm. SUBROUTINE READF ........................................... 140
nn. SUBROUTINE SCHED ........................................... 144
oo. SUBROUTINE SETQUE .......................................... 146
pp. SUBROUTINE TRKPUT .......................................... 148
qq. SUBROUTINE TRKTIM .......................................... 151
rr. SUBROUTINE TRUCK ............................................ 153

DESCRIPTION OF ASSOCIATED PROGRAMS .......................... 155
a. PROGRAM HJEDIT ................................................ 155
b. PROGRAM HUEEDIT ............................................. 155
c. SUBROUTINE EDIT .............................................. 155
d. SUBROUTINE UPDATE ........................................... 162
e. ROUTINE HJDATABASE ........................................... 165
f. ROUTINE HSDATABASE .......................................... 165
<table>
<thead>
<tr>
<th></th>
<th>PROGRAM HSREADJIF</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>g</td>
<td></td>
<td>169</td>
</tr>
<tr>
<td>h</td>
<td>PROGRAM HSREDJIFCH</td>
<td>171</td>
</tr>
<tr>
<td>i</td>
<td>PROGRAM TRKQUE</td>
<td>174</td>
</tr>
</tbody>
</table>

APPENDIXES

A. DISTRIBUTION LIST
LIST OF TABLES

TABLE 1. Composition of the ARM Common Blocks ............ 4
TABLE 2. Definitions of the COMMON Arrays and Variables ... 5
TABLE 3. Additional Data and Codes ......................... 10
TABLE 4. Use of COMMON Arrays and Variables by ARM Subroutines ..................... 12
TABLE 5. Use of COMMON Blocks by the ARM Subroutines ...

TABLE 7. Subroutine Calls in ARM ......................... 18
TABLE 8. Calling Subroutines in ARM ....................... 20

LIST OF FIGURES

FIGURE 1. ARM Methodology Flow Diagram .................... 2
FIGURE 2. ARM Driver Program Flow ........................ 16
PROGRAMMERS MANUAL

1. INTRODUCTION

a. This manual is intended for the programmer who has the task of maintaining, transferring, and/or modifying the Ammunition Resupply model (ARM). General information is presented first, followed by more detailed program descriptions.

   (1) Paragraph 2 provides a general overview of the program, its developers, users, hardware requirements, and major program components.

   (2) Paragraph 3 lists the COMMON blocks and defines the variables in each. Tables show the COMMON blocks used in each subroutine and the cross-indexed list of subroutines using each COMMON block.

   (3) Paragraph 4 addresses the subroutine structure of the program. Flow diagrams indicate the principal subroutine linkages, and tables show the calls in each subroutine as well as the cross-indexed list of the calling routines.

   (4) The final paragraph provides detailed information on the individual subroutines.

b. Descriptions of the input and output, how the ARM is run, and how the results are used will be found in volume I of the ARM documentation.

2. GENERAL INFORMATION

a. Summary of ARM Operations. ARM is a set of computer routines designed to assist an analyst in studying the ammunition flow from the Corps Support Area (CSA) to the individual weapons. The initial development objective was to provide a methodology for addressing ammunition supply implications of the Division 86 alternatives. A quick overview of ARM is at figure 1. The main driver routine directs control to five major sections:

   o Data Base Modification
   o Event Handling
   o Report Production
   o Event Processing
   o Check Point/Restart Logic

b. Program Developer. ARM was developed by the Combat Operations Analysis Directorate, CACDA, CAC, Fort Leavenworth, for use in the Division 86 study.
START

DEMAND

HELAR/ EMER RESUPPLY

RELOAD TRKS RELOAD WEAPONS*

HASPAR HELICOPTER ARRIVAL

UNITARY TRKS ARR @ UNIT

UNITDEP TRKS DEP UNIT*

ATPARV TRKS ARR @ ATP*

ATP TRK FILLED S&P EMPTIED*

ASPAR V TRKS ARR @ ASP

ATPAR S&P CSA - ATP

CSAARV S&P ARR CSA*

UNITARV TRKS ARR @ UNIT

ASPAR S&P ATP + ASP

ATPAR2 S&P ASP + ATP

ASP TRK FILLED*

ASPAR VA TRKS FILLED

UNITARY TRKS ARR @ UNIT

* OPERATIONAL AVAILABILITY AND INTERDICTION CHECKED

ARM METHODOLOGY

FIGURE 1
c. Program Specifications.

(1) Language and operating system. ARM is written in standard FORTRAN IV, with the exception of a CDC-specific DECODE instruction in subroutine RDJIFF, and currently runs on the CDC 6500 at the TRADOC Data Processing Field Office (DPFO).

(2) Program size. There are 43 subroutines consisting of approximately 3400 lines of code. The program requires a 150 K Octal interactive password to operate.

(3) Execution times. The model has been tested with a brigade set of units and requires 8 seconds of computer time to process the input resulting from a JIFFY-produced 4 hours of engagement. A division set of units requires less than 15 seconds of computer processing.

(4) Program input. Program input comes from the following sources.

(a) File T1, which contains the data base of ammunition system characteristics and unit ammunition basic requirements.

(b) File T2, which contains the impending events, if any, from the previous run.

(c) File T3, ammunition usage file created by an attrition model.

(d) Input, supplied by the operator during the run, which is of three types:

1. Yes/No answers to select program options.

2. New values for specific run parameters.

3. Requests for desired reports.

(5) Program output. There are three types of program output:

(a) At his request the operator can elect to print a step by step processing of the model.

(b) The data base can be displayed in part or in total.

(c) Reports can be generated at operator-specified control points in the processing to display the system status.

d. Operating Environment. ARM requires an interactive terminal with a printer and/or CRT. Output can be routed to a highspeed printer.
3. COMMON BLOCKS IN ARM. Two programming methods are used to transfer data among the ARM subroutines:

- Arrays and variables are passed in the subroutine calling sequence.
- Arrays and variables are stored in the COMMON blocks.

a. Composition of COMMON Block. The use of labeled or named COMMON blocks makes it possible to make available to each subroutine only those variables it uses. For quick reference, table 1 lists the arrays and single variables making up each ARM COMMON block. Definitions of those arrays and variables are given in table 2, and table 3 lists some additional data and codes.

b. COMMON Block Usage in ARM. Table 4 lists the COMMON blocks used by each ARM subroutine. Table 5 cross-indexes this information, showing the subroutines using each COMMON block.

Table 1. Composition of the ARM COMMON Blocks

<table>
<thead>
<tr>
<th>Block Name</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENTS</td>
<td>JSTAT(6), JEVDS(1024,4), IEVS(5,1024)</td>
</tr>
</tbody>
</table>
| LOG        | IATP(4,30), IASP(4,41), 
             | IUNIT(75,69), ITRUCK(560,7), 
             | ITYPE(6,6), IMix(40,23), INTER(9), 
             | IRSTME(20,3), IATPSD(5), 
             | IDAY, TIME, ICsA(20), LPPAR(5) |
|            | IASPAM(4,20), LUOUT, TCIST, 
             | TCILNG, LOOK(17) |
| QUENUM     | IHEAD(136) |
| QUEPNT     | ITEMS(560) |
Table 2. Definitions of the COMMON Arrays and Variables

<table>
<thead>
<tr>
<th>COMMON Blocks</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENTS</td>
<td>: Event handler.</td>
</tr>
</tbody>
</table>

**JSTAT(I)**
- I=1, - Pointer to the first event chronologically
- I=2, - Pointer to the last event chronologically
- I=3, - Pointer to the next empty record to place an event
- I=4, - Total number of events presently in storage
- I=5, - Number of additional events that can be placed in storage
- I=6, - Total number of events that can be placed in storage.

**JEVDS (I,J)**
- I=1-1024, - Event position-in-storage index
- J=1, - Pointer to the position of the next event
- J=2, - Pointer to the position of
Table 2 (continued)

<table>
<thead>
<tr>
<th>J</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>Basic ammo level, ammo 1</td>
</tr>
<tr>
<td>19-21</td>
<td>Ammo 2</td>
</tr>
<tr>
<td>22-24</td>
<td>Ammo 3</td>
</tr>
<tr>
<td>25-27</td>
<td>Ammo 4</td>
</tr>
<tr>
<td>28-30</td>
<td>Ammo 5</td>
</tr>
</tbody>
</table>

IASP(I, J)

| I-1-4 | Data sets for (ASP) one through four.                                       |
| J=1   | Distance to CSA                                                               |
| J=2   | Distance to ATP                                                               |
| J=3   | UTM Coordinate                                                                |
| J=4   | Empty                                                                         |
| J=5   | Number trucks to CSA                                                          |
| J=6   | A flag that = 0 if the routine queue has not served a truck this war, 1 otherwise |
| J=7   | Number routine servers active                                                  |
| J=8   | Number GSRS servers active                                                    |
| J=9   | Routine queue number                                                          |
| J=10  | GSRS queue number                                                             |
| *J=11 | A flag that = 0 if the GSRS queue has not served a truck this war, 1 otherwise |
| J=12  | Number trucks in routine queue                                                |
| *J=13 | Number trucks in GSRS queue                                                   |
| J=14  | Current ammo supply, ammo 1                                                   |
| J=15  | Ammo 2                                                                         |
| J=33  | Ammo 20                                                                        |
| J=34  | Empty                                                                         |
| J=41  | Empty                                                                         |

UNIT (I, J)

| I = 1-75 | Contains the data sets for units numbered 1 to 75.                            |
| J=1     | Type Unit                                                                      |
| J=2     | ATP number                                                                     |
| J=3     | ASP number                                                                     |
| J=4     | Distance to ATP in Km                                                         |
| J=5     | Distance to ASP in Km                                                         |
| J=6     | UTM coordinate                                                                |
| J=7     | Jiffy unit name                                                                |
| J=8     | First ammo type                                                               |
| J=9     | Number weapons alive, First ammo type                                         |
| J=10    | Number weapons short ammo, First ammo type                                    |
| J=11    | Number rounds short, (Wpns) First ammo type                                   |
| J=12    | Current ammo supply, (Wpns) First ammo type                                   |
| J=13    | Routine resupply level, (Per Wpn) First ammo type                             |
| J=14    | Critical resupply level, (Per Wpn) First ammo type                            |

*Note: Not used since each GSRS truck has its own crane and does not wait in queue.*
Table 2 (continued)

| J=15, | Basic ammo level, (Per Wpn) First ammo type |
| J=16, | Ammo on trucks, First ammo type |
| J=17, | Number of weapons killed in CI, First ammo type |
| J=18, | Number of weapons short ammo, First ammo type |
| J=19, | Total rounds short through whole CI, First ammo type |
| J=20-31, | Second ammo type |
| J=32-43, | Third ammo type |
| J=44-55, | Fourth ammo type |
| J=56-67, | Fifth ammo type |
| J=68, | Number of helicopters assigned |
| J=69, | = 0 If single pulse demand per CI |
|       | = 1 If multiple pulses per CI |
|       | = N |
Table 2 (continued)

\( \text{INTER}(I) \)

\( I=1, \) - Counter for zone 1 trucks killed in INTROK
\( I=2, \) - Counter for zone 2 trucks killed in INTROK
\( I=3, \) - Maximum number of trucks to be killed in zone 1
\( I=4, \) - Maximum number of trucks to be killed in zone 2
\( I=5, \) - Time to replace truck interdicted in zone 1
\( I=6, \) - Time to replace truck interdicted in zone 2
\( I=7, \) - Modulo of trucks to be killed in zone 1 and zone 2
\( I=8, \) - Number of zone 1 trucks entering INTROK
\( I=9, \) - Number of zone 2 trucks entering INTROK

\( \text{IRSTME}(I,J) \)

\( I=1-20, \) - Designates the ammunition type associated with the data set
\( J=1, \) - Weapon set-up time in minutes
\( J=2, \) - Load time per round in minutes
\( J=3, \) - Travel time to weapon in minutes

\( \text{IATPSD}(I) \)

\( I=1, \) - Maximum number of servers at the ATP
\( I=2, \) - Threshold 1 for queue 1 at an ATP
\( I=3, \) - Threshold 2 for queue 1 at an ATP
\( I=4, \) - Threshold 1 for queue 2 at an ATP
\( I=5, \) - Threshold 2 for queue 2 at an ATP

\( \text{IDAY} \)

0, - Night
1, - Day

\( \text{TIME} \)

Contains the present battle time of the simulation in decimal minutes

\( \text{ICSA}(I) \)

\( I=1-20, \) - Contains the number of rounds by 20 ammunition types, drawn from the corps storage area stock since the beginning of the game.

\( \text{LPPAR}(I) \)

\( I=1, \) - Total number of ammo codes (20)
\( I=2, \) - Number of ammo codes at ATP (5)
\( I=3, \) - Number of maneuver unit ammo codes at ATP (2)
\( I=4, \) - Number of transports(trucks) (LT 560)
\( I=5, \) - Number of helicopters available (LT 560)

\( \text{IASPAM}(I,J) \)

\( I=1-4, \) - Designates the ammunition supply point associated with the data set.
Table 2 (concluded)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>J</td>
<td>The number of rounds by ammunition type removed from the ammunition supply point</td>
</tr>
<tr>
<td>LUOUT</td>
<td>The logical file for write statements; = 2 if all output to CRT, = 6 if all nonoperator interface output to a local output file</td>
</tr>
<tr>
<td>TCIST</td>
<td>Decimal minutes battle time at the start of current run</td>
</tr>
<tr>
<td>TCILNG</td>
<td>Decimal minutes battle time length of current run.</td>
</tr>
<tr>
<td>LOOK(I)</td>
<td>Decimal minutes battle time length of current run.</td>
</tr>
<tr>
<td>I</td>
<td>Contains print control flag</td>
</tr>
<tr>
<td></td>
<td>= 1 if want to see all events of type I</td>
</tr>
<tr>
<td></td>
<td>= 0 if do not want to see events of type I</td>
</tr>
<tr>
<td>QUENUM</td>
<td>Contains the truck number of the first truck to enter queue I that is still in queue I</td>
</tr>
<tr>
<td>IHEAD(I)</td>
<td>Contains in space I the truck which is next in the same queue as truck I is in</td>
</tr>
<tr>
<td>QUEPNT</td>
<td></td>
</tr>
</tbody>
</table>
Table 3. ADDITIONAL DATA AND CODES

**Ammo Type Codes:**
1 - 105 mm (M60-A3/XM1)  
2 - TOW  
3 - Powder Canisters  
4 - 155 HE  
5 - 155 ICMDP  
6 - 155 Smoke  
7 - 155 CLGP  
8 - 8 inch HE  
9 - 8 inch ICMDP  
10 - GSRS  
11 - Mortars  
12 - DIVAD  
13 - Hellfire  
14 - XR-TOW  
15 - STINGER  
16 - DRAGON  
17 - BUSHMASTER  
18 - EMPTY  
19 - EMPTY  
20 - EMPTY  
25 - False code for handling TOW vehicles other than the ITV due to differing basic load parameters.

**Unit Type Codes:**
1 - Tank task force  
2 - Mech task force  
3 - Armored cav sqdn  
4 - 155 arty btry  
5 - 8 inch arty btry  
6 - GSRS btry  
7 - DIVAD gun plt  
8 - CBT avn plt

**Truck Type Codes:**
1 - 10 ton  
2 - 5 ton  
3 - 5 ton with 1 1/2 ton trailer  
4 - 10 ton w/15 ton trailer  
5 - 22 1/2 ton stake and platform  
6 - Helicopter, CH 47

**Truck Mission Type Codes:**
1 - Unit truck  
2 - CSA - ATP link  
3 - CSA - ASP link  
4 - ASP - ATP link  
5 - ASP - Unit (helicopter)

**Truck Status Type Codes:**
1 - In unit queue  
2 - In ATP queue
3 - In ASP queue  
4 - In transit  
5 - Unit truck going from ATP to ASP  
6 - Truck awaiting repair  
7 - Truck dead (interdicted)  

<table>
<thead>
<tr>
<th>QUEUE NUMBER</th>
<th>QUEUE TYPE</th>
<th>QUEUE USE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-75</td>
<td>1</td>
<td>At each unit</td>
</tr>
<tr>
<td>101-104</td>
<td>2</td>
<td>At ATPS for CSA-ATP trucks</td>
</tr>
<tr>
<td>105-108</td>
<td>3</td>
<td>At ATPS for ASP-ATP trucks</td>
</tr>
<tr>
<td>109-112</td>
<td>4</td>
<td>At ATPS for unit artillery server</td>
</tr>
<tr>
<td>113-116</td>
<td>5</td>
<td>At ATPS for unit maneuver server</td>
</tr>
<tr>
<td>117-120</td>
<td>6</td>
<td>Not used</td>
</tr>
<tr>
<td>121-124</td>
<td>7</td>
<td>At ASPS for CSA-ASP trucks (Not Use</td>
</tr>
<tr>
<td>125-128</td>
<td>8</td>
<td>At ASPS for routine server</td>
</tr>
<tr>
<td>129-132</td>
<td>9</td>
<td>At ASPS for GSRS server</td>
</tr>
<tr>
<td>133-136</td>
<td>10</td>
<td>Not used</td>
</tr>
</tbody>
</table>
Table 4. Use of Common Arrays and Variables by ARM Subroutines

<table>
<thead>
<tr>
<th>Routine</th>
<th>Arrays/Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM DRIVER</td>
<td>TIME</td>
</tr>
<tr>
<td>ASP</td>
<td>IASP, IASPAM, IDAY, IMIX, ITRUCK</td>
</tr>
<tr>
<td></td>
<td>ITPYPE, IUNIT, LPPAR, TIME</td>
</tr>
<tr>
<td>ASPPAR1</td>
<td>IASP, IASPAM, IDAY, IMIX, ITRUCK, ITPYPE, LPPAR, TIME</td>
</tr>
<tr>
<td>ASPARV</td>
<td>IASP, IDAY, IMIX, ITRUCK, TIME</td>
</tr>
<tr>
<td>ATP</td>
<td>IASP, IASPAM, IATP, IATPSD, IDAY, IMIX, ITRUCK, ITPYPE, IUNIT, LPPAR, LOUT, TIME</td>
</tr>
<tr>
<td>ATPAR1</td>
<td>IATP, IMIX, ITRUCK, LPPAR</td>
</tr>
<tr>
<td>ATPAR2</td>
<td>IATP, IMIX, ITRUCK, LPPAR</td>
</tr>
<tr>
<td>ATPARV</td>
<td>IATP, IDAY, IMIX, ITRUCK, ITPYPE, IUNIT, LPPAR, TIME</td>
</tr>
<tr>
<td>CONTROL</td>
<td>IATP, IMIX, ITRUCK, LPPAR</td>
</tr>
<tr>
<td>CSAARV</td>
<td>IASP, IATP, ICSA, IDAY, IMIX, ITRUCK, ITPYPE, LPPAR, TIME</td>
</tr>
<tr>
<td>DEMAND</td>
<td>IASPAM, IATP, ICSA, IDAY, IMIX, ITRUCK, ITPYPE, IUNIT, LPPAR, LOUT, TIME</td>
</tr>
<tr>
<td>EDIT</td>
<td>IASP, IASPAM, IATP, IATPSD, ICSA, IDAY, IMIX, ITRUCK, ITPYPE, IUNIT, LPPAR, LOUT, TIME</td>
</tr>
<tr>
<td></td>
<td>ICSA, IDAY, IMIX, ITRUCK, ITPYPE, IUNIT, LPPAR, LUOUT, TIME</td>
</tr>
<tr>
<td>ENDSIM</td>
<td>Writes out log common and queue pointer commons</td>
</tr>
<tr>
<td>FINTK</td>
<td>IMIX, ITRUCK, LOUT</td>
</tr>
<tr>
<td>HASPAR</td>
<td>ITRUCK, LPPAR</td>
</tr>
<tr>
<td>HELARV</td>
<td>IDAY, IMIX, ITRUCK, ITPYPE, IUNIT, TIME</td>
</tr>
<tr>
<td>INIT</td>
<td>Reads all log and queue files for restart</td>
</tr>
<tr>
<td>INTROK</td>
<td>INTER, ITRUCK, LOUT</td>
</tr>
<tr>
<td>IQ</td>
<td>None</td>
</tr>
<tr>
<td>LDPMWR</td>
<td>IDAY, IMIX, ITRUCK, ITPYPE, TIME</td>
</tr>
<tr>
<td>LOOKV</td>
<td>LOOK</td>
</tr>
<tr>
<td>OPERA</td>
<td>ITRUCK, ITPYPE, LOUT</td>
</tr>
<tr>
<td>RDIEKO</td>
<td>IUNIT, TCILNG, TCIST, TIME</td>
</tr>
<tr>
<td>RDJIFF</td>
<td>IUNIT, LUOUT, TCILNG, TCIST, TIME</td>
</tr>
<tr>
<td>READF</td>
<td>None</td>
</tr>
<tr>
<td>RELOAD</td>
<td>IASPAM, IMIX, ITRUCK, LOUT, TIME</td>
</tr>
<tr>
<td>REPORT</td>
<td>IASP, IATP, ICSA, ITRUCK, ITPYPE, IUNIT, LPPAR, LOUT</td>
</tr>
<tr>
<td>TRKPUT</td>
<td>None</td>
</tr>
<tr>
<td>TRKTIM</td>
<td>ITRUCK, ITPYPE, LPPAR</td>
</tr>
<tr>
<td>UNTARV</td>
<td>IMIX, ITRUCK, IUNIT, LPPAR, TIME</td>
</tr>
<tr>
<td>UNTDEP</td>
<td>IDAY, IMIX, ITRUCK, ITPYPE, IUNIT, LPPAR, TIME</td>
</tr>
<tr>
<td>CREEV</td>
<td>NONE</td>
</tr>
<tr>
<td>EVINIT</td>
<td>Reads unprocessed events from T2 into common events</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>EVSTOP</td>
<td>Writes unprocessed events onto tape 2 from common event</td>
</tr>
<tr>
<td>GETEVT</td>
<td>IEVS, JEVDS, JSTAT</td>
</tr>
<tr>
<td>GETQUE</td>
<td>IPNT, NHEAD, (ITEM)(IHEAD)</td>
</tr>
<tr>
<td>NXTQUE</td>
<td>None</td>
</tr>
<tr>
<td>PUTEVT</td>
<td>None</td>
</tr>
<tr>
<td>PUTQUE</td>
<td>IEVS, JEVDS, JSTAT</td>
</tr>
<tr>
<td>QINIT</td>
<td>IPNT, ITEM, (IHEAD)(NHEAD)</td>
</tr>
<tr>
<td>SCHED</td>
<td>IEVS, JEVDS, JSTAT</td>
</tr>
<tr>
<td>SETQUE</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>IPNT, NHEAD, (ITEM)(IHEAD)</td>
</tr>
</tbody>
</table>
Table 5. Use of COMMON Blocks by the ARM Subroutines

<table>
<thead>
<tr>
<th>COMMON Array/Variable</th>
<th>Using subroutines</th>
</tr>
</thead>
<tbody>
<tr>
<td>IASP</td>
<td>ASP, ASPARI, ASPARV, ATP, CSAARV, EDIT, REPORT</td>
</tr>
<tr>
<td>IASPA</td>
<td>ASP, ASPARI, ATP, DEMAND, EDIT, RELOAD</td>
</tr>
<tr>
<td>IATP</td>
<td>ATP, ATPAR1, ATPAR2, ATARAV, CSAARV, DEMAND, EDIT, REPORT</td>
</tr>
<tr>
<td>IATPSD</td>
<td>ATP, EDIT</td>
</tr>
<tr>
<td>ICDA</td>
<td>CSAARV, DEMAND, EDIT, REPORT</td>
</tr>
<tr>
<td>IDAY</td>
<td>ASP, ASPARV, ATP, ATPARV, CSAARV, DEMAND, EDIT, HELARV, LDPWDR, UNTDEP</td>
</tr>
<tr>
<td>IHEAD</td>
<td>GETQUE, PUTQUE, SETQUE</td>
</tr>
<tr>
<td>IMIX</td>
<td>ASP, ASPARI, ASPARV, ATP, ATPAR1, ATPAR2, ATPARV, CSAARV, DEMAND, EDIT, FINTK, HELARV, LDPWDR, RELOAD, UNTARY, UNTDEP</td>
</tr>
<tr>
<td>INTER</td>
<td>EDIT, INTDK</td>
</tr>
<tr>
<td>IRSTME</td>
<td>EDIT, RELOAD</td>
</tr>
<tr>
<td>IITEM</td>
<td>GETQUE, PUTQUE, SETQUE</td>
</tr>
<tr>
<td>ITRUCK</td>
<td>ASP, ASPARI, ASPARV, ATP, ATPAR1, ATPAR2, ATPARV, DEMAND, EDIT, FINTK, HASPAR, HELARV, INTRDK, LDPWDR, OPERA, RELOAD, REPORT, TRKTIM, UNTARY, UNTDEP</td>
</tr>
<tr>
<td>ITYPE</td>
<td>ASP, ASPARI, ATP, ATPARV, CSAARV, DEMAND, EDIT, HELARV, LDPWDR, OPERA, REPORT, TRKTIM, UNTDEP</td>
</tr>
<tr>
<td>IUNIT</td>
<td>ASP, ATP, ATPARV, DEMAND, EDIT, HELARV, RELOAD, REPORT, UNTARY, UNTDEP</td>
</tr>
<tr>
<td>LOOK</td>
<td>EDIT, LOOEV</td>
</tr>
<tr>
<td>LPPAR</td>
<td>ASP, ASPARI, ATPAR1, ATPAR2, ATPARV, CSAARV, DEMAND, EDIT, HASPAR, REPORT, TRKTIM, UNTARY, UNTDEP</td>
</tr>
<tr>
<td>LOUT</td>
<td>ATP, DEMAND, EDIT, INTRDK, OPERA, RELOAD, REPORT</td>
</tr>
<tr>
<td>TCLNG</td>
<td>EDIT, RDIEXO, RDJIFF</td>
</tr>
<tr>
<td>TCIST</td>
<td>EDIT, RDIEXO, RDJIFF</td>
</tr>
<tr>
<td>TIME</td>
<td>ARM DRIVER, ASP, ASPARI, ASPARV, ATP, ATPARV, CSAARV, DEMAND, EDIT, HELARV, LDPWDR, OPERA, RDIEXO, RDJIFF, RELOAD, UNTARY, UNTDEP</td>
</tr>
<tr>
<td>IEVS</td>
<td>GETEVT, PUTEVT, QINIT, EVINIT, EVSTOP</td>
</tr>
<tr>
<td>JEVDS</td>
<td>GETEVT, PUTEVT, QINIT, EVINIT, EVSTOP</td>
</tr>
<tr>
<td>JSTAT</td>
<td>GETEVT, PUTEVT, QINIT, EVINIT, EVSTOP</td>
</tr>
</tbody>
</table>
4. PROGRAM STRUCTURE - A MACRO VIEW

a. Purpose of This Report Section. This section provides an overview of the ARM subroutine structure. Information is presented in the following order:

- Outline of main driver routine, with operational flow diagrams.
- Additional description of overall processing
- List of subroutines called by each subroutine
- The cross reference list of the callers of each subroutine

b. Main Program Operations. The basic control program in ARM is the mainline driver PROGRAM ARM. Its principal functions as shown in figure 2 are as follows:

- Initialize the files
- Obtain the next event
- Call in the proper subroutine to process the next event

c. ARM Subroutine Structure.

(1) Major subroutine groups. ARM consists of 43 routines. Table 6 shows the major grouping of routines into the following categories:

- Event storage/retrieval
- Event functional processing
- Support special purpose routines

(2) Subordinate calling pattern in ARM. This review of the ARM structure is completed by the exhibition of the subroutine calling pattern. Table 7 is the list of routines called by each routine. Table 8 is the cross-referenced routines calling the list of routines.
START

SET COMMONS DIMENSIONS

FIRST CI?

Y

INITIALIZE STATUS FILES

CALL INITIALIZE ROUTINE

SELECT NEXT EVENT

EVENT TIME ≥ CONTROL TIME

Y

CALL REPORTS

STOP

N

DEMAND

HEL ARV

HASPAR

RELOAD

UNTARV

UNTDEP

ATPARV

ATP

ASPAR1

ATPAR2

CSAARV

ATPAR1

ASPARV

ASP

CTRL

Figure 2. Program Arm
<table>
<thead>
<tr>
<th>GROUP</th>
<th>Routines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Storage/Retrieval</td>
<td>GETEV, NEXEV, PUTEVT</td>
</tr>
<tr>
<td>Event Functional Processing</td>
<td>ASP, ASPARV, ASPARI, ATP, ATPARV,</td>
</tr>
<tr>
<td></td>
<td>ATPARI1, ATPARI2, CONTRL, CSAARV, DEMAND,</td>
</tr>
<tr>
<td></td>
<td>ENDSIM, HASPAR, HELARV, RELOAD, REPORT,</td>
</tr>
<tr>
<td></td>
<td>UNTARV, UNTDEP</td>
</tr>
<tr>
<td>Support Special Purpose Routines</td>
<td>CREEVT, EDIT, EVINIT, EVSTOP, FINTK,</td>
</tr>
<tr>
<td></td>
<td>GETQUE, INIT, INTRDK, IQ, LDPWDR,</td>
</tr>
<tr>
<td></td>
<td>LOOKEV, OPERA, NXTQUE, PUTQUE, QINIT,</td>
</tr>
<tr>
<td></td>
<td>RANF, RDIEXO, RDJIFF, READF, SCHED,</td>
</tr>
<tr>
<td></td>
<td>SETQUE, TRKPUT, RANF, TRKTIM</td>
</tr>
</tbody>
</table>
Table 7. Subroutine Calls in ARM

<table>
<thead>
<tr>
<th>Program</th>
<th>ARM Calls</th>
<th>GETEVT Calls</th>
<th>ATP Calls</th>
<th>CSAARV Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASP</td>
<td>None</td>
<td>None</td>
<td>FINTK</td>
<td>INTROK</td>
</tr>
<tr>
<td>ASPARV</td>
<td>GETEVT</td>
<td>GETQUE</td>
<td>OPERA</td>
<td>OPERA</td>
</tr>
<tr>
<td>ASPAR1</td>
<td>PUTEVT</td>
<td>INTRDK</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>ATP</td>
<td>None</td>
<td>IQ</td>
<td>LDPWDR</td>
<td>DEMAND</td>
</tr>
<tr>
<td>ATPARV</td>
<td>ASP</td>
<td>OPERA</td>
<td>RDIEXO</td>
<td>RDIEXO</td>
</tr>
<tr>
<td>ATPAR1</td>
<td>ASPARV</td>
<td>SCHED</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>ATPAR2</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>CONTRL</td>
<td>ASP</td>
<td>INTROK</td>
<td>OPERA</td>
<td>OPERA</td>
</tr>
<tr>
<td>CSAARV</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>DEMAND</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>ENOSIM</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>EVINTT</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>EVSTOP</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>HASPAR</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>HELARV</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>INIT</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>LOOKEV</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>NEXTEV</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>RELOAD</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>REPORT</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>UNTARV</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>UNTDEP</td>
<td>ASPARV</td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
</tr>
<tr>
<td></td>
<td>FINTK</td>
<td>READF</td>
<td>READF</td>
<td>RDIEXO</td>
</tr>
<tr>
<td></td>
<td>INTROK</td>
<td>EVINIT</td>
<td>INTROK</td>
<td>EOF</td>
</tr>
<tr>
<td></td>
<td>IQ</td>
<td>QINIT</td>
<td>IQ</td>
<td>SCHED</td>
</tr>
<tr>
<td></td>
<td>MINO</td>
<td>LDPWDR</td>
<td>MINO</td>
<td>SCHED</td>
</tr>
<tr>
<td></td>
<td>SCHED</td>
<td>RDJIFF</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>REPORT</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
</tr>
<tr>
<td></td>
<td>TRUCK</td>
<td>TRUCK</td>
<td>TRUCK</td>
<td>TRUCK</td>
</tr>
<tr>
<td>UNTARV</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
</tr>
<tr>
<td></td>
<td>IQ</td>
<td>GETEVT</td>
<td>GETEVT</td>
<td>GETEVT</td>
</tr>
<tr>
<td></td>
<td>PUTQUE</td>
<td>PUTQUE</td>
<td>PUTQUE</td>
<td>PUTQUE</td>
</tr>
<tr>
<td></td>
<td>SCHED</td>
<td>SCHED</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td></td>
<td>GETEVT</td>
<td>SCHED</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td>UNTDEP</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
<td>Calls:</td>
</tr>
<tr>
<td></td>
<td>INTROK</td>
<td>CONTRO</td>
<td>CONTRO</td>
<td>CONTRO</td>
</tr>
<tr>
<td></td>
<td>OPERA</td>
<td>RDJIFF</td>
<td>RDJIFF</td>
<td>RDJIFF</td>
</tr>
<tr>
<td></td>
<td>SCHED</td>
<td>TRKTIM</td>
<td>TRKTIM</td>
<td>TRKTIM</td>
</tr>
<tr>
<td></td>
<td>INTRDK</td>
<td>SCHED</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td></td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td></td>
<td>SCHED</td>
<td>INTRDK</td>
<td>INTRDK</td>
<td>INTRDK</td>
</tr>
<tr>
<td></td>
<td>OPERA</td>
<td>SCHED</td>
<td>SCHED</td>
<td>SCHED</td>
</tr>
<tr>
<td></td>
<td>SCHED</td>
<td>OPERA</td>
<td>OPERA</td>
<td>OPERA</td>
</tr>
<tr>
<td>Category</td>
<td>Calls</td>
<td></td>
<td>Category</td>
<td>Calls</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
<td>-------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>CONTRL Calls:</td>
<td>None</td>
<td></td>
<td>TRKPUT Calls:</td>
<td>GETQUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NXTQUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PUTQUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>READF</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SETQUE</td>
</tr>
<tr>
<td>CREEVT Calls:</td>
<td></td>
<td></td>
<td>PUTQUE Calls:</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>QUINT Calls:</td>
<td>None</td>
</tr>
<tr>
<td>EDIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPORT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCHED</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRKPUT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREEVT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>READF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCHED</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRKTIM Calls:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RANF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8. Calling Subroutines in ARM

<table>
<thead>
<tr>
<th>Routine Called by</th>
<th>Routine Called by</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM PROGRAM (ARM-P)</td>
<td>IQ ASP, ATP, ATPAR1, ATPAR2,</td>
</tr>
<tr>
<td>NEXTEV ARM-P</td>
<td>LDPWDR, UNTARV</td>
</tr>
<tr>
<td>GETEVT NEXTEV</td>
<td>ASP, ATP, ATPARV,</td>
</tr>
<tr>
<td>PUTEVT SCHED</td>
<td>ATPAR1, ATPAR2,</td>
</tr>
<tr>
<td>ASP ARM-P</td>
<td>UNTARV</td>
</tr>
<tr>
<td>ASPARV ARM-P</td>
<td>ASP, ATPAR2,</td>
</tr>
<tr>
<td>ASPAR 1 ARM-Y</td>
<td>LDPWDR</td>
</tr>
<tr>
<td>ATP ARM-P</td>
<td>ASPARV, ATP, ATPARV,</td>
</tr>
<tr>
<td>ATPARV ARM-P</td>
<td>ATPAR1, ATPAR2,</td>
</tr>
<tr>
<td>ATPAR1 ARM-P</td>
<td>CSAARV, DEMANDHELARV,</td>
</tr>
<tr>
<td>ATPAR2 ARM-P</td>
<td></td>
</tr>
<tr>
<td>CONTRL ARM-P</td>
<td></td>
</tr>
<tr>
<td>CSAARV ARM-P</td>
<td></td>
</tr>
<tr>
<td>DEMAND ARM-P</td>
<td></td>
</tr>
<tr>
<td>ENDSIM ARM-P</td>
<td></td>
</tr>
<tr>
<td>HASPAR ARM-P</td>
<td></td>
</tr>
<tr>
<td>HELARV ARM-P</td>
<td></td>
</tr>
<tr>
<td>RELOAD ARM-P</td>
<td></td>
</tr>
<tr>
<td>REPORT ARM-P, CONTRL</td>
<td></td>
</tr>
<tr>
<td>UNTARV ARM-P</td>
<td></td>
</tr>
<tr>
<td>UNTDEP ARM-P</td>
<td></td>
</tr>
<tr>
<td>CREEVT CONTRL</td>
<td></td>
</tr>
<tr>
<td>EDIT CONTRL</td>
<td></td>
</tr>
<tr>
<td>EVINIT CONTRL</td>
<td></td>
</tr>
<tr>
<td>EVSTOP ARM-P</td>
<td></td>
</tr>
<tr>
<td>FINTK ARM-P</td>
<td></td>
</tr>
<tr>
<td>GETQUE ASP, ATP, FINTK, TRKPUT, TRKTIM</td>
<td></td>
</tr>
<tr>
<td>INITD ARM-P</td>
<td></td>
</tr>
<tr>
<td>INTRODK ASPARV, ATPARV, CSAARV, RELOAD, UNTDEP, LDPWDR</td>
<td></td>
</tr>
</tbody>
</table>

5. DESCRIPTIONS OF THE INDIVIDUAL SUBROUTINES. Descriptions of the individual subroutines are given in this section. The following information is given for each routine:

- Name
- Purpose
- COMMON blocks
- Subroutines called by this routine
- Subroutines calling this routine
- Variables in calling sequence
- Local arrays
- Subroutine functions
a. ROUTINE: ARM Program-No Parameters

PURPOSE: To control the overall system flow for the ARM

COMMON BLOCKS: LOG
                QUENUM
                QUEPNT

CALLS:
    ASP
    ASPARV
    ASPAR1
    ATP
    ATPARV
    ATPAR1
    ATPAR2
    CONTRL
    CSAARV
    DEMAND
    ENDSIM
    EVINIT
    EVSTOP
    HASPAR
    HELARV
    INIT
    LOOKEV
    NEXTEV
    RELOAD
    REPORT
    UNTARV
    UNTDEP

IS CALLED BY: Operator

CALLING PARAMETERS: NONE

LOCAL ARRAYS:
    IPARM(5) -- Storage array for carrying event parameters to the event processing routines.

FUNCTIONS:
    Initializes event queues.
    Calls INIT to enable setting of parameters for this ARM cycle.

LOOP LOGIC: Retrieves the next chronological event.
            Passes control to appropriate routine to process the event.
I

** ARRIVAL OF ASP IN Curriculum **  
11. CALL FSP(a) (I-40)  
GO TO 6

** ARRIVAL OF ASP on the other side of the Curriculum **  
12. CALL ASP-1 (I-40)  
GO TO 6

** ARRIVAL OF HELICOPTER AT UNIT **  
130 CALL HEL (I-40)  
GO TO 6

** ARRIVAL OF HELICOPTER AT ASP **  
140 CALL HEL-ASP (I-40)  
GO TO 6

** FIGHT **  
150 CALL FIGHT (I-40)  
GO TO 6

** INACTIVE CONTROL **  
160 CALL INACTIVE (I-40)  
GO TO 6

** SIMULATION END **  
170 CALL STOP (I-20)  
CALL STOP  
STOP  
END
b. SUBROUTINE: GETEVT

PURPOSE: Retrieves the next event chronologically from the event queue

COMMON BLOCKS: EVENTS

CALLS: NONE

IS CALLED BY: NEXTEV

CALLING PARAMETERS:

IEVT(5) - 5 parameters describing the event.
ITH - Integer minutes of battle.
ITS - Decimal minutes of battle minus ITH times 3600 and integerized.
IHIT - 0 if no events in queue, 1 if event in the queue.

LOCAL ARRAY:

JFORE (1024) - Equivalenced to first 1024 words of JEVDS of
EVENTS and points to the previous event.

JBACK (1024) - Equivalenced to the second 1024 words of JEVDS
of COMMON EVENTS and points to the follow-on event.

FUNCTIONS:

Retrieves the 5 parameters of the next event.
Retrieves the time of the next event occurring.
Updates the forward and backward pointers to reflect the next retrievable event.
SURROUNDED GET event (LET, LTH, LTH)

GETS NEXT EVENT

BEGIN

CHECK FOR ANY EVENTS IN QUEUE ... IF NULL, RETURN

G 2: I = I + 1

LTH = LEV (JFRYPT, 1)

LTH = LEV (JFRYPT, 1)

IF (LTH = LEV) THEN

IF (LTH = LEV) THEN

BACK (LTH, 1)

BACK (LTH, 1)

RETURN

RETURN

NEXT

NEXT

END

END

26
c. SUBROUTINE: NEXTEV

PURPOSE: Interface between ARM driver routine and the GETEVT routine to retrieve the next event.

COMMON BLOCKS: NONE

CALLS: GETEVT

IS CALLED BY: ARM Driver

CALLING PARAMETERS:

  ITYPE - The event type.

  IPARM (5) - The 5 parameters describing the event.

  TIME - Present Simulation Time.

LOCAL ARRAYS: NONE

FUNCTIONS:

  Calls GETEVT to retrieve event.

  Computes a single time from the two times stored in the event logic.
SUBROUTINE NEXTEV (IYPE, ITIM, TIME)

INTERFACE ROUTINE TO SET NEXT EVENT

C----- M. JONES DEC 7-

DIMENSION IPAR(5)

CALL GETEV (IPEM, ITIM, ITS)

TIME = ITIM + ITS / 3600.

IYPE = EPAR(1)
RETURN
END
d. SUBROUTINE: PUTEVT

PURPOSE: Places an event record in the queue in chronological order and updates the queue pointer tables. If the placement is successful the flag (ICHECK) is set equal to 1.

COMMON BLOCKS: EVENTS

CALLS: NONE

IS CALLED BY: SCHED

CALLING PARAMETERS: IEVT(5) - Contains the 5 parameters describing the event to be stored.
ITH - Contains the integer portion of the event time.
ITS - Contains the decimal portion of the event time multiplied by 3600.
ICHECK - 0 if no room on the file, 1 if there is room on the file.

LOCAL ARRAYS:

JFORE (1024) - Equivalenced to the first 1024 words of JEVDS and points to the previous event.

JBACK (1024) - Equivalenced to the second 1024 words of JEVDS and points to the subsequent event.

JTIME (1024, 2) - Equivalenced to the last 2048 words of JEVDS and keeps the time data associated with the event.

FUNCTIONS:

Checks to see if space is available.

Places event record in ARRAY IEVS in COMMON EVENTS.
Update pointers in event directory.
SUBROUTINE PUTEVT(IEVT, ITH, ITS, ICHECK)
C
C PUTEVT PLACES AN EVENT RECORD IN THE QUEUE IN CHRONOLOGICAL
C ORDER AND UPDATES THE QUEUE DIRECTORY. ICHECK FLAG SET
C IF INSERT WAS UNSUCCESSFUL.
C
80B DAVISON 1978
C
COMMON/EVENTS/JSTAT(6),JEVDS(1024,4), IEVS(5,1024)
DIMENSION IEVT(5),JFORE(1024),JBACK(1024),JTIME(1024,2)
EQUIVALENCE (JFORE(1),JEVDS(1,1)),(JBACK(1),JEVDS(1,2)),
Z (JTIME(1,1),JEVDS(1,3)),(JSTAT(1),JFIRST),(JSTAT(2),JLAST),
Z (JSTAT(3),JEMPTY),(JSTAT(4),NUMEVIT),(JSTAT(5),NEMPTY),
4 (JSTAT(6),MAXEVT)
C CHECK IF SPACE AVAILABLE .. IF NONE, RETURN
ICHECK = 1024 - NEMPTY
IF(.NEMPTY.LE.0) GOTO 400
ICHECK=0
LSAVE=JFORE(JEMPTY)
C PUT EVENT RECORD IEVT IN IEVS
DO 20 IN = 1,5
IEVS(IN,JEMPTY) = IEVT(IN)
20 CONTINUE
C IF NO EVENTS IN QUEUE, PERFORM THE FOLLOWING
IF(.NUMEVT.GE.1) GOTO 200
JFORE(JEMPTY)=0
JBACK(JEMPTY)=0
JFIRST=JEMPTY
JLAST=JEMPTY
GOTO 380
C IF ONE EVENT IN QUEUE, PERFORM THE FOLLOWING
200 CONTINUE
ITFH=JTIME(JFIRST,1)
ITFS=JTIME(JFIRST,2)
IF(.NUMEVT.GT.1) GOTO 300
C IF LOWEST TIME EVENT, PERFORM THE FOLLOWING
IF(ITH.GT.ITFH)GOTO 210
IF(ITH.EQ.ITFH.AND.ITS.GE.ITFS)GOTO 210
JFORE(JEMPTY)=JFIRST
JBACK(JEMPTY)=0
JBACK(JFIRST)=JEMPTY
JLAST=JFIRST
JFIRST=JEMPTY
GOTO 380
C ELSE THIS TIME IS EQUAL TO OR LATER THAN THE LAST EVENT
210 CONTINUE
JFORE(JEMPTY)=0
JBACK(JEMPTY)=JFIRST
JFORE(JFIRST)=JEMPTY
JLAST=JEMPTY
GOTO 380
C IF TWO OR MORE EVENTS IN QUEUE, PERFORM THE FOLLOWING
300 CONTINUE
C IF EVENT TIME IS LESS THAN FIRST EVENT, MAKE IEVT THE FIRST EVENT
IF(ITH.GT.ITHH)GO TO 310
IF(ITH.EQ.ITHH.AND.ITS.LE.ITS)GO TO 310
JFORE(JEMPTY)=JFIRST
JBACK(JEMPTY)=0
JBACK(JFIRST)=JEMPTY
JFIRST=JEMPTY
GOTO 380
C IF EVENT TIME IS GREATER THAN OR EQUAL TO LAST EVENT, MAKE IEVT LAST
310 CONTINUE
ITHL=JTIME(JLAST,1)
ITLS=JTIME(JLAST,2)
IF(ITH.LT.ITHLH)GO TO 320
IF(ITH.EQ.ITHH.AND.ITS.LT.ITS)GO TO 320
JFORE(JEMPTY)=0
JBACK(JEMPTY)=JLAST.
JFORE(JLAST)=JEMPTY
JLAST=JEMPTY
GOTO 380
C EVENT TIME IS BETWEEN JTIME(JFIRST) AND JTIME(JLAST)
320 CONTINUE
NUM=NUMEV-T-1
C IF EVENT TIME CLOSER TO FIRST, START SEARCH AT FIRST EVENT
325 IF((ITH-ITHH)-(ITHL-ITH))326,325,350
326 IND1=1
ITLH=ITH
ITS=ITFS
IND2=JFORE(JFIRST)
IT2H=JTIME(IND2,1)
IT2S=JTIME(IND2,2)
DO 330 I=1,NUM
IF(ITH.GT.ITHH)GO TO 327
IF(ITH.EQ.ITHH.AND.ITS.LE.ITS)GO TO 327
GO TO 340
327 IND1=IND2
ITH=ITHH
ITS=IT2S
IND2=JFORE(IND2)
IT2H=JTIME(IND2,1)
IT2S=JTIME(IND2,2)
330 CONTINUE
ICHECK=2
GOTO 400
340 JFORE(IND1)=JEMPTY
JBACK(JEMPTY)=IND1
JFORE(JEMPTY)=IND2
JBACK(IND2)=JEMPTY
GOTO 380
C EVENT TIME CLOSER TO LAST, START SEARCH AT LAST EVENT
350 IND1=JLAST
ITH=ITLH
ITS=ITLS
IND2=JBACK(JLAST)
IT2H=JTIME(IND2,1)
IT2S=JTIME(IND2,2)
DO 360 I=1,NUM
IF(ITH.LT.IT2H)GO TO 355
IF(ITH.EQ.IT2H.AND.ITS.LT.IT2S)GO TO 355
GO TO 370
355  IND1=IND2
    ITH=IT2H
    ITIS=IT2S
    IND2=JBACK(IND2)
    IT2H=JTIME(IND2,1)
    IT2S=JTIME(IND2,2)
360  CONTINUE
    ICHECK=2
    GOTO 400
370  JFORE(IND2)=JEMPTY
    JBACK(JEMPTY)=IND2
    JFORE(JEMPTY)=IND1
    JBACK(IND1)=JEMPTY
C PERFORM THE FOLLOWING FOR ALL EVENTS
380  CONTINUE
    JTIME(JEMPTY,1)=ITH
    JTIME(JEMPTY,2)=ITS
    NUMEVT=NUMEVT+1
    NEMPTY=NEMPTY-1
    JEMPTY=LSAVE
400  RETURN
END
e. SUBROUTINE: ASP

PURPOSE: Services the unit trucks from the queues and maintains Ammunition Supply Point (ASP) bookkeeping.

COMMON BLOCKS: LOG

CALLS:
GETQUE
INTROK
OPERA
SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS

IPARM(5) - (1) -- 1 if routine queue, 2 if GSRS queue
                - (2) -- ASP Number

LOCAL ARRAYS: None

FUNCTIONS:

Retrieve truck from queues; If no truck, have false event, schedule another and return.

Determine ammunition mix, load time.

Record ammunition used by incrementing IASPAM and decrementing IASP.

Compute travel time.

Check for truck failure and/or interdiction.

Schedule truck to arrive back at the unit.

Schedule the next ASP event for this queue.
**** I1ND 100 -- Service of truck from queue at asc.

**** I, 100 -- 

**** IAR(1) -- 1 = Queue, 0 = last

**** IAR(2) -- asc number

**** Schedul -s -- Unit arrival of truck at unit

**** (1) Takes truck out of its que.

**** (2) Calculates load time as function of load mix

**** Numbers and queues of services active for this queue.

**** Checks -- delay due to unit and interpretation.

**** LOCAL VARIABLE DEFINITION

**** NU61 -- nu, of units to be serviced

**** Ii, 0 = truck to be served

**** Ii, 6 = number of services in the queue

**** NUNI -- number of unit owning truck

**** IINC -- number of load from queue to number of amino acids

**** IIX -- amino acid index to access unit

**** THAD -- time for a seq for truck to load one truck

**** STA -- truck to a service

**** NU51 -- unit input during the truck

**** NU52 -- number of units and unit

**** D -- asc to unit travel time

**** HIAD -- unit added to failn

**** KAD -- time of unit to interpretation

**** KAD -- time of unit to interpretation

**** I1ND -- time of unit to interpretation

**** I1ND = I1ND0
GO TO 100
10 CALL 0, TEMP(1), TEMP(1)
20 CALL 0, TEMP(2), TEMP(2)
30 CONTINUE
40 IF (IT > 5000, 50, 50) CALL TEMP(1)
50 IF (IT > 1500, 50, 50) CALL TEMP(2)
60 IF (IT > 1000, 50, 50) CALL TEMP(3)
70 IF (IT > 500, 50, 50) CALL TEMP(4)
80 IF (IT > 100, 50, 50) CALL TEMP(5)
90 IF (IT > 50, 50, 50) CALL TEMP(6)
100 IF (IT > 10, 50, 50) CALL TEMP(7)
110 IF (IT > 5, 50, 50) CALL TEMP(8)
120 IF (IT > 1, 50, 50) CALL TEMP(9)
130 RETURN
140 END
f. SUBROUTINE: ASPAR 1

PURPOSE: Processes an Ammunition Transfer Point (ATP) truck arriving at the Ammunition Resupply Point (ASP).

COMMON BLOCKS: LOG

CALLS: INTRDK
       OPERA
       SCHED

IS CALLED BY: ARM Driver.

CALLING PARAMETERS: IPARM(5) - (1) -- ATP Number
                     (2) -- TRUCK Number
                     (3) -- ASP Number

LOCAL ARRAYS: None

FUNCTIONS:

Determine the type ammunition on the truck and decrement IASP and increment IASPAM.

Check for truck failure and interdiction.

Schedule arrival back at the ATP as appropriate.
SUBROUTINE ASPARI (IPARM)

C***** EVENT ASPARI -- ARRIVAL OF ASP TRUCK AT ASP (FROM ATP)

C****

J. FOX JAN 79

C****

IPARM(1) -- ATP NUMBER

IPARM(2) -- TRUCK NUMBER

IPARM(3) -- ASP NUMBER

C

C**** SCHEDULES -- ATPAR2, ARRIVAL OF ASP TRUCK AT ATP

C**** CHECKS -- DELAY DUE TO MTBF AND INTERDICTION

COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IIRSTME (20,3), IATPSD(5), IDAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILING, LOOK(17)

DIMENSION IPARM(5)

LOCAL VARIABLES DEFINITION

20 MIX - AMMO ON TRUCK INDEX TO IMIX

IND - INDEX FOR AMMO INVENTORY CONTROL IN IASP

TRTM - TRAVEL TIME TO ATP

JLOOP - DO LOOP TOP FROM LPPAR EQUAL TO NUMB OF AMMO CD AT ATP

TKSP - TRUCK SPEED

ITKTYP - TRUCK TYPE

ITRUCK (IPARM(2),I)

IFAIL - TIME LOST DUE TO TRUCK FAILURE

TMIND - TIME LOST DUE TO INTERDICTION

TMLD - TIME TO LOAD AMMO AT ASP

TOTTIM - TIME OF ARRIVAL AT ATP

30 FIND AMMO MIX INDEX ON THE TRUCK - MIX

MIX = ITRUCK (IPARM(2),5)

DECREMENT ASP AMMO

JLOOP = LPPAR(2)

DO 5 I = 1,JLOOP

IND = I+13

IASP(IPARM(3), IND) = IASP(IPARM(3),IND) - IMIX(MIX,I)

INCREMENT AMMO USED FROM ASP

IASPAM (IPARM(3),I) = IASPAM(IPARM(3),I) + IMIX(MIX,I)

5 CONTINUE

SCHEDULE ATPAR2, COMPUTE NECESSARY PARAMETERS

ITKTYP = ITRUCK (IPARM(2),I)

TKSP = ITYPE(ITKTYP,IDAY+3)

TRTM = 60 * IASP(IPARM(3),I) / TKSP

COMPUTE TIME LOST DUE TO TRUCK FAILURE

CALL OPERA(IPARM(2),TRTM,ITFAIL)

COMPUTE INTERDICTION TIME LOST
CALL INTRDK(IPARM(2),TMIND)

C CONSIDER LOAD TIME AT ASP WHICH MIGHT BE ZERO
TMLD = IMIX(MIX,23)

50  C**** IF NO INTERDICTION, BYPASS
    IF (TMIND .LE. 0)GO TO 15
C**** DECREMENT AMMO AGAIN SINCE LOST A TRUCK LOAD
C**** ADD ANOTHER LOAD TIME
JLOOP = LPPAR(2)

55  DO 10 I = 1,JLOOP
    IND = I + 13
    IASP(IPARM(3),IND) = IASP(IPARM(3),IND) - IMIX(MIX,I)
    IASPAM(IPARM(3),I) = IASPAM(IPARM(3),I) + IMIX(MIX,I)
10  CONTINUE

60  TMIND = TMIND + TMLD
15  CONTINUE

C SCHEDULE ARRIVAL AT ATP AT TIME TOTTIM
TOTTIM = TIME +TRIM + TMIND + TFAIL + TMLD
ITRUCK(IPARM(2),6) = 100

65  CALL SCHED (11,IPARM,TOTTIM)

C
RETURN
END
g. SUBROUTINE: ASPARV

PURPOSE: To process the arrival of a unit truck at the Ammunition Resupply Point (ASP)

COMMON BLOCKS: LOG

CALLS: PUTQUE, SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- Unit Number
(2) -- Truck Number
(3) -- ASP Number

LOCAL ARRAYS: None

FUNCTIONS:

- Determines ammunition mix on truck.
- Determines if truck should be in GSRS or routine queue.
- Places truck in proper queue.
- Schedules ASP event if this is the first truck in the routine queue or is a GSRS truck.
LCLCJ

**** UNIT SERVICE -- SINGLE UNIT TRUCK AT ASP

**** 1. FAX  JAN 79

**** IP(I) -- UNIT NUMBER
**** IT(I) -- "TRUCK NUMBER"
**** IA(I) -- ASP NUMBER

*** THIS UNIT TRUCK IN PROPER ASP QUEUE.

**** SCHEDULES -- ASP SERVICE OF UNIT TRUCK FROM QUEUE AT ASP

(If ASP service for this queue is TOLI)

**COMMON** IAT(I),(T04,70), (IASP(IV+1)), IUNIT(75,69),
7 ITUCK(669,7), ITYPE(16,1), IPIX(40.23), INTQ(T),
7 ICOUNT(22,1), IATPQ(3), IAY, ITMP,
7 ICOUNT(2), ITMP(2), IASPB(4,20), LQOUT, TOST, TOILNG, LOOK(17)
DIMENSION IATPQ(3)

**LOCAL VARIABLES**
7 IA = IN THE INDEX OF THE ASP TYPE FROM IMIX
7 IINDEX = QUEUE FOR TRUCK
7 IIND = INDEX TO COUNT TRUCKS QUEUED
7 OUT = INDEX OF MY INDEX.
7 MIX = ITUCK(IASSP(3),9)
7 VSUM, NO 6.5, IN MIX
7 INDE = 1043, IASPB(3)
7 IND = 12
7 TOLI
7 INDE = 12
7 INDE = 11
CALL ROUTINE (ENDRT, INDEX)

CHANGE THE STATUS OF THE TRUCK

IF (NUMTRUCK(N) > INDEX) = INT (NUMTRUCK(N) + 1)

STOP

**** IF QUICKLY DETERMINE SCHEDULE ASP NOW ELSE GO TO 10

IF (FLAG(N) > 10) = 10

GO TO 10

**** IF GIVES DO NOT SET FLAG, SINCE EACH TRUCK HAS OWN SERVER

IF (INDEX(N) = 10) = 10

GO TO 10

STOP

SCHEDULE ASP NEXT

IF (INDEX(N) = 1)

IF (INDEX(N) > 12) = 12

GO TO 10

STOP

END
h. SUBROUTINE: ATP

PURPOSE: Services a unit truck waiting in the Ammunition Transfer Point (ATP) queue and updates the bookkeeping files as to ATP status.

COMMON BLOCKS: LOG

CALLS: FINTK
       GETQUE
       INTRDK
       IQ
       LDPWDR
       OPERA
       PUTQUE
       SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) -- 1 if artillery queue, 2 if maneuver queue
                    (2) -- ATP Number

LOCAL ARRAYS: IIPARM(5) - Used to schedule other events.

FUNCTIONS:

Determine if servers require shifting from one queue to another.

Obtain truck from queue; if no truck schedule another look (false event) 5 minutes later and return.

Determine the type of ammunition needed.

Unload from ASP-ATP truck if available, else unload from CSA-ATP truck

If empty ASP-ATP or CSA-ATP truck send for refill.

If artillery ammunition (4 or 5), load powder cylinders (type 3) also, schedule truck back to unit.

Check failure and interdiction for all trucks leaving the ATP.
SUBROUTINE ATP (IPARM)
C***** EVENT ATP -- SERVICE OF TRUCK FROM QUEUE AT ATP.
C
C***** J. FOX    JAN 79
C
C***** IPARM(1) -- 1 = ARTILLERY QUEUE, 2 = MANEUVER QUEUE
C***** IPARM(2) -- ATP NUMBER
C***** SCHEDULES -- CSAARV, ARRIVAL OF CSA-ATP TRUCK AT CSA
C     UNTARV, ARRIVAL OF TRUCK AT UNIT
C     ASPAR1, ARRIVAL OF ASP-ATP TRUCK AT ASP
C     ATP, SERVICE OF TRUCK FROM QUEUE AT ATP
C
C***** (1) TAKES TRUCK OUT OF ITS QUEUE
C***** (2) CALCULATES LOAD TIME AS FUNCTION OF LOAD MIX
C***** NUMBER AND NUMBER OF SERVERS ACTIVE FOR THIS QUEUE.
C
C***** CHECKS -- DELAY IN ARRIVAL DUE TO MTBF AND INTERDICTION.
    COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
    Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
    Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
    $ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
D ITRUCK(5)
C LOCAL VARIABLE DEFINITION
C NUMQ - QUEUE TO BE SERVED
C NUMTK - TRUCK TO BE SERVED
C NUMART - NUMBER OF ARTY QUEUE SERVERS
C NUMMAN - NUMBER OF MANEUVER AMMO SERVERS
C NINC - NUMBER OF FORKLIFT FROM INACTIVE TO ACTIVE
C MIX - INDEX OF AMMO MIX ON TRUCK
C NRNDNSN - NUMBER OF ROUNDS NEEDED BY THE TRUCK NUMTK
C IRNTYP - TYPE OF ROUNDS NEEDED BY NUMTK
C IPROG - EVENT TYPE TO BE SCHEDULED
C JLOOP - DO LOOP TOP FROM LPPAR = NUM OF AMMO CD AT ATP
C NASP - ASP NUMBER THAT THIS ATP BELONGS TO
C NFKLK - NUMBER OF FORK Lifts SERVING QUEUE
C NRND - NUMBER OF POWDER CHARGES NEEDED
C NASPQ -- NUMBER OF THE ASP-:ATP TRUCK QUEUE
C NASTK - NUMBER OF ASP ATP TRUCK
C NRONTK - NUMBER OF ROUNDS ON SUPPLY TRUCK
C MIXX - MIX INDEX OF AMMO ON SUPPLY TRUCK
C DIST - ROAD DIST TO BE TRAVELED
C ITKTyp = TRUCK TYPE
C TRIM - ROAD TRAVEL TIME
C TFAIL - TIME DELAY DUE TO FAILURE
C TMIND - TIME DELAY DUE TO INTERDICTION
C TOTTIM - TIME TO SCHEDULE ATP OR ASP ARRIVAL
C TPAR - TIME REQUIRED TO SHIFT A PARTIAL LOAD
C FRNA - FLOATING POINT NUMBER FOR ROUNDS AVAILABLE FOR THE PARTIAL
C FRNN - REAL VARIABLE FOR NUMBER OF ROUNDS NEEDED
C NCSAQ - CSA ATP QUEUE NUMBER
C TLOAD - LOAD TIME
DIMENSION IIPRAM(5)
DO 1 1 =1,5
IIPRAM(I) = 0
1 CONTINUE
NUMART = IATP(IPARM(2),9)
NUMMAN = IATP(IPARM(2),10)
NTOTWK = IATP(IPARM(2),10) + IATP(IPARM(2),9)
NINC = 0
C QUEUE THRESHOLD LOGIC
C IF NEITHER QUEUE IS LONGER THAN THRESHOLD, NO CHANGE(90)
IF(IATP(IPARM(2),14) .LT. IATPSD(2) .AND. IATP(IPARM(2),15)
Z .LT. IATPSD(4))GO TO 90
C IF NOT ABOVE THRESHOLD 2 AND OTHER GT 0, NO CHANGE(90)
IF(IATP(IPARM(2),14) .LT. IATPSD(3) .AND. IATP(IPARM(2),15)
Z .GT. 0)GO TO 2
C NEED TO CHANGE(5)
GO TO 5
2 IF(IATP(IPARM(2),15) .LT. IATPSD(5) .AND. IATP(IPARM(2),14)
Z .GT. 0)GO TO 20
NUMMAN = NUMMAN + NUMART
WRITE(LUOUT,6) NUMART, NUMMAN
6 FORMAT(I6," ARTY SERVERS HAVE MOVED TO HELP ",I4," MANVR SERVERS")
GO TO 30
10 IF(IATP(IPARM(2),15) .GT. 0)GO TO 20
C MANEUVER QUEUE EMPTY SHIFT SERVERS
NUMART = NUMART + NUMMAN
WRITE(LUOUT,15) NUMMAN, NUMART
15 FORMAT(I6," MNVR SERVERS HAVE MOVED TO HELP ",I4," ARTY SERVERS")
C IF ARTY GT THRESHOLD 2 WAKE UP SERVERS
20 IF(IATP(IPARM(2),14) .LT. IATPSD(3))GO TO 30
NINC = IATP(IPARM(2),9)*(IATPSD(1)-NTOTWK)/(1+NTOTWK)+1
NUMART = NUMART + NINC
WRITE(LUOUT,25) NUMART
25 FORMAT(" DUE TO THRESHOLD 2 ON ARTY," ,I4," SERVERS ARE NOW AWAKE")
GO TO 90
30 IF(IATP(IPARM(2),15) .LT. IATPSD(5))GO TO 90
KINC = IATP(IPARM(2),10)*(IATPSD(1)-NTOTWK)/(1+NTOTWK) + 1
NUMMAN = NUMMAN + KINC
WRITE(LUOUT,35) NUMMAN
35 FORMAT(" DUE TO THRESHOLD 2 ON MNVR," ,I4," SERVERS ARE NOW AWAKE")
IF(NINC .GT. 0)KINC = IATPSD(1) - (NUMMAN + NUMART)
NUMMAN = KINC + NUMMAN
C DETERMINE QUEUE NUMBER NUMQ
90 NUMQ = IQ(IPARM(1)+3,IPARM(2))
C REMOVE TRUCK FROM QUEUE
CALL GETQUE(NUMTK,NUMQ)
C**** CHECK FOR FALSE EVENT, NUMTK=0
IF(NUMTK.GT.0) GO TO 95
C**** HAVE FALSE EVENT SCHEDULE NEXT FALSE EVENT
TOTIM = TIME + 10.
CALL SCHED(6,IPARM,TOTIM)
RETURN
95 CONTINUE
C FIND AMMO MIX INDEX OF TRUCK MIX
MIX = ITRUCK(NUMTK,5)
C FIND AMMO TYPE WANTED. ASSUME ONLY ONE TYPE
JLOOP = LPPAR(2)
DO 100 I = 1,JLOOP
IF(IMIX(MIX,I) .GT. 0)GO TO 120
100 CONTINUE
C IF EXIT HERE NO AMMO IN THIS MIX.
WRITE(LUOUT,105)MIX
105 FORMAT(" MIX ",15," CONTAINS NO TYPES OF AMMO - ATP ")
RETURN
C CHARGE NUMBER OF ROUNDS NEEDED - NRNDSN AND TYPE OF ROUNDS
120 NRNDSN = IMIX(MIX,I)
IRNTYP = I
C NOW TO LOCATE TRUCK CONTAINING PROPER TYPE OF AMMO
C FIRST CHECK ASP TRUCKS. PASS AMMO AND QUEUE TO CHECK.
NASPQ = IQ(3,IPAW(2))
130 CALL FINTK(NASPQ,IRNTYP,NASTK)
C IF NO TRUCK, GO TO 140
IF(NASTK .EQ. 0)GO TO 140
C FIND THE NUMBER OF ROUNDS ON NASTK. IF SUFFICIENT, DECREMENT
C AMMO, SCHEDULE UNTARV, PUT TRUCK BACK IN ASP Q.
C IF INSUFFICIENT EMPTY ASP TRUCK, SENT TO ASP, DECREMENT
C THE NUMBER OF ROUNDS REQUIRED, FIND ANOTHER TRUCK WITH
C THE PROPER AMMO
C UPDATE PER CENT POUNDS ON THE TRUCK
MIXX = ITRUCK(NASTK,5)
NRONTK = (IMIX(MIXX,IRNTYP) * ITRUCK(NASTK,6) + 99) / 100
WRITE(LUOUT,300)MIX,MIXX,IRNTYP,NRONTK,NRNDSN,NUMTK,NASTK,NASPQ
300 FORMAT(" IATP ",816)
C IF INSUFFICIENT ROUNDS GO TO 150
IF(NRNDSN .GT. NRONTK)GO TO 150
C SUFFICIENT AMMO ON TRUCK. DECREMENT AMMO ON TRUCK.
C IF ARTY AMMO CALL LOAD POWDER TRUCK
NRND = IMIX(MIX,IRNTYP)
IF(IRNTYP .GT. LPPAR(3))CALL LDPWDR(NRND,IPARM)
ITRUCK(NASTK,6) = 100 * (NRONTK - NRNDSN) / IMIX(MIXX,IRNTYP)
C PUT TRUCK BACK IN QUEUE
C**** IF TRUCK IS EXACTLY EMPTY DO NOT PUT INTO QUEUE
IF(ITRUCK(NASTK,6) .EQ. 0)GO TO 150
CALL PUTQUE(NASTK,NASPQ)
C GO TO SCHEDULE UNTARV
GO TO 200
**** INSUFFICIENT AMMO OR EXACTLY ENOUGH AMMO ON S AND P
C TIME TO SHIFT PARTIAL LOAD
150 FRNN = NRNDSN
FRNA = NRONTK
TPAR = IMIX(MIX,22) * FRNA / FRNN
IF (IDAY .EQ. 0) TPAR = 1.54*TPAR
NRNDSN = NRNDSN - NRONTK
ITRUCK(NASTK,6) = 0
C DETERMINE ASPARI FOR NASTK
C
DIST = IATP(IPARM(2),2)
IF(NASPQ .EQ. IQ(2,IPARM(2)))DIST = IATP(IPARM(2),1)
ITKYP = ITRUCK(NASTK,1)
TRTM = 60 * DIST / ITYPE(ITKYP,IDAY+3)
ITRUCK(NASTK,3) = 4
C COMPUTE DELAY DUE TO FAILURE - TFAIL
CALL OPERA(NASTK,TRTM,TFAIL)
C INTERDICTION DELAY - TMIND
CALL INTRDK(NASTK,TMIND)
TOTTIM = TRTM + TIME + TFAIL + TMIND + TPAR
IIPRAM(1) = IPARM(2)
IIPRAM(2) = NASTK
IIPRAM(3) = IATP(IPARM(2),6)
C ASSUME ASP-ATP TRUCK
IPROG=12
C IF NASTQ THE CSA-ASP QUEUE THEN CHANGE CALL
IF(NASPQ .NE. IQ(2,IPARM(2))) GO TO 143
IPROG = 9
IIPRAM(3) = 1
143 CALL SCHED(IPROG, IIPRAM, TOTTIM)
C**** IF EXACTLY ENOUGH ROUNDS ON TRUCK, SEND TRUCK BACK TO UNIT
IF(NRNDSN.EQ.0) GO TO 200
C
C GO GET ANOTHER ASP-ATP TRUCK TO COMPLETE THE LOAD
C GO TO 130
C NA ASP-ATP TRUCK SO TRY CAS ATP TRUCK
C**** IF HAVE LOOKED AT CSA QUEUE, THERE IS NO AMMO GO TO 142, TRUCK LOS
140 IF(NASPQ.EQ.IQ(2,IPARM(2))) GO TO 142
NASPQ=IQ(2,IPARM(2))
GO TO 130

C**** WRITE FLAG
142 WRITE(2,142) IPARM(2), IRNTYP, TIME
141 FORMAT(" ATP NUMB ",12," IS OUT OF AMMO ",14," AT TIME ",F8.2)
RETURN
C
C HAVE SUFFICIENT AMMO, SCHEDULE UNTARV AND NEXT ATP DECREMENT
200 IATP(IPARM(2),IPARM(1)+13) = IATP(IPARM(2),IPARM(1)+13) - 1
C DECREMENT AMMO
IATP(IPARM(2),IRNTYP*3+13) = IATP(IPARM(2),IRNTYP*3+13) - 1
Z = IMIX(MIX,IRNTYP)
IATP(IPARM(2),IRNTYP*3+14) = IATP(IPARM(2),IRNTYP*3+14) - Z
IMIX(MIX,IRNTYP)
NFKLF = NUMART
IF(IPARM(1) .EQ. 2)NFKLF = NUMMAN
TLOAD=IMIX(MIX,22)/NFKLF
IF (IDAY.EQ.0) TLOAD = 1.54*TLOAD
TOTTIM = TIME + TLOAD
IF(IATP(IPARM(2),IPARM(1)+13) .GE. 1)CALL SCHED(6,IPARM, TOTTIM)
C**** IF QUEUE IS EMPTY SCHEDULE FALSE EVENT
TOTTIM=TOTTIM + 5.
IF(IATP(IPARM(2),IPARM(1)+13).EQ.0) CALL SCHED(6,IPARM,TOTIM)
C SCHEDULE UNTARV
ITRUCK(NUMTK,3) = 4
CALL INTRDK(NUMTK,TMIND)
C IF NO INTERDICTION, BYPASS
IF(TMIND .LE. 0) GO TO 160
C DECREMENT AMMO
NASP = IATP(IPARM(2),6)
IASPAM(NASP,IRNTYP) = IASPAM(NASP,IRNTYP) + IMIX(MIX,IRNTYP)
IASP(NASP,IRNTYP+13) = IASP(NASP,IRNTYP+13) - IMIX(MIX,IRNTYP)
TMIND = TMIND + IMIX(MIX,23)
160 CONTINUE
IPARM(1) = ITRUCK(NUMTK,4)
IPARM(2) = NUMTK
DIST = IUNIT(IPARM(1),4)
ITKYP = ITRUCK(NUMTK,1)
TRTM = 60 * DIST / ITYPE(ITKYP,IDAY+1)
CALL OPERA(NUMTK,TRTM,TFAIL)
TLOAD = IMIX(MIX,22)
IF (IDAY .EQ. 0) TLOAD = 1.54*TLOAD
TOTTIM = TIME + TRTM + TFAIL + TMIND + TLOAD
CALL SCHED(8,IPARM,TOTTIM)
ITRUCK(IPARM(2),6) = 100
RETURN
END
1. SUBROUTINE: ATPARV

PURPOSE: Processes the arrival of the unit truck at the Ammunition Transfer Point (ATP).

COMMON BLOCKS: LOG

CALLS: PUTQUE
       SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- Unit Number
                     (2) -- Truck Number
                     (3) -- ATP Number

LOCAL ARRAYS: None.

FUNCTIONS:

Determine ammunition needed by the unit truck.

If ammunition is not available at the ATP send truck to the ASP(ASPARV).

If ammunition is available at the ATP place truck in the ATP queue.

If first truck in the ATP queue, schedule an ATP event.
SUBROUTINE ATPARV (IPARM)
C**** EVENT ATPARV -- ARRIVAL OF UNIT TRUCK AT ATP
C
C**** J. FOX JAN 79
C
C**** IPARM(1) -- UNIT NUMBER
C**** IPARM(2) -- TRUCK NUMBER
C**** IPARM(3) -- ATP NUMBER
C
C**** SCHEDULES -- ASPARV, ARRIVAL OF UNIT TRUCK AT ASP
C (IF AMMO IS NOT CURRENTLY ON HAND FOR ALL TRUCKS IN QUEUE)
C -- ATP, SERVICE OF UNIT TRUCK FROM QUEUE AT ATP
C (IF ATP SERVICE WAS IDLE FOR THIS QUEUE)
C
C**** DATA REQUIRED -- AMMO REQUIRED BY TRUCKS IN QUEUE.
C
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
DIMENSION IPARM(5)
C
LOCAL VARIABLES DEFINED
C JLOOP - TOP OF DO LOOP FROM COMMON LPPAR
C NUMQ - ATP QUEUE FOR ARTY OR ROUTINE SERVICE
C MIX - INDEX OF AMMO MIX USED TO ACCESS IMIX.
C NEEDTK - NUMBER OF ROUNDS NEEDED TYPE I BY UNIT TRUCK.
C INDEX - INDEX COMPUTED FOR AMMO TYPE I TO ACCESS ONHAND AND WANTED BY TRUCK IN QUEUE.
C JONHAND - AMOUNT OF AMMO TYPE I PRESENTLY ON HAND AT ATP
C NEEDOT - AMOUNT OF AMMO I NEEDED BY OTHER TRUCKS IN QUEUE.
C MANART - FLAG SET TO 2 IF MANEUVER AMMO, 1 IF ARTY AMMO
C DIST - DIST FROM ASP TO ATP.
C RATE - TRUCK MOVEMENT SPEED
C ITKIYP - TRUCK TYPE FROM ITRUCK.
C TRTM - UNOPPOSED TRAVEL TIME.
C TFAIL - TRAVEL TIME INCREMENT DUE TO MECHANICAL FAILURE
C TMIND - TRAVEL TIME INCREMENT DUE TO INTERDICTION
C TOLRDS - TOTAL RDS NEEDED BY ALL ARMY TRKS
C TOTTIM - TIME OF TRUCK ARRIVAL AT ASP
C
Determine AMMO MIX WANTED BY THE TRUCK.
MIX = ITRUCK(IPARM(2),5)
IF(MIX.GT.0) GO TO 1
WRITE(2,2) IPARM(2)
2 FORMAT(" ATPARV -- ZERO MIX ON TRUCK ", 14)
RETURN
1 CONTINUE
C SINCE AT ATP CHECK FOR ATP AMMO 1 THRU LPPAR(2)
JLOOP = LPPAR(2)
DO 5 I = 1,JLOOP
C IF NO AMMO I IN MIX GO TO 5.
IF(IMIX,1).EQ. 0)GO TO 5

50
C AMMO I IS NEEDED HOW MUCH
NEEDTK = IMIX(MIX, I)
C ASSUME MANEUVER AMMO.
MANART = 2
C IF ARTY RESET MANART
IF(I .GT. LPAR(3)) MANART = 1
C HOW MANY ROUNDS ARE NEEDED BY THE OTHER TRUCKS IN THE QUEUE
INDEX = 15 + 3*I - 1
NEEDOT = IATP(IPARM(3), INDEX)
C HOW MANY ROUNDS I ARE AT ATP - JONHND
JONHND = IATP(IPARM(3), INDEX - 1)
C IF INSUFFICIENT ON HAND GO TO 4
IF(JONHND .LT. NEEDOT + NEEDTK) GO TO 4
C IF NOT ARTY GO TO 5
IF(MANART .EQ. 2) GO TO 5
C HAVE ARTY IS THERE SUFFICIENT POWDER
C HOW MANY RDS ARE NEEDED BY ALL ARTY TRKS IN QUEUE
TOLRDS = IATP(IPARM(3), 26) + IATP(IPARM(3), 29)
IF(IATP(IPARM(3), 22) .GE. TOLRDS + NEEDTK) GO TO 5
C INSUFFICIENT AMMO SEND TO ASP
C FIND DIST TO ASP
4 DIST = IATP(IPARM(3), 2)
C FIND TRUCK RATE OF MOVEMENT - RATE
ITKYP = ITRUCK(IPARM(2), 1)
RATE = ITYPE(ITKYP, IDAY + 3)
TRTM = DIST / RATE * 60.
C CHANGE TRUCK STATUS CODE
ITRUCK(IPARM(2), 3) = 5
C COMPUTE DELAY DUE TO FAILURE - TFAIL
CALL OPERA(IPARM(2), TRTM, TFAIL)
C COMPUTE INTERDICTICN DELAY - TMIND
CALL INTRD(IPARM(2), TMIND)
C ICOMPUTE ASP ARRIVAL TIME - TOTTIM
TOTTIM = TIME + TRTM + TFAIL + TMIND
IPARM(3) = IUNIT(IPARM(1), 3)
CALL SCHED(5, IPARMTOTTIM)
GO TO 25
5 CONTINUE
C HAVE AMMO ON HAND
C FIND QUEUE NUMBER - NUMQ
NUMQ = IATP(IPARM(3), MANART + 10)
CALL PUTQUE(IPARM(2), NUMQ)
ITRUCK(IPARM(2), 3) = 2
C ADD TO QUEUE DEMAND FOR AMMO TYPE
JLOOP = LPAR(2)
DO 10 I = 1, JLOOP
INDEX = 15 + 3*I - 1
IATP(IPARM(3), INDEX) = IATP(IPARM(3), INDEX) + IMIX(MIX, I)
C**** IF ARTY ADD TO POWDER , IF NOT GO TO 10
IF(MANART .EQ. 2) GO TO 10
IATP(IPARM(3), 23) = IATP(IPARM(3), 23) + IMIX(MIX, I)
10 CONTINUE
C INCREMENT NUMBER OF TRUCKS IN THE QUEUE
IATP(IPARM(3),MANART+13) = IATP(IPARM(3),MANART+13) + 1
C**** IF QUEUE HAS NOT BEEN USED SCHEDULE ATP NOW
 IFLAG=8
IF(MANART.NE.1) IFLAG=13
IF(IATP(IPARM(3),IFLAG).EQ.1) GO TO 25
IATP(IPARM(3),IFLAG)=1
IPARM(1) = MANART
IPARM(2) = IPARM(3)
CALL SCHED(6,IPARM,TIME)
25 CONTINUE
RETURN
END
j. SUBROUTINE: ATPARI

PURPOSE: Process the arrival of a CSA-ATP truck at the ATP.

COMMON BLOCKS: LOG

CALLS: IQ, PUTQUE

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- ATP Number
                     (2) -- Truck Number

LOCAL ARRAYS: None.

FUNCTIONS:

Determine the ammunition carried on the truck.

Update the ammunition available and place the truck in the CSA-ATP queue.
**A.** 

**B.**

**C.**

**D.**

**E.**

**F.**

**G.**
C****  READ THE CARD INTO QUEUE
INDEX = IO(2, IPM[11])
CALL READ (IPM[11], INDEX)

C****  ADD 3.0, 10 THAT AVAILABLE
JL000 = LOPA(5)
30 3-1 = 1, JL 10 
IND = 15 + 7*I + 2
IATP(IAP[11], IMO) = IATP(IAP[11], IND) + (IMO*X11X, I)
Z = HI-INC(IAP[11], 6) + 1)/1
GO J100

C****  UPDATE TRIP STATUS TO THE CSA-ARP QUEUE
I = J000 (IP, IO, 6), 5 = Z

*+/ 100
18
SUBROUTINE: ATPAR2

PURPOSE: Processes the arrival of an ASP-ATP truck at the Ammunition Transfer Point (ATP).

COMMON BLOCKS: LOG

CALLS: IQ, PUTQUE

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5)  
(1) -- ATP Number  
(2) -- Truck Number

LOCAL ARRAYS: None.

FUNCTIONS:

- Determines the ammunition mix on the truck.
- Updates the IATP for ammunition available.
- Places truck in ASP-ATP queue.
SUBROUTINE ADJ أفري (INPUT)

****  V.D. S.G. -- ARRIVAL OF LOOK AT ANF FROM ADP

****  I, FLX  Jan 7A

****  Mach(1) -- ATV NUMBER

****  Mach(2) -- TRUCK NUMBER

****  SCHEDULES -- NOTHING

****  PUT LOOK IN ADP-ATP GROUP

****  C0IN0. -- ATP AMMO SUPPLY

C

C C0IN0. ADJAF (5, 41), IC35 (4, 41), IUNIT (75, 40),
C  ITLOOK (60, 7), ITRIP (60, 7), IC10 (60, 23), ITLOOK (6),
C  ITTIME (30), IATP (6), JAY, PRD,
C  ICH04 (2), ICH04 (2), T0T0AM (4, 5), LCH02, TCH02, RT0AM, T0AM (17)
C
C

C**** LOCAL VARIABLES

C**** A1X -- AMM NUMBER OF AMMO CARRIED ON THE LOOK

C**** ALOOK -- BTNX OF ATLOOK FOR LOADED AMM TRUCKS

C**** ALOOK -- TOTAL AMM LOOK EXC AMM EQUAL TO 4NT OF AMM TYPES

C**** AM -- HTML FOR CURRENT AMM SUPPLY BASED ON CURRENT

C**** CURRENT INPUT

C**** FIRST LOOK ON TRUCK

C **** FLX = 1:006(FLX (2), 6)

C **** FLX (6, 3) + 1

C **** FLX (3) = 1(2)

C **** FLX (4) -- ARRAY IN TRUCK "T0M

C **** T0M
1. SUBROUTINE: CONTROL

PURPOSE: Enables interactive control to check or edit the data files, schedule control events, schedule a stop simulation event, create events, list or modify the truck assignments, and return to regular processing.

COMMON BLOCKS: LOG

CALLS: CREEVT
       EDIT
       REPORT
       SCHED
       TRKPUT

IS CALLED BY: ARM Driver

CALLING PARAMETERS: TIME -- Present model battle time.

LOCAL ARRAYS: IIPARM(5) -- Used to schedule other events.

FUNCTIONS:

  Provides menu of possible functions and requests operator's input.
  Reads operator's input and verifies input to be in the valid range.
  Branches to perform operator's requested function.
  Returns to the first function.
m. SUBROUTINE: CSAARV

PURPOSE: To receive and process a CSA-ASP or CSA-ATP truck at the Corps Supply Activity (CSA).

COMMON BLOCKS: LOG

CALLS: INTROK
       OPERA
       SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) -- (1) -- ATP Number or ASP Number
                     (2) -- Truck Number
                     (3) -- 1 if CSA-ATP Truck, 2 if CSA-ASP Truck

LOCAL ARRAYS: None.

FUNCTIONS:

   Determines ammunition mix need on the truck.

   Update ICSA for the number of rounds taken from the CSA.

   Schedule truck returning to ASP or ATP with delays for truck failure or interdiction as appropriate.
SUBROUTINE CSAARV (IPARM)
C****** EVENT CSAARV -- ARRIVAL OF TRUCK AT CSA
C
**** J. FOX JAN 79
C
C**** IPARM(1) -- ATP NUMBER OF ASP NUMBER
C**** IPARM(2) -- TRUCK NUMBER
C**** IPARM(3) -- 1 IF ATP, 2 IF ASP
C
C**** SCHEDULES -- ATPAR1, ARRIVAL OF TRUCK AT ATP
C
C**** CHANGES -- CSA AMMO SUPPLY.
C
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
S ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
DIMENSION IPARM(5)

C
C****** LOCAL VARIABLES:
C**** MIX -- AMMO MIX NUMBER ON TRUCK
C**** LDTIM -- TIME TO LOAD TRUCK
C**** DIST -- DIST BACK TO ASP OR ATP
C**** JLOOP -- TOP OF LOOP FROM LPPAR
C**** TRTM -- TRAVEL TIME
C**** ITKTyp -- TRUCK TYPE
C**** TRKSP -- TRUCK SPEED
C**** TFAIL -- DELAY ENROUTE DUE TO FAILURE
C**** TOttIM -- TIME OF ARRIVAL OF TRUCK BACK TO ATP
C**** TMIND -- INTERDICATION TIME DELAY
C
C**** FIND AMMO MIX TO BE LOADED ON TRUCK
MIX = ITRUCK(IPARM(2), 5)

C
C**** USE DO LOOP TO PROCESS EACH AMMO TO ADD TO ICSA
C**** THE AMOUNT LOADED.
JLOOP = LPPAR(I)
DO 5 I = 1,JLOOP
   ICSA(I) = ICSA(I) + IMIX(MIX, I)
5 CONTINUE

C
C**** FIND LOAD TIME FOR MIX
LDTIM = IMIX(MIX, 21)

C
C**** DETERMINE TIME TO RETURN TO ASP OR ATP
C**** (DIST, IF ATP DIST IS IN IATP)
IF(IPARM(3) .EQ. 2) GO TO 10

C
C**** ATP TRUCK
DIST = IATP(IPARM(1), 1)
GO TO 15
C
C**** ASP TRUCK SO IPARM(1) IS ASP NUMB
10 DIST = IASP(IPARM(1), 1)

C

C***** DETERMINE TYPE OF TRUCK (ITKTyp)
 15 ITKTyp = ITRUCK(IPARM(2), 1)
   TRKSP = ITYPE(ITKTyp, IDAY+3)

C

C***** CALCULATE TRAVEL TIME (TRTM)
   TRTM = DIST / TRKSP * 60.
   ITRUCK(IPARM(2),6) = 100
   ITRUCK(IPARM(2),3) = 4

C***** COMPUTE DELAY DUE TO INTERDICTION (TMIND) AND FAILURE (TFAIL)
   CALL INTRDK(IPARM(2), TMIND)
   IF(TMIND .LE. 0) GO TO 30

C***** CHARGE ADDITIONAL AMMO TO CSA
   JLOOP = LPPAR(1)
   DO 35 I = 1, JLOOP
     ICSA(I) = ICSA(I) + IMIX(MIX, I)
   35 CONTINUE

C***** INCREMENT DELAY BY LOAD TIME
   TMIND = TMIND + LDTIM

30 CONTINUE
   CALL OPERA(IPARM(2), TRTM, TFAIL)
   TOTTIM = TRTM + LDTIM + TIME + TFAIL + TMIND

C

C***** SCHEDULE ATPAR1 (IPARM IS ALREADY OK FOR ATPAR1)
C***** IF ASP TRUCK GO TO 25
   IF(IPARM(3) .EQ. 2) GO TO 25
   CALL SCHED(10, IPARM, TOTTIM)
   GO TO 20

C

C***** HERE WOULD BE LOGIC TO SCHEDULE A CSA TO ASP TRUCK
   25 CONTINUE
   WRITE(2, 100)
   STOP

C

20 RETURN

100 FORMAT(" NO LOGIC FOR CSA TO ASP LINK")
END
SUBROUTINE: DEMAND

PURPOSE: Updates the ammunition required by a unit because of a demand pulse.

COMMON BLOCKS: LOG

CALLS: OPERA
        RDIEXO
        SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- Unit Number

LOCAL ARRAYS: None.

FUNCTIONS:

Calls RDIEXO to update IUNIT with the latest demand pulse.

If UNIT is a FARP, moves ammunition from the ground available to aircraft.

If UNIT is artillery, checks to see if critical resupply exists to cause helicopter resupply to be initiated.

Schedules RELOAD event for the unit.
SUBROUTINE DEMAND (IPARM)

C**** EVENT DEMAND -- CHECKS AMMO DEMAND OF UNITS.
C
C**** D. HILLIS JAN 79
C
C**** IPARM(1) -- UNIT NUMBER
C
C**** SCHEDULES -- RELOAD, RESUPPLY OF UNITS.
C HELARV, ARRIVAL OF HELICOPTER AT UNIT
C DEMAND, CHECKS DEMAND AGAIN.
C**** LOCAL VARIABLE DEFINITIONS
C**
K - UNIT AMMO INDEX
C**** NFLAG = 0 RELOAD NOT SCHEDULED YET. 1 RELOAD ALREADY SCHEDULED
C**** IFLAG = 0 NORMAL MODE. 1 - 155 HE OR ICM AMMO BELOW CRL
C**** I - UNIT NUMBER
C**** IA - LOOP INDEX
C**** II - LOOP INDEX
C**** JLOOP - TOP OF DO LOOP FROM COMMON LPPAR
C**** TRIM - ROAD TRAVEL TIME
C**** TFAIL - TIME LOST DUE TO REMEDIAL MAINTENANCE
C**** TOTTIM - TIME TO SCHEDULE THE EVENT
C**** IRRL - ROUTINE RESUPPLY LEVEL FOR LIVE WPNS
C**** IBAM - BASIC AMMO LEVEL FOR LIVE WPNS
C**** IRGND - NO. RNDs ON GROUND AT FARP
C
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), IYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
DIMENSION IPARM(5)
I = IPARM(1)
C
CALL RDIEXO(I)
C
INITIALIZE FLAGS AND COUNTERS
IFLAG = 0
NFLAG = 0
C**** SELECT AN AMMO TYPE
DO 100 KK = 1,5
K = KK * 12 - 4
IF(IUNIT(I,K).EQ.0) GO TO 100
IBAM=IUNIT(I,K+1)*IUNIT(I,K+7)

C**** CHECK FOR A FARP
IF(IUNIT(I,1).EQ.8) GO TO 50
IF(IBAM-IUNIT(I,K+4).EQ.0) GO TO 100

C**** CHECK FOR ROUTINE RESUPPLY
IRRL=IUNIT(I,K+1)*IUNIT(I,K+5)
IF(IUNIT(I,K+4).GT.IRRL) GO TO 100
L = IUNIT(I,K)

C**** CHECK FOR 155 ARTY UNIT
IF(IUNIT(I,1).EQ.4) GO TO 65

C**** IS THERE AMMO OF THIS TYPE ON TRUCKS
35 IF(IUNIT(I,K+8).NE.0)GO TO 40

100 CONTINUE
IF(IFLAG .EQ. 1) GO TO 150
GO TO 100
C***** THERE IS AMMO ON A TRUCK
40 IF(IFLAG .EQ. 1) GO TO 110
IF(NFLAG .EQ. 1) GO TO 100
C SCHEDULE RELOAD IMMEDIATELY
CALL SCHED(2,IPARM,TIME)
NFLAG = 1
GO TO 100
C***** DETERMINE AMMO REQUIREMENT AT FARP
50 IRGND=IUNIT(I,K+4)-IBAM + IUNIT(I,K+3)
IF(IUNIT(I,K+3).GT.IRGND) GO TO 55
IRGND=IRGND-IUNIT(I,K+3)
IUNIT(I,K+4)=IRGND+IBAM
IUNIT(I,K+3)=0
IUNIT(I,K+2) = 0
WRITE(LUOUT,210) IUNIT(I,K+4),IRGND
210 FORMAT(" OD - FARP O/H- ON GRND= ",15)
GO TO 35
55 IUNIT(I,K+3)=IUNIT(I,K+3)-IRGND
IUNIT(I,K+4)=IBAM-IUNIT(I,K+3)
WRITE(LUOUT,210) IUNIT(I,K+4),IRGND
GO TO 35
C***** CHECK FOR AMMO TYPES 4 AND 5
55 IF(IUNIT(I,K) .EQ. 4 .OR. IUNIT(I,K) .EQ. 5) GO TO 70
GO TO 35
C***** CHECK TO SEE IF CURRENT AMMO SUPPLY GT CRITICAL RESUF LEVEL
70 IF(IUNIT(I,K+4) .GT. IUNIT(I,K+6)*IUNIT(I,K+1)) GO TO 35
IFLAG = 1
GO TO 35
C***** COMPARE AVAILABLE AMMO AGAINST CRL
110 IF(IUNIT(I,K+6) + IUNIT(I,K+4) .GT. IUNIT(I,K+6)*IUNIT(I,K+1))
2GO TO 120
IF(NFLAG .EQ. 1) GO TO 150
CALL SCHED(2,IPARM,TIME)
NFLAG = 1
GO TO 150
120 IF(NFLAG .EQ. 1) GO TO 130
CALL SCHED(2,IPARM,TIME)
NFLAG = 1
130 IFLAG = 0
GO TO 100
C***** HELICOPTER RESUPPLY LOGIC
C***** DOES UNIT ALREADY HAVE MAX NUMBER OF HELICOPTERS ASSIGNED
150 IF(IUNIT(I,68) .EQ. 2) GO TO 170
190 IF(LPPAR(5) .GT. 0) GO TO 180
IF(IUNIT(I,68) .EQ. 1) GO TO 160
WRITE(LUOUT,155)TIME
155 FORMAT(" AT ",F8.2," MIN. NO HELICOPTERS AVAILABLE ")
C
GO TO 170
160 WRITE(LUOUT,165)TIME
165 FORMAT(" AT ",F8.2," MIN. HELI SCHEDULED, NO OTHERS AVAIL. ")
170 IF(NFLAG .EQ. 1)GO TO 200
IFLAG = 0
GO TO 100
180 LPPAR(5) = LPPAR(5) - 1
C***
C FIND AVAILABLE HELI(MISSION = 5, STATUS = 3)
JLOOP = LPPAR(4)
DO 185 II = 1,JLOOP
IF(ITRUCK(II,2) .NE. 5)GO TO 185
IF(ITRUCK(II,3) .EQ. 6)GO TO 185
IF(ITRUCK(II,3) .EQ. 3)GO TO 175
185 CONTINUE
WRITE(LUOUT,186)
186 FORMAT(" CANNOT FIND THE AVAIL HELICOPTER-DEMAND ")
GO TO 200
C HAVE HELICOPTER II UPDATE STATUS
175 ITRUCK(II,3) = 4
C SCHEDULE ARRIVAL AT UNIT
IPARM(2) = II
C FIND TRAVEL TIME TRTM
TRTM = 60 * IUNIT(IPARM(1),5) / ITYPE(6,IDAY+1)
CALL OPERA(II,TRTM,TFAIL)
MIX=ITRUCK(II,5)
TOTTIM = TIME + TRTM + TFAIL + IMIX(MIX,23)
C INCREMENT ASP AMMO USED
JLOOP = LPPAR(1)
DO 187 IA = 1,JLOOP
IASPAM(IUNIT(I,3),IA) = IASPAM(IUNIT(I,3),IA) + IMIX(MIX,IA)
187 CONTINUE
C**** IF HELICOPTER FAILS IN ROUTE TO UNIT
C SEND ANOTHER HELICOPTER, IF AVAILABLE
C SCHED HELASP
C SET STATUS AS DOWN
IF(TFAIL .LE. 0)GO TO 188
ITRUCK(II,3)=6
CALL SCHED(14,IPARM,TOTTIM)
GO TO 190
188 IUNIT(I,68) = IUNIT(I,68) + 1
C***
CALL SCHED(13,IPARM,TOTTIM)
IATP(1,4) = IATP(1,4) + 1
IF(IUNIT(I,68) .EQ. 2)GO TO 170
C**** MIX 25 IS FOR THE CH47 HELICOPTER
IF(IMIX(25,L)+IUNIT(I,K+4).GT.IUNIT(I,K+6)*IUNIT(I,K+1))GO TO 170
GO TO 190
100 CONTINUE
200 RETURN
END
SUBROUTINE: ENDSIM

PURPOSE: Writes out LOG, QUENUM, QUEPNT to permanent file (FILE1) to give
checkpoint capability.

COMMON BLOCKS: LOG
                QUENUM
                QUEPNT

CALLS: None

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- Time of Simulation

LOCAL ARRAYS: None.

FUNCTIONS:

  Writes COMMONS to mass storage.

  Prints ending message.
SUBROUTINE: HASPAR

PURPOSE: Process the helicopter arriving at the Ammunition Supply Point (ASP) subsequent to carrying ammunition to the unit.

COMMON BLOCKS: LOG

CALLS: None.

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- None.
                       (2) -- Truck Number.

LOCAL ARRAYS: None.

FUNCTIONS:

  Increments the number of helicopters available for a mission.
  Changes the status code to "at the ASP."
  Sets percent loaded to 100% for future activities.
**COMPUTER OCCUR DUE (CUT 4)**

**EVENT OCCURS -- ARRIVAL OF U-5009 BACK AT BASE**

**I. FIX JAN 21**

**II. FIX (C) -- Track**

**III. SCHEDULE -- Nothing**

**IV. CHANGE MODEL OF U-500 IN USE**

```plaintext
C.O. 4007, VLOG, IAP(4, 71), TASP(+, 1), IUNIT (75, 69),
/ I/I K(0, 7), IY(0, 5), IFIX (50, 2, 3), IUPFPH (H),
/ L1 (1, 7), I= (50), LAY, TTX,
/ T(X,4), I= (1, 5), I= (4, 2, 3), LOUT, SET, TOLNG, L00K (Z) 
IN CASE 1 PARSER (F)
```

**V. LOCAL VARIABLES -- None**

**VI. I. LOG # NOT AVAILABLE FOR USE**

```plaintext
0 = LOG + 1
```

**VII. OTHER VARIABLES**

```plaintext
T(X,4) I= (1, 5), I= (4, 2, 3)
```

**VIII. OUTPUT**

```plaintext
I = 4, 6, 7
```
SUBROUTINE: HELARV

PURPOSE: Processes the arrival of a helicopter load of ammunition at a unit.

COMMON BLOCKS: LOG

CALLS: OPERA, SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) -
(1) -- Unit Number.
(2) -- Truck Number.

LOCAL ARRAYS: None.

FUNCTIONS:

Determines mix of ammunition carried by the helicopter.
Increments ammunition on hand at the unit (IUNIT).
Computes travel time back to the ASP.
Schedules arrival at the ASP (HASPAR).
I. RH:4) (1) TO A,**(1) TYP- 
THY(1) = HULI (1ºPAH(1),TH)
[UI (1ºPAH(1)Z) = HULI (1ºPAH(1),TH) + TUIX (HIX,TAMTYD)
[IUI (1ºPAH(1),TH-1) = HULI (1ºPAH(1),TH-1) - TUIX (HIX,TAMTYD)

2. CONT(0) 
3. SETO D AND T CONVEY 4 TO AS
4. T 4 = 41 - HUI (1ºPAH(1),TH) / HUI (S,1ºPAH(1))
5. 4. (4) DAYS: 30,100,100 WITH MECHANICAL FAILURES 
6. CALL BANK (PAH,1ºPAH(1),TH,FAIL)
7. **. DONT WANT D TO OPERATE THIS UNIT 
8. SHUT (1ºPAH(1),TH) = 1001 (1ºPAH(1),TH) = 1
9. (1ºPAH(1),TH) = 1001 (1ºPAH(1),TH) = 1
10. SETO D AND S CONVEY 4 TO AS
11. D 4 = 41 - HUI (1ºPAH(1),TH) / HUI (S,1ºPAH(1))
12. PUI (1ºPAH(1),TH) > HUI (1ºPAH(1),TH) + TUIX (HIX,TAMTYD)
13. CALL ADREAM (1ºPAH(1),TH,TAMTY)
14. STEP (-10) D AND S CONVEY 4 TO AS
15. D 4 = 41 - HUI (1ºPAH(1),TH) / HUI (S,1ºPAH(1))
16. PUI (1ºPAH(1),TH) > HUI (1ºPAH(1),TH) + TUIX (HIX,TAMTYD)
17. STOP
SUBROUTINE: RELOAD

PURPOSE: Replaces rounds expended at unit weapons from rounds on unit trucks or on the ground.

COMMON BLOCKS: LOG

CALLS: FINIK
        INTDK
        IQ
        SCHED
        MINO

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- Unit Number

LOCAL ARRAYS: None.

FUNCTIONS:

  Determines the number of rounds short at the weapons for each ammunition type.

  Checks to see if the unit has the ammunition available on trucks.

  If ammunition is available, sends trucks to weapons to reload.

  If no ammunition is available on the trucks, program checks the next ammunition type.

  If a truck is emptied, the program schedules a unit departure subsequent to unloading.

  Time lost for truck failure and interdiction losses is considered.

  If a truck is only partially emptied, the program schedules a unit arrival subsequent to reloading.
SUBROUTINE RELOAD (IPARM)
C***** EVENT RELOAD -- REPLACES ROUNDS OF AMMO AT UNIT WEAPONS.
C***** D. HILLIS     JAN 79
C***** IPARM(1) -- UNIT NUMBER
C
EVENTS SCHEDULED -- UNTDEP, DEPARTURE OF UNIT TRUCKS
UNTARV, ARRIVAL OF TRUCKS AT UNIT.
RELOAD WILL OCCUR AT THE TASK FORCE LEVEL FOR MANEUVER UNITS,
BATTERY LEVEL FOR ARTILLERY UNITS AND ADA UNITS, AND AT
BATTALION FARRP'S. THE RELOAD WILL BE CALLED FROM THE DEMAND
EVENT.
AMMO WILL BE CONSOLIDATED ON TRUCKS AT UNIT. NO MORE THAN 1 TRUCK
PER UNIT (PER TYPE OF AMMO) WILL BE AT LESS THAN FULL LOAD WHILE
LOCATED AT THE UNIT. A "SMALL LOAD" THRESHOLD MAY BE DEFINED BELOW
WHICH AMMO IS DUMPED TO GROUND TO ALLOW TRUCK TO RETURN TO ATP.

C***** LOCAL VARIABLE DEFINITION
C***** I - UNIT NUMBER
C***** K - UNIT AMMO INDEX
C***** TOTTIM - TIME OF SCHEDULED EVENT
C***** LOAD - NUMBER OF ROUNDS ON THE TRUCK
C***** DELO - AMMO DEMAND
C***** IT - TRUCK NUMBER
C***** MX - AMMO MIX INDEX
C***** L - AMMO TYPE
C***** KIND - EVENT TYPE
C***** TMIND - DELAY TIME DUE TO INTERDICTION
C***** NRPW - NUMBER OF ROUNDS PER WEAPON
C***** NW - NUMBER OF WEAPONS LOADED PER TRUCK
C***** NNW - NUMBER OF WEAPONS LOADED PER TRUCK TO HANDLE TYPE 8
C***** ICRL - FARP CRITICAL RESUPPLY LEVEL
C***** IBAM - BASIC AMMO LEVEL OF LIVE WPNS
C***** IRGND - NO. OF ROUNDS ON GROUND AT FARP
C***** IFLAG - 0 -FARP TRUCK AVAIL., 1 -NO FARP TRUCK AVAIL.
C
C**** SELECT AN AMMO TYPE
DO 100 KK=1,5
C
C
76
IFLAG=0
K=12 * KK - 4
L = IUNIT(I,K)
IF(L.EQ.0) GO TO 100
C**** IS THERE AMMO AVAILABLE ON THE TRUCKS
IF(IUNIT(I,K+8).EQ.0) GO TO 100
C**** CALCULATE AMMO DEMAND
10 ND = IUNIT(I,K+7) * IUNIT(I,K+1) - IUNIT(I,K+4)
   IF(IUNIT(I,1).EQ.0) ND = IUNIT(I,K+3)
   WRITE(LUOUT,200)L,ND
200 FORMAT(" RELOAD AFTER IQ",2IS)
   IF(ND .LE. 0) GO TO 100
C**** PULL TRUCK FROM QUEUE
20 CALL FINTK(II,L,IT)
   WRITE(LUOUT,201)IT
201 FORMAT(" RELOAD AFTER FINTK ",I5)
   IF(IUNIT(I,1).EQ.0 .AND. IT.EQ.0) GO TO 62
   IF(IUNIT(I,1).EQ.0) GO TO 30
   IF(IT.EQ.0) GO TO 100
C**** CHECK FOR INTERDICTION
   CALL INTRDK(IT,THIND)
   IF(THIND.EQ.0) GO TO 30
   TOTTIM = TIME + THIND
   IPARM(2) = IT
   IPARM(3) = IUNIT(I,1)
C**** SCHEDULE ASPARV FOR EMPTY TRUCK
   CALL SCHED(5,IPARM,TOTTIM)
   MX = ITRUCK(IT,5)
C**** DECREMENT UNIT AMMO ON TRUCKS
   IUNIT(I,K+8) = IUNIT(I,K+8) - (IMIX(MX,L) * ITRUCK(IT,6)+99)/100
C**** THIS LOGIC IS HERE TO SOLVE THE PROBLEM OF WEAPON
C**** SYSTEMS HAVING DIFFERENT BASIC LOADS FOR THE SAME AMMO
   IF(IUNIT(I,1).NE.1 .AND. IUNIT(I,1).NE.2) GO TO 20
   IF(IUNIT(I,K).NE.2) GO TO 20
   DO 2 JJ=8,56,12
   IF(K.EQ.JJ) GO TO 2
   IF(IUNIT(I,K).EQ.IUNIT(I,JJ)) GO TO 52
   2 CONTINUE
C NO EQUAL AMMO FOUND FOR 2 IN THIS UNIT GO TO 20
   GO TO 20
C 52 IUNIT(I,JJ+8) = IUNIT(I,JJ+8) - (IMIX(MX,L) * ITRUCK(IT,6)+99)/100
GO TO 20
C**** DETERMINE CRITICAL RESUPPLY AT FARP
60 ICRL = IUNIT(I,K+1) * IUNIT(I,K+6)
   IBAM = IUNIT(I,K+1)*IUNIT(I,K+7)
   WRITE(LUOUT,203)ICRL,IBAM,IUNIT(I,K+4)
203 FORMAT(" ICRL= ",5I5," BAM= ",5I5," CURRENT= ",I5)
   IF(IUNIT(I,K+4)-IBAM.GT.ICRL) GO TO 65
GO TO 20
30 MX = ITRUCK(IT,5)
C**** CALCULATE THE TRUCK AMMO LOAD
   LOAD = (IMIX(MX,L) * ITRUCK(IT,6) + 99) / 100
C**** CHECK AMMO DEMAND AGAINST TRUCK LOAD
C IF UNIT TYPE 8 UNLOAD THE WHOLE TRUCK
   IF(ND .LT. LOAD .AND. IUNIT(I,1) .NE. 8) GO TO 50
   ITRUCK(IT,6) = 0
   NEWLD = LOAD
   KIND = 3
C**** CALCULATE UNLOAD TIME FOR TRUCK
C**** CALCULATE THE NUMBER OF ROUNDS PER WEAPON
   40 NRPW = ND / IUNIT(I,K+2)
C**** CALCULATE THE NUMBER OF WEAPONS LOADED PER TRUCK
   NW = MIND(LOAD / NRPW,IUNIT(I,K+2))
C**** CALCULATE THE RELOAD TIME
   NNW = NW
   IF(IUNIT(I,1).EQ.8) NNW = 1
   IF(IUNIT(I,1).EQ.8) NRPW = LOAD
   DELAY = 2 * IRSME(L,3) + NNW * (IRSME(L,1) + Z IRSME(L,2) * NRPI / 100)
   TOTTIM = TIME + DELAY
   IPARM(2) = IT
C**** SCHEDULE A UNITARY OR UNITDEP DEPENDING ON VALUE OF KIND
   CALL SCHED(KIND,IPARM,TOTTIM)
C**** ADJUST AMMO ON TRUCKS AND CURRENT AMMO SUPPLY
   IUNIT(I,K+8) = IUNIT(I,K+8) - LOAD
C**** THIS LOGIC IS HERE TO SOLVE THE PROBLEM OF WEAPON
C**** SYSTEMS HAVING DIFFERENT BASIC LOADS FOR THE SAME AMMO
   IF(IUNIT(I,1).NE.1 .AND. IUNIT(I,1).NE.2) GO TO 45
   IF(IUNIT(I,K).NE.2) GO TO 45
   DO 1 JJ=8,56,12
   IF(K.EQ.JJ) GO TO 1
   IF(IUNIT(I,K).EQ.IUNIT(I,JJ)) GO TO 42
   CONTINUE
C NO EQUAL AMMO 2 IN THIS UNIT GO TO 45
   GO TO 45
42 IUNIT(I,JJ+8)=IUNIT(I,JJ+8)-LOAD
45 IUNIT(I,K+4) = IUNIT(I,K+4) + NEWLD
   IF(IUNIT(I,1).EQ.8) GO TO 60
C DECREMENT THE NUMBER OF ROUNDS SHORT
   IUNIT(I,K+3) = IUNIT(I,K+3) - NEWLD
   IUNIT(I,K+2) = IUNIT(I,K+2) - NW
   IF(IUNIT(I,1).EQ.8) GO TO 20
   GO TO 10
62 IFLAG = 1
65 IBAM=IUNIT(I,K+1)*IUNIT(I,K+7)
   IRGND=IUNIT(I,K+4)-IBAM
   IF(IUNIT(I,K+3).GT.IRGND) GO TO 70
   IRGND=IRGND-IUNIT(I,K+3)
   IUNIT(I,K+2)=0
   IUNIT(I,K+3)=C
   IUNIT(I,K+4)=IBAM+IRGND
   GO TO 100
70 IUNIT(I,K+3)=IUNIT(I,K+3)-IRGND
IUNIT(I,K+4)=IBAM-IUNIT(I,K+3)
IF(IFLAG.NE.1) GO TO 60
GO TO 100
C**** CALCULATE THE PARTIAL LOAD OF THE TRUCK
50 ITRUCK(IT,6) = 100 * (LOAD - ND)/IMIX(MX,L)
NEWLD = ND
KIND = 8
GO TO 40
100 CONTINUE
C
RETURN
END
s. SUBROUTINE: REPORT

PURPOSE: Provides a variety of reports to the operator given the present status of the battle.

COMMON BLOCKS: LOG

CALLS: None.

IS CALLED BY: CONTRL

CALLING PARAMETERS: IPARM (5) - (1) -- Number of Report Desired

LOCAL ARRAYS: IWPN (20) -- Alpha description of the ammunition codes.

FUNCTIONS:

- Branches to the major part of the code reference by the type report requested in the CALLING PARAMETERS.
- Requests additional information from the operator as required.
- Accepts the additional input and produces the resultant report.
SUBROUTINE REPORT (IPARM)
C**** WRITES REPORTS OF VARIOUS TYPES.
C**** J FOX FEB 79
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCIILNG, LOOK(17)
DIMENSION IPARM(5), IWPN(20)
DATA IWPN/4HTANK,3HTOW,4HPWDR,5H155HE,6H155ICM,6H155SMK,
Z 7H155CLGP,5HBINHE,6HBINICM,4HGSRS,6HMORTAR,5HDIVAD,3HAAH,4HAHIG,
Z 7HSTINGER,6HDRAGON,8HBUSHMSTR/
KIND = IPARM(1)
IF(KIND .LE. 0 .OR. KIND .GT. 9)RETURN
GO TO (10, 20, 30, 40, 50, 60, 70, 80, 20), KIND
C
C**** REPORT TYPE ONE
10 CONTINUE
C**** ITRUCK UNIT TRUCK REPORT
15 WRITE(2,325)
325 FORMAT(/," UNIT TRUCK REPORT PRINT OPTIONS:",",
Z " 1 - PRINT ALL",",
Z " 2 - SINGLE UNIT",",
Z " 3 - RETURN",",
Z " ? ")
READ(1,*) IANS
IF(IANS.LT.1.OR.IANS.GT.3) GO TO 15
GO TO (150,160,170) IANS
C**** CYCLE THROUGH THE UNITS
150 DO 100 I = 1,75
C**** IF TYPE UNIT IS ZERO, CONSIDER INACTIVE BYPASS
IF(IUNIT(I,1) .EQ. 0)GO TO 100
C**** IF UNNAMED UNIT GO TO 100
IF (IUNIT(I,7) .EQ. "")GO TO 100
CALL TRUCK (N)
IF(KIND.EQ.9) RETURN
IF(IANS .EQ. 2) GO TO 420
100 CONTINUE
170 IF(KIND .EQ. 9)GO TO 20
GO TO 90
160 WRITE(2,290)
READ(1,300) NAME
IF(NAME .EQ. "0") GO TO 15
DO 110 K=1,75
IF(IUNIT(K,1).EQ.0)GO TO 110
IF(IUNIT(K,7).EQ.NAME) GO TO 115
110 CONTINUE
WRITE(2,431) NAME
GO TO 160
115 WRITE(LUOUT,200) IUNIT(K,7)
200 FORMAT(1X,///,8X, " TRUCK STATUS REPORT FOR UNIT ",A10,///,
Z " TRK NM STATUS MIX PCLOAD NXFAIL")
DO 120 K=1,560
81
IF(ITRUCK(KK,4).NE.K) GO TO 120
NXFAIL=ITYPE(ITRUCK(KK,1),5)-ITRUCK(KK,7)
WRITE(LUOUT,205) KK,ITRUCK(KK,2),ITRUCK(KK,3),ITRUCK(KK,5),
Z ITRUCK(KK,6),NXFAIL
205 FORMAT(2X,6I7)
120 CONTINUE
GO TO 160
C**** REPORT TYPE TWO
20 CONTINUE
C**** IUNIT REPORT
25 WRITE(2,285)
285 FORMAT(/,"UNIT STATUS PRINT OPTIONS: ",/,
Z "1 - PRINT ALL"/,
Z "2 - SINGLE UNIT"/,
Z "3 - RETURN"/,
Z "? ")
READ(1,*) IANS
IF(IANS.LT.1.OR.IANS.GT.3) GO TO 25
GO TO (350,420,410) IANS
C**** LOOP THROUGH UNITS
350 DO 400 I = 1,75
C**** IF NO TYPE CODE BYPASS
IF(IUNIT(I,1).EQ.0)GO TO 400
C**** IF NO UNIT NAME GO TO 400
IF(IUNIT(I,7).EQ."") GO TO 400
C**** PRINT HEADER
WRITE(LUOUT,210)
210 FORMAT(42X,"UNIT STATUS",43X,"UNIT DATA",29X,"WPN DATA",/,
Z,15X,"AMMO-CODE WPN-TYP WPN-ALIVE CUR-SUP RNDS-SHORT PCBL-W ON-
ZTRKS NO WPN SH NO SH EA TOT OMD ")
WRITE(LUOUT,215)IUNIT(I,7),IUNIT(I,1),IUNIT(I,2),IUNIT(I,4),
Z IUNIT(I,3),IUNIT(I,5),IUNIT(I,68)
215 FORMAT(1X,A10,14," SER ATP ",I2,I3," KM ",/,
Z " SER ASP ",I2,I3," K ",/,
Z " NO HELO ",I2)
DO 395 J = 1,5
JJ = 12 * J - 4
IF(IUNIT(I,JJ).EQ.0)GO TO 395
NMSHT=0
IF(IUNIT(I,JJ+2).EQ.0) GO TO 216
NMSHT = IUNIT(I,JJ+3) / IUNIT(I,JJ+2)
216 CONTINUE
IPCBL=100*IUNIT(I,JJ+4)/(IUNIT(I,JJ+1)*IUNIT(I,JJ+7))
WRITE(LUOUT,220)IUNIT(I,JJ),IWPN(IUNIT(I,JJ)),IUNIT(I,JJ+1),IUNIT(I,JJ+2),IUNIT(I,JJ+3),
Z IUNIT(I,JJ+4),IUNIT(I,JJ+2),IPCBL,IUNIT(I,JJ+8),IUNIT(I,JJ+2),NMSHT,
Z IUNIT(I,JJ+11)
220 FORMAT(18X,I3,6X,A8,I7,I8,5X,I5,4X,I6,2X,I5,8X,I4," ",I4,5X,I5)
395 CONTINUE
C**** PRINT STATUS OF UNIT TRUCKS
N = I
CALL TRUCK (N)
400 CONTINUE
410 IF(KIND .EQ. 9)GO TO 30
GO TO 90

420 WRITE(2,290)

290 FORMAT(" ENTER JIFFY UNIT ID (INPUT 0 TO EXIT) ")
READ(1,300) NAME

IF (NAME .EQ. "0") GO TO 25

300 FORMAT(A10)

DO 430 K=1,75

IF(IUNIT(K,1).EQ.0) GO TO 430

IF(IUNIT(K,7).EQ.NAME) GO TO 435

430 CONTINUE

WRITE(2,431) NAME

431 FORMAT(" UNIT ",A10," NOT FOUND")
GO TO 420

435 WRITE(LUOUT,310) NAME

310 FORMAT(/,1X,"UNIT",1X,A10,4X,"UNIT DATA",22X,"WPN DATA",/,10X,
Z "WP",10X,"RND",18X,"# WPN # RND",/,1X,"WP-TYP",2X,
Z "LIVE CUR-SUP SHORT PCBL ON-TRKS",3X,"SHORT SH EA TOT-DMD")

DO 385 KK=8,56,12

IF(IUNIT(K,KK).EQ.0) GO TO 385

NMSHT=IUNIT(K,KK+3)/IUNIT(K,KK+2)

IPCBL=100*IUNIT(K,KK+4)/(IUNIT(K,KK+1)*IUNIT(K,KK+7))

WRITE(LUOUT,320)IWPN(IUNIT(K,KK)),IUNIT(K,KK+1),IUNIT(K,KK+4),IUNIT(K,KK+8),NMSHT,
Z IUNIT(K,KK+11)

320 CONTINUE

C**** PRINT STATUS OF UNIT TRUCKS

N = K
CALL TRUCK(K)
GO TO 420

C

C**** REPORT TYPE THREE

30 CONTINUE

C**** SINGLE ATP REPORT - HOW MANY ACTIVE ATP'S?

PRINT (2,"** ENTER NUMB OF ACTIVE ATP'S (1,2,3,0R4)"
READ (1,*) NATP

IF (NATP .LT. 1 .OR. NATP .GT. 4) NATP = 4

DO 475 I = 1,NATP

WRITE(LUOUT,235)I,IATP(I,9),IATP(I,10),IATP(I,14),IATP(I,15)

235 FORMAT(/,55X," ATP STATUS ",/5X,"ATP NO ",I3,/,,
Z 10X,"QUEUE ARTY MU",/,10X,
Z "SERVERS ACTIVE",2X,I3,8X,I3,/,10X,"TRUCKS IN Q",
Z 5X,I3,8X,I3,/,10X,"AMM-CODE CUR-DMD AMT-O/H BASIC-LVL"

DO 470 J = 1,5

JJ = J * 3 + 13

WRITE(LUOUT,240)J,IATP(I,JJ),IATP(I,JJ+1),IATP(I,JJ+2)

240 FORMAT(13X,I3,4X,I4,4X,I6,4X,I4,2X,I5,1X,I5)

470 CONTINUE

475 CONTINUE

IF(KIND .EQ. 9)GO TO 40

C

GO TO 90

C
C**** REPORT TYPE 4
40 CONTINUE
C**** IASP REPORT - HOW MANY ASPS
PRINT (2,*) " ENTER NUMBER OF ACTIVE ASPS,1,2,3, OR 4"
READ (1,*) NASP
IF (NASP .LE. 0 .OR. NASP .GT. 4) NASP = 4
DO 500 I = 1, NASP
C**** OUTPUT INFO
WRITE(LUOUT,245) I, IASP(I,7), IASP(I,8), IASP(I,12), IASP(I,13)
245 FORMAT(1X,///,55X," ASP STATUS ",///,5X," ASP NO ",I3,///,
Z 15X," QUEUE ROUTINE GSRS ",//,10X," SERVERS ACTIVE ",
Z 2X,15,8X,15,//,10X," TRUCKS IN Q",5X,15,8X,15,///," INVENTORY
ZAMMO-CODE AMT-O/H")
JLOOP=LPPAR(I)
DO 495 J = 1,JLOOP
JJ = J + 13
WRITE(LUOUT,250)J, IASP(I,JJ)
250 FORMAT(19X,13,4X,18)
495 CONTINUE
500 CONTINUE
IF(KIND .EQ. 9)GO TO 50
GO TO 90
C
C**** REPORT TYPE FIVE
C**** ICSA REPORT
50 CONTINUE
WRITE(LUOUT,225)
225 FORMAT(1X,///,55X," CSA-STATUS ",///,50X," ROUNDS DRAWN FROM CSA",
Z /21X," AMMO ",10X," NUMBER-DRAWN ")
JLOOP=LPPAR(I)
DO 450 I = 1,JLOOP
WRITE(LUOUT,230)I,ICSA(I)
230 FORMAT(22X,I3,14X,I6)
450 CONTINUE
IF(KIND .EQ. 9)GO TO 60
GO TO 90
C
C**** REPORT TYPE SIX
C**** MULTIPLE ATP REPORT
60 CONTINUE
C**** LOOP THROUGH ATP'S
C**** WRITE HEADERS
WRITE(LUOUT,255)
255 FORMAT(1X,///,45X," ATP STATUS - COMMAND INFO ",/,.60X,
Z" AMMO INVENTORY",/,.5X," ATP NO QUEUE TRKS 1 2 3
Z 4 5")
DO 600 I = 1,4
WRITE(LUOUT,260)I,IATP(I,15),IATP(I,16),IATP(I,19),IATP(I,14),
Z IATP(I,22),IATP(I,25),IATP(I,28)
260 FORMAT(9X,I2," MU ",17,6X,14,16,/,15X," ARTY ",16,18X,315)
600 CONTINUE
IF(KIND.EQ. 9)GO TO 70
C
GO TO 90
C*****
C***** REPORT TYPE SEVEN
70 CONTINUE
PRINT (2,*) " ENTER NUMBER OF ACTIVE ASPS "
READ (1,*) NASP
IF (NASP.LE.0.OR.NASP.GT.4) NASP = 4
C***** AMMO REMOVED FROM ASPS
C***** WRITE HEADER
WRITE(LUOUT,265)
C LOOP THROUGH ASPS
DO 700 I = 1,NASP
WRITE (LUOUT,265) I
265 FORMAT (/,'ASP ',12,'AMMO TYPE',10X,'AMMO REMOVED')
DO 690 J = 1,20
WRITE (LUOUT,270) J,IASPAM(I,J)
270 FORMAT (40X,12,15X,17)
690 CONTINUE
700 CONTINUE
IF(KIND.EQ. 9)GO TO 80
C
go to 90
C***** REPORT TYPE EIGHT
C***** TRUCKS THAT HAVE BEEN KILLED OR HAVE BROKEN
80 CONTINUE
C***** LOOP THROUGH TRUCKS FOR DEAD
LOOP = LPPAR(4)
DO 800 I = 1,LOOP
C***** IF NOT DEAD, GO TO 800
IF(ITRUCK(I,3).NE.7)GO TO 800
C***** HAVE DEAD TRUCK, PRINT NUT
WRITE(LUOUT,275)I,IUNIT(ITRUCK(I,4),7),ITRUCK(I,1),ITRUCK(I,5)
800 CONTINUE
DO 810 I = 1,LOOP
C***** IF NOT BEING REPAIRED GO TO 810
IF(ITRUCK(I,3).NE.6)GO TO 810
C***** HAVE BROKEN TRUCK, PRINT INFO
ISAVE=IUNIT(ITRUCK(I,4),7)
IF(ITRUCK(I,2).NE.1) ISAVE="NON-UNIT"
WRITE(LUOUT,280)I,ISAVE,ITRUCK(I,1),ITRUCK(I,5)
810 CONTINUE
90 RETURN
END
SUBROUTINE: UNTARV

PURPOSE: Processes the arrival of a unit truck from the ASP, ATP or reloading event.

COMMON BLOCKS: LOG

CALLS: IQ
       PUTQUE
       SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- Unit Number
                    (2) -- Truck Number

LOCAL ARRAYS: None.

FUNCTIONS:

Determines the mix of ammunition on the truck.

Puts the truck in the unit queue.

Changes the truck status code.

Updates the ammunition available at unit trucks.

If a reload is necessary, schedules a reload.
SUBROUTINE UNTARV (IPARM)
C**** EVENT UNTARV -- ARRIVAL OF TRUCK AT UNIT.
C
C**** J. FOX JAN 79
C
C**** IPARM(1) -- UNIT NUMBER
C**** IPARM(2) -- TRUCK NUMBER
C
C**** SCHEDULES -- RELOAD IF DEMAND EXISTS.
C
C**** CHANGES -- UNIT AMMO ON TRUCKS
C
C**** LOCAL VARIABLES:
C**** MIX -- INDEX OF AMMO MIX
C**** IND -- INDEX FOR IUNIT AMMO TYPE
C**** INDEX -- AMMO TYPE FOR UNIT AMMO TYPE I
C**** JLOOP -- TOP OF LOOP FROM LPPAR
C**** NUMR -- NUMBER OF ROUNDS OF TYPE INDEX ON THE TRUCK
C**** IRESFL -- RESUPPLY FLAG (0 = NO RESUP, 1 = SCHED RESUP)
C**** IPR -- UNIT TRUCK QUEUE NUMBER
C
C**** INITIALIZE RELOAD FLAG
IRESFL = 0
C
C**** DETERMINE AMMO MIX
MIX = ITRUCK(IPARM(2), 5)
IF (MIX.GT.0) GO TO 1
WRITE(2,6) IPARM(2)
6 FORMAT(IX, "UNTARV -- ZERO MIX ON TRUCK ", I3)
RETURN
C
C**** PUT TRUCK IN UNIT QUEUE
1 IPR = IQ(1, IPARM(1))
CALL PUTQUE(IPARM(2), IPR)
C
C**** CHANGE TRUCK STATUS CODE
ITRUCK(IPARM(2), 3) = 1
C
C**** ADD AMMO TO UNIT AVAILABLE AMMO AND CHECK FOR GENERATING RELOAD
JLOOP = LPPAR(5)
DO 5 I = 1, JLOOP
IND = I*12 - 4
INDEX = IUNIT(IPARM(1), IND)
IF (INDEX.EQ.0) GO TO 5
C
C**** IF NO AMMO OF THIS TYPE ON TRUCK GO TO 5

87
NUMBER = (IMIX(MIX,INDEX) * ITRUCK(IPARM(2),6) + 99) / 100
IF(NUMR .LE. 0) GO TO 5

**** HAVE THIS TYPE OF AMMO, ADD TO UNIT
IUNIT(IPARM(1), IND+8) = IUNIT(IPARM(1),IND+8) + NUMR

**** IF NOT A FARP GO TO 4
IF(IUNIT(IPARM(1),1).NE.8) GO TO 4

**** IF NO RELOAD GO TO 5
IF(IUNIT(IPARM(1),IND+4).GT.IUNIT(IPARM(1),IND+1)*IUNIT(IPARM(1),IND+7)+IUNIT(IPARM(1),IND+1)*IUNIT(IPARM(1),IND+6)) GO TO 5

**** SET RELOAD FLAG
IRESFL=1
GO TO 5

**** IF RELOAD IS NOT REQUIRED GO TO 5; ELSE SET RELOAD FLAG=IRESFL
4 IF(IUNIT(IPARM(1),IND+7) * IUNIT(IPARM(1),IND+1) .LE. Z IUNIT(IPARM(1),IND+4))GO TO 5

**** SCHEDULE RELOAD FLAG
IRESFL = 1
5 CONTINUE
   IF(IRESFL .EQ. 1) CALL SCHED(2, IPARM, TIME)

**** HARD-WIRED DATA TO HANDLE STINGER

**** AMMO TYPE 15,MIX 11,24,OR26 MORTA
IF(MIX.NE.11) GO TO 10
   IF(MIX.NE.24) GO TO 10
   IF(MIX.NE.26) GO TO 10

**** ADD AMMO TO THE STINGER WEAPONS
IUNIT(IPARM(1),60)=IUNIT(IPARM(1),60)+9

10 CONTINUE
RETURN
END
u. SUBROUTINE: UNTDEP

PURPOSE: Processes a truck departing a unit subsequent to being emptied in a reload event.

COMMON BLOCKS: LOG

CALLS: INTROK, OPERA, SCHED

IS CALLED BY: ARM Driver

CALLING PARAMETERS: IPARM (5) - (1) -- Unit Number
(2) -- Truck Number

LOCAL ARRAYS: None.

FUNCTIONS:

Determines ammunition mix required by the truck.

Routes truck to ATP or ASP as is appropriate from the ammunition mix.

Consider truck failures and interdiction in the computation of the travel time.
SUBROUTINE UNTDEP (IPARM)
C***** EVENT UNTDEP -- DEPARTURE OF TRUCK FROM UNIT.
C
C***** J. FOX  JAN 79
C
C***** IPARM(1) -- UNIT NUMBER
C***** IPARM(2) -- TRUCK NUMBER
C
C***** SCHEDULES -- ATPARV, ARRIVAL OF TRUCK AT ATP OR
C*=  -- ASPARV, ARRIVAL OF TRUCK AT ASP
C
C***** CHECKS -- DELAY IN ARRIVAL TIME AT ATP OR ASP DUE
C  TO MTF AND INTERDICTION.
C
C***** CHANGES -- UNIT TRUCK QUEUE
C
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
  Z ITUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
  Z IRSHE(20,1), IATPSD(5), ITAY, TIME,
  $ ICOSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
DIMENSION IPARM(5)
C
C***** LOCAL VARIABLES
C***** MIX IS THE AMMO MIX INDEX
C***** DISI = DISTANCE TO ATP OR ASP
C***** TTKYPT = TRUCK TYPE
C***** TRXM = ROAD TRAVEL TIME
C***** IASPFG = LOCAL FLAG = 1 IF GOING TO ASP
C***** = 2 IF GOING TO ATP
C***** ILOW = COMPUTED FROM LPPAR = FIRST NON ATP AMMO CODE
C***** JLOPP = FROM LPPAR, NO LOOP TOP FOE NUM OF AMMO CODES
C***** TLFAIL = TIME LOST DUE TO REPAIR OR MAINTENANCE
C***** TMIND = TIME DELAY DUE TO MTF AND INTERDICTION
C***** TOTTIM = TIME OF ARRIVAL AT ATP OR ASP
C***** ILOW = LPPAR(2) + 1
C**** DETERMINE AMMO MIX INDEX
MIX = ITRUCK(IPARM(2), 5)
C
C**** IF MIX CONTAINS AMMO OTHER THAN THAT AT ATP (1-5) GO TO ASP
JLOOP = LPPAR(1)
DO 5 I = ILOW, JLOOP
IF(IMIX(MIX, I), GT, 0) GO TO 10
5 CONTINUE
C
C**** TRUCK BOUND FOR ATP. LOOK UP DISTANCE TO ATP (DIST)
DIST = IUNIT(IPARM(1), 4)
IASPFG = 2
GO TO 15
C
C**** TRUCK BOUND FOR ASP. LOOK UP DISTANCE TO ASP (DIST)
10 DIST = IUNIT(IPARM(1), 5)
IASPFG = 1
C
C**** DETERMINE TRUCK TYPE (ITKYP)
15 ITKYP = ITRUCK(IPARM(2), 1)
C

JTINE UNTDEP      73/73     OPT=1

C**** DETERMINE ROAD TRAVEL TIME
TRM = 60 * DIST / ITYPE(ITKYP, IDAY+1)
C**** UPDATE TRUCK STATUS CODE
ITRUCK(IPARM(2), 3) = 4
C
C**** COMPUTE DELAY DUE TO FAILURE (TFAIL)
CALL OPEPA (IPARM(1), TRM, TFAIL)
C***** COMPUTE DELAY DUE TO INTERDICT (THIND)
   CALL INTROK (IPARM(2), THIND)
C
C***** COMPUTE TIME OF ARRIVAL
   TOTTIM = TIME + THIND + IFAIL + TRTM
   IF (IASPFG .EQ. 2) GO TO 25
C
C***** SCHEDULE ASP ARRIVAL.
   IPARM(3) = IUNIT(IPARM(1), 3)
   CALL SCHED (5, IPARM, TOTTIM)
   GO TO 30
C
C***** SCHEDULE ATP ARRIVAL.
   25 IPARM(3) = IUNIT(IPARM(1), 2)
   CALL SCHED (4, IPARM, TOTTIM)
C
   30 RETURN
C
   END
v. SUBROUTINE: CREEVT

PURPOSE: Enables interactive creation of events to occur later in the processing cycle.

COMMON BLOCKS: LOG

CALLS: READF
       SCHED

IS CALLED BY: Control

CALLING PARAMETERS: None.

LOCAL ARRAYS: None.

FUNCTIONS:

Displays instructions to the operator as to the procedures in creating an event.

Accepts parameters for an event from the operator and schedules the event.
SUBROUTINE CREEVT
C**** ENABLES INTERACTIVE CREATION OF EVENTS SUCH AS TRUCKS
C**** TO ARRIVE AT AN ATRP FROM THE CSA IN MIO-CI.
C**** JAMES FOX ESQ. DOT. TNT, MARCH NINETEEN HUNDRED AND SEVENTY NINE
C**** LOCAL VARIABLE DEFINITION
C**** IPARM - CONTAINS THE 5 PARAMETERS OF THE EVENT
C**** INTRG - "" UP TO 6 INTEGER VALUES FROM THE CONSOLE
C**** IWORD - "" UP TO 6 ALPHA VALUES FROM THE CONSOLE
C**** REAL - "" UP TO 6 REAL VALUES FROM THE CONSOLE
C**** IEND - "" END OF INPUT CHECK
C**** TOTIM - TIME OF SCHEDULED EVENT
C**** ITYP - EVENT TYPE
         DIMENSION IPARM(5), INTRG(6), IWORD(6), REAL(6)
         DATA IHELP //"HELP"//
         DATA IEND //"END"/
         5 WRITE(2,100)
           LU1=1
        10 WRITE(2,150)
           FORMAT(1X,"?")
           CALL READF(LU1,6,INTRG,REAL,IWORD)
C**** IF END OF INPUT RETURN (200)
      IF(INOFO(1),EQ.IEND) GO TO 200
      IF(INOFO(1),EQ.IHELP) GO TO 5
**LOAD EVENT TYPE, PARAMETERS, AND TIME**

ITYP=INTGR(1)
IF(ITYP.LE.0.OR.ITYP.GT.17) GO TO 10
DO 20 I=1,5
   IPARM(I)=INTGR(I+1)
20 CONTINUE
   TOTTIM=REAL(1)
   CALL SCHED(ITYP,IPARM,TOTTIM)
GO TO 10

**FORMAT OF CREATE AN EVENT, INPUT AS A GROUP SEPARATED BY**, /
Z " COMMAS OR SPACES THE FOLLOWING 7 VALUES **", /
Z " EVENT TYPE (INTEGER VALUES BETWEEN 1 AND 17) **", /
Z " 5 PARAMETERS FOR EACH EVENT (INTEGER, ODD, DECIMAL, REAL) **", /
Z "  AND TIME (DECIMAL MINUTES, REAL) **", /
Z "  EXAMPLE: 10,1,512,0,0,0,305 **", /
Z "  CSA-TO-ATP TRUCK 512 WILL ARRIVE AT ATP AT TIME = 305 MIN **"

**RETURN**
**END**
SUBROUTINE: EDIT

PURPOSE: Enables the listing and/or modification of the data stored in the block COMMON LOG.

COMMON BLOCKS: LOG

CALLS: READF

IS CALLED BY: CONTRL

CALLING PARAMETERS: None.

LOCAL ARRAYS:

- INTGR(10) -- Storage for up to 10 integer number fields input from the console.
- REAL(10) -- Storage for up to 10 real number fields input from the console.
- IWORD(10) -- Storage for up to 10 alpha numeric fields input from the console.
- NAME(19) -- Storage for the names of the 19 arrays and variables in COMMON LOG.
- LIMIT1(19) -- Storage for the upper limit on the first index of the arrays and variables in COMMON LOG.
- LIMIT2(19) -- Storage for the upper limit on the second index of the arrays and variables in COMMON LOG.

FUNCTIONS:

- Displays to the operator a message requesting input as to what array or variable in COMMON LOG is of interest.
- Accepts from the operator the message as to which array.
- Operator then inputs whether he wishes to list or change the array.
- Program branches to the proper logic and lists or updates.
- An input of "END" exits the logic.
SUBROUTINE EDIT
C**** ALLOWS EDITING OF DATA IN COMMON LOG
C**** H. JONES  FEB 79
C**** NOTE ALL VARIABLES IN COMMON LOG ARE 2 DIMENSIONAL
  COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
  Z ITRUCK(560,7), ITPF(6,6), I1MIX(40,23), INTER(1,9),
  Z ISTRME(20,1), IATP3O(1,5),
$ IUAY(1,1), TIME(1,1), ICSA(1,20), LPPAR(1,5),
  Z IASPAH(4,20), LOUT(1,1), TCIST(1,1), TCIILNG(1,1), LOOK(1,17)
  DIMENSION  INTGR(10), REAL(10), IWORD(10)
C
  DIMENSION NAME(18), LIMIT1(18), LIMIT2(18)
  DATA NAME /"IATP", "IASP", "IUNIT", "ITRUCK", "ITYPE",
  "IIMIX", "INFR", "ISTRME", "IATPSN", "IUAY",
  "TIME", "ICSA", "LPPAF", "IASPAH", "LOUT", "TCIST",
  "TCILNG", "LOOK"/
C
  DATA LIMIT1 /4, 4, 75, 560, 6,
  Z 40, 1, 20, 1, 1,
  Z 1, 1, 1, 4, 1, 1, 1, 17/
C
  DATA LIMIT2 /30, 41, 69, 7, 6,
  Z 23, 9, 3, 5, 1,
  Z 1, 20, 5, 20, 1, 1, 1/
  DATA IEND/"END"/
  NNAMES = 18
C
  10 WRITE(2,160)
    LUI = 1
    CALL READF (LUI, 10, INTGR, REAL, IWORD)
C
C**** BRANCH ON DATA TYPE
  15 IF(IWORD(1) .EQ. IEND) GO TO 95
    GO TO 20  KTYPE = 1,NNAMES
    IF(IWORD(1) .EQ. NAME(KTYPE)) GO TO 30
  20 CONTINUE
  25 GO TO 10
C**** SET LIMITS FOR DATA TYPE
31 ILOW = INTGR(1)
   IHIGH = INTGR(2)
   IFLG = 0
   IF(ILOW .EQ. 0 .AND. IHIGH .EQ. 0) IFLG = 1
   IF(IFLG .EQ. 1) ILOW = 1
   IF(IFLG .EQ. 1) IHIGH = LIMIT1(KTYPE)
   IF(IHIGH .EQ. 0) IHIGH = ILOW
   IF(IHIGH .GT. LIMIT1(KTYPE)) IHIGH = LIMIT1(KTYPE)
   IF(ILOW .GT. LIMIT1(KTYPE)) GO TO 10
C
C**** BACKGROUND HAS BEEN SET, READ CHANGE OR LIST COMMAND
40 WRITE(2,120)
   CALL RLNAF (LUN, 10, INTGR, REAL, IWORD)
   IF(IWORD(1) .EQ. "LIST" .OR. IWORD(1) .EQ. "L") GO TO 50
   IF(IWORD(1) .EQ. "CHANGE" .OR. IWORD(1) .EQ. "C") GO TO 80
   GO TO 15
C
C**** LIST COMMAND

TIME EDIT 73/73 OPT=1

50 IATT1 = INTGR(1)
   IATT2 = INTGR(2)
   IFLG = 0
   IF(IATT1 .EQ. 0 .AND. IATT2 .EQ. 0) IFLG = 1
   IF(IFLG .EQ. 1) IATT1 = 1
   IF(IFLG .EQ. 1) IATT2 = LIMIT2(KTYPE)
   IF(IATT2 .EQ. 0) IATT2 = IATT1
   IF(IATT2 .GT. LIMIT2(KTYPE)) IATT2 = LIMIT2(KTYPE)
IF (IAITI .GT. LIMIT2(KTYPE)) GO TO 40
DO 70 INDEX = ILONW, IHIGH
WRITE(2,140) NAME(KTYPE), INDEX
DO 70 IATT = IATT1, IATT2
IF (KTYPE .EQ. 1) IVALUE = IATP(INDEX, IATT)
IF (KTYPE .EQ. 2) IVALUE = IASP(INDEX, IATT)
IF (KTYPE .EQ. 3) IVALUE = IUNIT(INDEX, IATT)
IF (KTYPE .EQ. 4) IVALUE = ITRUCK(INDEX, IATT)
IF (KTYPE .EQ. 5) IVALUE = ITYPE(INDEX, IATT)
IF (KTYPE .EQ. 6) IVALUE = IMIX(INDEX, IATT)
IF (KTYPE .EQ. 7) IVALUE = INTER(INDEX, IATT)
IF (KTYPE .EQ. 8) IVALUE = IASHE(INDEX, IATT)
IF (KTYPE .EQ. 9) IVALUE = IATPSO(INDEX, IATT)
IF (KTYPE .EQ. 10) IVALUE = IDAY(INDEX, IATT)
IF (KTYPE .EQ. 11) IVALUE = TIME(INDEX, IATT)
IF (KTYPE .EQ. 12) IVALUE = ICSA(INDEX, IATT)
IF (KTYPE .EQ. 13) IVALUE = LPPAK(INDEX, IATT)
IF (KTYPE .EQ. 14) IVALUE = IASPAM(INDEX, IATT)
IF (KTYPE .EQ. 15) IVALUE = LUOUT(INDEX, IATT)
IF (KTYPE .EQ. 16) IVALUE = TCIST(INDEX, IATT)
IF (KTYPE .EQ. 17) IVALUE = TCINC(INDEX, IATT)
IF (KTYPE .EQ. 18) IVALUE = LOAK(INDEX, IATT)

C

IF (KTYPE .NE. 3) GO TO 60
IF (IAIT .NE. 6 .AND. IATT .NE. 7) GO TO 60
WRITE(2,167) IATT, IVALUE
GO TO 70

C

60 WRITE(2,150) IATT, IVALUE
70 CONTINUE
GO TO 40

C

C** CHANGE COMMAND
80 IATT = INTGR(1)
VALUE = INTGR(2) + RFAL(1)
IF (IAIT .GT. LIMIT2(KTYPE)) GO TO 40
UO 90 INDEX = ILow, IHigh
C
INSERT VALUE IN PROPER ARRAY
IF (KTYPE .EQ. 1) IATIP(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 2) IASP(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 3) IUNIT(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 3, .AND, (.NOT. IATT.EQ.6 .OR. IATT.EQ.7))
  Z IUNIT(INDEX, IATT) = IWORD(2)
IF (KTYPE .EQ. 4) ITRUCK(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 5) ITYPE(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 6) IINX(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 7) INT(INDEX, IATT) = VALUE

IF (KTYPE .EQ. 8) IPSTNE(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 9) IATPSD(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 10) I0DAY(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 11) TL4(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 12) ICSA(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 13) LPPAR(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 14) IASP4(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 15) LOGOUT(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 16) FCIST(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 17) TGIATA(INDEX, IATT) = VALUE
IF (KTYPE .EQ. 18) LOOK(INDEX, IATT) = VALUE

C
90 CONTINUE
GO TO 40
C
95 RETURN
100 FORMAT(1X,"VARIABLE NAME = ")
120 FORMAT(1X,"**")
140 FORMAT(1X,1X,A10,I5)
150 FORMAT(1X,"ATTRIBUTE ",I4," = ",I7)
160 FORMAT(1X,"ATTRIBUTE ",I4," = ",A10)
END
SUBROUTINE: EVINIT

PURPOSE: Reads a checkpoint/restart file for mass storage assigned as T2.

COMMON BLOCKS: EVENTS

CALLS: QINIT

IS CALLED BY: ARM Driver

CALLING PARAMETERS: None.

LOCAL ARRAYS: None.

FUNCTIONS:

Reads file containing unused events from previous ARM run.

Asks if operator wants to retain these events.

If operator answers "N" or "NO", calls QINIT to set all pointers to zero events.
SUBROUTINE EVINIT

C***** READS EVENT FILE

C***** 4. JONES  FEB 79
COMMON/EVENTS/JSTAT(6), JEVOS(1024), IEVS(5,1024)
READ(7) JSTAT, JEVOS, IEVS

C***** ERASE OLD EVENTS?
WRITE(2,10)
10 FORMAT ("EVENTS CURRENTLY SCHEDULED? (YES/NO)"")
READ(1,20) IANS
20 FORMAT(A16)
IF(IANS .EQ. "NO" .OR. IANS .EQ. "N") CALL QINIT
RETURN
END
y. SUBROUTINE: EVSTOP

PURPOSE: Writes event files to mass storage (Unit 7) tape2 for checkpoint/restart.

COMMON BLOCKS: EVENT

CALLS: None.

IS CALLED BY: ARM Driver

CALLING PARAMETERS: None.

LOCAL ARRAYS: None.

FUNCTIONS:

   Writes all of event file to mass storage to enable checkpoint/restart.
SUBROUTINE EVSTOP
C**** WRITES EVENT FILE
C**** H. JONES  FEB 79
COMMON/EVENTS/JSTAT(6),JEVDS(1024,4), IEVS(5,1024)
WRITE(8)  JSTAT, JEVDS, IEVS
RETURN
END
z. SUBROUTINE: FINTK

PURPOSE: Finds the truck in the queue (passed parameter) with the proper ammunition code (passed parameter) and the smaller percent load of ammunition.

COMMON BLOCKS: LOG

CALLS: GETQUE
       PUTQUE

IS CALLED BY: ATP
             LDPWDR
             RELOAD

CALLING PARAMETERS: NQUE -- Number of the queue to be searched.
                     NRND -- Round type required.
                     NTRUCK -- Number of truck found in NQUE, equal zero if no truck found in queue.

LOCAL ARRAYS: None.

FUNCTIONS:

Pull the first truck from the queue and save it, and put it back in the queue.

Search through the trucks in the queue, saving the one with the proper ammunition type and the smallest percentage load.

When you pull the check truck from the queue, the search is complete since the queues are first in first out (FIFO).
SUBROUTINE FINLT (NQUE, NRND, NTRUCK)

C**** DETERMINES NUMBER OF TRUCK (NTRUCK) IN QUEUE (NQUE) HAS
C**** THE SMALLEST PERCENTAGE LOAD OF ROUNDS OF TYPE (NRND)

C**** JIM FOX JAN 79

COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITTRUCK(564,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSIME(20,3), IATPSD(5), IUAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCIING, LOOK(17)

C
C**** LOCAL VARIABLES:
C**** NTKSAW -- SAVES TRUCK NUMBER WITH THE SMALLEST LOAD
C**** NPERSV -- SMALLEST PERCENT FOUND
C**** ITRCK -- TRUCK FROM QUEUE
C**** NCHTK -- END OF QUEUE CHECK
C**** MIX -- AMMO MIX INDEX

C
C**** INITIALIZE AMMO PERCENT
NPERSV = 110

C
C**** ASSUME NO TRUCK WITH PROPER AMMO
NTRUCK = 0

C
C**** BRING FIRST TRUCK FROM QUEUE (ITRCK)
CALL GETQUE (ITRCK, NQUE)

C
C**** IF QUEUE IS EMPTY RETURN
IF (ITRCK EQ 0) RETURN

C
C**** THERE ARE SOME TRUCKS IN QUEUE
C**** SEARCH FOR RIGHT TRUCK, STORE NCHTK AND PUT IT BACK IN QUEUE
NCHTK = ITRCK
CALL PUTQUE (ITPCK, NQUE)

C
C**** PULL TRUCK FROM QUEUE
25 CALL GETQUE (ITRCK, NQUE)
C***** DETERMINE AMMO MIX TYPE
   MIX = ITRUCK(ITRCK, 5)
C
C CHECK TO SEE THAT THE TRUCK HAS A VALID MIX
   IF(MIX .GT. 0) GO TO 10
   WRITE(LUOUT,250)ITPCK
   250 FORMAT(1X," FINIK - NONPOSITIVE MIX FOR TRUCK ",I5)
   GO TO 30
C***** IF RIGHT AMMO COMPARE LOAD SIZE; IF NOT GO TO CHECK END QUEUE
   10 IF(1MIX(MIX, NKNL) .GT. 0) GO TO 20
C
C***** WRONG TRUCK, PUT BACK IN QUEUE
   15 CALL PUTQUE(ITPCK, NQUE)
C
C***** IF LAST TRUCK, RETURN
   30 IF(ITPCK .EQ. NCHTK) RETURN
   GO TO 25
C
C***** HAVE FOUND CORRECT AMMO CHECK TO SEE IF SHOULD BE SAVED
   20 IF(ITRUCK(ITRCK, 6) .GT. NPERSV) GO TO 15
      IF(INTRUCK .NE. 0) CALL PUTQUE(INTRUCK, NQUE)

TIME FINIK 73/73 OPT=1

NPERSV = ITRUCK(ITRCK, 6)
INTRUCK = ITPCK
IF(INTRUCK .NE. NCHTK) GO TO 25
RETURN
END
aa. SUBROUTINE: GETQUE

PURPOSE: Removes the first truck from its queue.

COMMON BLOCKS: QUENUM QUEPNT

CALLS: None.

IS CALLED BY: ASP ATP FINTK TRKPUT

CALLING PARAMETERS: ITEM -- Truck number removed from the queue, zero if queue is empty.
                      NUMQUE -- Number of queue to be accessed.

LOCAL ARRAYS: None.

FUNCTIONS:

    Removes lead truck from the queue.
    Updates the queue printer tables.
SUBROUTINE GETQUE (ITEM, NUMQUE)
C**** GETS ITEM FROM QUEUE NUMQUE
C**** TO GET TRUCK FROM QUEUE 4 -- CALL GETQUE (N,4)
C**** 4. JONES DEC 78
   COMMON /QUENUM/ NHEAD(136)
   COMMON /QUEPNT/ IPNT(560)
   ITEM = 0
   LITEM = 0
   IPOINT = NHEAD(NUMQUE)
C
10 IF(IPOINT .EQ. 0) GO TO 20
   LITEM = ITEM
   ITEM = IPOINT
   IPOINT = IPNT(ITEM)
   GO TO 10
20 IF(LITEM .GT. 0) IPNT(LITEM) = 0
   IF(LITEM .EQ. 0) NHEAD(NUMQUE) = 0
C
RETURN
END
bb. SUBROUTINE: INIT

PURPOSE: Reads data base from TI into the LOG and queue COMMONS.

COMMON BLOCKS: LOG
QUENUM
QUEPNT

CALLS: CONTRL
RDJIFF
SCHED
TRKTIM

IS CALLED BY: ARM Driver

CALLING PARAMETERS: None.

LOCAL ARRAYS: None.

FUNCTIONS:

Reads data base into COMMONS LOG, QUENUM, QUEPNT.

Displays message requesting the planned battle time to stop the simulation.

Accepts operator input as to time to stop the simulation and schedule stop event.

Calls TRKTIM, CONTRL, RDJIFF to complete run initialization.
SUBROUTINE INIT
C***** Initializes simulation
C***** H. Jones JAN 79
C
COMMON /LOG/ IATP(4,31), IASP(4,41), IUNIT(75,63),
Z ITRUCK(550,7), ITYPE(6,6), IMIX(43,23), ITPH(9),
Z IRTIME(21,3), IATPS(5), IDAY, TIME,
$ ICXS(20), LPPAR(5), IASP1M(4,23), LUOUT, TOIST, TOLNG,
COMMON /QUENUM/ IHEAD(136)
COMMON /SEQNUM/ ITEMS(560)
DIMENSION IPARM(5)
DATA LOOK /17*0/
C
C***** Read files with all common data
READ(3) IATP, IASP, IUNIT, ITRUCK, ITYPE, IMIX, ITPH, IRTIME,
Z IATPS, IDAY, TIME, ICXS, LPPAR, IASP1M, LUOUT, TOIST,
Z TOLNG, IHEAD, ITEMS
C
IF building answer files bypass steps
WRITE(2,20)
20 FORMAT(" Are your creating an answer file (Y or N)?")
READ(1,21)IANS
21 FORMAT(A10)
IF(IANS.EQ."Y") GO TO 10
C
WRITE(2,22)
22 FORMAT(" Enter time to stop simulation ")
READ(1,*) TSTOP
IPARM(1) = "SCHEDULED"
IPARM(2) = " STOP"
IPARM(3) = " "
IPARM(4) = " "
CALL SCHED (17, IPARM, TSTOP)
C
CALL FEKSY
10 CALL CONTROL (TIME)
TIME = TOIST
IF(IANS.EQ."Y") OR. IANS.EQ."YES") RETURN
C
C***** Read file for demands
CALL ROJIFF
RETURN
END
SUBROUTINE: INTRDK

PURPOSE: Determines if truck is interdicted while en route.

COMMON BLOCKS: LOG

CALLS: None.

IS CALLED BY: ASP
             ASPARV
             ATP
             ATPARV
             CSAARV
             RELOAD
             UNTDEP
             LDPWDR

CALLING PARAMETERS:  NUMTK -- The number of the truck to be considered.
                      TLOST -- = 0 if no interdiction, = time lost if
                               interdiction occurs.

LOCAL ARRAYS: None.

FUNCTIONS:

Determine if truck is in zone 1 (mostly artillery interdiction),
or zone 2 (mostly Air Force interdiction).

Increments the accumulator of the number of trucks that have been
en route in zone 1 or zone 2.

Determine if truck is interdicted.

If yes, assess the time lost to system for truck replacement.
SUBROUTINE INTKUK(NU, TLOST)

C
C**** DETERMINES IF A TRUCK ABOUT TO TRAVEL A ROUTE
C**** WILL BEinterrupted ALONG THAT ROUTE AND ASSESSES
C**** TIME DELAY FOR A REPLACEMENT TRUCK
C**** TWO DEPTH ZONES ARE CONSIDERED
C
C**** J. FOX JAN 79
C
C**** ZONE 1 ALL UNIT TRUCKS SAVE THOSE DIVERTED FROM ATP TO ASP
C**** ALL OTHER REPLACEMENT TRUCKS.
C
C**** NUMTK IS THE NUMBER OF THE TRUCK BEING CONSIDERED.
C**** TLOST = 0 IF TRUCK IS NOT KILLED
C**** REPLACEMENT TIME IF THE TRUCK IS KILLED.
C
C**** SETS LOAD OF REPLACEMENT TRUCK TO 100 PER CENT
COMMON /LOG/ IATP(4,36), IASP(4,41), IUNIT(75,69),
Z ITRUCK(640,7), ITYPE(8,6), IMIX(40,23), INTER(9),
Z ITRM(20,3), IATPS(5), IDAY, TIME,
$ ICSA(20), LPFA(5), IASPAM(4,20), LOUT, TCIST, TCILNG, LOOK(17)
C**** ASSUME TRUCK MADE IT OKAY
TLOST = 0
C**** DETERMINE IF THE TRUCK IS IN ZONE 2, MISSION GT 1 STATUS = 5
C**** GO TO 15
C**** ITRUCK(NUMTK,?) .GT. 1 .OR. ITRUCK(NUMTK,3) .EQ. 5)GO TO 15
C**** TRUCK IS TRAVELING THROUGH ZONE 1
C**** INCREMENT COUNTER OF TRUCKS HN ZONEF 1
INTER(3) = INTER(3) + 1
C**** IF SUFFICIENT NUMBER OF KILLS THIS CI RETURN
IF(INTER(1) .GE. INTER(3))RETURN
C**** IF NOT EQUAL 0 MODULO INTER(7) DO NOT KILL, GO TO RETURN
MODCK = INTER(4) / INTER(7) * INTER(7)
IF(MODCK .NE. INTER(4))RETURN
INTER(1) = INTER(4) + 1
TLOST = INTER(6)
ITRUCK(NUMTK,3) = 7
WRITE (UNIT=30, FMT='') , '15', TIME LOST = '16'
RETURN

C*** HAVE KILLED ZONE 1 TRUCK. INCREMENT 1 TRUCK COUNTER.
C*** IF NOT ZERO MINDE INFR(1), RETURN
C*** SET TRUCK ZONE 32 TRUCK; 15, TIME LOST = '16'
RETURN

C*** IF NOT ZERO MINDE INFR(1), RETURN
C***SET ZONE 32 TRUCK; 15, TIME LOST = '16'
RETURN

C*** IF NOT ZERO MINDE INFR(1), RETURN
C***SET ZONE 32 TRUCK; 15, TIME LOST = '16'
RETURN
dd. SUBROUTINE: IQ

PURPOSE: To provide the queue number associated with the activity being processed.

COMMON BLOCKS: None

CALLS: None

IS CALLED BY: ATP
            ATPAR1
            ATPAR2
            ATPARV
            RELOAD
            UNTARV
            LDPWDR

CALLING PARAMETERS: ITYPE -- Type of queue being searched, varies from 1 to 10, see page 11 for codes.
                     NUM -- Which member of type ITYPE.

LOCAL ARRAYS: None.

FUNCTIONS:

    Check to see if queue type is valid.

    Branches to proper calculation of queue number based on ITYPE.
FUNCTION IQ(ITYPE, NUM)
C***** RETURNS QUEUE NUMBER
C***** JIM FOX JAN '79
C***** LOCAL VARIABLES:
C***** ITYPE IS THE TYPE OF QUEUE TO BE CONSIDERED
C
C***** CHECK FOR VALID QUEUE NUMBER
IF (ITYPE .GT. 1 .AND. ITYPE .LE. 10) GO TO 5
WRITE (2,300)
STOP
C
5 GO TO (11,20,30,40,50,60,70,80,90,100), ITYPE
C
C***** UNIT QUEUE
10 IQ = NUM
GO TO 210
C
C***** ATP QUEUE FOR CSA-ATP TRUCKS
20 IQ = 100 + NUM
GO TO 210
C
C***** ATP QUEUE FOR ASP-ATP TRUCKS
30 IQ = 104 + NUM
GO TO 210
C
C***** ARTILLERY SERVER QUEUE AT THE ATP
40 IQ = 108 + NUM
GO TO 200
C
C***** MANEUVER SERVER QUEUE AT THE ATP
50 IQ = 112 + NUM
GO TO 200
C
C***** NOT USED
60 CONTINUE
GO TO 210
C
C***** ASP QUEUE FOR CSA-ASP TRUCKS
70 IQ = 120 + NUM
GO TO 200
C
C***** ROUTINE SERVER QUEUE AT THE ASP
80 IQ = 124 + NUM
GO TO 200
C
C***** GSPS SERVER QUEUE AT THE ASP
90 IQ = 128 + NUM
GO TO 200
C
C***** NOT USED
100 CONTINUE
C
200 RETURN
300 FORMAT (" BAD QUEUE NUMBER IN FUNCTION IQ")
END
SUBROUTINE: LDPWDR

PURPOSE: Unloads the truck containing powder canisters (ammunition type 3) when 155 HE and ICM (ammunition codes 4 and 5) are removed from the ATP.

COMMON BLOCKS: LOG

CALLS: FINTK
       INTROK
       IQ
       OPERA
       PUTQUE
       SCHED

IS CALLED BY: ATP

CALLING PARAMETERS: NRNDS - Number of powder rounds needed.
                    IPARM (5) - (1) -- 1, indicates artillery.
                    (2) -- ATP number.

LOCAL ARRAYS: IIPARM (5) -- Builds the parameters to schedule trucks back to the ASP or CSA.

FUNCTIONS:

  Sets the ammunition type equal to 3.

  Checks the ASP-ATP queue for powder trucks.

  If insufficient ammunition in the ASP-ATP queue, the CSA-ATP queue is checked.

  Decrements powder ammunition files.

  Schedules empty powder trucks to arrive at ASP (ASPAR 1).
SUBROUTINE LDPWDR(NRNDS,IPARM)
C**** UNLOADS POWDER TRUCK WHEN ARTY AMMO TAKEN FROM ATP
C
C**** J. FOX JAN 79
C
C**** NRNDS IS NUMBER OF POWDER CANISTERS NEEDED
C**** IPARM IS IDENTICAL TO ATP
C**** NOTHING IS RETURNED
C**** SCHEDULES -- ASPAR1 ARRIVAL OF ASP-ATP TRUCK AT ASP
C**** -- CSAARV ARRIVAL OF CSA-ATP TRUCK AT CSA
C
COMMON /LOG/ IATP(4,30),IASP(4,41),IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
C
C**** LOCAL VARIABLE DEFINITION
C**** IIPRAM - PARAMETERS FOR SCHEDULING POWDER TRUCK REFILL
C**** MIX - NUMBER OF AMMO MIX ON TRUCK FOR COMMON IMIX
C**** NR - HARD WIRED AMMO CODE FOR POWDER CHARGES
C**** NRNTK - NUMBER OF POWER CHARGES ON THE TRUCK
C**** NUMTK - POWDER TRUCK ID NUMBER
C**** MR - AMMO TYPE FOR POWDER
C**** NRDSAV - NUMBER OF CANISTERS STILL NEEDED
C**** NQUE - ASP-ATP QUEUE NUMBER
C**** NUMTK - POWDER AMMO TRUCK
C**** NALTQ - CSA -ATP QUEUE NUMBER
C**** NRNTK - NUMBER OF CANISTERS ON TRUCK
C**** DIST - ROAD DISTANCE TRUCK WILL TRAVEL
C**** ICOD - EVENT TYPE TO BE SCHEDULED
C**** ITKTYP - TYPE OF TRUCK
C**** TRTM - ROAD TRAVEL TIME
C**** TFAIL - TIME LOST DUE TO FAILURE
C**** TMIND - TIME LOST DUE TO INTERDICTION
C**** TIME - TIME OF SCHEDULED EVENT
C
DIMENSION IPARM(5),IIPRAM(5)
DO 1 I = 1,5
IIPRAM(I) = 0
1 CONTINUE
C**** SET AMMO TYPE AND NUMBER OF ROUNDS NEEDED
NR = 3
NRDSAV = NRNDS
C**** FIND TRUCK, SAVE QUEUE WE ARE WORKING IN
5 NQUE = IQ(3,IPARM(2))
NNQ = NQUE
CALL FINTK(NNQ,NR,NUMTK)
C**** IF HAVE TRUCK GO TO 10, ELSE CHECK CSA QUEUE
IF(NUMTK .GT. 0)GO TO 10
NALTQ = IQ(2,IPARM(2))
NNQ = NALTQ
CALL FINTK(NNQ,NR,NUMTK)
C**** IF HAVE TRUCK GO TO 10, ELSE WRITE ERROR AND CALL CONTRL
IF(NUMTK .GT. 0)GO TO 10
WRITE(2,15)IPARM(2)
15 FORMAT(" NO POWDER AT ATP ", I2 )
WRITE(LUOUT,30)IPARM(2)
30 FORMAT(" NO POWDER AT ATP ",I2)
RETURN
C**** HAVE TRUCK. IF INSUFFICIENT AMMO, GO TO 20
10 MIX = ITRUCK(NUMTK,5)
 NRNTK = (IMIX(MIX,NR) * ITRUCK(NUMTK,6) +99) / 100
IF(NRNTK .LT. NRDSAV)GO TO 20
C**** SUFFICIENT AMMO, OFFLOAD AND PUT BACK IN QUEUE
ITRUCK(NUMTK,6) = (NRNTK - NRDSAV) * 100 / IMIX(MIX,NR)
CALL PUTQUE (NUMTK, NNQ)
C**** DECREMENT AMMO ON HAND AND DEMAND
IATP(IPARM(2),22) = IATP(IPARM(2),22) - NRNDS
IATP(IPARM(2),23) = IATP(IPARM(2),23) - NRNDS
RETURN
C**** INSUFFICIENT AMMO
20 ITRUCK(NUMTK,6) = 0
C**** DECREMENT ROUNDS NEEDED
NRDSAV = NRDSAV - NRNTK
C**** IF DESTINATION IS ASP GO TO 25
IF(NNQ .EQ. NQUE)GO TO 25
C**** GOING TO CSA
DIST = IATP(IPARM(2),1)
ICOD = 9
IIPRAM(3) = 1
GO TO 27
25 DIST = IATP(IPARM(2),2)
ICOD = 12
IIPRAM(3) = IATP(IPARM(2),6)
27 IIPRAM(1) = IPARM(2)
IIPRAM(2) = NUMTK
ITKTP = ITRUCK(NUMTK,1)
TRTM = 60 * DIST / ITYPE(ITKTP,IDAY+3)
ITRUCK(NUMTK,3) = 4
CALL OPERA(NUMTK,TRTM,TFAIL)
CALL INTRDK(NUMTK,TMIND)
TOTTIM = TIME + TRTM + TFAIL + TMIND
CALL SCHED(ICOD,IIPRAM,TOTTIM)
C**** GO GET ANOTHER TRUCK
GO TO 5
END
SUBROUTINE: LOOKEV

PURPOSE: Enables selective monitoring of event generation and processing.

COMMON BLOCKS: LOG

CALLS: None.

IS CALLED BY: ARM Driver

CALLING PARAMETERS: KIND -- Event code (range of values 1-17)
IPARM (5) -- Event parameters
TLTIME -- Time of scheduled event
IGET -- 1 if event is being processed, = 0 if event is being scheduled.

LOCAL ARRAYS: NAME(17) -- Contains the alphanumeric names of subroutines processing the event types and is used for display purposes.

FUNCTIONS:

Determines from the LOOK array if the type event being handled is to be displayed.

Display event data, if applicable.
SUBROUTINE LOOKEV (KIND, IPARM, TLTIME, IGET)

C**** PROVIDES MONITORING OF EVENTS (DEPENDENT ON LOOK(17))

C**** H. JONES MAR 79

DIMENSION IPARM(5), NAME(17)
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
$ ITKUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
$ IRSTM(20,3), IATPSO(5), IDAY, TIME,
$ ICSTAM(20), LPARR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
DATA NAME /"DEMEAND", "RELOAD", "UNDEP", "ATPAR", "ASPAR",
$ "ATP", "ASP", "UNITARY", "CSAARV", "ATPAR1", "ATPAR2",
$ "ASPAR1", "HELARV", "HASPAR", "REPORT", "CONTRL", "ENDSIM"/

C
IF (LOOK(KIND) .EQ. 0) GO TO 20
N = NAME(KIND)
IF (IGET .EQ. 1) WRITE (LUOUT,10) N, IPARM, TLTIME
10 FORMAT (1X,A10,"", PARM = ",5,16,"", TIME = ",F8.1)
IF (IGET .EQ. 0) WRITE (LUOUT,18) N, IPARM, TLTIME
18 FORMAT (1X,A10,"", PARM = ",5,16,"", SCHED TIME= ",F8.1)
20 RETURN
END
SUBROUTINE: NXTQUE

PURPOSE: Displays the first truck in the queue without changing the queue sequence.

COMMON BLOCKS: QUENUM, QUEPNT

CALLS: None.

IS CALLED BY: CONTRL, TRKPUT

CALLING PARAMETERS: ITEM -- Number of the first truck in the queue.
NUMQUE -- Number of the queue to be examined.

LOCAL ARRAYS: None.

FUNCTIONS:
Determines the number of the first truck in queue NUMQUE.
SUBROUTINE NXTQUE (ITEM, NUMQUE)

C***** SHOWS NEXT ITEM IN QUEUE (LEAVES IT IN)

C***** 4. JONES  FEB 73
COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPNT(360)
ITEM = 0
IPOINT = NHEAD(NUMQUE)

C

10 IF(IPOINT .NE. 0) GO TO 20
ITEM = IPOINT
IPOINT = IPNT(ITEM)
GO TO 10

C

20 RETURN
END

123
SUBROUTINE: OPERA

PURPOSE: Determines if reliability failure exists and assesses the resultant time lost.

COMMON BLOCKS: LOG

CALLS: None.

IS CALLED BY: ASP

ASPARV

ATP

ATPARV

CSAARY

DEMAND

HELARY

UNTDEP

LDPWDR

CALLING PARAMETERS: KTRUCK -- Truck being considered.

TRTME -- Unopposed travel time for the next link of the route.

DELAY -- Time to repair the truck if failure occurs, zero otherwise.

LOCAL ARRAYS: None.

FUNCTIONS:

Determines truck type.

Determines mean time between failure for truck type.

Determines time until the next failure.

If time to next failure is less than the travel time assesses repair time, and resets the time since last failure clock.
SUBROUTINE OPERA (KTRUCK, TTIME, DELAY)
C***** CALCULATES DELAY DUE TO RELIABILITY FAILURE
C***** EACH TRUCK HAS CLOCK OF TIME SINCE LAST FAILURE.
C***** H. JONES JAN 79
C
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRTIME(20,3), IATPSO(5), IOAY, TIME,
& ICSA(23), LPYAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
C
C***** LOCAL VARIABLES
C***** DELAY - TIME LOST DUE TO REMEDIAL MAINTENANCE
C***** KTRUCK - TRUCK NUMBER
C***** KTYPE - TRUCK TYPE
C***** MTBF - MEAN TIME BETWEEN FAILURES FROM ITYPE
C***** REMAIN - TIME LEFT UNTIL NEXT FAILURE BEFORE THIS MOVE
C***** TLLEFT - TIME LEFT UNTIL NEXT FAILURE AFTER THIS MOVE
C***** TTIME - TIME LENGTH OF THIS MOVE
C***** SET TIME LOST TO ZERO
   DELAY = 0.
C***** COMPUTE TRUCKS REMAINING TIME BEFORE FAILURE WITH TRANSIT TIME.
   KTYPE = ITRUCK(KTRUCK, 1)
   MTBF = ITYPE(KTYPE, 5)
   REMAIN = MTBF - ITRUCK(KTRUCK, 7)
   TLLEFT = REMAIN - TTIME
   IF(TLLEFT .GT. 0.) GO TO 10
C***** FAILURE OCCURS THIS TRANSIT
  DELAY = ITYPE(KTYPE, 6)
  ITRUCK(KTRUCK, 7) = - TLEFT

C***** WRITE RECORD FOR LOST TRUCK
  WRITE(LUOUT,5) KTRUCK, TIME
  5 FORMAT(" TRUCK NUMBER",15,", FAILED AT ",F8.0)
  ITRUCK(KTRUCK,3) = 6
  GO TO 20
  10 ITRUCK(KTRUCK,7) = ITRUCK(KTRUCK,7) + TRTIME
  20 RETURN
END
ii. SUBROUTINE: PUTQUE

PURPOSE: Places the truck in the queue by setting queue pointers.

COMMON BLOCKS: QUENUM
                QUEPNT

CALLS: None.

IS CALLED BY: ASPARI
              UNTARV
              FINTK
              LDPWDR
              TRKPUT

CALLING PARAMETERS: ITEM -- Truck to be placed in queue.
                     NUMQUE -- Queue number receiving truck.

LOCAL ARRAYS: None.

FUNCTIONS:

  Places truck in queue by updating pointer tables.
SUBROUTINE PUTQUE (ITEM, NUMQUE)

C***** PUTS ITEM IN QUEUE NUMQUE
C***** H. JONES  DEC 79
COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPNT(560)
IOLOH = NHEAD(NUMQUE)
NHEAD(NUMQUE) = ITEM
IPNT(ITEM) = IOLOH
RETURN
END
SUBROUTINE: QINIT

PURPOSE: Initializes the event queue directory.

COMMON BLOCKS: EVENTS

CALLS: None.

IS CALLED BY: EVINIT

CALLING PARAMETERS: None.

LOCAL ARRAYS: JFORE (1024) -- Equivalenced to the first 1024 words of array JEVDS and contains the pointers to the previous events.

JBACK (1024) -- Equivalenced to the second 1024 words of array JEVDS and contains the pointers to the subsequent events.

JTIME (1024, 2) -- Equivalenced to the last 2048 words of array JEVDS and contains the time parameters of the events.

FUNCTIONS:

Sets number of event space available to max of 1024.

Zeroes first event position, last event position.

Sets pointer arrays so that empty event one points to empty event two etc.

Sets the pointer of empty 1 to 0.

Sets forward pointer of empty event 1024 to zero.
SUBROUTINE QINIT

C
C THIS ROUTINE INITIALIZES THE EVENT QUEUE DIRECTORY /EVENTS/
C
C  Bob Davison
COMMON/EVENTS/JSTAT(6),JEVOS(1024,4),TEVS(5,1024)
DIMENSION JFORE(1024),JBACK(1024),JTIME(1024,2)
EQUIVALENCE (JFORE(1),JEVOS(1,1)),(JBACK(1),JEVOS(1,2)),
Z (JTIME(1,1),JEVOS(1,3)),(JFIRST,JSTAT(1)),(JLAST,JSTAT(2)),
Z (JEMPTY,JSTAT(3)),(NUMEV,JSTAT(4)),(NEMPTY,JSTAT(5)),
4 (MAXEV,JSTAT(6)),(JTIME(1,2),JEVOS(1,4))
NUMEV=0
NEMPTY = 1024
JFIRST=0
JLAST=0
JEMPTY=1
DO 100 I=1,NEMPTY
  JFORE(I)=I+1
  JBACK(I)=I-1
  JTIME(I,1)=0
  JTIME(I,2)=0
100 CONTINUE
  JFORE(NEMPTY)=0
  JBACK(1)=0
RETURN
END
SUBROUTINE: RDIEXO

PURPOSE: Updates IUNIT array for ammunition requirements of this demand pulse.

COMMON BLOCKS: LOG

CALLS: SCHED

IS CALLED BY: DEMAND

CALLING PARAMETERS: NUNIT -- Unit Number

LOCAL ARRAYS: IPARM(5) -- Parameters to schedule the demand event.

FUNCTIONS:

Zero IPARM array.

Determine the number of demand pulses for the unit this run.

SCHED a demand event based on the number of demand pulse being greater than one.

Update number of weapons alive, number of weapons short ammunition and the total number of rounds short for each ammunition type.
SUBROUTINE RDIEXO(NUNIT)
C UPDATES IUNIT EACH PULSE OF A MULTI-DEMAND AND SCHED DEMAND
C
JIM FOX - FEB 1979
COMMEN /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z TRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
3 ICSA(20), LPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
C LOCAL VARIABLES
C NUNIT - UNIT NUMBER
C NCELLS - NUMBER OF DEMAND PULSES IN DEMAND UNIT RECORD
C IFLIV - NEGATIVE OF NUMBER OF TUBES KILLED
C TOTTIM - COMPUTED TIME FOR SCHEDULING AN EVENT
C
DIMENSION IPARM(5)
DO 5 I = 1,5
IPARM(I) = 0
5 CONTINUE
C SET IPARM TO CALL TO SCHEDULE DEMAND
IPARM(1) = NUNIT
C FIND NUMBER OF DEMAND PULSE CELLS
XCELLS = IUNIT(NUNIT,69)
NCELLS = XCELLS
IF(NCELLS.LE.1) GO TO 100
C COMPUTE THE TIME OF THE NEXT DEMAND EVENT AND SCHEDULE IT.
TOTTIM = TIME + TCILNG / XCELLS
IF(TOTTIM .GT. TCIST + TCILNG)GO TO 100
CALL SCHED(1,IPARM,TOTTIM)
100 CONTINUE
IF(NCELLS.LE.1) NCELLS=1
IF(XCELLS.LE.1) XCELLS=1.
C UPDATE IUNIT WITH A PART OF THE DEMAND DATA
DO 20 I = 1,5
IND = I * 12 - 4
IF(IUNIT(NUNIT,IND+10) .LE. 0)GO TO 20
C COMPUTE WHICH PULSE THAT THIS UPDATE REPRESENTS
NUMPL = (TIME - TCIST) / (TCILNG / XCELLS) + .5
IF(NCELLS.LE.1) NUMPL=1
C COMPUTE NEGATIVE SURVIVOR FACTOR
C LOWER NUMBER OF SURVIVORS FOR THIS PULSE
C COMPUTE THE NUMBER OF DEAD TO BE ASSESSED THIS PULSE - NMDEAD
NMDEAD = (IUNIT(NUNIT,IND+9)+NUMPL-1) / NCELLS
C COMPUTE NUMBER OF RNDS LOST WITH DEAD WPN
NDEDRD=NMDEAD*IUNIT(NUNIT,IND+4)/IUNIT(NUNIT,IND+1)
C IF NEG. ROUNDS ON HAND - NONE LOST.
IF(IUNIT(NUNIT,IND+4) .LE. 0) NDEDRD = 0
IUNIT(NUNIT,IND+1) = IUNIT(NUNIT,IND+1) - NMDEAD
IF(IUNIT(NUNIT,IND+1) .LT. 0) IUNIT(NUNIT,IND+1)=0
C**** ASSUME SINGLE PULSE UNIT
IUNIT(NUNIT,IND+2)=IUNIT(NUNIT,IND+10)
132
C**** CHECK FOR ARTY UNITS
   IF(IUNIT(NUNIT,1).GE.4.AND.IUNIT(NUNIT,1).LE.6) IUNIT(NUNIT,IND+2)
   Z = IUNIT(NUNIT,IND+1)
C   IF FARP, NUMBER OF WEAPONS SHORT AMMO = NUM IN CELL
   IF(IUNIT(NUNIT,1).EQ.8)IUNIT(NUNIT,IND+2) = IUNIT(NUNIT,IND+2)
   Z = (IUNIT(NUNIT,IND+10) + NUMPL - 1) / NCELLS
C   UPDATE A PORTION OF ROUNDS SHORT
C   COMPUTE THE NUMBER OF ROUNDS SHORT TO BE ASSESSED THIS PULSE-NMRD
   NMRD = (IUNIT(NUNIT,IND+11) + NUMPL - 1) / NCELLS
   IUNIT(NUNIT,IND+3) = IUNIT(NUNIT,IND+3) + NMRD - (IUNIT(NUNIT,IND+7)
   Z = NMREAD - NDEDRD)
C   COMPUTE AMMO ON HAND
   IUNIT(NUNIT,IND+4) = IUNIT(NUNIT,IND+4) - (NDEDRD + NMRD)
20 CONTINUE
   RETURN
   END
11. SUBROUTINE: RDJIFF

PURPOSE: Reads output file created by the attrition model of ammunition usage and updates IUNIT for RDIEXO.

COMMON BLOCKS: LOG

CALLS: EOF
       SCHED
       DECODE

IS CALLED BY: INIT

CALLING PARAMETERS: None.

LOCAL ARRAYS: IRDJF (64) -- Array resulting from reading attrition model record and converting it for ARM usage.
                IPARM (5) -- Parameter list to schedule the demand events.
                RDJF (64) -- Array for temporary storage of attrition model record.

FUNCTIONS:

Reads attrition model-produced file record of 64 real words.
Copies words 2 through 64 into an integer array.
Decodes alphanumeric first word (unit name from attrition model) into the integer array.
Finds matching unit number and replaces unit name.
Updates IUNIT with ammunition usage data for use by subroutine RDIEXO.
Schedules the first demand pulse.
Determines the number of demand pulses and places in array IUNIT.
Branches to first function until records are processed.
SUBROUTINE RDJIFF
C
READS OUTPUT FILE CREATED BY JIFFY.
TRANSLATES THE JIFFY IDS TO ARM NUMBERS
SCHEDULES A DEMAND EVENT FOR EACH UNIT FIRING AMMO.
UPDATES IUNIT FOR SINGLE PULSE DEMAND UNITS.
C
JIM FOX - FEB 1979
C
LOCAL VARIABLE DEFINITION
UNTMAP - JIFFY UNIT NAMES ASSOCIATED WITH ARM UNIT NUMBERS.
IRDJF - JIFFY CREATED INTERFACE RECORD 64 WORDS LONG PER RECORD
  1 - JIFFY UNIT ID
  2 - NUMBER OF AH IN CELL(AH ONLY
  3 - ARM AMMO CODE
  4 - NUMBER OF WEAPONS ALIVE
  5 - NUMBER OF WEAPONS SHORT AMMO
  6 - TOTAL NUMBER OF ROUNDS SHORT
  7-11 ECT REPEAT OF 2 - 6
LUIN1 - JIFFY PRODUCED INPUT FILE
C
IAUN - LOOP INDEX
IND1 - COMPUTED INDEX TO ACCESS IUNIT AMMO TYPE
IAMM - AMMO TYPE FROM IUNIT
Ijf  - LOOP INDEX
IJFAM - COMPUTED INDEX TO ACCESS IRDJF FOR AMMO TYPE
IJAM - IRDJF AMMO TYPE
I - LOOP INDEX
III - LOOP INDEX
IIJ - LOOP INDEX
IJF - LOOP INDEX
IU - LOOP INDEX
IN - INDEX TO SEARCH FOR UNIT NAME
IN1 - MATCHED ARM UNIT NUMBER
IAUN - LOOP INDEX
IND1 - INDEX COMputed FROM IAUN TO ACCESS IUNIT FOR AMMO TYPE
IA LOOP INDEX
IAM - INDEX COMPUTED FROM IA TO SEARCH IRDJF
IAMM - AMMO TYPE FROM IRDJF
IJFAM - INDEX COMPUTED FROM IJF TO ACCESS IRDJF
IJAM - AMMO TYPE FROM IRDJF
IUA - COMPUTED INDEX FROM IU TO ACCESS IUNIT
IUAM - AMMO TYPE FROM IUNIT
NHELCL - ACCUMULATOR FOR NUMBER OF HELICOPTERS IN CELLS
NMDEAD - NUMBER OF AH LOST TO FARp
NNI - INDEX TO ACCESS IRDJF,2,7,12 ECT
NRNSH - ACCUMULATOR FOR NUMBER OF ROUNDS SHORT
NWPNAL - ACCUMULATOR FOR NUMBER OF AH RETURNING ALIVE
NWPNSH - ACCUMULATOR FOR NUMBER OF AH SHORT ROUNDS
NCELLS - NUMBER OF CELLS IN FARp RECORD
XCELL - REAL VARIABLE EQUAL TO NCELLS
DELTIM - TIME INTERVAL BETWEEN FARp CELL PROCESSING

135
C III - LOOP INDEX
C XII - REAL EQUAL TO III
C IIL - COMPUTED INDEX TO FILL IEXOUT
C IIJ - COMPUTED INDEX TO FIND IRDJF TO BUILD IEXOUT RECORD
C TOTTIM - TIME TO SCHEDULE EVENT

COMMON /LOG/IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
DIMENSION IRDJF(64), IPARM(5), RDJF(64)

DO 10 I = 1, 5
  IPARM(I) = 0
10 CONTINUE

C**** ZERO LAST EVENT'S DEMAND
C**** LOOP FOR UNITS
  DO 2 I1 = 1, 75
C**** LOOP FOR 5 AMMO'S
  DO 3 I2 = 1, 5
C**** LOOP FOR THREE ELEMENTS
  DO 4 I3 = 1, 3
C**** COMPUTE THE FUN INDEX
  I4 = 4 + I2 * 12 + I3
  IUNIT(I1,14) = 0
4 CONTINUE
3 CONTINUE
2 CONTINUE

C
  LUINI = 9
  REWIND LUINI
C READ RECORD FROM JIFFY PRODUCED FILE.
  20 READ(LUINI)RDJF
C IF END OF FILE LUINI GO TO WRAPUP (200)
  IF(EOF(LUINI))200,15
C HAVE A RECORD, CONVERT UNIT TO ARM NUMBER
C**** DECODE UNIT NAME AND COPY OTHER REALS TO INTEGER
  15 DO 5 I = 2, 64
     IRDJF(I) = RDJF(I)
 5 CONTINUE
C DECODE(10,1, RDJF(1)) IRDJF(1)
  1 FORMAT(A10)
  DO 16 IN = 1, 75
     IF(IUNIT(IN,7) .EQ. IRDJF(1)) GO TO 30
16 CONTINUE
C NO MATCH
  WRITE(LUOUT,60) IRDJF(1)
  60 FORMAT(" NO MATCH FOR JIFFY UNIT ", A10)
  GO TO 20
C HAVE A MATCH REPLACE UNIT NAME WITH IN
  30 IRDJF(1) = IN
     IN1 = IN
     IPARM(1) = IN1

136
C IF NOT SINGLE PULSE UNIT GO TO 100
IF(IUNIT(IN1,69) .GT. 0) GO TO 100
C HAVE A SINGLE PULSE UNIT, UPDATE IUNIT AND CALL SCHEDULE
C FIND PROPER AMMO IN ARM UNIT (IUNIT)
40 DO 80 IAUN = 1,5
   IND1 = 12 + IAUN - 4
   IAMMO = IUNIT(IN1,IND1)
   IF(IAMMO .EQ. 0) GO TO 80
   DO 79 IJF = 1,5
      IJFAM = IJF * 5 - 2
      IJAM = IRDJF(IJFAM)
   C****
   C****
   C SCENARIO DEPENDENT CODE TO READ IN SECOND AMMO CODE 2
   IF(IJAM .EQ. 25 .AND. IAUN .EQ. 3) GO TO 45
   C****
   C****
   IF(IJAM .EQ. 12) IRDJF(IJFAM + 3) = IRDJF(IJFAM + 3) * 90
   IF(IJAM .NE. IAMMO .OR. IJAM .EQ. 0) GO TO 79
   C HAVE EQUAL AMMO TYPES, UPDATE IUNIT WITH NEW DATA.
   C**** UPDATE FOR DEMAND DATA IN UNIT STATUS REPORT
   45 IUNIT(IN1,IND1+9) = IUNIT(IN1,IND1+1) - IRDJF(IJFAM+1)
   IF(IUNIT(IN1,IND1+9) .LT. 0) IUNIT(IN1,IND1+9) = 0
   IUNIT(IN1,IND1+10) = IRDJF(IJFAM+2)
   IUNIT(IN1,IND1+11) = IRDJF(IJFAM+3)
   IRDJF(IJFAM) = 0
   C END OF IUNIT UPDATE FOR THIS AMMO TYPE
   GO TO 80
79 CONTINUE
C NO UNIT AMMO MATCH
WRITE(LUOUT,78) IN1,IJAM
78 FORMAT(" NO AMMO MATCH IN IUNIT. UNIT ",I5," AMMO",I5)
80 CONTINUE
C SCHEDULE DEMAND
TOTTIM = TCIST + .5 * TCILN
CALL SCHED(1,IPARM,TOTTIM)
GO TO 20
C HAVE A MULTIPLUSE UNIT. IF ARTY GO TO 120
100 IF(IUNIT(IN1,1) .GT. 3 .AND. IUNIT(IN1,1) .LT. 7) GO TO 120
C HAVE A FARP COUNT THE CELLS
NCELLS = 0
NHCLCL = 0
NWPNAL = 0
NWPNSH = 0
NRNSH = 0
C DO 102 I = 1,10
   NNI = 5 * I - 3
   IF(IRDJF(NNI) .LE. 0) GO TO 102
   NCELLS = NCELLS + 1
   NHELCL = NHELCL + IRDJF(NNI)
NWPNAL = NWPNAL + IRDJF(NNI+2)
NWPNSH = NWPNSH + IRDJF(NNI+3)
NRNSH = NRNSH + IRDJF(NNI+4)
102 CONTINUE
C IF NO CELLS, GO TO THE NEXT UNIT RECORD
IF(NCELLS .LE. 0)GO TO 20
XCELL = NCELLS
C COMPUTE TIME BETWEEN DEMAND PULSES
DELTIM = TCILNG / XCELL
C UPDATE NUMBER OF DEMANDS FOR THIS FARP
IUNIT(IN1,69) = NCELLS
C TAKE CARE OF THE SINGLE CELL FARP
IF(NCELLS .GT. 1)GO TO 103
DELTIM = TCILNG / 2.
C FIND PROPER AMMO TO UPDATE
103 DO 104 III = 1,5
IIJ = III * 12 - 4
C IF WRONG AMMO GO TO 104
IF(IUNIT(IN1,IIJ) .NE. IRDJF(3))GO TO 104
C HAVE PROPER AMMO UPDATE HOLDING AREA IN IUNIT
IUNIT(IN1,IIJ+9) = NHELCL - NWPNAL
IUNIT(IN1,IIJ+10) = NWPNSH
IUNIT(IN1,IIJ+11) = NRNSH
C SCHEDULE FIRST DEMAND EVENT
TOTTIM = TIME + DELTIM
CALL SCHED(1,IPARM,TOTTIM)
GO TO 20
104 CONTINUE
GO TO 20
C HAVE AN ARTY UNIT. BUILD EXO AND SCHED DEMAND
102 TOTTIM = TCIST + 60.
CALL SCHED(1,IPARM,TOTTIM)
C FIND AMMO TYPES TO UPDATE IUNIT HOLDING FOR ARTY
IUNIT(IN1,69) = TCILNG / 60. + .5
DO 300 IA = 1,5
C SELECT AMMO RECORD FROM IRDJF
IAM = 5 * IA - 2
Iamm = IRDJF(IAM)
C FIND CORRESPONDING UNIT AMMO
DO 290 IU = 1,5
IUA = 12 * IU - 4
Iuam = IUNIT(IN1,IUA)
C IF NOT THE SAME AMMO GO TO 290
IF(IUAM .NE. Iamm)GO TO 290
C HAVE AMMO MATCH. SET UP FILE IUNIT.
IUNIT(IN1,IUA+9) = IUNIT(IN1,IUA+1) - IRDJF(IAM+1)
IF(IUNIT(IN1,IUA+9) .LT. 0)IUNIT(IN1,IUA+9) = 0
IUNIT(IN1,IUA+10) = IRDJF(IAM+2)
IUNIT(IN1,IUA+11) = IRDJF(IAM+3)
GO TO 300
290 CONTINUE
WRITE (LUOUT,291)IN1,Iamm
138
291 FORMAT(" NO IUNIT AMM MATCH - RDJIFF, UNIT ",I5," AMMO ",I5)
   GO TO 20
300 CONTINUE
   GO TO 20
200 WRITE(LUOUT,400)
   WRITE(2,400)
400 FORMAT(" HAVE FINISHED RDJIFF ")
   RETURN
END
SUBROUTINE: READF

PURPOSE: Accepts up to 10 integers, real and/or alpha fields from the operator.

COMMON BLOCKS: None

CALLS: EOF
FLOAT

IS CALLED BY: EDIT
TRKPUT

CALLING PARAMETERS: LU -- Logistical unit number of input.
NUM -- Maximum number of each type of field to be accepted in a single line.
INTGR -- Array for storing up to NUM integer fields.
REAL -- Array for storing up to NUM real fields.
IWORD -- Array for storing up to NUM alphanumeric fields.

LOCAL ARRAYS: ICHR(82) -- Local array to accept field of 80 characters input by the operator.
IALDIG(10) -- Local array to store the integers 1 through 0.

FUNCTIONS:

Reads 80 characters of input from logical unit LU.
Initialize integer, real and alphanumeric storage arrays.
Determines if each field is real, integer or alphanumeric.
Builds fields, character by character using blanks and/or commas as separators.
SUBROUTINE READF (LU, NUM, INTGR, REAL, IWORD)

C**** RETURNS UP TO NUM INTEGERS, REALS, AND STRINGS.
C**** BLANKS AND COMMAS ARE DELIMITERS
C****
H. JONES 1979

DIMENSION INTGR(1), REAL(1), IWORD(1)
DIMENSION ICHR(82), IALDIG(10)
DATA IBLANK /"","", IPERD /"","", ICOMM /"","", IMINUS "/-/
DATA IQOUT/1H"/
DATA IALDIG /"1"", "2"", "3"", "4"", "5"", "6"", "7"", "8"", "9"", "0"/
ICH(81) = IBLANK
ICH(82) = IQOUT

C**** READ RECORD, ZERO OUT OLD INTGR, REAL, IWORD
READ(LU,100) (ICH(I), I=1,80)
IF (EOF(LU) .NE. 0) GO TO 60
DO 4 I=1,NUM
INTGR(I)=0
REAL(I)=0.
4 IWORD(I) = IBLANK
KWORD=0
KINTGR=0
KREAL=0
N=0

C**** CHECK NEXT CHARACTER IN RECORD
C**** SKIPPING BLANKS ************
10 MINUS = 1
11 N=N+1
IF(N.EQ.81) GO TO 60
IF(ICH(N).EQ.IBLANK) GO TO 11
C**** DETERMINE IF CHAR IS NUMBER OF ALPHA
IF(ICH(N).EQ. IQOUT) GO TO 41
IF(ICH(N).EQ. IMINUS) GO TO 12
MINUS = -1
GO TO 11
12 ISTART = N
NUMB=0
IF(ICH(N).EQ.IPERD) GO TO 28
DO 15 I=1,10
IF(ICH(N).EQ.IALDIG(I)) GO TO 20
15 CONTINUE
GO TO 40
C**** BUILDING INTEGER OR INTEGER PART OF REAL
20 N=N+1
IF(ICH(N).NE. IBLANK .AND. ICHR(N).NE. IPERD
Z .AND. ICHR(N).NE. ICOMM ) GO TO 20
C**** CALCULATE VALUE OF INTEGER
IEND = N-1
NUMB=0
DO 25 I=ISTART, IEND
DO 24 J=1,9
IF(ICH(I) .EQ. IALDIG(J)) GO TO 25
24 CONTINUE
J=0
25 NUMB = NUMB + J * 10 **(IEND-I)
IF(ICH(N) .EQ. IPERD) GO TO 28
C
C**** NUMBER WAS INTEGER, STORE IT, CHECK FOR BLANKS
KINTGR = KINTGR +1
INTGR(KINTGR) = NUMB * MINUS
GO TO 10
C
C**** NUMBER WAS INTEGER PART OF REAL, NOW BUILD DECIMAL.
28 RNUMB = FLOAT(NUMB)
ISTART = N+1
IF(ICH(ISTART) .EQ. IBLANK) GO TO 39
30 N=N+1
IF(ICH(N).NE.IBLANK .AND. ICHR(N).NE.ICOMMA ) GO TO 30
C
C**** CALCULATE VALUE OF DECIMAL
IEND = N-1
IDECPL = 1
DO 38 I=ISTART,IEND
DO 34 J=1,9
IF(ICH(I) .EQ. IALDIG(J)) GO TO 35
CONTINUE
J=0
35 NUMB = NUMB + J * 10**(IEND-I)
38 IDECPL = IDECPL * 10
C
C**** ADD INTEGER AND DECIMAL
DECML=FLOAT(NUMB)/FLOAT(IDECPL)
RNUMB = RNUMB + DECML
39 KREAL = KREAL + 1
REAL(KREAL) = RNUMB * MINUS
GO TO 10
C
C**** BUILDING STRING ALPHANUMERIC
40 N=N+1
IF(ICH(N).NE.IBLANK .AND. ICHR(N).NE.ICOMMA ) GO TO 40
GO TO 44
41 ISTART = N+1
42 N=N+1
IF(ICH(N) .NE. IQUOT) GO TO 42
44 IEND = N-1
KWORD = KWORD + 1
LENSTR = IEND - ISTART + 1
IF(LENSTR .GT. 10) LENSTR = 10
ENCODER(LENSTR, 90, IWORD(KWORD)) (ICH(KKK), KKK=ISTART, IEND)
GO TO 10
C
GO TO 10
C
GO TO 10
C
GO TO 10
C
GO TO 10
90 FORMAT(10A1)
100 FORMAT(80A1)
END
SUBROUTINE: SCHED

PURPOSE: Schedules events.

COMMON BLOCKS: None

CALLS: CONTRL
       LOOKEV
       PUTEVT

IS CALLED BY: ASP
               ASPARV
               ASPARI
               ATP
               ATPARV
               CSAARV
               DEMAND
               HELARV
               RELOAD
               INIT
               LDPWDR
               RDIEXO
               RDJIFF

CALLING PARAMETERS: ITYPE -- ARM event code.
                      IPARM(5) -- Parameters for the event.
                      TIME -- Time that the event is scheduled to occur.

LOCAL ARRAYS: None.

FUNCTIONS:

Sets IPARM(5) to event type.
Calls LOOKEV to determine if event should be displayed.
Calls PUTEVT to place the event in the EVENT array.
If PUTEVT was unsuccessful displays message and calls CONTRL.
SUBROUTINE SCHED (ITYPE, IPARM, TIME)
C==== INTERFACE ROUTINE TO SCHEDULE EVENT
C==== H. JONES  DEC 79
DIMENSION IPARM(5)
IPARM(5)=ITYPE
CALL LOOKEV (ITYPE+1, IPARM, TIME+1, I)
ITH = TIME
ITS = (TIME - ITH) * 3600
CALL PUTEVT (IPARM, ITH, ITS, ICHECK)
IF (ICHECK .EQ. 0) GO TO 20
WRITE (2, 30) ICHECK
CALL CONTPL(TIME)
20 RETURN
30 FORMAT ("** TOO MANY EVENTS -- ", I6)
END
SUBROUTINE: SETQUE

PURPOSE: Initializes truck queues to zero by replacing all pointers with zeros.

COMMON BLOCKS: QUENUM QUEPNT

CALLS: None

IS CALLED BY: TRKPUT

CALLING PARAMETERS: ITEMS -- The number of trucks that will be placed in queues.
NUMQUE -- Total number of queues receiving trucks.

LOCAL ARRAYS: None.

FUNCTIONS:

Zeroes the queue directories.
SUBROUTINE SETQUE (ITEMS, NUMQUE)

C***** SETS UP NUMQUE EMPTY QUEUES FOR ITEMS.
C***** H. JONES DEC 79

COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPMT(560)

DO 10 I=1,NUMQUE

10 NHEAD(I) = 0

DO 20 I=1,ITEMS

20 IPMT(I) = 0

RETURN

END
SUBROUTINE: TRKPUT

PURPOSE: Enables interactive truck assignment, unassignment, and/or reassignment to queues.

COMMON BLOCKS: None

CALLS: GETQUE
       NXTQUE
       PUTQUE
       READF
       SETQUE

IS CALLED BY: CONTRL

CALLING PARAMETERS: None.

LOCAL ARRAYS: INTGR (10) -- Holds up to 10 integer fields from the operator input.
                REAL (10) -- Holds up to 10 real fields from the operator input.
                IWORD (10) -- Holds up to 10 alphanumeric fields from the operator input.

FUNCTIONS:

Displays names.
Accepts operator input by calling READF and does one of the following.
Puts truck into queue.
Pulls truck from queue.
Lists truck from a queue.
Initializes pointers removing all trucks from all queues.
SUBROUTINE TRKPUT

C**** ALLOWS INTERACTIVE TRUCK QUEUE RE-ASSIGNMENT
C**** H. JONES FEB 79

DIMENSION INTGR(10), REAL(10), IWORD(10)

WRITE(2,10)
10 FORMAT(1X,"COMMAND EXAMPLES :").
    Z 1X,"GET 3 FROM 35 ",/.
    Z 1X,"PUT 3, 10 IN 105 ",/.
    Z 1X,"LIST 105 ",/.
    Z 1X,"TAKE ALL OUT ",/.
    Z 1X,"END ",/)

WRITE(2,20)
20 FORMAT("...")

CALL READF (1, 10, INTGR, REAL, IWORD)
IF(IWORD(1).EQ. "END" .OR. IWORD(1).EQ. "E") GO TO 50
IF(IWORD(1).EQ. "PUT" .OR. IWORD(1).EQ. "P") GO TO 30
IF(IWORD(1).EQ. "LIST" .OR. IWORD(1).EQ. "L") GO TO 40
IF(IWORD(1).EQ. "GET" .OR. IWORD(1).EQ. "G") GO TO 25
IF(IWORD(1).EQ. "TAKE" .OR. IWORD(1).EQ. "T") GO TO 60
GO TO 15

C**** GET TRUCK FROM QUEUE WITHOUT RE-ORDERING QUEUE

25 II = INTGR(1)
    I2 = INTGR(2)
    IF(INTGR(3).NE. 0) GO TO 15
    IFLAG = 0
    CALL NXTQUE (IFIRST, I2)

26 CALL NXTQUE(NTRK, I2)
    IF(NTRK.EQ.0) GO TO 15
    IF(NTRK.EQ. IFIRST .AND. IFLAG .NE. 0) GO TO 15
    CALL GETQUE(NTRK, I2)
    IF(I1 .EQ. IFIRST) GO TO 15
    IF(I1 .NE. NTRK) CALL PUTQUE (NTRK, I2)
    IFLAG = 1
    GO TO 26

C

C**** PUT TRUCK IN QUEUE

30 II = INTGR(1)
    I2 = INTGR(2)
    I3 = INTGR(3)
    IF(INTGR(3).EQ. 0) I3 = INTGR(2)
    IF(INTGR(3).EQ. 0) I2 = INTGR(1)
    DO 35 I=I1, I2
35 CALL PUTQUE (I, I3)
    GO TO 15

C

C**** LIST TRUCKS IN QUEUE

40 CALL NXTQUE (IFIRST, INTGR(1))
    IF(IFIRST .EQ. 0) GO TO 15
42 CALL GETQUE(NTRK, INTGR(1))
    CALL PUTQUE(NTRK, INTGR(1))
WRITE(2,45) NTRK
45 FORMAT(1X,15)
   CALL NXTQUE (INEXT, INTGR(1))
   IF(INEXT .NE. IFIRST) GO TO 42
   GO TO 15
C
C**** TAKE ALL TRUCKS OUT OF QUEUES
   60 CALL SETQUE (560, 136)
   GO TO 15
C
50 RETURN
END
SUBROUTINE: TRKTIM

PURPOSE: Initializes the ITRUCK arrays with time since last failure.

COMMON BLOCKS: LOG

CALLS: RANF

IS CALLED BY: INIT

CALLING PARAMETERS: None.

LOCAL ARRAYS: None.

FUNCTIONS:

Asks operator if truck times since last failure should be initialized; if no, returns. Else loops through the trucks.

Determines the truck type and mean time between failure (MTBF) for the truck type.

Draws a number from a uniform distribution (0-1) and multiply it by MTBF to determine the time since the last failure.

Stores calculated time since the last failure in ITRUCK.
SUBROUTINE TRKTM

C

COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,21), INTER(9),
Z IRMME(20,3), IATPSN(5), IDAY, IIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCIHNG, LOOK(17)

C

WRITE(2,90)
90 FORMAT(" INITIALIZF TRUCK'S TIME SINCE LAST FAILURE? (YES/NO)")
READ(1,20) IANS
20 FORMAT(A10)
IF(IANS.EQ."NO", OR, IANS.EQ."N") GO TO 35

C**** LOOP THROUGH TRUCKS
ITOP = LPPAR(4)
DO 80 I = 1,ITOP
ITYP = ITRUCK(I,1)
C**** IF NOT ACTIVE Bypass
IF (ITYP .EQ. 1) GO TO 80
C**** FIND MTBF
XMTBF = ITYPE(ITYP,5)
C**** FINJ UNIFORM RANDOM NUMBER
UNRN =RANF(1)
C**** STORE TIME SINCE MAINT. FOR THIS TRUCK
ITRUCK(I,7) = XMTBF * UNRN
80 CONTINUE

C

35 RETURN
END
SUBROUTINE: TRUCK

PURPOSE: Writes the status of unit trucks

COMMON BLOCKS: LOG

CALLS: None

IS CALLED BY: Report

CALLING PARAMETERS: None.

LOCAL ARRAYS: None.

FUNCTIONS:

Finds trucks assigned to a given unit and prints the current status of each truck.
SUBROUTINE TRUCK (L)
C**** WRITES STATUS OF UNIT TRUCKS
C**** D REMEN JUN 79
    COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
       ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTR(9),
       INSN(20,1), IATPSD(5), IDAY, TIME,
       INCSA(20), LPPAR(6), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
C**** WRITE HEADER
    WRITE (LUOUT,20) IUNIT(L,7)
20 FORMAT(1X,///,AX,\" TRUCK STATUS REPORT FOR UNIT ",A10,///,
       \" TRK NM STATUS MIX PCLOAD NXFAIL\",/)
C**** LOOP THROUGH THE TRUCKS
    DO 5 J = 1,560
C**** IF TRUCK NOT OF THIS UNIT, BYPASS
       IF (ITRUCK(J,4) .NE. L) GO TO 5
       IF (ITPCK(J,2) .NE. 1) GO TO 5
C**** HAVE TRUCK OF THIS UNIT PRINT INFO
       NXFAIL = ITYPE(ITRUCK(J,1),5) - ITRUCK(J,7)
       WRITE (LUOUT,30) J, ITRUCK(J,2), ITRUCK(J,3), ITRUCK(J,5),
       ITRUCK(J,6), NXFAIL
30 FORMAT (2X,5I7)
    5 CONTINUE
    RETURN
END
6. DESCRIPTION OF ASSOCIATED PROGRAMS. Although ARM is self sufficient to accomplish all tasks associated with ammunition resupply simulation there are several other programs that have been developed to assist the ARM operator in accomplishing the tasks associated with data base development. This section will present a description of each of these programs and their functions. See appendix A for computer listings of these programs.

a. PROGRAM: HJEDIT

PURPOSE: To call HUEDIT which allows editing of data base separately from ARM.

COMMON BLOCKS: LOG, QUENUM, QUEPNT

CALLS: HUEDIT

IS CALLED BY: Operator

CALLING PARAMETERS: None

LOCAL ARRAYS: None

FUNCTIONS: Connects the data base and HUEDIT, generates an output file of revised data base, returns keyboard and binary file.

b. PROGRAM: HUEDIT

PURPOSE: To permit building of initial data base or modification of existing data base without calling HJARMANOTHER.

COMMON BLOCKS: LOG, QUENUM, QUEPNT

CALLS: EDIT, UPDATE

IS CALLED BY: HJEDIT

CALLING PARAMETERS: None

LOCAL ARRAYS: None

FUNCTIONS: Calls EDIT if editing of data is desired. Calls UPDATE if updating of arrays is desired.

c. SUBROUTINE: EDIT

PURPOSE: To edit data base, functions the same as the edit subroutine within ARM (see w. Subroutine: Edit)
PROGRAM HJEDIT
100=CONNECT, KEYBRD, DISPLAY.
110=ATTACH, AA, HCEDIT, MR=1, ID=CARTSV.
120=AA(KEYBRD, DISPLAY, TI, TAPE1, INPUT, OUTPUT, TAPE6=OUTPUT).
130=RETURN, KEYBRD, DISPLAY, AA.
140=REWIND, TAPE1, OUTPUT.
150=EOF
160=EOF

PROGRAM HUEDIT (TAPE1, TAPE2, TAPE3, TAPE4, INPUT, OUTPUT)
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(9),
Z IRSTME(20,3), IATPSD(5), IDAY, TIME,
$ ICSA(20), LPPAR(5), IASPAM(4,20), LUOUT, TCIST, TCILNG, LOOK(17)
COMMON /QUENUM/ IHEAD(136)
COMMON /QUEPNT/ ITEMS(560)
DATA LOOK /17*0/
READ(3) IATP, IASP, IUNIT, ITRUCK, ITYPE, IMIX,
Z INTER, IRSTME, IATPSD, IDAY, TIME, ICSA,
Z LPPAR, IASPAM, LUOUT, TCIST, TCILNG, IHEAD, ITEMS
10 WRITE(2,7)
7 FORMAT(" EDIT DATA FILE ? (YES/NO) ")
READ(1,17) IANS
17 FORMAT(A10)
IF(IANS.EQ."NO".OR.IANS.EQ."N") GO TO 20
CALL EDIT
GO TO 10
20 PRINT*, "UPDATE ARRAYS?"
READ21 ,IYN
21 FORMAT(A1)
IF(IYN.EQ."N")GOTO30
CALL UPDATE
GOTO10
30 WRITE(4) IATP, IASP, IUNIT, ITRUCK, ITYPE, IMIX,
Z INTER, IRSTME, IATPSD, IDAY, TIME, ICSA, LPPAR, IASPAM, LUOUT,
Z TCIST, TCILNG, IHEAD, ITEMS
STOP
END

156
SUBROUTINE EDIT

**** ALLOWS EDITING OF DATA IN COMMON LOG
**** J. JONES FEB 79

**** NOTE ALL VARIABLES IN COMMON LOG ARE 2 DIMENSIONAL
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69),
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(1,9),
ZIRSTME(20,3), IATPSD(1,5),
$ IDAY(1,1), TIME(1,1), ICsA(1,20), LPPAR(1,5),
ZIASPAM(4,20), LUOUT(1,1), TCIST(1,1), TCILNG(1,1), LOOK(1,17)
COMMON /QUENUM/
ITEMS(560)
DIMENSION INTGR(10), REAL(10), IWORD(10)
DATA NAME /"IATP", "IASP", "IUNIT", "ITRUCK", "ITYPE",
Z "IMIX", "INTER", "IRSTME", "IASPAM", "IDAY",
Z "TIME", "ICSA", "LPPAR", "IASPAM", "LUOUT", "TCIST",
$ "TCILNG", "LOOK"/
DATA LIMIT1 /4, 4, 75, 560, 6,
Z 40, 1, 20, 1, 1, 1,
Z 1, 1, 1, 4, 1, 1, 1, 17/
DATA LIMIT2 /30, 41, 69, 7, 6,
Z 23, 9, 3, 5, 5, 1,
Z 1, 20, 5, 20, 1, 1, 1, 1/ DATA IEND/"END"
NNAMES = 19
10 WRITE(2,100)
LUI - 1
CALL READF (LUI, 10, INTGR, REAL, IWORD)

**** BRANCH ON DATA TYPE
15 IF(IWORD(1) .EQ. IEND) GO TO 95
DO 20 KTYPE = 1, NNAMES
IF(IWORD(1) .EQ. NAME(KTYPE)) GO TO 30
20 CONTINUE
GO TO 10

**** SET LIMITS FOR DATA TYPE
30 ILOW = INTGR(1)
IHIGH = INTGR(2)
IFLG = 0
IF(ILOW .EQ. 0 .AND. IHIGH .EQ. 0) IFLG = 1
IF(ILOW .EQ. 0) ILOW = 1
IF(IHIGH .EQ. 0) IHIGH = LIMIT1(KTYPE)
IF(IHIGH .EQ. 0) ILOW = LIMIT1(KTYPE)
IHIGH = LIMIT1(KTYPE)
IF(ILOW .GT. LIMIT1(KTYPE)) GO TO 10

**** BACKGROUND HAS BEEN SET, READ CHANGE OR LIST COMMAND
40 WRITE(2,120)
CALL READF (LUI, 10, INTGR, REAL, IWORD)
IF(IWORD(1) .EQ. "LIST" .OR. IWORD(1) .EQ. "L") GO TO 50
IF(IWORD(1) .EQ. "CHANGE" .OR. IWORD(1) .EQ. "C") GO TO 80
GO TO 15

**** LIST COMMAND
50 IATT1 = INTGR(1)
IATT2 = INTGR(2)
IFLG = 0
IF(IATT1 .EQ. 0 .AND. IATT2 .EQ. 0) IFLG = 1
IF(IFLG .EQ. 1) IATT1 = 1
IF(IFLG .EQ. 1) IATT2 = LIMIT2(KTYPE)
IF(IATT2 .GT. LIMIT2(KTYPE)) IATT2 = LIMIT2(KTYPE)
IF(IATT1 .GT. LIMIT2(KTYPE)) GO TO 40
DO 70 INDEX = ILOW, IHIGH
WRITE(2,140) NAME(KTYPE), INDEX
DO 70 IATT = IATT1, IATT2
IF(KTYPE.EQ.1) IVALUE = IATP(INDEX, IATT)
IF(KTYPE.EQ.2) IVALUE = IASP(INDEX, IATT)
IF(KTYPE.EQ.3) IVALUE = IUNIT(INDEX, IATT)
IF(KTYPE.EQ.4) IVALUE = ITRUCK(INDEX, IATT)
IF(KTYPE.EQ.5) IVALUE = ITYPE(INDEX, IATT)
IF(KTYPE.EQ.6) IVALUE = IMIX(INDEX, IATT)
IF(KTYPE.EQ.7) IVALUE = INTER(INDEX, IATT)
IF(KTYPE.EQ.8) IVALUE = IRSTME(INDEX, IATT)
IF(KTYPE.EQ.10) IVALUE = IATPSD(INDEX, IATT)
IF(KTYPE.EQ.11) IVALUE = IDAY(INDEX, IATT)
IF(KTYPE.EQ.12) IVALUE = TIME(INDEX, IATT)
IF(KTYPE.EQ.13) IVALUE = ICSA(INDEX, IATT)
IF(KTYPE.EQ.14) IVALUE = LPPAR(INDEX, IATT)
IF(KTYPE.EQ.15) IVALUE = IASPAM(INDEX, IATT)
IF(KTYPE.EQ.16) IVALUE = LUOUT(INDEX, IATT)
IF(KTYPE.EQ.17) IVALUE = TCIID(INDEX, IATT)
IF(KTYPE.EQ.18) IVALUE = TCILNG(INDEX, IATT)
IF(KTYPE.EQ.19) IVALUE = LOOK(INDEX, IATT)
IF(KTYPE.NE.3) GO TO 60
IF(IATT.NE.6 .AND. IATT.NE.7) GO TO 60
WRITE(2,160) IATT, IVALUE
GO TO 70
60 WRITE(2,150) IATT, IVALUE
70 CONTINUE
GO TO 40
**** CHANGE COMMAND
80 IATT = INTGR(1)
VALUE = INTGR(2) + REAL(1)
IF(IATT .GT. LIMIT2(KTYPE)) GO TO 40
DO 90 INDEX = ILOW, IHIGH
INSERT VALUE IN PROPER ARRAY
IF(KTYPE.EQ.1) IATP(INDEX, IATT) = VALUE
IF(KTYPE.EQ.2) IASP(INDEX, IATT) = VALUE
IF(KTYPE.EQ.3) IUNIT(INDEX, IATT) = VALUE
IF(KTYPE.EQ.3 .AND. (IATT.EQ.6 .OR. IATT.EQ.7))
Z IUNIT(INDEX, IATT) = IWORD(2)
IF(KTYPE.EQ.4) ITRUCK(INDEX, IATT) = VALUE
IF(KTYPE.EQ.5) ITYPE(INDEX, IATT) = VALUE
IF(KTYPE.EQ.6) IMIX(INDEX, IATT) = VALUE
IF(KTYPE.EQ.7) INTER(INDEX, IATT) = VALUE
IF(KTYPE.EQ.8) IRSTME(INDEX, IATT) = VALUE
IF(KTYPE.EQ.10) IATPSD(INDEX, IATT) = VALUE
IF(KTYPE.EQ.11) IDAY(INDEX, IATT) = VALUE
IF(KTYPE .EQ. 12) TIME(INDEX, IATT) = VALUE
IF(KTYPE .EQ. 13) ICSA(INDEX, IATT) = VALUE
IF(KTYPE .EQ. 14) LPPAR(INDEX, IATT) = VALUE
IF(KTYPE .EQ. 15) IASPAM(INDEX, IATT) = VALUE
IF(KTYPE .EQ. 16) LUOUT(INDEX, IATT) = VALUE
IF(KTYPE .EQ. 17) TCIST(INDEX, IATT) = VALUE
IF(KTYPE .EQ. 18) TCIILG(INDEX, IATT) = VALUE
90 CONTINUE
GO TO 40
95 RETURN
100 FORMAT(I1, "VARIABLE NAME = ")
120 FORMAT(I1, "",")
140 FORMAT(/, I1, A10, I5)
150 FORMAT(I1, "ATTRIBUTE ", I4, " = ", I7)
160 FORMAT(I1, "ATTRIBUTE ", I4, " = ", A10)
END
SUBROUTINE READF (LU, NUM, INTGR, REAL, IWORD)
**** RETURNS UP TO NUM INTEGERS, REALS, AND STRINGS.
**** BLANKS AND COMMAS ARE DELIMITERS
**** H. JONES  1979
DIMENSION INTGR(1), REAL(1), IWORD(1)
DIMENSION ICHR(82), IALDIG(10)
DATA IBLANK / " ", IPERD / ". ", ICOMMA / " ", IMINUS / "-" /
DATA IQUOT/IH/
DATA IALDIG / "1", "2", "3", "4", "5", "6", "7", "8", "9", "0" /
ICHR(81) = IBLANK
ICHR(82) = IQUOT
**** READ RECORD, ZERO OUT OLD INTGR, REAL, IWORD
READ(LU,100) (ICHR(I), I=1,80)
IF(EOF(LU) .NE. 0) GO TO 60
DO 4 I=1,NUM
INTGR(I)=0
REAL(I)=0
4 IWORD(I) = IBLANK
KWORD=0
KINTGR=0
KREAL=0
N=0
**** CHECK NEXT CHARACTER IN RECORD
**** SKIPPING BLANKS *************
10 MINUS = 1
11 N=N+1
IF(N.EQ.81) GO TO 60
IF(ICHR(N).EQ.IBLANK) GO TO 11
**** DETERMINE IF CHAR IS NUMBER OR ALPHA
IF(ICHR(N) .EQ. IQUOT) GO TO 41
IF(ICHR(N) .NE. IMINUS) GO TO 12
MINUS = -1
GO TO 11
12 ISTART = N
NUMB=0
IF(ICHR(N).EQ.IPERD) GO TO 28
DO 15 I=1,10
IF(ICHR(N).EQ.IALDIG(I)) GO TO 20
15 CONTINUE
GO TO 40
**** BUILDING INTEGER OR INTEGER PART OF REAL
20 N=N+1
IF(ICHR(N) .NE. IBLANK .AND. ICHR(N) .NE. IPERD .Z .AND. ICHR(N) .NE. ICOMMA ) GO TO 20
**** CALCULATE VALUE OF INTEGER
IEND = N-1
NUMB=0
DO 25 I=ISTART,IEND
DO 24 J=1,9
IF(ICHR(I) .EQ. IALDIG(J)) GO TO 25
24 CONTINUE
J=0
25 NUMB = NUMB + J * 10 **(IEND-I)
IF(ICHR(N) .EQ. IPERD) GO TO 28

***** NUMBER WAS INTEGER, STORE IT, CHECK FOR BLANKS
KINTGR = KINTGR + 1
INTGR(KINTGR) = NUMB * MINUS
GO TO 10

***** NUMBER WAS INTEGER PART OF REAL, NOW BUILD DECIMAL.
28 RNUMB = FLOAT(NUMB)
ISTART = N+1
IF(ICHR(ISTART) .EQ. IBLANK) GO TO 39
30 N=N+1
IF(ICHR(N).NE.IBLANK .AND. ICHR(N).NE.ICOMMA ) GO TO 30

***** CALCULATE VALUE OF DECIMAL
IEND = N-1
IDECPL = 1
NUMB=0
DO 38 I=ISTART,IEND
   DO 34 J=1,9
      IF(ICHR(N) .EQ. IALDIGCJ)) GO TO 35
   34 CONTINUE
      J=0
35 NUMB = NUMB + J * 10**(IEND-I)
38 IDECPL = IDECPL * 10

***** ADD INTEGER AND DECIMAL
DECML=FLOAT(NUMB)/FLOAT(IDECPL)
RNUMB = RNUMB + DECML
39 KREAL = KREAL + 1
REAL(KREAL) = NUMB * MINUS
GO TO 10

***** BUILDING STRING ALPHANUMERIC
40 N=N+1
   IF(ICHR(N).NE.IBLANK .AND. ICHR(N).NE.ICOMMA ) GO TO 40
   GO TO 44
41 ISTART = N+1
42 N=N+1
   IF(ICHR(N) .NE. IQOUT) GO TO 42
44 IEND = N-1
KWORD = KWORD + 1
LENSTR = IEND - ISTART + 1
IF(LENSTR .GT. 10) LENSTR = 10
   ENCODE(LENSTR, 90, IWORD(KWORD)) (ICHR(KKK), KKK=ISTART, IEND)
   GO TO 10
60 RETURN
90 FORMAT(10A1)
100 FORMAT(80A1)
END
d. SUBROUTINE: UPDATE

PURPOSE: Allows faster update of data arrays in COMMON LOG.

COMMON BLOCKS: LOG, QUENUM, QUEPNT

CALLS: None

IS CALLED BY: HUEDIT

CALLING PARAMETERS: None

LOCAL ARRAYS: INTGR(10)--Storage for up to 10 real number fields input from the console.

REAL (10)--Storage for up to 10 real number fields input from the console.

IWORD (10)--Storage for up to 10 alpha numeric fields input from the console.

NAME (19)--Storage for the names of the 19 arrays and variables in COMMON LOG.

LIMIT (19)--Storage for the upper limit on the first index of the arrays and variables in COMMON LOG.

FUNCTIONS: Displays to the operator a message requesting input as to the variable name in COMMON LOG that is of interest.

Accepts from operator the message as to which variable.

Displays message requesting input as to whether a change or replacement of attribute values is desired.

Accepts operators response.

Displays message requesting variable word number, attribute number, and new value or change (--value/+ value).

An input of 0, 0, 0 exists the logic.
An input of "END" exists the program.
SUBROUTINE UPDATE  
**** ALLOWS FASTER UPDATE OF DATA IN COMMON/LOG/ 
**** G. MARTIN JULY 79  
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69), 
Z ITRUCK(560,7), ITYPE(6,6), IMIX(40,23), INTER(1,9), 
Z IRSTME(20,3), IATPSD(1,5), 
$ IDAY(1,1), TIME(1,1), ICSA(1,20), LPPAR(1,5) 
Z IASPAM(4,20), LUOUT(1,1), TCIST(1,1), TCILDN(1,1), LOOK(1,17) 
COMMON /QUENUM/ IHEAD(136) 
COMMON /QUEPNT/ ITEMS(560) 
DIMENSION INTGR(10), REAL(10), IWORD(10) 
DIMENSION NAME(19), LIMIT1(19), LIMIT2(19) 
DATA NAME/*IATP", "IASP", "IUNIT", "ITRUCK", "ITYPE", 
Z "IMIX", "INTER", "IRSTME", "IASPSD", "IATPSD", "IDAY", 
Z "TIME", "ICSAS", "LPPAR", "IASPAM", "LUOUT", "TCIST", 
$ "TCILDN", "LOOK"/ 
DATA LIMIT1 /4, 4, 75, 560, 6, 
Z 40, 1, 20, 1, 1, 1, 
Z 1, 1, 1, 4, 1, 1, 1, 1, 17/ 
DATA LIMIT2 /30, 41, 69, 7, 6, 
Z 23, 9, 3, 5, 1, 
Z 1, 20, 5, 20, 1, 1, 1, 1/ 
DATA END/"END"/ 
.100 PRINT*,"VARIABLE NAME (OR END)"- " 
READ1,NRAY 
1 FORMAT(A6) 
IF(NRAY.EQ."END ")GOTO1000 
DO 200 I=1,19 
INUM=I 
IF(NRAY.EQ.NAME(I))GOTO210 
200 CONTINUE 
PRINT2,NRAY 
2 FORMAT(" VARIABLE NAME ",A6," NOT VALID." 
GOTO100 
210 PRINT*,"CHANGE OR REPLACE? " 
READ3,ICR 
3 FORMAT(A1) 
IF(ICR.EQ."C")OR.ICR.EQ."R")GOTO300 
PRINT*,"ENTER C TO CHANGE (+ OR -) EXISTING VALUES " 
PRINT*,"R TO REPLACE EXISTING VALUES" 
GOTO210 
300 PRINT*,"ENTER WORD#,ATTRIBUTE#,NEW VALUE (OR CHANGE)" 
PRINT*,"(0,0,0 WHEN DONE)"- " 
310 READ*,IWD,IAT,VAL 
IF(IWD.EQ.0)GOTO1000 
IF(IWD.LT.0.OR.IWD.GT.LIMIT1(INUM))GOTO320 
IF(IWD.GT.0.AND.IAT.LE.LIMIT2(INUM))GOTO340 
320 PRINT*,"WORD# OR ATTRIB.# INVALID--ENTRY IGNORED" 
330 PRINT*,"NEXT- " 
GOTO340 
340 IF(ICR.EQ."C")GOTO350 
IF(INUM.EQ.1)IATP(IWD,IAT)=IFIX(VAL) 
IF(INUM.EQ.2)IASP(IWD,IAT)=IFIX(VAL)

163
IF(INUM.EQ.3)IUNIT(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.4)ITRUCK(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.5)ITYPE(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.6)IMIX(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.7)INTER(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.8)IRSTME(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.10)IATPSD(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.11)IDAY(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.12)TIME(IWD,IAT)=VAL
IF(INUM.EQ.13)ICSA(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.14)LPPAR(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.15)IASPAM(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.16)LOUT(IWD,IAT)=IFIX(VAL)
IF(INUM.EQ.17)TCIST(IWD,IAT)=VAL
IF(INUM.EQ.18)TCILNG(IWD,IAT)=VAL
IF(INUM.EQ.19)LOOK(IWD,IAT)=IFIX(VAL)
GOTO330

350 IF(INUM.EQ.1)IATP(IWD,IAT)=IATP(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.2)IASP(IWD,IAT)=IASP(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.3)IUNIT(IWD,IAT)=IUNIT(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.4)ITRUCK(IWD,IAT)=ITRUCK(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.5)ITYPE(IWD,IAT)=ITYPE(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.6)IMIX(IWD,IAT)=IMIX(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.7)INTER(IWD,IAT)=INTER(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.8)IRSTME(IWD,IAT)=IRSTME(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.10)IATPSD(IWD,IAT)=IATPSD(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.11)IDAY(IWD,IAT)=IDAY(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.12)TIME(IWD,IAT)=TIME(IWD,IAT)+VAL
IF(INUM.EQ.13)ICSA(IWD,IAT)=ICSA(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.14)LPPAR(IWD,IAT)=LPPAR(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.15)IASPAM(IWD,IAT)=IASPAM(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.16)LOUT(IWD,IAT)=LOUT(IWD,IAT)+IFIX(VAL)
IF(INUM.EQ.17)TCIST(IWD,IAT)=TCIST(IWD,IAT)+VAL
IF(INUM.EQ.18)TCILNG(IWD,IAT)=TCILNG(IWD,IAT)+VAL
IF(INUM.EQ.19)LOOK(IWD,IAT)=LOOK(IWD,IAT)+IFIX(VAL)
GOTO330

1000 RETURN
END
e. CALL ROUTINE: HJDATABASE

PURPOSE: Called to obtain a print out of the existing data base.

ATTACHES: HCDATABASE

IS CALLED BY: Operator

FUNCTIONS: Operator must first attach as TAPE 1 the existing data base he wants to print out. Then the operator calls HJDATABASE, ID=. This call routine connects the keyboard and display, attaches HCDATABASE and runs the program. When finished all auxiliary files are returned and the output tape rewound.

HJDATABASE

CONNECT, KEYBRD, DISPLY.
ATTACH, CC, HCDATABASE, ID=CARTSV.
CC(TAPE 1, KEYBRD, DISPLY, INPUT, OUTPUT, TAPE 6=OUTPUT)
RETURN, CC, KEYBRD, DISPLY, TAPE 1.
REWIND, OUTPUT
*EOR
*EOF

f. CALL ROUTINE: HCDATABASE (Binary File at Program DATA)

PURPOSE: To print data contained in data file.

COMMON BLOCKS: LOG, QUENUM, QUEPNT

CALLS: None

IS CALLED BY: HJDATABASE

LOCAL ARRAYS: None

FUNCTIONS: Displays to the operator a message requesting input as to files to be printed.

Accomplishes printing of file/files to output as requested by operator.
PROGRAM DATA (TAPE1, TAPE2, TAPE3, INPUT, OUTPUT, TAPE6=OUTPUT)

COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69), ITRUCK(560,7),
Z ITYPE(6,6), IMIX(40,23), INTER(9), IRSTME(20,3), IATPSD(5),
Z IDAY, TIME, ICSA(20), LPPAR(5), IASPAM(4,20), LOUT, TCIST,
Z TCILNG, LOOK(17)

COMMON /QUENUM/ IHEAD(136)
COMMON /QUEPNT/ ITEMS(560)

DATA LOOK /17*0/
READ(1) IATP, IASP, IUNIT, ITRUCK, ITYPE, IMIX, INTER, IRSTME, IATPSD, IDAY,
Z, TIME, ICSA, LPPAR, IASPAM, LOUT, TCIST, TCILNG, IHEAD, ITEMS
WRITE(6,110)

110 FORMAT(10X,"ARM DATA BASE")
10 WRITE(3,100)
100 FORMAT(" ARM DATA BASE PRINT OPTIONS :",/,
Z " (1) - PRINT ALL" /,
Z " (2) - ATP" /,
Z " (3) - ASP" /,
Z " (4) - UNIT" /,
Z " (5) - TRUCK" /,
Z " (6) - REMAINING" /,
Z " (7) - STOP" /,
Z " ? " )
READ(2,*) IANS
IF(IANS.LT.1.OR.IANS.GT.7) GO TO 10
GO TO (30,30,40,50,60,70,80), IANS

C**** ATP
30 WRITE(6,120)
120 FORMAT(//,26X,"****** ATP DATA ******",//,11X,"ATP 1",11X,
DO 20 I=1,30
20 WRITE(6,130) I,(IATP(J,I),J=1,4)
130 FORMAT(3X,12,5X,16,3(1OX,16))
IF(IANS.NE.1) GO TO 10

C**** ASP
40 WRITE(6,140)
140 FORMAT(//,26X,"****** ASP DATA ******",//,11X,"ASP 1",11X,
DO 21 I=1,41
21 WRITE(6,130) I,(IASP(J,I),J=1,4)
IF(IANS.NE.1) GO TO 10

C**** UNIT
50 WRITE(6,160)
160 FORMAT(//,11X,"****** UNIT DATA ******",//,1X,"UNIT",6X,
Z "SHORT",3X,"TOT RND"
K=1
L=7
55 DO 22 I=K,L
IF(IUNIT(I,1).EQ.0) GO TO 35
WRITE(6,170) I,(IUNIT(I,J),J=1,7),IUNIT(I,68),IUNIT(I,69)

166
170 FORMAT(/,2X,I2,9X,I1,8X,I1,9X,I1,10X,I2,13X,I3,8X,A10,6X,A10,
Z 8X,I1,9X,I1)
   WRITE(6,180) (ITUNIT(I,J),J=8,67)
180 FORMAT(12X,I2,8X,I2,11X,I2,9X,I5,7X,I5,3X,I5,5X,I5,3X,I5,2X,
Z I7,8X,I2,6X,I2,4X,I7)
35 IF(I.EQ.75) GO TO 57
22 CONTINUE
   K=L+1
   L=L+7
   WRITE(6,160)
   GO TO 95
57 IF(IANS.NE.1) GO TO 10
C**** TRUCK
60 WRITE(6,190)
190 FORMAT(/H1,29X,"TRUCK DATA "/*/1H"MISSION",4X,"STATUS",4X,"OWNER",6X,"MIX",6X,
Z ","LOAD",5X,"LAST FAIL")
   K=1
   L=56
65 DO 23 I=K,L
   WRITE(6,200) I,(ITRUCK(I,J),J=1,7)
200 FORMAT(2X,I1,8X,I1,11X,I1,7X,I3,8X,I2,7X,I3,9X,I4)
   IF(I.EQ.560) GO TO 67
23 CONTINUE
   K=L+1
   L=L+56
   WRITE(6,190)
   GO TO 65
67 IF(IANS.NE.1) GO TO 10
C**** MIX
70 WRITE(6,210)
210 FORMAT(/H1,41X,"MIX DATA "/*/1H"MIX",3X,"1",
   DO 24 I=1,40
   WRITE(6,220) I,(IMIX(I,J),J=1,23)
220 FORMAT(2X,I2,23(1X,I4))
   24 CONTINUE
C**** ATPSD
70 WRITE(6,230) (IATPSD(I),I=1,5)
230 FORMAT(/,5X,"ATP SDK SERVICE DATA "/*/5(3X,I2))
C**** DAY,TCIST,TCILNG,TIMD,LUOUT
   WRITE(6,240) IDAY,TCIST,TCILNG,TIMD,LUOUT
240 FORMAT(/,5X,"MISC DATA "/*/5X,"IDAY = ",I1,5X,
Z ,"LUOUT = ",I2)
C**** ASPAM
   WRITE(6,250)
250 FORMAT(1H1,26X,"ASP AMMO REMOVED "/*/11X,"ASP 1",
   DO 25 I=1,20
   25 WRITE(6,260) I,(IASPAM(J,I),J=1,4)
C**** RSTME
  WRITE(6,270)
  270 FORMAT(//,10X,"****** RESUPPLY TIME DATA ******",//,10X,"SETUP",
        Z 10X,"LOAD/100",10X,"TRAVEL",/)
  DO 26 I=1,20
  26 WRITE(6,280) I,(IRSTME(I,J),J=1,3)
  280 FORMAT(4X,I2,4X,13X,I4,13X,I4)
C**** TYPE
  WRITE(6,290)
  290 FORMAT(//,7X,"****** TRUCK SPEEDS, MTBF, AND MTTR ******",//,
        Z 3X,"TRUCK",3X,"2D NT",3X,"2D DAY",2X,"HI NT",3X,"HI DAY",3X,
        Z "MTBF",5X,"MTTR",/)
  DO 27 I=1,6
  27 WRITE(6,300) I,(ITYPE(I,J),J=I,6)
  300 FORMAT(5X,I2,4(5X,I3),2(5X,I4))
C**** INTER
  WRITE(6,310) (INTER(I),I=1,9)
  310 FORMAT(1H1,20X,"****** INTERDICTION DATA ******",//,9(5X,I3))
C**** CSA
  WRITE(6,320)
  320 FORMAT(//,5X,"****** AMMO FROM CSA ******",//,9X,"AMMO",11X,
        Z "AMT")
  DO 28 I=1,20
  28 WRITE(6,330) I,ICSA(I)
  330 FORMAT(10X,I2,10X,I5)
C**** LPPAR
  WRITE(6,340) (LPPAR(I),I=1,5)
  340 FORMAT(//,7X,"****** LPPAR ******",//,5X,I2,2(5X,I1),5X,I3,
        Z 5X,I2,/)!
  IF(IANS.NE.1) GO TO 10
  10 WRITE(6,350)
  350 FORMAT(1X,"END OF DATA")
STOP
END
g. PROGRAM: HSREADJIF
PURPOSE: To read Jiffy produce demand files that is provided as input to ARM.
COMMON BLOCKS: None
CALLS: None
IS CALLED BY: HJREADJIF
CALLING PARAMETERS: None
LOCAL ARRAYS: None
FUNCTIONS: Read Jiffy produced binary file.

Provides a means of looking at input generated by the attrition model.
PROGRAM CHECK(INPUT, OUTPUT, TAPIN, TAPE5=INPUT, TAPE6=OUTPUT, TAPE2=TAPIN)
DIMENSION INFILE(64)
1 READ(2) INFILE
   IF(EOF(2)) 100, 2
2 WRITE(6, 200)(INFILE(I), I=1, 26)
200 FORMAT(/, 1X, A10, 2X, 5F8.3, /, (13X, 5F8.3))
   GO TO 1
100 STOP
END
*EOR
*EOF
h. PROGRAM: HSRDJIFCH

PURPOSE: To enable the operator to change the ammunition expenditure data generated by Jiffy.

COMMON BLOCKS: None

CALLS: INPUT, OUTPUT

IS CALLED BY: Operator

CALLING PARAMETERS: None

LOCAL ARRAYS: FILE (64)--storage for up to 64 words read from the Jiffy produced binary file.

FUNCTIONS: Displays a message to the operator requesting to know what changes are to be made, single field, all of one ammunition type, or all ammunition of all records.

Accepts from operator desired response and displays subsequent to appropriate message.

Allows operator to change the various ammunition expenditures obtained from Jiffy by multiplying expenditures by a decimal factor.
PROGRAM CHANG(INPUT, OUTPUT, TAPIN, TAPOUT, TAPE1=TAPIN, TAPE2=TAPOUT *, TAPE6=OUTPUT, TAPE21)

C CAN READ, DISPLAY AND CHANGE BINARY INPUT FILE TO ARM.

DIMENSION FILE(64)
SET FOR INTERACTIVE USE.
CALL CONNEC(SLINPUT)
CALL CONNEC(SOUTPUT)
N1 = 1
N2 = 2

PRINT*, " IF CHANGE IS TO SINGLE FIELD IN 1 RECORD.....ENTER 1"
PRINT*, " IF CHANGE IS TO ALL OF 1 AMMO.............ENTER 2"
PRINT*, " IF CHANGE IS TO ALL AMMO OF ALL RECORDS......ENTER 3"
PRINT*, IAMMO = 0
READ*, IFLG
IF(IFLG.LT.1.OR.IFLG.GT.3) GO TO 1
IF(IFLG.NE.2) GO TO 3

PRINT*, " AMMUNITION TO BE CHANGED."
PRINT*,
READ*, IAMMO
IF(IAMMO.LT.1.OR.IAMMO.GT.25) GO TO 2

PRINT*, " AMMUNITION TO BE CHANGED."
PRINT*,
READ*, IAMMO
IF(IAMMO.LT.1.OR.IAMMO.GT.25) GO TO 2
IF(IFLG.EQ.1) GO TO 4

PRINT*," ENTER FACTOR (DECIMAL (1.5)) TO MULTIPLY BY."
PRINT*,
READ * FACTOR
C READ RECORDS ADD CHANGE.
4 READ(NI) FILE
IF(EQF(NI)) 50,5
5 IF(IFLG.GT.1) GO TO 7
WRITE(6,100) (FILE(I),I=1,26)
100 FORMAT(/1X,A1O,2X,5F8.1,/,(13X,5F8.1))

NOW MUST DECIDE IF WANT TO CHANGE A FIELD IN THIS RECORD.

PRINT *, " DO YOU WISH TO CHANGE A FIELD IN "
PRINT *, " THIS RECORD (Y OR N).";
PRINT*,
READ200,WISH
200 FORMAT(A1)
IF(WISH.EQ."N") GO TO 10
PRINT *, " ENTER FIELD NUMBER (2 - 26)."
PRINT*,
READ *.IFLD
PRINT *, " ENTER NEW VALUE."
PRINT*,
READ *.VALUE
FILE(IFLD) = VALUE
GO TO 6
C CHANGE RECORD.
  7 DO 8 I=1,5
       IND = 5 * (I-1) + 3
       IF(IAMMO.NE.FILE(IND)) GO TO 8
       FILE(IND+3) = FILE(IND+3) * FACTOR
  8 CONTINUE
C WRITE OUT RECORD.
 10 WRITE(N2) FILE
     GO TO 4
 50 PRINT*:" CHANGE ANOTHER AMMO ? (Y OR N)."
      PRINT*, " "
      READ 300,IANS
 300 FORMAT (A1)
      IF(IANS.EQ."N") GO TO 60
      REWIND 1
      REWIND 2
      N1= 3 - N1
      N2= 3 - N2
      GO TO 2
 60 PRINT*:" GOOD OUTPUT ON TAPE 21"
      REWIND N2
      DO 99 I=1,999
         READ(N2) FILE
         IF(EOF(N2)) 999,88
 88 WRITE(21) FILE
 99 CONTINUE
999 CONTINUE
      REWIND 21
      STOP
END
*EOR
*EOF
*EOR
*EOF
i. PROGRAM: TRKQUE

PURPOSE: To enable the operator to put the trucks in their respective queues as part of the initial data base development.

COMMON BLOCKS: LOG, QUENUM, QUEPNT

CALLS: TRKPUT PRINT

IS CALLED BY: HJTRKQUE

CALLING PARAMETERS: None

LOCAL ARRAYS: None

FUNCTIONS: Displays a message to the operator requesting to know if modification of truck queues is desired.

If a positive response is made it calls subroutine TRKPUT which allows modification of the truck queues.

If a negative response is made it asks if a printout of queue contents is desired.
PROGRAM TRKQUE(TAPE1, TAPE2, TAPE3, TAPE4, INPUT, OUTPUT, TAPE6=OUTPUT)
COMMON /LOG/ IATP(4,30), IASP(4,41), IUNIT(75,69), ITRUCK(560,7),
Z IYPE(6,6), IMIX(25,17), INTER(9), IRSTME(14,3), IATPSD(5),
Z IDAY, TIME, ICSA(14), LPPAR(5), IASPAM(4,14), LUOUT, TCIST,
Z Tcilng, LOOK(17)
COMMON /QUENUM/ IHEAD(136)
COMMON /QUEPNT/ ITEV1S(560)
DATA LOOK/17*0/
READ(1) IATP, IASP, IUNIT, ITRUCK, IYPE, IMIX, INTER, IRSTME, IATPSD, IDAY,
Z, TIME, ICSA, LPPAR, IASPAM, LUOUT, TCIST, Tcilng, IHEAD, ITEMS
10 WRITE(3,100)
100 FORMAT(" MODIFY TRUCK QUEUES? (YES/NO) ")
READ(2,110) IANS
110 FORMAT(A10)
   IF(IANS.EQ."NO".OR.IANS.EQ."N") GO TO 40
   CALL TRKPUT
40 WRITE(3,120)
120 FORMAT(" PRINT OUT CONTENTS OF QUEUES? (YES/NO) ")
READ(2,110) IANS
   IF(IANS.EQ."NO".OR.IANS.EQ."N") GO TO 50
   CALL PRINT
   GO TO 10
50 WRITE(4) IATP, IASP, IUNIT, ITRUCK, IYPE, IMIX, INTER, IRSTME,
Z IATPSD, IDAY, TIME, ICSA, LPPAR, IASPAM, LUOUT, TCIST, Tcilng,
Z IHEAD, ITEMS
STOP
END

SUBROUTINE TRKPUT
C**** ALLows INTERACTIVE TRUCK QUEUE RE-ASSIGNMENT
C**** H. JONES FEB 79
DIMENSION INTGR(10), REAL(10), IWORD(10)
C
WRITE(2,10)
10 FORMAT(A10)
   CALL READF (2, 10, INTGR, REAL, IWORD)
   IF(IWORD(1) .EQ. "END" .OR. IWORD(1) .EQ. "E") GO TO 50
   IF(IWORD(1) .EQ. "PUT" .OR. IWORD(1) .EQ. "P") GO TO 30
   IF(IWORD(1) .EQ. "LIST" .OR. IWORD(1) .EQ. "L") GO TO 40
   IF(IWORD(1) .EQ. "GET" .OR. IWORD(1) .EQ. "G") GO TO 25
   IF(IWORD(1) .EQ. "TAKE" .OR. IWORD(1) .EQ. "T") GO TO 60
   GO TO 15
C
175
* C**** GET TRUCK FROM QUEUE WITHOUT RE-ORDERING QUEUE

25 I1 = INTGR(1)
I2 = INTGR(2)
    IF(INTGR(3) .NE. 0) GO TO 15
    IFLAG = 0
    CALL NXTQUE (IFIRST, I2)

26 CALL NXTQUE(NTRK, I2)
    IF(NTRK.EQ.0) GO TO 15
    IF(NTRK .EQ. IFIRST .AND. IFLAG .NE. 0) GO TO 15
    CALL GETQUE(NTRK, I2)
    IF(I1 .EQ. IFIRST) GO TO 15
    IF(I1 .NE. NTRK) CALL PUTQUE (NTRK, I2)
    IFLAG = 1
    GO TO 26

* C**** PUT TRUCK IN QUEUE

30 I1 = INTGR(1)
I2 = INTGR(2)
I3 = INTGR(3)
    IF(INTGR(3) .EQ. 0) I3 = INTGR(2)
    IF(INTGR(3) .EQ. 0) I2 = INTGR(1)
    DO 35 I=I1,I2
35 CALL PUTQUE (I, I3)
    GO TO 15

* C**** LIST TRUCKS IN QUEUE

40 CALL NXTQUE (IFIRST, INTGR(1))
    IF(IFIRST .EQ. 0) GO TO 15
42 CALL GETQUE(NTRK, INTGR(1))
    CALL PUTQUE(NTRK, INTGR(1))
    WRITE(2,45) NTRK
45 FORMAT(IX,5)
    CALL NXTQUE (INEXT, INTGR(1))
    IF(INEXT .NE. IFIRST) GO TO 42
    GO TO 15

* C**** TAKE ALL TRUCKS OUT OF QUEUES

60 CALL SETQUE (560, 136)
    GO TO 15

* 50 RETURN

END
SUBROUTINE READF (LU, NUM, INTGR, REAL, IWORD)

C**** RETURNS UP TO NUM INTEGERS, REALS, AND STRINGS.

C**** BLANKS AND COMMAS ARE DELIMITERS

C**** H. JONES 1979

DIMENSION INTGR(1), REAL(1), IWORD(1)
DIMENSION ICHR(82), IALDIG(IO)
DATA IBLANK /* */ , IPERD /* . */, ICOMMA /* , */ , IMINUS /* - */
DATA IQUOT/IH /* /
DATA IALDIG/"1","2","3","4","5","6","7","8","9","0"/
ICH(81) = IBLANK
ICH(82) = IQUOT

C
C**** READ RECORD, ZERO OUT OLD INTGR, REAL, IWORD
READ(LU,100) (ICH(I), I=1,80)
IF(EOF(LU) .NE. 0) GO TO 60
DO 4 I=1,NUM
INTGR(I)=0
REAL(I)=0.
4 IWORD(I) = IBLANK
KWORD=0
KINTGR=0
KREAL=0
N=0

C
C**** CHECK NEXT CHARACTER IN RECORD
C**** SKIPPING BLANKS ***********
10 MINUS = 1
11 N=N+1
IF(N.EQ.81) GO TO 60
IF(ICH(N).EQ.IBLANK) GO TO 11


C
C**** DETERMINE IF CHAR IS NUMBER OR ALPHA
IF(ICH(N) .EQ. IQUOT) GO TO 41
IF(ICH(N) .NE. IMINUS) GO TO 12
MINUS = -1
GO TO 11
12 ISTART = N
NUMB=0
IF(ICH(N).EQ.IPERD) GO TO 28
DO 15 I=1,10
IF(ICH(N).EQ.IALDIG(I)) GO TO 20
15 CONTINUE
GO TO 40

C
C**** BUILDING INTEGER OR INTEGER PART OF REAL
20 N=N+1
IF(ICH(N) .NE. IBLANK .AND. ICHR(N) .NE. IPERD
  .AND. ICHR(N) .NE. ICOMMA ) GO TO 20

C
C**** CALCULATE VALUE OF INTEGER
IEND = N-1
NUMB=0
DO 25 I=ISTART,IEND

177
DO 24 J=1,9
   IF(ICHRI .EQ. IALDI(J)) GO TO 25
24 CONTINUE
   J=0
25 NUMB = NUMB + J * 10 **(IEND-I)
   IF(ICHRI .EQ. IPERD) GO TO 28
C
C***** NUMBER WAS INTEGER, STORE IT, CHECK FOR BLANKS
   KINTGR = KINTGR +1
   INTGR(KINTGR) = NUMB * MINUS
   GO TO 10
C
C***** NUMBER WAS INTEGER PART OF REAL, NOW BUILD DECIMAL.
28 RNUMB = FLOAT(NUMB)
   ISTART = N+1
   IF(ICHRI(ISTART) .EQ. IBLANK) GO TO 39
30 N=N+1
   IF(ICHRI(N) .NE. IBLANK .AND. ICHR(N) .NE. ICOMMA) GO TO 30
C
C***** CALCULATE VALUE OF DECIMAL
   IEND = N-1
   IDECPL = 1
   DO 38 I=ISTART,IEND
      DO 34 J=1,9
         IF(ICHRI(I) .EQ. IALDI(J)) GO TO 35
34 CONTINUE
      J=0
35 NUMB = NUMB + J * 10 **(IEND-I)
38 IDECPL = IDECPL * 10
C
C***** ADD INTEGER AND DECIMAL
   DECML=FLOAT(NUMB)/FLOAT(IDECPL)
   RNUMB = RNUMB + DECML
   KREAL = KREAL + 1
   REAL(KREAL) = RNUMB * MINUS
   GO TO 10
C
C***** BUILDING STRING ALPHANUMERIC
40 N=N+1
   IF(ICHRI(N) .NE. IBLANK .AND. ICHR(N) .NE. ICOMMA) GO TO 40
   GO TO 44
41 ISTART = N+1
42 N=N+1
   IF(ICHRI(N) .NE. IQUOT) GO TO 42
44 IEND = N-1
   KWORD = KWORD + 1
   LENSTR = IEND - ISTART + 1
   IF(LENSTR .GT. 10) LENSTR = 10
   ENCODE(LENSTR, 90, IWORD(KWORD)) (ICHRI(KKK), KKK-ISTART, IEND)
   GO TO 10
C
60 RETURN
SUBROUTINE NXTQUE (ITEM, NUMQUE)
C**** SHOWS NEXT ITEM IN QUEUE (LEAVES IT IN)
C**** H. JONES FEB 79
COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPNT(560)
ITEM = 0
LITEM = 0
IPOINT = NHEAD(NUMQUE)
C
10 IF(IPOINT .EQ. 0) GO TO 20
LITEM = ITEM
ITEM = IPOINT
IPOINT = IPNT(ITEM)
GO TO 10
C
20 RETURN
END

SUBROUTINE PUTQUE (ITEM, NUMQUE)
C**** PUTS ITEM IN QUEUE NUMQUE
C**** H. JONES DEC 78
COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPNT(560)
IOLDH = NHEAD(NUMQUE)
NHEAD(NUMQUE) = ITEM
IPNT(ITEM) = IOLDH
RETURN
END

SUBROUTINE GETQUE (ITEM, NUMQUE)
C**** GETS ITEM FROM QUEUE NUMQUE
C**** TO GET TRUCK FROM QUEUE 4 -- CALL GETQUE (N,4)
C**** H. JONES DEC 78
COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPNT(560)
ITEM = 0
LITEM = 0
IPOINT = NHEAD(NUMQUE)
C
10 IF(IPOINT .EQ. 0) GO TO 20
LITEM = ITEM
ITEM = IPOINT
IPOINT = IPNT(ITEM)
GO TO 10
20 IF(LITEM .GT. 0) IPNT(LITEM) = 0
IF(LITEM .EQ. 0) NHEAD(NUMQUE) = 0

SUBROUTINE SETQUE (ITEMS, NUMQUE)
C**** SETS UP NUMQUE EMPTY QUEUES FOR ITEMS.
C H. JONES DEC 78
COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPNT(560)
DO 10 I=1,NUMQUE
10 NHEAD(I) = 0
DO 20 I=1,ITEMS
20 IPNT(I) = 0
RETURN
END

SUBROUTINE PRINT
C**** PRINTS OUT THE CONTENTS OF EVERY TRUCK QUEUE
C D. HILLIS APR 79
COMMON /QUENUM/ NHEAD(136)
COMMON /QUEPNT/ IPNT(560)
DIMENSION NTRK(25)
DO 100 I=1,136
CALL NXTQUE(IFIRST,I)
IF(IFIRST.EQ.0) GO TO 50
DO 90 J=1,25
CALL GETQUE(NTRK(J),I)
CALL PUTQUE(NTRK(J),I)
CALL NXTQUE(INEXT,I)
IF(INEXT.EQ.IFIRST) GO TO 40
90 CONTINUE
40 WRITE(6,200) I
200 FORMAT(/,5X,"QUEUE ",3," TRUCKS")
WRITE(6,210) (NTRK(K),K=1,J)
210 FORMAT(10(IX,I3))
GO TO 100
50 WRITE(6,200) I
WRITE(6,220)
220 FORMAT(5X,"NONE")
100 CONTINUE
RETURN
END
<table>
<thead>
<tr>
<th>ORGANIZATION</th>
<th>NO. OF COPIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Training and Doctrine Command</td>
<td>1</td>
</tr>
<tr>
<td>Fort Monroe, VA 23651</td>
<td></td>
</tr>
<tr>
<td>ATCD-SI (Mr. Christman)</td>
<td></td>
</tr>
<tr>
<td>ATCD-C</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>Defense Documentation Center</td>
<td></td>
</tr>
<tr>
<td>Cameron Station</td>
<td></td>
</tr>
<tr>
<td>Alexandria, VA 22314</td>
<td></td>
</tr>
<tr>
<td>Director</td>
<td></td>
</tr>
<tr>
<td>USATRASANA</td>
<td></td>
</tr>
<tr>
<td>ATTN: ATAA-PFB</td>
<td></td>
</tr>
<tr>
<td>White Sands Missile Range, NM 88002</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>USA Logistics Center</td>
<td></td>
</tr>
<tr>
<td>Ft Lee, VA 23801</td>
<td></td>
</tr>
<tr>
<td>ATCL-C</td>
<td></td>
</tr>
<tr>
<td>ATCL-CF</td>
<td></td>
</tr>
<tr>
<td>ATCL-LE</td>
<td></td>
</tr>
<tr>
<td>ATCL-OS</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Air Defense Center &amp; Fort Bliss</td>
<td></td>
</tr>
<tr>
<td>ATTN: ATSA-CD-C</td>
<td></td>
</tr>
<tr>
<td>Fort Bliss, TX 79916</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Aviation Center &amp; Fort Rucker</td>
<td></td>
</tr>
<tr>
<td>ATTN: ATZQ-D-CC</td>
<td></td>
</tr>
<tr>
<td>Fort Rucker, AL 36362</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Armor Center &amp; Fort Knox</td>
<td></td>
</tr>
<tr>
<td>ATTN: ATSB-CD-S</td>
<td></td>
</tr>
<tr>
<td>Fort Knox, KY 40121</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Engineer Center &amp; Fort Belvoir</td>
<td></td>
</tr>
<tr>
<td>ATTN: ATSE-CD-CS</td>
<td></td>
</tr>
<tr>
<td>Fort Belvoir, VA 22060</td>
<td></td>
</tr>
<tr>
<td>ORGANIZATION</td>
<td>NO. OF COPIES</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Field Artillery School</td>
<td>2</td>
</tr>
<tr>
<td>ATTN: ATSF-CTD-S</td>
<td></td>
</tr>
<tr>
<td>Fort Sill, OK 73503</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Infantry School</td>
<td>2</td>
</tr>
<tr>
<td>ATTN: ATSH-CD-CS</td>
<td></td>
</tr>
<tr>
<td>Fort Benning, GA 31905</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Intelligence Center and School</td>
<td>1</td>
</tr>
<tr>
<td>ATTN: ATSI-CD-CS</td>
<td></td>
</tr>
<tr>
<td>Fort Huachuca, AZ 86611</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>US Army Missile and Munitions Center and School</td>
<td>2</td>
</tr>
<tr>
<td>ATTN: ATSK-CD-CS</td>
<td></td>
</tr>
<tr>
<td>Redstone Arsenal, AL 35809</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>USA Ordnance Center and School</td>
<td>1</td>
</tr>
<tr>
<td>ATTN: ATSL-CD-CS</td>
<td></td>
</tr>
<tr>
<td>Aberdeen Proving Grounds, MD 21005</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>USA Institute of Military Assistance</td>
<td>1</td>
</tr>
<tr>
<td>DCDMDT Cmt Tng Div</td>
<td></td>
</tr>
<tr>
<td>Fort Bragg, NC 28307</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>USA Transportation School</td>
<td>2</td>
</tr>
<tr>
<td>ATTN: ATSP-CD-CS</td>
<td></td>
</tr>
<tr>
<td>Fort Eustis, VA 23604</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>USA Concepts Analysis Agency</td>
<td>1</td>
</tr>
<tr>
<td>8120 Woodmont Avenue</td>
<td></td>
</tr>
<tr>
<td>Bethesda, MD 20014</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td></td>
</tr>
<tr>
<td>USA Combined Arms Combat Developments Activity</td>
<td></td>
</tr>
<tr>
<td>ATZLCA-CA</td>
<td>5</td>
</tr>
<tr>
<td>ATZLCA-SW</td>
<td>2</td>
</tr>
<tr>
<td>Fort Leavenworth, KS 66027</td>
<td></td>
</tr>
<tr>
<td>ATZLCA-CA</td>
<td></td>
</tr>
<tr>
<td>ATZLCA-SW</td>
<td></td>
</tr>
</tbody>
</table>

A-2
<table>
<thead>
<tr>
<th>ORGANIZATION</th>
<th>NO. OF COPIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commander</td>
<td>1</td>
</tr>
<tr>
<td>Command and General Staff College</td>
<td></td>
</tr>
<tr>
<td>ATTN: ATZLSW-TA</td>
<td></td>
</tr>
<tr>
<td>Fort Leavenworth, KS 66027</td>
<td></td>
</tr>
<tr>
<td>Deputy Commander</td>
<td>1</td>
</tr>
<tr>
<td>USAMSAA</td>
<td></td>
</tr>
<tr>
<td>ATTN: AMXSY-T</td>
<td></td>
</tr>
<tr>
<td>Aberdeen Proving Ground, MD 21005</td>
<td></td>
</tr>
<tr>
<td>US Air Force</td>
<td>2</td>
</tr>
<tr>
<td>Tactical Fighter Weapons Center/SATC</td>
<td></td>
</tr>
<tr>
<td>ATTN: TFWC-SA</td>
<td></td>
</tr>
<tr>
<td>Nellis AFB, NV 89191</td>
<td></td>
</tr>
<tr>
<td>Professor S. H. Parry, Code 55Py</td>
<td>1</td>
</tr>
<tr>
<td>Department of Operations Research</td>
<td></td>
</tr>
<tr>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td>Monterey, CA 73940</td>
<td></td>
</tr>
</tbody>
</table>

A-3