digital

VAX–11 PL/I

Installation and System Management Guide

Order No. AA-J179A-TE

VAX11

October 1980

Describes the procedure for installing the VAX–11 PL/I compiler; summarizes system management and problem reporting concerns; and includes notes on migrating PL/I programs from other implementations of PL/I.

VAX–11 PL/I
Installation and System Management Guide

Order No. AA-J179A-TE

SUPERSESSION/UPDATE INFORMATION:	This is a new document for this release.
OPERATING SYSTEM AND VERSION:	VAX/VMS V2.0
SOFTWARE VERSION:	VAX–11 PL/I V1.0

To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754

digital equipment corporation · maynard, massachusetts

First Printing, October 1980

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 only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

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
DEC
PDP
DECUS
UNIBUS
COMPUTER LABS
COMTEX
DDT
DECCOMM
ASSIST-11
VAX
DECnet
DATATRIEVE
DECsystem-10
DECtape
DIBOL
EDUSYSTEM
FLIP CHIP
FOCAL
INDAC
LAB-8
DECSYSTEM-20
RTS-8
VMS
IAS
TRAX
MASSBUS
OMNIBUS
OS/8
PHA
RSTS
RSX
TYPESET-8
TYPESET-11
TMS-11
ITPS-10
SBI
PDT

ZKA84-80

READ THIS FIRST

This manual describes how to install the VAX-11 PL/I compiler on the VAX/VMS operating system. It also includes:

- Information of general interest to a system manager in setting up accounts for PL/I programmers and reporting problems to DIGITAL.
- Notes on converting programs written for other implementations of PL/I to VAX-11 PL/I.

This manual is intended for use by system managers and PL/I programmers who are responsible for installing and maintaining the VAX-11 PL/I compiler.

It includes information of interest to programmers who are responsible for planning and implementing applications using PL/I, particularly those who are migrating to VAX-11 PL/I from another implementation of the PL/I language.

CONTENTS

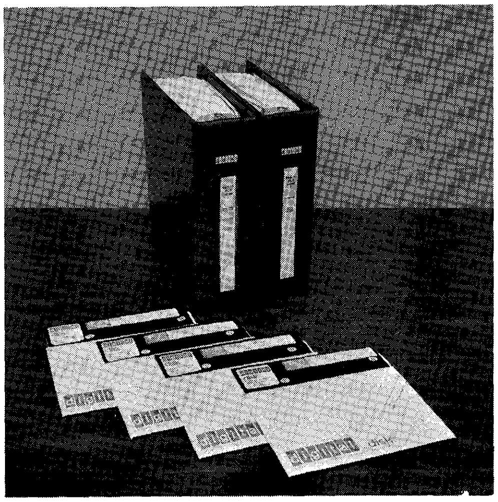


Figure 1-1 The VAX-11 PL/I Floppy Diskette Distribution Kit

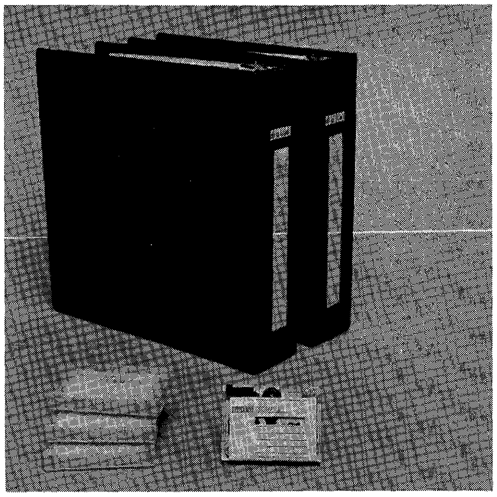


Figure 1-2 The VAX-11 PL/I TU58 Tape Cassette Distribution Kit

CHAPTER 1

INSTALLATION PROCEDURE

This section describes the procedure for installing the VAX-11 PL/I compiler on VAX. This procedure is automated and requires only that the person performing it insert distribution media in the console drive and respond to queries.

1.1 REQUIREMENTS FOR INSTALLATION

This installation can be performed by any system user who has access to the system manager's account or to an account that has the same privileges as the system manager's account.

The operation and use of the VAX-11 PL/I compiler require approximately 1200 blocks of disk space and 512K bytes of physical memory.

The installation procedure itself requires a minimum of 7000 blocks of free disk space.

1.2 CONTENTS OF THE DISTRIBUTION KIT

The VAX-11 PL/I compiler installation kit is distributed on either of two media:

- The RX01 floppy disk distribution kit, Order Number QE114-AY, is illustrated in Figure 1-1.
- The TU58 tape cassette distribution kit, Order Number QD114-AG, is illustrated in Figure 1-2.

Each distribution kit consists of volumes of the indicated media and a set of documentation. All files required to install the compiler and all files required for the compiler to run are contained on the four volumes.

Each volume is labeled with an external serial number corresponding to the VAX-11 PL/I compiler's product number and a unique volume label that distinguishes that volume from other volumes in the distribution kit. The volumes in the VAX-11 PL/I distribution kit are labeled as follows:

Floppy Diskette Number	**Volume Label**
AS-K234A-TE VAX-11 PL/I V1.0 BIN RX1 1/4	VAXPLI10A
AS-K235A-TE VAX-11 PL/I V1.0 BIN RX1 2/4	VAXPLI10B
AS-K236A-TE VAX-11 PL/I V1.0 BIN RX1 3/4	VAXPLI10C
AS-K237A-TE VAX-11 PL/I V1.0 BIN RX1 4/4	VAXPLI10D

Tape Cassette Serial Number	Volume Label
BE-K510A-BE VAX-11 PL/I V1.0 BIN TU58 1/4	VAXPLI10A
BE-K511A-BE VAX-11 PL/I V1.0 BIN TU58 2/4	VAXPLI10B
BE-K512A-BE VAX-11 PL/I V1.0 BIN TU58 3/4	VAXPLI10C
BE-K513A-BE VAX-11 PL/I V1.0 BIN TU58 4/4	VAXPLI10D

During the installation, you must mount these volumes in the order listed above.

1.3 LOGGING IN TO THE SYSTEM MANAGER'S ACCOUNT

Before you install the VAX-11 PL/I compiler, perform the following steps:

1. At the system console terminal, log into the system using the system manager's account, as in this example:

```
RET
Username:  SYSTEM RET
Password:      RET
```

2. If you are installing the VAX-11 PL/I compiler on the current system disk, perform Step 2(a). If you are installing the compiler on another system disk, perform Step 2(b).

   (a) If you are installing the compiler on the current system disk, enter the following commands in the order shown:

```
$ SET UIC [1,4]
$ SET DEFAULT [SYSUPD]
$ @VMSUPDATE
```

   If you receive an error message indicating that the command file VMSUPDATE.COM cannot be found, determine whether the account you logged into has the correct default disk device; the device must be the same as the device containing the VAX/VMS system. To determine whether the current default disk device is the correct disk, enter the commands:

```
$ SHOW LOGICAL SYS$DISK
$ SHOW LOGICAL SYS$SYSTEM
```

   If the physical disks are not the same, you must redefine SYS$DISK to be the same disk as that shown for SYS$SYSTEM, as follows:

```
$ DEFINE SYS$DISK ddcu:
```

   where ddcu is the device shown in the response to SHOW LOGICAL SYS$SYSTEM.

(b) If you are installing the compiler on a VAX/VMS system disk that is not the current system disk, enter the following commands in the order shown:

```
$ SET UIC [1,4]
$ SET DEFAULT ddcu:[SYSUPD]
$ @VMSUPDATE
```

where ddcu is the device name of a disk containing a VAX/VMS system. If you receive an error message indicating that the file VMSUPDATE.COM cannot be found, verify that the disk you have specified is a system disk.

When it begins executing, VMSUPDATE.COM displays the following:

```
This command procedure performs VAX/VMS software updates and
unbundled software installations for VAX/VMS Release 2. During
this sequence, the standard console medium will not be present in
the console drive. Therefore, the system may be vulnerable to
power failure or other fatal crash. If a system crash should
occur during this period the update sequence can be restarted at
the beginning of the first incomplete update.

Dismount the current console medium.[1]

Please place the first volume in the console drive.

Are you ready to continue?
```

You are now ready to begin inserting the VAX-11 PL/I installation kit volumes in the console drive. If you are unfamiliar with the media in your kit, you should read one of the following sections before proceeding:

- Section 1.4, "Instructions for Inserting Floppy Diskettes," gives step-by-step instructions for inserting and removing floppy diskette volumes.

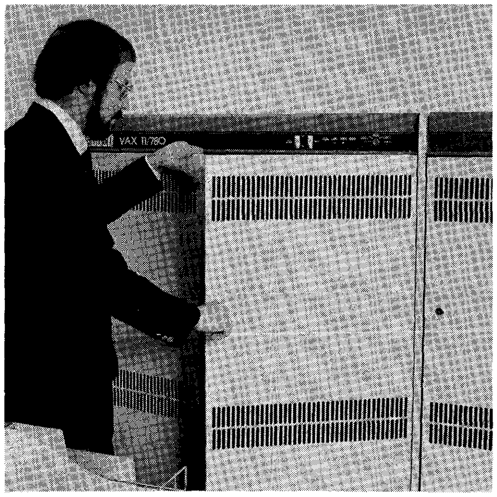
- Section 1.5, "Instructions for Inserting TU58 Tape Cassettes," gives step-by-step instructions for inserting and removing the TU58 cassette volumes.

If you are familiar with the media you are using, you can skip to Section 1.6, "Queries, Messages, and Responses."

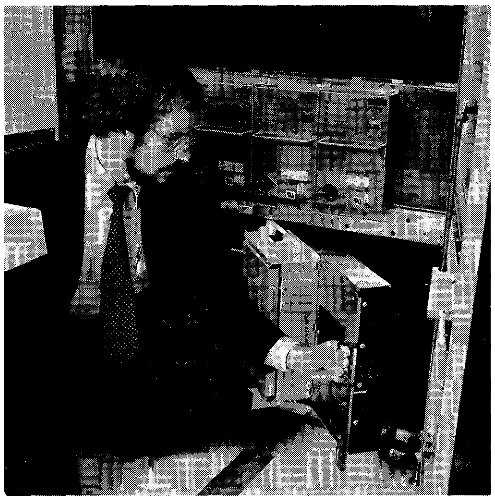
[1] If you receive the message DEVNOTMOUNT at this point, it indicates that there is no volume in the console drive or that the volume in the drive is not mounted. In either case, you can ignore this message and continue.

1.4 INSTRUCTIONS FOR INSERTING FLOPPY DISKETTES

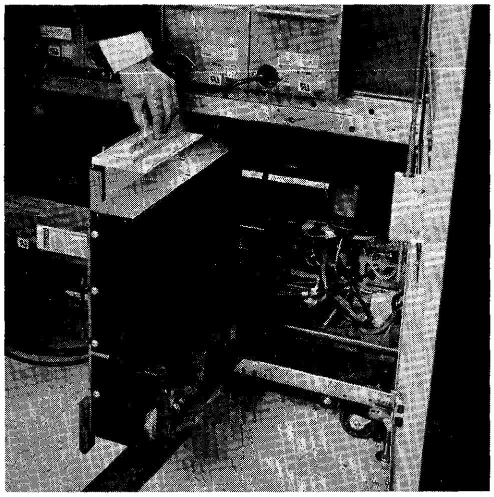
This section illustrates the step-by-step procedure for inserting floppy diskettes in the console floppy drive.



❶



❷



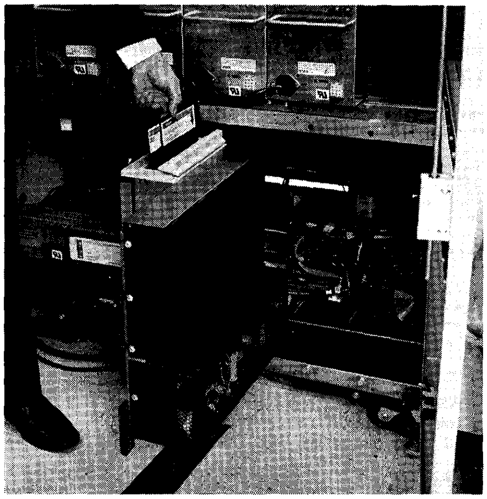
❸

❶ Unlock and open the cabinet doors of the central processor.

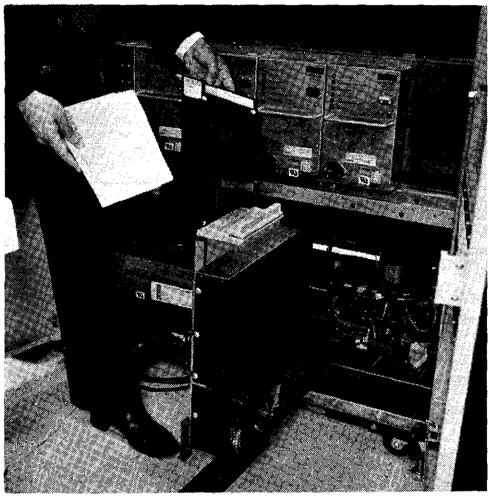
❷ Swing out the drive assembly until it is at a right angle to the cabinet.

The drive assembly is a rectangular, unpainted steel box in the lower right-hand corner of the central processor cabinet. There is a black handle on the right of the drive assembly. Pull the handle to swing out the drive assembly until it is perpendicular to the cabinet. If you do not swing it out far enough, or if you swing it out too far, you will not be able to insert the floppy diskettes.

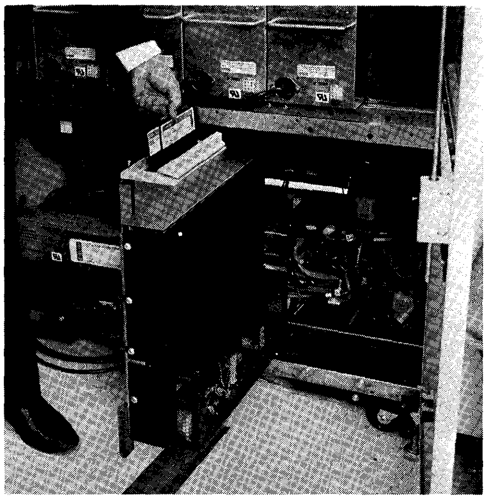
❸ Squeeze the rectangular black pushbutton to unlock the slot cover. The cover will spring open.



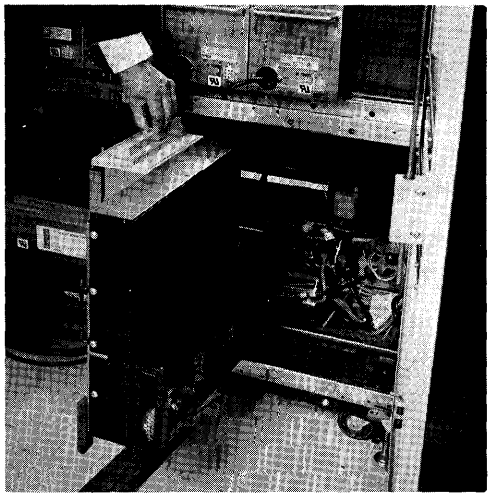
❹



❺



❻



❼

❹ If there is already a floppy diskette in the console floppy drive, remove it.

When you remove the floppy diskette, note the direction it is facing, that is, the direction of the label on the diskette.

❺ Select the first floppy diskette from the distribution kit, that is, the floppy labeled 1/4.

❻ Insert this floppy in the drive.

When you insert a diskette from the distribution kit, its label (on the smooth side of the diskette) should be at the top and should face the right-hand cabinet door. The oval slot on the diskette should be at the bottom.

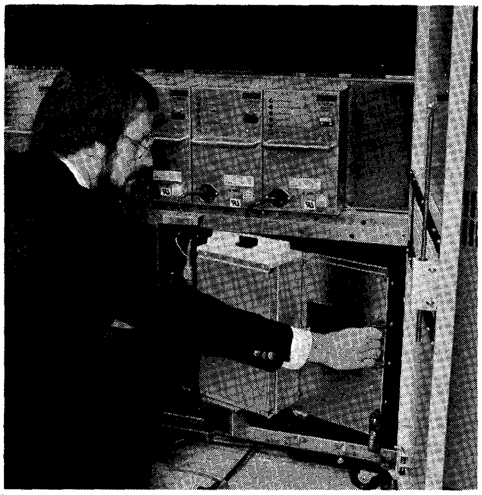
❼ Close the diskette slot cover.

You must repeat Steps 3 through 7 each time you insert a diskette. The installation procedure will tell exactly you when to insert each volume. The installation procedure takes approximately 25 minutes.

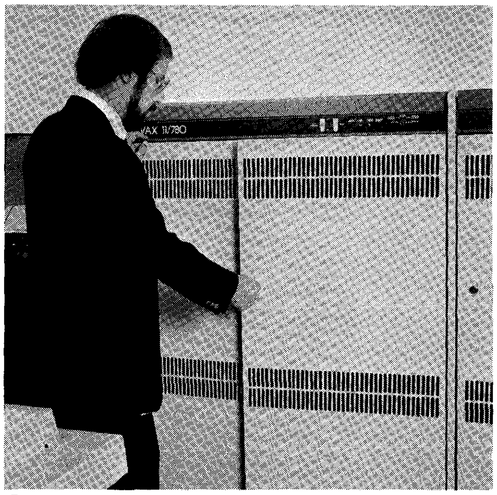
When you have completed the entire procedure (that is, you have completed the steps described in Section 1.6 of this Installation Guide), close up the cabinet as illustrated below:

❽ Swing the drive assembly back into the cabinet.

❾ Close and lock the door on the cabinet.



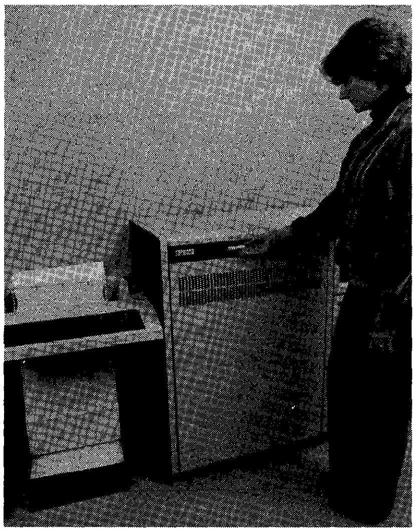
❽



❾

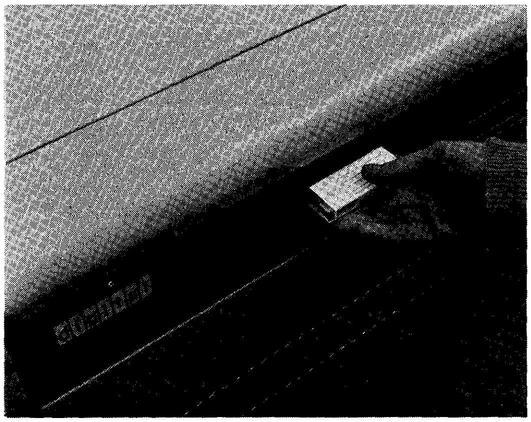
1.5 INSTRUCTIONS FOR INSERTING TU58 TAPE CASSETTES

This section illustrates the procedure for inserting TU58 tape cassettes.

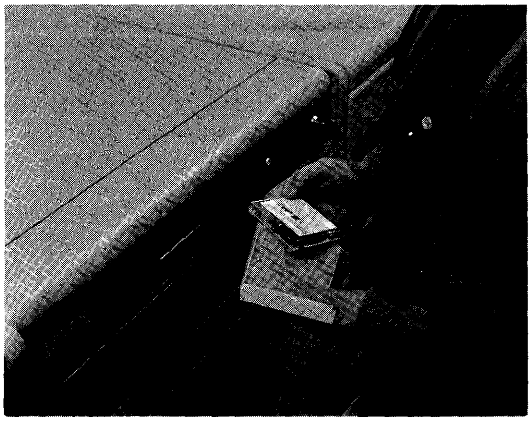


❶

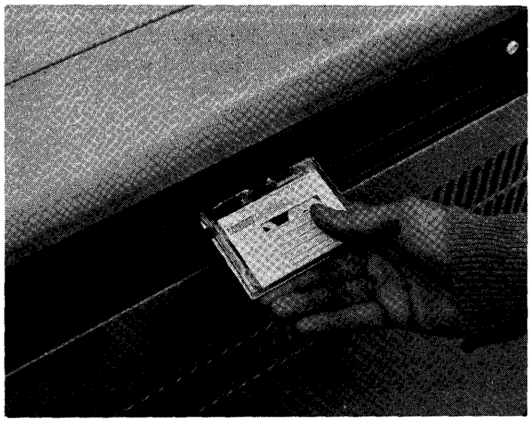
❶ Locate the cassette slot near the top of the central processor cabinet; it is slightly to the left of center.



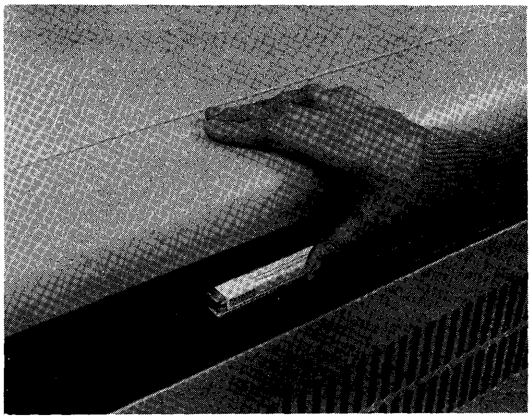
❷



❸



❹



❺

❷ If there is already a cassette in the slot, remove it by pulling it toward you.

You may need to give the cassette a little tug to pull it out.

❸ Select the first cassette volume from the distribution kit, that is, the cassette labeled 1/4.

❹ Insert this volume in the slot.

When you insert the tape cassette, its label should be facing up and its metal side facing down. The black round wheel should be facing the rear of the slot.

❺ Push the cassette into the slot as far is it will go. When it is completely inserted, the edge of the cassette sticks out from the cabinet approximately 1/2 inch.

You must repeat steps 2 through 5 each time you insert a cassette. The installation procedure will tell you exactly when to insert each volume.

1.6 QUERIES, MESSAGES, AND RESPONSES

During the installation, the command procedure requests you to insert volumes and waits for you to type Y after inserting each one. If you type N to respond to any queries, the command procedure deletes all files it has created to that point and exits.

After you have inserted the first volume, labeled 1/4, in the console drive, you must type Y in response to the last message, which was:

```
     Are you ready to continue?
```

After you type Y, the following message is displayed:

```
     %MOUNT-I-MOUNTED, VAXPLI10A   mounted on _CSA1:
```

Then, the command procedure VAXPLI10A.COM assumes control. It displays the message:

```
*** VAX-11 PL/I Installation beginning at dd-mmm-yyyy hh:mm:ss.ss ***
```

The procedure checks that there is adequate disk space to build the compiler. If not, it issues an error message and terminates.

Otherwise, the procedure continues by processing the files on the volume VAXPLI10A needed to build the compiler and libraries.

When all of the files on this volume have been processed, you are prompted to insert the next volume with the message:

```
     Place the second VAX-11 PL/I installation kit volume -
     "VAXPLI10B" in the console drive.
     Are you ready to continue?
```

When you have replaced the first volume with the second volume, labeled 2/4, enter Y to continue processing files. If you enter Y or Yes, the mount message is displayed:

```
     %MOUNT-I-MOUNTED, VAXPLI10B   mounted on _CSA1:
```

When all the files on the volume VAXPLI10B have been processed, you are prompted to insert the next volume with the message:

```
     Place the third VAX-11 PL/I installation kit volume - "VAXPLI10C"
     in the console drive.
     Are you ready to continue?
```

When you have replaced the second volume with the third volume, labeled 3/4, enter Y to continue processing files. The mount message is then displayed:

```
     %MOUNT-I-MOUNTED, VAXPLI10C    mounted on _CSA1:
```

When all the files on the volume VAXPLI10C have been processed, you are prompted to insert the next volume with the message:

```
     Place the fourth VAX-11 PL/I installation kit volume -
     "VAXPLI10D" in the console drive.
     Are you ready to continue?
```

When you have replaced the third volume with the fourth volume, labeled 4/4, enter Y to continue processing files. The mount message is displayed:

```
     %MOUNT-I-MOUNTED, VAXPLI10D    mounted on _CSA1:
```

When all the files on the volume VAXPLI10D have been processed, the command procedure now performs the following:

1. It copies [SYSLIB]STARLET.OLB to [SYSLIB]NEWSTAR.OLB and inserts the VAX-11 PL/I run-time modules in NEWSTAR.OLB.
2. It links the compiler and builds the image [SYSUPD]PLIG.EXE.
3. It copies [SYSHLP]HELPLIB.HLB to [SYSHLP]NEWHELP.HLB and inserts the VAX-11 PL/I HELP files in NEWHELP.
4. It verifies that the compiler and libraries were successfully built by running the Installation Certification Procedure (ICP).
5. If the ICP is successful, it renames the following files:

Temporary File	**Renamed To**
NEWSTAR.OLB	STARLET.OLB
NEWHELP.HLB	HELPLIB.HLB
[SYSUPD]PLIG.EXE	[SYSEXE]PLIG.EXE
[SYSUPD]PLISYSDEF.TLB	[SYSLIB]PLISYSDEF.TLB

If an error occurs at any point during the installation, the files created up to that time are deleted. During the Installation Certification Procedure, the following messages are displayed:

```
*** VAX-11 PL/I Installation Certification Procedure ***
   **** Beginning at dd-mmm-yyyy hh:mm:ss.ss ****


****** PL/I Installation Certification Procedure SUCCESSFUL ******
*** VAX-11 PL/I Installation Certification Procedure ***
   ***** Ending at dd-mmm-yyyy hh:mm:ss.ss *****
```

If the five steps listed above are successfully performed, there should be no other messages or notifications on the terminal.

If an error occurs, see Section 1.8, "Error Conditions."

1.7 COMPLETION OF THE PROCEDURE

When the installation procedure is completed, control returns to the system command procedure VMSUPDATE. This procedure issues the following message:

```
Are there more kits to process?
```

Respond with Y to this message only if you have additional software kits to install. Then, you will receive a request to insert the first floppy from another distribution kit.

If you type N, you receive the message:

```
Please place the system console floppy in the console drive.
```

You should immediately restore the standard console volume (removed at the beginning of the update procedure) to the console drive. After you have replaced the system console floppy, you receive the message:

```
Are you ready to continue?
```

Type Y. The console volume is automatically mounted. You receive the message:

```
     Requested update sequence is complete.
```

The VAX-11 PL/I compiler is now installed and can be invoked by the PLI command.

You may now enter the following commands to purge previous versions, if any, of files created or modified by the compiler:

```
     $ PURGE [SYSLIB]STARLET.OLB
     $ PURGE [SYSEXE]PLIG.EXE
     $ PURGE [SYSHLP]HELPLIB.HLB
     $ PURGE [SYSLIB]PLISYSDEF.TLB
```

If a previous version of the compiler was installed as a known image, you should run the INSTALL utility and replace the old version of the compiler with the new version, as follows:

```
     $ RUN SYS$SYSTEM:INSTALL
     INSTALL>PLIG.EXE/REPLACE
```

If the compiler has not been installed, and you wish to install it, read the next section. Otherwise, the installation of the VAX-11 PL/I compiler is now complete.

1.8 INSTALLING THE VAX-11 PL/I COMPILER AS A KNOWN IMAGE

If you have installed the compiler on a system that will be using PL/I extensively, you can reduce system overhead and memory requirements by installing the compiler as a known image.

To install the compiler as a known image, use the INSTALL utility while you are still logged in under the system manager's account. The following procedure describes how to install the PL/I compiler. For additional details on using the INSTALL utility, see the VAX/VMS System Manager's Guide.

1. Enter the following command to invoke the INSTALL utility:

   ```
   $ RUN SYS$SYSTEM:INSTALL
   ```

2. To determine whether there are enough global pages available to install the VAX-11 PL/I compiler, enter the command:

   ```
   INSTALL>/GLOBAL
   ```

   This command responds with a list of all currently known global sections. At the end of this list, there is a summary of the number of global sections used, the number of global pages used, and the number of global pages unused. The VAX-11 PL/I compiler requires approximately 584 pages. If there are fewer than 584 global pages unused, do not attempt to install PL/I as a known image. To increase the number of available global pages, you can delete an existing known image or increase the GBLPAGES system parameter with the SYSGEN utility.

3. Exit from the INSTALL utility:

   ```
   INSTALL>CTRL/Z
   ```

4. To determine whether there are enough global sections available to install the VAX-11 PL/I compiler, run the SYSGEN utility and enter the commands shown below:

   ```
   $ RUN SYS$SYSTEM:SYSGEN
   SYSGEN>  USE CURRENT
   SYSGEN>  SHOW GBLSECTIONS
   ```

   The first number displayed is the maximum number of global sections. Compare this number with the number of global sections in use displayed by INSTALL. The VAX-11 PL/I compiler requires six global sections -- thus, if the INSTALL amount plus six exceeds the system maximum, you must increase the number of global sections before you can install PL/I. To increase the amount, you must increase the GBLSECTIONS parameter with the SYSGEN utility.

   If the INSTALL amount plus six is less than the system maximum, you can continue.

5. After you have verified that system parameters are large enough to accommodate PL/I, exit from the SYSGEN utility and invoke the INSTALL utility, as follows:

   ```
   SYSGEN>  EXIT
   $ RUN SYS$SYSTEM:INSTALL
   ```

6. To install PL/I, enter the command:

   ```
   INSTALL>PLIG.EXE/OPEN/SHARED
   INSTALL>EXIT
   ```

It is recommended that you perform these steps on a system that has just been bootstrapped, since it is less likely that the available space in the global page table will be fragmented. If errors occur, consult the VAX/VMS System Manager's Guide.

To ensure that the VAX-11 PL/I compiler is installed as a known image each time the system is bootstrapped, include the command to install PL/I in the system startup file, SYSTARTUP.COM. This command file should have the command:

```
$ RUN SYS$SYSTEM:INSTALL
```

followed by the names of known images to be installed. You can insert the following command in this list to install PL/I:

```
PLIG.EXE/OPEN/SHARED
```

Each time the system is booted, the VAX-11 PL/I compiler will be installed.

1.9 ERROR CONDITIONS

If the installation procedure fails for any reason, the following message is displayed:

```
*** VAX-11 PL/I Installation Procedure FAILED ***
```

Then, the installation command procedure attempts to determine the reason for the error. If it detects any of the following, it displays a message notifying you of that fact:

- An insufficient system virtual page count parameter.
- An insufficient process paging file quota.
- An insufficient process working set quota.
- An insufficient system maximum working set.

If you are notified that any of these conditions exist, you should take the appropriate corrective action, as described in the message.

To change a system parameter, or to increase an authorized quota value, you may need to contact your installation system manager.

CHAPTER 2

ERRATA

The PL/I online HELP files contain information that is not contained elsewhere in the PL/I documentation or that summarizes current restrictions on the use of the compiler. To obtain this information, type:

```
     $ HELP PLI ERRATA
```

or

```
     $ HELP PLI ERRATA *
```

To obtain a line printer listing of this file, type:

```
     $ DEFINE/USER SYS$OUTPUT LP:ERRATA.OUT
     $ HELP PLI ERRATA *
```

2.1 HOW TO ADD PL/I HELP TO EDT'S HELP FILES

The VAX-11 installation procedure inserts PL/I HELP file module into the system help library, HELPLIB. This HELP module can also be inserted into EDT's HELP files for programmers who use EDT to develop PL/I programs. The following commands must be executed by a user who has system manager privileges:

```
     $ LIBRARY/HELP/EXTRACT=PLIHELP/OUT=PLIHELP SYS$HELP:HELPLIB
     $ LIBRARY/HELP/INSERT SYS$HELP:EDTHELP SYS$DISK:[]PLIHELP
     $ DELETE SYS$HELP:PLIHELP.HLP;*
```

The PL/I HELP module adds approximately 150 disk blocks to the size of the EDT HELP file.

2.2 NEW MESSAGES

The PL/I compiler diagnostic and run-time messages listed on the following pages are messages that are either new to the compiler or whose text has been changed. You may wish to insert these pages in the front of Appendix B in the VAX-11 PL/I User's Guide.

New Diagnostic Messages

AGGMISMAT The source and target of an assignment are aggregates that do not match as required by the language rules.

Error. An assignment statement assigning the value of one array to another or one structure to another references arrays or structures that are not identical. This form of assignment is valid only when arrays with the same data type, number of dimensions, and extents are used and when structures with the same hierarchy and data types are used.

User Action. Verify that the references in the assignment refer to the correct aggregates. To selectively assign values to elements of an array or to structure members when the aggregates are not identical, you must assign elements or members values in separate assignment statements.

BADAGGARG "name" is an invalid array, structure, or area argument. Such an argument must be a variable reference, must not be enclosed in parentheses, and must exactly match the corresponding parameter.

Error. An array dimension or a structure declaration specified in a parameter descriptor does not match the corresponding dimension or structure of the variable specified in the procedure reference. For example, this error occurs if a parameter descriptor specifies a scalar value and the procedure reference specifies the corresponding argument with a reference to an array.

User Action. Determine whether the parameter descriptor correctly specifies the data type, dimensions, and structure of the required parameter. If so, correct the declaration of the corresponding argument or the corresponding argument reference. If the argument is specified correctly, correct the parameter descriptor. If the procedure is a non-PL/I procedure, use of the ANY attribute in the parameter descriptor may be appropriate.

BADDEFBAS The base reference specified for the DEFINED variable "name" is not a connected variable reference.

Error. A variable is declared with the DEFINED attribute and the variable specified in the DEFINED attribute is a variable whose storage is unconnected.

User Action. Correct the variable declaration so that it refers to a variable whose storage is connected.

CIRCDECL The declaration of "name" is circular. Some reference or expression in this declaration depends on the declaration itself, as in "DECLARE P POINTER BASED(P)".

Error. The compiler cannot resolve a reference.

User Action. Correct the reference so that its definition depends on some other variable.

DEFBASCLA The base reference specified for the DEFINED variable "name" is itself DEFINED or BASED.

Warning. The VAX-11 PL/I language does not allow a defined variable to be defined on a variable that is either based or defined and thus issues this warning message to inform you of the condition.

User Action. If you intended to define the variable on a based or defined variable, you need do nothing; the results are likely to work as expected.

DESCRIBIF Invalid use of the DESCRIPTOR built-in function.

Error. The DESCRIPTOR built-in function may only be used in an argument list to specify that an argument be passed by descriptor to a non-PL/I procedure; the corresponding parameter must be declared with the attributes ANY VALUE. This message indicates that you are using the built-in function in an inappropriate context. Note that the PL/I compiler passes character strings and arrays by descriptor by default and so you need not specify the manner in which they are to be passed.

This message is also issued when the DESCRIPTOR built-in function is specified with more than one argument or with an argument that is noncomputational, is pictured, or is an array of noncomputational or pictured data.

User Action. Remove the reference to the built-in function from the statement.

MISDEFINE The DEFINED variable "name" does not match its specified base reference as required by the rules for defining.

Warning. The data type attributes of a variable with the DEFINED storage class do not match the attributes of the variable on which it is defined.

User Action. If the mismatch is acceptable, you need do nothing. Otherwise, correct the declaration so that it specifies the correct attributes.

NESTEDPSV This statement contains a nested pseudovariable reference to "name". Nested use of pseudovariables as in "SUBSTR(UNSPEC(x),3,1) = '0'b;" is invalid.

Error. One of the built-in functions that may be used on the left-hand side of an assignment contains a reference to another pseudovariable.

User Action. Correct the expression so that it does not contain a nested pseudovariable reference.

NODATATYP "Name" is declared without a data type. The default data type FIXED BIN(31) has been supplied.

Warning. The indicated name has been declared but was declared without a data type attribute. The compiler provides the default data type of FIXED BINARY(31).

User Action. If you want the variable to have the default data type, you need do nothing. Otherwise, correct the declaration of the variable so that it has a data type attribute.

NOLABSUB This statement contains a reference to an undefined subscripted label array element "name(subscript)".

Error. The compiler cannot resolve a reference to the indicated subscripted program label.

User Action. Verify that the label is correctly specified and that if it is an element of an array of label constants, the label is properly subscripted in the source program.

NOTCONNECT This statement contains an invalid reference to an unconnected array.

Error. A member of a structure that has the dimension attribute is referenced in an invalid context. Since the storage of such an array is not contiguous, the array cannot be referenced in an ADDR built-in function or in a STRING pseudovariable, as the base reference of a DEFINED variable, as an argument of a procedure, as the source or target in a record I/O statement, or in any other context in which aggregates must be connected.

User Action. If possible, change the declaration of the structure so that the array that caused the error becomes a connected array. Otherwise, do not reference the array in the context that caused the error.

PROLOGVAL The value of the AUTOMATIC or DEFINED variable
"name" is used in an extent
expression or initial-value expression in the declaration of
an AUTOMATIC or DEFINED variable in the same block.

Warning. Dependency on the value of an automatic or defined variable that is declared in the same block is not allowed. When the compiler generates code for the variable, the referenced automatic or defined variable may not have been allocated or initialized.

User Action. Correct the extent or initial value expression so that it specifies a constant value, a static variable, or parameter, so that it references a variable that is defined in a containing block.

TOTALDIM More than 8 dimensions have been specified for the array
"name". This may include
dimensions inherited from containing structures.

Error. An array has been declared with more than eight dimensions, or an array is declared within a structure and the sum of the array's dimensions and those of the structure exceeds a total of eight.

User Action. Correct the declaration of the array.

New Run-Time Message

AUTOINIT Error in automatic initialization.

Fatal. An automatic variable declared with the INITIAL attribute cannot be initialized. This error can be caused by a negative repetition factor in an initial list, or when too few or too many values are specified in the initialization of an array.

User Action. Correct the initial list.

CHAPTER 3

SYSTEM MANAGEMENT CONSIDERATIONS

This chapter contains information of interest to system managers and operators who are responsible for maintaining the VAX-11 PL/I compiler.

System managers and operators who have responsibility for installing and maintaining the VAX-11 PL/I compiler may find information about the operation of the compiler in the following VAX-11 PL/I documentation:

- Introduction to VAX-11 PL/I
- VAX-11 PL/I Encyclopedic Reference
- VAX-11 PL/I User's Guide

Related system operation and management information can be found in the following VAX-11/VMS documents:

- VAX/VMS System Manager's Guide
- VAX/VMS Operator's Guide
- VAX-11 Software Installation Guide

For a complete list of all VAX/VMS documents and the order numbers of documents, see the VAX-11 Information Directory.

3.1 COMPILER REQUIREMENTS AND QUOTAS FOR PL/I USERS' ACCOUNTS

This section summarizes recommended values to specify in the accounting file for PL/I users.

3.1.1 Compiler Requirements

The VAX-11 PL/I compiler requires approximately 1200 disk blocks. This space requirement includes the compiler, run-time library, HELP text, and the default INCLUDE library.

The VAX/VMS system must have a minimum of 512 kilobytes of physical memory.

3.1.2 Account Quotas

This section summarizes recommended account quotas for VAX-11 PL/I users.

3.1.2.1 Open File Quota - It is recommended that the open file quota be set at a minimum of 16. This will allow for the compiler's source, object, and listing files; a maximum of four INCLUDE files; libraries; and SYS$INPUT, SYS$OUTPUT, and SYS$ERROR devices.

3.1.2.2 Paging File Quota - A process that executes the PL/I compiler requires a minimum paging file quota of 4000 pages.

3.1.2.3 Working Set Size Quota - The minimum working set size (WSQUOTA) for a process must be at least 250 pages; better performance results if the minimum working set size is set between 400 and 500 pages.

3.2 PROBLEM DETERMINATION AND REPORTING

If an error occurs during the compilation or execution of a PL/I program and there is reason to believe that the error occurs because of a problem in the PL/I compiler or run-time system, you should report the problem to DIGITAL using the SPR (Software Problem Report) form.

The following diagnostic messages indicate an internal error that should not occur:

- Compiler messages with the identification codes BADOP or BUGCHECK.

- Run-time messages with the identification codes ENVPARM, FILEIDENT, INVFORMAT, INVNUMOPT, PARM, READOP, and VIRMEMDEAL.

You may also suspect an error in the compiler or run-time system when a program or a compilation produces unexpected results.

When you prepare to submit an SPR, it is recommended that you do the following:

1. Provide as complete a description of the problem as possible, trying not to overlook any details.

2. Reduce the problem to as small a size as possible, preferably to a program unit that is 200 lines or less of source code.

3. Remember to include listings of INCLUDE files or library modules used in the compilation, as well as any command files, listings, linker maps, data files required, and so on.

4. Provide a listing of the program's output.

CHAPTER 4

MIGRATION NOTES

This chapter contains notes and comments about migration issues. In particular, it lists and describes keywords and functions of the PL/I language that are available in other implementations of PL/I, but that are not included in either the General Purpose (G) Subset or in VAX-11 PL/I. For a description of the General Purpose Subset and a summary of VAX-11 PL/I deviations from the subset, see the VAX-11 PL/I Encyclopedic Reference.

The information in this chapter is not intended to represent either a complete or a formal description of migration issues. The information in this chapter is presented informally, and has not been subjected to extensive testing and verification.

4.1 KEYWORDS NOT SUPPORTED

Table 4-1 summarizes PL/I keywords used in other implementations of PL/I that are not used in VAX-11 PL/I. Table 4-1 does not include ENVIRONMENT keywords, implementation-specific language extensions, or preprocessor statements. (VAX-11 PL/I supports only the %INCLUDE and %REPLACE preprocessor statements.)

Table 4-1
PL/I Keywords Not Supported by VAX-11 PL/I

Keyword	Use
ADD	built-in function
AFTER	built-in function
ALL	built-in function
ALLOCATION	built-in function
ANY	built-in function
AREA	condition
ATTENTION	condition
BACKWARDS	attribute, option of OPEN statement
BEFORE	built-in function
BY NAME	option of assignment statement
C	format item
CALL	option of INITIAL attribute
CASE	option of DO statement
CHECK	statement, condition, condition prefix
COMPLETION	built-in function, pseudovariable
COMPLEX	attribute, built-in function, pseudovariable
CONDITION	attribute, condition
CONJG	built-in function

(Continued on next page)

Table 4-1 (Cont)
PL/I Keywords Not Supported

Keyword	Use
CONNECTED	attribute
CONSTANT	attribute
CONTROLLED	attribute
CONVERSION	condition, condition prefix
COPY	option of GET statement
COUNT	built-in function
CURRENTSTORAGE	built-in function
DATA	stream I/O transmission mode
DATAFIELD	built-in function
DECAT	built-in function
DEFAULT	statement
DELAY	statement
DESCRIPTORS	option of DEFAULT statement
DIMENSION	attribute keyword
DISPLAY	statement
DOT	built-in function
EMPTY	built-in function
ERF	built-in function
ERFC	built-in function
EVENT	attribute, option of several statements
EVERY	built-in function
EXCLUSIVE	attribute
EXIT	statement
FETCH	statement
FOFL	abbreviation of FIXEDOVERFLOW condition name
FORMAT	attribute
GENERIC	attribute
HALT	statement
HIGH	built-in function
IGNORE	option of READ statement
IMAG	built-in function, pseudovariable
IN	option of ALLOCATE and FREE statements
IRREDUCIBLE	attribute
LEAVE	statement
LIKE	attribute
LIST	option of OPEN statement
LOCAL	attribute
LOCATE	statement
LOW	built-in function
MEMBER	attribute
NAME	condition
NOCHECK	statement, condition prefix
NOCONVERSION	condition prefix
NOFIXEDOVERFLOW	condition prefix
NOLOCK	option of READ statement
NONE	option of DEFAULT statement
NONVARYING	attribute
NOOVERFLOW	condition prefix
NOSIZE	condition prefix
NOSTRINGRANGE	condition prefix
NOSTRINGSIZE	condition prefix
NOSUBSCRIPTRANGE	condition prefix
NOZERODIVIDE	condition prefix
OFL	abbreviation of OVERFLOW condition
ONCHAR	built-in function, pseudovariable

(Continued on next page)

Table 4-1 (Cont)
PL/I Keywords Not Supported

Keyword	Use
ONCOUNT	built-in function
ONFIELD	built-in function
ONLOC	built-in function
ONSOURCE	built-in function, pseudovariable
ORDER	option of BEGIN and PROCEDURE statements
OTHERWISE	clause of select-group
OVERFLOW	condition prefix
PARAMETER	attribute
PENDING	condition
POLY	built-in function
PRECISION	built-in function
PRIORITY	option of CALL statement, built-in function, pseudovariable
PROD	built-in function
RANGE	option of DEFAULT statement
REAL	attribute, built-in function, pseudovariable
RECORD	condition
REDUCIBLE	attribute
REENTRANT	option of OPTIONS option
REFER	option of BASED attribute
RELEASE	statement
REORDER	option of BEGIN and PROCEDURE statements
REPEAT	built-in function
REPLY	option of DISPLAY statement
REVERSE	built-in function
SAMEKEY	built-in function
SELECT	statement
SIZE	condition, condition prefix
SNAP	option of ON and PUT statements
SOME	built-in function
STATUS	built-in function, pseudovariable
STORAGE	built-in function
STRINGRANGE	condition, condition prefix
STRINGSIZE	condition, condition prefix
STRUCTURE	attribute
iSUB	dummy variable of DEFINED attribute
SUBTRACT	built-in function
SUBSCRIPTRANGE	condition, condition prefix
SUM	built-in function
SYSTEM	option of ON and DECLARE statements
TAB	option of OPEN statement
TASK	attribute, option of OPTIONS option, option of CALL statement
TRANSIENT	attribute
TRANSMIT	condition
UFL	abbreviation of UNDERFLOW condition name
UNALIGNED	attribute
UNBUFFERED	attribute, option of OPEN statement
UNDF	abbreviation of UNDEFINEDFILE condition name
UNLOCK	statement
UNTIL	option of DO statement
WAIT	statement
WHEN	option in GENERIC declaration, clause of select-group
ZDIV	abbreviation of ZERODIVIDE condition name

4.2 MISCELLANEOUS DIFFERENCES

The list below summarizes a number of minor differences between VAX-11 PL/I and other PL/I compilers that require you to edit and modify your source files to avoid compilation errors. In some cases, differences require reprogramming.

1. The characters @ and # are not allowed in identifiers; thus, you must change all identifiers that contain either of these characters.

2. All names (except internal procedure and label constants) must be explicitly declared. There is no "I through N rule" that provides an implicit declaration of FIXED BINARY to undeclared names. In fact, the VAX-11 PL/I compiler defaults all undeclared names to FIXED BINARY and issues a warning message to that effect.

3. If the attribute FLOAT is specified and neither the attributes BINARY nor DECIMAL is specified with it, the VAX-11 PL/I compiler provides a default of BINARY. This should not present a problem; however, if a precision was specified under the assumption that the default floating-point base was DECIMAL, overflow conditions can result if you do not correct the declaration.

4. Internal entry constants and subscripted label constants must not be explicitly declared: in VAX-11 PL/I, these names are implicitly declared by their appearance. You must remove the declarations from the source file.

5. ON-units for conditions that are not supported must be reprogrammed. For example, ON-units for SIZE, CONDITION, RECORD, and TRANSMIT should be modified so that they are invoked for the ERROR condition, which is the condition that is signaled in VAX-11 PL/I for any of these errors.

6. String constants that are declared with replication factors must be edited; replication factors are not supported. To replicate a constant, use a COPY function in an assignment.

7. Variables with the BASED attribute cannot be initialized in an INITIAL attribute. You must use assignment statements to initialize them.

8. The ALLOCATE and FREE statements permit only a single variable to be allocated and freed in a statement.

9. The OPEN and CLOSE statements permit only a single file to be opened or closed in a statement.

10. The ALIGNED attribute may not be specified for a structure in which string variables are to be aligned. The attribute must be specified in the declaration of each variable that is to be aligned.

11. Parameters may not be passed directly to main procedures from the command stream. For examples of techniques for passing values or data to a main procedure, see the VAX-11 PL/I User's Guide.

12. When the division operator (/) is used for fixed-point binary variables, the compiler issues a warning message to notify you that the precision is being lost. To avoid this message, rewrite the statement so that it references the DIVIDE built-in function to perform the division.

13. Expressions are not allowed in format items. The "arguments" of a format item, such as the field width, must be specified using integer constants. The sections of a program that compute expressions in format items usually must be rewritten.

4.2.1 Implicit Conversions

The VAX-11 PL/I compiler issues warning-level messages when it performs implicit conversions between arithmetic and string data types and between bit-string and character-string data types. It issues these messages for all such conversions, not just those excluded by the G subset.

You can avoid the messages by compiling your programs with the /NOWARNINGS qualifier. Or, you can edit your program, locate the occurrences of implied conversions, and change them to explicit conversions, as follows:

- Arithmetic to character -- use the CHARACTER built-in function.
- Arithmetic to bit string -- use the BIT built-in function.[1]
- Bit-string to arithmetic -- use the BINARY built-in function
- Bit-string to character -- use the CHARACTER built-in function
- Character-string to arithmetic -- use the BINARY, DECIMAL, FIXED, or FLOAT built-in function, according to the target data type
- Character-string to bit-string -- use the BIT built-in function

[1] Note that this conversion is based on the way bit strings are printed by PUT LIST (the first bit of the string is the high-order bit if the printed string is viewed as a binary integer) rather than on the internal representation (the first bit of the string is then in the low-order digit position in memory).

4.2.2 Simulating the REFER Option for BASED Structures

Some implementations of PL/I support the REFER option in the declaration of based structures. For example:

```
        DCL  01 STRUCTURE BASED (P),
                02 SIZE       FIXED BINARY (31),
                02 STRING     CHARACTER(STR_SIZE REFER (STRUCTURE.SIZE));
        DCL  P                POINTER;
        DCL  STR_SIZE         FIXED BINARY (31);
        .
        .
        STR_SIZE=132;
        ALLOCATE STRUCTURE SET (P);
```

In this example, STRUCTURE is allocated with a fixed-length character string string 132 bytes long. STRUCTURE.SIZE is updated to contain the length of the string, 132. All subsequent references to the size of STRUCTURE.STRING use this value.

In VAX-11 PL/I, the results are somewhat different:

```
        DCL  01 STRUCTURE BASED (P),
                02 SIZE       FIXED BINARY (31),
                02 STRING     CHARACTER(STR_SIZE);
        DCL  P                POINTER;
        DCL  STR_SIZE         FIXED BINARY (31);
        .
        .
        STR_SIZE=132;
        ALLOCATE STRUCTURE SET (P);
        STRUCTURE.STRING = STR_SIZE;
```

In this example, STR_SIZE will always be referred to when PL/I needs to determine the size of STRUCTURE.STRING, and not to the contents of STRUCTURE.SIZE, as is the case when the REFER option is used.

To simulate the effect of the REFER option in a based structure, you can use the following technique:

```
        DCL  01 STRUCTURE BASED (P),
                02 SIZE       FIXED BINARY (31),
                02 STRING     CHARACTER (SIZE);
        DCL  P                POINTER;
        DCL  SPACE(-3:STR_SIZE) CHARACTER (1) BASED (P);
        DCL  STR_SIZE         FIXED BINARY (31);
        .
        .
        STR_SIZE=132;
        ALLOCATE SPACE SET (P);
        STRUCTURE.SIZE = STR_SIZE;
```

In this example, the length of STRING is self-defining, based on the value of SIZE. The array SPACE allocates four bytes (elements -3 through 0) for the FIXED BINARY (31) SIZE member. The remaining elements account for the length of the string. After SPACE is allocated, its first four bytes are set to the length of the string. Thus, references to STRUCTURE.STRING will refer to the length in STRUCTURE.SIZE.

Note that this technique is required only when ALLOCATE statements are used to associate storage with STRUCTURE. If the ADDR built-in function or some other method is used, the contents of the first four bytes will always determine the length of the following string.

VAX-11 PL/I
Installation and
System Management Guide
Order No. AA-J179A-TE

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

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

Did you find errors in this manual? If so, specify the error and the page number.

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

□ Assembly language programmer
□ Higher-level language programmer
□ Occasional programmer (experienced)
□ User with little programming experience
□ Student programmer
□ Other (please specify)____________________

Name______________________________ Date______________

Organization____

Street____

City____________________ State__________ Zip Code__________
or
Country

— — — Do Not Tear - Fold Here and Tape —

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

BSSG PUBLICATIONS ZK1-3/J3-5
DIGITAL EQUIPMENT CORPORATION
110 SPIT BROOK ROAD
NASHUA, NEW HAMPSHIRE 03061

— — — Do Not Tear - Fold Here —