DTIC FILE COPY

ARI Research Note 87-09

AD-A179 892

Program GDETAP Documentation

Jack A. Briscoe
The BDM Corporation

for

ARI Field Unit at Presidio of Monterey, California
James H. Banks, Chief

TRAINING RESEARCH LABORATORY
Jack H. Hiller, Director

DTIC
ELECTE
APR 2 3 1987

U. S. Army
Research Institute for the Behavioral and Social Sciences

January 1987

Approved for public release; distribution unlimited.

87 4

DISCLAIMER NOTICE

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

U. S. ARMY RESEARCH INSTITUTE

FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the

Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Technical Director

WM. DARRYL HENDERSON
COL, IN
Commanding

Research accomplished under contract
for the Department of the Army

The BDM Corporation

Technical Review by

James H. Banks

QUALITY INSPECTED 1

A1 23

This report, as submitted by the contractor, has been cleared for release to Defense Technical Information Center (DTIC) to comply with regulatory requirements. It has been given no primary distribution other than to DTIC and will be available only through DTIC or other reference services such as the National Technical Information Service (NTIS). The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other official documentation.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
ARI Research Note 87-09

2. GOVT ACCESSION NO. AD-A179892

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Program GDETAP Documentation

5. TYPE OF REPORT & PERIOD COVERED
Final Report
May 1986

6. PERFORMING ORG. REPORT NUMBER
BDM/ARI-TR-0041-86

7. AUTHOR(s)
Jack A. Briscoe

8. CONTRACT OR GRANT NUMBER(s)
MDA 903-85-C-0472

9. PERFORMING ORGANIZATION NAME AND ADDRESS
The BDM Corporation
2600 Garden Road, North Building
Monterey, CA 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
2Q263743A794
511C1

11. CONTROLLING OFFICE NAME AND ADDRESS
ARI Field Unit at Presidio of Monterey, California
P.O. Box 5787,
Presidio of Monterey, CA 93944

12. REPORT DATE
January 1987

13. NUMBER OF PAGES
42

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
U.S. Army Research Institute for the Behavioral and Social Sciences, 5001 Eisenhower Avenue, Alexandria, VA 22333-5600

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
- -

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
- -

18. SUPPLEMENTARY NOTES
James H. Banks, contracting officer's representative

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Input	Processing	VAX
Output	GDETAP Program	Fortran
RDMS	Digital Data	National Training Center
Data	Initialization List	

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

GDETAP is a VAX Fortran program which can read the GDE log tape produced at the National Training Center (NTC), and produce reports and files on the data the tape contains. There are two separate sources of digtal data at NTC, the GDE tape, and BACKUP tapes created from files set up using the CIS VAX-780 computer.

Comparing the position/location data from the GDE tapes with the files created from the BACKUP tapes will be useful for several reasons: (OVER)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ARI Research Note 87-09

20. Abstract (continued)

a) validation of the NTC research database system
b) access to data not available currently
c) the possibility of combining data sources to clean data.

In order to access data on the log tapes, it was necessary to develop a VAX program to read them. This has been done, and documentation of the tape-reading program **GDETAP** is the subject of this report.

PROGRAM GDETAP DOCUMENTATION

CONTENTS

Page

1.0 Background 1

2.0 Introduction 1

3.0 Program Description 2

3.1 Input 2

3.2 Processing 2

3.3 Output 3

Appendix A - User's Guide

Introduction A-1

1.0 Compiling and Linking A-1

2.0 Tape A-1

3.0 Executing the Program A-2

4.0 Outputs A-2

4.1 Output Processing A-3

Appendix B - Sample Outputs B-1

Appendix C - GDETAP Source Listing C-1

LIST OF FIGURES

3-1 GDETAP Flow Chart: Main Program..... 4

3-2 GDETAP Flow Chart: Subroutine EVBRK..... 5

3-3 GDETAP Flow Chart: Subroutine PPROC..... 6

3-3 GDETAP Flow Chart: Subroutine EVPROC..... 7

4-1 Event File Format.....A-4

B-1 Example of a Combined Event/Commo Report.....B-1

B-2 Example of a Commo Report.....B-2

B-3 Example of an Event Report.....B-3

B-4 Example of a Personnel Report.....B-4

Draft Report - Program to read GDE data tapes from the NTC

1.0 Background

There are two separate sources of digital data from the NTC. For over a year, ARI-POM has received BACKUP tapes created from NTC files created on the CIS VAX 11-780 computers. Files restored from those tapes can be processed by the TRANSLATOR and LOADER programs to create INGRES tables available for research/ analysis use. The restored files also directly support the use of the NTC workstations in historian mode.

The second source of digital NTC data is the GDE log tape from the SEL computer that processes RDMS data and calculates position/ location data. This tape contains all data passed to the CIS, including unfiltered event data. Comparing the contents of this log tape with the files created from the BACKUP tape will be useful for several reasons, including:

1) Validation of the NTC research data base system,
2) Access to data not available currently, and
3) Possibility of combining data sources to "clean" data.

To access data on the log tapes, it was necessary to develop a VAX program to read them. This has been done; documentation of the tape-reading program GDETAP is the subject of this report.

2.0 Introduction

GDETAP is a VAX FORTRAN program which can read the GDE log tape produced at NTC and produce reports/files of the data contained thereon. Program GDETAP consists of the main program, GDETAP, and ten subroutines. The calling structure is represented below:

```
GDETAP - Main Program
     EVBRK   - Process one block of messages
          EVPROC - Process event and commo messages
               REPACK - Reverses order of bytes in the block
               CSCON  - Get Date & Time
                    JDAY   - Returns Mo, Day from Julian Date
                    MSCON  - Returns time of day
               ECODE  - Returns description, given event code
                    COMST  - Compute commo time deltas
          PPROC  - Process player list messages
               REPACK - Reverse order of bytes in the block
               BREP   - Process player information
```

3.0 Program Description

The following the details of GDETAP input/output and operation. There are three sections, Input, Processing, and Output.

3.1 Input

GDETAP requires two kinds of input, the magnetic tape from NTC, and user inputs.

The mag tapes must be mounted as device TAPE$GDE. The DEC Command Language (DCL) command to do this is presented as part of Appendix A, the Users' Guide.

There are two user inputs. The first lets the user specify the output report(s) that he wants. At the present time, there are three kinds of reports available. Samples are included as Appendix B. The three options are Event listings, Communication listings, and Player lists. The option is requested by entering the first letter(s) of all reports requested. For example, to get an event listing the user enters E, while to get all reports he enters ECP. The second user input specifies which tape block processing is to begin. This allows the user to skip preliminary data and get to what he wants. The program will read up to the block indicated, but will not process the data.

3.2 Processing

Program GDETAP operation is graphically represented in the flowcharts presented as figures 3-1 - 3-4 These flowcharts document the high-level logic for the main program, GDETAP, and the three principal subroutines, EVBRK, EVPROC, and PPROC. A short description of each subroutine is presented below, and a complete listing of program GDETAP source code with comments is included as Appendix C.

The GDETAP program reads a mag tape generated on the CIS SEL at the National Training Center (NTC) and generates one or two (user option) reports. Report one is an event listing. The user may chose all events, commo events, or just non-commo events. Report two creates a player list, including logical player number, B-Unit number, player type, and a validity code.

EVBRK isolates messages in the physical block read from the tape, then CALLs appropriate processing subroutine.

EVPROC processes all the event type messages encountered in a GDE tape block.

PPROC processes all the player type messages encountered in the GDE tape block.

CSCON gets the time of day and date from an input containing the number of centiseconds since the beginning of the year.

JDAY returns a nine character ASCII string containing the date, given an integer that contains the Julian date+1 (number of days since last 12/31).

MSCON returns an eight character ASCII string containing the time of day, given an integer that contains the number of milliseconds since midnight.

COMST checks COMMO messages read from the GDE tape

For a KEY DOWN message:

The time of the message is saved in the KEY array.

For a KEY UP message:

If there is a time saved for this player and net, calculate the time from KEY DOWN to KEY UP.

If there is no time saved for this player and net, save error message `No Key Down Message.'

ECODE decodes the event messages and returns two descriptive ASCII strings to be printed.

3.3 Output

GDETAP produces reports in two different formats. The former is used for the event and commo message reports, and the second is used for the player list reports. Appendix B contains a sample of each type of report.

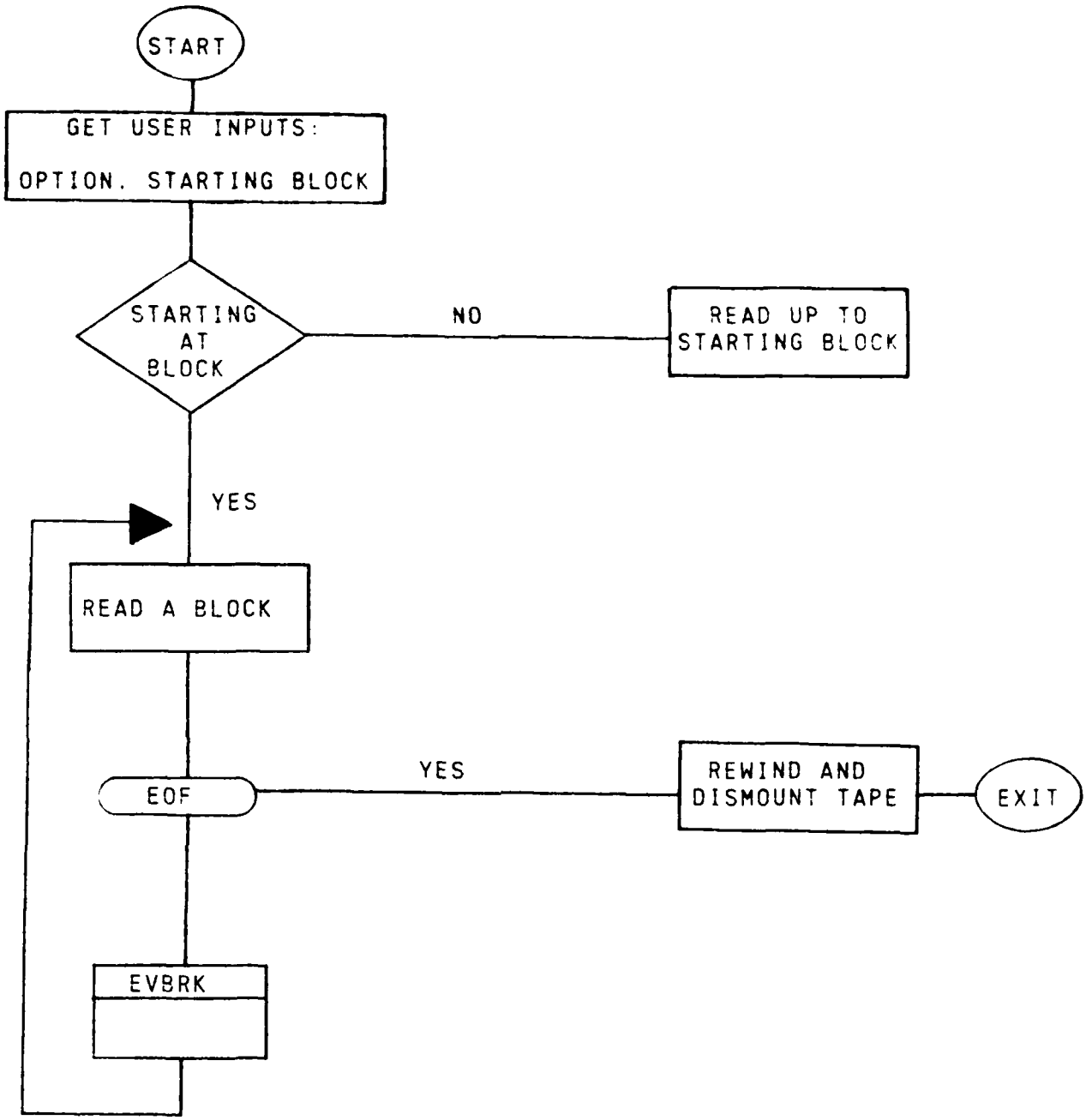

Figure 3-1
GDETAP Flow Chart: Main Program

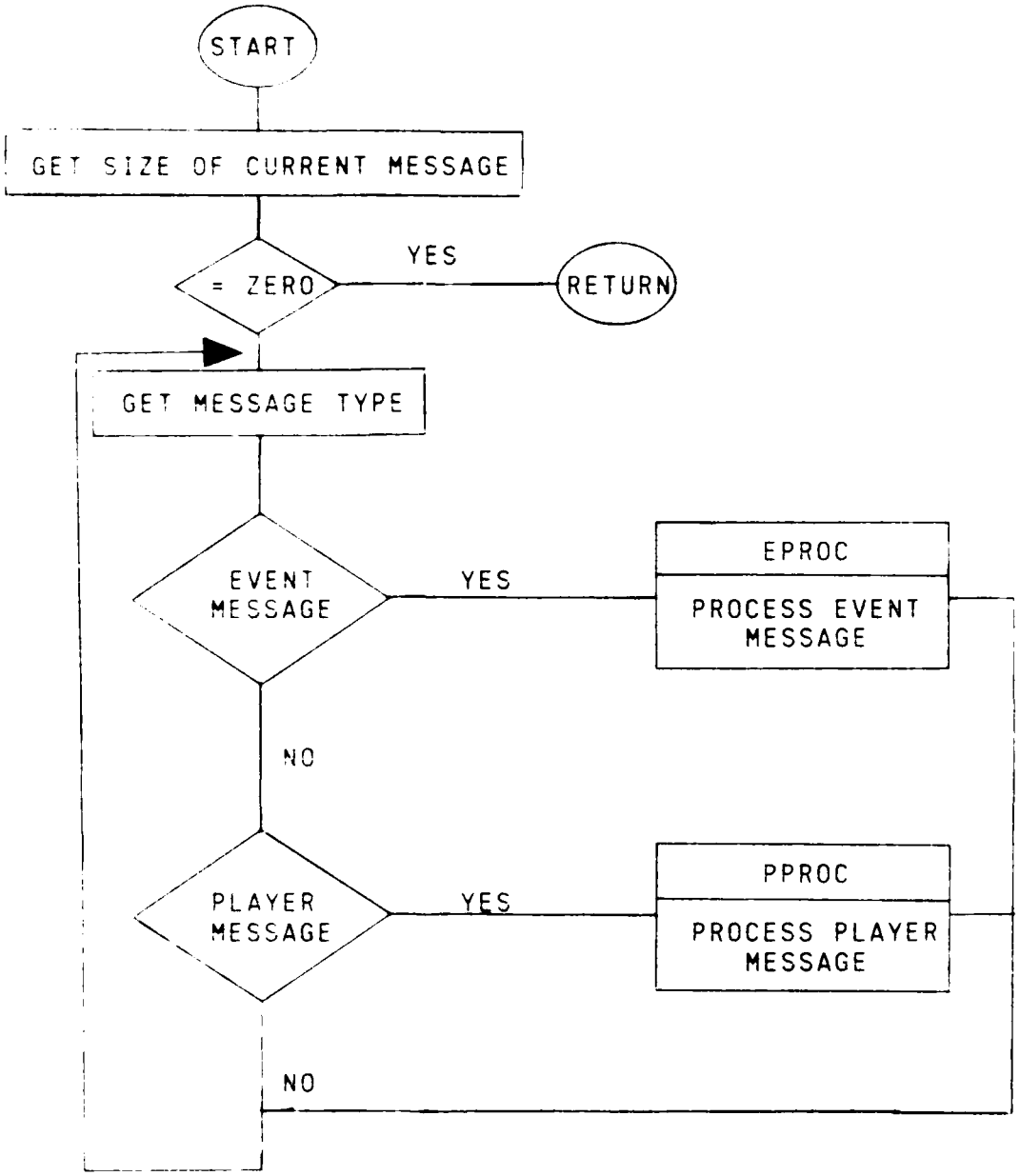

Figure 3-2
GDETAP Flow Chart: Subroutine EVBRK

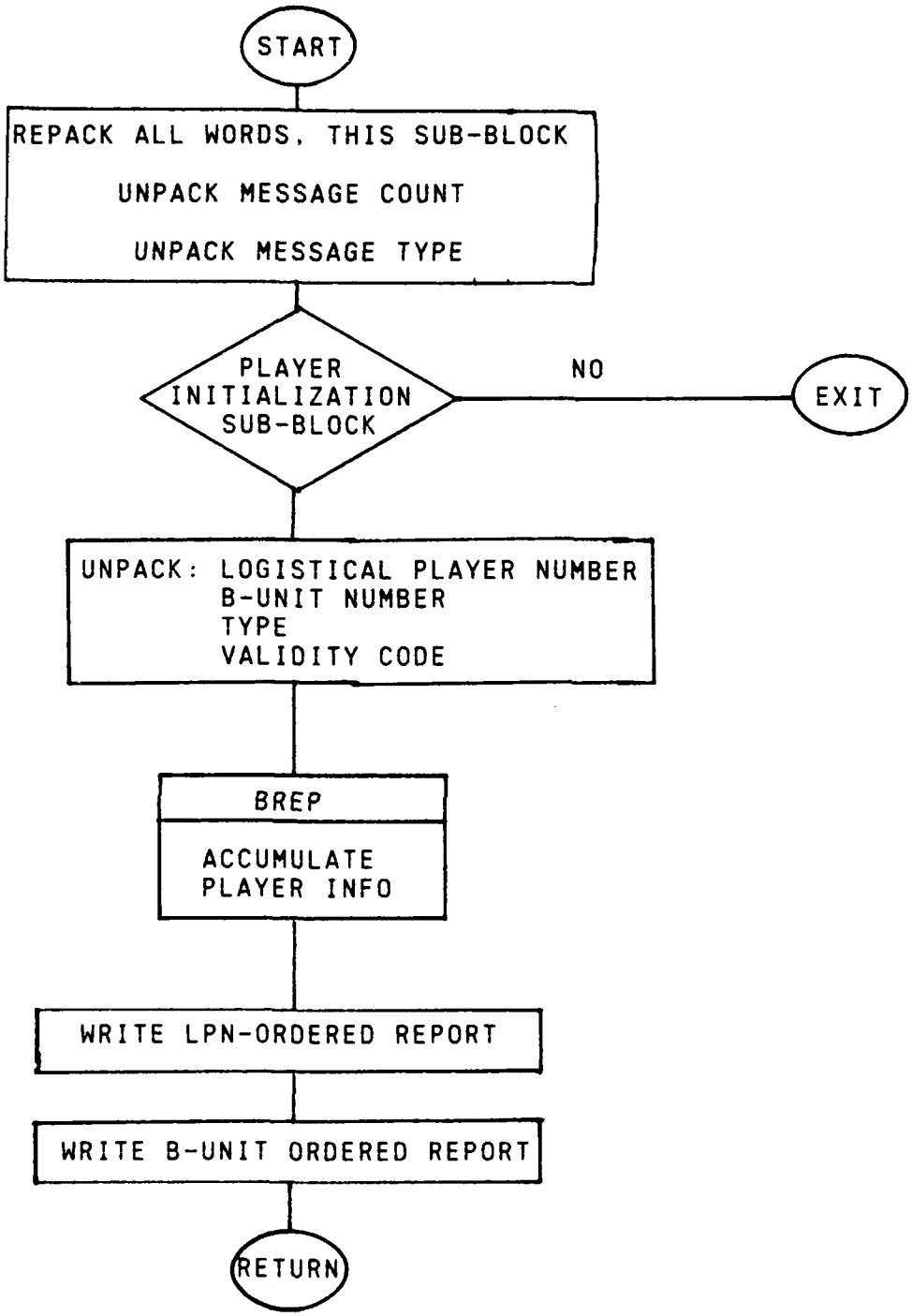

Figure 3-3
GDETAP Flow Chart: Subroutine PPROC

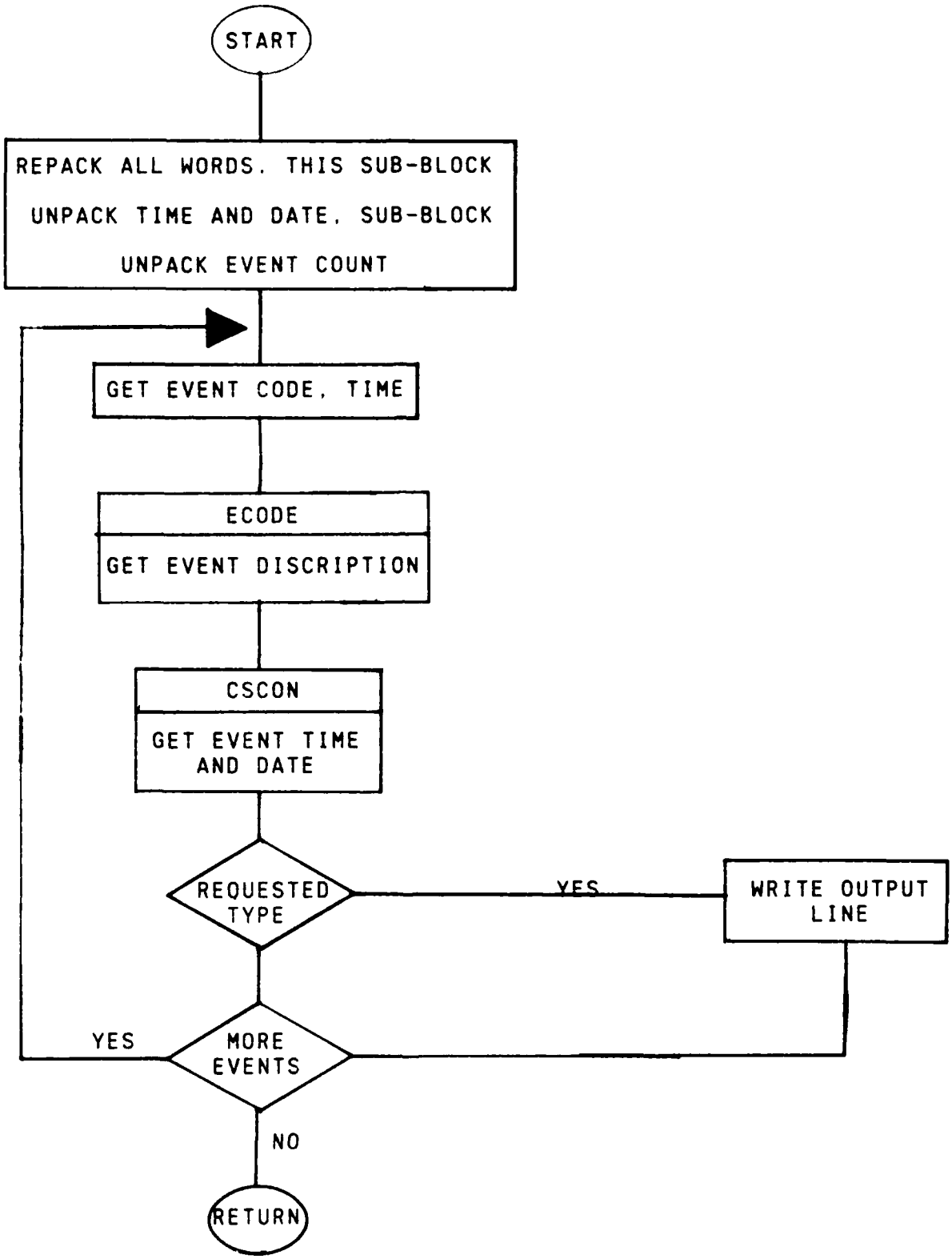

Figure 3-4
GDETAP Flow Chart: Subroutine EVPROC

Appendix A - User's Guide

Appendix A User's Guide

Introduction.

This appendix was written to document the information necessary to use program GDETAP. It includes instructions for compiling and linking the program, mounting the input tape, executing the program, and processing the output.

1.0 Compiling and Linking.

Program GDETAP was written in VAX FORTRAN. It consists of the main program, GDETAP.FOR, and the following subroutines/files:

SUBROUTINE NAME	FILE NAME
COMST	COMST.FOR
CSCON	CSCON.FOR
ECODE	ECODE.FOR
EVBRK	EVBRK.FOR
EVPROC	EVPROC.FOR
JDAY	JDAY.FOR
MSCON	MSCON.FOR
PPROC	PPROC.FOR
REPACK	REPACK.FOR
BREP	PPROC.FOR

In addition to the files containing the FORTRAN source code, two files are INCLUDE'd in the source and must be present for the compile step. They are:

BLOCK.INC
ALPHA.INC

The command to compile GDTAP is:

```
FOR/LIS/CON=99 GDETAP,EVBRK,EVPROC,ECODE,COMST,PPROC,CSCON,MSCON,
JDAY,REPACK
```

And to link:

```
LINK GDETAP,EVBRK,EVPROC,ECODE,COMST,PPROC,CSCON,MSCON,JDAY,REPACK
```

2.0 Tape

The tape must by mounted as device TAPE$GDE. The DCL command is:

```
MOUNT/FOREIGN/BLOCKSIZE=2200 MFA0:GDE
```

3.0 Executing the Program

The program is executed by entering:

RUN GDETAP

The program will prompt with:

Please enter the option requested for dumping the raw Tape Data you may enter from one to three letters, depending on the report(s) wanted
For an Event listing (without commo) enter E
For a listing of commo events, enter C
For a list of player B-Units vs LPN, enter P
For a combination, simply enter the letters for all options requested

What are your choices?

The user must then enter, in any order, any combination of the letters E,P, and C. The program will parrot the selected options, then ask for the starting block number. If no number is entered, the program will start at the beginning of the tape.

4.0 Outputs

GDETAP supports two output formats. One contains MILES and communications events, and the other contains the player listings. The event listing format is described in figure 4-1.

There are two kinds of player listings, the initial load list and the changes. The load list contains the logical player number (LPN), the B-Unit number (in both octal and hex), the player type (undocumented), and a validity code (0/1). The load list is printed twice, first ordered by LPN, and then sorted by B-Unit number. The player change record is:

COLUMN	DESCRIPTION
1-15	PL CH FOR PN
16-18	LPN
19-20	BLANK
21-24	PREVIOUS B-UNIT NUMBER (HEX)
25-26	BLANK
27-30	NEW B-UNIT NUMBER (HEX)
31-35	BLANK
36-39	PREVIOUS PLAYER TYPE
40-44	BLANK
45-48	NEW PLAYER TYPE
49-53	BLANK
54-57	PREVIOUS PLAYER VALIDITY
58-62	BLANK
63-66	NEW PLAYER VALIDITY

4.1 Output Processing

An auxiliary program, EVCUT has been written to allow the user to re-process the event listing.

EVCUT automatically deletes messages that are illegal (i.e. not supported by the RDMS). These messages occur frequently as a result of garbled RDMS transmissions. EVCUT also eliminates the MILES SELF TEST message.

The second capability of EVCUT is to eliminate all messages from a given player. For instance, if logical player number (LPN) 67 was reporting a high percentage of illegal messages, the probability is that even those which are apparently valid, are not. As another example, occasionally a B-unit will "stick" and transmit the same message constantly, overwhelming the system. EVCUT can also eliminate this player. The basic operation of EVCUT is as shown below.

```
                                                    NEW FILE

ORIGINAL                    EVCUT
 FILE
                                                    EXCEPTIONS FILE
```

All messages that are deleted are written to the exceptions file, while the others are written to the new file. The program prompts the user for the names of all these files.

The VAX environment contains a number of tools that can also be used to manipulate the files, including editors and the SORT utility.

COLUMN	DESCRIPTION
1- 2	BLANK
3- 8	DATE (E.G. 18 APR)
9	BLANK
10-17	TIME EVENT WAS RECIEVED BY RDMS (HH:MM:SS)
18-21	BLANK
22-29	TIME EVENT OCCURRED (HH:MM:SS)
30-32	BLANK
33-35	LOGISTICAL PLAYER NUMBER
36-38	BLANK
39	MILES FLAG (1 FOR MILES EVENT)
40-42	BLANK
43	MILES EVENT TYPE 0 Command Acknowledge 1 Weapon Fire 2 Hit 3 Kill 4 Near Miss 5 Accidental Kill by Weapon Key 6 Rounds Remaining/Firer ID 7 Self Test
44-45	BLANK
46-47	MILES EVENT INFO

EVENT TYPE	OCTAL EVENT INFORMATION
0	10 octal for KILL command acknowledge and 36 octal for RESURRECT.
1	Weapon code (see Table XIV)
2	
3	Weapon code (see Table XIV)
4	
5	Not used
6	For rounds remaining: When bit 4=1, bits 0-3 contain the most significant digit in rounds remaining count. When bit 4=0, bits 0-3 contain the least significant digit. The rounds remaining event shall follow the weapon fire event. For firer identification: When bits 4,3 = 0,0, bits 0-2 contains the least significant 3 bits (bits 0-2) of the nine bit firer identification. When bits 4,3 = 0,1, bits 0-2 contain the middle three bits (bits 3-5). When bits 4,3 = 1,0, bits 0-2 contain the most significant three bits (bits 6-8). Catenate bits 0-2, bits 3-5, and bits 6-8 to form the 9 bit firer identification (0-330 unique codes).
7	All 1's for self test passed and all 0's for self test failed.

FIGURE 4-1
Event File Format

48-49 BLANK

50-51 DISCRETE EVENT CODE

VALUE	DEFINITION
0	Not used
1	Man Worn Laser Detection System Kill
2	Man Worn Laser Detection System Near Miss
3	TOW, Dragon, Viper, Vulcan, and Chaparral-MILES Weapon fire
4	TOW-MILES Kill
5	TOW-MILES Near Miss
6	Tactical Radio No. 1 key up
7	Tactical Radio No. 1 key down
8	Tactical Radio No. 2 key up
9	Tactical Radio No. 2 key down
10-255	Not used

52-53 BLANK

54-123 DESCRIPTION OF EVENT

for MILES events, includes event type and info for COMMO, contains key down/up and radio net, plus duration of message for key up

FIGURE 4-1
Event File Format

Appendix B - Sample Outputs

Event Received	Occurred	LPN	MFL	MTY	MEV	EUA	Description	
17 Apr 23 37 02	23 36 04	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 02	23:36:04	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 02	23:36:04	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 03	23:36:04	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 04	23:36:05	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 04	23:36:05	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 12	23:37:07	65	0	0	7	7	Tac Radio 1 - Key Down	
17 Apr 23 37 12	23:36:05	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 12	23:36:05	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 13	23:37:10	85	0	0	9	9	Tac Radio 2 - Key Down	
17 Apr 23 37 13	23:36:28	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 13	23:37:10	85	0	0	7	7	Tac Radio 1 - Key Down	
17 Apr 23 37 13	23:36:28	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 14	23:36.28	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 14	23:36:29	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 14	23:36:29	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 15	23:36:29	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 16	23:37:12	65	0	0	6	6	Tac Radio 1 - Key Up	Length = 5 sec
17 Apr 23 37 16	23:36:30	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 16	23:36:30	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23.37 17	23:36 30	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 19	23:36:30	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 19	23:36:30	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 20	23:36:31	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 20	23:37:10	167	0	0	7	7	Tac Radio 1 - Key Down	
17 Apr 23 37.21	23:37:16	167	0	0	6	6	Tac Radio 1 - Key Up	Length = 6 sec
17 Apr 23 37 22	23 36 31	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 22	23:36:31	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 23	23:36:31	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37.23	23:36:31	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 24	23 36 31	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 24	23:36:31	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 25	23 36 31	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 25	23:36:31	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 25	23 36.31	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 26	23 36 31	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 26	23 36 36	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 27	23:36 36	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 27	23 36 36	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 28	23 36 36	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 29	23 36 36	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 29	23 36 36	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 30	23 36 36	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 30	23 36 36	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 31	23 37 24	167	0	0	7	7	Tac Radio 1 - Key Down	
17 Apr 23 37 32	23 36 36	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 32	23 36 37	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 33	23 36 37	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10
17 Apr 23 37 33	23 36 37	177	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
17 Apr 23 37 34	23 36 37	177	1	1	27	59	WEAPON FIRE	BLUFOR M16, M60 MG, Coax
17 Apr 23 37 34	23 36 37	177	1	6	17	81	ROUNDS REMAINING/ FIRER ID	Rounds remaining 10

FIGURE B-1
Example of a Combined Event/Commo Report

Event Received	Occurred	LPN	MFL	MTY	MEV	EUA	Description		
17 Apr 23 37 12	23:37:07	65	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 37 13	23:37:10	85	0	0	9	9	Tac Radio 2 - Key Down		
17 Apr 23 37 13	23:37:10	85	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 37 16	23:37:12	65	0	0	6	6	Tac Radio 1 - Key Up	Length =	5 sec
17 Apr 23 37 20	23:37:10	167	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 37 21	23:37:16	167	0	0	6	6	Tac Radio 1 - Key Up	Length =	6 sec
17 Apr 23 37 31	23:37:24	167	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 37 35	23:37:30	167	0	0	6	6	Tac Radio 1 - Key Up	Length =	6 sec
17 Apr 23 37 47	23:37:41	65	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 37 50	23:37:04	131	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 37 50	23:37:04	131	0	0	9	9	Tac Radio 2 - Key Down		
17 Apr 23 37 50	23:37:48	65	0	0	6	6	Tac Radio 1 - Key Up	Length =	7 sec
17 Apr 23 38 01	23:37:43	167	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 38 01	23:37:51	167	0	0	6	6	Tac Radio 1 - Key Up	Length =	8 sec
17 Apr 23 38 01	23:37:55	167	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 38 02	23:38:00	167	0	0	6	6	Tac Radio 1 - Key Up	Length =	5 sec
17 Apr 23 38 15	23:38:12	111	0	0	9	9	Tac Radio 2 - Key Down		
17 Apr 23 38 21	23:38:17	111	0	0	8	8	Tac Radio 2 - Key Up	Length =	5 sec
17 Apr 23 38 44	23:38:38	115	0	0	9	9	Tac Radio 2 - Key Down		
17 Apr 23 38 51	23:38:45	111	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 38 51	23:38:48	65	0	0	7	7	Tac Radio 1 - Key Down		
17 Apr 23 38 55	23:38:51	116	0	0	9	9	Tac Radio 2 - Key Down		
17 Apr 23 38 57	23:38:53	65	0	0	6	6	Tac Radio 1 - Key Up	Length =	5 sec
17 Apr 23 39 00	23:38:55	116	0	0	8	8	Tac Radio 2 - Key Up	Length =	4 sec

FIGURE B-2
Example of a Commo Report

Event Received	Occurred	LPN	MFL	MTY	M+V	EDA	Description	
18 Apr 03 00 22	03 00 18	19	1	7	31	127	SELF TEST	
18 Apr 03 01 26	03 01 20	12	1	7	31	127	SELF TEST	
18 Apr 03 01 27	03 01 25	12	1	7	31	127	SELF TEST	
18 Apr 03 01 47	03 01 45	85	1	4	7	7	NEAR MISS	BLUFOR - TOW M220/M902 OPFOR AT-6/HIND
18 Apr 03 02 06	03 02 03	66	1	7	31	127	SELF TEST	
18 Apr 03 02 07	03 02 03	85	1	4	3	3	NEAR MISS	OPFOR Sagger - BMP (M551 surrogate)
18 Apr 03 02 17	03 02 14	79	1	7	31	127	SELF TEST	
18 Apr 03 02 44	03 02 38	86	1	7	31	127	SELF TEST	
18 Apr 03 02 57	03 02 54	85	1	4	2	2	NEAR MISS	********** NOT IMPLEMENTED **********
18 Apr 03 03 06	03 03 03	29	1	7	31	127	SELF TEST	
18 Apr 03 03 20	03 03 16	54	1	4	30	30	NEAR MISS	Light Weapon Miss spare
18 Apr 03 03 51	03 03 47	13	1	7	31	127	SELF TEST	
18 Apr 03 05 10	03 05 02	54	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 05 11	03 05 03	54	1	4	29	29	NEAR MISS	Light Weapon Miss
18 Apr 03 05 11	03 05 07	66	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 05 25	03 05 22	82	1	4	7	7	NEAR MISS	BLUFOR - TOW M220/M902 - OPFOR AT-6/HIND
18 Apr 03 05 52	03 05 48	67	1	4	8	8	NEAR MISS	BLUFOR-DRAGON/OPFOR-Spandrel BRDM
18 Apr 03 05 56	03 05 53	88	1	7	31	127	SELF TEST	
18 Apr 03 06 37	03 06 33	67	1	4	8	8	NEAR MISS	BLUFOR-DRAGON/OPFOR-Spandrel BRDM
18 Apr 03 07 48	03 07 47	67	1	6	16	80	ROUNDS REMAINING/ FIRER ID	Rounds remaining 0
18 Apr 03 07 48	03 07 47	67	1	6	13	77	ROUNDS REMAINING/ FIRER ID	Rounds remaining 13
18 Apr 03 07 49	03 07 47	67	1	6	1	65	ROUNDS REMAINING/ FIRER ID	Rounds remaining 1
18 Apr 03 07 49	03 07 47	67	1	3	14	110	KILL	OPFOR 73MM BMP
18 Apr 03 08 58	03 08 56	67	1	4	2	2	NEAR MISS	********** NOT IMPLEMENTED **********
18 Apr 03 09 35	03 09 28	36	1	4	30	30	NEAR MISS	Light Weapon Miss spare
18 Apr 03 09 35	03 09 31	42	1	4	31	31	NEAR MISS	Heavy Weapon Miss spare
18 Apr 03 09 35	03 09 33	42	1	4	29	29	NEAR MISS	Light Weapon Miss
18 Apr 03 10 12	03 10 11	40	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 10 17	03 10 13	40	1	4	29	29	NEAR MISS	Light Weapon Miss
18 Apr 03 10 18	03 10 16	100	1	7	31	127	SELF TEST	
18 Apr 03 10 46	03 10 44	81	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 10 51	03 10 45	81	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 11 02	03 10 57	89	1	4	31	31	NEAR MISS	Heavy Weapon Miss spare
18 Apr 03 11 04	03 11 02	86	1	4	31	31	NEAR MISS	Heavy Weapon Miss spare
18 Apr 03 11 05	03 11 04	86	1	4	31	31	NEAR MISS	Heavy Weapon Miss spare
18 Apr 03 11 21	03 11 20	88	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 11 22	03 11 21	88	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 11 59	03 11 53	79	1	4	31	31	NEAR MISS	Heavy Weapon Miss spare
18 Apr 03 12 47	03 12 41	12	1	7	31	127	SELF TEST	
18 Apr 03 15 53	03 15 49	85	1	2	4	68	HIT	********** NOT IMPLEMENTED **********
18 Apr 03 16 21	03 16 17	59	1	4	30	30	NEAR MISS	Light Weapon Miss spare
18 Apr 03 16 26	03 16 21	59	1	4	30	30	NEAR MISS	Light Weapon Miss spare
18 Apr 03 16 27	03 16 24	59	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 16 36	03 16 34	59	1	4	23	23	NEAR MISS	OPFOR 30MM HIND-D/ BLUFOR 20MM/ Vulcan
18 Apr 03 17 03	03 17 00	85	1	4	8	8	NEAR MISS	BLUFOR-DRAGON/OPFOR-Spandrel BRDM
18 Apr 03 18 31	03 18 28	86	1	4	31	31	NEAR MISS	Heavy Weapon Miss spare
18 Apr 03 19 20	03 19 17	86	1	4	28	28	NEAR MISS	Heavy Weapon Miss
18 Apr 03 20 10	03 20 05	85	1	4	3	3	NEAR MISS	OPFOR Sagger - BMP (M551 surrogate)
18 Apr 03 21 42	03 21 38	87	1	7	31	127	SELF TEST	
18 Apr 03 23 03	03 22 59	67	1	4	2	2	NEAR MISS	********** NOT IMPLEMENTED **********
18 Apr 03 23 04	03 23 00	67	1	6	16	80	ROUNDS REMAINING/ FIRER ID	Rounds remaining 0
18 Apr 03 23 08	03 23 00	67	1	6	16	80	ROUNDS REMAINING/ FIRER ID	Rounds remaining 0
18 Apr 03 23 08	03 22 12	82	1	7	31	127	SELF TEST	
18 Apr 03 23 08	03 23 00	67	1	6	13	77	ROUNDS REMAINING/ FIRER ID	Rounds remaining 13
18 Apr 03 23 11	03 23 00	67	1	6	4	68	ROUNDS REMAINING/ FIRER ID	Rounds remaining 4
18 Apr 03 23 11	03 23 00	67	1	2	23	87	HIT	OPFOR 30MM HIND-D/ BLUFOR 20MM/ Vulcan
18 Apr 03 23 11	03 23 11	67	1	4	7	7	NEAR MISS	BLUFOR - TOW M220/M902 - OPFOR AT-6/HIND
18 Apr 03 23 23	03 23 22	67	1	4	3	3	NEAR MISS	OPFOR Sagger - BMP (M551 surrogate)
18 Apr 03 23 40	03 23 35	67	1	2	15	79	HIT	Viper

FIGURE B-3
Example of an Event Report

Player load list from CDC tape - List 1 Ordered by Logical Player Number (LPN) Page 1

T=Player Type, V=Valid (1=yes,0=no); B-UNIT is HEX/OCTAL

LPN	B-UNIT #	T	V	LPN	B-UNIT #	T	V	LPN	B-UNIT #	T	V	LPN	B-UNIT #	T	V	LPN	B-UNIT #	T	V
1	03[illegible]/1462	2	1	2	02EA/1352	2	1	3	021C/1034	2	1	4	02F9/1371	2	1	5	023[illegible]/1063	2	1
6	027[illegible]/1176	2	1	7	03BB/1673	2	1	8	0240/1100	2	1	9	02DB/1333	2	1	10	027D/1175	2	1
11	03[illegible]/1546	2	1	12	024E/1116	2	1	13	022A/1052	2	1	14	025C/1134	2	1	15	03C1/1701	2	1
16	02D[illegible]/1326	2	1	17	03C3/1703	2	1	18	02B4/1264	2	1	19	02EC/1354	2	1	20	0293/1223	2	1
21	037[illegible]/1572	2	1	22	02C4/1304	2	1	23	0251/1121	2	1	24	028C/1214	2	1	25	0204/1004	2	1
26	02C[illegible]/1316	2	1	27	0203/1003	2	1	28	0340/1500	2	1	29	0335/1465	1	1	30	03EF/1757	1	1
31	03A[illegible]/1650	1	1	32	02CC/1314	2	1	33	0338/1470	1	1	34	0397/1627	2	1	35	0385/1665	2	1
36	02[illegible]/1336	2	1	37	0274/1164	2	1	38	0219/1031	2	1	39	0388/1610	2	1	40	034B/1513	2	1
41	031[illegible]/1436	2	1	42	03E2/1742	2	1	43	02C7/1307	2	1	44	028D/1275	2	1	45	0257/1127	2	1
46	03[illegible]/1407	2	1	47	030E/1616	2	1	48	0287/1207	2	1	49	03EC/1754	2	1	50	02D0/1320	2	1
51	026[illegible]/1147	2	1	52	0361/1541	2	1	53	0207/1007	2	1	54	0206/1006	2	1	55	03FC/1774	2	1
56	03A[illegible]/1250	2	1	57	02CB/1313	2	1	58	034C/1514	2	1	59	03AB/1653	2	1	60	03D9/1731	2	1
61	037[illegible]/1570	2	1	62	02FD/1375	2	1	63	02E3/1343	2	1	64	025F/1137	2	1	65	0229/1051	2	1
66	03[illegible]/1636	2	1	67	02F2/1362	2	1	68	02FB/1370	2	1	69	020D/1015	2	1	70	0247/1107	2	1
71	030[illegible]/1556	2	1	72	02A6/1246	2	1	73	02CF/1317	2	1	74	030A/1412	2	1	75	02D4/1324	2	1
76	03D[illegible]/1357	2	1	77	03EE/1756	2	1	78	039F/1637	2	1	79	02FA/1372	2	1	80	032B/1453	2	1
81	026[illegible]/1141	2	1	82	02E4/1344	2	1	83	03D1/1721	2	1	84	03AC/1654	2	1	85	03C2/1702	2	1
86	02C6/1306	2	1	87	01F3/0763	2	1	88	03AF/1657	2	1	89	028F/1217	2	1	90	0390/1620	2	1
91	038[illegible]/1600	2	1	92	0216/1026	2	1	93	031D/1435	1	1	94	03B0/1660	2	1	95	03C7/1707	2	1
96	02EB/1353	2	1	97	02C1/1301	2	1	98	0281/1201	2	1	99	0381/1601	2	1	100	0384/1664	2	1
101	03F[illegible]/1767	2	1	102	029D/1235	2	1	103	0236/1066	2	1	104	0252/1122	2	1	105	037D/1575	2	1
106	0324/1444	2	1	107	036C/1554	2	1	108	0280/1200	2	1	109	03ED/1755	2	1	110	0317/1427	2	1
111	02F[illegible]/1374	2	1	112	026A/1152	2	1	113	03F8/1770	2	1	114	034A/1512	2	1	115	02E9/1351	2	1
116	02A[illegible]/1257	2	1	117	03F1/1761	2	1	118	032A/1452	2	1	119	02BB/1273	2	1	120	02F0/1360	2	1
121	031[illegible]/1425	2	1	122	028E/1216	2	1	123	03E1/1741	2	1	124	0384/1604	2	1	125	02FE/1376	2	1
126	027[illegible]/1170	2	1	127	034D/1515	2	1	128	02ED/1355	1	1	129	03E5/1745	2	1	130	02F6/1366	1	1
131	034[illegible]/1516	2	1	132	0347/1507	2	1	133	028D/1215	2	1	134	02B0/1260	2	1	135	0353/1523	2	1
136	03BA/1672	2	1	137	02B3/1263	2	1	138	029C/1234	2	1	139	0304/1404	2	1	140	0239/1071	2	1
141	03[illegible]0/1520	2	1	142	0318/1430	2	1	143	03D5/1725	2	1	144	0215/1025	2	1	145	0260/1140	2	1
146	03D[illegible]/1727	2	1	147	026C/1154	2	1	148	0299/1231	2	1	149	022B/1053	2	1	150	0374/1564	2	1
151	0316/1426	2	1	152	0269/1151	1	1	153	0282/1202	2	1	154	03A4/1644	1	1	155	02A2/1242	1	1
156	03[illegible]/1474	1	1	157	03CF/1717	2	1	158	03E8/1750	2	1	159	03FE/1776	2	1	160	03DD/1735	2	1
161	03A1/1641	2	1	162	0321/1441	2	1	163	0246/1106	2	1	164	03FD/1775	2	1	165	036A/1552	2	1
166	02DF/1337	2	1	167	0302/1402	2	1	168	0294/1224	1	1	169	0352/1522	2	1	170	0337/1467	2	1
171	03A3/1643	2	1	172	0329/1451	2	1	173	03B1/1661	2	1	174	02A1/1241	2	1	175	03DB/1733	2	1
176	02F7/1367	2	1	177	0360/1540	2	1	178	0217/1027	2	1	179	03E0/1740	2	1	180	02FB/1373	2	1
181	0306/1406	2	1	182	0376/1566	2	1	183	03D2/1722	2	1	184	0386/1606	2	1	185	0249/1111	2	1
186	02D1/1321	2	1	187	024A/1112	2	1	188	02AA/1252	2	1	189	01F1/0761	2	1	190	02B8/1270	2	1
191	0254/1124	2	1	192	01EF/0757	2	1	193	000E/0016	6	1	194	0200/1000	2	1	195	0265/1145	2	1
196	02AB/1253	2	1	197	03A7/1647	2	1	198	023D/1075	2	1	199	02EE/1356	2	1	200	0351/1521	2	1
201	02F[illegible]/1363	2	1	202	03D0/1720	2	1	203	0313/1423	2	1	204	0369/1551	2	1	205	0279/1171	2	1
206	032B/1450	2	1	207	00CC/0014	6	1	208	00E3/0343	6	1	209	013E/0476	6	1	210	0359/1531	2	1
211	00DA/0332	6	1	212	0105/0405	6	1	213	00E7/0347	6	1	214	0210/1020	2	1	215	0058/0130	6	1
216	03C9/1711	2	1	217	0394/1624	2	1	218	025B/1133	2	1	219	0382/1602	2	1	220	0373/1563	2	1
221	03C5/1705	2	1	222	0295/1225	2	1	223	032C/1454	2	1	224	03FF/1777	2	1	225	035F/1537	2	1
226	027D/1160	2	1	227	02A3/1243	2	1	228	0385/1605	2	1	229	03B2/1662	2	1	230	0364/1544	2	1
231	03CB/1713	2	1	232	03BF/1677	2	1	233	0264/1144	2	1	234	0245/1105	2	1	235	02DC/1334	2	1
236	026D/1143	2	1	237	03B7/1607	2	1	238	0235/1065	2	1	239	021E/1036	2	1	240	02AD/1255	2	1
241	01[illegible]0/0770	2	1	242	030D/1415	2	1	243	0305/1405	1	1	244	0226/1046	2	1	245	02E1/1341	2	1
247	0317/1431	2	1	247	03DF/1737	2	1	248	0343/1503	2	1	249	02B9/1271	2	1	250	0354/1524	2	1

FIGURE B-4
Example of a Personnel Report

Appendix C - GDETAP Source Listing

```
C-----------------------------------------------------------------
C
C       NAME - GDETAP
C
C       AUTHOR - JACK BRISCOE
C
C       DATE - 15 APRIL, 1985
C
C       DESCRIPTION - The GDETAP program reads a mag tape generated
C               on the CIS SEL at the National Training Center (NTC)
C               and generates one or two (user option) reports.
C               Report one is an event listing.  The user may choose
C               all events, commo events, or just non-commo events.
C               Report two creates a player list, including logical
C               player number, B-unit number, player type, and a
C               validity code.
C
C       INCLUDE FILES - BLOCK.INC - The input buffer common, containing
C               the input buffer, the current record count, and the
C               options selected by the user.
C
C       CALLING SEQUENCE DESCRIPTION - None - Main Program
C
C       INPUTS - Mag Tape - Logical name TAPE$GDE
C               USER INPUTS - Option, starting block number
C
C       OUTPUTS - Reports discussed above.
C
C-----------------------------------------------------------------

        IMPLICIT INTEGER*4 (A-Z)

        CHARACTER*3 U

C      *BLOCK.INC contains the read buffer & record count
        INCLUDE 'BLOCK.INC/LIST'

        DATA RSTART/1/

C      *Set all option flags FALSE until user chooses
        EVENT=.FALSE.
        COMMO=.FALSE.
        PLAYR=.FALSE.

C      *Apprise user of his options
        PRINT 1010
1010    FORMAT(' Please enter the option requested for dumping'
     1,         ' the raw Tape Data '
     2,//,'   You may enter from one to three letters, depending'
     3,    ' upon the report(s) wanted'
     4,//,'   For an Event listing (without commo) enter E'
     5,/, '   For a listing of commo events, enter       C'
     6,/, '   For a list of player B-Units vs LPN, enter P'
     7,//,'   For a combination, simply enter the letters for all'
     8,    ' options requested'///,' What are your choices ?  '$)
```

```
C       *Get his answer - Look for up to three characters
         ACCEPT 1011,O
1011     FORMAT(A3)
C       *check each character for every possibillity
         DO 10 I=1,3
                 IF((O(I:I).EQ.'E').OR.(O(I:I).EQ.'e')) EVENT=.TRUE
                 IF((O(I:I).EQ.'C').OR.(O(I:I).EQ.'c')) COMMO=.TRUE
                 IF((O(I:I).EQ.'P').OR.(O(I:I).EQ.'p')) PLAYR=.TRUE
10       CONTINUE

C       *Tell User what he's chosen
         PRINT 1012,EVENT,COMMO,PLAYR
1012     FORMAT(///,' Options selected were :    Event ',L1,'  Commo ',L1
     1,                                '  Player list ',L1)

C       *Write the title on the event/commo listing, if necessary
         IF(EVENT.OR.COMMO) WRITE(8,1005)
1005     FORMAT('  Event Received      Occurred    LPN MFL MTY MEV EBA    '
     1,       'D e s c r i p t i o n '
     2,/,     '  ---------------     --------    --- --- --- --- ---    '
     3,     '-----------------------------------------------------')

C       *See if the user wants to start at the beginning, or wherever
         PRINT 1000
1000     FORMAT(' Which record do you want to start with ? '$)
         ACCEPT 1001,RSTART
1001     FORMAT(I)

C       *Open the tape file as FORTRAN UNIT 2.  The tape must have been
C       *mounted as device TAPE$GDE.  The command is
C       *        MOUNT/FOREIGN/BLOCKSIZE=2200 MFAO  GDE

         OPEN(2,FILE='TAPE$GDE'
     1,          RECORDTYPE='VARIABLE'
     2,          STATUS='OLD'
     3,          ORGANIZATION='SEQUENTIAL'
     4,          FORM='UNFORMATTED'
     5,          ACCESS='SEQUENTIAL'
     6,          IOSTAT=IOWORD)

C       *Initialize record count
         BLIDNO=RSTART-1
C       *If RSTART was set, read up to that many records
         IF(BLIDNO.GT.0) THEN
                 DO 25 I=1,BLIDNO
                         READ(2,IOSTAT=IOWORD,END=900) BLOCK
25               CONTINUE
         ENDIF
C       *Do until end-of-file
1        BLIDNO=BLIDNO+1
C       *Read a record   Each record may be up to 2047 words   Reading
```

```
C       *this way will cause IOSTAT to be set to 67, but the tape will
C       *be properly read.
C       *At end-of-file, go to statement 900 to wrap things up

        READ(2,IOSTAT=IOWORD,END=900) BLOCK
C          *Call appropriate subroutine(s) for selected option(s)
           CALL EVBRK
        GOTO 1
900     CONTINUE
        REWIND 2
        CLOSE(2)
        STOP 'Okay'
        END
```

```
      SUBROUTINE EVBRK
C-----------------------------------------------------------------
C
C     NAME - EVBRK
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - Isolates messages in physical block read from
C                   tape, then CALLs appropriate processing subroutine.
C
C     INCLUDE FILES
C       BLOCK.INC - BLO COMMON: Read buffer, current block no., flags
C
C     CALLING SEQUENCE DESCRIPTION - None
C
C     INPUTS - This routine is CALLed each time a new buffer is
C       read into BLOCK.
C
C     OUTPUTS - None
C
C     SUBROUTINES CALLED
C       EVPROC - Process events
C       PPROC  - Process player info
C       MVBITS - Unpack a word (System routine)
C
C-----------------------------------------------------------------
      IMPLICIT INTEGER*4 (A-Z)

      INCLUDE 'BLOCK.INC/LIST'

      DATA ETYPE/'0000001D'X/
      DATA PTYPE/'0000001F'X/

      NW=1
C    *ISOLATE SIZE OF CURRENT MESSAGE IN BLOCK
      CALL MVBITS(BLOCK(NW),24,8,SIZE,0)
C    *WHILE NOT AT END-OF-BLOCK
      DO WHILE(SIZE.NE.0)
C       *ISOLATE MESSAGE TYPE
        CALL MVBITS(BLOCK(NW+1),24,8,TYPE,0)
C       *IF(THIS IS AN EVENT MESSAGE)
        IF(TYPE .EQ. ETYPE) THEN
C          *GO PROCESS MESSAGE FOR MILES OR COMMO
           IF(COMMO.OR.EVENT) CALL EVPROC(NW,SIZE)
        ELSE
           IF(TYPE.EQ.PTYPE) THEN
C             *PLAYER TYPE MESSAGE
              IF(PLAYR) CALL PPROC(NW,SIZE)
           ENDIF
        ENDIF
```

```
C          *INCREMENT LOCATION IN BUFFER
            NW=NW+SIZE
            CALL MVBITS(BLOCK(NW),24,8,SIZE,0)
        END DO
        RETURN
        END
```

```
      SUBROUTINE EVPROC(SUB,LEN)
C-----------------------------------------------------------------------
C
C     NAME - EVPROC
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - EVPROC processes all the event type messages
C       encountered in a GDE tape block
C
C     INCLUDE FILES
C       BLOCK.INC - BLO COMMON. Read buffer, current block no., flags
C
C     CALLING SEQUENCE DESCRIPTION
C       SUB     - Subscript of first word of message in array BLOCK
C       LEN     - Length of packed message(s)
C
C     INPUTS - Array BLOCK, containing all messages
C
C     OUTPUTS - Print of each message
C
C     SUBROUTINES CALLED
C      CSCON  - Converts centiseconds since midnight to date, time
C      ECODE  - Returns ASCII description of events
C      REPACK - Swaps bytes in each word
C
C-----------------------------------------------------------------------
      IMPLICIT INTEGER*4 (A-Z)

      CHARACTER*8 BTIME,ETIME
      CHARACTER*9 BDAY,EDAY
      CHARACTER*20 RETSTR
      CHARACTER*70 DESC
      LOGICAL GO

      INCLUDE 'BLOCK.INC/LIST'

C    *MOVE ALL BYTES
      DO 100 I=1,LEN
              CALL REPACK(BLOCK(SUB+I-1))
100   CONTINUE

C    *GET TIME & DATE FOR BIG EVENT BLOCK
      CALL CSCON(BLOCK(SUB+3),BDAY,BTIME)

C    *SET NEW SUBSCRIPT LOCATION - FOR FIRST SUB-BLOCK
      NEWSUB=SUB+5
C    *GET MESSAGE COUNT FOR FIRST SUB-BLOCK
      CALL MVBITS(BLOCK(NEWSUB),8,8,SLEN,0)
```

```
C       *GET NUMBER OF MESSAGES IN SUB-BLOCK
         NMES=(SLEN-1)/2
         PRINT 1002,BLIDNO,SUB,LEN,SLEN,NMES,BDAY,BTIME
1002     FORMAT(' BLIDNO=',I12,' SUB=',I6,'  LEN=',I4,'  SLEN=',I4
     1,          '  NMES=',I4,' ',A9,A8)

C       *DO(FOR EACH SUB-BLOCK MESSAGE)
         DO 200 I=1,NMES
                 DESC = ' '
                 EVCODE=BLOCK(NEWSUB+1)
                 EVTIME=BLOCK(NEWSUB+2)
                 CALL ECODE(EVCODE,RETSTR,DESC,GO,EVTIME)
                 CALL CSCON(EVTIME,EDAY ETIME)
                 IF(GO) WRITE(8,1001) BDAY,BTIME,ETIME,RETSTR,DESC
1001             FORMAT(A9,A8,4X,A8,A22,2X,A70)
C               *RESET SUBSCRIPT
                 NEWSUB=NEWSUB+2
200      CONTINUE
         RETURN
         END
```

```
      SUBROUTINE ECODE(CODENO,RETSTR,DESCR,GO,CSTIME)
C-----------------------------------------------------------------------
C
C     NAME - ECODE
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - ECODE Decodes the event messages and returns
C                   two descriptive ASCII strings to be printed.
C
C     INCLUDE FILES -
C       BLOCK.INC - BLO COMMON: Read buffer, current block no., flags
C       ALPHA.INC - Event descriptions
C
C     CALLING SEQUENCE DESCRIPTION (I=Input, O=Output)
C     I  CODENO - Event code
C     O  RETSTR - ASCII string returned with parsed message
C     O  DESCR  - 70-character ASCII string with event description
C     O  GO     - Flag set to indicate if post-ECODE processing
C                 will be necessary
C     I  CSTIME - ime of event, in centiseconds since midnight
C
C     INPUTS - Calling sequence
C
C     OUTPUTS - Calling Sequence
C
C     SUBROUTINES CALLED
C       COMST  - Calculate message length
C-----------------------------------------------------------------------
      IMPLICIT INTEGER*4(A-Z)
      CHARACTER*40 WC(0:31),CODES(120)
      CHARACTER*30 MILES(0:7)
      CHARACTER*55 EBAD(0:9)
      CHARACTER*70 DESCR
      CHARACTER*20 RETSTR,RR,COMMSG
      LOGICAL GO

      INCLUDE 'BLOCK.INC/LIST'
      INCLUDE 'ALPHA.INC/LIST'

C     *SET TO WRITE UNLESS TURNED OFF LATER
      GO= FALSE.

C     *SPLIT EVENT INTO COMPONENT PARTS
      CALL MVBITS(CODENO,17,10,LPN,0)
      CALL MVBITS(CODENO, 8, 1,MFL,0)
      CALL MVBITS(CODENO, 5, 3,MTY,0)
      CALL MVBITS(CODENO, 0, 5,MEV,0)
      CALL MVBITS(CODENO, 0, 7,EBA,0)
```

```
C       *NOW PROCESS THE INDIVIDUAL PARTS
         WRITE(RETSTR,2000) LPN,MFL,MTY,MEV,EBA
2000     FORMAT(5I4)
C       *IF(THIS IS A REGULAR EBA - NOT MILES)
         IF(MFL.EQ.0) THEN
C                *CHECK FOR EBA VALUE VALIDITY - FORCE IT GOOD
                  IF(EBA.GT.9) EBA=0
C                *SET GO FLAG ACCORDING TO COMMO AND EVENT FLAGS
                  IF(EBA.GT.5) THEN
                           GO=COMMO
                           CALL COMST(EBA,LPN,CSTIME,COMMSG)
                           DESCR(1:30)=EBAD(EBA)(1:30)
                           DESCR(31:50)=COMMSG
                  ELSE
                           IF(EBA.GT.0) GO=EVENT
C                         *LOAD EBA DESCRIPTION
                           DESCR(1:55)=EBAD(EBA)
                  ENDIF
         ELSE
C                *IT'S GOTTA BE A MILES EVENT
                  DESCR(1:30)=MILES(MTY)
C                *SET GO IF EVENT FLAG SET
                  GO=EVENT
C                *IF(Weapon fire, Hit, Kill, or Near Miss) get weapon
                  IF((MTY.GE.1).AND.(MTY.LE.4)) THEN
                           DESCR(31:70)=WC(MEV)
                  ENDIF
C                *IF(ROUDS REMAINING MILES MESSAGE)
                  IF(MTY.EQ.6) THEN
C                         *IF(BIT 4 SET - MOST SIGNIFICANT DIGIT)
                           IF(MEV.GE.16) RRVAL=10*(MEV-16)
C                         *OTHERWISE, LEAST SIGNIFICANT DIGIT
                           IF(MEV.LT.16) RRVAL=MEV
                           WRITE(RR,2001) RRVAL
2001                       FORMAT(' Rounds remaining',I3)
                           DESCR(31:50)=RR
                  ENDIF
         ENDIF
         RETURN
         END
```

```
      SUBROUTINE COMST(EBA,LPN,CST,RETMSG)
C-----------------------------------------------------------------
C
C     NAME - COMST
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - COMST checks COMMO messages read from the GDE tape
C       For a KEY DOWN message:
C         The time of the message is saved in the KEY array.
C       For a KEY UP message:
C         If there is a time saved for this player and net,
C           Calculate the time from KEY DOWN to KEY UP
C         If there is NO time saved for this player and net,
C           Save error message ' No Key Down Message'
C
C     INCLUDE FILES - None
C
C     CALLING SEQUENCE DESCRIPTION
C             EBA - Message Code ( 6 : KEY DOWN, Net 1;
C                                  7 : KEY UP  , Net 1;
C                                  8 : KEY DOWN, Net 2; and
C                                  9 : KEY UP,   Net 2)
C             LPN - Logical Player Number
C             CST - Message time in centiseconds since midnight
C             RETMSG - Return message (either elapsed time or error)
C                      (20 characters max)
C
C     INPUTS - From calling sequence
C
C     OUTPUTS - To calling sequence
C
C-----------------------------------------------------------------
      IMPLICIT INTEGER*4 (A-Z)

      CHARACTER*20 RETMSG

      DIMENSION KEY(2,500)

C    *Blank out return message
      RETMSG=' '
C    *Get message between 1 and 4
      CMSG=EBA-5
C    *Set Net to either 1 or 2
      INDEX=1
      IF(CMSG.GT.2) INDEX=2
C    *IF KEY UP
      IF(MOD(CMSG,2).NE.0) THEN
C             *If there was a key down time saved
              IF(KEY(INDEX,LPN).NE.0) THEN
```

```
C                       *Compute & convert elapsed time to ASCII
                         DELTA=CST/100-KEY(INDEX,LPN)
                         WRITE(RETMSG,1000) DELTA
1000                     FORMAT(' Length =',i7,' sec')
C                       *Zero out KEY DOWN time
                         KEY(INDEX,LPN)=0
C                *Else no time saved for KEY DOWN
                 ELSE
C                       *Save an error message
                         RETMSG =' No Key Down Message'
                 ENDIF
C        *Else KEY DOWN message
         ELSE
C                *Save KEY DOWN time
                 KEY(INDEX,LPN)=CST/100
         ENDIF
         RETURN
         END
```

```
      SUBROUTINE PPROC(SUB,LEN)
C-----------------------------------------------------------------
C
C     NAME - PPROC
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - PPROC processes all the player type messages
C       encountered in a GDE tape block.
C
C     INCLUDE FILES
C       BLOCK.INC - BLO COMMON: Read buffer, current block no., flags
C
C     CALLING SEQUENCE DESCRIPTION
C       SUB     - Subscript of first word of message in array BLOCK
C       LEN     - Length of packed message(s)
C
C     INPUTS - Array BLOCK, containing all messages
C
C     OUTPUTS - Print of each message
C
C     SUBROUTINES CALLED
C      BREP   - Writes player change messages
C      REPACK - Swaps bytes in each word
C
C-----------------------------------------------------------------
      IMPLICIT INTEGER*4 (A-Z)

      LOGICAL FIRST
      COMMON/PLAY/LPN,PB,PT,PV,PBU(500),PTY(500),PVA(500),FIRST

      DIMENSION PBUT(500),LBUT(500)

      INCLUDE 'BLOCK.INC/LIST'

      DATA FIRST/ TRUE /

      BSUB=SUB
C    *MOVE ALL BYTES
      DO 100 I=1,LEN
              NSUB=BSUB+I-1
              CALL REPACK(BLOCK(NSUB))
100   CONTINUE

C    *SET NEW SUBSCRIPT LOCATION - FOR FIRST SUB-BLOCK
      NEWSUB=BSUB+3
C    *GET MESSAGE COUNT FOR FIRST SUB-BLOCK
      CALL MVBITS(BLOCK(NEWSUB),0,16,SLEN,0)
C    *GET MESSAGE TYPE
      CALL MVBITS(BLOCK(NEWSUB),24,8,TYPE,0)
```

```
      PRINT 10010,BLIDNO,LEN,BSUB,SLEN,TYPE
10010 FORMAT('  BLIDNO,LEN,BSUB,SLEN,TYPE  ',3I5,2I15)

      NEWSUB=NEWSUB+1
C    *IF(PLAYER INIT SUBBLOCK)
      IF(TYPE EQ 150) THEN

         DO 200 RNO=NEWSUB,NEWSUB+SLEN-2,3
              LPN=0
              PB=0
              PT=0
              PV=0

              CALL MVBITS(BLOCK(RNO  ), 0,16,LPN,0)
              IF(LPN GT.MAXLPN) MAXLPN=LPN
              CALL MVBITS(BLOCK(RNO  ),16,16,PB,0)
              CALL MVBITS(BLOCK(RNO+1),24, 8,PT,0)
              CALL MVBITS(BLOCK(RNO+1),16, 8,PV,0)
              CALL BREP
              IF((RNO+2).GT.(BSUB+LEN-1)) GOTO 200
              LPN=0
              PB=0
              PT=0
              PV=0
              CALL MVBITS(BLOCK(RNO+2),16,16,LPN,0)
              IF(LPN.GT.MAXLPN) MAXLPN=LPN
              CALL MVBITS(BLOCK(RNO+1), 0,16,PB,0)
              CALL MVBITS(BLOCK(RNO+2), 8, 8,PT,0)
              CALL MVBITS(BLOCK(RNO+2), 0, 8,PV,0)
              CALL BREP
200      CONTINUE
         IF(FIRST) THEN
            IF(MAXLPN.GE.500) THEN
              DO 300 I=1,MAXLPN,5
                      WRITE(9,2000) I,PBU(I),PBU(I),PTY(I),PVA(I)
     1,                       I+1,PBU(I+1),PBU(I+1),PTY(I+1),PVA(I+1)
     2,                       I+2,PBU(I+2),PBU(I+2),PTY(I+2),PVA(I+2)
     3,                       I+3,PBU(I+3),PBU(I+3),PTY(I+3),PVA(I+3)
     4,                       I+4,PBU(I+4),PBU(I+4),PTY(I+4),PVA(I+4)
2000                  FORMAT(5(I8,2X,Z4.4,'/',O4.4,2I3))
300           CONTINUE
              DO 320 I=1,MAXLPN
320           PBUT(I)=PBU(I)

              DO 360 I1=1,MAXLPN
                      PBSAV=999999
                      DO 340 I2=1,MAXLPN
                              IF(PBUT(I2).LT.PBSAV) THEN
                                      PBSAV=PBUT(I2)
                                      ISAV=I2
                              ENDIF
340                   CONTINUE
```

```
                        LBUT(I1)=ISAV
                        PBUT(ISAV)=999999
360               CONTINUE
C                *GO TO TOP OF NEXT PAGE FOR B-UNIT ORDERED LIST
                  WRITE(9,2002)
2002              FORMAT('1')

C                *PRINT B-UNIT ORDERED LIST
                  DO 380 J=1,MAXLPN,5
                        I=LBUT(J)
                        IP1=LBUT(J+1)
                        IP2=LBUT(J+2)
                        IP3=LBUT(J+3)
                        IP4=LBUT(J+4)
                        WRITE(9,2000) I,PBU(I),PBU(I),PTY(I),PVA(I)
     1,                        IP1,PBU(IP1),PBU(IP1),PTY(IP1),PVA(IP1)
     2,                        IP2,PBU(IP2),PBU(IP2),PTY(IP2),PVA(IP2)
     3,                        IP3,PBU(IP3),PBU(IP3),PTY(IP3),PVA(IP3)
     4,                        IP4,PBU(IP4),PBU(IP4),PTY(IP4),PVA(IP4)
380               CONTINUE
                ENDIF
              ENDIF
              IF(MAXLPN.EQ.500) FIRST=.FALSE.
            ENDIF
            RETURN
            END
            SUBROUTINE BREP
            IMPLICIT INTEGER*4 (A-Z)

            LOGICAL FIRST

            COMMON/PLAY/LPN,PB,PT,PV,PBU(500),PTY(500),PVA(500),FIRST

            IF((FIRST) .OR. ((.NOT.FIRST).AND.
     1                  ((PBU(LPN).NE.PB).OR.
     2                   (PTY(LPN).NE.PT).OR.
     3                   (PVA(LPN).NE.PV)))) THEN
                  IF(.NOT.FIRST) THEN
                     PRINT 1000,LPN,PBU(LPN),PB,PTY(LPN),PT,PVA(LPN),PV
                     WRITE(9,1000) LPN,PBU(LPN),PB,PTY(LPN),PT,PVA(LPN),PV
1000                 FORMAT(' PL CH FOR LPN ',I3,2Z6.4,4I9)
                  ENDIF
                  PBU(LPN)=PB
                  PTY(LPN)=PT
                  PVA(LPN)=PV
            ENDIF
            RETURN
            END
```

```
      SUBROUTINE CSCON(WORDIN,DATE,TIMOUT)
C-----------------------------------------------------------------
C
C     NAME - CSCON
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - CSCON gets the time of date and date from
C             an input conatining the number of centiseconds
C             since the beginning of the year.
C
C     INCLUDE FILES - None
C
C     CALLING SEQUENCE DESCRIPTION
C             WORDIN - Word containing centiseconds since 12/31
C                      of previous year.
C             DATE   - 9 character date returned (i.e. 10-JUN with
C                      2 blanks leading, 1 following)
C             TIMOUT - 8 character time of day (i.e. 13:03:35)
C
C     INPUTS - From calling sequence
C
C     OUTPUTS - To calling sequence
C
C     SUBROUTINES CALLED -- JDAY - Returns date for julian date
C                           MSCON - Returns time of day for
C                                   milliseconds since midnight
C-----------------------------------------------------------------
      IMPLICIT INTEGER*4(A-Z)
      CHARACTER*9 DATE
      CHARACTER*8 TIMOUT
      REAL*8 CSTIM,CSVAL

      DATA CSVAL/8640000.0/

C    *CONVERT TIME TO REAL*8 TO SOLVE SIGN PROBLEM
      CSTIM=DBLE(ABS(WORDIN))
C    *If sign bit was set, add the value back in
      IF(WORDIN.LT.0) CSTIM=CSTIM+2.0**31
C    *JD is really julian date-1
      JD=INT(CSTIM/CSVAL)
C    *get remainder in milliseconds
      MS=10*INT(CSTIM-JD*CSVAL)

C    *If Julian date looks kosher
      IF((JD.GE.0).AND.(JD.LE.3661)) THEN
C       *Get the date in English
        CALL JDAY(JD,DATE)
C     ENDIF
C    *Put the time into as ASCII string
```

```
CALL MSCON(MS,TIMOUT)
RETURN
END
```

```
      SUBROUTINE MSCON(MS,TIME)
C-----------------------------------------------------------------
C
C     NAME - MSCON
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - MSCON returns an 8-character ASCII string
C                   containing the time of day, given an integer that
C                   contains the number of milliseconds since midnight.
C
C     INCLUDE FILES - None
C
C     CALLING SEQUENCE DESCRIPTION
C                   MS      - Word containing milliseconds since midnight
C                   TIME    - 8 character time of day (i.e. 13:03:35)
C
C     INPUTS - From calling sequence
C
C     OUTPUTS - To calling sequence
C
C     SUBROUTINES CALLED - None
C-----------------------------------------------------------------
      IMPLICIT INTEGER*4 (A-Z)
      CHARACTER*8 TIME

C    *Convert milliseconds, first to seconds, then to
C    * hours, minutes, and seconds
      TSEC=MS/1000
      HRS = TSEC/3600
      MIN = (TSEC-HRS*3600)/60
      SEC = (TSEC-HRS*3600-MIN*60)
C    *Now put time into an ASCII string  HH:MM:SS
      WRITE(TIME,2000) HRS,MIN,SEC
2000  FORMAT(I2.2,':',I2.2,':',I2.2)
      RETURN
      END
```

```
      SUBROUTINE JDAY(JD,RETSTR)
C-----------------------------------------------------------------
C
C     NAME - JDAY
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - JDAY returns a 9-character ASCII string
C             containing the date, given an integer that contains
C             the Julian date+1 (number of days since last 12/31).
C
C     INCLUDE FILES - None
C
C     CALLING SEQUENCE DESCRIPTION
C             JD     - Word containing days since 12/31 previous year
C             RETSTR - 9 character date i.e. ##10#JUN#, where #
C                      represents a blank character.
C
C     INPUTS - From calling sequence
C
C     OUTPUTS - To calling sequence
C
C     SUBROUTINES CALLED - None
C-----------------------------------------------------------------
      IMPLICIT INTEGER*4 (A-Z)
      CHARACTER*9 RETSTR
      CHARACTER*3 DST,MON(12)
      DIMENSION MONNO(0:12)
      DATA MON/'Jan','Feb','Mar','Apr','May','Jun'
     1,        'Jul','Aug','Sep','Oct','Nov','Dec'/
      DATA MONNO/0,31,59,90,120,151,181,212,243,273,304,334,365/

C    *Blank out output ASCII string to start
      RETSTR='         '

      I=0
      DO WHILE ((JD.GE.MONNO(I)) .AND. (I .LE.12))
              I=I+1
      END DO
      IF(I.LE.12)THEN
C        *Set 3-character month name into characters 6-8
         RETSTR(6:8)=MON(I)
C        *Encode ASCII day of month into characters 3-5
         DDIF=JD-MONNO(I-1)
         WRITE(DST,1001) DDIF
1001     FORMAT(I2,' ')
         RETSTR(3:5)=DST
      ENDIF
      RETURN
      END
```

```
      SUBROUTINE REPACK(WORD)
C-----------------------------------------------------------------
C
C     NAME - REPACK
C
C     AUTHOR - JACK BRISCOE
C
C     DATE - 15 APRIL, 1985
C
C     DESCRIPTION - REPACK takes a 32-bit word and reverses the
C                   order of the bytes.  If the bytes were originally
C                   numbered 1234, they would wind up 4321.
C
C     INCLUDE FILES - None
C
C     CALLING SEQUENCE DESCRIPTION
C                   WORD - The word in which the bytes are to be reordered.
C
C     INPUTS - See calling sequence
C
C     OUTPUTS - See calling sequence
C
C     SUBROUTINES CALLED
C                   MVBITS - System subroutine to manipulate bits
C
C-----------------------------------------------------------------
      IMPLICIT INTEGER*4 (A-Z)

      CALL MVBITS(WORD, 0,8,NUWORD,24)
      CALL MVBITS(WORD, 8,8,NUWORD,16)
      CALL MVBITS(WORD,16,8,NUWORD, 8)
      CALL MVBITS(WORD,24,8,NUWORD, 0)
      WORD=NUWORD
      RETURN
      END
```