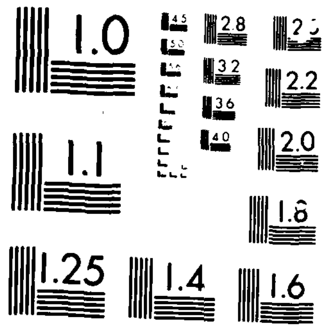
AD-A166 031 INTEGRATED BATTLEFIELD EFFECTS RESEARCH FOR THE NATIONAL TRAINING CENTER..(U) SCIENCE APPLICATIONS INTERNATIONAL CORP LA JOLLA CA D ERICKSON ET AL. 1/1

UNCLASSIFIED 31 DEC 84 SAIC-R-LJF-84-019-APP-K F/G 5/9 NL



MICROCOPY RESOLUTION TEST CHART

1

DNA-TR-85-13-AP-K

INTEGRATED BATTLEFIELD EFFECTS RESEARCH FOR THE NATIONAL TRAINING CENTER

Appendix K—ARTBASS Conversion Design and Results

Science Applications International Corporation
P. O. Box 2351
La Jolla, CA 92038-2351

31 December 1984

Technical Report

CONTRACT No. DNA 001-81-C-0273

THIS WORK WAS SPONSORED BY THE DEFENSE NUCLEAR AGENCY
UNDER RDT&E RMSS CODE S400082466 V99QAXNL00125 H2590D.

Prepared for
Director
DEFENSE NUCLEAR AGENCY
Washington, DC 20305-1000

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY: N/A since Unclassified	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A since Unclassified	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): R:LJF-84-019	5. MONITORING ORGANIZATION REPORT NUMBER(S): DNA-TR-85-13-AP-K
6a. NAME OF PERFORMING ORGANIZATION: Science Applications International Corporation; 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: Director Defense Nuclear Agency
6c. ADDRESS (City, State, and ZIP Code): P.O. Box 2351 La Jolla, CA 92038-2351	7b. ADDRESS (City, State, and ZIP Code): Washington, DC 20305-1000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION; 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: DNA 001-81-C-0273
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO 62715H; PROJECT NO V99QAXN; TASK NO L; WORK UNIT ACCESSION NO DH065313

11. TITLE (Include Security Classification)
INTEGRATED BATTLEFIELD EFFECTS RESEARCH FOR THE NATIONAL TRAINING CENTER
Appendix K—ARTBASS Conversion Design and Results

12. PERSONAL AUTHOR(S)
Erickson, D.; Ickler, J.; McKeown, P.; Metzger, L.; Plock, R.; Packard, B.; and Birney, J.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM 830613 TO 841230	841231	76

16. SUPPLEMENTARY NOTATION
This work was sponsored by the Defense Nuclear Agency under RDT&E RMSS Code S400082466 V99QAXNL00125 H2590D.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	
15	7		Training; Integrated Battlefield; Military Strategy; Military Doctrine
5	9		

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

Research performed to evaluate and develop enhancements for integrated battlefield training at the U.S. Army National Training Center is described. These enhancements had been identified and concepts developed for their application in earlier phases of this research. The report consists of the basic volume summarizing the research tasks, approach, results, conclusions, and recommendations; plus twelve appendices which provide details on the nine major tasks into which the research was divided. Research performed and the associated appendices are as follows:

Development of nuclear and chemical environmental and effects software:

Analysis of nuclear algorithms	Appendix A
Requirements specification for nuclear and chemical model algorithms at the NTC	Appendix B
Chemical model algorithm description	Appendix C

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Betty L. Fox	22b. TELEPHONE (Include Area Code): (202) 325-7042; 22c. OFFICE SYMBOL: DNA/STTI

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. ABSTRACT (Continued)

Demonstration of the system for combining live and notional battalions for training higher level staffs in integrated battlefield (IB) command and control:

Functional requirements analysis for IB command and control simulation	Appendix D
Report on the demonstration	Appendix E

Analysis and design of field simulators for nuclear and chemical warfare:

Technical and operational impacts of field simulators	Appendix F
Capability of off-the-shelf paging system to communicate at Ft. Irwin	Appendix G
Designs of field simulators	Appendix H

Adaptation of nuclear and chemical software to other Army training models:

Feasibility of transferring ARTBASS Code from Perkin-Elmer to VAX	Appendix I
Division/Corps training simulation functional analysis	Appendix J
ARTBASS conversion to VAX	Appendix K
Requirements specification for adding nuclear and chemical models to ARTBASS	Appendix L

This research provided the following products:

Software which models nuclear and chemical environment and effects with appropriate fidelity and timing for training and which is ready for installation on NTC computers.

A demonstrated capability for combining actions of real battalions with computer simulated notional battalions for training brigade/division commanders and staffs.

An analysis of the impacts of using field simulators at the NTC for nuclear and chemical warfare training, and the designs of the selected simulators (i.e., common control system, radiacmeters, dosimeters, chemical detectors).

Analysis of the application of nuclear and chemical models to other Army battalion training models; conversion of the ARTBASS model to operate on the VAX 11/780; incorporation of the nuclear and chemical models into ARTBASS; and demonstration of the nuclear and chemical models using ARTBASS.

CONVERSION FACTORS FOR U.S. CUSTOMARY TO METRIC (SI) UNITS OF MEASUREMENT

To Convert From	To	Multiply By
angstrom	meters (m)	1.000 000 x E -10
atmosphere (normal)	kilo pascal (kPa)	1.013 25 X E +2
bar	kilo pascal (kPa)	1.000 000 X E +2
barn	meter^2 (m^2)	1.000 000 X E -28
British thermal unit (thermochemical)	joule (J)	1.054 350 X E +3
cal (thermochemical)/cm^2	mega joule/m^2 (MJ/m^2)	4.184 000 X E -2
calorie (thermochemical)	joule (J)	4.184 000
calorie (thermochemical)/g	joule per kilogram (J/kg)*	4.184 000 X E +3
curie	giga becquerel (Gbq) †	3.700 000 X E +1
degree Celsius	degree kelvin (K)	$t_K = t°_C + 273.15$
degree (angle)	radian (rad)	1.745 329 X E -2
degree Farenheit	degree kelvin (K)	$t_K = (t°_F + 459.67)/1.8$
electron volt	joule (J)	1.602 19 X E -19
erg	joule (J)	1.000 000 X E -7
erg/second	watt (W)	1.000 000 X E -7
foot	meter (m)	3.048 000 X E -1
foot-pound-force	joule (J)	1.355 818
gallon (U.S. liquid)	meter^3 (m^3)	3.785 412 X E -3
inch	meter (m)	2.540 000 X E -2
jerk	joule (J)	1.000 000 X E +9
joule kilogram (J/kg) (radiation dose absorbed)	gray (Gy)*	1.000 000
kilotons	terajoules	4.183
kip (1000 lbf)	newton (N)	4.448 222 X E +3
kip/inch^2 (ksi)	kilo pascal (kPa)	6.894 757 X E +3
ktap	newton-second/m^2 (N-s/m^2)	1.000 000 X E +2
micron	meter (m)	1.000 000 X E -6
mil	meter (m)	2.540 000 X E -5
mile (international)	meter (m)	1.609 344 X E +3
ounce	kilogram (kg)	2.834 952 X E -2
pound-force (lbf avoirdupois)	newton (N)	4.448 222
pound-force inch	newton-meter (N·m)	1.129 848 X E -1
pound-force/inch	newton/meter (N/m)	1.751 268 X E +2
pound-force/foot^2	kilo pascal (kPa)	4.788 026 X E -2
pound-force/inch^2 (psi)	kilo pascal (kPa)	6.894 757
pound-mass (lbm avoirdupois)	kilogram (kg)	4.535 924 X E -1
pound-mass-foot^2 (moment of inertia)	kilogram-meter^2 (kg·m^2)	4.214 011 X E -2
pound-mass/foot^3	kilogram-meter^3 (kg/m^3)	1.061 846 X E -1
rad (radiation dose absorbed	gray (Gy)*	1.000 000 X E -2
roentgen	coulomb/kilogram (C/kg)	2.579 760 X E -4
shake	second (s)	1.000 000 X E -8
slug	kilogram (kg)	1.459 390 X E -1
torr (mm Hg, 0° C)	kilo pascal (kPa)	1.333 22 X E -1

*The gray (Gy) is the accepted SI unit equivalent to the energy imparted by ionizing radiation to a mass and corresponds to one joule/kilogram.

†The becquerel (Bq) is the SI unit of radioactivity; 1 Bq = 1 event/s.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

SECTION 1

INTRODUCTION

The conversion of ARTBASS from the prototype Perkin-Elmer/Lexidata computer system to the VAX/DeAnza computer system was guided by a set of ground rules. These ground rules were initially established in Phase 2 of the project, and were refined in the beginning stages of Phase 3. The following paragraphs briefly describe these ground rules.

There were two main reasons for performing the conversion task. First, a battle simulation suitable for training battalion and brigade command groups was needed to support the C3I Training task (SOW Task 11), and it was mutually agreed that the prototype version of the ARTBASS simulation was the superior choice for this purpose. Also, CAORA personnel needed ARTBASS working on VAX equipment to allow training concept development activities to occur in the time frame before a production version of ARTBASS would be available, and to permit concept development and experimentation to occur in a VAX-based system during the operational phase of the ARTBASS life cycle.

To support the C3I Training task, ARTBASS needed to be installed on NTC-compatible equipment, namely, a VAX computer with the DeAnza-based NTC workstation. To support training concept development activities, ARTBASS needed to be installed on a VAX computer in such a manner that it would operate with a Lexidata-based ARTBASS workstation, although such a workstation was not available at the time. To satisfy both requirements, it was decided to convert ARTBASS for operation on the NTC equipment, but perform the software design such that the ability to operate with the standard ARTBASS workstation would also be retained.

Due to the differences between the two computer systems, all system services and utilities required by the application software needed to be changed. These differences are well-isolated and easily defined, and have no effect on the operation of the ARTBASS model. The applications software needed to be left intact. This would allow any future modifications and enhancements to be compatible between the two computer systems. To the extent that modifications are required to the applications, they should be well documented, and specific rules developed and documented for accomodating the modifications during transfer of software between the two systems.

It was agreed that there would be no attempt to convert the 3-dimensional map and Computer Assisted Instruction capabilities of ARTBASS to the VAX computer system. These capabilities were not needed for the C3I Training task, and were not considered as necessary in the context of a training concept development function.

Based on the ground rules discussed above, a design for the conversion effort was developed, and the conversion performed. SECTION 2 describes the design behind conversion of the ARTBASS model, and SECTION 3 contains the design of the Interactive Display Component (the interface of the model with the two types of workstations). SECTION 4 describes the conversion goals accomplished through use of software tools performing standard conversion tasks. Software changes necessary for ARTBASS to operate on the VAX, but are not compatible with the Perkin-Elmer computer system are identified in SECTION 5, and those changes made that are compatible with the Perkin-Elmer are described in SECTION 6. During performance of the conversion activities, errors were discovered in the ARTBASS code. Most of these errors were corrected on the VAX version of ARTBASS. All of the errors are defined in SECTION 7. Finally, SECTION 8 defines the procedures and techniques necessary to transfer ARTBASS code from the VAX system back to a Perkin-Elmer system.

SECTION 2

MODEL CONVERSION DESIGN

2.1 INTRODUCTION

The ARTBASS model was installed on the VAX/VMS operation system using the same design as on the Perkin-Elmer. The model is divided into numerous modules, each module performing a specific function and executing as a subprocess. The subprocesses are loaded and controlled by an executive process. The executive process uses a source database to determine which suprocesses should execute in a specific mode. By having the model divided into modes, only the subprocesses required for a particular mode are required to be loaded into memory. This provides for a large savings in memory requirements. Paragraph 1.2 describes the modes, paragraph 1.3 describes the subprocesses and the subprocesses/mode relationship is described in paragraph 1.4.

2.2 MODEL MODES

The model can execute in twelve different modes. The mode the model executes in is determined by the startup parameters or by the simulation control menu. The model modes consist of the following:

REINITIALIZATION	- Starts the model over at the beginning with a new scenario.
REPLAY	- Replays the graphical representation of an exercise.
RESTART	- Starts the model over using the same initialization and scenario.
END OF GAME	- Terminates the model, prints the postexercise reports and stops the command and control subprocesses.
REPLAY TERMINATION	- Terminates the replay subprocesses.

FREEZE	- Stops the model until a simulation control RUN is executed.
RUN	- Starts the model after a freeze action.
SIMULATION	- Normal model simulation.
REPLICATE	- Same as a normal simulation except the events are read from a replicate file instead of receiving the events from the menu system.
RECOVERY	- Recovers the model to any fifteen minute increment of an exercise.
INITIALIZATION	- Initializes the model by reading in a scenario database and processing the controller initialization menu inputs.
REPLAY INITIALIZATION	- Sets up the globals for start of a replay.

2.3 MODEL SUBPROCESSES

All the ARTBASS model subprocesses are started by the executive when a mode is enabled. The main program of each subprocess (usually named MAIN) opens the required databases and enables a queue trap for that process. From then on, the process is woke up by the executive queuing it, and execution begins at the queue trap handler routine (usually named QT). When the process completes, it goes into a hibernation state until it is queued again.

The following list gives a brief explanation of all the subprocesses that make up the ARTBASS model:

ADJBL	- Adjusts the unit equipment and ammunition basic load.
ADJDIR	- Adjusts the direction a unit is facing.
ADJROM	- Adjusts the unit rate-of-movement.
ADMINLOG	- Administrative/logistics resource management module.
AIRMOV	- Air movement module.
AIRMOV2	- Air-to-ground and ground-to-air damage processor.
ALERTPP	- Tactical alert postprocessor.
ASSIGN	- Validates the reinitialization scenario selection.
CASREP	- Prints the casualty report.

CCMDAFG	- Creates the work version of the cross-country movement database.
CCMREST	- Restores the cross-country movement database to a 15 minute increment for recovery.
CCMXEC	- Cross-country movement processor.
CCTERM	- Terminates the command and control processes.
CLEANUP	- Generates the end-of-game RAMALERT and tactical alert.
COMPDIAG	- Computes the unit diagonal distance.
DETECT	- Unit detection processor (visual, aural and radar).
ENGAGE	- Unit engagement module.
EVENTPP	- Event postprocessor. Prints the interactive menu inputs.
EXEC	- ARTBASS model executive.
FORSIGX	- Processes the events from the interactive menu process.
FREEZE	- Stops the model until the simulation control RUN is executed.
GNDFIR	- Direct and indirect ground fire processor.
INIT1	- Generates the "scenario initialization" RAMALERT.
INIT2	- Generates the "last chance to relocate units" RAMALERT.
INPUT	- Reads the scenario database and namelist.
LOSINP	- Unit line-of-sight processor.
MINEFLD	- Minefield processor.
MOVMNT	- Unit movement module.
MSC10	- Moves arrays into global for access by the menu and graphic display software.
MSC6	- Sets the radar parameters and CCM block boundary coordinates.
OBSUPDA	- Obstacle update module.
OPPLAN	- Operational group processor.
PGSRPT	- Produces the postgame reports.

RCOVERY	- Recovers the model to any 15 minute increment of the exercise.
REDIST	- Redistributes unit assets accoding to manning priorities.
REPLAY	- Replays the graphical representation of an exercise.
RPTERM	- Terminates a replay.
RPVALID	- Replay validation module.
RSTART	- Restarts the model with the same scenario and initialization.
SAVE	- Saves the globals to the save databases for REPLAY and RECOVERY.
SAVEBD	- Saves the blockdata initialization to disk.
START	- Generates the end-of-initialization RAMALERT and saves the unit basic load data.
STATRPT	- Prints the unit, equipment, ammunition and other model status reports.
STEP	- Performs end of timestep analysis and alert generation.
SUPRES	- Unit supression processor.
TMOVLY	- Queues the graphic display process to redraw.
WETHR	- Weather processor.

2.4 SUBPROCESS/MODE RELATIONSHIP

This section describes which subprocess are executed as a part of each mode. The mode is listed followed by all the subprocesses in the mode.

REINITIALIZATION

SAVEBD
ASSIGN
INPUT

REPLAY

REPLAY
TMOVLY

RESTART

- RSTART
- TMOVLY
- FREEZE

END OF GAME

- TMOVLY
- SAVE
- CLEANUP
- FREEZE
- STATRPT
- CASREP
- EVENTPP
- ALERTPP
- PGSRPT
- CCTERM
- FREEZE

REPLAY TERMINATION

- RPTERM
- TMOVLY
- FREEZE

FREEZE

RUN

SIMULATION

- SAVE
- FORSIGX
- COMPDIA
- MSC10
- TMOVLY
- WETHR
- REDIST
- STATRPT
- SUPRES
- OPPLAN
- ADJROM
- DETECT
- AIRMOV2
- GNDFIR
- AIRMOV
- CCMXEC
- ADMINLOG
- MOVMNT
- OBSUPDA
- ADJDIR
- ENGAGE
- STEP
- LOSINP

REPLICATE

SAVE
FORSIGX
COMPDIA
MSC10
TMOVLY
WETHR
REDIST
STATRPT
SUPRES
OPPLAN
ADJROM
DETECT
AIRMOV2
GNDFIR
AIRMOV
CCMXEC
ADMINLOG
MOVMNT
OBSUPDA
ADJDIR
ENGAGE
STEP
LOSINP

RECOVERY

RCOVERY
CCMDAFG
START
FREEZE

INITIALIZATION

LOSINP
MSC6
MINEFLD
COMPDIA
INIT1
FREEZE
FORSIGX
TMOVLY
FREEZE
FORSIGX
TMOVLY
INIT2
FREEZE
FORSIGX
TMOVLY
ENGAGE
ADJBL
CCMDAFG
LOSINP
START
FREEZE

REPLAY INITIALIZATION

RPVALID
FREEZE

SECTION 3

INTERACTIVE DISPLAY COMPONENT DESIGN

3.1 INTRODUCTION

The interactive display component (IDC) was installed on the VAX/VMS operating system using the same basic design as on the Perkin-Elmer. The Perkin-Elmer IDC design consisted of two computers, one for the color display (MAP) and one for the other peripherals (CIP). The software design on the VAX retained the two computer structure; however, much of the interprocess communications was simplified because all the processes are executing in one processor.

An additional difference in the design is the controller station peripherals that were used. These hardware differences will be addressed individually in paragraph 3.2. The subprocesses that make up the IDC will be described in paragraph 3.3.

3.2 HARDWARE DIFFERENCES

The Perkin-Elmer controller stations consist of the following hardware:

Lexidata 3400 color graphics processor
Lexidata 19" color monitor
Multifunction keyboard
Graph tablet and pen
P/E alphanumeric terminal
Control station printer

The VAX controller station consists of the following hardware:

DeAnza VC-23 color graphics processor
DeAnza 19" color monitor
VT-105 alphanumeric/graphic terminal
Graph tablet and pen

The following paragraphs will describe the design used to overcome the hardware differences listed above.

3.2.1 Color Graphics Processor

The Lexidata color graphics processor has a pixel resolution of 640 x 512 versus 512 x 480 in the DeAnza. The interactive menus used the full 640 pixels to display and interact with the menus. The leftmost 90 pixels are used for time display and the rest for the interactive menu. To implement the interactive menus without rewriting all the application routines, the time portion of the menu was moved from the main menu to an additional page and the remaining 550 pixels were squeezed down to 512 by the graphic display utilities.

Two additional problems with the graphics processors include the location of the origin and the display of the cursor. The origin in the Lexidata is in the upper left corner of the screen and the origin of the DeAnza is in the lower left corner. To keep from rewriting all the graphic display application software, the graphic display utilities were written to convert Lexidata coordinates into DeAnza coordinates.

The display of the cursor on the Lexidata was a very complicated process because the graph tablet was attached to a different computer than the Lexidata. Because of the hardware design, there is a large amount of communications software to pass the cursor locations and state to the Lexidata. In the VAX version, the graph tablet is attached to the DeAnza, allowing cursor display without interrupting the VAX. Cursor tracking is done by the LSI in the DeAnza and only valid pen downs are transferred to the VAX.

The digital map display on the Lexidata used the full 640 x 480 pixels. Rather than convert the map display software to work in a 512 x 480 pixel display, the National Training Center (NTC) map display software was implemented. The only additional software required, was some coordinate conversion utilities to allow for correct map to pixel translations.

3.2.2 Alphanumeric Display

The Perkin-Elmer controller station has Perkin-Elmer terminals attached to the CIP computer and the VAX controller station has VT-100 terminals attached to the DeAnza. As it turns out, the escape sequences and terminal characteristics of the P/E and VT-100 are very different. To keep from changing the application software, conversion software was written for the LSI processor to convert P/E escape sequences into equivalent VT-100 commands. The conversion process is very complicated and causes poor response time on the alphanumeric display.

3.2.3 Multifunction Keyboard

A multifunction keyboard is used on the Perkin-Elmer controller station to select interactive menus and graphic display options. Being the VAX hardware does not include a multifunction keyboard, a master menu was designed to provide the same functions. The master menu resides in a memory plane in the DeAnza and is called up by selecting the master menu button on the tablet overlay.

By having both the master menu and the graph tablet in the DeAnza, many of the map display options are handled by software in the DeAnza LSI without interrupting the VAX. A tablet overlay that matches the master menu is shown in figure 1.

3.2.4 Graph Tablets

Being the graph tablets are attached to the DeAnza in the VAX version, the NTC graph tablet software was used instead of converting the Perkin-Elmer software.

3.3 SUBPROCESS DESCRIPTION

The Interactive Display Component consists of a series of subprocesses to display and interact with interactive menus, display RAMALERTS, display digital maps, display alerts and status reports and display the tactical symbology. The interprocess communication (see figure 2) is controlled by the MAP processor executive and the CIP executive. The following gives a brief description of the subprocesses that make up the Interactive Display Component:

ALERTSn	- Alert processor where n = the controller number. ALERTSn handles the interaction with the alphanumeric terminal. It accepts function key inputs from process FKLSIP and performs the appropriate actions. It also accepts alerts and status reports and passes them to the LSI for display.
ALTHNDLR	- Alert handler. ALTHNDLR accepts alerts from the CIP executive and queues them for the alert processor (ALERTSn).
EXECE3	- CIP executive. Controls the interprocess communications for the processes belonging to the CIP computer. In the VAX version this consists of alerts and status reports.
EXECMP	- MAP executive. Controls the interprocess communication for all the MAP subprocesses.

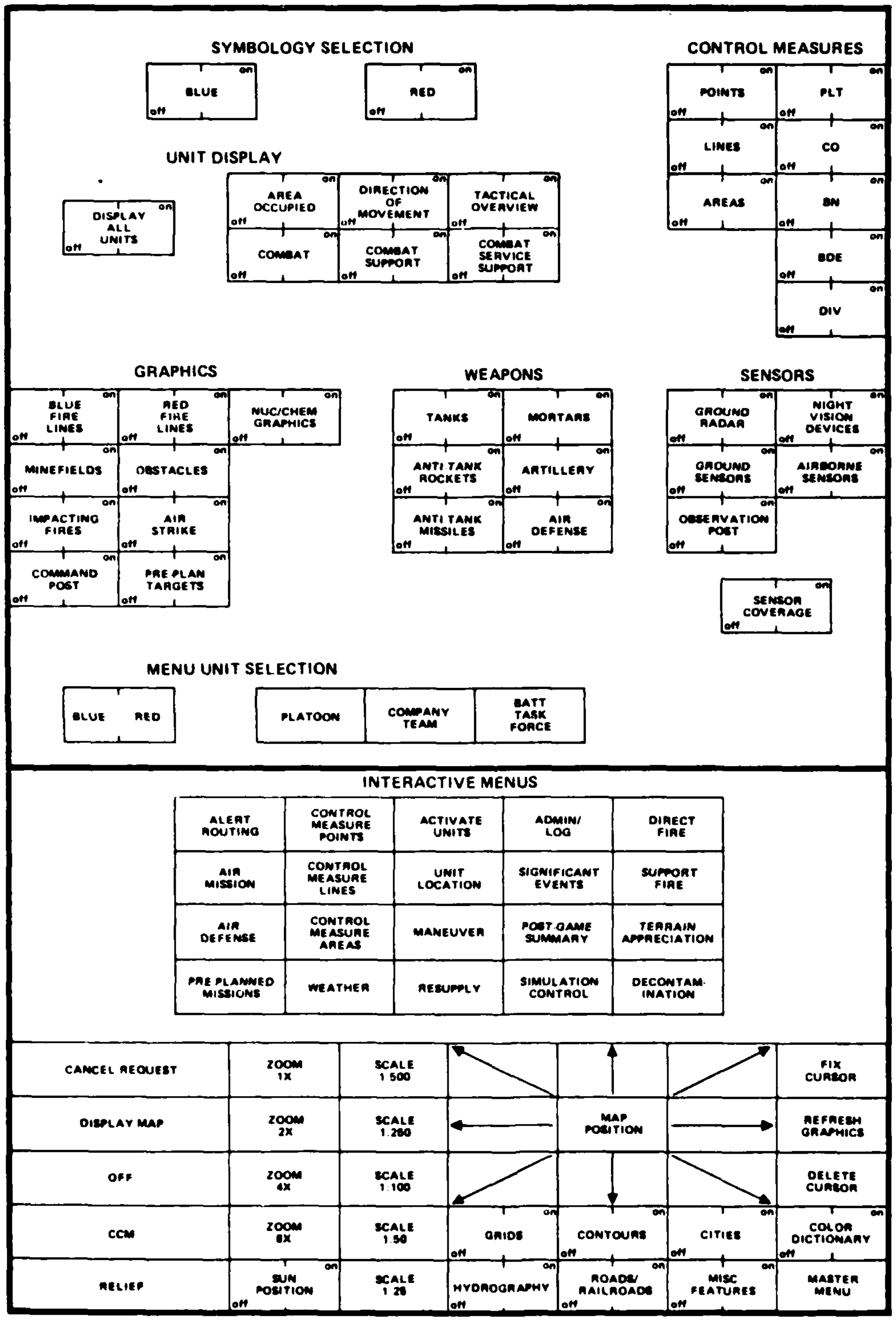


Figure 1. Tablet overlay.

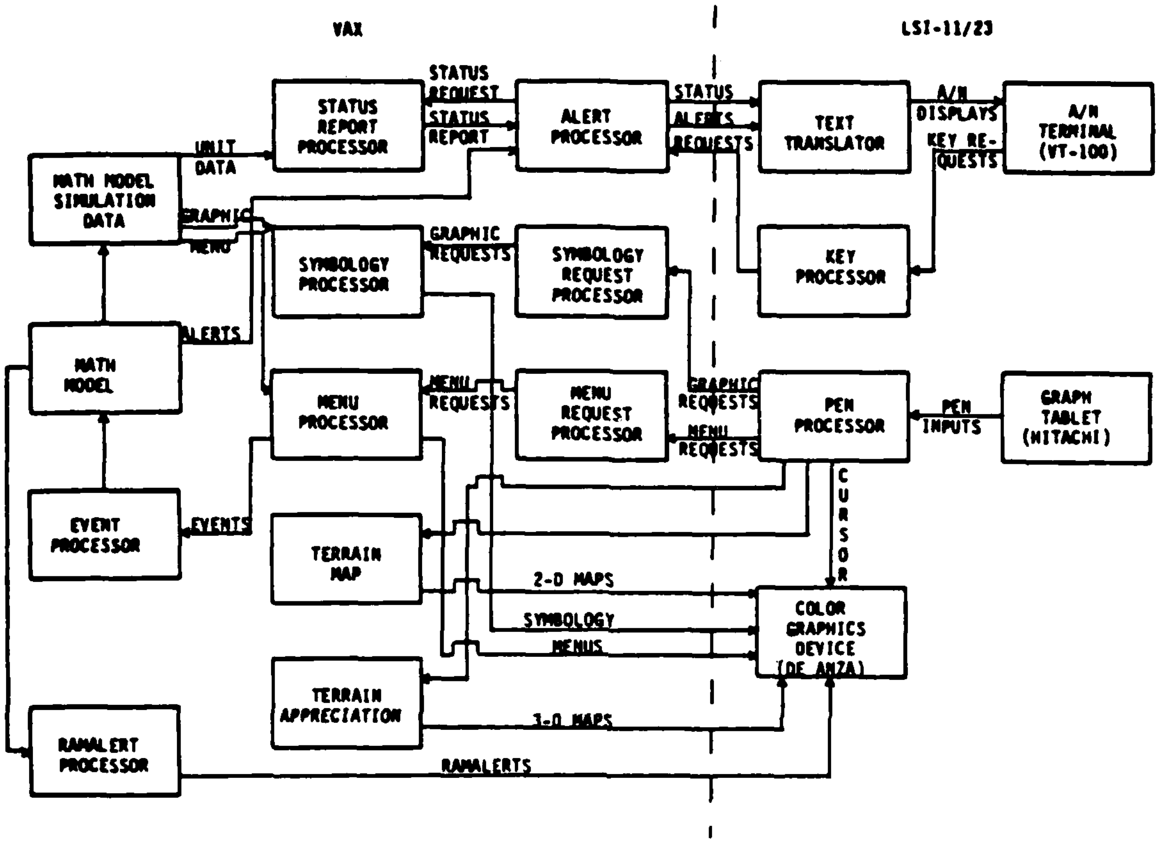


Figure 2. Blast architecture using DeAnza.

EXRAMQP	- RAMALERT processor. EXRAMQP accepts RAMALERTS from the ARTBASS model and queues them until they are requested by the interactive menu processor (MENUn).
FGEVENT	- Event processor. FGEVENT accepts events from the interactive menu processor (MENUn) and passses them to the event processor in the model (FORSIGX).
FKLSIP	- Function key processor. FKLSIP accepts function key inputs from the LSI and passes them to the alerts processor (ALERTSn).
GDn	- Graphic display processor where n = the controller number. GDn displays the tactical symbology on the color display.
MAPXEC	- Digital map display process. MAPXEC transfers the map display parameters to the DeAnza LSI and receives map display requests from the LSI. When MAPXEC receives a map display request, the map parameters are loaded into ARTBASS globals for correct map to pixel conversions and the new map is displayed. MAPXEC also queues the executive to request a symbology redraw for the new map.
MENUn	- Interactive menu processor where n = the controller number. MENUn displays and processes the interactions with all the interactive menus.
MNLSIP	- Menu input processor. MNLSIP accepts graph tablet pen inputs from the LSI and passes them to the menu system by queuing the MAP executive.
MP2LSI	- MAP to LSI communication process. MP2LSI instructs the LSI in the DeAnza when to turn the cursor on and off due to a menu being requested or completed.
SBLSIP	- Symbology input processor. SBLSIP accepts the bitmaps of the symbology button settings on the master menu and transfers them to the graphic display process by queuing the MAP executive.
STATRPT	- Status report generator. STATRPT receives a unit name from the CIP processor, validates it, creates the unit status report and transfers it to the CIP for display.

SECTION 4

MAJOR CONVERSIONS

This SECTION describes major categories of software changes required for the implementation on the VAX. The changes include differences in FORTRAN statements, memory storage techniques and common application subroutine changes that had to be made to many subroutines. The first part of the SECTION describes the software changes and the second part describes the utilites that were developed to remain compatible with the Perkin-Elmer.

4.1 MASS SOFTWARE CHANGES

4.1.1 ENCODE And DECODE Statements

All ENCODE and DECODE statements in the ARTBASS system had to be converted to the VAX format for the ARTBASS implementation. The Perkin-Elmer format is

```
ENCODE (buf, f) olist
```

Where

buf	is the name of a variable, array, or array element into which data transfer begins.
f	is a format identifier.
olist	is an output list.

The VAX FORTRAN ENCODE's format is

```
ENCODE (c, f, buf) olist
```

Where

c	is the number of characters to be translated to character format.
f	is a format identifier.
buf	is the name of a variable, array, or array element into which data transfer begins.
olist	is an output list

4.1.2 HEX Constants

To load the ARTBASS system on to the VAX all HEX constants were converted from the Perkin-Elmer format to the VAX format.

SIZE	P/E FORM	VAX FORM
32 bit	x'nnnnnnnn'	'nnnnnnnn'x
16 bit	y'nnnn'	N/A

4.1.3 BYTE Order

The order the bytes are numbered in the Perkin-Elmer is different than in the VAX.

Perkin-Elmer Order

```
                WORD 1                                  WORD 2
  ------ ------ ------ ------      ------ ------ ------ ------
 |  0   |  1   |  2   |  3   |    |  4   |  5   |  6   |  7   |
  ------ ------ ------ ------      ------ ------ ------ ------
```

VAX Order

```
                WORD 1                                  WORD 2
  ------ ------ ------ ------      ------ ------ ------ ------
 |  3   |  2   |  1   |  0   |    |  7   |  6   |  5   |  4   |
  ------ ------ ------ ------      ------ ------ ------ ------
```

This difference in order caused severe problems in the conversion because there was a large mixture of byte utilities and masks in the ARTBASS code. Many of the problems were fixed by replacing the masks and shifts with the byte utilities. However, in displaying ASCII text on the color display many of the shifts and masks were too complex to use the utilities and still allow the code to be transferred to the Perkin-Elmer. To correct this problem, a general purpose integer to ASCII function was implemented to handle almost all required conversions without worrying about the size of the destination. The format of this utility is:

```
log = BINASC ( dest, ind, num, val, hex, zero)
```

Where

log	is a local logical variable
	.true. = error in conversion
	.false. = no error in conversion.
dest	is the destination to store ASCII string.
ind	is the offset into (dest).
num	is the number of characters to convert.
val	is the binary value to convert.
hex	is a hexadecimal output logical flag
	.true. = output ASCII in HEX
	.false. = output ASCII in decimal.
zero	is a zero fill flag
	.true. = zero fill on ASCII output
	.false. = no zero fill on ASCII output.

A utility function similar to this could be written for the Perkin-Elmer so code developed on the VAX can be transferred to the Perkin-Elmer.

4.1.4 Halfword Order

The order the halfwords are numbered in the Perkin-Elmer is different than in the VAX.

Perkin-Elmer Order

```
            WORD 1                          WORD 2
   ------ ------ ------ ------   ------ ------ ------ ------
  |       0     |      1      | |      2      |      3      |
   ------ ------ ------ ------   ------ ------ ------ ------
```

VAX Order

```
            WORD 1                          WORD 2
   ------ ------ ------ ------   ------ ------ ------ ------
  |       1     |      0      | |      3      |      2      |
   ------ ------ ------ ------   ------ ------ ------ ------
```

This problem was fixed by making sure if halfwords are packed with masks and shifts, they are unpacked with masks and shifts and if halfwords are packed with halfword utilities, they are unpacked with utilities.

4.1.5 Time Page Menu

The Perkin-Elmer version graphically displayed the time page on the left side of the graphics monitor (pixels 0-90). There was not sufficient space to display the time page and the menus on the VAX version graphics monitor so a separate time page had to be created. This new time page is accessed by the selection of TIME on the menus.

The TIME area is displayed and erased in conjunction with the DONE/REPEAT areas on the menus. The TIME area is directly above the DONE. Listed below are the subroutines which required a change consistent with the above statement.

ADDAR	ADDLN	ADDPT	AIRDEFD
ALAIR2EX	ALCHGOLD	ALGRNDEX	ALROUTXY
ALSUBEX	ALUPAIR2	APG2HAND	APRTS
ARSBHAND	DELAREA	DELLINE	DELPT
FIRECHG	FIREXEC	HANDLPPEN	LNPTS
LNPTS2	MANEUVER	MANEUVER2	MANNAME
MKCODEAR	MKCODEPT	MOVELN	MOVEPT
PG2HAND	PSTGMEX	PTPTS	PTPTS2
ROUTEXY	SIGPROC	SPTFCHG	SPTFEXEC
SUPPG2UP	UNDEACTI	UNITREUP	UPALRT
UPSIGVNT	UPWETH	WETH	

4.1.6 Menu Display Of ASCII Characters

The PERKIN/ELMER used the utility LTRS in displaying ASCII characters on the screen in the MENU system. However,the call often included an unnecessary ISHFT for the VAX due to byte order differences or a combination of ISHFT and INTCODE.

The following subroutines deleted an ISHFT before calling LTRS:

ALUPDEF	ALDEFEX	ALUPAIR1	FIREDUR
MISSPICK			

The following subroutines had the combination of ISHFT/INTCODE replaced with a BINASC call.

ALLOADEX	ALUPAIR1	ALAIR1EX	ALUPAIR2
ALAIR2EX	ALPNTS	SIMUPSIG	SIMHANDL
PSTGTIME			

4.1.7 ITOC Function

The integer to character function was rewritten for the VAX version. The new version required the function to be defined as a character function. The following subroutines were changed to accomodate this change.

OSDLQT1	OSDLQT2	FIRERALT	PGSTIME	PGSUNIT

4.2 UTILITIES

This section describes the utilities that were converted from the Perkin-Elmer version to the VAX version. The utilities consist of the following categories: Perkin-Elmer, Lexidata, new utilities and ARTBASS utilities.

4.2.1 Perkin-Elmer Utilities

The following Perkin-Elmer system utilities were written to perform the same functions on the VAX:

ABL	- Add item to the bottom of a list.
ABORT	- Abort a subprocess.
ATL	- Add item to the top of a list.
BCLR	- Bit clear.
BSET	- Bit set.
BTEST	- Bit test.
CHNGE	- Change the priority of a subprocess.
CONMSG	- Display a message on the console device.
DATE	- Retrieve current date from the computer.
DEFLST	- Define a circular list.
ENABLE	- Enable a subprocess to accept queue traps.
HALFLD	- Halfword retrieval.
HALFST	- Halfword store.
HOLD	- Suspend a subprocess.
ICLOCK	- Retrieve current time from the computer.
ILBYTE	- Byte retrieval.
INIT	- Queue trap initialization.
IRND	- Integer rounding routine.
ISBYTE	- Byte store.
ISC	- Circular shift.
LOAD	- Load and execute a subprocess.

QUEUE	- Wake up a process and pass it a parameter.
RBL	- Retrieve an item from the bottom of a list.
RELSE	- Release a subprocess from a hold.
RTL	- Retrieve an item from the top of a list.
START	- Start a subprocess.
STTSK	- Retrieve the status of a subprocess.
SYSIO	- System input/output routine.
UNEST	- Deinstall a subprocess.

4.2.2 LEXIDATA Utilities

The following utilities were converted from the LEXIDATA to the DeAnza:

ADD	- Set or clear the character additive mode.
BACK	- Set or clear the inverse color mode.
BLANK	- Erase or fill a rectangular block.
CIRCLE	- Draw a circle.
COLOR	- Establish the current color.
DIREC	- Display a compass rose.
DLINE	- Draw a double horizontal line.
DRAW	- Draw a line to a point.
ERASE	- Erase the screen
HW	- Establish the height and width of the characters.
LTRS	- Display ASCII characters.
MAPTOPIX	- Convert map coordinates to pixels.
MOVR	- Move the current start point to an X,Y point.
PAINT	- Display a canned image from disk.
PIXTOMAP	- Convert pixel coordinates to map coordinates.
PRELEX	- Initialize the LEXIDATA.
RAST	- Display a passed bit image.

RASTER	- Display a downloaded symbol.
XMIT	- Transfer the graphics commands to the graphics device.

4.2.3 ARTBASS Utilities

The following ARTBASS utilities were rewritten to work on the VAX version:

CTOI	- CHARACTER to INTEGER conversion.
DMA2UTM	- Convert DMA map coordinates to UTM coordinates.
EXFACC	- Perkin-Elmer timing utility.
EXFACCP	- Perkin-Elmer timing utility.
EXSACCP	- Perkin-Elmer timing utility.
INTCODE	- 4-digit INTEGER to ASCII conversion.
INTCODE2	- 4-digit, zero fill INTEGER to ASCII conversion.
ITOC	- INTEGER to CHARACTER conversion.
MLTCODE	- 5 or more digit INTEGER to ASCII conversion.
OSDGREQ	- Shared memory utility.
RANDU	- Random number generator.

4.2.4 New Utilities For The VAX Version

The following utilities were written to implement ARTBASS on a VAX:

ARTERR	- General purpose error display routine.
DZ2LX	- Convert DeAnza coordinates to LEXIDATA coordinates.
GSCNCT	- Connect a subprocess to the DeAnza.
LSVCIN	- Initialize link between the VAX and the LSI in the DeAnza.
LSVCRD	- Initiate a read from the LSI in the DeAnza.
LSVCST	- Check the status of a read or write operation to the LSI in the DeAnza.
LSVCWT	- Initiate a write to the LSI in the DeAnza.
PRCSAD	- Get the address and process id of a subprocess.

SECTION 5

SOFTWARE CHANGES NOT COMPATIBLE WITH THE PERKIN-ELMER

This SECTION describes the FORTRAN source code changes made to implement ARTBASS on the VAX that will not successfully transfer to the Perkin-Elmer. In other words, if these routines are transferred to the Perkin-Elmer as is , they will have to be modified on the Perkin-Elmer. The SECTION is divided into three sections, corresponding to the three computers in ARTBASS (MAIN, MAP and CIP). Within each section the programs are listed alpabetically and the subroutines within each program are listed alphabetically.

5.1 MAIN PROCESSOR

5.1.1 AIRMOV2

MAIN - OPEN statements not required for the VAX version were deleted.

5.1.2 ALIBPROG

ALIBPROG - An OPEN statement for unit 2 with "dispose=delete" was added thereby eliminating any affects on the data file EXEC.DAT when a write was issued to unit 2.

5.1.3 ALRTOFF

ALRTLIST - 1. The SYSIO call to read was replaced with a call to a SAIC FSUTY subroutine, RVBWT.

2. Code was added to open the alert save file (ALERTS.DAT) and the alert template file (ALTTXT.DAT).

3. An OPEN to logical unit 9 was inserted for the VAX printer to print all WRITE statements to printer after being loaded in ALTLST.LIS.

4. The SAIC utility MOVE was used to strip off blanks in the time variable, ITIMELOC, instead of shifting bytewise and checking for blanks.

ALRTMAIN - OPEN statements not required for the VAX version were deleted.OP

FORMMESS - 1. The subroutine LWRITE is called with 4 bytes in order to obtain the space character found in a Hex string.

2. The variable KAR is not shifted.

MWRITE - The SYSIO call was replaced with a FORTRAN WRITE so that alerts are written to the printer bytewise.

5.1.4 CASMAIN

CASMAIN - Open statements not required for the VAX version were commented out.

5.1.5 CASREP

CASREP - The SYSIO call to read was replaced with a FORTRAN direct read to read the ammunition resupply database.

CASMAIN - The OPEN statement for the file PRTCASREP was changed.

5.1.6 CCMDAFG

CCMDAFG - 1. The OPEN statements which were commented out in MAIN for CCM.DAT and CCMWORK.DAT were inserted.

2. The SYSIO calls to read and write were replaced with RVBWT and WVBWT.

MAIN - OPEN statements not required for the VAX version were deleted.

5.1.7 CCMRESTR

CCMRESTR - The SYSIO calls to read or write were replaced with FSUTY utilities RVBWT and WVBWT.

CMCHNGRD - The SYSIO call was replaced with a FORTRAN read.

MAIN - The Perkin-Elmer OPENs of the CCM transactions and CCM database were replaced with a FORTRAN OPEN and a FSUTY OPN respectively.

5.1.8 CCMXEC

MAIN - The Perkin-Elmer OPENs of the CCM transactions and CCM database were replaced with a FORTRAN OPEN and a FSUTY OPN respectively.

5.1.9 CCTERM

CCTERM - 1. Ramalerts were sent to one console rather than five.

2. The call to OSL8MON was deleted because it is not required for normal termination on the VAX version.

5.1.10 CLEANUP

CLEANUP - Ramalerts were sent to one console rather than five.

5.1.11 DETECT

MAIN - Open statements not required for the VAX version were deleted.

5.1.12 EVENTPP

EVENTPP - 1. Code was added to open the foreground events file (FEFILE.DAT) using FSUTY utility OPN.

2. The SYSIO to read the events database was replaced with the FSUTY utility RVBWT.

EVNTMAIN - Code was added to open the output print file.

5.1.13 EXEC

EXEC - Since EXTIMER is now a separate process, the call to EXTIMER is commented out and the EXTERNAL statement for EXTIMTRP is no longer needed.

EXIODATA - Code was inserted to open and close EXEC data file (EXEC.DAT).

EXMODE - The arguments in the CALL WAIT are changed. In the load loop and the QUEUE loop, calls to WAIT were inserted.

EXNXTTSK - The call of QUEUE(EXECID,,) was changed to QUEUE('0',,) for a correct self- directed queue trap.

EXREMOVE - 1. Code was inserted to exit after the first call to ABORT.

2. The arguments in the CALL WAIT are changed. A "GOTO 9990" statement was added after the process is dormant.

EXSIMCTL - 1. The HOLD of process SPARE was commented out because SPARE is not used in the VAX version.

2. The call to RELSE was deleted.

EXTIMER - A routine has been written to call RELSE (to restart the timer).

EXTIMOFF - A routine to call HOLD was written to suspend time for a while.

EXTIMTRP - The call QUEUE(0,,) was changed to QUEUE('0',,) for correct execution on the VAX.

EXTIMSTP - The call QUEUE(EXECID,,) was changed to QUEUE('0',,) to correct the self-directed queue trap.

5.1.14 FGEVENT

FEQT - The SYSIO call to write was replaced with a call to the FSUTY utility WVBWT.

FGEVENT - Using the FSUTY utility, OPN, code was inserted to open the event database (EVENTS.DAT).

5.1.15 FORSIGX

ALNUALRT - The SYSIO write of alerts to a save file was deleted. The write was moved to the CIP executive alert processor (EXALTBUF).

EVENTS - The SYSIO call to read was replaced with FSUTY utility RVBWT.

EVMAIN - The open statement for the file (AMMORSPL) was changed.

FORSIGX - 1. The SYSIO call was replaced with the FSUTY utility RVBWT.

2. The replicate end-of-file check was replaced with a check of IDATA(1).

PEVENT - The SYSIO call to write was replaced with the FSUTY utility WVBWT.

PREPLAN - The SYSIO call to read was replaced with the FSUTY utility RVBWT.

RESUPPLY - The SYSIO call to write to the ammunition resupply file was replaced with a FORTRAN direct access write.

REVENT - The SYSIO call to read was replaced with the FSUTY utility RVBWT.

5.1.16 FREEZE

FREEZE1 - The call to HOLD(0,status) was changed to HOLD('0',status).

5.1.17 INIT1

INIT1 - The queue call to OSL8MON was commented out for the VAX version.

5.1.18 INIT2

I2MAIN - The OPEN statement and its associated error checking was deleted.

5.1.19 INPUT

ASRINP - The format statement 3000 was changed so that the two F4 descripters were changed to F4.0.

FINISHUP - 1. The I/O of events file (EVENTS.DAT) was changed to use FSUTY utilities.

2. TMP1IUDT and TMP2IUDT were added to namelist PSEVENT. TMP1IUDT is Character*20 and TMP2IUDT is Character*56 and they are equivalenced to IVDT(3) and (9) respectively.

3. All OPEN statements were changed to include STATUS = 'UNKNOWN'.

IONML - IRBFLAG was replaced with TMPRBFLG in namelist, TMPRBFLG being dimensioned to Character*4, and, TMPRBFLG and IRBFLAG were equivalenced.

INPTMAIN - All OPEN statements were changed to include STATUS = 'UNKNOWN'.

INPUT - 1. The call to OPEN was deleted.

2. A subroutine RADLPT to read the alert pointers was added.

PPLANINP - The I/O of preplanned events file (PBIG.DAT) was changed to use FSUTY utilities and to compute the record number accordingly.

RDALPT - The change to read the three alert files and close them was incorporated.

READECKS - All OPEN statements were changed to include STATUS = 'UNKNOWN'.

TALUTINP - A call to LIB$SIGNAL for error handling was added.

TMLUTINP - A declaration of Char*4 TEMPM2FT was added and then TEMPM2FT was equivalanced with MTRSFEET.

5.1.20 LIBRARY (MNLIB)

ALERTGEN - 1. The dimension of array INBUF was changed to 1 to correct the error received in the VAX when ILENGTH is zero.

2. The SYSIO call to write the alerts to the alert save file was commented out. The alerts are now written by the CIP executive alert processor (EXALTBUF).

3. Because the whole system is in one computer, the queue call to OSL8MON was changed to queue the CIP executive directly (EXECE3).

GETVAL - The left justification of indefinite field integers was changed by replacing IOR, IAND, and ISHFT with a different algorithm for determining the length.

KNTCHAR - The ISC function was replaced with ILBYTE.

RAMALERT - Because the whole system is in one computer on the VAX, the queue call to OSL8MON was changed to queue the map executive (EXECMP) directly.

RANDU - An algorithm was replaced with a call to RAN(I) a VAX utility.

UNITINFO - 1. The UTM location variable, LET, is loaded without IORing with '20200000'X.

2. Time values are loaded by calculating time as "NDAYE * 10000 + NHOUR * 100 + NTIME".

5.1.21 LOSINP

AIRPOINT - The SYSIO to read was changed to two RVBWT reads.

INTELEU - The SYSIO read was changed to two RVBWT reads.

LOSCOMP - 1. Because of the difference of the byte order in the Perkin-Elmer and the VAX, the shift of variables IBLOS and IBLOSX was changed from 8 to -8.

2. The SYSIO read was changed to two RVBWT reads.

3. Line 999 contained an IF statement which caused a 'no path to this statement' error. The IF statement was changed.

4. The IF statement in DO LOOP 10 was changed to read ".LE." rather than ".LT.".

LOSMAIN - The open statements and their associated error checks were commented out.

5.1.22 MSC10

MAIN - Open statements not required for the VAX version were commented out.

5.1.23 MOVMNT

MOVE - The subroutine MOVE was changed to subroutine MOVE1.

MOVMNT - 1. The data statement TASK, dimensioned at 10, contained 11 elements. The last element, ".08" was deleted from the list.

2. The two calls to MOVE were changed to MOVE1.

5.1.24 PGSEXEC

PGSEXEC - 1. Code to open the output print file was added.

2. Since the MAIN subroutine of SAVE originally opened all the necessary files used in PGSEXEC, code was added to open the data files RPFILE.DAT, RSFILE.DAT, and DLFILE.DAT for the subroutines PGSERP, PGSSRP, PGSERS, PGSSRS, PGSEDEL, and PGSSDEL and closed these same files after the subroutines were finished.

3. The SYSIO read was replaced with a RVBWT.

PGSERP - 1. The SYSIO read was replaced with a RVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

PGSSRP - 1. The SYSIO read was replaced with a RVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

PGSERS - 1. The SYSIO read was replaced with a RVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

PGSSRS - 1. The SYSIO read was replaced with a RVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

PGSEDEL - 1. The SYSIO read was replaced with a RVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

PGSSDEL - 1. The SYSIO read was replaced with a RVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

5.1.25 RCOVERY

RCOVERY - 1. Since the MAIN subroutine of SAVE originally opened all the necessary files for RCOVERY, code was inserted to open and close the data files RPFILE.DAT, RSFILE.DAT, CMFILE.DAT, FEREPL.DAT, BLFILE.DAT.

2. All SYSIOs were changed to RVBWT for each data file.

3. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

4. Since the entire system is on one computer, no queue is made to OS8LIMID.

5. A Ramalert was sent to only one console instead of five.

5.1.26 REDIST

MAIN - Open statements not required for the VAX version were deleted.

5.1.27 REPLAY

MAIN - Code was inserted to open files RPFILE.DAT and CMFILE.DAT.

REPLAY1 - 1. SYSIOs were replaced with RVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

5.1.28 RM

ALOGMAIN- The Perkin-Elmer OPEN was changed to correct the format for the VAX.

5.1.29 RPTERM

RPTERM1 - 1. Since the MAIN subroutine of SAVE originally opened all the necessary files used by RPTERM, code was inserted to open and close the data files RPFILE.DAT and CMFILE.DAT.

2. SYSIOs were replaced with RVBWT for each file.

3. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

4. The conversion of model time from binary to ASCII was changed using variables for time from the GRPS symbol dictionary.

5. Ramalerts were sent to only one console instead of all five.

5.1.30 RPVALID

RPVALID - 1. Ramalerts were sent to only one console instead of all five.

2. The conversion of model time from binary to ASCII was changed to use variables containing the proper time wanted.

5.1.31 RSTART

RSTART - 1. Since the MAIN subroutine of SAVE originally handled the opening and closing of the files needed by RSTART, code was inserted to do so for RPFILE.DAT, RSFILE.DAT, CMFILE.DAT, and

FEREPL.DAT.

2. SYSIOs were replaced with RVBWT for each file.

3. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

4. Ramalerts were sent to only one console instead of all five.

5.1.32 SAVE

MAIN - 1. FORTRAN OPEN's were replaced with FSUTY OPN's.

2. Only those files needed for SAVE are opened instead of all files needed for programs SAVE, REPLAY, RCOVERY, RSTART, RPTERM, START, PGSEXEC, and RPVALID. The files opened are: RPFILE.DAT, RSFILE.DAT, CMFILE.DAT, and DLFILE.DAT.

SAVE - 1. SYSIOs were changed to WVBWT.

2. Since the Perkin-Elmer read records of length of 256 bytes, and the FSUTY utilities for reading/writing read records of length 512, the computation of the record number had to be changed.

3. The conversion of the model time from binary to ASCII was changed by moving the wanted time variable into an I*4 variable before converting via INTCODE2 as the BINASC used in the rewritten VAX version of INTCODE2 cannot convert a variable into itself.

5.1.33 SCENARIO DATABASE

N4C26.DAT - The aircraft type input was edited since the hex values were in the wrong byte order for the VAX.

5.1.34 START

START - 1. Since the subroutine MAIN of SAVE originally opened and closed the data file, BLFILE.DAT, code was inserted to do so in this subroutine.

2. The SYSIO was changed to WVBWT.

5.1.35 STATRPT

IOAMMO - The calls to SYSIO were replaced with a FORTRAN read.

QT - The OPEN statement opening the ammunition resupply database was deleted.

STATMAIN - 1. Open statements not required for the VAX version were deleted.

2. The OPEN statements were changed to the correct VAX format.

TLINE - 1. The dimension of the text buffer was changed to 34*N.

2. The write statement to print was changed to print only the first 132 characters of each line.

3. The line clear loop was changed to clear all lines in the buffer.

5.1.36 TIMER

EXTIMTRP - 1. The arguments to the calls to QUEUE were changed.

2. The status error checking was changed to reflect the above change.

5.1.37 TMOVLY

TMOVLY - Because the whole system is in one computer on the VAX, the queue call to OSL8MON was changed to queue the map executive (EXECMP) directly.

5.1.38 WETHR

WETHR - Open statements not required for the VAX version were deleted.

5.2 MAP PROCESSOR

5.2.1 EXEC

EXEOTQT - The clearing of the menu up flag (EXMENUP) was deleted. The clearing is now done in the new map to LSI process (MP2LSI).

EXMENUS - Code was added to queue process MP2LSI to instruct the LSI to turn the cursor on.

EXQTRAP - The number of consoles loop was changed from eight to one because only one console is installed in the VAX version.

EX10GD - Code was added to clear a bit in variable EX10GDX when it is found set.

EX10MENU - Code was added to clear a bit in variable EX10MENUX when it is found set.

5.2.2 GDN

DRAWCONT - NHUGEN = '7FFFFFFF'X was changed to NHUGEN = '00000FFF'X.

MAIN - A call to subroutine GSCNCT was added to connect process GDN to the DeAnza.

5.2.3 MENUN

ADDAR - The time page is displayed when TIME has received a pen-down, a name has been chosen, and there are more than two points chosen. If the type is a Minefield, there must be more than 3 points chosen. If all of the conditions have been met, the time page is displayed and the map is erased.

ADDLN - When the TIME area is selected, the line name must be chosen and the line points must be greater than 1, in order for the time page to be displayed.

ADDPT - When the TIME area is selected, a name must have been chosen and a point selected in order to allow the map to be erased and the time page displayed.

AIRDEFD - The time page is displayed when UPTIME is called.

ALAIR2EX - 1. In a calculation of MTIME, the VAX version does not add 1000.

2. Due to the relocation of the hover time area, if a pen-down occurs in the hover time area, the correct actions are taken. If the hover time area is not changed when it is displayed, then blank out the hover time area. After a change in the hover time area, blank out the hover time area and set the appropriate flags.

3. If the time area is chosen and a "POINT DONE" message is displayed then call UPTIME to display the time page.

4. The Y pixel limit for the bottom of the 'point chosen' list is changed so as not to include the TIME area. The X pixel limit is increased 5 pixels for the Target Selection initials.

ALDEFEX - The use of 1R blank was changed to '2D'X thereby displaying dashes in the menu instead of blanks.

ALERTEX - 1. When the TIME area is chosen, the time page is displayed through a call to UPTIME.

2. The GOTO 200 in the Perkin-Elmer version is commented out because it is not needed for the VAX.

ALGRNDEX - When the TIME area is selected and a destination and a route are chosen, the route will be erased from the map and the TIME page will be displayed.

ALPNTS - The display of the current point number was changed to allow for the display of a number larger than nine.

ALSUBEX - 1. A flag IFLAGDP is added and set to 1 when the drop point is chosen on the map.

2. The conversion of UTM coordinates algorithm was changed to one using MOVE and BINASC due to IORing, IANDing, and ISHFTing.

ALUPAIR1 - The display of the current point number was changed to allow for the display of a number larger than nine.

ALUPLOAD - A page heading display was commented out as it was using pixels 0-90 and the information being diplayed could be found elsewhere on the menu.

ALWRTOT - Since the whole system is on one computer, the QUEUE to O8SLMONN was changed to FGEVENT.

APG2HAND - 1. In a calculation of MTIME, the VAX version does not add 1000.

2. Due to the relocation of the hover time area, if a pen-down occurs in the hover time area, the correct actions are taken. If the hover time area is not changed when it is displayed, then blank out the hover time area. After a change in the hover time area, call MNTIME2 for point selections then erase hover time area.

3. If the time area is chosen and a "POINT DONE" message is displayed then call UPTIME to display the time page.

4. The X pixel limit is increased 5 pixels for the flight type mode.

5. If an ordnance is chosen and the TIME area is chosen, a "POINT DONE" message is displayed and the points on the map are erased and UPTIME is called to display the time page.

6. An ordnance must be chosen to process a DONE/REPEAT.

ARSBHAND - The DONE/REPEAT will not process until an ordnance has been selected.

DELAREA - When the menu is exited and an obstacle or minefield is displayed, the cross will be erased.

DELLINE - When TIME is selected, the type of control measure must be chosen and the number of lines must be greater than 0 in order to erase the map and display the time page.

DELPNT - When the TIME area is selected, the type of point must have been chosen in order to allow the map to be erased and the time page displayed.

DISRAMAL - 1. Code was added to wait on flag PPIBSY to be cleared by subroutine RADQ in process RAMQP indicating a RAMALERT is ready to be displayed.

2. The "1R:" was changed to "1H:".

EXECCC - 1. To improve the appearance of the menu display, code was added to black out the menu area before displaying the menu.

2. Because of the 512 pixel limit on the DeAnza, the call to UPTIME was commented out.

FIREDUR - The IAND was removed from an IF statement concerning the variable IFIRET.

FIREXEC - When the TIME area is selected and the target is selected on the map, the target coordinates are set, the map is erased, and the time page is displayed.

MAIN - A call to subroutine GSCNCT was added to connect process MENUN to the DeAnza.

MANEUVER - If the TIME area is selected and the destination type is chosen then:

a. If the destination type is an XY(1) point or unit(2) and the point is chosen, the point is erased and the time page is displayed.

b. If the destination type is an old route(3) or a special route(4) and the line route is chosen, the line is erased and the time page is displayed.

MANEUVER2 - If the TIME area is selected and if one of the first three OP types is chosen, the time page is displayed.

MISSPICK - 1. The VAX will not recognize _ (underscores) so they were changed to zeros and dashes.

2. The 'STD LD' was moved 1 X pixel.

MKCODE - 1. The TIME area appears above the DONE area when the DONE is displayed.

2. All "1R " were changed to "1H ".

MKCODEAR - All "1R " were changed to "1H ".

MKCODEPT - All "1R " were changed to "1H ".

MNTIME - The X coordinates have been shifted 285 pixels to the right due to the change in the format of the time page display.

MOVEAR - If the TIME area is chosen, more than 2 points must have been chosen on the map and the type of control measure must be defined in order to allow the map to be erased and the time page displayed.

MOVELN - When the TIME area is selected, the line type must have been chosen and the number of lines must be greater than 0 in order to allow the time page to be displayed.

MOVEPT - When the TIME area is selected, the type of point must be chosen, and the number of points must be greater than 0 in order to allow the point to be erased and the time page displayed.

PG1EXEC - 1. The Perkin-Elmer use of 1 hollerith blank character was changed to use a variable which was loaded with a hex data statement of '2D'X.

2. The use of IAND was unnecessary for the VAX.

PSTGMEX - When the TIME area is chosen, the report type and the units must be chosen for unit types 2, 3, and 5 in order for the time page to be displayed.

SIMHANDL - Because the whole system is contained in one computer, the queue call to OSL8MON was changed to EXEC.

SPTFAMT - The VAX version does not require the use of IAND in the calculations of IFIRET.

SPTFEXEC - When the TIME area is chosen:
a. If the percent of fire and the map location are chosen, then the location point is erased from the map and the time page is displayed.
b. If a cease fire or cancel fire is chosen, then the time page is displayed.

UNDEACTI - 1. When the TIME area is selected and a change has not been made in unit selection and a slot has not been selected, display the time page.

2. The Y coordinate for the bottom of the activation area was moved from 375 to 365.

3. The Y coordinate for the bottom of the deactivation area was moved from 415 to 395.

UNDEACUP - The activate units and deactivate units selection areas were made smaller to allow for the TIME area.

UNITRELO - 1. When TIME is chosen, the location marker is erased from the map and the time page is displayed.

2. The Fortran line 'GOTO 200' was deleted for the VAX version.

UPPG2NP - All "1R " were changed to "1H ".

UPPG2NL - All "1R " were changed to "1H ".

UPPSTGM - The VAX version calls UPTIME3 for a separate time page. The TIME selection is not displayed on the main menu.

UPSPTFUN - FORTRAN code was added to the VAX version to check for a subscript of zero. The code is contained in two IF statements: one checks for NETU(NEXT)=0 and the second checks for NETU(LPOINT)=0.

UPTIME - The time clock now appears on its own page; therefore, the X coordinates have been shifted 285 pixels to the right.

UPTIME3 - This subroutine was created to produce the time clock page for the Post Game Summary menu module.

WRITEOUT - Because the whole system is contained in one computer, the queue call to OSL8MON was changed to FGEVENT.

5.2.4 RAMQP

RADQ - Code was added to clear flag PPIBSY which tells subroutine DISRAMAL in process MENUN that a RAMALERT is loaded into common and ready for display.

RAQTRAP - When RAMALERTS are received from the model, they can not be displayed if a menu or another RAMALERT is displayed. Process MP2LSI checks if a RAMALERT is waiting to be displayed when a menu is completed. Therefore, a queue call to MP2LSI was added to initiate processing of a RAMALERT.

5.2.5 STATRPTN

AIRFIL - The call to INTCODE of "ISPEED" was changed to BINASC with only three digits, thereby allowing subroutine ARHDUP to load this same variable without shifting before calling LOADZ.

ARHDUP - The shifting of variable, ISPEED, was deleted before the call to LOADZ.

BLKDATSS - The data statements AIRFORM, GRDFORM, IEQFIL, RMAIR, and RMGRD were rewritten for the VAX because of the bit and byte storage difference between the VAX and P/E for data being stored in hex.

SPSTEX - 1. In the VAX version, the MAP and CIP processes are in the same computer; therefore, the CIP processes can be queued directly instead of through the processor to processor interface (PPI). The call QUEUE was changed from OSPPMON to EXECE3.

2. The call HOLD(0) was changed to HOLD('0').

5.3 CIP PROCESSOR

5.3.1 ALERTQ

ALRTINIT - The alert buffer data file, ALERTQ.DAT is opened to write alerts via WRTFILE.

5.3.2 ALERTSN

ACCUNIT - 1. In the Perkin-Elmer version, CIP data required by the MAP processor is passed through a 10HZ buffer. In the VAX version, the CIP and MAP processes are in the same computer; therefore, the 10HZ buffer is not required and code was added to move the unit name into the MAP common and to queue the MAP executive (EXECMP).

2. The call QUEUE(0,,) was changed to QUEUE('0',,).

ADDSEND - The number of bytes sent via MWRITE to the LSI was changed to multiples of 4.

ALPHALRT - The number of bytes sent via MWRITE to the LSI was changed to multiples of 4.

ATTCHMSS - 1. The call QUEUE(0,,) was changed to QUEUE('0',,).

2. The SYSIO to read from the alert data file,ALERTQ.DAT was replaced with RVBWT.

BLKREAD - The call QUEUE(0,,) was changed to QUEUE('0',,).

COCINIT - 1. To assign the process, ALERTSN, to the DeAnza, the global parameter include file (GPARMS) had to be inserted.

2. The call to LSVCIN was inserted to assign process ALERTSN to the DeAnza.

3. The SYSIOs were replaced with a FSUTY OPN for alert data files, ALERTQ.DAT and ALTTXT.DAT.

COCQTRAP - In the Perkin-Elmer, the key inputs were read directly by process ALERTSN. In the VAX, the key inputs are read by process FKLSIP and passed to ALERTSN via a queue call. A call to subroutine INSUB was added under case (4) to process the key inputs.

CURADD - Because of the difference in the ordering of bytes in the VAX and Perkin-Elmer, the byte index for the call to ISBYTE is changed to 1.

DROPSCRN - The call QUEUE(0,,) was changed to QUEUE('0',,).

DROPSEG - The call QUEUE(0,,) was changed to QUEUE('0',,).

ESCSEQ - The argument "N" is not shifted.

ESCTAB - Due to the lack of function keys on the VT-100, the "PAGE DROP" function key has multiple functions when in status report mode, by changing the data of variable F1 to '1B190000'X. Thus, when a page drop is requested and a status report is already up, the status report will simply be regenerated.

FORMMESS - 1. The number of bytes sent via MWRITE to the LSI was changed to multiples of 4.

2. The data statement for "NL" was changed to '1F1F1F1F'X instead of '1F000000'X so that VAX could pick up the one character being selected to send via LWRITE which was then called with 4 bytes instead of one.

3. The variable KAR contained only one character so it was not shifted before being sent.

FRSTMESS - 1. The number of bytes sent via MWRITE to the LSI was changed to multiples of 4.

2. The code was changed so that VAX terminal control does not rewrite top line of screen if in scanning mode.

GETINDEX - Subroutine GETINDEX was rewritten to get the task index number in the VAX operating environment.

INSUB - The use of the alerts busy flag (COCBSY) was moved to subroutine COCQTRAP, since the key inputs are being sent via a queue trap in the VAX version.

MAKERATT - The call QUEUE(0,,) was changed to QUEUE('0',,).

MWRITE - Subroutine MWRITE was completely rewritten to send the text to the LSI in the DeAnza for display.

PAGE - MODE is changed to -6 before sending a status buffer to MWRITE for display. This allows the LSI to be notified that no bytes are to be swapped before translating.

PUTUP - 1. The call QUEUE(0,,) was changed to QUEUE('0',,).

2. The SYSIO was replaced with RVBWT to alert data file, ALERTQ.DAT.

RCVDEST - The call QUEUE(0,,) was changed to QUEUE('0',,).

REQUNIT - The reading of the unit name for a status report was changed to a call to LSVCWT to initiate the read from the LSI in the DeAnza.

SAVERATT - The call QUEUE(0,,) was changed to QUEUE('0',,).

SAVESEG - The call QUEUE(0,,) was changed to QUEUE('0',,).

SCANOFF - The call QUEUE(0,,) was changed to QUEUE('0',,).

SCANON - The call QUEUE(0,,) was changed to QUEUE('0',,).

SENDMESR - The call QUEUE(0,,) was changed to QUEUE('0',,).

SENDMESS - The call QUEUE(0,,) was changed to QUEUE('0',,).

SETMESS - The SYSIO was replaced with RVBWT from alert format file,ALTTXT.DAT.

SETRATT - The call QUEUE(0,,) was changed to QUEUE('0',,).

SNDRATT - The call QUEUE(0,,) was changed to QUEUE('0',,).

STATEXEC - 1. The call QUEUE(0,,) was changed to QUEUE('0',,).

2. The number of bytes sent via MWRITE to the LSI was changed to multiples of 4.

3. MODE was changed to -6 before sending the status buffer to MWRITE to display. This is the method used for determining whether the LSI should swap bytes or not before translating to VT-100.

TOTALDRP - The call QUEUE(0,,) was changed to QUEUE('0',,).

VIDCLR - This subroutine is not used for the VAX version by simply inserting a RETURN at the beginning.

VIDRESET - This subroutine is returned prematurely so video attributes are not reset for VAX implementation of terminal support.

5.3.3 EXEC

EXALTBUF - 1. Because the VAX version has the whole system in one computer, the alert queuing processes in the map processor were eliminated. Therefore, the alert is retrieved from the global buffer ALERT.

2. Code was added to clear the alert busy flag (ALTBSY) so the model can continue to generate alerts.

3. Code was added to write the alerts to the alert save file (ALERTS.DAT).

4. The call QUEUE(0,,) was changed to QUEUE('0',,).

5.3.4 GLOBAL

BLKDATP3 - The data statements for NDISABLE and NENABLE were changed to '(' and ')' respectively instead of the hex values for each of these symbols. This was done for sending the correct escape sequence to the LSI whenever the subroutine ESCSEQ was called with either of these.

SECTION 6

SOFTWARE CHANGES COMPATIBLE WITH THE PERKIN-ELMER

This SECTION describes the FORTRAN source code changes made to implement ARTBASS on the VAX that will successfully transfer to the Perkin-Elmer. In other words, if these routines are transferred to the Perkin-Elmer as is , they will execute on the Perkin-Elmer without modification. The SECTION is divided into three sections, corresponding to the three computers in ARTBASS (MAIN, MAP and CIP). Within each section the programs are listed alpabetically and the subroutines within each program are listed alphabetically.

6.1 MAIN PROCESSOR

6.1.1 AIRMOV

AIRMOV - AIQT was loaded with the byte utility,ISBYTE instead of hardcoding with '01000000'X.

6.1.2 AIRMOV2

AIRGRND - To avoid an integer overflow problem, the day in the time computation was changed to INTEGER*4.

GRNDAIRV - To avoid an integer overflow problem, the day in the time computation was changed to INTEGER*4.

6.1.3 ALRTOFF

FORMMESS - The DATA statement for NL was changed to '1F1F1F1F'X instead of '1F000000'X so that the P/E or the VAX can use this variable for IANDing, and the proper byte will be picked up.

6.1.4 CCMXEC

AIRALERT - To avoid an integer overflow problem, the day in the time computation was changed to INTEGER*4.

CCMDMG - Code to retrieve support fire impact points from IXYIM was changed from using ands and shifts because it is done that way everywhere else.

CCMENGRS - The masks and shifts were replaced with halfword utilities to increment the obstacle type in IOBSTOP.

CCMSPT - 1. To avoid an integer overflow problem, the day in the time computation was changed to INTEGER*4.

2. Checks of the bridge damage indicator were inserted before using the indicator as a subscript.

DIRMOV - The ENCODEs and PLINE calls were changed to output one line at a time.

6.1.5 COMPDIAG

COMPDIAG - A line containing the FLOAT function was divided to call FLOAT in two places rather than one.

6.1.6 DETECT

DETECT - 1. To avoid an integer overflow problem, the day in the time computation was changed to INTEGER*4.

2. The ENCODEs and PLINE calls were changed to output one line at a time.

VISUAL - The ENCODEs and PLINE calls were changed to output one line at a time.

6.1.7 EVENTPP

ADMLOG - The shifts, ands, and call ILBYTES were replaced with the halfword utility HALFLD.

SELUNT - The use of masks to get the next unit in the list was replaced with the byte utility ILBYTE.

6.1.8 FORSIGX

AIREVENT - To retrieve the mission type correctly from INVDT, the halfword utility,HALFLD was used.

AMOLD - The shifts used to retrieve the unit number were replaced with the halfword utility (HALFLD).

MINEADD - 1. The ENCODEs and PLINE calls were changed from multiple line calls to one call per output line. Multiple line calls will not work on the VAX.

2. The ands and shifts were replaced with the halfword utility HALFLD to retrieve the old X and Y coordinate for a mine delete.

3. The retrieval and update of array MINEDATA uses the halfword utilities HALFLD and HALFST.

PEVENT - The loading of array INEVNT was changed to use halfword utilities instead of shifts.

6.1.9 GNDFIR

EFFNS - In the unit size computation, the unit width was changed to INTEGER*4 to correct an integer overflow problem.

FIRALO - The use of masks and shifts was replaced with byte utilities to access the target list (ITGTLST) and fire override (IFIROVRD) arrays.

RDSON - The unit size computation was changed to REAL to avoid an integer overflow.

SPTALO - The use of masks and shifts was replaced with byte utilities to access the target list (ITGTLST) and fire override (IFIROVRD) arrays.

SPTFIR - The use of masks and shifts was replaced with byte utilities to access the target list (ITGTLST) and fire override (IFIROVRD) arrays.

6.1.10 INPUT

ASRINP - The FORMAT statement 4000 was changed from F,2X to F10.0,2X.

MKUNLIST - 1. The shifts and ors were replaced with a call to ISBYTE to load array LISTUN.

2. A check of IFWD > IM was added after statement label 40 to exit the loop when the end of data is reached.

SIGINP - The calculation of JTEMP was replaced from using ISBYTE to using addition and shifts.

UNINP - The shifts and ors were replaced with a call to HALFST to load manning levels.

WTRINP - A data statement was added to initialize variable MODFOG to 7 to correct a subscript out-of-range error.

6.1.11 LIBRARY (MNLIB)

FIRSORT - The use of masks and shifts was replaced with byte utilities to load the pointer for impact points into ITGTLST and to access IFIROVRD.

ILLUM - The use of masks and shifts was replaced with byte utilities to access the target list (ITGTLST) and fire override (IFIROVRD) arrays.

MAPTOPIX - Two I*4 variables, XCENTR and YCENTR, were added for FLOAT computations.

PIXTOMAP - If statement symbol replacements included: "<" with ".LT.", ">" with ".GT.", and "!" with ".OR.".

SMOKE - The use of masks and shifts was replaced with byte utilities to load the pointer for impact points into ITGTLST and to access IFIROVRD.

WETHRC - "I*2 NDAYE" was moved into "I*4 N4DAYE" for the LINE(1) calculation.

6.1.12 LOSINP

LOSCOMP - All references to array LOSIND to use byte and halfword utilities to load and access LOS check flag and block number.

LOSINP - 1. The anding and oring of array LOSIND was replaced with byte utility ISBYTE.

2. The use of masks and shifts was replaced with byte utilities to load the pointer for impact points into ITGTLST and to access IFIROVRD.

LOSVEG - The shift to retrieve the vegetation type was replaced with the halfword retrieval utility (HALFLD).

6.1.13 MINEFLDS

MINEFLDS - ENCODEs and call PLINEs were changed to output a single line at a time.

6.1.14 MOVMNT

ARRIVE - ENCODEs and call PLINEs were changed to output a single line at a time.

MOVMNT - ENCODEs and call PLINEs were changed to output a single line at a time.

OBSLDEAY - ENCODEs and call PLINEs were changed to output a single line at a time.

OBSTACLE - ENCODEs and call PLINEs were changed to output a single line at a time.

OVERUN - To avoid an integer overflow problem, the day in the time computation was changed to INTEGER*4.

6.1.15 STATRPT

IOPERS - The shifts and ands to retrieve personnel levels were replaced with the halfword retrieval utility (HALFLD).

6.1.16 STEP

LOWALKT - The retrieval of the obstacle number was changed to use halfword utility (HALFLD).

STEP - 1. The masks and shifts were replaced with halfword utilites HALFLD and HALFST to add people back into EM category if too many personnel were killed.

2. The loading of arrays IDEADVEH and KILRATIO was changed to masks and ors instead of addition to correct an integer overflow problem.

6.1.17 SUPRES

SUPRES - 1. In the unit size computation the unit width was changed to INTEGER*4 to correct an integer overflow problem.

2. The HALFLD and ILBYTE utilities were replaced with shifts so that the values are removed from the variable the way they were read in.

6.2 MAP PROCESSOR

6.2.1 GDN

CM - The shifts and ands were replaced with a call to HALFLD to retrieve the control measure X and Y coordinates.

IMPACF - The use of masks and shifts was replaced with byte utilities to load the pointer for impact points into ITGTLST and to access IFIROVRD.

6.2.2 MENUN

ALAIR2EX - A shift was replaced with a BINASC for MTIME and DTIME.

ALGETVAL - The algorithm of loading a value was changed by loading it directly into the I*4 variable ALGETVAL instead of loading it first into two elements of an I*2 array which is equivalenced with an I*4 variable.

ALPNTS - The multiplication of X and Y coordinates was changed so that I*4 multiplication will take place instead of I*2.

ALSUBEX - The multiplication of X and Y coordinates was changed so that I*4 multiplication will take place instead of I*2.

ALUPAE - The dimension of NAMES(3,80) was changed to NAMES(3,LIMIT) which will allow up to 100 names. APG2HAND - A shift was replaced with a BINASC.

ARSBHAND - 1. Due to an overfill problem, the calculations for IX3 and IY3 were changed from one equation to two equations.

2. The shifts were replaced with calls to BINASC.

DELPNT - 1. If the next page, done, repeat, or ignore is selected, the point is erased.

2. The VAX does not recognize the "1H " specification so this was changed. A data statement was initialized to be '2D'X and then the variable was compared with the data statement rather than 1 Hollerith blank.

FIREDUR - The use of INTCODE was replaced with a call to BINASC for the VAX.

FIREXEC - The function ISHFT was replaced with a call to BINASC before the variable IVAL is used in call LTRS.

MNTIME - The functions ISHFT and INTCODE were replaced with a call to the SAIC utility BINASC.

NAMESEG - The shifts were replaced with calls to BINASC.

PG1EXEC - The shifts were replaced with calls to BINASC.

PG2UPAIR - The shifts were replaced with calls to BINASC.

PNTS - 1. Due to an overfill problem, the calculations for IX3 and IY3 were changed from one equation to two equations.

2. The shifts were replaced with calls to BINASC.

SPTFAMT - The use of the functions ISHFT and INTCODE were replaced by the SAIC utility BINASC.

SPTFDUR - The use of the functions ISHFT and INTCODE were replaced by the SAIC utility BINASC.

SPTFEXEC - 1. The use of the functions ISHFT and INTCODE were replaced by the SAIC utility BINASC.

2. The use of IAND was replaced with a call to ILBYTE.

STDLOAD - The shifts were replaced with calls to BINASC.

TGTNUM - Due to an overfill problem, the calculations for IX3 and IY3 were changed from one equation to two equations.

UNGOOD - The VAX version checks for unit deactivation.

UPINFO - The shifts were replaced with calls to BINASC.

UPNXALR - The IAND utility is replaced with a call to ILBYTE.

UPTIME - The use of ISHFT was replaced with a data statement for the variable NTIME in the call to LTRS for the variable LX.

XTRACT - The Perkin-Elmer version ored a four byte variable into a two byte variable expecting the results to overflow into the next two bytes in memory. In the VAX this procedure caused an interger overflow. To correct the problem, the ored result was stored into an INTEGER*4 that is equivalenced to the INTEGER*2 variable.

6.2.3 STATRPTN

EQUAMUP - The ISHFT in the call to LOADZ was deleted since the variables MENBL and MENCL were loaded in GRNDFIL with use of the HALFLD utility.

GRNDFIL - The shifts and masks used to load variables MENBL, MENCL, and IVEGT were replaced with calls to HALFLD.

LOADZ - The dimension statement "IARAY(LEN)" was changed to "IARAY(1)" thereby eliminating a zero-dimensioned array.

6.3 CIP PROCESSOR

6.3.1 ALERTSN

ESCSEQ - The shift and or statement was changed to utility ISBYTE for storing the escape code.

LIB - 1. Halfwords were loaded and stored via HALFLD and HALFST instead of by bytes via ILBYTE and ISBYTE.

2. The variable JPOINT is calculated as a halfword instead of a byte.

STORE - 1. Halfwords were loaded with HALFST instead of ISBYTE and ISHFT.

2. The variable JPOINT is calculated as a halfword instead of as a byte.

SECTION 7

ARTBASS CORRECTIONS

This SECTION describes the FORTRAN source code changes made to implement ARTBASS on the VAX that corrected logic problems found in the model. The SECTION is divided into three sections, corresponding to the three computers in ARTBASS (MAIN, MAP and CIP). Within each section the programs are listed alpabetically and the subroutines within each program are listed alphabetically.

7.1 MAIN PROCESSOR

7.1.1 AIRMOV

AIRMOV - Due to illogical code, an air mission was not being processed properly. The correction was the move of an "ENDIF" so that when "TEND.GT.ENDSTEP",the statement "AIRBRNSN(IAU)=AIBRN(NRP,IAU)" would not be performed since that NRP would equal zero at that point.

7.1.2 AIRMOV2

ADWAIMCS - The variable ICOL was being used, but it was not being set or saved anyplace due to the fact that it was not in a database. To correct the problem, ICOL was set to IAIRSIDE(IAU1) as it was seen previously done in ADWSETUP.

AIRCAS1 - Array IFIREMIN should be dimensioned to 7. This subroutine was compiled with no check to stop a blow-up, but the array should be changed.

7.1.3 CASREP

CASREP - Code was added to exclude air units in the unit loop to avoid a subscript out of range error.

7.1.4 CCMXEC

CCMAIN - An error was detected in calculating the second number for writing out CCM square changes to CCMCHNG.DAT. Instead of "TEMP2 = AINT(TEMP2)", TEMP2 should equal "AINT(TEMP1)".

CCMOBS - The call to HALFST was replaced with a call to HALFLD to retrieve the obstacle type from array IOBSTOP.

7.1.5 EVENTPP

GRDFIR - The coordinate print format was expanded to six digits.

GRDMNV - The coordinate print format was expanded to six digits.

RELUNT - The coordinate print format was expanded to six digits.

7.1.6 FORSIGX

FORSIGX - The controller number was being loaded into INVDT(CBBUFSIZ) with HALFLD causing the VAX to retrieve the wrong halfword. Since INVDT(CBBUFSIZ) was not being used for anything else, the controller number is loaded into the whole word of INVDT(CBBUFSIZ).

MANEUVER - Array ICCMXBRG is a bit packed array, however in this routine it is indexed as a word array. The problem was corrected by using bit utility BCLR to clear the appropriate bit.

RESUPPLY - When loading a new equipment type into a unit and the unit did not have any equipment, a subscript out of range error occurred at statement label 13. To correct the problem, a check for zero equipment types was added with an appropriate branch to avoid the error.

7.1.7 GNDFIR

WPNFIR - The call to RDSON was changed so it is only called for unit targets.

7.1.8 INPUT

INITIAL - 1. Two data statements were changed so that the number of datum matched the dimensions of the variable. The data statements were SCENFILE and UGSTBL.

2. RSTYP was defined Character*16 yet it needed to be defined Character*16 RSTYP(2).

MKUNLIST - After the line "40 IFWD=IFWD + 1", add the line "IF(IFWD .GT. IM)GOTO 75".

SIGINP - Array ISSLCT should be dimensioned to 200. This subroutine was compiled with no check to stop a blow-up, but the array should be changed.

UTINP - 1. Array IWT should be dimensioned to 70. This subroutine was compiled with no check to stop a blow-up, but the array should be changed.

2. Array IPR should be dimensioned to 70. This subroutine was compiled with no check to stop a blow-up, but the array should be changed.

WTRINP - A data statement MODFOG /7/ was added.

7.1.9 LIBRARY

GROSCHK - The call to PPANGL(...IXY(NII,8)) was changed to PPANGL(...IXY(NII,2)) because IXY is dimensioned (200,2).

7.1.10 LOSINP

AIRPOINT - A MOD was being performed with an I*4 variable and the I*2 variable NLOSBND(2); the type of each argument must be the same. The correction involved moving NLOSBND(2) into the local variable NYBLK and then taking the MOD using NYBLK as one of the arguments.

INITELEV - A MOD was being performed with an I*4 variable and the I*2 variable NLOSBND(2); the type of each argument must be the same. The correction involved moving NLOSBND(2) into the local variable NYBLK and then taking the MOD using NYBLK as one of the arguments.

LOSCOMP - A MOD was being performed with an I*4 variable and the I*2 variable NLOSBND(2); the type of each argument must be the same. The correction involved moving NLOSBND(2) into the local variable NYBLK and then taking the MOD using NYBLK as one of the arguments.

7.1.11 MOVMNT

ARRIVE - Variables LOCI and LOCII were defined as INTEGER*2 and the unit destination was moved into them causing an integer overflow. The variables were expanded to INTEGER*4 in symbol dictionary UM to correct the problem.

ENGRSPT - The last part of subroutine was lost in the conversion (probably) a tape error). To correct the problem, the CATTS code was entered into the subroutine.

USEFUEL - Logic was added to skip an equipment type if the equipment type (KOFEQ) is equal to zero.

7.1.12 RECOVRY

RECOVRY - 1. ARTBASS will not recover if a air mission is active at the recover time because array AIMOF is not saved on the recovery save files. This problem was not corrected in the VAX version.

2. Array NCDDP is required for unit movement on special routes, but it is not saved on the recovery save file. To correct the problem NCDDP was moved from common P1 to RS.

7.1.13 SCENARIO DATABASE

N4C01.DAT - The IEQCOD for a AVLB was changed from zero to five.

N4C03.DAT - The value of RTTABLE(1,1) was changed to zero to correct a blow-up in subroutine VISUAL in process DETECT.

N4C06.DAT - 1. Ammo type 18 was added to unit 1/A/4/172 to correct a subscript out of range error.

2. Unit 2/B/1-8 was entered as having seven ammo types, but only six were entered in the database. To correct the problem, the number of ammo types was changed to six.

3. Unit 3/B/1-8 was entered as having seven ammo types, but only six were entered in the database. To correct the problem, the number of ammo types was changed to six.

N4C29 - 1. $TMNML was added to the top of the file.

2. The first "/*" was replaced with "$<CR>$CNTRDATA".

3. The second "/*" was replaced with "$END".

4. "MTRSFEET = 'MTRS'" was changed to "TEMPM2FT = 'MTRS'".

5. All lines were indented one column.

N4C30 - 1. "$TAIN" was added to the top of the file.

2. The "/*" was replaced with "$END".

3. All lines were indented one column.

N4C23 - 1. "$INPUTNML" was added to the top of the file.

2. "/*" was replaced with "$END".

3. "IRBFLAG = Y'2020524C'" was changed to "TMPRBFLG = ' RL'".

4. All lines were indented one column.

N4PSE - 1. "$PSEVENT" was added to the top of the file.

2. "/*" was replaced with "$<CR>$PSEVENT".

3. The last "/*" was replaced with "$END".

4. All lines were indented one column.

5. IVDT was changed by replacing all ASCII data with zero values and putting the ASCII data in temporary character variables.

7.1.14 STATRPT

IOEQUP - This routine had to be compiled with no check to avoid subscript out of range errors.

IOAMMO - This routine had to be compiled with no check to avoid subscript out of range errors.

7.1.15 STEP

STEP - Logic was added to check for valid multikill indexes. If an index is invalid, the multikill update is skipped.

7.2 MAP PROCESSOR

7.2.1 GDN

ZOBFORT - The misspelling of variable JITYPE was corrected.

7.2.2 MENUN

ALAMTEX - The limit of the ammunition amount was changed from 65000 to 32000 since only two bytes were allowed for storage.

SUPAMT - The limit of the ammunition amount was changed from 65000 to 32000 since only two bytes were allowed for storage.

UPAE - The local equipment arrays were expanded to 100.

7.3 CIP PROCESSOR

There were no corrections made to the CIP software.

SECTION 8

VAX TO PERKIN-ELMER SOURCE TRANSFER

8.1 INTRODUCTION

The concept of having a common model on a VAX requires the capability to develop modules on the VAX ARTBASS and transfer them to the Perkin-Elmer without having to make major modifications to the software. The application software was installed on the VAX basically the same as it was on the Perkin-Elmer; therefore, only minor modifications will have to be made to transfer the code. This SECTION describes the transfer process, the translations done by the computer and the changes that have do be done manually.

8.2 TRANSFER PROCESS

The transfer of source code is accomplished by creating a magnetic tape of the source on the VAX and reading the tape on the Perkin-Elmer. The tape is written in 80 character fixed length ASCII records. The tape is created by a source translator which translates FORTRAN from VAX to Perkin-Elmer. In addition to the FORTRAN source, the updates to the symbol dictionaries can be transferred to the Perkin-Elmer.

When the tape is to be loaded on the Perkin-Elmer, a program will have to be written to read the source tape. After the files have been transferred to the Perkin-Elmer, the symbol dictionaries have to be updated and the command files to compile and link the new module will have to be created. Figure 3 shows a flow diagram of the transfer process.

8.3 SOURCE TRANSLATOR

The source translator on the VAX will convert some differences between VAX FORTRAN and P/E FORTRAN and will flag other items that will have to be changed manually. The items that the translator will convert include the ENCODE statement, HEXIDECIMAL constants and code changed for implementation on the VAX.

The translator converts the ENCODE from the VAX format:

```
ENCODE (len,fmt,dest)
```

to the Perkin-Elmer format:

```
ENCODE (fmt,dest)
```

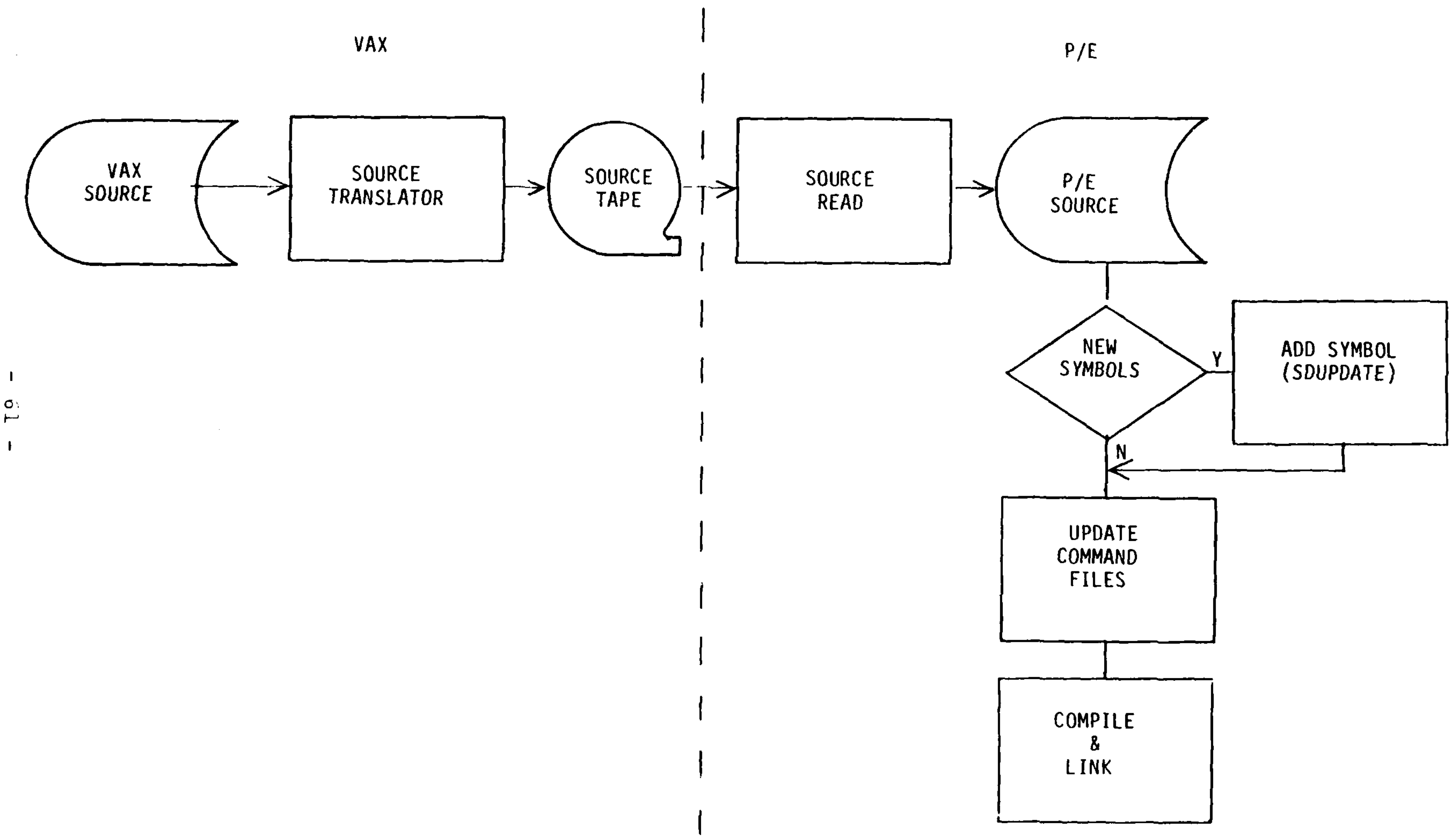


Figure 3. VAX to Perkin-Elmer source transfer.

The translator converts the HEX constants from the VAX format:

```
'F0A4361B'X
```

to the Perkin-Elmer format:

```
X'F0A4361B'
```

The translator will convert changes that were made as part of the implementation onto the VAX. These changes have to be identified with a special code to successfully translate back to the Perkin-Elmer format. The special code consists of a C##VAX to identify the start of VAX unique changes. All Perkin-Elmer code that is commented out must be commented out with a C#. The end of the change section is identified by a C##END. During the translation process all lines found between the C##VAX and the C##END with a C# in column one, will be changed to executable statements. All executable statements between to C##VAX and the C##END will be commented out with a C#. See figure 4 for a sample conversion.

```
BEFORE TRANSLATION             AFTER TRANSLATION

 C##VAX                         C##VAX
 C# P/E CODE                       P/E CODE
 C# P/E CODE                       P/E CODE
    VAX CODE                    C# VAX CODE
    VAX CODE                    C# VAX CODE
 C##END                         C##END
```

Figure 4. Source conversion

In addition to the conversions, the translator will flag some items as code that will have to be manually changed on the Perkin-Elmer. These items are input/output and binary to ASCII utilities and include the following:

Input/output	Binary to ASCII
RVB	ASCBIN
RVBWT	BINASC
WVB	MOVE
WVBWT	
OPN	
CLS	

DISTRIBUTION LIST

DEPARTMENT OF DEFENSE

Armed Forces Staff College
ATTN: Library

Assist to the Sec of Def, Atomic Energy
ATTN: Mil Appl, C. Field
ATTN: R. Wagner

Defense Advanced Rsch Proj Agency
ATTN: TTO

Defense Intell Agency
ATTN: Library
ATTN: RTS-2B

Defense Nuclear Agency
ATTN: NASF
ATTN: NATF
ATTN: NAWE
ATTN: RAAE
ATTN: RAAE, K. Schwartz
ATTN: RAEE
ATTN: RAEV
ATTN: SPSS
ATTN: SPTD
ATTN: STBE
ATTN: STNA
ATTN: STRA
ATTN: STSP
4 cys ATTN: STTI-CA

Defense Tech Info Center
12 cys ATTN: DD

Dep Under Sec of Def
ATTN: S&TNF, T. Jones

Field Command, DNA, Det 2
Lawrence Livermore National Lab
ATTN: FC-1

DNA PACOM Liaison Ofc
ATTN: J. Bartlett

Field Command, Defense Nuclear Agency
ATTN: FCPRW
ATTN: FCTT, W. Summa
ATTN: FCTXE

Interservice Nuc Wpns School
ATTN: Doc Control

Joint Chiefs of Staff
ATTN: J-3, Strat Opns Div
ATTN: J-5, Nuc/Chem Plcy Br, J. Steckler
ATTN: J-5, Nuc Div/Strat Div
ATTN: J-5, Strat Div, W. McClain
ATTN: JAD/SFD
ATTN: JAD/SSD

National Defense University
ATTN: NWCLB-CR

Ofc of the Sec of Def, Net Assessments
ATTN: Doc Control

DEPARTMENT OF DEFENSE (Continued)

Principal Dep Under Sec of Def, Rsch & Engrg
ATTN: J. Wade Jr.

Program Analysis & Evaluation
ATTN: S. Johnson
ATTN: Strat Programs

US European Command
ATTN: ECJ-3
ATTN: ECJ-5

US Natl Mil Representative, SHAPE
Attention US Doc Ofc for
ATTN: Nuc Plans
ATTN: Intel
ATTN: Pol, Nuc Concepts

US Readiness Command
ATTN: J-3

Under Sec of Def for Policy
ATTN: Dir Plng & Requirements, M. Sheridan

Under Secy of Def for Rsch & Engrg
ATTN: K. Hinman

United States Central Command
ATTN: CCJ3-OX, Daigneault

DEPARTMENT OF THE ARMY

Asst Ch of Staff for Intell
ATTN: DAMI-FIT

Chemical Rsch & Dev Ctr
ATTN: SMCCR-OPR

Dep Ch of Staff for Ops & Plans
ATTN: DAMO-NCN
ATTN: DAMO-RQA, Firepower Div
ATTN: DAMO-RQS
ATTN: DAMO-SSM, Pol-Mil Div
ATTN: Tech Advisor
5 cys ATTN: DAMO-NC, Nuc Chem Dir

National Training Ctr
ATTN: TAF-NBC

US Army Armament Rsch Dev & Cmd
ATTN: DRDAR-LCN-E

US Army Ballistic Rsch Lab
ATTN: DRDAR-BLA-S, Tech Lib
ATTN: DRDAR-BLV
ATTN: R. Reisler

US Army Chemical School
ATTN: ATZM-CM-F
ATTN: ATZN-CM-CC
ATTN: ATZN-CM-N

US Army Comd & General Staff College
ATTN: DTAC
3 cys ATTN: Combined Arms Rsch Lib

DEPARTMENT OF THE ARMY (Continued)

US Army Comb Arms Combat Dev Acty
ATTN: ATZL-CAP-DT
ATTN: ATZL-SWN
ATTN: ATZL-SWP
ATTN: ATZL-SWT
ATTN: ATZL-TAS-S

US Army Concepts Analysis Agency
ATTN: CSSA-ADL, Tech Lib

US Army Engineer School
ATTN: Library

US Army Europe & Seventh Army
ATTN: AEAGC-NC-C

US Army Forces Command
ATTN: AF-OPTS
ATTN: AFOP-TN

US Army Foreign Science & Tech Ctr
ATTN: DRXST-SD-1

US Army Infantry Ctr & Sch
ATTN: ATSH-CD-CSO

US Army Intel Threat Analysis Det
ATTN: AIAIT-HI

US Army Intell Ctr & School
ATTN: ATSI-CD-CS

US Army Logistics Ctr
ATTN: ATCL-OOL, S. Cockrell

US Army Material Command
ATTN: DRCDE-D

US Army Materiel Sys Analysis Actvy
ATTN: X5, W3JCAA

US Army Mobility Equip R&D Cmd
ATTN: DRDME-WC, Tech Lib, Vault

US Army Nuclear & Chemical Agency
ATTN: Library
ATTN: MONA-CM
ATTN: MONA-NW
ATTN: MONA-OPS
ATTN: MONA-OPS, B. Thomas
ATTN: MONA-OPS, J. Ratway

US Army TRADOC Sys Analysis Actvy
ATTN: ATAA-TAC
ATTN: ATOR-TDB

US Army Training & Doctrine Comd
ATTN: ATCD-FA
ATTN: ATCD-N
ATTN: ATIC-NC

US Army War College
ATTN: AWCAC, F. Braden, Dept of Tactics
ATTN: Library
ATTN: War Gaming Facility

US Army Comb Arms Opns Rsch Acty
ATTN: ATOR-CAT-T

DEPARTMENT OF THE ARMY (Continued)

USA Military Academy
ATTN: Doc Lib

USA Missile Command
ATTN: DRSMI-RH
ATTN: DRSMI-XF

V Corps
ATTN: G-2
ATTN: G-3

VII Corps
ATTN: G-2
ATTN: G-3

DEPARTMENT OF THE NAVY

Marine Corps
ATTN: Code OTOO-31
ATTN: DCS, P&O, Requirements Div
ATTN: DCS, P&O, Strat Plans Div

Marine Corps Dev & Education Command
ATTN: Commander

Naval Postgraduate School
ATTN: Code 1424, Library

Naval Research Laboratory
ATTN: Code 2627, Tech Lib

Naval War College
ATTN: Code E-11, Tech Svc

Nuclear Weapons Tng Gp, Atlantic
ATTN: Nuclear Warfare Dept

Nuclear Weapons Tng Gp, Pacific
ATTN: Nuc Warfare Dept

DEPARTMENT OF THE AIR FORCE

Air Force Operational Test & Eval Ctr
ATTN: OA

Air University Library
ATTN: AUL-LSE

Assist Ch of Staff, Studies & Analysis
2 cys ATTN: AF/SAMI, Tech Info Div

Dep Ch of Staff, Plans & Opns
ATTN: AFXOOR, Opns, Opnl Spt

Foreign Technology Div
ATTN: SD
ATTN: TQ

DEPARTMENT OF ENERGY AGENCY

Sandia National Laboratories
ATTN: Tech Lib, 3141

DEPARTMENT OF DEFENSE CONTRACTORS

Kaman Tempo
ATTN: C. Anderson
ATTN: DASIAC

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

Science Applications International Corp
ATTN: B. Packard
ATTN: D. Erickson
ATTN: J. Birney
ATTN: J. Ickler
ATTN: J. Martin
ATTN: L. Metzger
ATTN: M. Drake
ATTN: P. McKeown
ATTN: R. Plock

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

Kaman Tempo
ATTN: DASIAC