IBM XL Fortran for Linux, V16.1

Migration Guide for Little Endian Distributions

Version 16.1

GC27-8044-00

IBM XL Fortran for Linux, V16.1

Migration Guide for Little Endian Distributions

Version 16.1

GC27-8044-00

Note

Before using this information and the product it supports, read the information in "Notices" on page 11.

First edition

This edition applies to IBM XL Fortran for Linux, V16.1 (Program 5765-J10; 5725-C75) and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

Contents

About this document

This document contains migration considerations applicable to IBM® XL Fortran for Linux, V16.1.

Who should read this document

This document is intended for Fortran developers who are to use IBM XL Fortran for Linux, V16.1 to compile programs that were previously compiled on different platforms, by previous releases of XL Fortran, or by other compilers.

How to use this document

Throughout this document, the **xlf** compiler invocation is used to describe the behavior of the compiler. You can, however, substitute other forms of the compiler invocation command if your particular environment requires it, and compiler option usage remains the same unless otherwise specified.

While this document covers migration considerations applicable to IBM XL Fortran for Linux, V16.1, it does not include the following topics:

- Compiler installation: see the *XL Fortran Installation Guide*.
- Compiler options: see the *XL Fortran Compiler Reference* for detailed information about the syntax and usage of compiler options.
- The Fortran programming language: see the *XL Fortran Language Reference* for information about the syntax, semantics, and IBM implementation of the Fortran programming language.
- Programming topics: see the *XL Fortran Optimization and Programming Guide* for detailed information about developing applications with XL Fortran, with a focus on program portability and optimization.

Conventions

Typographical conventions

The following table shows the typographical conventions used in the IBM XL Fortran for Linux, V16.1 information.

Table 1. Typographical conventions

Typeface	Indicates	Example
lowercase bold	Invocation commands, executable names, and compiler options.	The compiler provides basic invocation commands, **xlf**, along with several other compiler invocation commands to support various Fortran language levels and compilation environments. The default file name for the executable program is **a.out**.

Table 1. Typographical conventions (continued)

Typeface	Indicates	Example
italics	Parameters or variables whose actual names or values are to be supplied by the user. Italics are also used to introduce new terms.	Make sure that you update the *size* parameter if you return more than the *size* requested.
underlining	The default setting of a parameter of a compiler option or directive.	nomaf | maf
`monospace`	Examples of program code, reference to program code, file names, path names, command strings, or user-defined names.	To compile and optimize myprogram.f, enter: `xlf myprogram.f -O3`.
UPPERCASE bold	Fortran programming keywords, statements, directives, and intrinsic procedures. Uppercase letters may also be used to indicate the minimum number of characters required to invoke a compiler option/suboption.	The **ASSERT** directive applies only to the **DO** loop immediately following the directive, and not to any nested **DO** loops.

Qualifying elements (icons and bracket separators)

In descriptions of language elements or programming models, this information uses icons and marked bracket separators to delineate segments of text as follows:

Table 2. Qualifying elements

Icon	Bracket separator text	Meaning
F2008 F2008	Fortran 2008 begins / Fortran 2008 ends	The text describes an IBM XL Fortran implementation of the Fortran 2008 standard.[1]
F2003 F2003	Fortran 2003 begins / Fortran 2003 ends	The text describes an IBM XL Fortran implementation of the Fortran 2003 standard, and it applies to all later standards.[1]
TS 29113 TS 29113	TS 29113 begins / TS 29113 ends	The text describes an IBM XL Fortran implementation of Technical Specification 29113, referred to as TS 29113.[1]
IBM IBM	IBM extension begins / IBM extension ends	The text describes a feature that is an IBM XL Fortran extension to the standard language specifications.
CUDA Fortran CUDA Fortran	CUDA Fortran begins / CUDA Fortran ends	The text describes CUDA Fortran, the CUDA Fortran support provided by IBM XL Fortran, or both.

Table 2. Qualifying elements (continued)

Icon	Bracket separator text	Meaning
GPU▶ ◀GPU	GPU begins / GPU ends	The text describes the information that is relevant to offloading computations to the NVIDIA GPUs.

Note:

1. If the information is marked with a Fortran language standard icon or bracket separators, it applies to this specific Fortran language standard and all later ones. Otherwise, it applies to all Fortran language standards.

Syntax diagrams

Throughout this information, diagrams illustrate XL Fortran syntax. This section helps you to interpret and use those diagrams.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

  The ▶▶─── symbol indicates the beginning of a command, directive, or statement.

  The ───▶ symbol indicates that the command, directive, or statement syntax is continued on the next line.

  The ▶─── symbol indicates that a command, directive, or statement is continued from the previous line.

  The ───▶◀ symbol indicates the end of a command, directive, or statement.

  Fragments, which are diagrams of syntactical units other than complete commands, directives, or statements, start with the |─── symbol and end with the ───| symbol.

  IBM XL Fortran extensions are marked by a number in the syntax diagram with an explanatory note immediately following the diagram.

  Program units, procedures, constructs, interface blocks and derived-type definitions consist of several individual statements. For such items, a box encloses the syntax representation, and individual syntax diagrams show the required order for the equivalent Fortran statements.
- Required items are shown on the horizontal line (the main path):

```
▶▶──keyword──required_argument──────────────────────────────────────▶◀
```

- Optional items are shown below the main path:

```
▶▶──keyword──┬───────────────────┬──────────────────────────────────▶◀
             └─optional_argument─┘
```

  Note: Optional items (not in syntax diagrams) are enclosed by square brackets ([and]). For example, [UNIT=]u
- If you can choose from two or more items, they are shown vertically, in a stack.

  If you *must* choose one of the items, one item of the stack is shown on the main path.

```
►►──keyword──┬─required_argument1─┬──────────────────────────────────────►◄
             └─required_argument2─┘
```

If choosing one of the items is optional, the entire stack is shown below the main path.

```
►►──keyword──┬────────────────────┬──────────────────────────────────────►◄
             ├─optional_argument1─┤
             └─optional_argument2─┘
```

- An arrow returning to the left above the main line (a repeat arrow) indicates that you can make more than one choice from the stacked items or repeat an item. The separator character, if it is other than a blank, is also indicated:

```
             ┌─,──────────────────┐
             ▼                    │
►►──keyword────repeatable_argument┴──────────────────────────────────────►◄
```

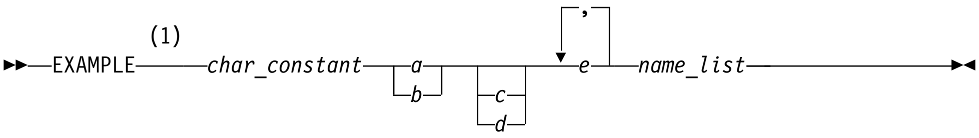
- The item that is the default is shown above the main path.

```
             ┌─default_argument───┐
►►──keyword──┴─alternate_argument─┴──────────────────────────────────────►◄
```

- Keywords are shown in nonitalic letters and should be entered exactly as shown.
- Variables are shown in italicized lowercase letters. They represent user-supplied names or values. If a variable or user-specified name ends in *_list*, you can provide a list of these terms separated by commas.
- If punctuation marks, parentheses, arithmetic operators, or other such symbols are shown, you must enter them as part of the syntax.

Sample syntax diagram

The following is an example of a syntax diagram with an interpretation:



Notes:

1 IBM extension

Interpret the diagram as follows:

- Enter the keyword EXAMPLE.
- EXAMPLE is an IBM extension.
- Enter a value for *char_constant*.
- Enter a value for *a* or *b*, but not for both.
- Optionally, enter a value for *c* or *d*.
- Enter at least one value for *e*. If you enter more than one value, you must put a comma between each.
- Enter the value of at least one *name* for *name_list*. If you enter more than one value, you must put a comma between each. (The *_list* syntax is equivalent to the previous syntax for *e*.)

How to read syntax statements

Syntax statements are read from left to right:

- Individual required arguments are shown with no special notation.
- When you must make a choice between a set of alternatives, they are enclosed by { and } symbols.
- Optional arguments are enclosed by [and] symbols.
- When you can select from a group of choices, they are separated by | characters.
- Arguments that you can repeat are followed by ellipses (...).

Example of a syntax statement

EXAMPLE *char_constant* {*a*|*b*}[*c*|*d*]*e*[,*e*]... *name_list*{*name_list*}...

The following list explains the syntax statement:

- Enter the keyword EXAMPLE.
- Enter a value for *char_constant*.
- Enter a value for *a* or *b*, but not for both.
- Optionally, enter a value for *c* or *d*.
- Enter at least one value for *e*. If you enter more than one value, you must put a comma between each.
- Optionally, enter the value of at least one *name* for *name_list*. If you enter more than one value, you must put a comma between each *name*.

Note: The same example is used in both the syntax-statement and syntax-diagram representations.

Examples in this information

The examples in this information, except where otherwise noted, are coded in a simple style that does not try to conserve storage, check for errors, achieve fast performance, or demonstrate all possible methods to achieve a specific result.

The examples for installation information are labelled as either *Example* or *Basic example*. *Basic examples* are intended to document a procedure as it would be performed during a default installation; these need little or no modification.

Notes on the terminology used

Some of the terminology in this information is shortened as follows:

- The term *free source form format* often appears as *free source form*.
- The term *fixed source form format* often appears as *fixed source form*.
- The term *XL Fortran* often appears as *XLF*.

Related information

The following sections provide related information for XL Fortran:

Available help information

IBM XL Fortran information

XL Fortran provides product information in the following formats:

- Quick Start Guide

  The Quick Start Guide (`quickstart.pdf`) is intended to get you started with IBM XL Fortran for Linux, V16.1. It is located by default in the XL Fortran directory and in the `\quickstart` directory of the installation DVD.
- README files

  README files contain late-breaking information, including changes and corrections to the product information. README files are located by default in the XL Fortran directory, and in the root directory and subdirectories of the installation DVD.
- Installable man pages

  Man pages are provided for the compiler invocations and all command-line utilities provided with the product. Instructions for installing and accessing the man pages are provided in the *IBM XL Fortran for Linux, V16.1 Installation Guide*.
- Online product documentation

  The fully searchable HTML-based documentation is viewable in IBM Knowledge Center at http://www.ibm.com/support/knowledgecenter/SSAT4T_16.1.0/com.ibm.compilers.linux.doc/welcome.html.
- PDF documents

  PDF documents are available on the web at https://www.ibm.com/support/knowledgecenter/SSAT4T_16.1.0/com.ibm.compilers.linux.doc/download_pdf.html.

  The following files comprise the full set of XL Fortran product information.

  Note: To ensure that you can access cross-reference links to other XL Fortran PDF documents, download and unzip the .zip file that contains all the product documentation files, or you can download each document into the same directory on your local machine.

Table 3. XL Fortran PDF files

Document title	PDF file name	Description
What's New for IBM XL Fortran for Linux, V16.1, GC27-8043-00	`whats_new.pdf`	Provides an executive overview of new functions in the IBM XL Fortran for Linux, V16.1 compiler, with new functions categorized according to user benefits.
Getting Started with IBM XL Fortran for Linux, V16.1, GI13-3566-00	`getstart.pdf`	Contains an introduction to XL Fortran, with information about setting up and configuring your environment, compiling and linking programs, and troubleshooting compilation errors.
IBM XL Fortran for Linux, V16.1 Installation Guide, GC27-8040-00	`install.pdf`	Contains information for installing XL Fortran and configuring your environment for basic compilation and program execution.
IBM XL Fortran for Linux, V16.1 Migration Guide, GC27-8044-00	`migrate.pdf`	Contains migration considerations for using XL Fortran to compile programs that were previously compiled on different platforms, by previous releases of XL Fortran, or by other compilers.
IBM XL Fortran for Linux, V16.1 Compiler Reference, SC27-8050-00	`compiler.pdf`	Contains information about the various compiler options and environment variables.
IBM XL Fortran for Linux, V16.1 Language Reference, SC27-8048-00	`langref.pdf`	Contains information about the Fortran programming language as supported by IBM, including language extensions for portability and conformance to nonproprietary standards, compiler directives and intrinsic procedures.
IBM XL Fortran for Linux, V16.1 Optimization and Programming Guide, SC27-8049-00	`proguide.pdf`	Contains information on advanced programming topics, such as application porting, interlanguage calls, floating-point operations, input/output, application optimization and parallelization, and the XL Fortran high-performance libraries.
Getting Started with CUDA Fortran programming using IBM XL Fortran for Linux, V16.1, GI13-3565-00	`getstart_cudaf.pdf`	Contains detailed information about the CUDA Fortran support that is provided in XL Fortran, including the compiler flow for CUDA Fortran programs, compilation commands, useful compiler options and macros, supported CUDA Fortran features, and limitations.

To read a PDF file, use Adobe Reader. If you do not have Adobe Reader, you can download it (subject to license terms) from the Adobe website at http://www.adobe.com.

More information related to XL Fortran, including IBM Redbooks® publications, white papers, and other articles, is available on the web at http://www.ibm.com/support/docview.wss?uid=swg27036672.

For more information about the compiler, see the XL compiler on Power® community at http://ibm.biz/xl-power-compilers.

Other IBM information

- *ESSL product documentation* available at http://www.ibm.com/support/knowledgecenter/SSFHY8/essl_welcome.html?lang=en

Standards and specifications

XL Fortran is designed to support the following standards and specifications. You can refer to these standards and specifications for precise definitions of some of the features found in this information.

- *American National Standard Programming Language FORTRAN, ANSI X3.9-1978*.
- *American National Standard Programming Language Fortran 90, ANSI X3.198-1992*.
- *ANSI/IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Std 754-1985*.
- *Federal (USA) Information Processing Standards Publication Fortran, FIPS PUB 69-1*.
- *Information technology - Programming languages - Fortran, ISO/IEC 1539-1:1991*. (This information uses its informal name, Fortran 90.)
- *Information technology - Programming languages - Fortran - Part 1: Base language, ISO/IEC 1539-1:1997*. (This information uses its informal name, Fortran 95.)
- *Information technology - Programming languages - Fortran - Part 1: Base language, ISO/IEC 1539-1:2004*. (This information uses its informal name, Fortran 2003.)
- *Information technology - Programming languages - Fortran - Part 1: Base language, ISO/IEC 1539-1:2010*. (This information uses its informal name, Fortran 2008. We currently provide partial support to this standard.)
- *Information technology - Further interoperability of Fortran with C, ISO/IEC TS 29113:2012*. (This information uses its informal name, Technical specification 29113, referred to as TS 29113. We currently provide partial support to this specification.)
- *Military Standard Fortran DOD Supplement to ANSI X3.9-1978, MIL-STD-1753* (United States of America, Department of Defense standard). Note that XL Fortran supports only those extensions documented in this standard that have also been subsequently incorporated into the Fortran 90 standard.
- *OpenMP Application Program Interface Version 3.1* (full support), *OpenMP Application Program Interface Version 4.0 (partial support)*, and *OpenMP Application Program Interface Version 4.5 (partial support)*, available at http://www.openmp.org

Other IBM information

- *ESSL product documentation* available at http://www.ibm.com/support/knowledgecenter/SSFHY8/essl_welcome.html?lang=en

Technical support

Additional technical support is available from the XL Fortran Support page at https://www.ibm.com/support/home/product/U128148Q26691I65/XL_Fortran_for_Linux. This page provides a portal with search capabilities to a large selection of Technotes and other support information.

If you cannot find what you need, you can send an email to compinfo@cn.ibm.com.

For the latest information about XL Fortran, visit the product information site at https://www.ibm.com/us-en/marketplace/xl-fortran-linux-compiler-power.

How to send your comments

Your feedback is important in helping us to provide accurate and high-quality information. If you have any comments about this information or any other XL Fortran information, send your comments to compinfo@cn.ibm.com.

Be sure to include the name of the manual, the part number of the manual, the version of XL Fortran, and, if applicable, the specific location of the text you are commenting on (for example, a page number or table number).

Chapter 1. Migrating from Linux for big endian distributions to Linux for little endian distributions

IBM XL Fortran for Linux, V15.1.1 or later is compatible with other versions of the compiler running on the POWER8® big endian systems. There are, however, some differences to consider.

Porting from big endian systems

- To help migrate programs from big endian systems, you can use the **-qaltivec=be** or **-qaltivec=le** option to toggle the vector element sequence in registers to big endian or little endian element order.

  For more information, see "Migrating program containing vector intrinsic procedures from big endian systems."
- To make big endian data files compatible in little endian systems, you can use the **-qufmt=be** option so that the I/O operations on unformatted data files use the big endian byte order.

Related information in the *XL Fortran Compiler Reference*

-qufmt

-qaltivec

Migrating program containing vector intrinsic procedures from big endian systems

When migrating the programs that contain the Vector Multimedia Extension (VMX) and Vector Scalar Extension (VSX) intrinsic procedures from big endian systems, you can use `-qaltivec=be` to minimize program changes, but you need to pay attention in specific cases.

The following table shows what users need to pay attention when migrating code from big endian systems by using `-qaltivec=be`.

Table 4. Attention when `-qaltivec=be`

Case	Attention
If the existing program contains only VMX load and store intrinsic procedures	Using `-qaltivec=be` may affect the program performance; using `-qaltivec=le` may affect the performance in different ways.
If the existing program contains only VSX load and store intrinsic procedures	In the existing programs, you can use the **VEC_XL** and **VEC_XST** procedures to replace the VSX load and store intrinsic procedures to maximally simplify the code changes.
If the existing program contains both VMX and VSX load and store intrinsic procedures	You need to pay attention to the differences of the element order of vectors that are operated by the VMX and VSX intrinsic procedures in little endian systems.
If the existing program contains the vector initialization by using the EQUIVALENCE statement with arrays	You need to use the **VEC_LD** or **VEC_XL** procedure to load the vectors explicitly, instead of using the EQUIVALENCE with arrays, or you can reverse the element order of the array used for vector initialization.

Related information in the *XL Fortran Compiler Reference*

-qaltivec

Related information in the *XL Fortran Language Reference*

Vector intrinsic procedures (IBM extension)

Chapter 2. Migrating from earlier versions to the latest version

When you migrate applications from earlier versions to the latest version, consider factors including changed compiler options, intrinsic procedures, and environment variables.

Changed compiler options

-D
Starting from IBM XL Fortran for Linux, V15.1.6, the **-D** option is updated to define a macro as in a `#define` preprocessor directive. In previous releases, **-D** was the short form of **-qdlines**.

For more information, see -D in the *XL Fortran Compiler Reference*.

-U
Starting from IBM XL Fortran for Linux, V15.1.6, the **-U** option is updated to undefine a preprocessor macro defined by the compiler or by the **-D** compiler option. In previous releases, **-U** was the short form of **-qmixed**.

For more information, see -U in the *XL Fortran Compiler Reference*.

Changed environment variables

XLSMPOPTS
Starting from IBM XL Fortran for Linux, V15.1.6, **XLSMPOPTS=target=optional** is renamed to **XLSMPOPTS=target=default** with the identical functionality, and **XLSMPOPTS=target=disable** is renamed to **XLSMPOPTS=target=disabled** with the identical functionality.

For more information, see XLSMPOPTS in the *XL Fortran Optimization and Programming Guide*.

Changed intrinsic procedures

`VEC_CNTLZ(ARG1)`
Starting from IBM XL Fortran for Linux, V15.1.5, the data types of the returned value are changed: now the compiler returns the same type as the argument, instead of always returning an unsigned type.

When you migrate programs from earlier versions to release versions starting from IBM XL Fortran for Linux, V15.1.5 for little endian distributions, this change might cause incompatibility. It is recommended that you change your code according to the new behavior.

For more information, see **VEC_CNTLZ(ARG1)** in the *XL Fortran Language Reference*.

Chapter 3. Resolving the compatibility issues of IPA object files

It is recommended that you use the latest version of the compiler to compile and link the IPA object files to avoid compatibility issues. If any compatibility issues occur, you can try these resolutions.

IPA object files that are compiled using earlier versions but are linked by a newer version

When IPA object files that are compiled with earlier versions of compilers are linked by a newer version, errors might occur if the IPA object is compiled by one of the following compilers.

- XL Fortran, V15.1.2 or earlier
- XL C/C++, V13.1.2 or earlier

Try resolving the compatibility issue using one of the following methods:

- Recompile and link your object files with the latest XL compiler if you want to use IPA.
- Do not enable the **-qipa** option.

IPA object files that are compiled using newer versions but are linked by an earlier version

If IPA object files that are compiled with newer versions of compilers are linked by an earlier version, errors occur during the link step. You might be able to resolve the issue by recompiling and linking the IPA object files with the latest XL compiler.

For more information, see Interprocedural analysis (IPA) in the *XL Fortran Optimization and Programming Guide*.

Chapter 4. Porting programs to XL Fortran

XL Fortran provides many features intended to make it easier to take programs that were originally written for other computer systems or compilers and recompile them with XL Fortran.

Outline of the porting process

To port a typical program, you need to identify the nonportable features, modify the source files to remove or work around these features, and compile the program with XL Fortran so you can compare the output with the output from the other system.

The process for porting a typical program looks like this:

1. Identify any nonportable language extensions or features that you used in the original program. Check to see if any of them are supported by XL Fortran:
   - Language extensions are identified in the *XL Fortran Language Reference*.
   - Some extensions require you to specify an XL Fortran compiler option; you can find these options listed in the *Portability and migration options* table in the *XL Fortran Compiler Reference*.
2. For any nonportable features that XL Fortran does not support, modify the source files to remove or work around them.
3. Do the same for any implementation-dependent features. For example, if your program relies on exact bit-pattern representation of floating-point values or uses system-specific file names, you may need to change it.
4. Compile the program with XL Fortran. If any compilation problems occur, fix them and recompile and fix any additional errors until the program compiles successfully.
5. Run the XL Fortran-compiled program and compare the output with the output from the other system. If the results are substantially different, there are probably still some implementation-specific features that need to be changed. If the results are only marginally different (for example, if XL Fortran produces a different number of digits of precision or a number differs in the last decimal place), decide whether the difference is significant enough to investigate further. You might be able to fix these differences.

Before porting programs to XL Fortran, read the tips in the following sections so that you know in advance what compatibility features XL Fortran offers.

Portability of directives

XL Fortran supports many directives available with other Fortran products. This ensures easy portability between products.

If your code contains *trigger_constants* other than the defaults in XL Fortran, you can use the **-qdirective** compiler option to specify them. For instance, if you are porting CRAY code contained in a file `xx.f`, you would use the following command to add the CRAY *trigger_constant*:

```
xlf95 xx.f -qdirective=mic\$
```

For fixed source form code, in addition to the **!** value for the *trigger_head* portion of the directive, XL Fortran also supports the *trigger_head* values **C**, **c**, and *.

For more information, see the **-qdirective** option in the *XL Fortran Compiler Reference*.

XL Fortran supports a number of programming terms as synonyms to ease the effort of porting code from other Fortran products. Those terms that are supported are dependent on context, as indicated in the following tables:

Table 5. PARALLEL DO Clauses and their XL Fortran synonyms

PARALLEL DO Clause	**XL Fortran Synonym**
LASTLOCAL	LASTPRIVATE
LOCAL	PRIVATE
MP_SCHEDTYPE *and* CHUNK	SCHEDULE
SAVELAST	LASTPRIVATE
SHARE	SHARED
NEW	PRIVATE

Table 6. PARALLEL DO scheduling types and their XL Fortran synonyms

Scheduling Type	**XL Fortran Synonym**
GSS	GUIDED
INTERLEAVE	STATIC(1)
INTERLEAVED	STATIC(1)
INTERLEAVE(n)	STATIC(n)
INTERLEAVED(n)	STATIC(n)
SIMPLE	STATIC

Table 7. PARALLEL SECTIONS clauses and their XL Fortran synonyms

PARALLEL SECTIONS Clause	**XL Fortran Synonym**
LOCAL	PRIVATE
SHARE	SHARED
NEW	PRIVATE

Common industry extensions that XL Fortran supports

XL Fortran allows many of the same FORTRAN 77 extensions as other popular compilers.

These extensions include:

Extension	**Refer to** *XL Fortran Language Reference* **Section(s)**
Typeless constants	Typeless literal constants

Extension	**Refer to** *XL Fortran Language Reference* **Section(s)**
**len* length specifiers for types	Data types
BYTE data type	Byte
Long variable names	Names
Lower case	Names
Mixing integers and logicals (with **-qintlog** option)	Evaluation of expressions
Character-count **Q** edit descriptor (with **-qqcount** option)	Q (Character Count) Editing
Intrinsics for counting set bits in registers and determining data-object parity	POPCNT, POPPAR
64-bit data types (**INTEGER(8)**, **REAL(8)**, **COMPLEX(8)**, and **LOGICAL(8)**), including support for default 64-bit types (with **-qintsize** and **-qrealsize** options)	Integer Real Complex Logical
Integer **POINTER**s, similar to those supported by CRAY and Sun compilers. (XL Fortran integer pointer arithmetic uses increments of one byte, while the increment on CRAY computers is eight bytes. You may need to multiply pointer increments and decrements by eight to make programs ported from CRAY computers work properly.)	POINTER(integer)
Conditional vector merge (CVMGx) intrinsic functions	CVMGx (TSOURCE, FSOURCE, MASK)
Date and time service and utility functions (rtc, irtc, jdate, clock_, timef, and date)	Service and utility procedures
STRUCTURE, **UNION**, and **MAP** constructs	Structure components, Union and map

Finding nonstandard extensions

XL Fortran supports a number of extensions to various language standards. Many of these extensions are so common that you need to keep in mind, when you port programs to other systems, that not all compilers have them. To find such extensions in your XL Fortran programs before beginning a porting effort, use the **-qlanglvl** option:

```
$ # -qnoobject stops the compiler after parsing all the source,
$ # giving a fast way to check for errors.
$ # Look for anything above the base F77 standard.
$ xlf -qnoobject -qlanglvl=77std f77prog.f
  ...
$ # Look for anything above the F90 standard.
$ xlf90 -qnoobject -qlanglvl=90std use_in_2000.f
  ...
$ # Look for anything above the F95 standard.
$ xlf95 -qnoobject -qlanglvl=95std use_in_2000.f
  ...
```

Relation reference in the *XL Fortran Language Reference*

-qlanglvl

-qport

Mixing data types in statements

The **-qctyplss** option lets you use character constant expressions in the same places that you use typeless constants. The **-qintlog** option lets you use integer expressions where you can use logicals, and vice versa. A kind type parameter must not be replaced with a logical constant even if **-qintlog** is on, nor by a character constant even if **-qctyplss** is on, nor can it be a typeless constant.

Date and time routines

Date and time routines, such as **dtime**, **etime**, and **jdate**, are accessible as Fortran subroutines.

Other libc routines

A number of other popular routines from the **libc** library, such as **flush**, **getenv**, and **system**, are also accessible as Fortran subroutines.

Changing the default sizes of data types

For porting from machines with larger or smaller word sizes, the **-qintsize** option lets you specify the default size for integers and logicals.The **-qrealsize** option lets you specify the default size for reals and complex components.

Name conflicts between your procedures and XL Fortran intrinsic procedures

If you have procedures with the same names as any XL Fortran intrinsic procedures, the program calls the intrinsic procedure. This situation is more likely with the addition of the many new Fortran 90, Fortran 95, Fortran 2003, and Fortran 2008 intrinsic procedures.

If you still want to call your procedure, add explicit interfaces, **EXTERNAL** statements, or **PROCEDURE** statements for any procedures with conflicting names, or use the **-qextern** option when compiling.

Reproducing results from other systems

XL Fortran provides settings through the **-qfloat** option that help make floating-point results consistent with those from other IEEE systems; this subject is discussed in Duplicating the floating-point results of other systems in the *XL Fortran Optimization and Programming Guide*.

Notices

Programming interfaces: Intended programming interfaces allow the customer to write programs to obtain the services of IBM XL Fortran for Linux.

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who want to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided “AS IS”, without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 1998, 2018.

This software and documentation are based in part on the Fourth Berkeley Software Distribution under license from the Regents of the University of California. We acknowledge the following institution for its role in this product's development: the Electrical Engineering and Computer Sciences Department at the Berkeley campus.

PRIVACY POLICY CONSIDERATIONS:

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, or to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details in the section entitled “Cookies, Web Beacons and Other Technologies,” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at http://www.ibm.com/software/info/product-privacy.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml.

Adobe and the Adobe logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

NVIDIA and CUDA are either registered trademarks or trademarks of NVIDIA Corporation in the United States, other countries, or both.

Index

Product Number: 5765-J10; 5725-C75

Printed in USA