VSI Fortran for OpenVMS Integrity servers

Release Notes

August 2016

This document contains information about new and changed features in this version of VSI Fortran for OpenVMS Integrity servers.

Software Version:	VSI Fortran V8.3 ECO 02
Operating System:	OpenVMS Integrity servers Version 8.4-1H1 or later

VMS Software, Inc.
Bolton, Massachusetts

© Copyright 2016 Hewlett Packard Enterprise Development, LP

HPE CONFIDENTIAL. This software is confidential proprietary software licensed by Hewlett Packard Enterprise Development, LP, and is not authorized to be used, duplicated or disclosed to anyone without the prior written permission of HPE.

© Copyright 2016 VMS Software, Inc.

VMS SOFTWARE, INC. CONFIDENTIAL. This software is confidential proprietary software licensed by VMS Software, Inc., and is not authorized to be used, duplicated or disclosed to anyone without the prior written permission of VMS Software, Inc.

This document was prepared using DECdocument, Version 3.3-1n.

Contents

1 VSI Fortran Version V8.3 ECO 02 Overview

1.1 Operating System Compatibility 1–1
1.2 Getting Help and Reporting Problems 1–1

2 New and Changes Features

2.1 New and Changes Features in V8.3 ECO 02 2–1
2.2 Corrected Problems in V8.3 ECO 02 2–1

3 New and Changed Features in Previous Releases

3.1 New and Changed Featuers in V8.3 ECO01 3–1
3.2 New and Changed Features in V8.3 3–1
3.2.1 Corrected Problems in V8.3 3–1
3.3 New and Changed Features in V8.2 ECO 01 3–2
3.3.1 Predefined Symbols for Floating-Point Formats 3–2
3.3.2 Corrected Problems in V8.2 ECO 01 3–3
3.4 New and Changed Features in V8.2 3–4
3.4.1 New /ALIGN=COMMON=PAD_ALIGN_SIZE qualifier 3–4
3.4.2 New /ASSUME=64BIT_STRING_PARAMS qualifier 3–4
3.4.3 Additional Changes and Corrected Problems in V8.2 3–5
3.5 Problems Corrected in V8.1 ECO 01 3–5
3.6 New and Changed Features in V8.1 3–6
3.6.1 IA64 and _IA64_ predefines are implemented. 3–6
3.6.2 Additional Changes and Corrected Problems in V8.1 3–6
3.7 New and Changed Features in V8.0 3–7

4 Differences from HP Fortran for OpenVMS Alpha Systems

4.1 Floating-Point Arithmetic 4–1
4.2 No F77 Compiler 4–2
4.3 Dirty Zeros Trigger Exceptions 4–3
4.4 Math Algorithm Differences 4–3
4.5 Math Library Exceptions 4–3
4.6 /ARCH and /OPT=TUNE qualifiers 4–3

5 Known Problems and Incomplete Functionality

5.1 Bugs Not Fixed 5–1
5.2 Disabled /CHECK=ARG_INFO,FP_EXCEPTIONS,UNDERFLOW 5–1
5.3 IEEE Exceptions Detected During Compilation 5–1

6 VSI Fortran Documentation and Online Information

1

VSI Fortran Version V8.3 ECO 02 Overview

VSI Fortran Version V8.3 ECO 02 provides a version of the VSI Fortran compiler which includes corrections to problems discovered in earlier versions.

This chapter provides general information about the VSI Fortran product. The remaining chapters provide the following information:

- Chapter 2 highlights new and changed features in this version of VSI Fortran.
- Chapter 3 highlights new and changed features in previous versions of VSI Fortran.
- Chapter 4 highlights major differences between VSI Fortran and HP Fortran.
- Chapter 5 details some known problems and incomplete functionality in this version of the compiler.
- Chapter 6 lists the organization of the VSI Fortran documentation set.

1.1 Operating System Compatibility

VSI Fortran is supported on OpenVMS Integrity servers V8.4-1H1, the latest which has been released at the time of printing. It is likely that VSI Fortran will work correctly on all later versions of OpenVMS Integrity servers. If you have questions about version compatibility, please contact us as described in Section 1.2.

1.2 Getting Help and Reporting Problems

If you encounter a problem while using VSI Fortran, report it to VSI or HP, depending on where you purchased software support.

If an error occurs while VSI Fortran is in use and you believe the error is caused by a problem with VSI Fortran, take one of the following actions:

- If you have a Software Product Services Support Agreement, consider contacting HP Global Services by telephone (in the United States, 1-800-354-9000) or by using the electronic means provided with your support agreement.
- Customers without a service contract can arrange for per-call support.

When you initially contact HP Global Services, please indicate the following:

- The name (OpenVMS Integrity servers) and version number of the operating system you are using
- The name (VSI Fortran) and complete version number of the compiler you are using (the version number is displayed at the end of a compiler listing file)
- The hardware system you are using (such as a model number)

- How critical the problem is
- A very brief description of the problem (one sentence if possible)

When you submit information electronically or are speaking on the phone to the appropriate support specialist, you can provide more detailed information. This includes the specific commands used to compile and link the program, the error messages displayed, and relevant detailed information (possibly including source program listings). Please try to narrow the cause of the problem to a specific source module or lines of code.

Support personnel might ask for additional information, such as listings of any command files, INCLUDE files, relevant data files, and so forth. If the program is longer than 50 lines, submit a copy of it electronically or provide machine-readable media.

Experience shows that problem reports sometimes do not contain enough information to duplicate or identify the problem. Concise, complete information helps HP give accurate and timely service to software problems.

To obtain information about purchasing HP support services, please contact your local HP sales representative.

You can also send comments, questions and suggestions about the VSI Fortran product to `fortran@hp.com` or `info@vmssoftware.com`. Note that these addresses are for informational inquiries and is not a formal support channel.

2
New and Changes Features

2.1 New and Changes Features in V8.3 ECO 02

VSI Fortran V8.3 ECO 02 corrects a regression in VSI Fortran V8.3.

The image identification for the Fortran compiler is "F90 V8.3-104957".

The FORTRAN/VERSION string is "V8.3-104957-50Q83".

2.2 Corrected Problems in V8.3 ECO 02

- QXCM1001386870/IF512: Fix a recent regression that incorrected issued an error message (The highest data type rank permitted is INTEGER(KIND=8)) for IF conditionals with nested structure references.
- QXCM1001434336/IF513: Fix a compiler ACCVIO while attempting to issue an error about ALLOCATABLE elements in COMMON or EQUIVALENCE statements. The compiler was accessing unallocated memory and sometimes would ACCVIO and sometimes print the correct message.

  ```
  %F90-E-ERROR, Variables containing ultimate allocatable
   array components cannot appear in COMMON or EQUIVALENCE
   statements.
  ```

- QXCM1001326600/IF514: An incorrect run-time error (FOR-F-INVREALLOC) would occur when using 64-bit addressing on OpenVMS I64. The compiler did not properly construct the 64-bit array descriptor.
- QXCM1001374529/IF515: An ACCVIO would occur when allocation many 64-bit arrays in P2 space. The compiler did not probably reference a 64-bit address field in a 64-bit array descriptor.
- QXCM1001383057/IF516: Specifying "OPTIONS /EXTEND_SOURCE" in the source file would incorrect override a /CHECK=BOUNDS qualifier specified on the command line.
- QXCM1001382426/IF517: The effective address for a multi-dimensional 64-bit array allocated in P2 space was incorrectly truncated to 32-bits.
- QXCM1001402376/IF518: The effective address for a multi-dimensional 64-bit array parameter was incorrect truncated to 32-bits.
- qxcm1001420475/IF519: Using an empty MODULE with no body would sometimes result in an ACCVIO during compilation. The compiler was accessing unallocated memory.

3

New and Changed Features in Previous Releases

3.1 New and Changed Featuers in V8.3 ECO01

- One of the fixes included in V8.3 caused a regression that generated an Internal Compiler Error for certain array-to-array assignment statements. The fix for QXCM1001276035 (see Section 3.2.1) was incorrect. The change has been removed.

  Looking further at the problem reported by QXCM1001276035, the Fortran program was actually invalid. It assigned a smaller array constructor to a larger array. This violates Fortran 95's rule about arrays must have the same shape to be assignment compatible. The original behavior of getting a FLTINV error is perfectly reasonable.

3.2 New and Changed Features in V8.3

VSI Fortran V8.3 is the VSI release of HP Fortran. It is based on the most recent HP Fortran ECO kit, but contains a newer backend and includes bug fixes made by HP subsequent to the release of the HP Fortran ECO kit from 2012.

The image identification for the Fortran compiler is "F90 V8.3-104955".

The FORTRAN/VERSION string is "V8.3-104955-50P3Q".

3.2.1 Corrected Problems in V8.3

- QXCM1001028919/IF492: The DCOTAND intrinsic returns the wrong result when compiling the program with /FLOAT=D_FLOAT.
- QXCM1001174630/IF500: The compiler would generate invalid code when calling a character function parameter when the calling routine was inside a module.
- QXCM1001052753/QXCM1001101077/QXCM1001287314/IF493/495/505/510: Assorted programs would get an Internal Compiler Error when using string arguments in routine calls.

  The exact problem is difficult to describe with Fortran source has it requires particular transformations of basic-blocks such that the actual parameter is a reference to a string value in an earlier basic block.

  The solution is to save that earlier string value into a temporary variable and then fetch that temporary variable as part of the routine call.

- QXCM1001276035/IF506: A program might generate an invalid floating error (FLTINV) at run-time when assigning a smaller array constructor into a larger array and using VAX floating.
- QXCM1001257380/IF507: The compiler generated incorrect code when passing an allocatable object to a routine that returns a string whose size is determined by the size of the allocatable object.
- QXCM1001188805/IF501/IF511: The compiler generated incorrect code handling complex expressions involving derived types 3 or more levels deep.
- QXCM1001230271/IF503: The /FLOAT, /ASSUME=DUMMY, or /[NO]G_FLOATING options with the OPTIONS statement only worked when they were the last option specified.

3.3 New and Changed Features in V8.2 ECO 01

This section provides highlights of new and changed features, and problems corrected in HP Fortran V8.2 ECO 01.

The image idenfication string for the Fortran 95 compiler is V8.2-104953-50M2N.

3.3.1 Predefined Symbols for Floating-Point Formats

In this release, the IEEE_FLOAT, G_FLOAT, and D_FLOAT are the three new predefined symbols for floating-point formats that are added in the HP Fortran compiler.

A sample program demonstrates the usage of the new predefined symbols:

```
    program m_p_test
cDEC$ IF DEFINED (IEEE_FLOAT)
    write (*,*) 'Using IEEE floating-point'
cDEC$ ELSE   IF DEFINED (G_FLOAT)
    write (*,*) 'Using G_Float floating-point'
cDEC$ ELSE   IF DEFINED (D_FLOAT)
    write (*,*) 'Using D_Float floating-point'
cDEC$ ELSE
    write (*,*) 'None of IEEE,G_FLOAT,D_FLOAT used'
cDEC$ ENDIF
    stop
    end
$ fortran dec
$ link dec
$ run dec
Using IEEE floating-point
$ fortran dec/float=g_float
$ link dec
$ run dec
Using G_Float floating-point
$ fortran dec/float=d_float
$ link dec
$ run dec
Using D_Float floating-point
```

3.3.2 Corrected Problems in V8.2 ECO 01

- QXCM1000785692: On Integrity servers, the RANDU subroutine results in the out of range error (where, the range is between 0.0 to 1.0). This issue is fixed in this release.
- QXCM1000809115: When compiling Fortran code that contains nested include files, using /LIST/SHOW=ALL qualifier does not expand all the include files in the resultant .LIS file. This issue is fixed in this release.
- QXCM1000888536: When using IADDR function, the following error message is displayed:

  ```
  %F90-F-FATAL, **Internal compiler error: internal abort**
  ```

  This problem occurs due to the usage of potential future intrinsic called IADDR function. This issue is fixed in this release.

  Note

  Internal compiler errors might be generated in several scenarios wherein the compiler detects an unsupported behavior.

- QXCM1001052753: The compiler generates the following error message:

  ```
  Assertion failure: Compiler internal error - please submit problem report
  %GEM-F-ASSERTION, Compiler internal error -  please submit problem report
  ```

  This error occurs as the compiler frontend does not generate the correct intermediate language when using XML files in multiple scenarios in the Fortran program. This issue is fixed in this release.
- QXCM1001107164: When compiler is setting attributes for ACTPOS variables in an expression, certain attributes such as length are not set for concatenated strings. Now, the compiler is modified to get the length.
- QXCM1000857610: While compiling applications on Integrity using /OPT=LEVEL=3 and above, there is a possibility that certain variables might get optimized. Due to this, the application fails with the following error message:

  ```
  %SYSTEM-F-INTDIV, arithmetic trap, integer divide by zero
  at PC=00000000037CA971, PS=0000001B
  ```

  This issue is fixed in this release.
- In certain cases, the DWARF data generated for bit fields are insufficient. If the bit size is multiple of eight, the bit field tags are not generated. This issue is fixed in this release.
- When INTEGER*8 variable is defined in a structure that has a union definition, the members of the union are not aligned correctly. Due to this, the field of type INTEGER*8 does not get populated correctly. This issue is fixed in this release.
- Unwind descriptors are not generated correctly. This issue is fixed in this release.

- The /ASSUME=F77RTL qualifier is enabled again. It specifies whether certain run-time I/O defaults match the HP Fortran 77 compilers. The default is /ASSUME=NOF77RTL qualifier, which specifies that Fortran 95 semantics are to be followed.
- The Fortran compiler is changed to use __STDC__ macro to compile with HP C Version 7.3.

3.4 New and Changed Features in V8.2

This section describes new and changed features and problems corrected in HP Fortran V8.2.

The image identification string for the Fortran 95 compiler is V8.2-104939-50H96.

3.4.1 New /ALIGN=COMMON=PAD_ALIGN_SIZE qualifier

When the qualifier /ALIGN=COMMON=MULTILANGUAGE is in effect, COMMON blocks within the compiled Fortran source are padded at the end of each such COMMON block in the same way that the C compiler would pad it, i.e. by a pad size based on the data types in the block.

HP Fortran 95 previously contained an error (fixed in V7.6) which caused the padding to be based on the alignment specified in a directive. This error would cause a COMMON block that was aligned on an 8K boundary to also be padded out to an 8K size, for example.

The new /ALIGN=COMMON=PAD_ALIGN_SIZE qualifier provides the previous padding behavior.

3.4.2 New /ASSUME=64BIT_STRING_PARAMS qualifier

By default, when a string variable is passed as an argument, it is passed with a 32-bit descriptor. When a 64-bit descriptor is desired (for example, when calling a routine written in another language using 64-bit descriptors), you need to individually designate each string variable for which you would like the 64-bit descriptor.

Some customers have requested an option to allow all such string variables used within a source file to be passed or received with 64-bit descriptors without needing to so specify for each individual string variable.

The new [NO]64BIT_STRING_PARAMS keyword for the /ASSUME qualifier can be used to specify that requested behavior. When /ASSUME=64BIT_STRING_PARAMS is in effect, then all string variables used as parameters will be so used with 64-bit descriptors.

Note

This feature must be used with caution, as unexpected side effects may cause you problems. For example, because the Fortran Runtime Library expects to receive parameters with 32-bit addresses, statements that access the RTL either implicitly (e.g. print statements) or explicitly may fail when passed string parameters with 64-bit addresses.

The default is /ASSUME=NO64BIT_STRING_PARAMS.

3.4.3 Additional Changes and Corrected Problems in V8.2

- QXCM1000375413: aka Fortran 3248: V8.0 no longer allows OPEN/CLOSE file= clause to refer to an array.
- QXCM1000378452: aka IF448: Using the DEC$FORRTL that comes with V8.3, the user\222s application fails when calling LIB$SIGNAL.
- QXCM1000453936: aka IF467: Compilation fails erroneously on complex structure symbol.
- QXCM1000413720: aka IF457: Unexpected compiler errors when using complex structures in V7.6 and above.
- QXCM1000403722: aka IF455: Unexpected compiler errors moving from F77 to F90
- QXCM1000443205: aka IF465: Integrity only. VMS Fortran RTL changes floating point trap bits.

3.5 Problems Corrected in V8.1 ECO 01

This section provides highlights of problems corrected in VSI Fortran V8.1 ECO 01

The image identification string for the Fortran compiler is V8.1-1-104930-50GBS.

- 75-115-26 (Fortran RunTime Library bug, fixed in OpenVMS Integrity Servers Version 8.3R): Interaction problems with LIB$ESTABLISH(LIB$SIG_TO_ RET) and CLI$DCL_PARSE.
- QXCM1000372297: Erroneous %SYSTEM-F-NOCALLTRANS: A program trying to call a VAX subroutine processed through the VEST binary translator–although compiled with the correct switch (/TIE) and linked with the correct option (/NONATIVE)–results in an erroneous %SYSTEM-F-NOCALLTRANS failure.
- QXCM1000362711: Erroneous FATAL error when defining 21 or more CDD dictionaries: When a source file defines 21 or more CDD dictionaries, the compiler fails with F90-F-FATAL, Include files and/or DICTIONARY statements nested too deeply although the DICTIONARY statements are not nested.
- QXCM1000353431: Fortran compiler generates bad code involving stack pointer GB11074.
- QXCM1000349663: Integrity only: Erroneous %DEBUG-E- NOSYMBOL, symbol '\222'A\222: Look for a -what fixed this?
- QXCM1000343352: Array element argument passed incorrectly when the code is compiled nooptimized.

  When the argument being passed is an array element using a constant subscript, and the code is compiled /NOOPT, the first element of the array is passed rather than the specified element. When compiled /OPT, the correct argument is passed.
- QXCM1000340837: Erroneous syntax errors are generated for logical statements containing 2-level structure records.

When the source code contains logical expressions that are not fully qualified with parentheses and those expressions contain 2-level deep structure records, incorrect error messages are generated by the compiler.

- IF425: Internal compiler error after processing an INTENT attribute on an entry point name.

  Although the compiler correctly diagnoses the invalid statement, it continues on to fail with an internal compiler error.

- IF424: Erroneous syntax errors are generated for logical statements using .not. and multi-level structure syntax.

- PTR 75-13-1667: %F90-F-FATAL internal compiler error occurs following this valid I message. %F90-I-INFO, A name that identifies a global entity is used to identify another global entity in the same program. [JSCT]

- IF417: Declaring an EXTERNAL routine with the same name as a common block results in %F90-E-ERROR instead of the expected %F90-I-INFO.

- IF415: F90 asserts when compiling an initialized UNION definition.

- IF412: /SHOW=MAP emits no information for modules.

3.6 New and Changed Features in V8.1

This section provides highlights of problems corrected in HP Fortran V8.1.

The image identification string for the Fortran compiler is HP Fortran V8.1-104922-50G4H.

3.6.1 IA64 and _IA64_ predefines are implemented.

3.6.2 Additional Changes and Corrected Problems in V8.1

- PTR 75-102-1771: The compiler incorrectly reports an error for a statement where a recordseparator single quote has a partner in a comment following an I/O statement.

- PTR 75-102-1783: Use of /NOI4 causes an OPEN statement to result in "%FOR-F-INVARGFOR, invalid argument to FORTRAN RunTime Library." For KEY= in OPEN statement, the data_type should be either longword or byte_string (integer or character), independent of the kind of integer. Thus, /NOI4 should have no effect on the data_ type in the key block.

- PTR 221-2-876: A program aborts at run time when it initializes the elements of an allocatable array before the array has been allocated.

- PTR 221-2-1067, QXCM1000208533: Compiletime (Unix) or runtime failure due to problems with ACTPOS. Generally an ACTPOS is treated as a variable, but when it is an expression under the covers, we need to be more careful in handling it. In this particual example, a character function result gets its size from a call to another function, passing an expression (the ACTPOS) as its argument.

- QXCM1000330109: Compiler gets an access violation on certain instances of BLOCK DATA programs which initialize records.

- QXCM1000314367: Erroneous syntax errors are generated for certain combinations of parentheses and single quotes.

- QXCM1000214854: F90-F-FATAL: ICE when compiling program with a record named "OR".
- QXCM1000221329: Runtime failure due to incorrectly computing the address of the character structure component X%ARRAY(1)%STRING that is passed to CTRIM.
- QXCM1000221087: Public components of an object are not accessible in certain cases.
- QXCM1000226426: F90 does not pad/align the same as F77 and C when compiled with the /ALIGN=COMMON=MULTILANGUAGE option.
- QXCM1000242932: The compiler loops infinitely while loading a module in certain cases.
- QXCM1000246439: %SIZEOF (i) gives %F90-F-FATAL, **Internal compiler error**. Occurs for function references which have \222%\222 prefix and which are not %LOC, %LOC2, %REF, %VAL or %DESCR.
- QXCM1000283038: Problems when a statement function reference occurs while in a hoisting context. Ensure that the code to set up the statement function arguments precedes the code that uses them. Example is obscure: A(I:I,J)=SF(M,J)
- QXCM1000296751: In some cases, /CHECK=BOUNDS is subverted by optimization and the program continues past the specified bound.
- QXCM1000303300: The HP Fortran for OpenVMS User Manual had incorrect information about form feeds in column 1 in a source code listing. The manual has been corrected to say: "If a source line contains a formfeed character, that line is printed but the formfeed character is ignored (does not generate a new page)."
- IF323: The second line of FDML warnings are occasionally included in the listing file (though not in the code) as if they were in the actual source.
- IF351: nowarn only catches front end warnings and informationals. Suppress the middle end warnings and informationals as well.
- IF354: Floating-point test runs slower on Integrity than on Alpha.
- IF390: Compiler is "slow" and/or ends abnormally with an "insufficient virtual memory" error when a record containi

3.7 New and Changed Features in V8.0

HP Fortran V8.0 was the initial version of HP Fortran provided for OpenVMS for Integrity Servers.

4

Differences from HP Fortran for OpenVMS Alpha Systems

HP Fortran for OpenVMS Integrity server systems features the same command-line options and language features as HP Fortran for OpenVMS Alpha Systems with a few exceptions. A new appendix has been added to the HP Fortran for OpenVMS User Manual called "Differences Between HP Fortran on OpenVMS Integrity servers and OpenVMS Alpha Systems," which fully details those exceptions. This chapter highlights only the major differences.

4.1 Floating-Point Arithmetic

The white paper "OpenVMS floating-point arithmetic on the Intel Itanium architecture" is essential reading. This document is at:

http://h71000.www7.hp.com/openvms/integrity/i64-floating-pt-wp.pdf

Note: the URL is one continuous line (broken here to fit).

That white paper, together with these release notes, describes how HP Fortran for OpenVMS Integrity server systems handles floating-point issues. Here are some highlights to keep in mind:

- IEEE is the default floating-point datatype (that is, the default is /FLOAT=IEEE_FLOAT)

  HP Fortran for OpenVMS Alpha Systems defaults to the VAX G_float floating-point format (/FLOAT=G_FLOAT). On OpenVMS Integrity servers systems, however, there is no hardware support for floating-point representations other than IEEE. Instead, the compiler supports VAX floating-point formats by generating run-time code which converts operands to IEEE format, performs the needed arithmetic operations, and then converts the IEEE result back to the appropriate VAX format. Depending on the application, this may impose significant additional run-time overhead and some loss of accuracy compared to performing the operations in hardware on an Alpha (or VAX).

  This software support for the VAX formats is an important functional compatibility requirement for certain applications that need to deal with on-disk binary floating-point data, but its use should be strongly discouraged. This change is similar to the change in default from /FLOAT=D_FLOAT on VAX to /FLOAT=G_FLOAT on Alpha.

  If at all possible, users should use /FLOAT=IEEE_FLOAT (the default) for the highest performance and accuracy.

  Note that the changed /FLOAT default will have implications for the use of /CONVERT=NATIVE (the default). This switch causes unformatted data to remain unconverted on input, on the assumption that it matches the prevailing floating-point datatype.

Files written from Fortran applications built with /FLOAT=G_ FLOAT/CONVERT=NATIVE (the default) on Alpha can be read by Integrity server applications built with /FLOAT=G_FLOAT/CONVERT=NATIVE or /FLOAT=IEEE/CONVERT=VAXG.

- The /IEEE_MODE qualifier defaults to /IEEE_MODE=DENORM_ RESULTS This differs from Alpha, where the default is /IEEE_MODE=FAST. Despite the name, /IEEE_MODE=FAST does not have a significant effect on run-time performance on Integrity servers (or on Alpha processors from EV6 onward).
- On Integrity servers, users will have to pick one compile-time /FLOAT value and one compile-time /IEEE_ MODE value and stick with it for the whole of their application. This is because mixed-mode applications will not (in general) work on OpenVMS Integrity servers systems as a result of architectural differences in the hardware. This is a change from OpenVMS Alpha systems, where mixed-mode applications work. In particular, per-routine/per-file/per-library settings of a mode will not work.
- If your code changes the floating-point exception mode at runtime, it is your responsibility to change it back on routine-exit, including exits by means of exception handlers. Failure to do so may lead to unexpected results, as the compilers will assume that that no called routine changes the current mode. This is also a requirement on user-written libraries. The new appendix in the HP Fortran for OpenVMS User Manual explains in detail how to perform this exception-mode change. Please note that this requirement is also in effect for user-written libraries.
- Exception handlers will be entered with the floating- point mode in effect at the time the exception was raised, not the mode with which the handler was compiled.

4.2 No F77 Compiler

Only the F90 compiler is supported on OpenVMS Integrity servers. The F77 compiler, invoked on OpenVMS Alpha with the /OLD_F77 qualifier, is not available. To ease the transition from F77 to F90, CDD (Common Data Dictionary) and DML (Data Manipulation Language) support, which were previously available only in the Alpha F77 compiler, have now been implemented in the F90 compiler for OpenVMS Integrity servers.

Code previously compiled with the F77 compiler that does not compile with the F90 compiler should be reported through standard problem reporting mechanisms. Although it may not be possible to eliminate all differences between the two compiler versions, the HP Fortran development team will attempt to minimize these differences by enhancing the F90 compiler.

Note

The unsupported /OLD_F77 qualifier should not be confused with the /F77 qualifier, which instructs the compiler to use FORTRAN-77 interpretation rules for those statements that have a meaning incompatible with FORTRAN- 66; /F77 is still fully supported.

4.3 Dirty Zeros Trigger Exceptions

VAX-format "dirty zeros" (bit patterns which represent the value 0.0 but have some bits set in should-be-zero fields) may trigger exceptions where the equivalent Alpha program would not.

4.4 Math Algorithm Differences

Typically, the true function result for a math library function such as sin, atan, log, or exp is irrational, and cannot be represented exactly in any binary floating-point format (whether VAX or IEEE). In addition, because of architectural considerations, some of the algorithms used in the Integrity servers math library differ from the algorithms used on Alpha. As a result of these factors, various Integrity server mathematics builtins may produce rounded floating-point results that are very accurate, but slightly different from the equivalent Alpha results.

4.5 Math Library Exceptions

User code built with any of the three IEEE modes (FAST, UNDERFLOW, or DENORM_RESULTS) will work with the math library provided with the OpenVMS Integrity servers release. Exceptions will be raised by the math library as specified by the mode. For example, if you specify /IEEE_ MODE=DENORM_ RESULTS, a denormal or NaN result will not cause an exception; if you specify FAST, a NaN will cause an exception and a denormal result will be set to zero.

The math library runs using its own exception mode, not the mode currently in effect when a math library routine is called. The math library checks the operands and the mode set by the user. When a calculation would cause an exception for the user, the math library will raise a math library exception, as it does on Alpha.

Other system libraries provided with this release run in the exception mode in effect when the library routine is called.

4.6 /ARCH and /OPT=TUNE qualifiers

- HP Fortran for OpenVMS Integrity servers is tuned by default for Itanium 2 processors, which are included in all systems supported by OpenVMS Integrity servers Version 8.2. Therefore, the use of /ARCH and /OPT=TUNE qualifiers is not supported in this release.
- For the sake of "compile-and-go" compatibility, Alpha values for the /ARCH and /OPT=TUNE qualifiers are accepted on the compiler invocation command. An informational is printed saying they are ignored.

5
Known Problems and Incomplete Functionality

5.1 Bugs Not Fixed

- QXCM1000416852: VMS DST Specification and DST records are not in sync.
- 3431: INVREALLOC error due to allocation too large for the Fortran Runtime Library.
- IF431: Unexpected error during port: Conflicting attributes or multiple declaration of name stops compilation previously allowed.

5.2 Disabled /CHECK=ARG_INFO,FP_EXCEPTIONS,UNDERFLOW

The full frameworks needed to provide run-time checks of argument types and counting of IEEE exceptions are not yet available on OpenVMS Integrity server systems. As a result, the ARG_INFO, FP_EXCEPTIONS, and UNDERFLOW keywords have been disabled for the /CHECK qualifier. A warning to this effect will be issued during compilation if these keywords are specified on the command line. In addition, these checks are not currently enabled by specifying /CHECK=ALL.

5.3 IEEE Exceptions Detected During Compilation

If the compiler detects that a calculation will result in an exceptional IEEE value (such as a denormalized number), it may issue a compilation warning or error regardless of the IEEE exception mode chosen when compiling the program.

6

VSI Fortran Documentation and Online Information

The VSI Fortran documentation set includes the following:

- *HP Fortran Installation Guide for OpenVMS Integrity servers* (AA-PU3AF-TE)

  Explains how to install VSI Fortran on an OpenVMS I64 operating system, including requirements.

  The installation guide is included with the software kit. It is not available in printed format.

- *HP Fortran for OpenVMS Language Reference Manual* (AA-Q66SD-TK)

  This manual contains the complete description of the HP Fortran programming language, which includes Fortran 95 and Fortran 90 features. It contains information about language syntax and semantics, adherence to various Fortran standards, and extensions to those standards.

  This document is included with the HP Fortran documentation kit, BA368MN. It is also on the Online Documentation Library CD-ROM, and is on the HP Fortran web site.

- *HP Fortran for OpenVMS User Manual* (AA-QJRWD-TE)

  This manual provides information about HP Fortran program development and the run-time environment. It describes compiling, linking, running, and debugging HP Fortran programs, run-time error-handling and I/O, performance guidelines, data types, numeric data conversion, calling other procedures and library routines, and compatibility with Compaq Fortran 77.

  This document is included with the HP Fortran documentation kit, BA368MN. It is also on the Online Documentation Library CD-ROM, and is on the HP Fortran web site.

The VSI Fortran Software Product Description (SPD) is provided on the Software Product Library CD-ROM.

The following VSI Fortran online information is available (once installed on the system):

- VSI Fortran online release notes

  Provide more information on this version of VSI Fortran, including known problems.

  To display or print the VSI Fortran release notes before installing, use the PRODUCT EXTRACT RELEASE_NOTES FORTRAN command (see the *HP Fortran Installation Guide for OpenVMS Integrity servers*).

  Once installed, the ASCII version of the online release notes are located in:

  SYS$HELP:FORTRAN.RELEASE_NOTES

Other forms of the release notes (PostScript and Bookreader) are also provided, using the file name:

SYS$HELP:FORTRAN_RELEASE_NOTES.*

- VSI Fortran online help

  The VSI Fortran HELP module in SYS$HELP:HELPLIB.HLB provides online access to VSI Fortran help, including descriptions of command qualifiers, a summary of the language elements (statements, intrinsic procedures, and so on), error message descriptions, and other information.

  To view the online VSI Fortran help file using the HELP command, type:

  ```
  $ HELP FORTRAN
  ```

  You can specify topics to navigate the help hierarchy. For example:

  ```
  $ HELP FORTRAN /ALIGN
  ```