VAX FORTRAN HPO
Installation Guide/Release Notes

AA–PC38A–TE

May 1990

This manual contains instructions for installing VAX FORTRAN HPO on the VMS operating system. It also includes the release notes for the initial release of the product.

This manual applies to VAX FORTRAN HPO Version 1.0 and all future maintenance updates to that version.

Revision/Update Information:	This is a new manual.
Operating System and Version:	VMS Version 5.4 or higher. Future maintenance releases may require higher versions.
Software Version:	VAX FORTRAN HPO Version 1.0

digital equipment corporation
maynard, massachusetts

First Printing, May 1990

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.

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

Restricted Rights: Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

© Digital Equipment Corporation 1990.

All Rights Reserved.
Printed in U.S.A.

The postpaid Reader's Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	UNIBUS
DEC/CMS	EduSystem	VAX
DEC/MMS	IAS	VAXcluster
DECnet	MASSBUS	VMS
DECsystem–10	PDP	VT
DECSYSTEM–20	PDT	
DECUS	RSTS	
DECwriter	RSX	digital™

ZK-5517

This document was prepared using VAX DOCUMENT, Version 1.2.

Contents

1 Preparing to Install VAX FORTRAN HPO

1.1 Required Operating System Components 1–2
1.2 Prerequisite and Optional Software 1–2
1.3 License Registration 1–3
1.4 Installation Procedure Requirements 1–3
1.4.1 Installation Account Privileges and Disk Space 1–4
1.4.2 System Parameters 1–4
1.4.2.1 Calculating the Values for GBLPAGES and GBLSECTIONS 1–5
1.4.2.2 Changing System Parameter Values with AUTOGEN 1–6
1.4.3 VMSINSTAL Requirements 1–6
1.4.4 Backing Up Your System Disk 1–7
1.4.5 VAX FORTRAN HPO Requirements 1–7

2 Installing VAX FORTRAN HPO

2.1 The Installation Procedure 2–1
2.2 Error Recovery 2–9

3 After Installing VAX FORTRAN HPO

3.1 Running the Installation Verification Procedure Separately 3–1
3.2 VAXcluster Considerations 3–2
3.3 User Account Requirements 3–3
3.3.1 User Account Privileges 3–3
3.3.2 User Account Quotas 3–3
3.4 Enhancing VAX FORTRAN HPO Performance 3–3
3.4.1 Tuning Your System 3–3

3.4.2 Installing VAX FORTRAN HPO as a Shared Image 3–3
3.5 Determining and Reporting Problems 3–4
3.6 VAX FORTRAN Version 5 Maintenance Updates 3–5

4 Release Notes

4.1 Overview of VAX FORTRAN HPO 4–1
4.1.1 New Qualifiers and Qualifier Parameters 4–2
4.1.1.1 /ASSUME Qualifier 4–3
4.1.1.2 /BLAS Qualifier 4–4
4.1.1.3 /CHECK Qualifier (ASSERTIONS Parameter) 4–6
4.1.1.4 /DESIGN Qualifier 4–6
4.1.1.5 /DIRECTIVES Qualifier 4–6
4.1.1.6 /HPO Qualifier 4–6
4.1.1.7 /PARALLEL Qualifier (AUTOMATIC and MANUAL Parameters) 4–7
4.1.1.8 /SHOW Qualifier (DATA_DEPENDENCES and LOOPS Parameters) 4–8
4.1.1.9 /SYNCHRONOUS_EXCEPTIONS Qualifier 4–9
4.1.1.10 /TERMINAL Qualifier 4–9
4.1.1.11 /VECTOR Qualifier 4–10
4.1.1.12 /WARNINGS Qualifier (ALIGNMENT and TRUNCATED_SOURCE Parameters) 4–10
4.1.2 Assertion Declarations 4–11
4.2 VAXset Program Design Facility 4–12
4.2.1 /DESIGN Compile Command Qualifier 4–13
4.2.1.1 Allowed Contexts for Placeholders 4–15
4.3 DECwindows Compile Verb Support in FileView 4–16
4.4 VAX FORTRAN Performance Guide 4–17
4.5 Compiler Deficiencies in VAX FORTRAN Version 5 and VAX FORTRAN HPO 4–17
4.5.1 Restrictions on Standards Checking 4–17
4.5.2 Using /DML and /PARALLEL Qualifiers 4–18
4.5.3 Maintaining Password Security 4–18
4.5.4 Status of Bug Fixes 4–18
4.6 Additional Information Needed for FORTRAN Parallel Processing 4–18
4.6.1 Tuning Issues for Parallel Processing 4–18
4.6.1.1 System Parameters 4–19
4.6.1.2 User Parameters 4–19
4.6.1.3 Working Set Parameters 4–19
4.6.2 Restrictions on Parallel-Processing Applications 4–20
4.6.3 Linking Against FORTRAN Parallel-Processing Support 4–20
4.6.3.1 Linking with the /NOSYSSHR Qualifier 4–20

4.6.4 Using PPL$ Run-Time Library Routines 4–20
4.7 Compatibility Issues Involving VAX FORTRAN HPO 4–20
4.7.1 Compilation Compatibility 4–21
4.7.2 Source Compatibility 4–21
4.7.3 Object Compatibility 4–21
4.7.4 Image Compatibility 4–22
4.8 Source Program Diagnostic Messages 4–22
4.8.1 Messages Supporting Dependence Analysis of DO Loops 4–22
4.8.2 Messages Supporting Vectorization of Specific Statements . . . 4–24
4.8.3 Messages Supporting Vectorization 4–25
4.8.4 Messages Supporting the VAXset Program Design Facility . . . 4–26

A Sample Installation

Tables

1–1 Prerequisite and Optional Software 1–2
1–2 Disk Space Requirements 1–4
1–3 Process Quotas for the Installing User Account 1–7

1

Preparing to Install VAX FORTRAN HPO

This chapter discusses the preparations and requirements necessary for installing VAX FORTRAN HPO.

This manual applies to Version 1.0 of VAX FORTRAN HPO and all subsequent maintenance releases up to the next major release of the product. (A major version number is represented by the digits to the left of the decimal point, and the maintenance update version number is indicated by the digits to the right of the decimal point.)

Keep this manual with your installation kit. You will need it to install maintenance updates or to reinstall VAX FORTRAN HPO for any other reason.

Your bill of materials (BOM) and indented bills report (BIL) specify the number and contents of your media. Be sure to verify the contents of your kit with this information. If your kit is damaged or if you find that parts of it are missing, contact your Digital representative.

VAX FORTRAN HPO provides online release notes. Digital strongly recommends that you read the release notes before proceeding with the installation. For information on accessing the online release notes, see Section 2.1 (step 5). The online release notes contain information that is likely to change from release to release.

Note that hardcopy release notes are contained in Chapter 4. The hardcopy release notes cover only information that will be included in the *VAX FORTRAN Language Reference Manual* and *VAX FORTRAN User Manual* when they are revised.

1.1 Required Operating System Components

VAX FORTRAN HPO Version 1.n requires VMS Version 5.4 or higher.

The VMS operating system comes with a variety of support options, or classes. Classes include such features as networking and RMS journaling. To use VAX FORTRAN HPO, your system should be running a version of VMS that includes the following classes:

- Programming support
- Utilities

For a complete list of the required classes, see the Software Support Addendum (SSA), which comes with the VAX FORTRAN HPO Software Product Description (SPD).

1.2 Prerequisite and Optional Software

Table 1–1 provides information on the software you must have installed on your system before installing VAX FORTRAN HPO. The table also includes information about optional software that you can use together with VAX FORTRAN HPO. The System Software Addendum (SSA) contains a complete list of prerequisite and optional software and their required version numbers.

Table 1–1 Prerequisite and Optional Software

Prerequisite Products	Purpose
VAX FORTRAN Version 5.4	Compiler for /NOHPO qualifier
VAX Language-Sensitive Editor Version 3.0	Display of data dependence diagnostics

Optional Products	Purpose
VAX Performance Coverage Analyzer Version 3.0	Analysis of program performance
VAX Common Data Dictionary Version Version 3.4	DICTIONARY statement support
VAX Source Code Analyzer Version 2.0	Program design and analysis
Digital Extended Math Library Version 1.0	High-performance mathematical routines

1.3 License Registration

You must register the VAX FORTRAN HPO license through the VMS License Management Facility (LMF) in accordance with the license agreement signed by your site. The license registration information you need is contained in the Product Authorization Key (PAK) shipped with each software product. The PAK is a paper certificate containing information about the license to run a particular software product.

You must register three software licenses through LMF when you install VAX FORTRAN HPO:

- A VAX FORTRAN license
- A VAX FORTRAN HPO license
- A VAX Language-Sensitive Editor (LSE) license

If you install this product and register only the VAX FORTRAN license, you will be able to use the compiler, but you will not be able to use the new features provided by VAX FORTRAN HPO and VAX LSE for vector processing or automatic parallelism support.

During the installation, the system asks whether you have registered the VAX FORTRAN HPO license and loaded the appropriate PAK. You must register and load your license for VAX FORTRAN HPO *before* you start the installation.

To register a license under VMS, first log in to the system manager's account, SYSTEM. You then have a choice of two ways to perform the registration:

- Invoke the SYS$UPDATE:VMSLICENSE.COM procedure. When it prompts you for information, respond with data from your PAK.
- At the DCL prompt, enter the LICENSE REGISTER command with the appropriate qualifiers that correspond to PAK information.

If you plan to use VAX FORTRAN HPO on more than one node in a VAXcluster, you will need to load all three licenses on the other nodes after you complete this installation. See Section 3.2.

For complete information on using LMF, see the manual on the License Management Utility in the VMS documentation set.

1.4 Installation Procedure Requirements

This section and those that follow discuss various requirements for installing VAX FORTRAN HPO.

The installation takes approximately 5 to 40 minutes, depending on your type of media, your system configuration, and the kit components chosen.

Installing VAX FORTRAN HPO and running the Installation Verification Procedure (IVP) on a standalone VAX 6000-300 system takes 20 minutes.

1.4.1 Installation Account Privileges and Disk Space

To install VAX FORTRAN HPO, you must be logged in to an account that has the SETPRV privilege or at least the following privileges:

- CMKRNL
- WORLD
- SYSPRV

Note that VMSINSTAL turns off BYPASS privilege at the start of the installation.

VAX FORTRAN HPO's requirements for free disk storage space are different during installation and after installation. Table 1–2 summarizes the storage requirements.

Table 1–2 Disk Space Requirements

Kit	Blocks During Installation	Blocks After Installation
VAX FORTRAN HPO Version 1.0	8,500	6,500

Disk space usage can be reduced by omitting certain components of VAX FORTRAN HPO during installation. See the description of step 7 in Chapter 2 for more information about component selection.

To determine the number of free disk blocks on the current system disk, enter the following command at the DCL prompt:

```
$  SHOW DEVICE SYS$SYSDEVICE
```

1.4.2 System Parameters

Installing VAX FORTRAN HPO requires certain system parameter settings.

The minimum number of free global pages and global sections needed for the installation depends on whether SYS$SYSTEM:FORTRAN-HPO.EXE was previously installed as a shared known image:

- If not previously installed as shared—1 global section and a sufficient number of contiguous free global pages to contain the file SYS$LIBRARY:DCLTABLES.EXE

- If previously installed as shared—2 global sections and approximately 1500 contiguous free global pages plus a sufficient number of contiguous free global pages to contain the file SYS$LIBRARY:DCLTABLES.EXE

These values represent the number of free global pages and global sections required for the installation, not the total number you need to run your system and other software.

Note *You must ensure that your system has the necessary global page and global section SYSGEN quotas for the installation. Failure to do so could cause the DCL tables to be corrupted in some situations.*

The following sections show you how to:

- Calculate values for the GBLPAGES and GBLSECTIONS system parameters
- Change parameter values with the VMS AUTOGEN command procedure

1.4.2.1 Calculating the Values for GBLPAGES and GBLSECTIONS

To install and run VAX FORTRAN HPO, you must have sufficient free global pages and global sections. You must first find out how many free global pages and sections you have on your system. Then use AUTOGEN if you need to increase the GBLPAGES and GBLSECTIONS system parameters.

Enter the following DCL command to determine the number of global pages required by SYS$LIBRARY:DCLTABLES.EXE:

```
$  DIR/SIZE SYS$LIBRARY:DCLTABLES.EXE  RETURN
```

This command returns the size (in blocks) of SYS$LIBRARY:DCLTABLES.EXE. As an approximation, use this figure as the number of global pages needed for the file.

You can use the WRITE command with the F$GETSYI lexical function to find the number of free contiguous global pages and free global sections. The following example shows how to get this information at your terminal (the default for SYS$OUTPUT):

```
$  WRITE SYS$OUTPUT F$GETSYI("CONTIG_GBLPAGES")
15848
$  WRITE SYS$OUTPUT F$GETSYI("FREE_GBLSECTS")
24
```

If the value of free global pages or global sections is less than the required value (see Section 1.4.2), you must increase the system parameter setting.

Section 1.4.2.2 describes the procedures for increasing these values using AUTOGEN.

1.4.2.2 Changing System Parameter Values with AUTOGEN

If you do not have enough free global pages or free global sections to install VAX FORTRAN HPO, you can either delete an existing known image using the VMS Install Utility or use the AUTOGEN command procedure to increase the GBLPAGES of GBLSECTIONS system parameters. AUTOGEN automatically adjusts values for parameters that are associated with the values you reset manually. To change system parameters with AUTOGEN, edit the following file:

```
SYS$SYSTEM:MODPARAMS.DAT
```

Use an editor to access the file. To change a parameter value listed in this file, delete the current value associated with that parameter and enter the new value.

To add a new parameter, add a line to the file that includes both the name of the parameter and its value. For example:

```
WSMAX = 8096
```

To modify incremental parameters such as GBLPAGES and GBLSECTIONS, use ADD_. The following example increases the global page setting by 2000:

```
ADD_GBLPAGES = 2000
```

After you make all your changes, exit from the editor and execute the AUTOGEN procedure to recalculate your system parameters. Enter the following command at the DCL prompt:

```
$  @SYS$UPDATE:AUTOGEN GETDATA REBOOT
```

When you specify REBOOT, AUTOGEN performs an automatic system shutdown and then reboots the system. Any users logged on to the system are immediately disconnected during the shutdown. The automatic reboot puts the new parameter values into effect.

The AUTOGEN Utility automatically adjusts some of the SYSGEN parameters based on the consumption of resources since the last reboot. If you do not want to take advantage of this automatic adjustment, include the NOFEEDBACK qualifier on the AUTOGEN command line.

For more information about using AUTOGEN, see the VMS System Management Subkit.

1.4.3 VMSINSTAL Requirements

When you invoke VMSINSTAL, it checks the following:

- Whether you have set your default device and directory to SYS$UPDATE
- Whether you are logged in to a privileged account
- Whether you have adequate quotas for installation

- Whether any users are logged in to the system

Note that the VMSINSTAL requires that the installation account must have the minimum process quotas shown in the following table:

Table 1–3 Process Quotas for the Installing User Account

Account Quota	Value
ASTLM (AST limit)	24
BIOLM (Buffered I/O quota)	18
BYTLM (Buffered I/O byte count quota)	32,768
DIOLM (Direct I/O quota)	18
ENQLM (Enqueue quota)	200
FILLM (Open file limit)	100

User account quotas are stored in the file SYSUAF.DAT. Use the VMS Authorize Utility (AUTHORIZE) to verify and change user account quotas. For information on modifying account quotas, see the description of the Authorize Utility in the VMS System Management Subkit.

If VMSINSTAL detects any problems during the installation, it notifies you and asks if you want to continue the installation. In some instances, you can enter YES to continue. To stop the installation process and correct the situation, enter NO or press RETURN. Then correct the problem and restart the installation.

1.4.4 Backing Up Your System Disk

At the beginning of the installation, VMSINSTAL asks if you have backed up your system disk. Digital recommends that you do a system disk backup before installing any software.

Use the backup procedures that are established at your site. For details on performing a system disk backup, see the section on the Backup Utility in the VMS System Management Subkit.

1.4.5 VAX FORTRAN HPO Requirements

The VAX Vector Instruction Emulation Facility (VVIEF) is a programming development tool that allows you to execute vectorized applications on VAX systems that do not have vector hardware. See the *New and Changed Features* manual for VMS Version 5.4 for information on how to install VVIEF on your system.

Vector instructions in a vectorized application are automatically executed either in vector hardware or by the VVIEF, depending on the vector processing capabilities of the VAX system being used.

2

Installing VAX FORTRAN HPO

This chapter explains how to install VAX FORTRAN HPO:

- Section 2.1 contains a step-by-step description of the installation procedure.
- Section 2.2 covers error recovery considerations.

2.1 The Installation Procedure

The VAX FORTRAN HPO installation procedure consists of a series of questions and informational messages. See Appendix A for a sample installation.

To abort the installation procedure at any time, press CTRL/Y. When you press CTRL/Y, the installation procedure deletes all files it has created up to that point and exits. You can then start the installation again.

Defaults appear in brackets ([ ]) throughout the installation procedure.

1 Log in to a privileged account and set your default device and directory to SYS$UPDATE.

```
Username: SYSTEM
Password:
$ SET DEFAULT SYS$UPDATE
```

2 Invoke VMSINSTAL.

```
@SYS$UPDATE:VMSINSTAL saveset-name device-name OPTIONS N
```

To start the installation, invoke the VMSINSTAL command procedure from a privileged account, such as the SYSTEM account. VMSINSTAL is in the SYS$UPDATE directory. The elements that you specify when you invoke the VMSINSTAL command procedure are as follows:

saveset-name

The installation name for the component. For VAX FORTRAN HPO, use the following installation name:

```
FORTHPO01n
```

Where n represents the maintenance update of VAX FORTRAN HPO. Refer to the label on the VAX FORTRAN HPO distribution kit or the VAX FORTRAN HPO BOM or BIL for the maintenance update number of your kit.

device-name

The name of the device on which you plan to mount the media. For example, MUA0: is the device name used in examples in this document.

OPTIONS N

An optional parameter that indicates you want to see the question on release notes. If you do not include the OPTIONS N parameter, VMSINSTAL does not ask you about the release notes. You should review the release notes before proceeding with the installation in case they contain additional information about the installation. If you are restarting the installation and have already reviewed the release notes, you do not need to specify OPTIONS N.

Note that there are several other options you can select when you invoke VMSINSTAL.

- Auto_answer option (A): Initially creates a file that contains your answers to VMSINSTAL questions and prompts. You can then use the option (and the answer file) to save time during a reinstallation (typically after upgrading your system).
- Get save set option (G): Lets you store product save sets temporarily on a magnetic tape or in a disk directory.
- File log option (L): Logs all activity to the terminal during installation.
- Alternate root option (R): Lets you install the product to a system root other than that of the running system.

See the VMS documentation on software installation in the VMS System Management Subkit for detailed information on these options. If you specify more than one option, separate the options with commas (OPTIONS A,N).

The following example invokes VMSINSTAL to install VAX FORTRAN HPO and shows the system response. This example uses the OPTIONS N release note parameter.

```
$  @SYS$UPDATE:VMSINSTAL FORTHPO01n CSA1: OPTIONS N

       VAX/VMS Software Product Installation Procedure V5.4


It is dd-mmm-yyyy at hh:mm.
Enter a question mark (?) at any time for help.
```

If you do not supply either the product name or the device name, VMSINSTAL prompts you for this information later in the installation procedure. VMSINSTAL does not prompt you for any options, so be sure to include OPTIONS N on the VMSINSTAL command line to access the release notes during the installation.

When you invoke VMSINSTAL, it checks the following:

- Whether you logged in to a privileged account. Digital recommends that you install software from the system manager's account with your default device and directory set to SYS$UPDATE.
- Whether you have adequate quotas for installation. VMSINSTAL checks for the quota values shown in Table 1–3.

VMSINSTAL then checks to determine whether any users are logged in to the system. If users are logged in, you will be asked whether you want to continue the installation. If you want to continue, enter YES. If you want to stop the installation, press RETURN.

3 Confirm system backup.

```
* Are you satisfied with the backup of your system disk [YES]?
```

VMSINSTAL asks if you are satisfied with your system backup. You should always back up your system disk before performing an installation. If you are satisfied with the backup of your system disk, press RETURN. Otherwise, enter NO to discontinue the installation. After you back up your system disk, you can restart the installation.

4 Mount the media.

```
Please mount the first volume of the set on MUA0:.
* Are you ready?  YES
%MOUNT-I-MOUNTED, VAX FORTRAN HPO MOUNTED ON _$$MUA0:  (NODE 1)
The following products will be processed:
  VAX FORTRAN HPO V1.n

    Beginning installation of VAX FORTRAN HPO V1.n at 20:01

%VMSINSTAL-I-RESTORE, Restoring product saveset A...
```

You should now mount the first distribution volume on the device you specified when you invoked VMSINSTAL. The device name appears in the line preceding the question. VMSINSTAL then asks you if you are ready to continue with the installation.

If you respond YES to indicate that you are ready, VMSINSTAL displays a message that the media containing VAX FORTRAN HPO has been mounted on the specified device and that the installation has begun.

If you entered the wrong device name when you invoked VMSINSTAL and need to restart the installation, enter NO in response to the ''Are you ready?'' question. To abort the installation for other reasons, press CTRL/Y.

5 Select a release notes option.

```
Release Notes Options:

 1.  Display release notes
 2.  Print release notes
 3.  Both 1 and 2
 4.  Copy release notes to SYS$HELP
 5.  Do not display, print, or copy release notes

* Select option [2]:
```

If you specified OPTIONS N when you invoked VMSINSTAL, you are now asked to choose one of the five options for reviewing the release notes.

- If you select option 1, VMSINSTAL displays the release notes immediately on the console terminal. You can terminate the display at any time by pressing CTRL/C.
- If you select option 2, VMSINSTAL prompts you for the name of the print queue that you want to use:

  ```
  * Queue name [SYS$PRINT]:
  ```

  You can press RETURN to send the file to the default output print device or you can enter another queue name.
- If you select option 3, VMSINSTAL displays the release notes immediately on the console terminal and then prompts you for a queue name for the printed version.
- Option 4 causes the installation to copy the release notes to the help directory.
- Select option 5 if you have already reviewed the release notes and are restarting the installation.

Next, VMSINSTAL displays the following question:

```
* Do you want to continue the installation [N]?:  YES
%VMSINSTAL-I-RELMOVED, The product's release notes have been
                                    successfully moved to SYS$HELP.
```

To continue the installation, enter YES. Otherwise, press RETURN. In either case, the release notes are copied to a file in the SYS$HELP directory:

```
SYS$HELP:FORTHPO01.RELEASE_NOTES
```

After the installation, you can enter the following command to review the release notes through the Help facility:

```
$  HELP FORTRAN RELEASE_NOTES
```

Note *The name of the release notes file installed by VMSINSTAL consists of the current product name and version number. Do not delete release notes for previous versions of VAX FORTRAN HPO (if any).*

6 Purge files.

```
* Do you want to purge files replaced by this installation [YES]?
```

The directories SYS$HELP, SYS$LIBRARY, SYS$MESSAGE, SYS$SYSTEM, SYS$TEST, and VUE$LIBRARY may contain previous versions of VAX FORTRAN HPO files. These files are replaced during the installation with new files, but the old files are not automatically purged. Purging is recommended. In response to the purging prompt, press RETURN to purge the files or enter NO to keep them.

7 Select installation options.

The installation procedure allows you to select the components of the kit that you want to install. The choices are as follows:

- VAX FORTRAN HPO compiler
- FORSYSDEF.TLB system definition file
- DECwindows Compiler Interface

Select the components you wish to install by answering YES or NO to the following questions. If you want a more detailed description of any component, enter a question mark (?). After the explanation is displayed, you will be prompted again for a response.

```
* Do you want to install the compiler [YES]?
```

If you do not want to install the VAX FORTRAN HPO compiler, enter NO. If you enter NO, steps 8 and 9 are omitted.

```
* Do you want to build a new FORSYSDEF.TLB [YES]?
```

SYS$LIBRARY:FORSYSDEF.TLB is a library of VMS system interface definitions that can be used by FORTRAN programs. FORSYSDEF.TLB is built from a master set of definitions that is provided with VMS, and contains declarations of VMS system services, Run-Time Library routines, structures, and constants.

Because FORSYSDEF.TLB is generated from VMS definitions, you do not need to rebuild it unless this is a new installation or you have installed a new version of VMS since the last time you installed VAX FORTRAN HPO. (The installation procedure will make a recommendation on whether FORSYSDEF.TLB should be rebuilt.)

If you enter YES, the installation procedure builds a new system definition file, FORSYSDEF.TLB, based on the system definitions for the version of VMS installed on your system.

```
* Do you want to install the DECwindows Compiler Interface [YES]?
```

The DECwindows Compiler Interface (DWCI) for VAX FORTRAN HPO allows you to use a window-style interface to select compiler options when you compile a FORTRAN program. See the Release Notes Chapter 4 for more information on DWCI.

The DWCI option requires that the FileView directories [VUE$LIBRARY.SYSTEM] and [VUE$LIBRARY.USER] be present on the target disk. For more information on FileView, see the *VMS DECwindows User's Guide*.

The installation procedure then displays the components you have selected and issues the following prompt:

```
* Is this correct [YES]?
```

If you enter NO, the installation procedure reissues the prompts for selecting components.

As noted previously, if you answered NO to the first prompt in this step ("Do you want to install the compiler"), the following two steps (steps 8 and 9) are omitted.

8 Choose the Installation Verification Procedure (IVP) option.

```
* Do you want to run the IVP after the installation [YES]?
```

The installation procedure now asks if you want to run the IVP. The IVP for VAX FORTRAN HPO checks to be sure that the installation is successful. It is recommended that you run the IVP.

After VAX FORTRAN HPO is installed, you can run the IVP independently to verify that the software is available on your system. You might need to run the IVP after a system failure to make sure that users can access VAX FORTRAN HPO.

9 Respond to license registration queries.

VAX FORTRAN HPO supports the VMS License Management Facility (LMF). The installation procedure displays license information about VAX FORTRAN, VAX FORTRAN HPO, and VAX LSE. It asks whether you have registered and loaded your authorization key for each product. The following is an example of the information and the queries presented by the installation procedure:

```
  VAX FORTRAN HPO requires Product Authorization Keys (PAKs) for
  VAX FORTRAN, VAX FORTRAN HPO, and VAX Language-Sensitive Editor to
  be registered and loaded.   Answer the following three questions with
  YES if a PAK with the specified product name and a release date equal
  to or later than the date shown is registered and loaded, otherwise
  answer NO.

      Product:      FORTRAN
      Producer:     DEC
      Version:      n.n
      Release Date: dd-mmm-yyyy

* Does this product have an authorization key registered and
  loaded?:  YES [RETURN]

      Product:      FORTRAN-HPO
      Producer:     DEC
      Version:      n.n
      Release Date: dd-mmm-yyyy

* Does this product have an authorization key registered and
  loaded?:  YES [RETURN]

      Product:      LSE
      Producer:     DEC
      Version:      n.n
      Release Date: dd-mmm-yyyy

* Does this product have an authorization key registered and
  loaded?:  YES [RETURN]
```

If you have not registered and loaded your PAKs, you must answer NO to these questions. You have to register and load your PAKs to successfully complete the installation. If you have not done so, you should stop the installation, register and load your PAKs, and then restart the installation.

10 Read informational messages.

At this point, the installation procedure displays a number of informational messages that report on the progress of the installation. There are no further questions. If the installation procedure has been successful up to this point, VMSINSTAL moves the new or modified files to their target directories, updates help files, and updates DCL tables, if necessary. The new and modified files consist of the following:

```
SYS$HELP:FORTHPO01n.RELEASE_NOTES                [new]
SYS$SYSTEM:FORTRAN-HPO.EXE                       [new]
SYS$MESSAGE:FORTERR1.EXE                         [new]
SYS$MESSAGE:FORTERR2.EXE                         [new]
SYS$TEST:FORTRAN-HPO$IVP.COM                     [new]
SYS$LIBRARY:DCLTABLES.EXE                        [modified]
SYS$HELP:HELPLIB.HLB                             [modified]
SYS$LIBRARY:FORSYSDEF.TLB                        [new]
VUE$LIBRARY:FORTRAN$DWCI.EXE                     [new]
VUE$LIBRARY:FORTRAN$DWCI.UID                     [new]
VUE$LIBRARY:VUE$COMPILE.COM                      [new]
SYS$HELP:FORTRAN$DWCI.HLB                        [new]
```

The SYS$HELP:HELPLIB.HLB and SYS$LIBRARY:DCLTABLES.EXE files are updated to include the new versions of the VAX FORTRAN HPO help text and the FORTRAN command line definition.

Also, if you answered YES in response to the prompt that requests a file purge at step 6, previous versions of the VAX FORTRAN HPO files are now purged.

The end of the installation is indicated by the following message:

```
%VMSINSTAL-I-MOVEFILES, files will now be moved to their
                        target directories...
```

11 Observe the installation Verification Procedure (IVP).

If you chose to run the IVP (step 8), VMSINSTAL now invokes the IVP to verify that VAX FORTRAN HPO was installed successfully.

```
Beginning VAX FORTRAN HPO Installation Verification Procedure

Compiler image identification is FORTHPO V1.n-eee

Test compiler operation and error messages...

Compile, link, and run program with parallel processing features...

Compile, link, and run program with vector processing features...

Test FORSYSDEF.TLB system definitions...

Test I/O and Run-Time Library operation...
```

If VAX FORTRAN HPO was installed successfully, the IVP displays the following:

```
VAX FORTRAN HPO V1.n-eee TEST PASSED

End of VAX FORTRAN HPO Installation Verification Procedure
```

12 End the installation procedure.

```
    Installation of VAX FORTRAN HPO T1.n completed at hh:mm

    VMSINSTAL procedure done at hh:mm

$  LOGOUT
SYSTEM     logged out at dd-mmm-yyyy hh:mm:ss.ss
```

When the installation is complete, you can choose to install more products or to log out (as shown here). If you removed any media from the console drive before beginning the installation, you should replace it now.

Note that VMSINSTAL deletes or changes entries in the process symbol tables during the installation. Therefore, if you are going to continue using the system manager's account and you want to restore these symbols, you should log out and log in again.

2.2 Error Recovery

If errors occur during the installation itself or when the Installation Verification Procedure (IVP) is running, VMSINSTAL displays failure messages. If the installation fails, you see the following message:

```
%VMSINSTAL-E-INSFAIL, The installation of VAX FORTRAN HPO V1.n has failed.
```

If the IVP fails, you see these messages:

```
The VAX FORTRAN HPO V1.n Installation Verification Procedure failed.

%VMSINSTAL-E-IVPFAIL, The IVP for VAX FORTRAN HPO V1.n has failed.
```

Errors can occur during the installation if any of the following conditions exist:

- The operating system version is incorrect.
- A prerequisite software version is incorrect.
- Quotas necessary for successful installation are insufficient.
- System parameter values for successful installation are insufficient.
- The VMS help library is currently in use.
- The product license has not been registered and loaded.

For descriptions of the error messages generated by these conditions, see the VMS documentation on system messages, recovery procedures, and VMS software installation. If you are notified that any of these conditions exist, you should take the appropriate action as described in the message. For information on installation requirements, see Section 1.4.

If the installation fails, you must restart the installation procedure from step 2. If the installation fails due to an IVP failure, contact a Digital field service representative.

3

After Installing VAX FORTRAN HPO

VAX FORTRAN HPO is now installed and can be invoked by all users with the FORTRAN command.

The installation procedure modifies the DCL command table so that the FORTRAN command is recognized and processed. However, the previous command table is still in effect for those users who are currently logged in. All logged-in users who want to use the FORTRAN command must log out and log in again, or use the DCL command:

```
$  SET COMMAND /TABLES=SYS$LIBRARY:DCLTABLES
```

The following tasks can be performed after VAX FORTRAN HPO is installed:

- Running the Installation Verification Procedure separately.
- Making the product usable on a VAXcluster.
- Setting user account quotas.
- Enhancing performance.
- Reporting problems.

The following sections describe these tasks in detail.

3.1 Running the Installation Verification Procedure Separately

You usually run the Installation Verification Procedure (IVP) during installation. If you want to run the IVP separately to ensure the integrity of installed files if system problems should occur, execute the following command procedure:

```
$  @SYS$TEST:FORTRAN-HPO$IVP.COM.
```

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

```
VAX FORTRAN HPO VERSION V1.n TEST FAILED
```

3.2 VAXcluster Considerations

If you want to run VAX FORTRAN HPO on multiple nodes of a VAXcluster, first check to see that you have the appropriate software license (see Section 1.3). Then, perform the following steps *after* you install VAX FORTRAN HPO:

1. Issue the LICENSE LOAD command, as described in the *VMS License Management Utility Manual*, to activate the license on each node in the VAXcluster on which VAX FORTRAN HPO is to be executed.
2. Log in to the installation node and use the VMS INSTALL Utility to determine whether FORTRAN-HPO.EXE is a known image. This is done as follows:

   ```
   $ INSTALL
   INSTALL> LIST SYS$SYSTEM:FORTRAN-HPO.EXE.
   ```

   If the output of the preceding LIST command shows FORTRAN-HPO.EXE to be a known image, perform all of the following steps. If FORTRAN-HPO.EXE is not a known image, omit step 4b.
3. Log in to a node on the cluster.
4. Run the VMS Install Utility as described in Section 3.4.2, with the following exceptions:

   a. After invoking the VMS Install Utility, enter the following command:

      ```
      INSTALL> REPLACE SYS$LIBRARY:DCLTABLES.EXE
      ```

      This command updates the DCL tables. It should be issued on each node in the cluster, regardless of whether you activated the VAX FORTRAN HPO license (step 1) on all of the nodes.

   b. Then, if FORTRAN-HPO.EXE was established as a known image using the Install Utility, enter the following command:

      ```
      INSTALL> REPLACE SYS$SYSTEM:FORTRAN-HPO.EXE
      ```

      This command replaces the INSTALL entry for FORTRAN-HPO.EXE. It should be issued *only* on those nodes on which you activated the VAX FORTRAN HPO license in step 1. (In most cases, all of the nodes will be licensed.)
5. Repeat steps 3 and 4 for each node of the cluster.

It is possible to license VAX FORTRAN HPO on a subset of the nodes in a cluster. In that case, the nodes without the VAX FORTRAN HPO license will be unable to use the high-performance features of the compiler, specifically the /VECTOR and /PARALLEL=AUTOMATIC qualifiers.

3.3 User Account Requirements

Some products cannot be used unless certain privilege and quota requirements are met by user accounts. The following sections describe the privilege and quota requirements needed to work with VAX FORTRAN HPO.

3.3.1 User Account Privileges

No special privileges are needed to use VAX FORTRAN HPO. There is no benefit from installing the compiler with privileges.

3.3.2 User Account Quotas

To work with VAX FORTRAN HPO, a user account does not need to meet any special quota requirements. However, for parallel-processing applications, it may be advantageous to adjust account quotas in order to improve performance. See Section 4.6.1 for information about tuning issues related to parallel processing.

3.4 Enhancing VAX FORTRAN HPO Performance

This section includes information on system tuning and explains how to install VAX FORTRAN HPO as a shared image.

3.4.1 Tuning Your System

After you install VAX FORTRAN HPO, you might want to adjust your system to enhance performance or lower the use of some system resources. Information about tuning your system is included in the *VAX FORTRAN Performance Guide*. Tuning issues related to parallel processing are addressed in Section 4.6.1.

3.4.2 Installing VAX FORTRAN HPO as a Shared Image

If you expect VAX FORTRAN HPO to be used extensively on your system, you can reduce the system overhead and memory requirements by installing it as a shared image. To install VAX FORTRAN HPO as a shared image on a system that is currently running, use the VMS Install Utility (INSTALL). It is recommended that you install VAX FORTRAN HPO as shared on a system that has been rebooted recently because the available space in the global page table is less likely to be fragmented.

To install VAX FORTRAN HPO as a shared image, you must first determine the number of available global pages and global sections on your system. See Section 1.4.2.1 for information on how to verify and modify the number of global pages and global sections. Section 1.4.2 contains the guidelines for setting these parameters.

Once you have reset the system parameter values, invoke the VMS Install Utility from a privileged account and then install VAX FORTRAN HPO as a shared image:

```
$  INSTALL
INSTALL>  ADD SYS$SYSTEM:FORTRAN-HPO.EXE/OPEN/SHARED/HEADER_RESIDENT
INSTALL>  EXIT
```

Now, add the following lines to the appropriate system startup command file so that VAX FORTRAN HPO is available as a shared image each time the system is started. (If your site has modularized its startup procedures, be sure you add the lines to the correct file.) The default startup command file for VMS Version 5.0, for example, is SYS$MANAGER:SYSTARTUP_V5.COM.

```
$ INSTALL
ADD SYS$SYSTEM:FORTRAN-HPO.EXE/OPEN/SHARED/HEADER_RESIDENT
EXIT
```

When you install VAX FORTRAN HPO as a shared image you need to increase the global pages and global sections parameters (see Section 1.4.2).

3.5 Determining and Reporting Problems

If you encounter a problem while using VAX FORTRAN HPO, report it to Digital. Depending on the nature of the problem and the type of support you have, you can take one of the following actions:

- Call Digital if your software contract or warranty agreement entitles you to telephone support.
- Submit a Software Performance Report (SPR).
- Fill out and submit a Reader's Comments form if the problem has to do with the VAX FORTRAN HPO documentation. Reader's Comments forms are at the back of each manual. Use the form from the manual in which you found the error. Include the section and page number.

Review the Software Product Description (SPD) and Warranty Addendum for an explanation of warranty. If you encounter a problem during the warranty period, report the problem as indicated above or follow alternate instructions provided by Digital for reporting SPD nonconformance problems.

3.6 VAX FORTRAN Version 5 Maintenance Updates

This section describes components of the VAX FORTRAN Version 5 product that are replaced by equivalent components of the VAX FORTRAN HPO product. These replacements allow use of both products simultaneously on the same system. This sharing of components has an important implication with respect to the installation of future upgrades to VAX FORTRAN Version 5.

When you install a maintenance update of VAX FORTRAN, you must be careful not to overwrite any of the VAX FORTRAN HPO components. If the installation procedure detects that you have selected to overwrite some common components (files), it notifies you that it will not overwrite these files and asks you to confirm this. If these files are overwritten, you will have to reinstall VAX FORTRAN HPO to reestablish the components that it requires.

The following VAX FORTRAN HPO components are used by both products:

- The command line definition file for the FORTRAN command is replaced by the VAX FORTRAN HPO installation process. It is a superset of and replaces the FORTRAN command line definition file used by VAX FORTRAN Version 5. Use of the Version 5 compiler is supported on systems where the VAX FORTRAN HPO kit is installed by compiling with the /NOHPO qualifier.
- The help file for the FORTRAN command is replaced by the VAX FORTRAN HPO installation process. It is a superset of and replaces the help file used by VAX FORTRAN Version 5. Some of the information in the VAX FORTRAN HPO is not pertinent to VAX FORTRAN. For example, use of the new directives, the new qualifiers, and the statement form of ASSERT is not accepted by VAX FORTRAN Version 5.
- The error message definition files for the FORTRAN command are replaced by the VAX FORTRAN HPO installation process. They are a superset of and replace the error message definition files used by VAX FORTRAN Version 5.
- The DECwindows compiler support files for the FORTRAN command are replaced by the VAX FORTRAN HPO installation process. They are a superset of and replace the FORTRAN command processing support used by VAX FORTRAN Version 5.

4

Release Notes

This chapter contains information about the features of VAX FORTRAN HPO Version 1.0, differences between VAX FORTRAN HPO and previous VAX FORTRAN versions, corrections included in VAX FORTRAN HPO Version 1.0, and other topics. These topics are of interest to both system management and application programmers.

4.1 Overview of VAX FORTRAN HPO

This section briefly describes the new and changed features of VAX FORTRAN HPO, as compared with VAX FORTRAN Version 5.

The VAX FORTRAN High-Performance Option (VAX FORTRAN HPO) compiler is an updated version of the VAX FORTRAN Version 5 compiler that offers the following major enhancements:

- Autodecomposition of DO loops for parallel execution on VAX multiprocessor systems
- Vectorization of DO loops for execution on VAX processors that support the VAX Vector Architecture, such as the VAX 9000 series and the VAX 6000-400 series

The VAX FORTRAN HPO compiler supports the entire VAX FORTRAN language. Several new qualifiers, the ASSERT statement (which can also be used as a directive), and the INIT_DEP_FWD directive are also supported.

Although the documentation for this product remains largely unchanged, three major and several minor new functional capabilities are supported. The major new capabilities are as follows:

- Automatic analysis of array references inside DO loops (called *Dependence Analysis*)

- Generation of code for running eligible loops in parallel for high-performance execution on VAX multiprocessor systems (called *Autodecomposition*)
- Generation of code for running eligible loops using vector instructions for high-performance execution on VAX vector processors (called *Vectorization*).

In addition, VAX FORTRAN HPO supports new capabilities by the VAX Source Code Analyzer (SCA) and VAX Language-Sensitive Editor (LSE) products:

- VAX FORTRAN HPO supports new capabilities in VAX SCA Version 2, with new and improved generation of Analysis Data records in .ANA files and support for the VAXset Program Design Facility (PDF).
- VAX FORTRAN HPO generates new diagnostic records for use with VAX LSE Version 3 to allow enhanced diagnosis of source program constructs that inhibit autodecomposition and vectorization.

PDF features are described in detail in Section 4.2. The *VAX FORTRAN Performance Guide* has additional information on the use of LSE capabilities.

4.1.1 New Qualifiers and Qualifier Parameters

Several new qualifiers and qualifier parameters have been implemented to invoke and control the new features mentioned in the previous section.

FORTRAN users logged in to the system when the installation of VAX FORTRAN HPO is performed must log out and then log back in again to obtain the new command line definition.

The new qualifiers and qualifier parameters are:

/ASSUME=([NO]ACCURACY_SENSITIVE,[NO]DUMMY_ALIASES)
Default: /ASSUME=(ACCURACY_SENSITIVE,NODUMMY_ALIASES)

/CHECK=[NO]ASSERTIONS
Default: /CHECK=NOASSERTIONS

/BLAS=([NO]INLINE,[NO]MAPPED)
Default: /BLAS=(INLINE,NOMAPPED)

/DESIGN=([NO]COMMENTS,[NO]PLACEHOLDERS)
Default: /NODESIGN

/DIRECTIVES=[NO]DEPENDENCE
Default: /DIRECTIVES=DEPENDENCE

/[NO]HPO
Default: /HPO

/PARALLEL=([NO]AUTOMATIC, [NO]MANUAL)
Default: /NOPARALLEL

/SHOW=([NO]DATA_DEPENDENCES, [NO]LOOPS)
Default: /SHOW=(NODATA_DEPENDENCES,NOLOOPS)

/SYNCHRONOUS_EXCEPTIONS
Default: /NOSYNCHRONOUS_EXCEPTIONS

/TERMINAL=STATISTICS
Default: /NOTERMINAL

/VECTOR
Default: /NOVECTOR

/WARNINGS=([NO]ALIGNMENT,[NO]TRUNCATED_SOURCE)
Default: /WARNINGS=(ALIGNMENT,NOTRUNCATED_SOURCE)

The /HPO qualifier is particularly important because it controls whether the FORTRAN command is processed by the VAX FORTRAN Version 5 compiler image or the VAX FORTRAN HPO Version 1.n compiler image.

You should also take special note of the new /ASSUME qualifier, which is supported by the VAX FORTRAN HPO compiler. It can influence optimization behavior and affect whether programs compiled with VAX FORTRAN and VAX FORTRAN HPO get identical results.

Detailed information on these new qualifiers and qualifier parameters is provided in the sections that follow.

4.1.1.1 /ASSUME Qualifier

The /ASSUME qualifier specifies what the compiler can assume about program behavior without affecting correct results when it optimizes code.

The qualifier has the following form:

/ASSUME = ([NO]ACCURACY_SENSITIVE,[NO]DUMMY_ALIASES)

[NO]ACCURACY_SENSITIVE

Specifies whether the results of floating-point operations are strongly affected by certain code transformations and controls whether these transformations are allowed. These changes may or may not be detrimental to the overall accuracy of program results.

If NOACCURACY_SENSITIVE is specified, the compiler is free to reorder floating-point operations, based on algebraic identities (inverses, associativity, and distribution). This allows the compiler to generate a vectorized form of sum reductions (such as dot products), to make calls

to vectorized first order linear recurrence routines, and to move divide operations outside loops.

If ACCURACY_SENSITIVE is specified, the compiler uses only naive scalar rules for calculations. This setting can prevent use of the optimizations that are allowed when you specify NOACCURACY_SENSITIVE.

[NO]DUMMY_ALIASES

Specifies whether dummy (formal) arguments are permitted to share memory locations with common block variables or other dummy arguments.

This option is provided because many important opportunities for vector processing are lost when the compiler is forced to assume that this type of memory sharing does occur in any given subprogram.

If DUMMY_ALIASES is specified, program semantics require frequent recomputations on expressions involving dummy arguments. This is the default assumption for all previous releases of VAX FORTRAN. It assures identical results but inhibits some vectorization and some scalar optimizations, and thus slows performance.

If NODUMMY_ALIASES is specified, program semantics allow optimizations on expressions involving dummy arguments and common block variables. This is the default assumption for VAX FORTRAN HPO. It results in higher performance but may cause result differences if the programs depend on such aliases. It should not be used for source programs that violate FORTRAN–77 rules for overlapping memory references in dummy arguments (see the ANSI standard for FORTRAN–77 (X3.9-1978), Section 15.9.3.6).

If a program containing dummy aliasing is compiled with the NODUMMY_ALIASES in effect, the run-time behavior of the program will be unpredictable. The run-time behavior will depend on the exact optimizations that are performed. In some cases, normal results will occur; however, in other cases, results will differ because the values used in computations involving the offending shared locations will differ.

4.1.1.2 /BLAS Qualifier

The /BLAS qualifier controls how the compiler processes calls to the Basic Linear Algebra Subroutines (BLAS). The BLAS routines supported by the /BLAS qualifier are Level 1 routines (vector-vector routines). These routines were defined by Lawson, Hanson, Kincaid, and Krogh (1978). They are widely used for applications that repetitively apply simple transforms to two-dimensional arrays, such as matrix inversion or the solution of systems of linear equations.

The /BLAS qualifier has the following form:

```
/BLAS=([NO]INLINE,NOMAPPED)
```

INLINE

Specifies that calls to the routines supported by the /BLAS qualifier are to be treated as known language constructs and VAX object code is to be generated to compute the corresponding operations at the call site, instead of calling a user-supplied routine.

MAPPED

Specifies that calls to the routines supported by the /BLAS qualifier are to be treated as calls to the optimized implementations of these routines in the BLAS1$ and BLAS1$V portions of the Run-Time Library MTH$ facility.

The /BLAS qualifier is also accepted on the OPTIONS statement. The default is /BLAS=INLINE.

Command line qualifiers have the following effects on the processing of BLAS routine calls:

- If /NOBLAS is specified, *all* BLAS calls are treated as ordinary external routines.
- If /VECTOR or /PARALLEL=AUTO is in effect and /BLAS=INLINE is specified, the generated code for the loops may use vector instructions or be decomposed to run on multiple processors.

The following BLAS routines are supported by the /BLAS qualifier:

- IxAMAX, find the index of the maximum of the absolute values of the elements of a vector.
- xASUM, sum the absolute values of the elements of a vector.
- xAXPY, scale the elements of a vector and add in another vector.
- xCOPY, copy the elements of a vector to a new location.
- xDOT, take the dot product of a vector.
- xNRM2, take the Euclidean norm of a vector.
- xROT, apply a Givens plane rotation.
- xROTG, generate the elements for a Givens plane rotation.
- xSCAL, scale the elements of a vector.
- xSWAP, swap the elements of two vectors.

The data types supported by the /BLAS qualifier are REAL (S prefix), DOUBLE PRECISION (D prefix), COMPLEX (C prefix), and DOUBLE COMPLEX (Z prefix).

4.1.1.3 /CHECK Qualifier (ASSERTIONS Parameter)

The ASSERTIONS parameter enables or disables assertion checking. Assertions provide the compiler with information that can resolve unknown data dependences. See the *VAX FORTRAN Performance Guide* for detailed information about the use of assertions.

If /CHECK=ASSERTIONS, the compiler will generate the appropriate run-time code to evaluate the asserted expression. If the assertion tests false at run time, the program is stopped and an error message is issued.

If /CHECK=NOASSERTIONS, the compiler is free to use the assertion information in its optimizations and transformations, but no code is generated to test the validity of the assertions at run time.

Regardless of the qualifier setting, if the compiler can determine at compile time that the assertion is false, it will issue a diagnostic.

Assertions can be coded as statements or general compiler directives. (See the *VAX FORTRAN Language Reference Manual* for syntax information.)

4.1.1.4 /DESIGN Qualifier

See Section 4.2.1 for details about the /DESIGN qualifier.

4.1.1.5 /DIRECTIVES Qualifier

The /DIRECTIVES qualifier allows the programmer to enable and disable the recognition of general compiler directives (comments that begin with the prefix CDEC$). Recognition of parallel processing directives (comments that begin with the prefix CPAR$) is controlled by the /PARALLEL=MANUAL qualifier.

The qualifier has the following form:

```
/DIRECTIVES=([NO]DEPENDENCE)
```

Dependence directives are the only class currently supported by the /DIRECTIVE qualifier. The default of /DIRECTIVES=DEPENDENCE allows dependence directives (CDEC$ INIT_DEP_FWD) to be selected. The negative setting disables recognition. See the *VAX FORTRAN Performance Guide* for a description of the use of this directive for debugging vectorized programs.

4.1.1.6 /HPO Qualifier

The /HPO qualifier allows the VAX FORTRAN HPO Version 1.n compiler and the VAX FORTRAN Version 5 compiler to coexist on the same system, while still allowing both compilers to be invoked with the FORTRAN command. It specifies which compiler is to be used to perform a compilation.

The qualifier has the following form:

```
/[NO]HPO
```

/HPO invokes the VAX FORTRAN HPO Version 1.n compiler and /NOHPO invokes the VAX FORTRAN Version 5 compiler.

If /NOHPO is specified and the command line contains a qualifier that is specific to the VAX FORTRAN HPO compiler (for example, /ASSUME), a DCL error message is issued. (Note: The VAX FORTRAN Version 5.4 compiler supports the /NOHPO form as well. This allows development of command procedures that operate identically regardless of whether VAX FORTRAN HPO is installed.)

/HPO is the default when VAX FORTRAN HPO Version 1.n is installed on a system.

In a cluster environment, it is safe to install the Command Language Definition file (CLD file) for VAX FORTRAN HPO on all cluster nodes, even if the VAX FORTRAN HPO compiler is not installed or licensed on some system. This will insure that the /[NO]HPO qualifier is recognized on all systems.

Using DCL symbols to specify command qualifiers can cause problems. For example, if you establish the following DCL symbol, the introduction of the /HPO qualifier will cause DCL to issue a warning to the effect that all qualifiers preceding /HPO will be ignored.

```
$ F :== "FORTRAN/STANDARD=ALL"
. . .
$ F MYPROG.FOR
```

DCL issues the warning because of the way it processes alternate command syntax qualifiers such as /HPO (the /DML and /RSX qualifiers behave the same way). To avoid this error condition, specify /HPO or /NOHPO explicitly in the symbol definition, for example:

```
$ F :== "FORTRAN/HPO/STANDARD=ALL"
```

The CLD file controlling this behavior is provided with the compiler kit. The installation procedure places it in the file SYS$LIBRARY:FORTCLD.CLD. If sites have any operational difficulties due to these choices or need to modify FORTCLD.CLD to solve an unexpected problem, please submit a problem report detailing the circumstances and the change.

4.1.1.7 /PARALLEL Qualifier (AUTOMATIC and MANUAL Parameters)

The format of the /PARALLEL qualifier is as follows:

/PARALLEL=([NO]AUTOMATIC,[NO]MANUAL)

AUTOMATIC

Specifies that DO loops are to be automatically decomposed, enabling them to run in parallel mode on multiprocessor systems.

MANUAL

Specifies that CPAR$ directives are to be parsed and parallel code is to be generated for loops preceded by CPAR$ DO_PARALLEL.

The default is /NOPARALLEL.

Specifying /NOPARALLEL or /PARALLEL=(NOAUTOMATIC,NOMANUAL) has the same effect as omitting the /PARALLEL qualifier. Compiler directives relating to parallel processing (DO_PARALLEL, PRIVATE, CONTEXT_ SHARED, SHARED, LOCKON, LOCKOFF) are treated as comments and no object code is generated to support execution of parallel DO loops.

Specifying /PARALLEL without any options has the same effect as specifying /PARALLEL=(AUTOMATIC,MANUAL). This causes the compiler to do the following:

- Generate object code that supports execution of parallel DO loops
- Interpret parallel-processing compiler directive statements contained in the source code

See the *VAX FORTRAN User Manual* and the *VAX FORTRAN Performance Guide* for details on the use of the /PARALLEL qualifier.

4.1.1.8 /SHOW Qualifier (DATA_DEPENDENCES and LOOPS Parameters)

The effects of the DATA_DEPENDENCES and LOOPS parameters on the /SHOW qualifier are as follows:

DATA_DEPENDENCES

Specifies that diagnostic information about the following types of loops is to be included in the listing file:

- Loops that are ineligible for dependence analysis
- Loops that contain inhibitors to vectorization or decomposition

It also specifies, when the /DIAGNOSTICS qualifier is used, that detailed information about data dependences and other inhibitors to vector or parallel processing are to be included in the LSE diagnostic file. The LSE diagnostic file is the only acceptable vehicle for capturing these detailed diagnostics; the listing file and terminal are not sufficiently flexible to allow these diagnostics to be displayed in a usable form.

LOOPS

Specifies that loop structure and statement execution order chosen by dependence analysis, autodecomposition, and vectorization are to be included in the listing file.

4.1.1.9 /SYNCHRONOUS_EXCEPTIONS Qualifier

The /SYNCHRONOUS_EXCEPTIONS qualifier specifies whether vector arithmetic exceptions (such as floating-point overflow on a vector add instruction) are reported immediately after the instruction that caused the exception.

The qualifier has the following form:

/[NO]SYNCHRONOUS_EXCEPTIONS

The default is /NOSYNCHRONOUS_EXCEPTIONS.

The effect of this qualifier on exception reporting varies depending on whether the vector emulator or vector hardware is being used:

- If the vector emulator is being used, exceptions are reported immediately after the instruction that caused them regardless of which qualifier was specified (/NOSYNCHRONOUS_EXCEPTIONS or /SYNCHRONOUS_EXCEPTIONS).
- If vector hardware is being used, the exception is reported immediately only if the /SYNCHRONOUS_EXCEPTIONS qualifier was specified. If /NOSYNCHRONOUS_EXCEPTIONS was specified, exceptions are not reported at predictable program locations and the actual locations reported may vary from one vector processor to another. In the latter case, exceptions are reported on some vector instruction encountered after the instruction that caused the error.

If the default is in effect, exceptions are reported at any point after completion of the operation but before completion of the procedure containing the error condition.

The default setting allows uninterrupted completion of vector operations. This results in higher performance, but it also makes it more difficult to isolate the location of any exceptions.

The default behavior does not apply to special entry points used for the Run-Time Library (RTL) routines that implement the FORTRAN intrinsic functions.

4.1.1.10 /TERMINAL Qualifier

The /TERMINAL qualifier specifies whether certain compilation messages are displayed at the user's terminal during an interactive compile session.

The qualifier has the following form:

/[NO]TERMINAL = { [NO]STATISTICS | ALL | NONE }

[NO]STATISTICS

Specifies whether the compiler displays the program unit name and the number of code bytes generated when it completes processing that program unit.

ALL

Same interpretation as /TERMINAL=STATISTICS.

NONE

Same interpretation as /TERMINAL=NOSTATISTICS.

The default is /NOTERMINAL.

4.1.1.11 /VECTOR Qualifier

The /VECTOR qualifier specifies whether the compiler should analyze source code for data dependences and include vector instructions in the generated code. This process is called *vectorization*.

The qualifier has the following form:

/[NO]VECTOR

The default is /NOVECTOR.

4.1.1.12 /WARNINGS Qualifier (ALIGNMENT and TRUNCATED_SOURCE Parameters)

The effects of the ALIGNMENT and TRUNCATED_SOURCE parameters on the /WARNINGS qualifier are as follows:

ALIGNMENT

Causes diagnostic messages to be issued when the compiler detects variables or arrays (created in COMMON or EQUIVALENCE statements) that cross natural boundaries for their data size. For example, a diagnostic message is issued if /WARNINGS=ALIGNMENT is in effect and the virtual address of a REAL*8 variable is not a multiple of 8. This can occur, for example, if the REAL*8 variable is in a common block and is preceded by an odd number of INTEGER variables.

If misaligned data inhibits vectorization, this inhibition will be reported in a dependence diagnostic when the /VECTOR, /DIAGNOSTICS and /SHOW=DATA_DEPENDENCES qualifiers are specified.

The default is ALIGNMENT.

TRUNCATED_SOURCE

Causes a warning diagnostic message (EXCCHASRC) to be issued when the compiler reads a source line with a statement field that exceeds the maximum column width. The maximum column width is column 72 or 132, depending on the value of the /EXTEND_SOURCE qualifier or the OPTIONS statement qualifier in effect.

This option has no effect on truncation; lines that exceed the maximum column width are always truncated.

The default is NOTRUNCATED_SOURCE.

4.1.2 Assertion Declarations

Assertions allow the programmer to provide the compiler with information about the values that can be taken on by specified variables. They are used to make testable declarations concerning run-time properties of the program unit. An assertion can be coded in the form of a statement or a directive:

```
ASSERT (e)          (Statement form)

CDEC$ ASSERT (e)          (Directive form)
```

The notation "e" represents a logical expression.

Assertions allow the compiler to perform vectorization or decomposition that would otherwise be inhibited because of possible incorrect program results. For example, when compiling a loop of the following form, the compiler will inhibit vectorizing the loop because it cannot determine whether statement 9 or statement 10 must appear first in the vectorized code (as explained in the section on dependence analysis in the *VAX FORTRAN Performance Guide*.

```
0008    DO I=2,N
0009      A(I) = B(I) + C(I)
0010      B(I+K) = X * C(I)
0011    ENDDO
```

However, if the programmer knows that the value of K is always greater than 0, the programmer would use an assertion before the beginning of the loop. For example:

```
    ASSERT (K .GT. 0)
```

The compiler would apply the assertion to the unknown value in the array reference B(I+K), and this would allow the compiler to vectorize the statement. If the /CHECK=ASSERTIONS qualifier is specified, the compiler would also issue an error message at run time if the value of K does not meet the condition. (See Section 4.1.1.3 for more details on how the /CHECK=ASSERTIONS qualifier interacts with assertion declarations.)

Assertions offer a number of advantages over other forms of directives or statements that have been used for this purpose in earlier FORTRAN compilers:

- They can be tested, thereby making the code more resistant to errors associated with unexpected changes.
- They can apply to more than one loop in a routine.
- They can be used to improve scalar optimizations and decomposition as well as vectorization performance.
- They provide more specific information that can be used in dependence analysis, optimization, and transformation decisions.

One important property of assertions is that they are context sensitive. In particular, they are not "global" assertions; they are assumed to be true only at the location where they appear. If a variable that appears in the asserted expression is modified, the asserted expression is presumed not to hold any longer. Thus, in order to be effective, assertions must be placed so as to remain valid at all array references to which they relate.

The results of the unknown dependence diagnosis can be used to determine which loops might benefit from assertions.

See the *VAX FORTRAN Performance Guide* for examples and information about the use of the ASSERT statement.

4.2 VAXset Program Design Facility

The VAXset Program Design Facility (PDF) consists of enhancements to language processors (such as VAX FORTRAN), the VAX Language-Sensitive Editor (LSE), and the VAX Source Code Analyzer (SCA) to provide the ability to embed application design information in the program source, to have this information analyzed by PDF components, and to allow for inquiries and reports on the design information.

VAX FORTRAN's support of PDF lies in two areas:

- The ability to analyze the text of source comments for design information and to add that information to the SCA analysis data file for later access through SCA.

- The ability to allow LSE placeholders, including a new pseudocode placeholder, to be used in place of certain FORTRAN language syntax. Information about the use of placeholders is added to the SCA analysis data file, if selected.

The following sections discuss only those portions of PDF support that are specific to the VAX FORTRAN compiler. For more information on PDF, see the VAX Language-Sensitive Editor and the VAX Source Code Analyzer documentation.

4.2.1 /DESIGN Compile Command Qualifier

The /DESIGN qualifier specifies that the compiler is to analyze the source code for program design information in comment text or LSE placeholders.

The qualifier has the following form:

/[NO]DESIGN = { [NO]COMMENTS / [NO]PLACEHOLDERS }

COMMENTS

Specifies that the text of each comment—either full-line or end-of-line comment—is to be analyzed for program design information. Comment text is scanned for tags that are defined in the current LSE environment file. Certain tags are predefined by LSE; however, you can add, change, or remove tags as desired. (See the VAX Language-Sensitive Editor documentation for details.)

If you also specify the /ANALYSIS_DATA qualifier, information about tags found, text of tagged and untagged comments, and their locations in the source file is included in the SCA analysis data file. You can then use SCA and LSE commands to retrieve the design information.

At compile time, the compiler reports any errors detected by comment analysis routines. If LSE and SCA are not installed or if a severe error is reported by the comment analysis routines, the compilation is aborted.

Groups of full-line comments are identified by the compiler as "comment blocks", including lines which consist of white space followed by a "!" comment introducer. Differences in indentation of full-line comments is ignored for the purposes of grouping comments into blocks. A comment block is ended when a completely blank line or a line which is not a full-line comment is detected.

End-of-line comments, where non-blank text is followed by a "!" comment introducer, are treated as self-contained comment blocks.

Comments from source that is included by means of the INCLUDE or DICTIONARY statements are also analyzed for design information. The end of an INCLUDE module or DICTIONARY definition, by itself, does not end a comment block.

Comments beginning with a "D" or "d" in column 1 are not examined for program design information.

PLACEHOLDERS

Specifies that LSE placeholders (in well-defined contexts) are to be treated as valid program syntax.

VAX FORTRAN recognizes five types of placeholders defined by LSE:

```
{required-placeholder}
{required-list-placeholder}...
[optional-placeholder]
[optional-list-placeholder]...
«pseudocode-placeholder»
```

An alternative syntax that uses the characters "<<" and ">>" for a pseudocode placeholder is also accepted by VAX FORTRAN:

```
<<pseudocode-placeholder>>
```

Note that there is no list form of pseudocode placeholder.

A placeholder must be terminated before the end of the line, and the closing delimiter must correspond to the opening delimiter. For example, the following placeholder is invalid:

```
[invalid-delimiter}
```

Any text may appear inside a delimiter except for the closing delimiter. There is no ability to "quote" a closing delimiter in order to allow its inclusion as text. Nesting of delimiters is not supported.

If you also specify the /ANALYSIS_DATA qualifier, information about placeholders and their use is included in the SCA analysis data file.

If placeholder analysis is performed on a compilation unit that contains placeholders in valid contexts, the compiler does not generate an object file. An object file is generated, however, if no placeholders are found in the compilation unit undergoing placeholder analysis.

If LSE placeholders are present in the source file and you do not specify the /DESIGN=PLACEHOLDER qualifier, the compiler will treat the placeholders as invalid syntax and will generate an error message.

4.2.1.1 Allowed Contexts for Placeholders

A placeholder can be used to replace either an entire FORTRAN statement or a part of a statement. As a replacement for an entire statement, there are no restrictions on where a placeholder may appear. Otherwise, as a replacement for a portion of a statement, a placeholder is generally treated as an undeclared identifier whose implicit type is INTEGER. This implicit typing is unaffected by the IMPLICIT or IMPLICIT NONE statements, but is affected by the /[NO]I4 command qualifier or the OPTIONS statement qualifier.

Each occurrence of a placeholder is considered as a unique instance and is unaffected by earlier uses of a syntactically identical placeholder. For example, in the following code sequence, each occurrence of "[abc]" is considered to be a never-before-seen identifier.

```
INTEGER [abc]
REAL [abc]
```

The implicit typing of placeholders as INTEGER allows them to be used in most reasonable contexts, but if they are used in contexts where an integer identifier would not be allowed, semantic error messages are generated. For example, the following statement is invalid due to the type mismatch:

```
CHAR_VAR = CHAR_VAR // [placeholder]
```

Like any other undeclared variable, placeholders are considered scalars and cannot be used in a context that requires the variable to be declared as an array.

Given these restrictions, placeholders can appear in the following contexts:

- As a variable name in a type specification, COMMON, DATA, DIMENSION, EQUIVALENCE, RECORD, SAVE, or VOLATILE statement, and CPAR$ CONTEXT_SHARED or CPAR$ PRIVATE directives
- As the symbolic name in an EXTERNAL or PARAMETER statement
- As an integer expression
- As the target of an assignment statement
- As the name of subroutine in a CALL statement
- As the name of a function in a function reference
- As an integer variable in an ASSIGN or assigned GOTO statement
- As the control variable in a DO statement or implied DO list in an I/O statement
- As the lock variable in a CPAR$ LOCKON or CPAR$ LOCKOFF directive (the placeholder is treated as LOGICAL*4 in these contexts only)

- As an integer variable in an I/O statement other than the unit or format specifier

Placeholders are not allowed in the following contexts:

- As the symbolic name in a BLOCK DATA, ENTRY, FUNCTION, INTRINSIC, PROCEDURE, or PROGRAM statement
- As formal (dummy) argument names in an ENTRY, FUNCTION, or PROCEDURE statement
- As the name of a common block
- As a structure name
- In a NAMELIST statement, either as the group name or as one of the variables to be included in the group
- As the unit or format specifier in an I/O statement

4.3 DECwindows Compile Verb Support in FileView

Support for the DECwindows Compiler Interface (DWCI), as accessed through FileView's Compile verb, was introduced in VAX FORTRAN Version 5.3. It has been updated to support all of the new qualifiers that apply specifically to VAX FORTRAN HPO (see Section 4.1.1).

The Compile verb is contained in FileView's Programming menu. The Compile verb allows compile command options to be selected using a window interface instead of the usual DCL qualifiers. The compiler defaults are used as the initial configuration of compile command options; new configurations with different option settings can be established for use by later compilations.

The DWCI interface is used automatically when you use the Compile verb to compile a source file that has a .FOR file type. More information is available by using the Help menu in the dialog box with the name: Compile a FORTRAN Program. (This dialog box appears when you select the Compile verb in the Programming menu).

The DWCI interface for VAX FORTRAN is entirely separate from the compiler image and is contained in the following files:

```
SYS$COMMON:[VUE$LIBRARY.USER]FORTRAN$DWCI.EXE
SYS$COMMON:[VUE$LIBRARY.USER]FORTRAN$DWCI.UID
SYS$COMMON:[SYSHLP]FORTRAN$DWCI.HLB
```

4.4 VAX FORTRAN Performance Guide

The *VAX FORTRAN Performance Guide* is a new manual in the FORTRAN documentation set. It describes the concepts discussed in these release notes in more detail and gives usage instructions as well as performance information on other aspects of VAX FORTRAN usage.

4.5 Compiler Deficiencies in VAX FORTRAN Version 5 and VAX FORTRAN HPO

The following sections document restrictions and known problems that are shared by VAX FORTRAN HPO Version 1.0 and VAX FORTRAN Version Version 5.4.

4.5.1 Restrictions on Standards Checking

Source statements that do not conform to FORTRAN language standards are detected by the VAX FORTRAN compiler under the following circumstances:

- The statements contain ordinary syntax and semantic errors.
- A source program containing nonconforming statements is compiled with the /STANDARD or /CHECK qualifiers.

Given these circumstances, the compiler is able to detect most instances of nonconforming usage. It does not, however, detect all instances because the /STANDARD qualifier does not produce checks for all nonconforming usage at compile time. In general, the unchecked cases of nonconforming usage arise from the following situations:

- The standard violation results from conditions that cannot be checked at compile time.
- The compile-time checking would be prone to false alarms.

Most of the unchecked cases occur in the interface between calling and called subprograms. However, other cases are not checked, even within a single subprogram.

The following list identifies those items that are known to be unchecked. Section references in parentheses refer to the FORTRAN–77 Standard, ANSI X3.9-1978.

1. Use of a data item before it is defined. (6.6)
2. Use of the SAVE statement to ensure that data items or common blocks retain their values when reinvoked. (8.9)
3. Association of a character data item that is on both the right and left sides of a character assignment statement. (10.4)

4 Mismatch in order, number, or type in passing actual arguments to subprograms. This includes improper array dimensioning. (15.9)

5 Association of one or more actual arguments with a data item in a common block when calling a subprogram that assigns a new value to one or more of the arguments. (15.9.3.6)

4.5.2 Using /DML and /PARALLEL Qualifiers

Because of the restriction on performing I/O operations within a parallel loop (see Section 15.2.1 in the *VAX FORTRAN User Manual*), care must be taken using parallel loops with database applications. In particular, applications using the VAX DBMS product, especially the /DML qualifier on the FORTRAN command line, should not use the DO_PARALLEL directive for any loop that executes database operations. Such use will cause unpredictable results due to inconsistent I/O control states among the processes executing the loop iterations.

4.5.3 Maintaining Password Security

The FORTRAN compiler echoes, exactly as entered, the command line in the compilation summary section of any listing file that it generates. If an account password is required on an input or output file, we suggest that a logical name be defined for the password and that the logical name be used on the command line. Otherwise, the password will be printed in the command line echo part of the compilation summary section.

4.5.4 Status of Bug Fixes

See the online release notes for descriptions of bug fixes and known bugs.

4.6 Additional Information Needed for FORTRAN Parallel Processing

This section gives additional information that will be helpful to those who are using parallel-processing features provided by the compiler. The system interface used with VAX FORTRAN HPO is the same as the one used with directed decomposition in VAX FORTRAN Version 5, so the same concepts apply to both.

4.6.1 Tuning Issues for Parallel Processing

Parallel-processing programs may fail to execute because of insufficient system resources. You may have to adjust some resource-utilization parameters, both for the entire system and for individual user accounts. You may also want to adjust some parameters in order to achieve better performance for programs executing in parallel. For more details on these parameters as they relate to the FORTRAN parallel-processing support, refer to the *VAX FORTRAN User Manual*, Chapter 15.

4.6.1.1 System Parameters

When a parallel application is executed, much of the local memory and many common blocks of the application are mapped to global sections. (VMS uses this method to share data between processes.) Users must ensure that the number of global sections, global pages, and global page file sections required by a parallel application are available. To allow enough space for this global data, some of the system's SYSGEN parameters may need to be increased. The three SYSGEN parameters that are most important are GBLPAGFIL, GBLPAGES, and GBLSECTIONS. Note that these parameters are not dynamic; you must reboot your system before any modifications to them can take effect. You should adjust the parameters one at a time in order to avoid modifying some of them unnecessarily.

See the discussion of these parameters in the *VAX FORTRAN User Manual*, Section 15.4.1. With respect to the total amount of global pages (GBLPAGES parameter) and global page file size (GBLPAGFIL parameter), it is important to note that common blocks are all shared by default when /PARALLEL=AUTOMATIC is in effect. This means you must sum the size of all common block program sections to compute the number of pages needed.

4.6.1.2 User Parameters

One of the authorization quotas, the PGFLQUO quota, may need to be adjusted for any account that will be running parallel applications. This quota is shared by all processes in a job and thus may require an adjustment to allow for the additional processes used in parallel processing.

You can adjust some programs to reduce their use of pagefile quota. For example, it is advisable to move a large array into a common block with an attribute of shared if the array has a memory allocation attribute of context-shared and it is declared within either the main program or a subprogram that is never invoked from within a parallel loop. This is advisable because context-shared variables cause the FORTRAN run-time parallel-processing support to allocate one shared copy and *n* private copies, one for each process. These copies use up pagefile quota and often require the quota to be raised above what would be needed for the corresponding serial program. If the private copies are never needed, you can reduce the pagefile quota requirements by changing the variable from context-shared to shared (which is allowed only for common blocks).

4.6.1.3 Working Set Parameters

Parallel-processing applications typically use large amounts of memory. To achieve better performance for an application, you may find it advisable to make adjustments to the working set size parameters, both for the system and for user accounts. Refer to the *Guide to VMS Performance Management* for information on how to adjust working set size.

4.6.2 Restrictions on Parallel-Processing Applications

The following restrictions apply to parallel-processing applications (in addition to restrictions documented in the *VAX FORTRAN User Manual*):

- Images using the FORTRAN parallel-processing support cannot run from a detached process where there is no DCL CLI.
- The name of any common blocks defined to be shared within a routine compiled with the /PARALLEL=MANUAL qualifier must not contain more than 26 characters.

4.6.3 Linking Against FORTRAN Parallel-Processing Support

The following section describes how to link agains the FORTRAN run-time parallel-processing support.

Section 4.7.3 describes problems that can occur when linking objects compiled with /PARALLEL.

4.6.3.1 Linking with the /NOSYSSHR Qualifier Please note that the FORTRAN run-time parallel-processing support references a global symbol CTL$GL_PSTFLAGS defined in the system symbol table (SYS$SYSTEM:SYS.STB).

If you prefer to include the FORTRAN parallel-processing run-time object modules (included in SYS$LIBRARY:STARLET.OLB) in your image, you need to include the system symbol table as input in your linking operation. This allows the linker to resolve this external symbol reference.

The following is an example of a linking operation:

```
$ LINK/NOSYSSHR MY_OBJECT.OBJ,SYS$SYSTEM:SYS.STB
```

4.6.4 Using PPL$ Run-Time Library Routines

There is a known problem with the interaction between routines compiled with /PARALLEL and the run-time library PPL$ routines. The result of this problem is that the FORTRAN parallel-processing application completes, but the subprocesses that are created during its execution are not properly deleted and are left in a spin loop, using up CPU time. These subprocesses must be stopped by the user. The easiest way to stop them is to either log out of the account that executed the parallel application or explicitly stop them using the DCL command STOP.

4.7 Compatibility Issues Involving VAX FORTRAN HPO

The following sections describe the compatibility among source programs that are compiled, linked, and run using VAX FORTRAN HPO and previous versions of VAX FORTRAN.

4.7.1 Compilation Compatibility

Some of the default qualifier settings for VAX FORTRAN HPO differ from those for the VAX FORTRAN Version 5 compiler. The differences can be summarized as follows:

- VAX FORTRAN Version 5 does not support the /ASSUME qualifier. Its behavior corresponds to /ASSUME=(ACCURACY_SENSITIVE, DUMMY_ALIASES). The VAX FORTRAN HPO default for this qualifier is /ASSUME=(ACCURACY_SENSITIVE, NODUMMY_ALIASES).
- VAX FORTRAN Version 5 does not support values on the /PARALLEL qualifier. The effective default when /PARALLEL is specified in VAX FORTRAN is /PARALLEL=(NOAUTOMATIC,MANUAL). The VAX FORTRAN HPO default when /PARALLEL is specified is /PARALLEL=(AUTOMATIC,MANUAL).
- VAX FORTRAN Version 5 does not support the ALIGNMENT value on the /WARNINGS qualifier, but the behavior corresponds to the /WARNINGS=NOALIGNMENT setting. The VAX FORTRAN HPO default in this case is /WARNINGS=ALIGNMENT.
- VAX FORTRAN Version 5 does not support the /BLAS qualifier. Its behavior corresponds to /NOBLAS. The VAX FORTRAN HPO default is /BLAS=INLINE.

For a description of the behavior of the qualifiers, see Section 4.1.1 and the *VAX FORTRAN Performance Guide*.

Note that the VAX FORTRAN Version 5 defaults apply when the FORTRAN /NOHPO command is specified. If problems arise with programs compiling or executing in ways that differ from VAX FORTRAN Version 5, it is possible that the new defaults are inappropriate for the program.

4.7.2 Source Compatibility

VAX FORTRAN HPO supports all source code supported by VAX FORTRAN Version 5. New diagnostics for data dependences and misaligned data are optional.

4.7.3 Object Compatibility

Object files created by the VAX FORTRAN HPO Version 1.n compiler are fully compatible with object files created by the VAX FORTRAN Version 5 compiler. They can be freely intermixed in the same executable image.

The following rule must be followed when linking objects compiled with /PARALLEL:

Note *If any parallel loop appears in the program, then all routines on the calling path from the main program to the routine containing the parallel loop must be*

compiled with either the AUTOMATIC or MANUAL parameters (or both) on the /PARALLEL qualifier. If this is not done, linker errors will occur.

If a routine uses a parallel DO loop compiled with /PARALLEL in VAX FORTRAN Version 5 and you do not want VAX FORTRAN HPO to create any additional parallel loops, then compile with /PARALLEL=MANUAL explicitly.

Programs compiled with the VAX FORTRAN HPO compiler refer to VMS system support routines that are not supported on VMS systems prior to Version 5.4. Therefore, the linking and execution of output modules associated with VAX FORTRAN HPO compilations requires VMS Version 5.4 or higher.

4.7.4 Image Compatibility

The images linked with VAX FORTRAN HPO object modules are executable only on VMS Version 5.4 or higher.

4.8 Source Program Diagnostic Messages

New source program diagnostic messages included with VAX FORTRAN HPO Version 1.0 are described in the sections that follow.

4.8.1 Messages Supporting Dependence Analysis of DO Loops

The following errors inhibit dependence analysis from being performed on a DO loop. These messages are generated if either the /VECTOR or /PARALLEL=AUTOMATIC and the /SHOW=DATA_DEPENDENCES command line qualifiers are specified. They are generated to the listing file and, if one is requested, to the diagnostic file.

Mnemonic	Error Code	Text/Explanation
NODEPNEST	I	Dependence analysis inhibited due to inhibitor in loop nested inside DO loop A loop nested within this DO loop has been inhibited from having dependence analysis performed on it, thus eliminating this DO loop from being analyzed.
NODEPDEEP	I	Dependence analysis inhibited due to more than 7 loops nested inside DO loop Dependence analysis is only performed on the deepest seven levels.

Mnemonic	Error Code	Text/Explanation
NODEPFLOW	I	Dependence analysis inhibited due to transfer into or out of DO loop Due to the potential that a DO loop is entered or exited without executing all iterations, neither vectorization nor autodecomposition can be performed on the DO loop. A STOP or RETURN statement is considered as a transfer out of a loop. The compiler will not perform the dependence analysis on the loop.
NODEPINT4	I	Dependence analysis inhibited due to non-integer control variable Dependence analysis is only performed for DO loops whose control variable is a 4-byte INTEGER.
NODEPVOLAT	I	Dependence analysis inhibited due to volatile control variable Dependence analysis is only performed for DO loops whose control variable is not volatile.
NODEPCOMMN	I	Dependence analysis inhibited due to call in DO loop and COMMON loop control variable As a side-effect of the call, the DO loop control variable has the potential of being modified, thus changing the number of iterations performed for the DO loop.
NODEPEQUIV	I	Dependence analysis inhibited due to equivalenced loop control variable Dependence analysis is only performed for DO loops whose control variable is a 4-byte INTEGER.
NODEPPAR	I	Dependence analysis inhibited due to parallel loop nested inside DO loop A nested, manually decomposed loop prevents dependence analysis on outer DO loops.
NODEPOVLP	I	Dependence analysis inhibited due to potential overlap dummy arguments and common variables in DO loop Due to the potential overlap of a dummy argument and a common variable, dependence analysis will not be performed. This error is controlled by the command line setting for the /ASSUME=[NO]DUMMY_ALIASES qualifier.

Mnemonic	Error Code	Text/Explanation
NODEPSTMT	I	Dependence analysis inhibited due to use of inhibitor statement inside DO-loop Certain FORTRAN statements inhibit the analysis of a DO-loop. Some of them include: Assigned GOTOs Computed GOTOs Invocation of character function LOCKON or LOCKOFF directive specified Reference to format containing variable format expression Direct access I/O operation

4.8.2 Messages Supporting Vectorization of Specific Statements

The following errors inhibit vectorization from being applied on specific statements. These messages are generated if the /VECTOR/SHOW=DATA_DEPENDENCE/DIAGNOSTICS qualifiers are specified. These errors are only output to the diagnostic file, not to the listing file or terminal.

Mnemonic	Error Code	Text/Explanation
NOVECVOLARR	I	Vectorization inhibition: Use of volatile array Volatile arrays have dependences which the compiler is unaware of, thus they can not be vectorized.
NOVECDTARR	I	Vectorization inhibition: Data type not supported for array There are no vector instructions to work on the data type specified for the array.
NOVECVOLIND	I	Vectorization inhibition: Use of volatile array index The compiler can not identify all of the dependences for volatile array indexes, thus they can not be vectorized.
NOVECDTIND	I	Vectorization inhibition: Use of unsupported data type in array index Data type for index prevents vectorization.
NOVECALIGN	I	Vectorization inhibition: Misaligned array Data used for vector instructions must be aligned on natural boundaries, based on its data type.

Mnemonic	Error Code	Text/Explanation
NOVECSUBP	I	Vectorization inhibition: Reference to subprogram Vectorization is inhibited for any data which potentially can be modified in the subprogram.
NOVECACC	I	Vectorization inhibition: Qualifier /ASSUME=ACCURACY prevents recurrence vectorization A recurrence has been identified, but the transformation required to vectorize the statement can not be performed because the /ASSUME=ACCURACY qualifier is in effect.

4.8.3 Messages Supporting Vectorization

The following errors identify when vectorization was inhibited and attempt to identify why it was inhibited. These messages are generated if the /VECTOR /SHOW=DATA_DEPENDENCE/DIAGNOSTICS qualifiers are specified. These errors are only output to the diagnostic file, not to the listing file or terminal.

Message Texts and Explanations
Unknown dependence using *xxxx* The compiler has analyzed the dependences between array references and is unable to determine the order in which the references must occur. This is often due to not knowing the possible range of values for an expression used as part of the array index.
Nonuniform dependence The compiler has detected that the relative order of references to common memory locations, with respect to two array references in the loop, are not uniform for all such locations. For example: `A(I) = B(I) * A(N-I).` Some locations are accessed first by A(I) and others by A(N-I).
Outer level vectorization inhibited The compiler has analyzed the dependences between array references at different DO-loop levels and is unable to determine the order in which the references must occur.

These errors relate to assertion declarations. See Section 7.2.1 in the *VAX FORTRAN Performance Guide* for information on how to handle these errors.

Within LSE, you can use Review Mode to apply suggested assertions to handle these errors. To do so, move the cursor to the suggested assertion that you wish to use and press CTRL/G. This instructs LSE to apply the suggested assertion to the source file. LSE then prompts for a confirmation that this change is desirable. If so, enter YES and the suggested assertion will be inserted.

4.8.4 Messages Supporting the VAXset Program Design Facility

The following source program diagnostic messages have been added in support of the VAXset Program Design Facility.

Mnemonic	Error Code	Text/Explanation
DESCOMERR	W	Error in processing design information The comment analysis routines have detected an error in the text of a comment, such as an undefined keyword in a structured tag. Additional messages relating to the error are also displayed.
DESCOMSEVERR	F	A serious error has occurred processing /DESIGN=COMMENTS The comment analysis routines have detected a serious error that prevents further comment analysis in the current compilation. Additional messages relating to the error are also displayed. The compilation is aborted.
DESIGNTOOOLD	F	/DESIGN=COMMENT processing routines are too old for the compiler The version of the comment analysis routines present on the system is not supported by this version of VAX FORTRAN. Install a newer version of the VAX Language-Sensitive Editor to obtain the most recent version of the comment analysis routines.
PLACEEOL	F	Placeholder not terminated before end of line The closing delimiter of a placeholder was not found before the end of the line.
PLACENODESIGN	E	Placeholder not valid without /DESIGN=PLACEHOLDERS A placeholder was found but /DESIGN=PLACEHOLDERS was not specified on the compile command line.
PLACENODOT	E	Repetition of pseudocode placeholder not allowed A pseudocode placeholder followed by an ellipsis (...) was found, indicating a list placeholder, which is not supported.

Mnemonic	Error Code	Text/Explanation
PLACENOOBJ	I	Placeholders detected—no object code generated One or more placeholders were found in the current compilation unit and therefore no object code was generated for that unit.
PLACENOTVAL	E	Placeholder not valid in this context A placeholder was found in a context that is not supported.

A

Sample Installation

This appendix contains a sample installation of each of the components on the VAX FORTRAN HPO kit. Note that this is only a sample. Sometimes the installation procedure is modified in maintenance updates of this product.

Because this installation guide is not usually revised for maintenance updates, be sure to read the *Read Before Installing or Using VAX FORTRAN HPO* letter before you install this product. This letter notes any significant changes to the installation sample shown here.

This sample was run on a system that had no previous version of VAX FORTRAN HPO installed. Depending on which layered products you have on your system, you might see additional messages and questions when you perform your installation.

The sample installation assumes that DECnet has been shut down, that no users are logged on to your system, and that OPTIONS N is specified to print the release notes. The Installation Verification Procedure (IVP) runs at the end of the installation.

Sample VAX FORTRAN HPO Installation

```
$ @sys$update:vmsinstal forthpo010 mua0:

        VAX/VMS Software Product Installation Procedure V5.4


It is 4-APR-1990 at 18:05.

Enter a question mark (?) at any time for help.

* Are you satisfied with the backup of your system disk [YES]?

Please mount the first volume of the set on MUA0:
* Are you ready? YES
%MOUNT-I-MOUNTED, FORHPO mounted on _MUA0:

The following products will be processed:
  FORTHPO V1.n
```

```
        Beginning installation of FORTHPO V1.n at 18:05

%VMSINSTAL-I-RESTORE, Restoring product save set A ...
%VMSINSTAL-I-RELMOVED, Product's release notes have been moved to SYS$HELP.

                        VAX FORTRAN HPO V1.n-1

  Copyright © Digital Equipment Corporation, 1990.  All rights reserved.


* Do you want to purge files replaced by this installation [YES]?

    The VAX FORTRAN HPO kit is divided into three components:

        Compiler (includes help text, command definition, IVP)
        FORSYSDEF.TLB (VMS system symbol definition library)
        DECwindows Compiler Interface (FileView Compile verb support)

    Please select the components you wish to install by answering YES or NO
    to the following questions.  If you want a more detailed description of
    any component, answer with a question mark "?"; after the explanation is
    displayed, you will be prompted again for a response.

* Do you want to install the compiler [YES]?

    FORSYSDEF.TLB, which provides VMS interface definitions for
    FORTRAN applications, is not present and therefore should be
    created.  For more information on FORSYSDEF.TLB, enter a
    question mark "?" at the following prompt.

* Do you want to create a new FORSYSDEF.TLB [YES]?
* Do you want to install the DECwindows Compiler Interface [YES]?

    You have selected the following components:

      Compiler
      FORSYSDEF.TLB
      DECwindows Compiler Interface

* Is this correct [YES]?

    VAX FORTRAN HPO requires Product Authorization Keys (PAKs) for
    VAX FORTRAN, VAX FORTRAN HPO, and VAX Language-Sensitive Editor to
    be registered and loaded.   Answer the following three questions with
    YES if a PAK with the specified product name and a release date equal
    to or later than the date shown is registered and loaded, otherwise
    answer NO.

        Product:      FORTRAN
        Producer:     DEC
        Version:      5.n
        Release Date: dd-mmm-yyyy

* Does this product have an authorization key registered and loaded? yes

        Product:      FORTRAN-HPO
        Producer:     DEC
        Version:      1.n
        Release Date: dd-mmm-yyyy
```

```
* Does this product have an authorization key registered and loaded? yes

        Product:      LSE
        Producer:     DEC
        Version:      3.0
        Release Date: 1-NOV-1989

* Does this product have an authorization key registered and loaded? yes

    This kit contains an Installation Verification Procedure
    (IVP) to verify the correct installation of the VAX FORTRAN HPO
    compiler.  The IVP will be placed in:

                SYS$TEST:FORTRAN-HPO$IVP.COM

    After the installation is complete, you can invoke the
    command file at any time to reverify that VAX FORTRAN HPO is
    installed and working correctly.

* Do you want to run the IVP after the installation [YES]? yes

        All questions regarding the installation of VAX FORTRAN HPO
        have now been asked.  Depending upon your configuration,
        time estimates for the installation(s) have been provided.

        Compiler:               1  to  10 minutes
        FORSYSDEF.TLB:          5  to  30 minutes
        DWCI support            1  to  2  minutes

        +------------------------------------------------------+
        |     Installing the VAX FORTRAN HPO Compiler          |
        +------------------------------------------------------+

        +------------------------------------------------------+
        |             Installing FORSYSDEF.TLB                 |
        +------------------------------------------------------+

        +------------------------------------------------------+
        |       Installing DECwindows Compiler Interface       |
        +------------------------------------------------------+

    Your VMS system will now be updated to include the
    following new and modified file(s):

    SYS$HELP:FORTRAN-HPO010.RELEASE_NOTES       [new]
    SYS$SYSTEM:FORTRAN-HPO.EXE                  [new]
    SYS$MESSAGE:FORTERR1.EXE                    [new]
    SYS$MESSAGE:FORTERR2.EXE                    [new]
    SYS$TEST:FORTRAN-HPO$IVP.COM                [new]
    SYS$HELP:HELPLIB.HLB                        [modified]
    SYS$LIBRARY:DCLTABLES.EXE                   [modified]
    SYS$LIBRARY:FORSYSDEF.TLB                   [new]
    VUE$LIBRARY:FORTRAN$DWCI.EXE                [new]
    VUE$LIBRARY:FORTRAN$DWCI.UID                [new]
    SYS$HELP:FORTRAN$DWCI.HLB                   [new]
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target
                        directories...
```

```
Beginning VAX FORTRAN HPO Installation Verification Procedure

Compiler image identification is FORT-HPO V1.n-1

Test compiler operation and error messages...

Compile, link and run program with parallel processing features...

Test FORSYSDEF.TLB system definitions...

Test I/O and Run-Time Library operation...
VAX FORTRAN HPO V1.n-1 IVP PASSED
End of VAX FORTRAN HPO Installation Verification Procedure
    Installation of FORTHPO V1.n completed at 18:20
```