PATCH XVM UTILITY MANUAL

DEC-XV-UPUMA-A-D

XVM Systems

digital

PATCH XVM
UTILITY MANUAL

DEC-XV-UPUMA-A-D

digital equipment corporation · maynard. massachusetts

First Printing, December 1975

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

The postage prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-10
DECCOMM		TYPESET-11

CONTENTS

FIGURES

TABLES

LIST OF ALL XVM MANUALS

The following is a list of all XVM manuals and their DEC numbers, including the latest version available. Within this manual, other XVM manuals are referenced by title only. Refer to this list for the DEC numbers of these referenced manuals.

BOSS XVM USER'S MANUAL	DEC-XV-OBUAA-A-D
CHAIN XVM/EXECUTE XVM UTILITY MANUAL	DEC-XV-UCHNA-A-D
DDT XVM UTILITY MANUAL	DEC-XV-UDDTA-A-D
EDIT/EDITVP/EDITVT XVM UTILITY MANUAL	DEC-XV-UETUA-A-D
8TRAN XVM UTILITY MANUAL	DEC-XV-UTRNA-A-D
FOCAL XVM LANGUAGE MANUAL	DEC-XV-LFLGA-A-D
FORTRAN IV XVM LANGUAGE MANUAL	DEC-XV-LF4MA-A-D
FORTRAN IV XVM OPERATING ENVIRONMENT MANUAL	DEC-XV-LF4EA-A-D
LINKING LOADER XVM UTILITY MANUAL	DEC-XV-ULLUA-A-D
MAC11 XVM ASSEMBLER LANGUAGE MANUAL	DEC-XV-LMLAA-A-D
MACRO XVM ASSEMBLER LANGUAGE MANUAL	DEC-XV-LMALA-A-D
MTDUMP XVM UTILITY MANUAL	DEC-XV-UMTUA-A-D
PATCH XVM UTILITY MANUAL	DEC-XV-UPUMA-A-D
PIP XVM UTILITY MANUAL	DEC-XV-UPPUA-A-D
SGEN XVM UTILITY MANUAL	DEC-XV-USUTA-A-D
SRCCOM XVM UTILITY MANUAL	DEC-XV-USRCA-A-D
UPDATE XVM UTILITY MANUAL	DEC-XV-UUPDA-A-D
VP15A XVM GRAPHICS SOFTWARE MANUAL	DEC-XV-GVPAA-A-D
VT15 XVM GRAPHICS SOFTWARE MANUAL	DEC-XV-GVTAA-A-D
XVM/DOS KEYBOARD COMMAND GUIDE	DEC-XV-ODKBA-A-D
XVM/DOS READER'S GUIDE AND MASTER INDEX	DEC-XV-ODGIA-A-D
XVM/DOS SYSTEM MANUAL	DEC-XV-ODSAA-A-D
XVM/DOS USERS MANUAL	DEC-XV-ODMAA-A-D
XVM/DOS V1A SYSTEM INSTALLATION GUIDE	DEC-XV-ODSIA-A-D
XVM/RSX SYSTEM MANUAL	DEC-XV-IRSMA-A-D
XVM UNICHANNEL SOFTWARE MANUAL	DEC-XV-XUSMA-A-D

PREFACE

This manual describes the operation and use of the PATCH XVM Utility Program.

NOTE

> PATCH XVM (PATCH) should be used only by the System Manager.

It was assumed in the preparation of this manual that the user is familiar with DIGITAL XVM hardware and operating system. If there are any questions in this area, consult the XVM/DOS User's Manual.

The following is a list of XVM documents which either support directly or contain information useful in understanding PATCH and its functions.

XVM/DOS User's Manual
XVM/DOS Keyboard Command Guide
SGEN XVM Utility Manual
PIP XVM Utility Manual
MACRO XVM Assembler Language Manual
XVM/DOS V1A System Installation Guide

CHAPTER 1
INTRODUCTION

1.1 PATCH PROGRAM

The PATCH Utility program enables the DIGITAL XVM (XVM) user to view and modify System Programs and data on specific device data blocks. The commands provided by PATCH may be entered via the console keyboard or, in the monitor Batch mode, via either paper tape or card reader unit.

This manual assumes that PATCH commands have been entered via the keyboard. The same commands can be entered under Batch mode; see the XVM/DOS User's Manual for details.

1.2 MANUAL ORGANIZATION AND USE

This manual is intended for users who are familiar with:

a) The general operating procedures (i.e., use of equipment and startup procedures), and

b) the elements, structures and use of the particular monitor software system in which PATCH is to be used.

Introductory information and detailed description of each PATCH function and their applications are given in Chapters 2 through 5.

Chapter 6 summarizes the PATCH startup and operating procedures.

New users of PATCH should familiarize themselves with the contents of Chapters 1 through 5; thereafter, they need only refer to Chapter 6 for concise information.

Two Appendices are included which contain supplementary System Block (SYSBLK) and System Program descriptions.

SYSBLK is described in Appendix A, System Programs are discussed in Appendix B.

1.3 MAJOR PATCH FUNCTIONS

The major functions for which PATCH may be used are:

a) the loading of relocatable programs assembled by MACRO XVM into a system as a System Program (described in Chapter 3);

b) the loading of absolute programs assembled by MACRO XVM and data onto a System Program area (described in Chapter 4);

c) the modification of registers located in

   1) System Programs,

   2) Data blocks,

   3) System SYSBLK (system block).

The items on which PATCH functions may be performed together with a description of the commands and procedures needed for their selection are given in Chapter 2.

1.4 PATCH, I/O DATA FLOW PATHS

Figure 1-1 illustrates the primary data flow paths employed during PATCH operations.

User command/response operations are carried out via a console/printer unit. Peripheral devices/UFD's containing the files to be patched must be assigned to .DAT -14; those devices containing auxiliary input files must be assigned to .DAT -10.

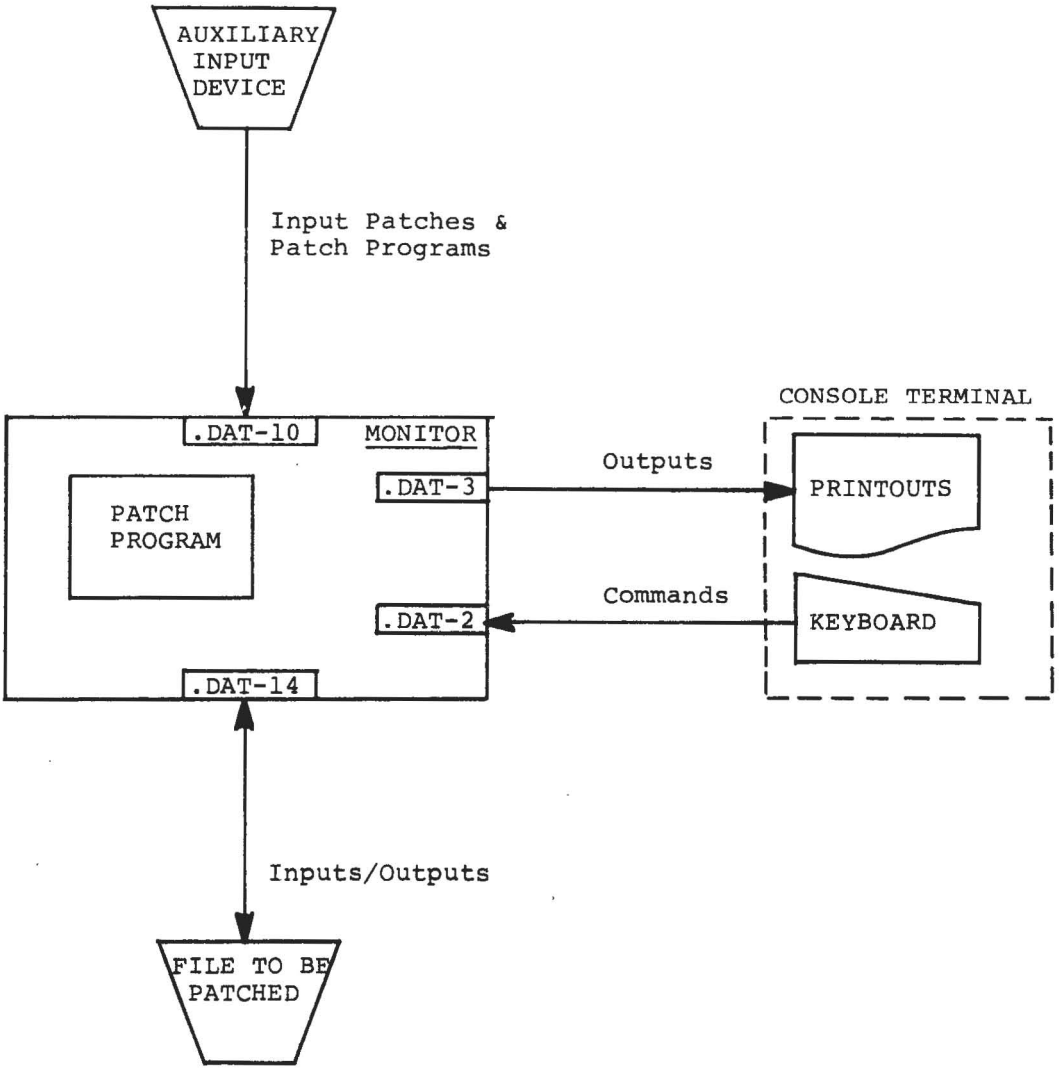


Figure 1-1
PATCH Operations, Information Flow Diagram

CHAPTER 2
SELECTION OF ITEM TO BE PATCHED

All operations explained in this section are done under control of the PATCH program.

2.1 SELECT FUNCTION

The first step required in any PATCH procedure is the selection of the item (.DAT -14) to be patched. Two basic commands "Program Select", and "Block Select" are provided for this purpose.

2.1.1 Program Select Command

The System Programs located on the device assigned to .DAT -14 are selected for PATCH operations by the entry of a unique name which is stored in a monitor table; these unique names enable the monitor to identify the requested program.

The format for a Program Select command is:
(The right angle bracket is the PATCH program user-prompt character.)

```
>name)
```

For example, to select the Editor program the entry

```
>EDIT)
```

is entered. The entry of a System Program name merely identifies the item to be operated upon; no further action is taken until a PATCH function is initiated by another command.

NOTE

The Program Select command may be used only if the device assigned to .DAT -14 contains a System Block (SYSBLK).

In addition to System Programs, system Device System Block (SYSBLK) and CTRL Q (↑QAREA) areas may also be selected for PATCH operations. Table 2-1 lists the unique names of the programs normally supplied as System Programs with the XVM/DOS software system.

Table 2-1
PATCH Accessible System Programs

	PROGRAM	NAME
1)	XVM/DOS Resident Monitor	RESMON
2)	System Loader	.SYSLD
3)	Control Q Area	↑QAREA
4)	Standard Editor Program	EDIT
5)	Text Editor for VT15 Graphics Display	EDITVT
6)	Text Editor for VP15 Storage Tube Display Unit	EDITVP
7)	Peripheral Interchange Program	PIP
8)	Macro-Assembler Program for DIGITAL XVM (MACRO)	MACRO
9)	Cross Reference Program	CREF
10)	System Overlay Build Program	CHAIN
11)	System Generator	SGEN
12)	FORTRAN IV	F4
13)	Core Dump Program	QFILE
14)	General Dump Program	DUMP
15)	System Patch Program	PATCH
16)	Library UPDATE Program	UPDATE
17)	Source Compare Program	SRCCOM
18)	XVM/DOS non-resident MONITOR	DOS15
19)	Macro Assembler Program for PDP-11 MAC11	MAC11[1]
20)	Input/Output SPOOLER Program	SPOOL[1]
21)	Magtape File Manipulation Utility	MTDUMP
22)	Spooler Disk-Area Generator	SPLGEN[1]
23)	Install Spooler as System Program	SPLOAD[1]
24)	BOSS pre-processor	B.PRE
25)	BOSS non-resident monitor	BOSS
26)	PDP-8 to XVM Translator	8TRAN
27)	Install MAC11 as System Program	MCLOAD

[1] UNICHANNEL-15 systems only.

2.1.2 Block Select (B)

A single 400_8-word data block located on the device assigned to .DAT -14 is selected for PATCH operations, using a "B": Block Select command. It is not necessary to select a system program before performing this operation.

FORMATS: a) >B␣n)

or

b) >B+␣n) (DECtape only)

or

c) >B-␣n) (DECtape only)

where:

1) B identifies the Block Select command

2) + indicates the selected block should be read in the forward direction.

3) - indicates that the selected block should be read in the reverse (counter-clockwise) direction.

4) n represents an octal number which identifies a logical block on the PATCH I/O device (.DAT -14). The value of n must be greater than or equal to 0.

DESCRIPTION:

The entry of a "B" select command relocates the identified word block into a PATCH Block Buffer in core. In relocating the block, the load address is set to 0, the size is set to 400_8, and the block number is as specified. If the block number (n) in any of the select commands is followed by a space, any data on the remainder of the input line is treated as a comment and is ignored. For example:

>B␣100␣COMMENT)

is permitted.

The 'L' operation (refer to Section 5.3) is permitted under the "B" command.

CHAPTER 3
LOADING RELOCATABLE PROGRAMS AS SYSTEM PROGRAMS

3.1 PATCH LOAD FUNCTION

A PATCH load function enables the user to convert a relocatable file into an absolute SYS file format and load the converted file onto a system device as a System Program. This section describes in detail the PATCH command used, the procedure required, the relocatable file requirements, and the operations performed by PATCH.

The PATCH load function can be used only when the system on .DAT -14 has been prepared by SGEN to receive the converted file. During SGEN procedures (refer to the SGEN XVM Utility Manual) the user must enter the name of each program to be installed, and specify .DAT slot usage. The program name, number of system blocks required, and .DAT slot usage information entered during SGEN is stored in the system device SYSRLK (see appendix A).

3.2 PATCH READR LOAD COMMAND

The command "READR" initiates the PATCH relocatable-to-SYS file program load function. The general format of this command is as follows:

>READR␣[nnnnn]␣[filename]␣[ext]␣[comments])

where: nnnnn = the highest 13-bit core address to be occupied by the relocated file when it is loaded into core as a SYS file. If an address is not specified, the first free register below the bootstrap is assumed by PATCH. If specified, the value of nnnnn must be $\leq 17636_8$; a larger value will cause an error.

filename = the name of the relocatable file[1].

ext = filename extension; if not given, BIN is assumed.

comments = if desired, a comment may be added to the command string, but only if the extension is given explicitly.

[1]NOTE: Filename may be omitted for nonfile oriented auxiliary devices (.DAT -1Ø).

3.2.1 Procedure

The READR command must be preceded by a program select command which identifies the system device area (named in SGEN) into which the modified file is to be written. For example, the sequence:

```
>8TRAN↵
>READR  8TRAN↵
```

must be entered to install the file 8TRAN onto a prepared system device at .DAT-14, given the name 8TRAN.

On the execution of a READR command, PATCH obtains the named file from .DAT-1Ø and checks it for required size and program characteristics (see paragraph 3.2.2). If the file is not acceptable, the load operation is halted and an appropriate error message is output to the user.

If the file is acceptable, PATCH relocates addresses within the input file, builds a table of the transfer vectors contained by the file, and appends the table and a Bank Bit Initialization routine to the file. The modified file and its appended routine and vector table are then loaded as a unit, in core image form, onto the system device on .DAT-14.

PATCH automatically changes the patch parameters in SYSBLK. Relocation and the Bank Bit Initialization routines are described in paragraphs 3.3 and 3.4, respectively.

3.2.2 Input File Requirements

In order to be installed onto a system as a SYS file, a relocatable file on .DAT -1Ø must:

1) not contain more than 256 transfer vectors
2) not contain external .GLOBL references
3) not use indexed instructions
4) not have a core image size greater than the number of system blocks allocated multiplied by 400_8
5) not cause the total program size to exceed 8K (see the note)
6) not contain COMMON references

NOTE

The total installed size of a program may be computed by adding: 1) the size of the binary program, 2) the number of transfer vectors, and, 3) 32_8 locations which are occupied by the Bank Bit Initialization routine.

3.3 RELOCATION

MACRO outputs relocatable programs addressed as if they were assembled starting in location Ø. Relocatable programs are normally loaded into core for execution by the Linking Loader System Program which, at load time, computes the difference between the MACRO-assigned addresses and the addresses of the actual core area into which the program is to be loaded for execution. This computed difference, called the relocation factor, is then added to each location address contained by the program as it is loaded into memory. The relocation factor properly orients the relocatable program within memory.

3.3.1 PATCH READR File Relocation

During the execution of a READR command, PATCH calculates the required 13-bit relocation factor by subtracting the program size from the final (highest) address the program is to occupy. The final address is either specified in the READR command by the user or, by default, 17636_8 is assumed. For example, the relocation factor (#) for a relocatable program $4ØØ_8$ words in size with no address specified in the READR command would be calculated:

$$\# = 17636_8 - 4ØØ_8 + 1 = 17237_8$$

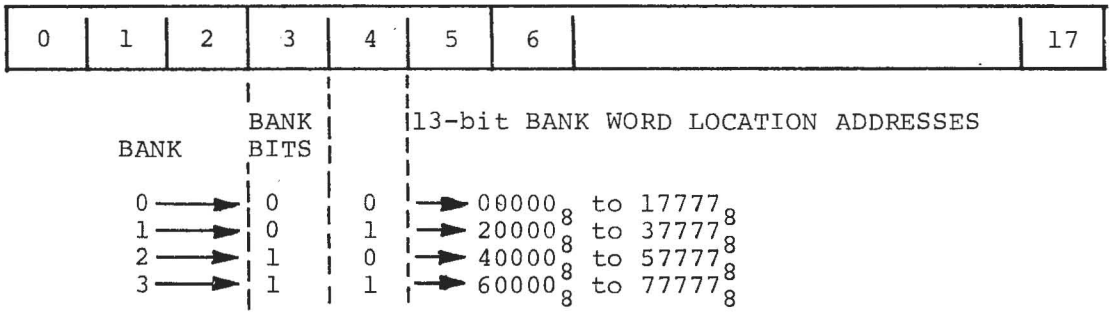
In order to relocate the program addresses (Ø through 377) for conversion to the absolute values required for SYS files, PATCH then adds the relocation factor to each program address, i.e.,

+ Ø = address of first word location (17237_8)
:
+ 377 = address of last program word location (17636_8)

PATCH also relocates the addresses contained by the memory reference instructions and transfer vectors found in the input program.

3.4 BANK BIT INITIALIZATION (BBI) ROUTINE

Transfer vectors contain 15-bit addresses which include bank identity bits and the 13-bit, 8K word location addresses. The format of a Transfer Vector data word and the interpretation of its address portion are:



During READR relocation operations the relocation factor is added to the least significant 13 word bits of the Transfer Vector to relocate the bank word address which it contains. The bank bits of a Transfer Vector cannot be relocated at the time the file is installed as a System Program since the number of the highest bank of a system varies according to the overall size of the system.

To ensure that transfer vector bank bits are relocated prior to run time, PATCH, during READR operations, develops a table of transfer vector locations, adds the table to a Bank Bit Initialization routine, and appends the routine to the input program. The BBI routine is installed onto the system device medium as part of the new System Program.

When a READR-installed program is loaded into core, its associated BBI is also loaded immediately below the program (see Figure 3-1). The loader, .SYSLD, on completion of its functions, turns control over to the BBI routine. The BBI routine determines what bank the System Program has been loaded into and updates the bank bits of all Transfer Vectors contained by the program. The routine identifies

the vector words from the Transfer Vector table developed during PATCH READR operations.

On completion of its initialization function, this routine sets the contents of the .SCOM+3 pointer to the address of the last location occupied by the routine and turns control over to the System Program.

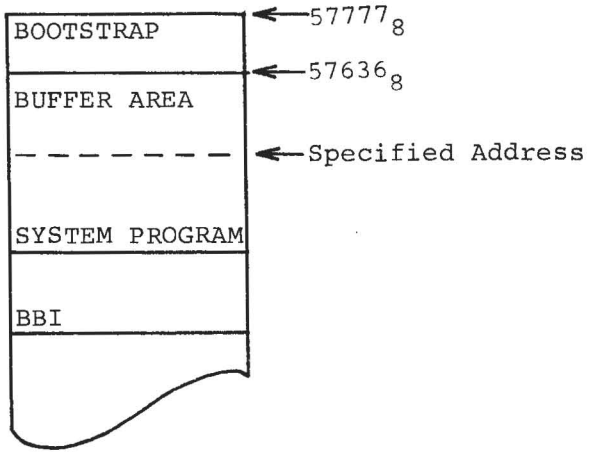
The modification of .SCOM+3 at the end of BBI functions frees the core occupied by this routine for use by the System Program.

The basic portion of the BBI routine requires 32_8 word locations. The routine Transfer Vector table requires an additional location for each Vector in the program. A maximum of 400_8 vectors is permitted.

The PATCH appended BBI routine relieves the user of the responsibility of including a routine to perform this function.

3.5 READR ESTABLISHED SYSTEM COMMUNICATIONS AREA

The ability to specify in the READR command the highest load address for the program being loaded permits a buffer area to be set up between the last program address and the first address of the bootstrap. Such a buffer would provide READR-loaded programs with a common core-resident buffer for intra-program communications and/or common data storage. See below and Figure 3-1.



3.6 LOADING PATCH-RELOCATED CORE IMAGE SYSTEM PROGRAMS

READR-installed Systems Programs and their associated Device Handlers are loaded into core from the system device by the System Loader (.SYSLD) program.

The placement of READR-installed System Programs and their device handlers in core is illustrated in Figure 3-1.

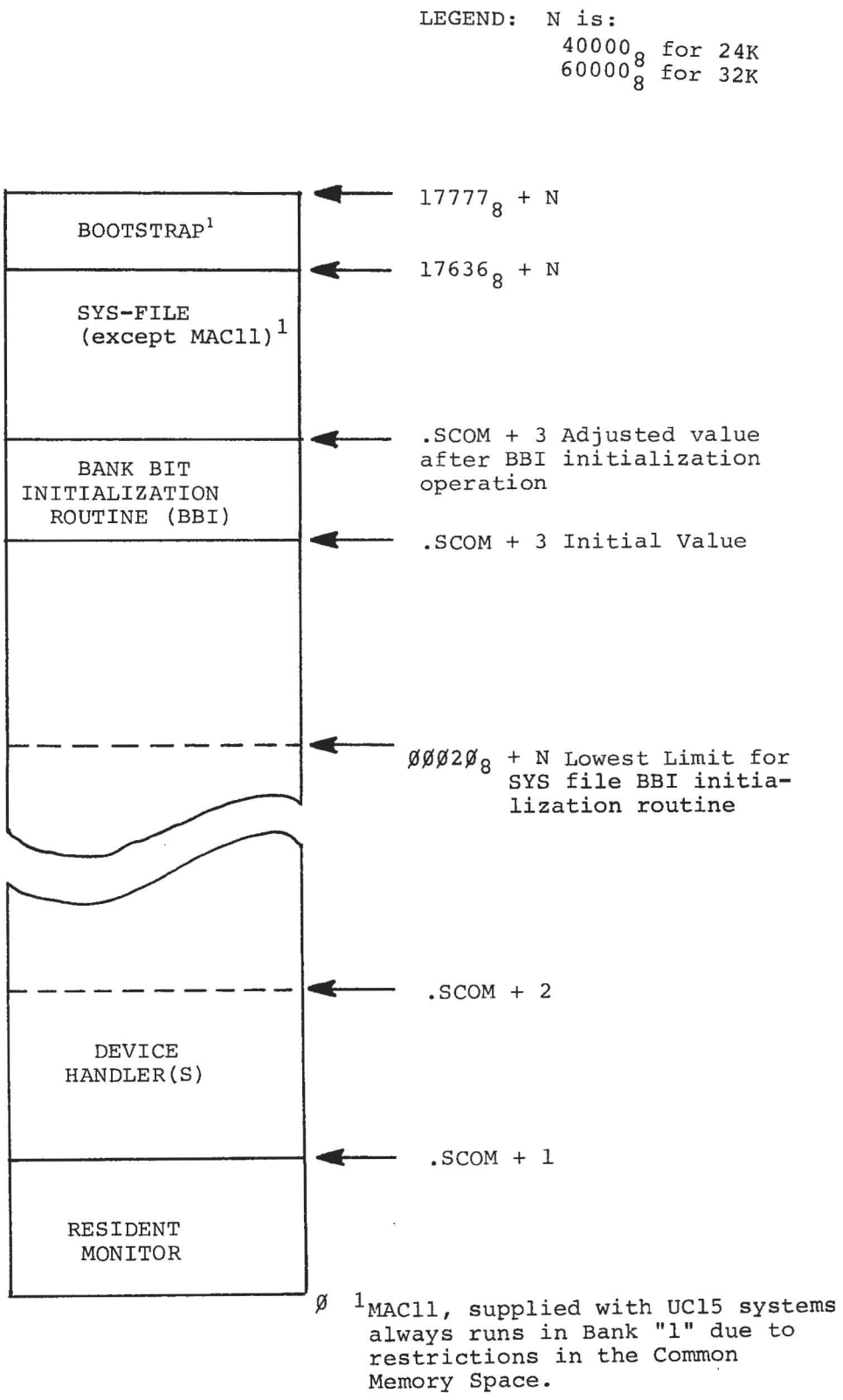


Figure 3-1
Memory Map, Loading a READR-Installed SYS-file

CHAPTER 4
LOADING ABSOLUTE PROGRAMS AND DATA

4.1 READ AND PATCH FILE FUNCTIONS

This section contains detailed descriptions of the READ load function, the function user commands, and the load operation. Patch files, the input sources to PATCH during READ load function, are also described in detail.

4.2 READ LOAD FUNCTION

The "READ" load function enables the user to transfer absolute "Patch files" from the auxiliary input device (.DAT-10) into a user-selected program or data block located on the device on .DAT -14.

A Patch file is an absolute program written by the user to operate with the PATCH READ load function; its purpose is to direct the PATCH program in loading information contained by the file into a specific register or series of registers located within a program or data block of the device on .DAT-14. Entire programs or data blocks, as well as selected items, may be loaded from Patch files onto the device on .DAT -14, using the PATCH READ load function.

Patch files are used as a means of making changes (i.e. patches) to existing SYS files or a device data block or for installing a new version of a System Program onto a system device. They are particularly useful when the same patches are to be made to a SYS file contained by more than one system device.

NOTE

When patching or replacing a System Program, care must be taken to ensure either that the existing SYSBLK parameters are not affected or that they are changed to comply with the characteristics of the replacement version of the program (see paragraph 4.5).

4.3 READ COMMAND

The command READ initiates the performance of the absolute load function. This command may have any of the following formats:

a) >READ↵ (for non-directoried devices only)

b) >READ␣[filename]↵

c >READ␣[filename]␣[ext]␣[comment]↵

where: 1) filename is the name of the Patch file to be input from .DAT-10. The use of a filename extension (format c) is optional. If not given, the extension is assumed to be ABS.

2) Comments may be added to command strings having format c.

3) Patch files on paper tapes do not require a filename (format a).

NOTE

The READ command must be preceded by either a Program Select or Block Select command. The patch file read must be a .ABS macro program.

4.3.1 READ Load Operation

During READ command load operations, PATCH loads into core one 400_8-word block at a time from both the Patch File and the selected program or data block. Once both blocks are loaded, PATCH processes in sequence each word of the Patch File block. Using reference data supplied in the Patch File, PATCH determines the address within the user selected program or data block which corresponds to that addressed by the Patch File word being processed. If the addressed register is within the selected program or data block currently in core, the contents of the Patch File word are loaded into that register by PATCH.

If the addressed register is not in the block currently in core, the block in core is written back into the device at .DAT-14 and the block containing the addressed register is loaded. After the new block is loaded, PATCH locates the addressed register and writes the contents of the Patch File word into it.

A user-selected data block is handled, during READ operations, as if it were an absolute program having addresses ranging from ØØØØØ through $ØØ377_8$. If more than one data block is to be patched, a separate Patch File is required for each block and individual Select Block and READ commands must be issued for each.

If a Patch File word addresses a location not within the limits of the selected program or block, PATCH outputs an ADDRESS OUT OF RANGE error to the user console teleprinter (refer to Section 6 for description of error messages).

4.4 WRITING PATCH PROGRAMS

The following is an example of the proper format for a PATCH assembly source program to be assembled and output by MACRO XVM:

PATCH FILE		MEANING AND OPERATION PERFORMED
.TITLE	anything	
.ABS	NLD	No Loader.
.LOC	234	Set location counter to address 234.
734777		Set contents of 234 to 734777, and go to next location.
6ØØ261		Set contents of 235 to 6ØØ261.
.LOC	35Ø	Advance counter to location 35Ø.
-3		Set contents of 35Ø to -3.
.LOC	371	Advance location counter to address 371.
Ø4Ø674		Write Ø4Ø674 into LOC 371
.		.
.		.
.		.
.END		End program.

The commands available for patching system programs assembled under MAC11 are the 'B' (block) command and the 'L' (location) command. See Section 5.6.

As illustrated in the above example, .LOC statements in the patch program are recognized by PATCH as pointers. These pointers direct PATCH to start a replacement operation at a specified address in the program or data block being patched. In the replacement (overwrite) operation, the contents of the Patch File location immediately following the .LOC statement is written, by PATCH, into the selected program or data block location pointed to by the .LOC address. PATCH then writes, in sequence, the contents of the following Patch File locations into the registers of the file being patched immediately following the one pointed to by the patch file .LOC statement. All patches are written as if the program is to be loaded into the first bank (13 bit addresses). All transfer vectors in a .ABS program must be bank-bit initialized before use.

The current replacement operation continues until another .LOC statement is reached in the input Patch File. The new .LOC statement ends the current replacement operation and directs PATCH to go to a new start location and to start another replacement operation.

The Patch File operations are terminated when PATCH reaches the PATCH File .END statement.

4.4.1 Partial Overlay Patch Programs

A Patch Program written to change the contents of specific locations or groups of contiguous locations throughout a selected program or data block acts as a partial patch to the file being modified; only those items specified in the Patch Program are affected. The Patch Program in the example in paragraph 4.4 represents a partial patch.

Patch programs may be used only to change the contents of registers already existing within a selected program or data block; they cannot be used to insert information between existing program lines.

4.4.2 Replacing Complete Programs or Data Blocks

Patch Programs may be written to replace an entire absolute System Program or 400_8-word data block. To accomplish this, the user sets the first Patch File .LOC statement to point to the first location of the program or data block to be replaced and follows it with the replacement program or data block. For example:

```
.TITLE          (Patch Program name)
.ABS            NLD
.LOC            (First Location of selected program or
  ↓             data block)
Replacement
Program or
Block
  ↓
.END
```

4.5 PRECAUTIONS TO BE OBSERVED FOR PATCH READ OPERATIONS

The PATCH READ load operation does not modify (i.e., update) SYSBLK on completion of the function. If any changes are made to a System Program which affect the parameters stored for it in SYSBLK, the user must

change these parameters using the PATCH List function (described in Section 5). Indexed instructions are not permitted.

When overlaying a complete program with a new version, the user must ensure that the new version will fit into the area allotted to that program during SGEN. All .ABS program transfer vectors must be bank-bit initialized by the program itself before it uses them if the core image is loaded above 8K.

CHAPTER 5
LISTING AND MODIFYING PROGRAM AND DATA BLOCK REGISTERS

5.1 REGISTER MODIFICATION FUNCTIONS

This section describes the functions, commands and procedures which enable the user to select and modify individual registers within:

a) SYS files initially written as absolute programs,
b) READR-installed SYS files,
c) a system device SYSBLK Program Parameter Table, and
d) a system save, ↑QAREA, area.

5.2 PATCH LIST FUNCTION

PATCH register modification is carried out as a list operation in which the address and contents of each register selected by the user are listed (printed) on the user's console teleprinter. On completion of the print operation, the user may enter, if desired, an expression to replace the contents of the currently listed register. When the user is through with the currently selected register, he terminates that operation and either terminates the function or directs PATCH to list the contents and address of another register. The program or data block containing the register(s) to be modified must be identified by the entry of a Select command (see Section 2); the specific register to be viewed and modified (if desired) is selected by the user with a "List" command. Three types of List commands are provided by PATCH:

a) L command, used to select and initiate the list operation for registers within any selected SYS-file, data block, SYSBLK or QAREA contained by the device on .DAT -14.

b) LR command, used only for the selection and listing of registers within READR-installed SYSfiles.

c) SYSBLK Program Parameter Commands, five separate commands, each of which initiates the list operation for a specific SYSBLK parameter for the currently selected SYSfile. Used only with SYS files.

The format and use of the above List commands are described in paragraphs 5-3, 5-4, and 5-5, respectively.

5.2.1 List Operation

The PATCH operations performed during a List operation are:

a) On the entry of a list command, PATCH loads the 400_8 word data block or selected Program block containing the user identified register into core. The selected register is opened by PATCH and its address and contents are printed on the console teleprinter unit.

b) PATCH waits for the entry of a modifier and/or a terminator.

c) PATCH writes the input modifier (if made) into the opened register and examines the line terminator to determine the next operation to be performed. Depending on the terminator entered (see 5.2.3) PATCH terminates the List function or opens another register.

5.2.2 Expressions

Expressions, as defined for MACRO XVM, are strings of symbols and numbers separated by arithmetic or Boolean operators. Octal numbers of from one to six digits (Ø to 777777_8) or alphanumeric symbols of from one to three characters may be used as expressions or as components of an expression.

In PATCH List operations, expressions are entered as replacements for the contents of a program or data block register. Expressions entered to change or modify a data word consist of up to six octal digits, originally preceded by a minus sign; those entered for an instruction word consist of an instruction operation code and an operand. PATCH contains symbol tables for all of the XVM basic instructions op-codes and operate group instructions with the octal values of each. The user can, therefore, patch registers using symbolic rather than octal representations of MACRO XVM instructions (e.g., an expression may be entered as LAC 17536 instead of its octal form, 217536).

The operators recognized by PATCH are listed in Table 5-1.

Table 5-1
Operators Recognized by PATCH

OPERATOR	FUNCTION
-	Two's complement subtraction
!	Inclusive OR
+	Two's complement addition
→| (tab)	Two's complement addition
*	Two's complement addition

The asterisk (*) in addition to its use as an operator, also sets the indirect bit of the expression by causing 20000_8 to be XORed into the value of the expression. The XOR operation occurs each time that the symbol (*) is encountered; two sequential asterisks negate the setting of the indirect bit.

** = 20000 XOR 20000 = 0

The value of a PATCH expression is null (no modification is made to the opened register) unless the expression contains a number, a symbol or an asterisk.

Expressions are evaluated from left to right assuming an initial value of zero followed by a + operator (i.e., 0 + User's expression). Leading and trailing operators are legal in an expression but the latter are ignored. Whenever a string of consecutive operators is used, only the last one in the string is used by PATCH in evaluating the expression.

NOTE

The LAW is a special case in PATCH; it should be used only in the following ways:

a) LAW␣n which is equivalent to 760000+n

b) LAW␣-n which is equivalent to -n.

The use of LAW (LOAD ACCUMULATOR WITH this number) in any manner other than that described above will result in an error.

5.2.3 List Operation, Line Terminator

When all list operations for an opened register are completed, one of four possible Line Terminators is entered to close the register and indicate the next list operation. The line terminators are described in Table 5-2.

Table 5-2
List Operations, Command String Terminators

ENTRY	DESCRIPTION	OPERATION
)	Carriage Return	Open and list the next sequential register
↑)	Up Arrow and Carriage Return	Open and list the register which precedes the current register. Treat any entries after ↑ as a comment.
←)	Back Arrow and Carriage Return	Open and list the contents of the register pointed to by the address portion of the word contained in the currently opened register. Treat any entries after ← as a comment.
(ALT MODE)	ALT MODE	Terminate current list operation; write PATCH buffer block onto .DAT -14 and wait for next entry.

5.3 L LIST COMMAND

The L List command has the following format:

L␣n)

where

a) L initiates the list operation, and

b) n represents the octal form of the 13-bit address of the location to be opened. When the L command is used to open registers of READR-installed SYS files the address n must be the relocated (i.e., address after relocation) value.

Listing and Modifying Program and Data Block Registers

The response to an L command is a printout having the following format:

>address of opened location) / (contents of opened location)>

EXAMPLE:

```
>L  132
␣ØØ132/777435>        (location 132₈; contents 777435₈)
```

After outputting the address and contents of the accessed location, PATCH prints its go-ahead symbol (>) and waits for:

1) the entry of an expression to replace the current contents, to modify the opened location.

   For example:

   >(address)/(current contents)>(expression or new contents)

2) the entry of selected string terminator to indicate the next sequential list operation.

5.3.1 Example: Select, List and Modification Procedures

The following illustrates the use of PATCH select and list commands in the modification of a hypothetical system program named JOVE.

PROCEDURE	DESCRIPTION
>JOVE	Select System Program JOVE.
>L 1ØØ ↵	Initiate List operation starting at location 1ØØ
>ØØ1ØØ/777435> ↵	PATCH prints address and contents of opened location; user enters ↵ to go to and open next sequential location.
>ØØ1Ø1/6ØØ2ØØ>213775 ↵	Address and contents of location 1Ø1 are printed, user changes contents from 6ØØ2ØØ to 213775 and opens the next sequential location.
>ØØ1Ø2/111215>ØØ3ØØ← ↵	User modifies contents of location 1Ø2 and commands PATCH to go to and open the location addressed by the contents of location 1Ø2 (i.e., location 3ØØ).
>ØØ3ØØ/ØØØ11Ø>↑ ↵	User enters ↑ ↵ to open and examine contents of preceding location (i.e., 277).
>ØØ277/7Ø3112>(ALT MODE)	User terminates current list operation by entering ALT MODE (symbol $).
>	PATCH prints a go-ahead symbol to indicate that it is ready for the next operation.

5.4 LR LIST COMMAND

The LR List command has the following format:

```
>LR␣n)
```

where LR initiates the listing operation for a previously selected READR-installed Systems Program, and n represents the 13 bit "unrelocated" address (in octal) of the program location to be opened.

NOTE

PATCH automatically calculates and adds the needed relocation factor to the user's entry (i.e., n). This enables the user to work directly from original unrelocated listings.

The major difference between the "L" and "LR" list operations is that in L operations only absolute address information is output in responses and is required in modification procedures. In LR operations, however, the user must deal with both relocated and unrelocated addresses in list printouts and in register modification procedures.

When the LR command is used to open word locations containing either data or non-memory reference instructions, the responses output are similar to those described for the L command. The only difference is that in the LR responses the address of the accessed location is given in relocated form relative to the original MACRO XVM assigned address.

NOTE

In the following paragraphs the symbol # is used to denote the relocation factor used by PATCH in the conversion of an unrelocated program into a system program.

To illustrate, assume that 6 is the unrelocated address of a user-selected location containing a non-memory reference instruction or data and that the relocation factor (#) is 17344. The response to the command:

Listing and Modifying Program and Data Block Registers

```
>LR␣6)
```

is printed in the following form:

```
>17344 + 6   /   nnnnnn   >

 Relocated  /Contents of location (6 digits)
 Address    /Op-Code and Operand or Data
```

For example:

```
>17352/017206>
```

To illustrate, assume that a relocation factor of 17642 is current and that the selected program location 1Ø1 contains a JMP .+1 instruction. The response to the command:

```
>LR␣1Ø1)
```

is printed in the following form:

```
>17743/617744<ØØ1Ø2>
```

```
Relocated   /Contents, Op-Code + # + Operand  ⟨Unrelocated Operand,⟩
Location   /  (6ØØØØØ + 17642 + 1Ø2           \Referenced Address /
Address /          JMP=6ØØØØØ                  (i.e., ØØ1Ø2)
(17642+101)   .+1 = 1Ø1+1 = 1Ø2
```

Printing an address operand in its unrelocated form enables the user to easily recognize the instruction containing it, and the referenced location without calculating and subtracting the PATCH relocation factor (#).

The unrelocated referenced address within a memory-reference type instruction is not printed by PATCH if it is less than the first address of the program; the address printed in angle brackets is always positive, though it may be zero.

5.4.1 LR, Register Modification Procedure

The contents of registers opened during LR operation are changed, as for L list operations, by the entry of an expression which is to replace the contents of the current location. New contents are entered immediately after the listed response and replace the current contents on termination of the user input.

Input expressions are formed using:

a. Octal numbers up to six digits in length; PATCH always assume a six-digit input and right-justifies all numeric inputs (e.g., an entry of 300_8 is recognized as 000300_8).
b. Operators (symbols) as listed in Table 5-1.
c. Symbolic representation of all DIGITAL XVM basic instruction op-codes and operate group instructions.
d. The symbol # to represent the relocation factor of the currently selected, READR-installed System Program.

Command string terminators indicate the next sequential operation to be carried out (refer to Table 5-2).

5.4.2 Relocation and LR List Operation Examples

The following examples use a single memory reference instruction to illustrate the relocation operations performed by READR in the installation of a SYS file and the manner in which the instruction may subsequently be examined using the LR command.

EXAMPLE:

a. Assume the instruction LAC␣300 located in MACRO assigned location 200 of an unrelocated 400_8-word program.

b. The program is selected by the user and is patched onto a prepared system device medium as a SYS file using the READR function/command.

Program Name	>Filename (name given at SGEN time) >READR Filename (current name of program)

c. During READR operations, a relocation factor is calculated and added to all location addresses contained by the program. For a 400_8-word program # = 17237_8. Therefore, the relocatable input memory reference instruction

Address	Instruction
200_8	LAC 300

is relocated as

(#+200) (LAC #+300)

which equals

Address	Instruction
17437_8	LAC 17537_8

or, in full octal form:

17437	217537

The relocated instruction is then output to the system device.

d. To examine the instruction given in step c. in a READR-installed SYS file, the user calls PATCH, selects the program, and uses the LR command

```
$PATCH
>filename,
>LR␣200)
```

The responses output on the I/O console will be:

```
>17437/217537<ØØ3ØØ>
```

e. To change the contents of the location, the user enters the desired expression and terminates the command string. If the contents contain an address, the relocation factor must be added to the new entry.

For example, to change the address references in the contents of the location opened in step d from the 300_8 to 305_8 (unrelocated values), the user must enter the expression:

217544_8

if he knows the value of the relocation factor # (i.e., 17237 + 305).

or

LAC #+305_8

EXAMPLE:

```
>17436/217537<ØØ3ØØ> LAC#+3Ø5 )
```

changes the location to

17436/217544

and the terminator) causes the next sequential location to be opened and a response output.

5.5 SYSTEM PROGRAM SYSBLK PARAMETER COMMANDS

PATCH provides, as a convenience for the user, five separate commands which obtain and modify, if desired, SYSBLK parameters for the user selected program currently in core. By entering the appropriate command from the following list and using suitable terminators as listed in Table 5-2, the user may start the listing at any specific point and proceed subsequently through the desired parameters.

The commands, the parameters opened, and the word positions are as follows:

Command	SYSBLK Parameter Opened	Word Position
a) FB↵	Number of first block used	3
b) NB↵	Number of blocks allotted to program	4
c) FA↵	First core address occupied when the SYSfile is loaded into core (13-bits)	5
d) PS↵	Program Size	6
e) SA↵	Starting Address (13-bits)	7

When examining the parameter group of a READR-installed System Program, the user must remember that:

FA will give the address of the first (lowest) core locations occupied by the System Program/BBI combination when it is loaded.

PS will give the TOTAL size of the System Program/BBI combination.

SA will give the starting address of the BBI routine (first address above Transfer Vector address table).

NOTE

The program system parameter commands listed above may not be used:

a. when a program has not been selected or the command CTRL P (↑P) has been entered;

b. when SYSBLK has been selected;

c. when the commands B, B+, or B- are in effect.

Listing and Modifying Program and Data Block Registers

The five SYSBLK Parameter commands operate in a manner similar to the List (L) command. When entered, the corresponding parameter location address and contents are printed out followed by a go-ahead symbol (>) as for the L command. As in other list operations, the user may, according to the manner in which terminators are used: 1) modify the opened register, 2) view the next sequential register, 3) view the preceding register, 4) jump to and view the location addressed by the contents of the last opened register, or 5) terminate the operation.

The operations initiated by a Parameter command are carried out on the SYSBLK information contained by the PATCH command table. This SYSBLK data is obtained by PATCH when it is first loaded into core and prior to the start of user PATCH operations. On completion of each SYSBLK Parameter command operation (ALT MODE terminator used), the modified (or unchanged) command table SYSBLK is copied onto the system on .DAT -14.

The five separate commands are provided by PATCH as a convenience for the user; they permit the user to obtain a specific parameter or to start listing operations at a specific point with a simple straightforward command. The complete series of five program parameters may be obtained simply by starting with the FB command and advancing the listing operation through the next four sequential word locations. This method is the only way to modify SYSBLK parameters without reloading PATCH since it also modifies the PATCH command table.

EXAMPLE:

The following example illustrates the use of a PATCH SYSBLK parameter command to obtain and modify the first address occupied, program size and starting address parameters for the system program currently in core.

```
> FA↵                                parameter command
 00066/013670>13660↵                first address opened and modified
 00067/003747>3757↵                 program size opened and modified
 00070/013671>13661(ALT MODE)       starting address opened and
>                                   modified and operation terminated.
```

5.6 MODIFYING REGISTERS IN DATA BLOCKS

Users can modify registers in data blocks (on the device assigned to .DAT -14) by using the 'B' command like the 'L' operation.

This facility is utilized in a UC15 system to patch PDP-11 programs, through octal expressions.

CHAPTER 6
SUMMARY, OPERATING PROCEDURES AND ERROR MESSAGES

6.1 CONTENTS

Brief procedural descriptions of the operations needed before loading the PATCH program and for the performance of each PATCH function are given in this Section. Error detection and error messages are also described.

6.2 PRE-LOAD OPERATIONS

6.2.1 .DAT Slot Assignments

The user must make the .DAT slot assignments given in Table 6-1 immediately prior to loading the PATCH program.

Table 6-1
Required .DAT Slot Assignments

.DAT Slot	Used to
-14	Input from and output to the device on which patches are to be made. The device handler is required only to perform .TRAN.
-10	Input from the auxiliary device. The device handler must handle Dump Mode input and, if it is for a non-file oriented device, must handle image alpha mode.
Permanently Assigned -3	Output to the teleprinter.
Permanently Assigned -2	Input from the console keyboard or batch processing device.

.DAT slot -10 can be assigned to no device handler (NONE) if auxiliary input is not required. .DAT slots -3 and -2 cannot be changed.

6.2.2 I/O Device Preparation

The user must ensure that the I/O device assigned to .DAT-14 (material to be patched) is WRITE ENABLED before starting PATCH operations.

If the device is a

1. DECtape unit - set the unit's WRITE ENABLED/WRITE LOCK switch to the WRITE ENABLED position.
2. Disk - set the assigned logical disk unit's WRITE LOCKOUT switches to their ENABLED positions.

6.3 CALLING PATCH

PATCH is called by issuing the command PATCH from the system I/O console. When PATCH is loaded and running, it outputs its name and version number. For example:

```
XVM/DOS Vnxnnn
$PATCH↵
PATCH XVM Vnxnnn
>
```

6.4 PATCH OPERATIONS

The operations which may be performed using PATCH fall into four (4) functional groups. Procedures, in table form, are given for the List "L" and "LR" functions.

The Examination and modification of registers in absolute programs are summarized in Table 6-2. The examination and modification of registers in Patch-READR-installed system programs are summarized in Table 6-3.

6.4.1 Replacing Absolute Programs from External PATCH Files

The required procedure is to:

a. Select Program or data block to be patched (e.g. >EDIT↵).
b. Specify READ operation and input file with the command string

>READ␣[filename]↵ (e.g., >READ␣EDPCH↵)

6.4.2 Installation of Relocatable Programs as Absolute SYS Files

The required procedure is:

a. Select the system device area prepared (by SGEN) to receive the new SYS file (e.g., >NUFIL↵).

b. Enter the PATCH READR command using the following format:

>READR␣[nnnnn]␣[filename]↵ (e.g., >READR␣176ØØ␣NUFIL↵)

NOTE

"nnnnn" represents the highest register the new SYS file will occupy when loaded into core. In the example, this is specified as $176ØØ_8$ which would leave a 36_8-word buffer between the top of the SYS file and the first Bootstrap location. The default value of nnnnn=17636_8.

6.5 ERROR DETECTION

PATCH, on the detection of an error, causes the following:

1) the current function is terminated;

2) the data block (program or selected block) current in the PATCH Block Buffer is output to the device on .DAT-14 if modifications had been made to the block;

3) an appropriate error message is output at the user's I/O terminal.

In the event of an .IOPS error, control is not automatically returned to the monitor. PATCH can be restarted by the entry of a CTRL P (↑P) command. CTRL P is also useful in terminating a read operation in the event of an equipment malfunction (e.g., paper tape reader jams).

In the event of an I/O device not ready error (IOPS4), CTRL R may be used to continue the current operation after the device involved has been made ready. CTRL C may also be used to return control to the monitor; however, care must be taken not to use this command when a register is open since any modification made to the current block in core will be lost.

6.6 ERROR MESSAGES

The error messages output by PATCH are listed and described in Table 6-4.

Table 6-2

Procedure for Modifying Registers in Absolute System Programs

Operation	Item to be Patched		
	BLOCK	PROGRAM	PROGRAM SYSBLK PARAMETERS
SELECT ITEM TO BE PATCHED	>B␣n↵ or >B+␣n↵ or >B-␣n↵	>Prog. Name↵	>Prog. Name↵
SELECT REGISTER OR PARAMETER	>L␣n↵	>L␣n↵	>FA↵ or >FB↵ or >NB↵ or >PS↵ or >SA↵
MODIFY OPENED ITEM	ENTER NEW CONTENTS IMMEDIATELY AFTER RESPONSE OUTPUT WHEN LOCATION WAS OPENED. e.g., >Address/ Old Contents > New Contents		
SELECT NEXT OPERATION	TERMINATE RESPONSE LINE ACCORDING TO DESIRED NEXT OPERATION: ↵ go to next sequential location open and list address and contents ↑↵ go to preceding register, open and list address and contents ←↵ go to register identified by contents of current register, open and list address and contents (Ⓢ) terminate current list operations		

Table 6-3

Procedure for Modifying Registers in PATCH-READR-Installed System Programs

Operation	Item To Be Patched	
	PROGRAM	**PROGRAM** **SYSBLK** **PARAMETERS**
SELECT ITEM	>Prog. Name ↵	>Prog. Name ↵
SELECT REGISTER OR PARAMETER	>LR␣n ↵	>FA ↵ or >FB ↵ or >NB ↵ or >PS ↵ or >SA ↵
MODIFY OPENED ITEM	ENTER NEW CONTENTS IMMEDIATELY AFTER RESPONSE OUTPUT WHEN LOCATION WAS OPENED. USE RELOCATION FACTOR (#) WHEN CHANGING MEMORY REFERENCE OR TRANSFER VECTOR REGISTERS. e.g., standard register > relocated address / old contents > new contents memory reference instruction or transfer vector > relocated address / old contents < unrelocated address of contents > new contents + #	
SELECT NEXT OPERATION	TERMINATE RESPONSE LINE ACCORDING TO DESIRED NEXT OPERATION: ↵ go to next sequential location open and list address and contents ↑↵ go to preceding register, open and list address and contents ←↵ go to register identified by contents of current register, open and list address and contents. (Ⓢ) terminate current list operations	

Table 6-4

ERROR MESSAGES

Printout	Meaning
ILLEGAL COMMAND	a. An attempt was made to issue a command before a file or block was selected. b. Command unrecognizable.
NOT OCTAL DIGIT	A character other than an octal digit was encountered where an octal digit was expected.
TOO MANY DIGITS	The command string contains an octal number with more than 6 digits.
ADDRESS OUT OF RANGE	a. The address requested is not within the legal range of the currently selected program or block. b. Address or size argument of a SYSBLK command is greater than 17645_8 or is negative.
CHECKSUM ERROR	Bad data read in from auxiliary input device (checksum, parity, or validity bits) - try again.
FILE NOT FOUND	The file named in a READ or READR command does not exist on the auxiliary device.
ILLEGAL BLOCK #	The block number specified in a Block Selection command was less than Ø.
ILLEGAL SIZE	a. The program size (PS command) exceeds the number of blocks allotted for the current program. b. The size of the program (including the bank bit initialization routine supplied by PATCH) input by a READR command exceeds the number of blocks allotted (during system generation) for the SYS file or is greater than nnnnn-$2Ø_8$.
MORE THAN 256 TRANSFER VECTORS	The program being input by a READR command contains more than 256 transfer vectors.
NOT RELOCATABLE BINARY	The program being input by a READR command is not a relocatable binary file.
.GLOBL NOT ALLOWED	The program being input by a READR command contains an external .GLOBL (internal .GLOBL ignored).
LAST ADDRESS GREATER THAN 17636	The last address of the program input by a READR command is greater than 17636_8.

APPENDIX A

SYSTEM BLOCK (SYSBLK) DESCRIPTION

A.1 SYSTEM BLOCK (SYSBLK)

In DIGITAL XVM systems the monitor contains a System Block table (abbreviated as SYSBLK) which contains the name, physical parameters and I/O information for each system program in the software system.

The information contained by SYSBLK in the initial operating system is determined during System Generation procedures; refer to the SGEN XVM Utility Manual. SYSBLK is used by the system loader programs in locating and loading selected (called) System Programs into core.

PATCH, when first loaded into core, checks for a SYSBLK on .DAT -14 and, if one is present, it loads the SYSBLK Parameter Table into core as a PATCH Command Table. This command table is used unchanged during all subsequent PATCH System Program operations, including SYSBLK modification operations.

SYSBLK, itself, is divided into two distinct areas (see Figure A-1), a Parameter Table, and a Communications Table. PATCH operations normally concern only the SYSBLK Parameter Table.

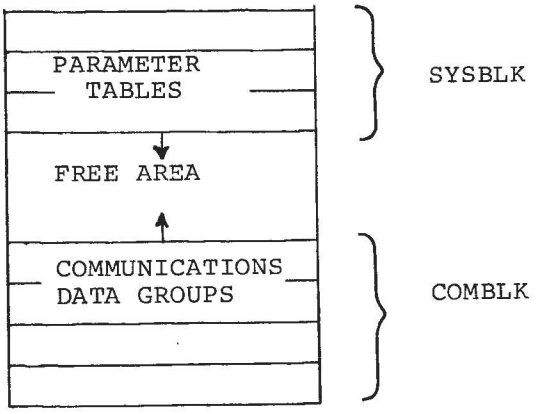


Figure A-1
System Block, Overall Configuration

A.1.1 SYSBLK Parameter Table

The SYSBLK parameter tables contain separate seven-word parameter groups for each System Program in the operating system. Each parameter group consists of the following:

WORD	CONTENTS
1 & 2	Name of the System Program or overlay (i.e., PATCH, DDT, EDIT, etc.) in .SIXBT form.
3	The number of the first system device block occupied by this System Program or overlay.
4	The amount of system device blocks allotted to contain the program. For PATCH-installed programs, this number is allocated during user SGEN procedures.
5	The address (13-bit) of the first core locations that will be occupied by the System Program when it is loaded.
6	The size of the program.
7	The program starting address (13-bit).

WORD	OCTAL PRINTOUT CONTENT	INTERPRETATION
ØØØ1Ø ØØØ11	Ø5Ø411 24ØØØØ	Program name "EDIT" in .SIXBT code.
ØØØ12	ØØØ63Ø	The first block occupied by EDIT is $63Ø_8$.
ØØØ13	ØØØØ12	Twelve blocks (octal) are allotted to hold the program EDIT.
ØØØ14	Ø13ØØØ	The first core location[1] occupied by EDIT is $13ØØØ_8$.
ØØØ15	ØØ4636	EDIT consists of 4636_8 words.
ØØØ16	Ø13ØØØ	Program start address[1] is at core location $13ØØØ_8$.

Figure A-2
System Program EDIT SYSBLK Parameter Group

A.1.2 Communications Table

The system block COMBLK area contains individual communication data groups which store .DAT slot, buffer, and overlay information for each system program.

[1]At load time, the bank bits of the highest 8K system bank are added to this address to properly locate the program in the highest bank.

APPENDIX B

SYSTEM PROGRAMS, GENERAL DESCRIPTION

System Programs have a unique feature in that they are loaded into core and are started by the entry of a single user command (i.e., program name) at the system console keyboard. This feature is implemented by name, control parameters, and I/O device information contained for each System Program in the system SYSBLK Parameter and Communication (COMBLK) tables. The information contained in SYSBLK enables the Monitor to recognize System Program names (as stored in SYSBLK) and causes it to initiate the loading of the identified program into core.

System Programs and their associated Device Handlers are loaded into core from the system device by the System Loader (.SYSLD) program.

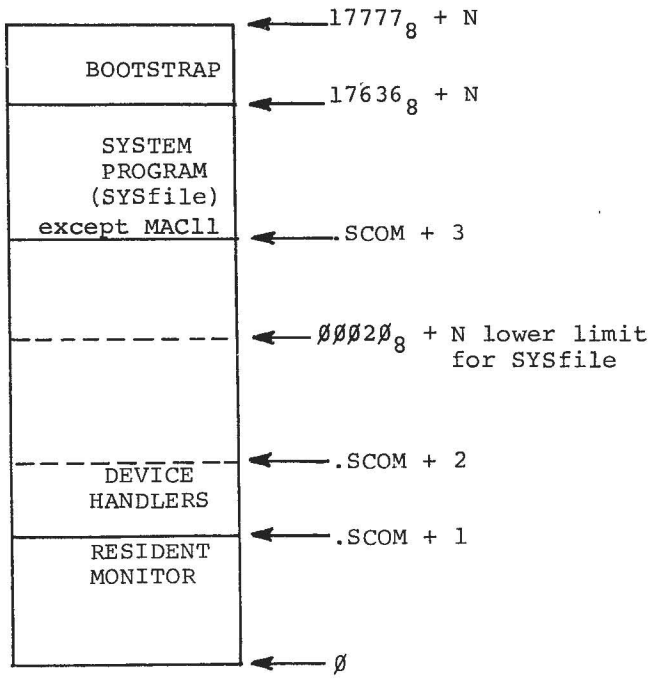
System programs may not be loaded above 32K and should not use wide addressing.

The placement of System Programs and their device handlers in core is illustrated in Figure B-1. As shown:

1) System Programs are always loaded immediately below the Bootstrap in the highest portion of the highest 8K bank in the system, except for MAC11 which is always loaded into 'Bank 1' of the system.

2) The lowest register which a System Program may occupy is 20_8 in the highest 8K bank of the system; handlers and free core may be located anywhere in the system.

System Programs are stored on the system device as absolute executable files and are commonly referred to as "SYS files". Software is supplied to the user and contains a standard group of commonly used utility and language programs installed on some medium (e.g., DECtape) as SYS files. The SGEN and PATCH utility programs can be used to delete the supplied SYS files or replace them with other DIGITAL-supplied programs or with user programs. Table 2-1 lists the names and SYSBLK entry names of the System Programs supplied with DIGITAL XVM software systems.



LEGEND: N = 4ØØØØ for 24K systems
= 6ØØØØ for 32K systems

Figure B-1
Memory Map, Loading System Programs

INDEX

PATCH XVM
Utility Manual
DEC-XV-UPUMA-A-D

Please cut along this line.

READER'S COMMENTS

NOTE: This form is for document comments only. Problems with software should be reported on a Software Problem Report (SPR) form.

Did you find errors in this manual? If so, specify by page.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

If you require a written reply, please check here. ☐

---------- Fold Here ----------

---------- Do Not Tear - Fold Here and Staple ----------

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Software Communications
P. O. Box F
Maynard, Massachusetts 01754

digital

digital equipment corporation

PRINTED IN U.S.A.