DECmigrate Version 1.1A for OpenVMS AXP Systems

Release Notes

November 1994

This document describes product restrictions and known problems.

Revision/Update Information:	This manual supersedes *DECmigrate Version 1.1 for OpenVMS AXP Release Notes*.
Operating System and Version:	OpenVMS AXP Version 6.1 or higher
Software Version:	DECmigrate Version 1.1A for OpenVMS AXP Systems

Digital Equipment Corporation
Maynard, Massachusetts

November 1994

This document was prepared using VAX DOCUMENT, Version 2.1.

Contents

Examples

Tables

1 DECmigrate for OpenVMS AXP Systems

DECmigrate for OpenVMS AXP Systems supports the migration of OpenVMS VAX applications to OpenVMS AXP systems. A DECmigrate utility, the VAX Environment Software Translator (VEST), translates executable and shareable OpenVMS VAX images into functionally equivalent images that run on OpenVMS AXP systems. A translated image is an OpenVMS AXP image containing both VAX code translated into Alpha AXP code as well as the original OpenVMS VAX image.

DECmigrate Version 1.1A is a maintenance release. It includes bug fixes but no new features.

These release notes describe:

- Associated documents (Section 2)
- Known restrictions and problems (Section 3) - changes are indicated in the margin.
- Documentation changes, errors, and omissions (Section 4)
- Unsupported features (Section 5)

2 Associated Documents

Refer to the following two documents for detailed information on installing and using DECmigrate:

- *DECmigrate for OpenVMS AXP Systems Installation Guide*

  This manual describes the procedures for installing DECmigrate for OpenVMS AXP Systems.

- *DECmigrate for OpenVMS AXP Systems Translating Images*

  This manual describes:

  - How to use VEST to convert most user-mode OpenVMS VAX images into translated images that can be run on OpenVMS AXP systems
  - How to improve the run-time performance of translated images
  - How to use VEST to trace OpenVMS AXP incompatibilities in an OpenVMS VAX image back to the original source files
  - How to use VEST to support compatibility among native and translated run-time libraries (RTLs). The manual also includes complete VEST command reference information.

3 Restrictions and Known Problems

This section describes restrictions and known problems in DECmigrate Version 1.1A.

The Version 1.1 problems that have been fixed are:

- Global symbols added to a shareable image that already has a .SIF file (previously Section 4.8)
- Stack mismatch when more than 63 arguments are used in a CALLS instruction (previously Section 4.9)

- Erroneous read of call frame reported (READCF0) (previously Section 4.11)

3.1 General Restrictions

- DECmigrate Version 1.1A can translate any translatable image that runs on OpenVMS VAX Version 5.4-2 or later. With some exceptions, a translatable image is any nonprivileged OpenVMS VAX image that was linked on VAX/VMS Version 4.0 or later and that runs on OpenVMS VAX Version 5.4-2. The appendix *Translation and Performance Restrictions* in the manual *DECmigrate for OpenVMS AXP Systems Translating Images* explains which images are not translatable.

Note

An otherwise translatable image that was linked on a version of OpenVMS VAX later than Version 5.4-2 may cause IDENTMISMATCH fatal errors when translated and run on OpenVMS AXP because of incompatible run-time library shareable images.

- DECmigrate Version 1.1A runs on OpenVMS AXP Version 6.1 and the images it translates require this version or later to execute. Translated images in general are forward compatible but not backward compatible— that is, images translated with DECmigrate Version 1.1A can run only on OpenVMS AXP Version 6.1 or later while images translated with DECmigrate Version 1.0 can run on OpenVMS AXP Version 1.0 and later. Refer to Table 1 to check the versions of OpenVMS AXP systems supported by the different versions of DECmigrate.

Table 1 Compatibility between Translated Images and OpenVMS AXP Versions

Images translated by DECmigrate	run on?	OpenVMS AXP V1.0	OpenVMS AXP V1.5	OpenVMS AXP V6.1
Version 1.0		Yes	Yes	Yes
Version 1.1		No	Yes	Yes
Version 1.1A		No	No	Yes

- Images linked under OpenVMS AXP Version 6.1 are translatable *if* they have no dependencies—that is, the images use only system services that were available for OpenVMS VAX Version 5.4-2.
- VEST cannot translate an image accessed by means of DECnet. If you include a node identifier in the file specification, VEST issues the message %VEST-F-BADEXE or %VEST-F-NOVM and completes with the message %VEST-F-TRANSFATAL. Copy the image to your local node before translating the image.
- A translated image incurs a fatal exception if it first opens the maximum number of files permitted to it and then tries to execute a complex VAX instruction. The crash occurs because TIE$SHARE cannot open TIE$EMULAT_TV; the limit on open files has already been reached. (TIE$EMULAT_TV emulates VAX instructions.)

3.2 Interoperability Restrictions

On an OpenVMS AXP system, native and translated images can interoperate with one another: a native image can call a translated image and a translated image can call a native image. The following restrictions pertain to interoperation between native and translated images:

- A native routine that either calls or is called by a translated image must be compiled with the /TIE qualifier and linked with the /NONATIVE_ONLY qualifier. Checking for interoperability between native and translated images occurs at run time. If the /TIE and /NONATIVE_ONLY qualifiers were not used to compile and link the native routine, an error occurs at run time when the native routine and a translated image attempt to interoperate. If such an error occurs, recompile and relink the native routine appropriately.

  Note

  The VAX MACRO-64 Assembler for OpenVMS AXP Systems does not support the /TIE switch. When coding procedure descriptors in VAX MACRO-64, the developer must specify a procedure signature block if the routine can be called by translated code. (Refer to the *OpenVMS Calling Standard*.)

- An access violation can occur at run time if a native routine that was not compiled with the /TIE qualifier makes an indirect call to a translated routine. The indirect call is made through a variable that contains the translated routine's address. When this happens, there is no autojacketing code in place to assist the native to translated call. The native code attempts to use the routine address as a native procedure descriptor. The code address of a native procedure is at offset PDSC$L_ENTRY, whose value is 8, from the base of the procedure descriptor. Because the translated routine address is treated as a procedure descriptor, the value at offset 8 from that address is used as the code to call. This usually results in an access violation.

  If you suspect you are encountering this problem, use a debugger to check the following:

  - Check that R27 points into a translated image.
  - Check that bits <31:2> of 8(R27) equal bits <31:2> of the ACCVIO address. (All bits are not used because Alpha AXP instructions are longword aligned.)
  - Check that R26 points into a native image.
  - Check that -4(R26) is a JSR R26,(R26) instruction.

  If all these checks prove to be true, recompile the native routine with the /TIE qualifier to enable autojacketing at run time.

- If translated code specifies a native routine as a change mode handler in a call to the $DCLCMH system service, the program either crashes or gives incorrect results.

- DECmigrate does not support shared PSECTs (for example, VAX COBOL EXTERNAL, VAX Basic COMMON, or VAX FORTRAN COMMON statements) between translated and native images. Sharing data in shared PSECTs between translated images is supported. In addition, sharing data allocated at run time (using $CRMPSC, for example) is supported between translated and native images.

- Translated code cannot call a jsbentry in native code, nor can native code call a jsbentry in translated code. The Alpha AXP calling standard supports call entries only.

3.3 Layered Product Shareable Libraries

Some OpenVMS AXP layered products do not allow both translated and native images to exist within the same process. Such a restriction may exist because translated and native versions of the product's shareable libraries may be incompatible. As you migrate an application, check for any such incompatibility for the products your application depends on.

For example, the DECwindows Motif for OpenVMS AXP software includes support for translated images for OpenVMS VAX DECwindows Motif images. (Select translated image support during the installation procedure for the DECwindows Motif product. See the chapter on installing the software in the *DECwindows Motif for OpenVMS AXP Version 1.1 Installation Guide* for instructions.)

The DECwindows Motif for OpenVMS AXP shareable images that are used with translated images are different from and not compatible with the shareable images used for native images. As a result, the following image restrictions apply:

- Do not use both native and translated images that use DECwindows Motif for OpenVMS AXP software in a process.

  Either port or translate all the images that use DECwindows Motif for OpenVMS AXP and that call each other.

- Translated images that dynamically activate a DECwindows Motif image using LIB$FIND_IMAGE_SYMBOL from a translated image are not supported.

3.4 Circular Dependencies Between Shareable Images

If a circular dependency exists between two OpenVMS VAX shareable images, but neither image has initialization code (LIB$INITIALIZE), the images activate. However, if they have initialization code, they do not activate. These kinds of images do not activate on OpenVMS AXP when translated because VEST always adds initialization code to translated shareable images. The OpenVMS AXP system reports a circular dependency.

3.5 Unreported Call Mask Overlaps

A bug in VEST's analysis phase occasionally causes VEST not to report call mask overlaps when more than one call mask uses a common byte. This bug does not affect code generation.

3.6 Unreported Invalid Case Statement

VEST may fail to report an invalid case statement if the case limit itself is invalid and if some of the case statement branches point to invalid code.

3.7 Unable to Translate Due To Environment Problems

If environment problems cause VEST to be unable to translate an image, VEST may not report what the problem is. An insufficient quota or an offline output disk are examples of environment problems.

3.8 Erroneous Non-Standard CALL Reported (NONSTDCALL)

VEST often reports a non-standard CALLed routine when both of the following conditions are true:

- Two or more entry points feed into common exit code.
- The call masks for the entry points are different.

This bug does not affect code generation.

3.9 VEST/DEPENDENCY and Shareable Images

OpenVMS VAX images refer to shareable images supplied with the OpenVMS VAX system. VEST/DEPENDENCY cannot distinguish between shareable images for which .IIF files have been supplied by Digital or a third party and shareable images that are part of the software being translated. As a result, VEST /DEPENDENCY attempts to analyze all shareable images in the dependency tree. As shown in Example 1, when run on OpenVMS AXP, VEST/DEPENDENCY may report that files such as LIBRTL.EXE and MTHRTL.EXE cannot be opened for analysis. Whether VEST/DEPENDENCY reports these messages or not, edit the MMS file produced to remove all references to .IIF files and shareable images that are not a part of the software being translated. After editing the MMS file, you can process the file normally. Example 2 contains an example of an MMS file before it is edited. Example 3 contains an example of the same MMS file after editing.

Example 1 Output from $VEST/DEPENDENCY SIEVE

```
$ VEST/DEPENDENCY SIEVE
%VEST-I-READIMAGE, Reading image file DISK:[USER]SIEVE.EXE;
%VEST-F-OPENIN, Error opening SYS$COMMON:[SYSLIB]VAXCRTL.EXE; as input
-RMS-E-FNF, file not found
%VEST-F-DEPENDFATAL, Complete dependency analysis was impossible
```

Example 2 MMS file produced by $VEST/DEPENDENCY SIEVE

```
!  DISK$:[USER]SIEVE.MMS; created by VEST/DEPENDENCY on Mon May 17 16:01:18 1993

! Define default VEST switches.  They can be overridden from the command line
!  by saying MMS/MACRO="VESTSWITCHES=/FLOWGRAPH" for example
!
VESTCOMMAND = VEST $(MMS$SOURCE) $(VESTSWITCHES)

TARGET : DISK$:[USER]SIEVE_TV.EXE
  !

DISK$:[USER]SIEVE_TV.EXE : DISK$:[USER]SIEVE.EXE, -
 VEST$FULL_INCLUDE:VAXCRTL.IIF
  $(VESTCOMMAND)
```

Example 3 Edited MMS file produced by $VEST/DEPENDENCY SIEVE

```
!  DISK$:[USER]SIEVE.MMS; created by VEST/DEPENDENCY on Mon May 17 16:01:18 1993

! Define default VEST switches.  They can be overridden from the command line
!  by saying MMS/MACRO="VESTSWITCHES=/FLOWGRAPH" for example
!
VESTCOMMAND = VEST $(MMS$SOURCE) $(VESTSWITCHES)

TARGET : DISK$:[USER]SIEVE_TV.EXE
  !

DISK$:[USER]SIEVE_TV.EXE : DISK$:[USER]SIEVE.EXE
  $(VESTCOMMAND)
```

3.10 FLOWGRAPH and DSTGRAPH /OUTPUT qualifier

If the /OUTPUT qualifier is specified with only a directory specification, FLOWGRAPH and DSTGRAPH generate an output file named:

```
.PS
```

For example,

```
$ FLOWGRAPH/OUTPUT=DISK$:[USER] FOO.GRAPH
```

results in an output file named

```
DISK$:[USER].PS
```

3.11 Global Pages and Global Sections Not Checked During Installation

The DECmigrate kit does not check to see if the system has sufficient global pagelets and global sections at installation time. DECmigrate requires that there be two free global sections and 26 free global pagelets for installation of message files.

If you install DECmigrate on a system that does not have enough free global pagelets or global sections, the installation fails with a warning similar to the following:

```
%INSTALL-E-FAIL, failed to REPLACE entry for DISK$SYSTEM:<SYS0.SYSCOMMON.SYSEXE>DCLTABLES.EXE
-SYSTEM-F-GPTFULL, global page table is full
%INSTALL-W-DELETED, previous Known File Entry has been deleted
```

3.12 /PRESERVE=INSTRUCTION_ATOMICITY Restriction

The following restrictions apply when you use the /PRESERVE=INSTRUCTION_ATOMICITY qualifier setting when translating an image:

- Preserved instruction atomicity applies only to code that is translated, not to complex VAX instructions. (Complex VAX instructions are executed by the shareable image TIE$EMULAT_TV.)
- Instruction atomicity is never preserved for the following VAX instructions, even when translated:

  ASHL
  ASHQ
  BSBB
  BSBW
  CALLG
  CALLS
  EDIV
  INSV
  JSB

3.13 Universal Jsbentry Restriction

If a shareable image contains any universal jsbentry routines that return information by means of condition codes, it must be translated with the /PRESERVE=CONDITION_CODES qualifier.

3.14 Source-language-specific Restrictions

The following restrictions pertain to specific languages.

3.14.1 VAX Ada

Applications written in VAX Ada cannot be translated. To migrate such applications, you must recompile the source code and relink on an OpenVMS AXP system.

3.14.2 VAX BASIC

If an image contains any VAX BASIC code that uses a WHEN ERROR construct, it must be translated with the /OPTIMIZE=NOSCHEDULE qualifier. The WHEN ERROR construct may not function properly if a complex instruction emulation routine is executing, even if you translated the image with the /OPTIMIZE=NOSCHEDULE qualifier.

3.14.3 VAX COBOL

The translated version of a VAX COBOL program that contains either the CANCEL verb or the IS INITIAL program phrase of the PROGRAM-ID paragraph encounters a run-time error like the following:

```
$ RUN COBOLTEST_TV
%COB-F-CANFAIL, CANCEL failed on routine (MODULE-NAME)
-SYSTEM-F-PAGOWNVIO, page owner violation
```

If you encounter this run-time error, you must relink the original VAX COBOL program with the module SYS$LIBRARY:COBRESTVA.OBJ (distributed in the DECmigrate kit) and then retranslate. VEST is unable to detect this problem when translating the image.

3.14.4 VAX DIBOL

If an image contains any VAX DIBOL code, it must be translated with the /PRESERVE=SAFETY qualifier.

3.14.5 VAX C

The following restrictions pertain to code in VAX C:

- By default, VAX C programs are linked with VAXCRTL. When VEST translates a VAX C program linked with VAXCRTL, it must translate the entire library as well as the actual program. It is recommended that you relink a C program on OpenVMS VAX before you translate it. For example:

  ```
  $link sieve,sys$input:/opt
  sys$library:vaxcrtl.exe/share
  <Ctrl/Z>
  ```

  Linking a VAX C program in this way causes it to share VAXCRTL rather than include it. Translating the image will be quicker and will incur fewer error messages.

- If a program uses the VAXCRTL routine brk( ) to release dynamic memory (that is, a lower break address is requested than the current break address), the next attempt by TIE to use a complex instruction routine may result in a fatal memory access violation. This may happen because the complex instruction routines are in a separate image, TIE$EMULAT_TV.EXE, which is dynamically activated via LIB$FIND_IMAGE_SYMBOL on the first use of one of the routines. Depending on when this occurs and the address passed to the brk( ) call that releases memory, the memory into which TIE$EMULAT_TV.EXE was loaded may also be released.

To avoid this problem, never use brk( ) to release memory or be sure to execute a complex VAX instruction prior to getting the break address that is later used to release memory. The use of brk( ) to allocate memory translates without problems.

- A translated VAX C program that uses vfork( ) and any executive function may hang at run time. If the child process of the VAX C program aborts erroneously, it may hang waiting for a mailbox I/O to be completed. One workaround is to prevent the child process from aborting.

3.15 Linked with /NOSYSSHR Restriction

VEST is unable to translate any image that has been linked with the qualifier /NOSYSSHR and whose system version array is not empty. (Use the ANALYZE /IMAGE command to examine the input image's system version array information to determine whether or not the array is empty.) Relink such an image without this qualifier before attempting to translate it.

3.16 Floating Point Restrictions

The following restrictions pertain to floating point operations:

- Certain code in the Math Run-Time Library (MTHRTL) assumes D56 precision. Because the VEST default is to use D53 precision (/FLOAT=D53_FLOAT), a translated image can generate inaccurate results (`11 MOD 2 = 3`, for example) if it happens to use the D56-dependent MTHRTL. If such obviously inaccurate results occur, retranslate the image using the qualifier /FLOAT=D56_FLOAT. The translated image then uses MTHRTL_D56_TV.EXE rather than MTHRTL_D53_TV.EXE.
- In some cases, floating point instructions operating on the same data generate a trap on an OpenVMS AXP system but not on an OpenVMS VAX system. Specifically, VAX floating point instructions on an OpenVMS AXP system generate traps for the *dirty zeros* that VAX hardware handles as zeros. Dirty zeros are floating point values that are alternate encodings for zero. To retain compatibility with translated code that performs operations using dirty zeros, the TIE includes a condition handler that corrects the dirty zeros and retries the floating point operation. However, the handler succeeds only if the qualifier /PRESERVE=FLOAT_EXCEPTIONS was used when the image was translated.

  Images that were not translated with /PRESERVE=FLOAT_EXCEPTIONS and that perform an operation on a dirty zero incur an HPARITH exception with a summary status that has bit 1 set. If your translated application incurs one of these exceptions, retranslate with /PRESERVE=FLOAT_EXCEPTIONS. VAX dirty zeros commonly result from not initializing floating data to 0. In this case, if you are planning to eventually recompile and relink the application on an OpenVMS AXP system, you may need to make changes to the source code that uses dirty zeros for the port to be successful.
- Alpha AXP D53 floating point (D_floating point as a 53-bit fraction instead of a 56-bit fraction) is VAX D_floating converted to G_floating representation. This conversion leads to the following problem. Consider the following VAX instruction sequence:

```
MOVD    (SP),R2
MOVD    R2,-(SP)
```

VEST translates these VAX instructions into Alpha AXP code like the following:

```
LDG     F2,0(R14)       ! Pickup D float
CVTDG   F2,F2           ! Convert to Canonical G Form with rounding
CVTGD   F2,F17          ! Convert back to D Form for storing
STG     F17,-8(R14)     ! Store the result
```

At run time, the VEST-generated code uses rounding to obtain the most accurate G_floating value when converting the D56 floating point to G canonical form. In some cases, the conversion to G canonical form may round up the D_floating value to create an exponent that cannot be represented in D_floating. When this happens, the CVTGD operation incurs an HPARITH trap with floating overflow as the summary reason.

If a translated image incurs this problem at run time, it must be retranslated with the VEST qualifier /FLOAT=D56_FLOAT to execute properly.

- Regardless of the /FLOAT setting for floating point instructions when you translate an image, the following VAX opcodes are always executed with D56 precision at run time:

  ACBD
  EMODD
  POLYD
  CVTDH
  CVTHD

3.17 PROBER and PROBEW Restriction

The VAX instructions PROBER and PROBEW detect neither fault-on-read (FOR), fault-on-execute (FOE), nor fault-on-write (FOW), three types of faults not present on VAX-based systems. If a native OpenVMS AXP image sets pages in memory as FOR, FOE, or FOW, and then calls a routine in a translated RTL that uses PROBER or PROBEW on an address within the protected range, PROBER or PROBEW fails to detect that an access violation occurs when that page is referenced.

3.18 BISPSW, RET, or REI Instruction Restriction

When a BISPSW, RET, or REI instruction operates on the program status word (PSW) and forces the resulting program status longword (PSL) to have both the N and Z bits set (an inconsistent VAX value), a conditional branch based on the value of the PSL could go the wrong way.

3.19 Condition and Exception Handler Restrictions

- A restriction on the type of condition handler that can be established exists for both native and translated images. A native routine cannot establish a translated condition handler, nor can a translated routine establish a native condition handler. If a native or translated image violates this restriction, the run-time results are unpredictable.
- The OpenVMS AXP system does not support a native caller passing a translated routine address to $SETEXV. An error occurs when a native caller attempts to do so. Users should ensure that translated condition handlers are established by translated code.

- Translated images with exception handlers that depend on receiving the correct Program Status Longword (PSL) might not function properly. When exceptions are reported, the Alpha AXP Program Status (PS) is reported in the signal array instead, since there is no VAX PSL.
- Translated images with exception handlers that depend on modifying the PSL in the signal array will not function properly. The modified PSL will not be propagated back to the faulting code.
- LIB$DECODE_FAULT is the Decode Instruction Stream During Fault routine in the OpenVMS VAX run-time library (RTL). It is a routine for building condition handlers that process instruction fault exceptions. It is called from a condition handler.

  This routine is very specific to the VAX architecture. Programs that depend on the VAX behavior of LIB$DECODE_FAULT to execute properly do not execute translated. In addition, code in an application that depends on LIB$DECODE_FAULT needs to be rewritten before porting the application to an OpenVMS AXP system.
- VAX code that uses DECthreads TRY/CATCH exception handling is not translatable. DECthreads exception handling depends on executing in the frame of its caller. Because translation forces an extra frame between the caller's and DECthread's exception code, exception handling won't work properly.

4 Documentation Changes, Errors and Omissions

The following corrections apply to the documentation.

4.1 Increasing System Quotas to Translate Large Images

To translate large images, you may need to increase system quotas so that VEST has enough memory to complete the translation. The relevant quotas that you or a system manager may need to adjust are:

BYTLM
WSDEF
WSQUO
WSEXTENT
PGFLQUO

4.2 User Action for ISDCONFLICT Error Message

In the manual *DECmigrate for OpenVMS AXP Systems Translating Images*, change the **User Action:** section that describes the ISDCONFLICT error message to read:

> Relink the image using /BPAGE=16 as a qualifier to the LINK command ($ LINK/BPAGE=16) on an OpenVMS VAX Version 5.5 or later system.

4.3 PRIVSS Error Message

The manual *DECmigrate for OpenVMS AXP Systems Translating Images* does not include the following description of the PRIVSS error message:

PRIVSS, Image calls system service 'service'. Execution of privileged code in a translated context is not supported.

Facility: VEST, VAX Environment Software Translator utility

Explanation: The image makes a call to the change mode service noted in the PRIVSS message. VEST can translate the image, but if the translated image calls the service and then attempts to execute privileged code, a fatal error occurs.

User Action: The user action depends on the image being translated.

4.4 No Code Optimizations with /PRESERVE=SAFETY

The VEST qualifier /PRESERVE=SAFETY turns off all code optimizations. This setting generates safe code that includes no assumptions about call targets or resource usage or any other assumptions that can cause VEST to generate erroneous translated code based on faulty analysis.

5 Unsupported Feature—GIRDLE Information File Compressor

The kit includes an unsupported program called the GIRDLE Information File Compressor.

Note

Because this program is unsupported, Digital does not guarantee its performance. Please do not submit a Software Performance Report for this feature.

The GIRDLE program takes a list of hand-edited information files (.HIF files) for the same image and creates a single .HIF output file. The program removes duplicate entries, merges sections based on the same translated version of an image, and preserves any entries presumed to be entered by hand. GIRDLE may also be used with image information files (.IIF files) to sort entries to make them more understandable.

After DECmigrate has been installed, the files GIRDLE_TV.EXE and GIRDLE.CLD are located in SYS$SYSROOT:[SYSHLP.EXAMPLES.VEST].