IBM XL C/C++ for Linux V13.1

XL C/C++ 入门

V 13.1

S151-2064-00

IBM XL C/C++ for Linux V13.1

XL C/C++ 入门

V 13.1

S151-2064-00

注意

在使用本资料及其支持的产品之前，请参阅第 53 页的『声明』中的信息。

第一版本

本版本适用于 IBM XL C/C++ for Linux V13.1（程序 5765-J08; 5725-C73）及所有后续发行版和修订版，直到在新版本中另有声明为止。确保您使用的是本产品级别的正确版本。

本版本适用于 PRODNAME_VL（程序程序 5765-J08; 5725-C73）及所有后续发行版和修订版，直到在新版本中另有声明为止。确保您使用的是本产品级别的正确版本。

© Copyright IBM Corporation 1996, 2014.

目录

关于本文档 v
约定 v
相关信息 viii
IBM XL C/C++ 信息 viii
标准和规范 x
其他 IBM 信息 xi
其他信息 xii
技术支持 xii
如何发送您的意见 xii

第 1 章 XL C/C++ 简介 1
与其他 IBM 编译器的通用性 1
操作系统支持 1
高度可配置的编译器 1
语言标准一致性 2
与 GNU 的兼容性 3
源代码迁移和一致性检查 3
库 4
工具，实用程序和命令 5
程序优化 6
64 位对象能力 6
共享内存并行化 7
诊断列表 8
符号调试器支持 8

第 2 章 IBM XL C/C++ for Linux V13.1 的新增内容 9
支持 POWER8 处理器 9
Advance Toolchain 7.0 支持 10
C++11 功能 10
C11 功能 11
OpenMP 4.0 12
内置函数 12
编译器选项和编译指示伪指令 16
性能和优化 18

第 3 章 从先前版本迁移 19
V12.1 中新增的增强功能 19
C++11 功能 19
C11 功能 21
OpenMP 3.1 22
性能和优化 23
诊断报告 23
内置函数 24
编译器选项和编译指示伪指令 25
V11.1 中新增的增强功能 27
支持 POWER7 处理器 27
C++11 功能 28
性能和优化 31
新的诊断报告 32
使用情况跟踪和报告工具 34
新增或已更改的编译器选项和伪指令 34
内置函数 37
V10.1 中新增的增强功能 39
C++11 功能 39
与其他 XL C/C++ 语言相关的更新 40
OpenMP 3.0 41
性能和优化 41
编译器选项和编译指示伪指令 42
预定义宏 43

第 4 章 设置和定制 XL C/C++ 45
使用定制编译器配置文件 45
配置编译器使用情况跟踪和报告 45

第 5 章 使用 XL C/C++ 开发应用程序 47
编译器阶段 47
编辑 C/C++ 源文件 47
使用 XL C/C++ 进行编译 47
调用编译器 48
编译并行化 XL C/C++ 应用程序 48
指定编译器选项 49
XL C/C++ 输入和输出文件 49
将已编译的应用程序与 XL C/C++ 链接在一起 . . . 50
动态和静态链接 51
运行已编译的应用程序 51
XL C/C++ 编译器诊断辅助 52
调试已编译的应用程序 52
确定使用哪一个级别的 XL C/C++ 52

声明 53
商标和服务标记 55

索引 57

© Copyright IBM Corp. 1996, 2014

关于本文档

本文档包含 IBM® XL C/C++ for Linux V13.1 编译器的概述和基本用法信息。

应该阅读本文档的人

本文档是为那些想要了解 XL C/C++ 的介绍性概述和用法信息的 C 和 C++ 开发者编写的。它假设您已经较为熟悉命令行编译器，掌握基本的 C 和 C++ 编程语言知识和基本的操作系统命令知识。对 XL C/C++ 不熟悉的程序员可以使用本文档来查找有关 XL C/C++ 独有的功能和功能部件的信息。

如何使用本文档

除非另有声明，否则此参考中的所有文本均适合 C 和 C++ 语言。在这些语言之间有差别时，将会通过符合约定标准的文本和图标来指示这些差别，如『约定』中所述。

本文档中通篇使用 **xlc** 和 **xlc++** 编译器调用来描述编译器的操作。但是，如果特定环境需要，您也可以采用其他形式的编译器调用命令来代替，除非另有声明，否则编译器选项用法将保持不变。

虽然本文档涵盖有关配置编译器环境以及使用 XL C/C++ 编译器来编译和链接 C 或 C++ 应用程序的信息，但它不包含下列主题:

- 编译器安装: 请参阅 *XL C/C++ Installation Guide*，以获得有关安装 XL C/C++ 的信息。
- 编译器选项: 请参阅 *XL C/C++ Compiler Reference*，以获得有关编译器选项的语法和用法的详细信息。
- C 或 C++ 编程语言: 请参阅 *XL C/C++ Language Reference* 以获取有关 C 或 C++ 编程语言的语法、语义和 IBM 实现的信息。
- 编程主题: 请参阅 *XL C/C++ Optimization and Programming Guide*，以获得有关使用 XL C/C++ 开发应用程序的详细信息，它们着重于程序的可移植性和优化。

约定

印刷约定

下表说明了 中使用的印刷约定。IBM XL C/C++ for Linux V13.1 信息。

表 1. 印刷约定

字型	指示	示例
粗体	小写命令、可执行文件名称、编译器选项和伪指令。	编译器提供了基本的调用命令 **xlc** 和 **xlC** (**xlc++**) 以及若干其他编译器调用命令，以支持各种 C/C++ 语言级别和编译环境。
斜体字	由用户提供实际名称或值的参数或变量。斜体字还用于介绍新术语。	如果返回的值超过请求的 *size*，请确保更新 *size* 参数。

© Copyright IBM Corp. 1996, 2014

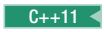
表 1. 印刷约定 (续)

字型	指示	示例
下划线	编译器选项或伪指令的参数的缺省设置。	nomaf | maf
等宽字体	编程关键字和库函数、编译器内置函数、程序代码示例、命令字符串或用户定义的名称。	要编辑 并优化 myprogram.c，请输入: `xlc myprogram.c -O3`。

限定元素（图标）

本信息中描述的大多数功能部件都同时适用于 C 和 C++ 语言。在语言元素描述中，如果某个功能部件是一种语言所独有的或者功能在不同语言之间存在差异，那么此信息将使用图标来描绘文本段， 如下所示:

表 2. 限定元素

限定符/图标	含义
仅限于 C，或仅限于 C 开始   仅限于 C 结束	该文本描述仅在 C 语言中才受支持的功能部件或描述特定于 C 语言的行为。
仅限于 C++，或仅限于 C++ 开始   仅限于 C++ 结束	该文本描述仅在 C++ 语言中才受支持的功能部件或描述特定于 C++ 语言的行为。
IBM 扩展或 IBM 扩展开始   IBM 扩展结束	该文本描述一个功能部件，该功能部件是标准语言规范的 IBM 扩展。
C11 或 C11 开始   C11 结束	该文本描述一个功能部件，该功能部件是作为 C11 的一部分引入至标准 C 的。
C++11 或 C++11 开始   C++11 结束	该文本描述一个功能部件，该功能部件是作为 C++11 的一部分引入至标准 C++ 的。

语法图

在本信息中，自始至终都用图来举例说明 XL C/C++ 语法。本节将帮助您解释和使用这些图。

- 沿着线条的走向，从左至右、从上至下阅读语法图。

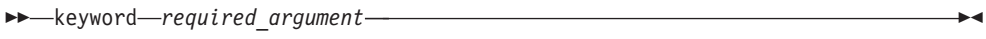
  ▸▸── 符号指示命令、伪指令或语句的开始。

  ──▸ 符号指示命令、伪指令或语句语法在下一行继续。

  ▸── 符号指示命令、伪指令或语句续上一行。

  ──▸◂ 符号指示命令、伪指令或语句的结束。

  这些片段是语法单元图，不同于完整的命令、伪指令或语句，它们以 |── 符号开始，以 ──| 符号结束。
- 必需的项显示在水平线（主路径）上:

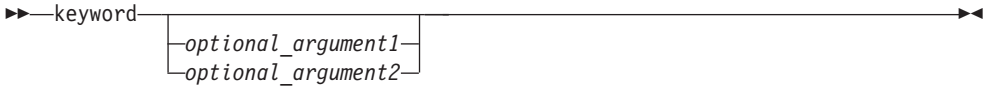


- 可选的项显示在主路径之下:

- 如果可以从两个或更多项中进行选择，那么将它们垂直堆叠在一起。

  如果*必须*选择这些项的其中一项，那么堆叠中有一项显示在主路径上。

  如果这些项是可选的，那么所有项都会显示在主路径下。



- 主线上向左折返的箭头（重复箭头）指示您可以从堆叠的项中选择多项或重复单个项。如果分隔符为非空白字符，那么还会指出该分隔符:

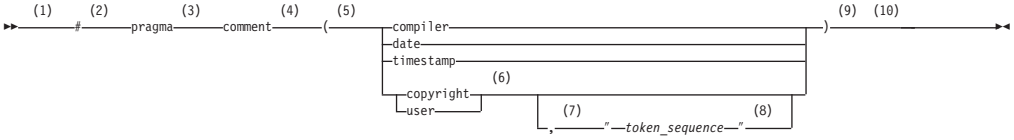
- 缺省项显示在主路径之上。

- 关键字以非斜体字母的形式显示，应严格按所显示的进行输入。

- 变量以斜体小写字母的形式显示。它们表示用户提供的名称或值。
- 如果显示了标点符号、圆括号、算术运算符或其他这样的符号，那么必须将它们作为语法的一部分输入。

样本语法图

以下语法图示例显示了 **#pragma comment** 伪指令的语法。



注:

1 这是语法图的开始。

2 符号 # 必须首先出现。

3 关键字 pragma 必须跟在 # 符号之后。

4 编译指示的名称 comment 必须跟在关键字 pragma 之后。

5 必须提供左圆括号。

6 输入的注释类型必须仅为以下所指示的类型之一: compiler、date、timestamp、copyright 或 user。

7 在注释类型 copyright 或 user 与可选字符串之间必须有一个逗号。

8 字符串必须跟在逗号之后。必须用双引号将字符串括起。

9 需要右圆括号。

10 这是语法图的结束。

按照以上显示的图表，**#pragma comment** 伪指令的下列示例在语法上是正确的:

```
#pragma comment(date)
   #pragma comment(user)
   #pragma comment(copyright,"This text will appear in the module")
```

本信息中的示例

除非另有说明，否则本信息中的示例都是用简单样式进行编码，此样式不会尝试保存存储器、检查错误、获取快速性能或演示获取特定结果的所有可能方法。

安装信息的示例已标注为*示例*或*基本示例*。*基本示例*旨在描述基本或缺省安装期间执行的过程，它们只需要进行极小的修改或甚至不需要修改。

相关信息

下列各节提供了 XL C/C++ 的相关信息:

IBM XL C/C++ 信息

XL C/C++ 以下列格式提供产品信息:

- 自述文件

自述文件包含最新的信息，其中包括对产品信息的更改和纠错。缺省情况下，自述文件文件位于 XL C/C++ 目录和 CD安装的根目录。

- 可安装的联机帮助页

  为编译器调用和随产品提供的所有命令行实用程序提供了联机帮助页。*《IBM XL C/C++ for Linux V13.1安装指南》*中提供了安装和访问联机帮助页的指示信息。

- 信息中心

  可在以下 Web 查看基于 HTML 的完全可搜索文档: http://www.ibm.com/support/knowledgecenter/SSXVZZ_13.1.0/com.ibm.compilers.linux.doc/welcome.html。

- PDF 文档

  缺省情况下，PDF 文档位于 /opt/ibm/xlC/13.1.0/doc/*LANG*/pdf/ 目录，其中 *LANG* 是以下其中一项: en_US、zh_CN 或 ja_JP。另外，Web 上也提供了 PDF 文件，网址为 http://www.ibm.com/support/docview.wss?uid=swg27036675。

  下列文件组成了一套完整的 XL C/C++ 产品信息:

表 3. XL C/C++ PDF 文件

文档标题	PDF 文件名	描述
《IBM XL C/C++ for Linux V13.1安装指南》, S151-2069-00	install.pdf	包含有关安装 XL C/C++ 和配置环境以执行基本的编译和程序的信息。
《IBM XL C/C++ for Linux V13.1 入门》, S151-2064-00	getstart.pdf	包含 XL C/C++ 产品的简介，提供了有关设置和配置环境、编译和链接程序以及对编译错误进行故障诊断的信息。
IBM XL C/C++ for Linux V13.1 Compiler Reference, SC27-4249-00	compiler.pdf	包含有关各种编译器选项、编译指示、宏、环境变量和内置函数（包括那些用于并行处理的函数）的信息。
IBM XL C/C++ for Linux V13.1 Language Reference, SC27-4250-00	langref.pdf	包含有关 IBM 支持的 C 和 C++ 编程语言（包括为了获取对非专有标准的可移植性和一致性的语言扩展）的信息。
IBM XL C/C++ for Linux V13.1 Optimization and Programming Guide, SC27-4251-00	proguide.pdf	包含有关下列高级编程主题的信息，例如应用程序移植、使用 Fortran 代码的语言间调用、库开发、应用程序优化和并行化以及 XL C/C++ 高性能库。

要阅读 PDF 文件，请使用 Adobe Reader。如果没有安装 Adobe Reader，那么可以依据许可条款从 Adobe Web 站点下载该软件，网址为 http://www.adobe.com。

可以在 Web 上获得与 XL C/C++ 相关的更多信息（包括 IBM Redbooks® 出版物、白皮书、教程、文档勘误表和其他文章），网址为:

http://www.ibm.com/support/docview.wss?uid=swg27036675

注: 文档勘误表仅在英文版的信息中心中显示。

有关提高性能、效率和可移植性的更多信息，请参阅 C/C++ café，网址为 http://www.ibm.com/software/rational/cafe/community/ccpp。

标准和规范

XL C/C++ 旨在支持下列标准和规范。您可以查阅这些标准以获取此信息中出现的某些功能部件的精确定义。

- *Information Technology - Programming languages - C, ISO/IEC 9899:1990*，也称为 *C89*。
- *Information Technology - Programming languages - C, ISO/IEC 9899:1999*，也称为 *C99*。
- *Information Technology - Programming languages - C, ISO/IEC 9899:2011*，也称为 *C11*。（部分支持）
- *Information Technology - Programming languages - C++, ISO/IEC 14882:1998*，也称为 *C++98*。
- *Information Technology - Programming languages - C++, ISO/IEC 14882:2003*，也称为*标准 C++*。
- *Information Technology - Programming languages - C++, ISO/IEC 14882:2011*，也称为 *C++11*。（部分支持）
- *Information Technology - Programming languages - Extensions for the programming language C to support new character data types, ISO/IEC DTR 19769*. C++ 标准委员会已接受此草稿技术报告，可以在 http://www.open-std.org/JTC1/SC22/WG14/www/docs/n1040.pdf 中找到此报告
- *Draft Technical Report on C++ Library Extensions, ISO/IEC DTR 19768*. 此草稿技术报告已提交给 C++ 标准委员，可以在 http://www.open-std.org/JTC1/SC22/WG21/docs/papers/2005/n1836.pdf 中找到此报告。
- *AltiVec Technology Programming Interface Manual*, Motorola Inc。此针对向量数据类型的规范用于支持向量处理技术，可以在 http://www.freescale.com/files/32bit/doc/ref_manual/ALTIVECPIM.pdf 中找到此规范。
- *Information Technology - Programming Languages - Extension for the programming language C to support decimal floating-point arithmetic, ISO/IEC WDTR 24732*. 此草稿技术报告已提交给 C++ 标准委员会，可以在 http://www.open-std.org/JTC1/SC22/WG14/www/docs/n1176.pdf 中找到此报告。
- *Decimal Types for C++: Draft 4* http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2006/n1977.html
- *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*.（本信息使用其非正式名称 Fortran 90。）
- *Information technology - Programming languages - Fortran - Part 1: Base language, ISO/IEC 1539-1:1997*.（此信息使用其非正式名称 Fortran 95。）
- *Information technology - Programming languages - Fortran - Part 1: Base language, ISO/IEC 1539-1:2004*.（此信息使用其非正式名称 Fortran 2003。）
- *Information technology - Programming languages - Fortran - Part 1: Base language, ISO/IEC 1539-1:2010*.（此信息使用其非正式名称 Fortran 2008）。我们当前提供了对此标准的部分支持。）

- *Military Standard Fortran DOD Supplement to ANSI X3.9-1978, MIL-STD-1753*（美国国防部标准）。注意，XL Fortran 仅支持此标准中已记录并且还在随后已合并到 Fortran 90 标准中的那些扩展。
- 在 http://www.openmp.org 处提供的 *OpenMP Application Program Interface Version 4.0 (Partial support)*

其他 IBM 信息

- *Parallel Environment for AIX: Operation and Use*
- *Blue Gene/Q Application Development, SG24-7948*，在以下地址提供：http://w3.itso.ibm.com/abstracts/sg247948.html
- *Blue Gene/Q System Administration, SG24-7869*，在以下地址提供：http://w3.itso.ibm.com/abstracts/sg247869.html
- *Blue Gene/Q Safety Considerations, REDP-4656*，在以下地址提供：http://w3.itso.ibm.com/abstracts/redp4656.html
- *Blue Gene/Q Hardware Overview, SG24-7872*，在以下地址提供：http://w3.itso.ibm.com/abstracts/sg247872.html
- *Blue Gene/Q High Availability Service Node, REDP-4657*，在以下地址提供：http://w3.itso.ibm.com/abstracts/redp4657.html
- *Blue Gene/Q Code Development and Tools Interface, REDP-4659*，在以下地址提供：http://w3.itso.ibm.com/abstracts/redp4659.html
- *Blue Gene/Q Install and Maintenance, SG24-7974*，在以下地址提供：http://w3.itso.ibm.com/abstracts/sg247974.html
- 在 http://publib.boulder.ibm.com/infocenter/systems/topic/eiccg/eiccgkickoff.htm 处提供的 *IBM C/C++ Language Extensions for Cell Broadband Engine Architecture*
- http://www.ibm.com/chips/techlib/techlib.nsf/products/Cell_Broadband_Engine 处提供了 Cell Broadband Engine 体系结构的规范、白皮书和其他技术信息。
- Cell Broadband Engine 资源中心位于 http://www.ibm.com/developerworks/power/cell/，它是包括文章、教程、编程指南和培训资源在内的技术信息的中央存储库。
- 您可以在 IBM Systems Information Center（位于 http://publib.boulder.ibm.com/infocenter/systems/index.jsp）中找到有关 IBM 系统的技术信息，包括 Cell Broadband Engine 解决方案。
- IBM Systems Information Center（位于 http://publib.boulder.ibm.com/infocenter/systems/index.jsp?topic=/com.ibm.aix.doc/doc/base/aixparent.htm）是 AIX® 信息的资源。

  您可以找到特定 AIX 系统的下列书籍：

  – *AIX Commands Reference, Volumes 1 - 6*
  – *Technical Reference: Base Operating System and Extensions, Volumes 1 & 2*
  – *AIX National Language Support Guide and Reference*
  – *AIX General Programming Concepts: Writing and Debugging Programs*
  – *AIX Assembler Language Reference*
- *ESSL for AIX V5.1/ESSL for Linux on POWER V5.1 Guide and Reference*，可以在 Engineering and Scientific Subroutine Library (ESSL) and Parallel ESSL Web 页面中找到此信息。

其他信息

- *Using the GNU Compiler Collection*，网址为 http://gcc.gnu.org/onlinedocs

技术支持

可以从 XL C/C++ 支持页面获得其他技术支持，网址为: http://www.ibm.com/support/entry/portal/overview/software/rational/xl_c_for_aixhttp://www.ibm.com/support/entry/portal/overview/software/rational/xl_c~c++_for_linuxhttp://www.ibm.com/support/entry/portal/overview/software/rational/xl_fortran_for_linuxhttp://www.ibm.com/software/awdtools/czos/support/。此页面提供一个具有搜索能力的门户网站，可以在该网站找到大量的技术说明和其他支持信息。

如果找不到所需要的内容，那么可以发送电子邮件至 compinfo@ca.ibm.com。

有关 XL C/C++ 的最新信息，请访问产品信息站点，网址为: http://www.ibm.com/software/awdtools/xlc/aix/http://www.ibm.com/software/products/us/en/xlcpp-linux/http://www.ibm.com/software/products/us/en/xlfortran-linuxhttp://www.ibm.com/software/awdtools/czos/。

如何发送您的意见

您的反馈对于帮助提供准确和高质量的信息非常重要。如果您对本资料或任何其他 XL C/C++ 资料有任何意见，请将您的意见通过电子邮件发送至 compinfo@ca.ibm.com。

请确保包含以下信息: 资料的名称、资料的章节号、XL C/C++ 的版本以及您的意见文本的特定位置，例如页号或表号。

第 1 章 XL C/C++ 简介

IBM XL C/C++ for Linux V13.1是一种高性能的高级编译器，可以用于开发复杂且要进行大量计算的 程序，包括使用 C和 Fortran 程序进行的语言间调用。

本节包含有关较高级别的 XL C/C++ 编译器的功能部件的信息。它面向要对该编译器进行评测的人员以及需了解关于该产品的更多信息的新用户。

与其他 IBM 编译器的通用性

IBM XL C/C++ for Linux V13.1是较大的 IBM C、C++ 和 Fortran 编译器系列的一部分。XL C/C++ 与 XL Fortran 一起组成 XL 编译器系列。

这些编译器是从共享用于大量平台和编程语言的编译器功能和优化技术的公共代码库派生的。编程环境包括 IBM AIX、IBM Blue Gene®/P、IBM Blue Gene®/Q、IBM i、选定的 Linux 分发版、IBM z/OS® 和 IBM z/VM®。公共代码库除了符合国际编程语言标准之外，还有助于在多个操作系统和硬件平台上保持编译器性能的一致且使程序更加容易移植。

操作系统支持

本节描述 IBM XL C/C++ for Linux V13.1支持的操作系统。

IBM XL C/C++ for Linux V13.1支持下列操作系统:

- SUSE Linux Enterprise Server 11 Service Pack 2 (SLES 11 SP2) 或更高版本
- Red Hat Enterprise Linux 6.4 (RHEL 6.4) 或更高版本
- Red Hat Enterprise Linux 7.0 (RHEL 7.0) 或更高版本

有关完整的需求列表，请参阅自述文件和 *XL C/C++ Installation Guide* 中的 Before installing XL C/C++。

IBM XL C/C++ for Linux V13.1 在大尾数法系统上受支持。编译器、编译器的库及编译器生成的对象程序将在具有所需软件和磁盘空间的 POWER5、POWER5+、POWER6®、POWER7®、POWER7+™和 POWER8™ 系统上运行。

为了利用各种受支持的硬件配置，编译器提供了选项以根据运行已编译的应用程序的硬件类型对应用程序的性能进行调整。

高度可配置的编译器

您可以使用各种编译器调用命令和选项来定制编译器以满足您的独特编译需求。

编译器调用命令

XL C/C++ 提供了几个可以用来调用编译器的不同命令，例如，**xlC**, **xlc++** 和 **xlc**。每个调用命令都是唯一的，它们指示编译器定制编译输出以遵循特定语言级别规范。提供编译器调用命令是为了支持所有标准化的 C/C++ 语言级别和许多流行的语言扩展。

© Copyright IBM Corp. 1996, 2014

编译器还提供了大多数调用命令的相应“**_r**”版本，例如 **xlc_r** 和 **xlC_r**。“**_r**”调用指示编译器将对象文件链接和绑定至线程安全组件与库，并为编译器创建的数据和过程生成线程安全对象代码。

有关 XL C/C++ 编译器调用命令的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Invoking the compiler。

编译器选项

您可以从大量编译器选项中选择某个选项来控制编译器行为。您可从对以下任务使用不同选项获得优势:

- 调试您的应用程序
- 优化和调整应用程序性能
- 选择语言级别和扩展以与其他 C 或 C++ 编译器支持的非标准功能部件和行为相兼容
- 执行将否则需要更改源代码的许多其他常见任务

您可以通过环境变量、编译器配置文件、命令行选项和程序源文件中嵌入的编译器伪指令语句的组合指定编译器选项。

有关 XL C/C++ 编译器选项的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler options reference。

定制编译器配置文件

安装过程会创建缺省的编译器配置文件，该文件包含定义了编译器选项缺省设置的节。

如果您经常指定 XL C/C++ 的缺省设置以外的编译器选项设置，那么可使用 makefile 来定义设置。或者，您也可以创建定制配置文件来定义您自己经常使用的选项设置。

有关使用定制编译器配置文件的更多信息，请参阅第 45 页的『使用定制编译器配置文件』。

使用情况跟踪配置文件

使用情况和报告工具可以用于检测组织对编译器的使用是否超过了许可证权利。

编译器的使用情况跟踪和报告功能具有它自己的配置文件。主要的编译器配置文件包含指向此文件的条目。编译器产品的不同安装可以使用单个使用情况跟踪配置文件来集中管理使用情况跟踪和报告功能。

有关使用情况跟踪和报告功能的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 Tracking and reporting compiler usage。

语言标准一致性

IBM XL C/C++ for Linux V13.1 支持以下 C/C++ 编程语言规范。

C 语言规范

- 部分支持 ISO/IEC 9899:2011（称为 C11）
- ISO/IEC 9899:1999（称为 C99）

- ISO/IEC 9899:1990（称为 C89）

C++ 语言规范

- 部分支持 ISO/IEC 14882:2011（称为 C++11）
- ISO/IEC 14882:2003（称为标准 C++）
- ISO/IEC 14882:1998，C++ 语言的第一个正式规范（称为 C++98）

除了标准语言级别之外，XL C/C++ 还支持以下语言扩展:

- 部分支持 OpenMP Application Program Interface V4.0
- OpenMP Application Program Interface V3.1
- 用于支持向量编程的语言扩展
- GNU C 和 C++ 语言扩展的子集

有关 C/C++ 语言规范和扩展的更多信息，请参阅 *XL C/C++ Language Reference* 中的 Language levels and language extensions。

与 GNU 的兼容性

XL C/C++ 支持 GNU 编译器命令选项的子集，以便于移植用 **gcc** 和 **g++** 编译器开发的应用程序。

当 **gxlc** 或 **gxlc++** 调用命令与选择 GNU 编译器选项一起使用时，此支持可用。在调用编译器之前，编译器尽可能地将 GNU 选项映射至它们的对等 XL C/C++ 编译器选项。

调用命令使用纯文本配置文件来控制 GNU 至 XL C/C++ 选项映射和缺省值。可以定制此配置文件以满足您的独特编译需求。有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″Reusing GNU C/C++ compiler options with gxlc and gxlc++″。

XL C/C++ 将 GNU C 和 GNU C++ 头文件与 GNU C 和 C++ 运行时库一起使用，以生成与用 GNU 编译器集合 (GCC) 生成的代码是二进制兼容的代码。应用程序的一些部分可用 XL C/C++ 构建并可将它们与用 GCC 构建的一些部分组合在一起来生成应用程序，并且该应用程序能正常运行就好像它是单独用 GCC 构建的一样。

要获取与 GCC 编译的代码的二进制兼容性，用 XL C/C++ 编译的程序应包括与驻留在同一系统上的 GNU 编译器所使用的一样的头文件。要确保系统中存在头文件和运行时库的适当版本，在安装 XL C/C++ 之前，必须先安装必备的 GCC 编译器。

有关此关系的其他一些值得注意的方面是:

- IBM 内置函数与 GNU C 内置函数可以共存。
- C 和 C++ 程序的编译使用 GNU C 和 GNU C++ 头文件。
- 编译将 GNU 汇编程序用于汇编程序输入文件。
- 已编译的 C 代码将链接至 GNU C 运行时库。
- 已编译的 C++ 代码将链接至 GNU C 和 GNU C++ 运行时库。
- 进行调试操作将使用 GNU 调试器 **gdb**

源代码迁移和一致性检查

XL C/C++ 提供编译器调用命令，以指示编译器将应用程序代码编译到特定语言级别。

还可以使用 **-qlanglvl** 编译器选项来指定语言级别。如果程序源中的语言或语言扩展元素不符合所指定的语言级别，那么编译器会发出诊断消息。

有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qlanglvl。

库

XL C/C++ 包括了包含多个库的运行时环境。

Mathematical Acceleration Subsystem 库

Mathematical Acceleration Subsystem (MASS) 库由为了在受支持的处理器体系结构上获取最佳性能而专门进行调整的标量和向量数学内部函数组成。您可以选择 MASS 库来在各种各样的处理器上支持高性能计算，也可以选择为支持特定处理器系列而调整的库。

MASS 库函数支持 32 位和 64 位编译方式，并在缺省 `libm` 数学库例程上提供了改进的性能。这些库是线程安全的，并且当您请求对应用程序执行特定级别的优化时会自动调用这些库。您也可以显式地调用 MASS 库函数，而不必考虑优化选项是否生效。

有关更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 ″Using the Mathematical Acceleration Subsystem″。

基本线性代数子程序

libxlopt 库中附带提供了高性能代数函数的基本线性代数子程序 (BLAS) 集合。可使用这些功能来:

- 计算一般矩阵或其转置矩阵的矩阵向量乘积。
- 对一般矩阵或其转置矩阵执行组合的矩阵乘法和加法。

有关使用 BLAS 函数的更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using the Basic Linear Algebra Subprograms。

其他库

还随 XL C/C++ 提供了以下库:

- SMP 运行时库，它支持显式并行处理和自动并行处理。请参阅 *XL C/C++ Optimization and Programming Guide* 中的 SMP Runtime Library。
- XL C++ 运行时库包含编译器所需要的支持例程。

对 Boost 库的支持

IBM XL C/C++ for Linux V13.1 提供部分支持 Boost V1.55.0 库。提供了用于修改 Boost V1.55.0 库的补丁文件，以便可以构建这些库并将其与 XL C/C++ 应用程序配合使用。补丁或修改文件不会扩展 Boost 库，也不会向其提供其他功能。

要访问用于构建 Boost 库的补丁文件，请转至 Boost Library Regression Test Summaries 并为您的编译器发行版和平台选择下载必需的 `Boost` 修改文件。

可以在以下地址下载最新的 Boost 库: http://www.boost.org/。

有关支持库的更多信息，请搜索 XL C/C++ 编译器支持页面，网址为 http://www.ibm.com/support/entry/portal/overview/software/rational/xl_c~c++_for_linux。

工具，实用程序和命令

本主题介绍 XL C/C++ 中提供的主要工具、实用程序和命令。本主题并未介绍所有编译器工具、实用程序和命令。

工具

使用情况报告工具

使用情况报告工具生成一个报告来描述组织对编译器的使用情况。这些报告有助于确定组织对编译器的使用是否与编译器许可证权利匹配。**urt** 命令包含一些可以用于对报告进行定制的选项。 有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Tracking and reporting compiler usage。

实用程序

gxlc 和 gxlc++ 实用程序

gxlc 和 **gxlc++** 实用程序将 GNU C 和 GNU C++ 调用命令转换为相应的 **xlc** 和 **xlc++** 命令，然后再调用 XL C/C++ 编译器。这些实用程序的目的是最大程度地减少对用 GNU 编译器构建的现有应用程序的 makefile 的更改并使得向 XL C/C++ 编译器的过渡更方便。有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Reusing GNU C/C++ compiler options with gxlc and gxlc++。

new_install

new_install 实用程序在您安装编译器后对 IBM XL C/C++ for Linux V13.1 进行配置以在系统上使用。从 V13.1 开始，可使用 **new_install** 实用程序来促进 XL C/C++ 与 IBM Advance Toolchain 配合使用。有关详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 Using IBM XL C/C++ for Linux V13.1 with the Advance Toolchain。

xlc_configure

vac_configure 实用程序将创建其他编译器配置文件，以包含您自己定制的编译器选项缺省设置集合。有关更多信息，请参阅 *XL C/C++ Installation Guide* 中的 Running the xlc_configure utility directly (for advanced users)。

命令

genhtml 命令

genhtml 命令将转换由 **-qlistfmt** 选项生成的现有 XML 诊断报告。您可以选择通过使用 **-qlistfmt** 选项来生成 XML 或 HTML 诊断报告。该报告可以帮助进行优化。有关如何使用此命令的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 **genhtml** command。

与概要文件定向反馈 (PDF) 相关的命令

cleanpdf 命令

cleanpdf 命令将从写入概要文件定向反馈数据的目录中除去所有 PDF 文件或指定的 PDF 文件。

mergepdf 命令

mergepdf 命令提供了在将两个或更多 PDF 记录合并为单个记录时对这些记录的重要性进行权衡的能力。PDF 记录必须从相同的可执行文件派生。

resetpdf 命令

resetpdf 命令的当前行为与 **cleanpdf** 命令的行为相同，保留它是为了与其他平台上的较早发行版兼容。

showpdf 命令

showpdf 命令将显示 PDF 运行期间执行的所有过程（在 **-qpdf1** 选项下进行编译）的下列类型的概要分析信息:

- 块计数器概要分析
- 调用计数器概要分析
- 值概要分析
- 高速缓存不命中概要分析（如果在 **-qpdf1** 阶段中指定了 **-qpdf1=level=2** 选项）。

可以使用文本或 XML 格式来查看头两种类型的概要分析信息。然而，只能使用 XML 格式来查看值概要分析和高速缓存不命中概要分析信息。

有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qpdf1，-qpdf2。

程序优化

XL C/C++ 提供了几个可以帮助您控制程序优化和性能的编译器选项。

使用这些选项，您可以执行下列任务:

- 选择不同级别的编译器优化。
- 控制对循环、浮点和其他类型的操作的优化。
- 根据程序将在何处运行，针对特定类别的机器或非常具体的机器配置优化程序。

优化变换可以给予应用程序更好的整体执行性能。XL C/C++ 提供了一些针对各种受支持硬件的优化变换的组合。这些变换提供下列优点:

- 减少为关键操作执行的指令数
- 重构生成的对象代码以便以最佳方式使用 Power Architecture
- 改进内存子系统的使用
- 利用体系结构的能力来处理大量的共享内存并行化

有关更多信息，请参阅以下相关主题:

- *XL C/C++ Optimization and Programming Guide* 中的 Optimizing your applications
- *XL C/C++ Compiler Reference* 中的 Optimizing and tuning options
- *XL C/C++ Compiler Reference* 中的 Compiler built-in functions

64 位对象能力

XL C/C++ 编译器的 64 位对象能力满足了用户对存储器容量需求和处理能力的不断增长的需要。

Linux 操作系统提供一个环境，该环境允许您开发和执行通过使用 64 位地址空间来利用 64 位处理器的程序。

为了支持可以适合 64 位地址空间的更大的可执行文件，使用了单独的 64 位对象格式。链接程序绑定这些对象以创建 64 位的可执行文件。绑定在一起的对象必须具有相同的对象格式。不允许使用以下方案，将无法装入和/或执行:

- 从 32 位库或共享库中引用符号的 64 位对象或可执行文件
- 从 64 位库或共享库中引用符号的 32 位对象或可执行文件
- 尝试显式地装入 32 位模块的 64 位可执行文件
- 尝试显式地装入 64 位模块的 32 位可执行文件

XL C/C++ 主要通过使用 **-q64** 和 **-qarch** 编译器选项来支持 64 位方式。此组合确定目标体系结构的位方式和指令集。

有关更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 ″Using 32-bit and 64-bit modes″。

共享内存并行化

XL C/C++ 支持多处理器系统体系结构的应用程序开发。

您可以使用下列任何一种方法通过 XL C/C++ 来开发并行化应用程序:

- 基于伪指令的共享内存并行化
- 指示编译器自动生成共享内存并行化
- 基于共享或分布式内存并行化 (MPI) 的消息传递

有关更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Parallelizing your programe 。

OpenMP 伪指令

OpenMP 伪指令是一组基于 API 的命令，它们受 XL C/C++ 和许多其他 IBM 以及非 IBM C、C++ 和 Fortran 编译器的支持。

可以使用 OpenMP 伪指令来指示编译器如何并行化特定循环。源代码中存在这些伪指令可使得编译器无需对并行代码执行任何并行分析。OpenMP 伪指令要求存在 Pthread 库以提供用于并行化的必要基础结构。

OpenMP 伪指令解决并行化应用程序的三个重要问题:

1. 子句和伪指令可用于限定作用域的变量。通常，不应共享变量；即，每个处理器都应具有它自己的变量副本。
2. 工作共享伪指令指定应如何将包含在代码的并行区域中的工作分布在各个处理器上。
3. 伪指令可用来控制处理器之间的同步。

从 IBM XL C/C++ for Linux V13.1 开始，XL C/C++ 支持 OpenMP API V3.1 和 OpenMP API V4.0 规范的选定功能。有关详细信息，请参阅 第 12 页的『OpenMP 4.0』。

有关程序性能优化的更多信息，请参阅:

- *XL C/C++ Optimization and Programming Guide* 中的 Optimizing your applications
- The OpenMP API specification for parallel programming

诊断列表

编译器输出列表和 XML 或 HTML 报告提供了重要的信息，以帮助您更有效地开发和调试应用程序。

列表信息已加到您可以包括或省略的可选节中。有关适用的编译器选项以及该列表本身的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler messages andlistings。

还可以从编译器中获取有关编译器之前执行或未执行的某些优化的 XML 或 HTML 格式的诊断信息。可使用此信息来在调整应用程序时减少编程工作量，对于高性能应用程序更是如此。该报告通过 XML 模式进行定义，由您创建的用于读取和分析结果的工具能够轻易使用该报告。有关此报告以及如何进行使用的详细信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 ″Using reports to diagnose optimization opportunities″。

符号调试器支持

您可使用 **-g** 编辑器选项的不同级别来指示 XL C/C++ 将调试信息包含在已编译的对象中。

更多详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 **-g**。

调试信息可以由 **gdb** 或任何其他 上受支持的符号调试器检查，以便帮助您调试程序。

第 2 章 IBM XL C/C++ for Linux V13.1的新增内容

本节描述添加到 IBM XL C/C++ for Linux V13.1中的功能部件和增强功能。

支持 POWER8 处理器

XL C/C++ for Linux V13.1 支持 POWER8 处理器。

为了支持 POWER8 处理器而引入的新功能和增强功能分为以下几个类别:

- POWER8 处理器的 MASS 库
- POWER8 处理器的编译器选项
- POWER8 处理器的内置函数

POWER8 处理器的 Mathematical Acceleration Subsystem (MASS) 库

向量库

向量 MASS 库 **libmassvp8.a** 包含已针对 POWER8 体系结构进行调整的向量函数。可以在 32 位方式或 64 位方式下使用这些函数。

有关向量库的更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using the vector libraries。

SIMD 库

MASS SIMD 库 **libmass_simdp8.a** 包含一组已加速的常用数学内部函数，它们的性能优于相应的标准系统库函数。

有关 SIMD 库的更多信息，请参阅 *XL C/C++ Optimization and Programming Guide*中的 Using the SIMD libraries。

POWER8 处理器的编译器选项

-qarch 编译器选项指定生成代码所针对的处理器体系结构。**-qtune** 编译器选项调整指令选择、调度和其他与体系结构有关的性能增强功能，以在特定的硬件体系结构中以最优状态运行。

新的 **-qarch=pwr8** 子选项将生成对象代码，该对象代码包含将在 POWER8 硬件平台上运行的指令。如果指定了新的 **-qtune=pwr8** 子选项，那么将针对 POWER8 硬件平台对优化进行调整。

有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qarch 和 *XL C/C++ Compiler Reference* 中的 -qtune。

POWER8 处理器的内置函数

添加了新硬件内在机制以支持以下 POWER8 处理器功能:

- 用于向量处理的 POWER8 内在机制
- POWER8 二进制编码的十进制函数

© Copyright IBM Corp. 1996, 2014

- POWER8 密码术函数
- POWER8 四字四则运算函数
- POWER8 负荷和保留/存储条件指令
- POWER8 高速缓存和数据预取控制函数
- POWER8 事务型内存函数
- POWER8 预取函数

有关 XL C/C++ 提供的内置函数的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler built-in functions。

Advance Toolchain 7.0 支持

IBM XL C/C++ for Linux V13.1 完全支持 IBM Advance Toolchain 7.0，该应用程序是一组开放式源代码开发工具和运行时库。通过 IBM Advance Toolchain，您可以在 Linux 上利用最新 POWER® 硬件功能部件，尤其是已调整的库。

有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 “Using IBM XL C/C++ for Linux V13.1 with the Advance Toolchain”。

C++11 功能

除了现有 C++11 功能之外，在此发行版的 XL C/C++ 中还支持新的 C++11 功能。

注: IBM 支持 C++11（在批准之前称为 C++0x）的所选功能。IBM 将继续开发和实现此标准的功能部件。此语言级别的实现基于 IBM 对该标准的解释。在 IBM 完成所有 C++11 功能（包括支持新的 C++11 标准库）的实现之前，该实现可能会随发行版的不同而改变。IBM 在实现新的 C++11 功能时，不会尝试在源代码、二进制代码或列表以及其他编译器接口方面保持与先前发行版的兼容性。

XL C/C++ V13.1 中引入了下列功能:

- 缺省函数和已删除的函数
- `nullptr` 关键字

XL C/C++ V13.1 中增强了一般化的常量表达式功能。

可使用 **-qlanglvl=extended0x** 选项来启用大多数 C++ 功能和当前支持的所有 C++11 功能。有关详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qprefetch。

缺省函数和已删除的函数

此功能引入了两种新的函数声明形式以显式地定义缺省函数和已删除的函数。对于显式缺省函数，编译器将生成比手动编程的实现更有效的缺省实现。编译器将禁用已删除的函数以避免调用不需要的函数。

可使用 **-qlanglvl=defaultanddelete** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Explicitly defaulted functions (C++11) 和 Deleted functions (C++11)。

一般化的常量表达式

一般化的常量表达式功能扩展了允许在常量表达式中使用的表达式集合。在 XL C/C++ V12.1 中，此功能的实现是 C++11 标准中所定义的相应功能的部分实现。在此发行版中，将进行增强以支持用户定义的 constexpr 对象和 constexpr 指针或者对 constexpr 函数和对象的引用。

可以使用 **-qlanglvl=constexpr** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 "Generalized constant expressions (C++11)"。

nullptr 关键字

此功能引入了 nullptr 作为 null 指针常量。对于超负荷函数，nullptr 常量可与整数 0 区分开。常量 0 和 NULL 视为超负荷函数的整数类型，而 nullptr 只能显式转换为指针类型、至成员的指针类型和 bool 类型。

可以使用 **-qlanglvl=nullptr** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 langref.pdf#nullpt。

XL C/C++ Compiler Reference 中的相关信息

-qlanglvl

C11 功能

除了现有 C11 功能之外，此发行版的 XL C/C++ 中还支持新增 C11 功能。

注: IBM 支持 C11（批准之前又称为 C1X）的所选功能。IBM 将继续开发和实现此标准的功能部件。此语言级别的实现基于 IBM 对该标准的解释。在 IBM 完成 C11 标准的所有功能（包括支持新的 C11 标准库）的实现之前，该实现可能会随着发行版的不同而改变。IBM 在实现新的 C++11 功能时，不会尝试在源代码、二进制代码或列表以及其他编译器接口方面保持与先前发行版的兼容性。

IBM XL C/C++ for Linux V13.1 中引入了下列功能:

- typedef 重新声明
- 类属选择

typedef 重新声明

通过使用 typedef 重新声明，您可以重新定义同一作用域中先前名称为 typedef 的名称，以引用同一类型。IBM XL C 编译器支持所有类型，包括便于修改的类型。IBM 有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 "typedef definitions"。

类属选择

类属选择提供了一个机制，以在编译时根据给定类型名称选择表达式。一般用法是定义类型类属宏。有关更多信息，请参阅 Generic selection (C11)。

OpenMP 4.0

IBM XL C/C++ for Linux V13.1部分支持 OpenMP Application Program Interface V4.0 规范。XL C/C++ 实现基于 IBM 对 OpenMP Application Program Interface V4.0 的解释。

此版本的 XL C/C++ 支持以下 OpenMP 4.0 功能:

- `update` 和 `capture` 子句增强功能
- OMP_DISPLAY_ENV 环境变量

`update` 和 `capture` 子句增强功能

已扩展 `atomic` 构造的 `update` 和 `capture` 子句以支持更多表达式形式。

OMP_DISPLAY_ENV 环境变量

可使用环境变量 OMP_DISPLAY_ENV 来显示与环境变量关联的内部控制变量 (ICV) 的值以及有关运行时库的特定于构建的信息。

相关信息

- *XL C/C++ Compiler Reference* 中的 ″OpenMP environment variables″
- *XL C/C++ Compiler Reference* 中的 ″Pragma directives for parallel processing″
- The OpenMP API specification for parallel programming

内置函数

以下主要类别的内置函数 是此发行版新增的。

用于向量处理的 POWER8 内置函数

增加了以下向量内置函数:

- 向量 gather-bits-by-bytes 双字函数
  - `vec_gbb`
- 向量计数前导零函数
  - `vec_cntlz`
- 向量填充计数函数
  - `vec_popcnt`
- 扩展的向量逻辑运算函数
  - `vec_eqv`
  - `vec_nand`
  - `vec_orc`
- 128 位整数增减函数
  - `vec_add_u128`
  - `vec_sub_u128`
  - `vec_adde_u128`
  - `vec_sube_u128`

  - vec_addc_u128
  - vec_subc_u128
  - vec_addec_u128
  - vec_subec_u128
  - vec_bperm

已扩展以下内置函数以支持双字类型:

- 向量压缩函数
  - vec_pack
  - vec_packs
  - vec_packsu
- 向量解压缩函数
  - vec_unpackh
  - vec_unpackl
- 向量增减函数
  - vec_add
  - vec_sub
- 向量最大和最小函数
  - vec_max
  - vec_min
- 向量切换和旋转函数
  - vec_rl
  - vec_sl
  - vec_sr
  - vec_sra
- 向量比较函数
  - vec_cmpeq
  - vec_cmpgt
  - vec_cmpge
  - vec_cmplt
  - vec_cmple

POWER8 二进制编码的十进制内置函数

已添加以下内置函数以支持二进制编码的十进制 (BCD) 四则运算和比较:

- BCD 增减函数
  - __bcdadd
  - __bcdsub
- 针对溢出函数的 BCD 测试增减
  - __bcdadd_ofl
  - __bcdsub_ofl

  - __bcd_invalid
- BCD 比较函数
  - __bcdcmpeq
  - __bcdcmpgt
  - __bcdcmpge
  - __bcdcmplt
  - __bcdcmple
- BCD 负荷函数和存储函数
  - __vec_ldrmb
  - __vec_strmb

POWER8 密码术内置函数

已提供以下内置函数以执行密码操作:

- 高级加密标准 (AES) functions
  - __vcipher
  - __vcipherlast
  - __vncipher
  - __vncipherlast
  - __vsbox
- 安全散列算法 (SHA) 函数
  - __vshasigmad
  - __vshasigmaw
- 其他函数
  - __vpmsumb
  - __vpmsumh
  - __vpmsumw
  - __vpmsumd
  - __vpermxor

POWER8 非向量内置函数

已添加以下内置函数以改进高速缓存的效率:

- __dcbtna
- __icbt

已使用以下函数扩展负荷和存储内置函数以支持更多类型:

- __lqarx
- __lharx
- __lbarx
- __stqcx
- __sthcx
- __stbcx

POWER8 事务型内存内置函数

事务型内存是用于并行编程的模型。在此模型中，您可以指定要自动处理的指示信息或语句的块操作。

可使用以下内置函数来标记事务的开始或结束，并对故障原因进行诊断:

- 事务开始和结束函数
  - `__TM_begin`
  - `__TM_end`
  - `__TM_simple_begin`
- 事务异常中止函数
  - `__TM_abort`
  - `__TM_named_abort`
- 事务查询函数
  - `__TM_failure_address`
  - `__TM_failure_code`
  - `__TM_is_conflict`
  - `__TM_is_failure_persistent`
  - `__TM_is_footprint_exceeded`
  - `__TM_is_illegal`
  - `__TM_is_named_user_abort`
  - `__TM_is_nested_too_deep`
  - `__TM_is_user_abort`
  - `__TM_nesting_depth`

POWER8 预取内置函数

以下内置函数采用直觉、可移植和优化友好方式显示 Data Stream Control Register (DSCR) 问题状态控制:

- 瞬态属性启用函数
  - `__hardware_transient_enable`
  - `__load_transient_enable`
  - `__software_transient_enable`
  - `__store_transient_enable`
- 单元计数启用和设置函数
  - `__hardware_unit_count_enable`
  - `__software_unit_count_enable`
  - `__set_prefetch_unit_count`
- 预取深度函数
  - `__default_prefetch_depth`
  - `__depth_attainment_urgency`
- 负荷流启用和禁用函数

- `__load_stream_disable`
- `__stride_n_stream_enable`

- DSCR 函数
  - `__prefetch_get_dscr_register`
  - `__prefetch_set_dscr_register`

注: 仅当设置 **-qarch** 以将 POWER8 处理器作为目标时，POWER8 内置函数才有效。

有关 XL C/C++ 提供的内置函数的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler built-in functions。

编译器选项和编译指示伪指令

本节描述新增的或已更改的编译器选项和编译指示伪指令。

可以在命令行上指定编译器选项。还可以通过应用程序源文件中嵌入的编译指示伪指令来修改编译器行为。有关 XL C/C++ 编译器选项的详细描述和用法信息，请参阅 *XL C/C++ Compiler Reference*。

-qarch

该选项缺省值已更新为 **pwr5**。表示旧硬件系列的子选项将以静默方式升级到更新的体系结构。

增加或更新了下列子选项:

-qarch=pwr7

此子选项将生成对象代码，该对象代码包含将在 POWER7、POWER7+ 或 POWER8 硬件平台上运行的指令。

-qarch=pwr8

此子选项将生成对象代码，该对象代码包含将在 POWER8 硬件平台上运行的指令。

-qcheck

增加或更新了下列子选项:

-qcheck=stackclobber

此子选项将检测您的程序中某种类型的堆栈溢出。

-qcheck=unset

此子选项将检查在运行时设置自动变量之前使用的这些变量。

-qdbgfmt

添加了以下子选项:

-qdbgfmt=dwarf

此子选项将使用 DWARF 3 格式生成调试信息。

-qdbgfmt=dwarf4

此子选项将使用 DWARF 4 格式生成调试信息。

-qhelp 此选项将显示编译器的联机帮助页。

-qinfo

编译器不会针对下列文件发出参考消息:

- 编译器的标准搜索路径中的文件和系统头文件。
- 编译器的标准搜索路径中的文件最终包括的文件和系统头文件。

增加或更新了下列子选项:

-qinfo=mt

此子选项通知您可能需要同步的位置。

-qinfo=unset

此子选项将检测设置自动变量之前使用的这些变量，并使用编译时的参考消息对其进行标记。

-qlanglvl

增加或更新了下列子选项:

C++11 **-qlanglvl=defaultanddelete**

此子选项将启用缺省函数和已删除的函数功能部件，通过此功能部件，您可以显式地定义其实现由编译器生成的缺省函数，以达到更高的效率。通过此功能部件，还可以定义由编译器禁用的已删除的函数，以避免调用不需要的函数。 C++11

C++11 **-qlanglvl=nullptr**

此子选项启用 `nullptr` 功能。通过此功能，您可以使用 `nullptr` 常量初始化 null 指针。null 指针可转换为指针类型，至成员的指针类型或 bool 类型。对于超负荷函数，`nullptr` 常量可与整数 0 区分开。

C++11

-qpdf1=unique

此子选项将在运行时期间为每个进程创建唯一的 PDF 文件。

-qprefetch=dscr

此子选项有助于改进应用程序的运行时性能。可根据系统体系结构指定 `dscr` 的值。

-qsimd=auto

此子选项控制自动，该操作由不推荐使用的 **-qhot=simd** 选项执行。

-qstaticlink=xllibs

此子选项以静态方式链接 XL 编译器库。

-qtune

已更新选项缺省值。

增加或更新了下列子选项:

-qtune=pwr7

此子选项指定针对 POWER7 或 POWER7+ 硬件平台对优化进行调整。

-qtune=pwr8

此子选项指定针对 POWER8 硬件平台对优化进行调整。

SMT 子选项

新的 **-qtune** 同步多线程技术 (SMT) 子选项允许您指定目标 SMT 来为在该方式下获得最佳性能而定向优化。

-qunroll=*n*

此子选项提示编译器由于因素 *n* 而展开循环。如果循环次数少于 *n* 个迭代，那么将完全对其展开循环。

-qvisibility

此选项指定实体的可视性属性。实体可视性属性描述了在一个模块中定义的实体是否以及如何可在其他模块中引用或使用。可视性属性只影响具有外部链接的实体，并且不会增加其他实体的可视性。

新增或已更改的编译指示伪指令

#pragma GCC visibility push, #编译指示 GCC 可视性弹出

这一对编译指示伪指令是等效于 **-qvisibility** 选项的编译指示。编译指示伪指令用于指定外部链接符号的可视性属性。

性能和优化

其他功能部件和增强功能可以帮助进行性能调整和应用程序优化。

IBM

实体的可视性属性

实体可视性属性描述了在一个模块中定义的实体是否以及如何可在其他模块中引用或使用。通过对实体使用可视性属性，您可以获得以下优点:

- 降低共享库的大小
- 减少符号冲突的可能性
- 允许进一步对编译和链接阶段进行优化
- 改进动态链接的效率

有关更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using visibility attributes (IBM extension)。

IBM

有关性能调整和程序优化的更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Optimizing your applications 和 Coding your application to improve performance 。

第 3 章 从先前版本迁移

通过迁移至最新版本的编译器，就能够使用编译器的新功能，其中包括用于帮助提升正在编译的应用程序的性能的功能。

较高版本的编译器包括新增功能和增强功能。这些功能可以提供下列优点:

- 改进了已编译的应用程序的优化并具有其他性能优点
- 支持新的语言标准，以便于在多个操作系统与硬件平台之间移植代码
- 在新的硬件和操作系统中开发最新的功能

在迁移至最新版本的编译器后，您就能够开发新功能，并且还会获得以下优点：提高优化性能、支持新的语言规范以及开发新的硬件和软件环境。

有关当前发行版中的新功能的详细信息，请参阅 第 9 页的第 2 章，『IBM XL C/C++ for Linux V13.1的新增内容』。白皮书 *Upgrading XL C/C++ Compilers* 也提供了有关编译器迁移的其他信息。

通常情况下，新版本的编译器与其先前版本的编译器兼容。然而，可能存在例外情况。例如，可能会生成不同的诊断消息。在迁移编译器之前，请始终备份源代码和其他重要数据。

以下各节列示了对先前版本的编译器增加的增强功能，这些增强功能可以帮助您迁移至较高版本的编译器。

V12.1 中新增的增强功能

本节描述添加到V12.1 中的编译器的功能部件和增强功能。

C++11 功能

C++11 是新的 C++ 编程语言标准。在批准之前，C++11 称为 C++0x。除了现有 C++11 功能之外，XL C/C++ V12.1 中支持新的 C++11 功能。

注: IBM 支持 C++11（在批准之前称为 C++0x）的所选功能。IBM 将继续开发和实现此标准的功能部件。此语言级别的实现基于 IBM 对该标准的解释。在 IBM 完成所有 C++11 功能（包括支持新的 C++11 标准库）的实现之前，该实现可能会随发行版的不同而改变。IBM 在实现新的 C++11 功能时，不会尝试在源代码、二进制代码或列表以及其他编译器接口方面保持与先前发行版的兼容性。

XL C/C++ V12.1 中引入了下列功能:

- 显式转换运算符
- 一般化的常量表达式
- 引用折叠
- 右尖括号
- 右值引用
- 有作用域的枚举

© Copyright IBM Corp. 1996, 2014

- 跟踪返回类型

可以使用 **-qlanglvl=extended0x** 选项来启用大多数 C++ 功能和当前支持的所有 C++11 功能。有关详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qprefetch。

显式转换运算符

显式转换运算符功能支持将 `explicit` 函数说明符应用于用户定义的转换函数的定义。您可以在可能不需要隐式转换的情况下使用此功能来阻止应用隐式转换，因此您可以在具有较少模糊错误的情况下，编写更加强大的类。

可以使用 **-qlanglvl=explicitconversionoperators** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Explicit Conversion Operators (C++11)。

一般化的常量表达式

一般化的常量表达式功能扩展了允许在常量表达式中使用的表达式集合。常量表达式是可以在编译时进行求值的表达式。

可以使用 **-qlanglvl=constexpr** 选项来启用此功能。

注: 在 XL C/C++ V12.1 中，此功能是 C++11 标准中所定义的相应功能的部分实现。

引用折叠

借助引用折叠功能，可以使用下列其中一个上下文来形成对引用类型的引用。

- `decltype` 说明符
- `typedef` 名称
- 模板类型参数

可以使用 **-qlanglvl=referencecollapsing** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Reference collapsing (C++11)。

右尖括号

在 C++ 语言中，必须用空格来分隔两个连续的右尖括号 (>)，否则这些右尖括号将解析为按位右移运算符 (>>)。右尖括号功能将取消连续的右尖括号之间必须有空格的要求，从而使编程更方便。

可以使用 **-qlanglvl=rightanglebracket** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Class templates (C++ only)。

右值引用

借助右值引用功能，可以根据参数的值类别来重载函数并以类似方式通过模板参数推导来检测左值。还可以将右值绑定至右值引用并通过该引用来修改此右值。这将启用

一种编程技巧，您可以使用该技巧来复用到期对象的资源，从而提高库的性能，尤其是在将类属代码与类类型（例如，模板数据结构）配合使用时。另外，在编写转发函数时可以考虑使用值类别。

可以使用 **-qlanglvl=rvaluereferences** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using rvalue references (C++11)。

有作用域的枚举

借助有作用域的枚举功能，您可以获得下列优点:

- 能够声明有作用域的枚举类型，其枚举符在该枚举的作用域中进行声明。
- 能够在不提供枚举符的情况下声明枚举。在不提供枚举符的情况下声明枚举称为正向声明。
- 能够显式地指定枚举的底层类型。
- 由于不会将枚举符（或枚举类型的对象）的值转换为整数，从而提高了类型安全性。

可以使用 **-qlanglvl=scopedenum** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Enumeration。

跟踪返回类型

在声明下列类型的模板和函数时，跟踪返回类型功能很有用:

- 其返回类型取决于函数参数类型的函数模板或类模板的成员函数
- 具有复杂返回类型的函数或类的成员函数
- 完美的转发函数

可以使用 **-qlanglvl=autotypededuction** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Trailing return type (C++11)。

XL C/C++ Compiler Reference **中的相关信息**

-qlanglvl

C11 功能

XL C/C++ V12.1 引入了对 C11 的选定功能的支持，这是新增的 C 编程语言标准。

注: IBM 支持 C11（批准之前又称为 C1X）的所选功能。IBM 将继续开发和实现此标准的功能部件。此语言级别的实现基于 IBM 对该标准的解释。在 IBM 完成 C11 标准的所有功能（包括支持新的 C11 标准库）的实现之前，该实现可能会随着发行版的不同而改变。IBM 在实现新的 C++11 功能时，不会尝试在源代码、二进制代码或列表以及其他编译器接口方面保持与先前发行版的兼容性。

XL C/C++ V12.1 中引入了下列 C11 功能:

- 匿名结构

- 复杂类型初始化
- 新的语言级别 - **extc1x**
- _Noreturn 函数说明符
- 静态断言

匿名结构

此功能允许在 **extc1x** 语言级别声明匿名结构。有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Anonymous structures。

复杂类型初始化

将在标准头文件 complex.h 中定义宏 CMPLX、CMPLXF 和 CMPLXL，以允许在 **extc1x** 语言级别对复杂类型进行初始化。有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Initialization of complex types (C11)。

新的语言级别 - extc1x

在此发行版中，新的子选项已添加至 **-qlanglvl** 选项。使用 C 编译器进行编译时，可以使用 **-qlanglvl=extc1x** 来启用 XL C/C++ 当前所支持的 C11 功能。使用 C++ 编译器进行编译时，还可以使用某些 C11 功能。有关更多信息，请参阅描述各个功能的章节。

_Noreturn 函数说明符

_Noreturn 函数说明符用于声明不会将函数返回至其调用者。您可以使用此函数说明符来定义自己的不进行返回的函数。编译器可以通过忽略该函数返回时所出现的情况来生成更好的代码。有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 The _Noreturn function specifier。

静态断言

在 C 语言中增加静态断言具有下列优点:

- 库可以在编译时检测一般用法错误。
- C 标准库的实现可以检测和诊断一般用法错误，从而提高可用性。

可以声明静态断言以在编译时检查重要的程序不变量。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 _Static_assert declaration (C11)。

OpenMP 3.1

XL C/C++ V12.1 支持 OpenMP Application Program Interface V3.1 规范。XL C/C++ 实现基于 IBM 对 OpenMP Application Program Interface V3.1 的解释。

OpenMP 3.1 包括下列对 OpenMP 3.0 的更新:

- 将 `final` 和 `mergeable` 子句添加到 `task` 构造以支持优化。
- 添加 `taskyield` 构造以允许用户指定程序可以在何处执行任务切换。
- 添加 `omp_in_final` 运行时库函数以支持最终任务区域的特例化。

- 扩展 atomic 构造以包括 read、write 和 capture 格式；添加 update 子句以应用 atomic 构造的现有格式。
- 添加两个归约运算符：min 和 max。
- 允许在 firstprivate 子句中指定用常量限定的类型。
- 添加 OMP_PROC_BIND 环境变量以控制是否允许 OpenMP 线程在处理器之间移动。
- 扩展 OMP_NUM_THREADS 环境变量以指定要用于嵌套并行区域的线程数。

相关信息

- *XL C/C++ Compiler Reference* 中的 ″OpenMP environment variables″
- *XL C/C++ Compiler Reference* 中的 ″Pragma directives for parallel processing″
- www.openmp.org

性能和优化

XL C/C++ V12.1 中的其他功能和增强功能可以帮助进行性能调整和应用程序优化。

有关编译器优化的报告

对列表报告的许多功能进行了增强，以便为您提供有关编译器如何优化代码的更多信息。可以使用此信息来从编译器的优化功能中获取更多的优点。有关这些所增强报告的更多详细信息，请参阅『诊断报告』。

有关性能调整和程序优化的其他信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Optimizing your applications。

诊断报告

XL C/C++ V12.1 中添加的新诊断报告可以帮助您确定改进代码性能的时机。

HTML 格式的编译器报告

现在，可获取有关下列内容的 XML 或 HTML 格式的信息：编译器之前能够执行的优化以及未执行的优化。此信息可用来在调整应用程序时减少编程工作量，对于高性能应用程序更是如此。

可以使用 **-qlistfmt** 选项及其相关联的子选项来生成 XML 或 HTML 报告。缺省情况下，现在此选项在您未指定内容的类型时生成所有可用内容。

要查看已生成的 XML 报告的 HTML 版本，现在可以使用 **genhtml** 工具。有关如何使用此工具的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 **genhtml** command。

有关此报告以及如何进行使用的详细信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 ″Using reports to diagnose optimization opportunities″。

对概要分析报告的增强

对列表文件增加了新节以帮助对程序进行分析。将 **-qreport** 选项与 **-qpdf2** 选项配合使用时，可以将下列各节添加至列表文件中标题为 PDF Report 的节中：

概要分析数据的相关性

本节显示 **-qpdf1** 阶段中概要分析数据与源代码的相关性。相关性由范围 0 到

100 中的数字指示。该数字越大，概要分析数据与源代码的相关性就越高，并且使用概要分析数据可以获得的性能增益更多。

缺少概要分析数据

本节可能包括有关缺少概要分析数据的警告消息。将对编译器找不到其概要分析数据的每个函数发出该警告消息。

过时的概要分析数据

本节可能包括有关过时的概要分析数据的警告消息。编译器将对 **-qpdf1** 阶段之后进行了修改的每个函数发出该警告消息。当优化级别从 **-qpdf1** 阶段更改为 **-qpdf2** 阶段时，也会发出该警告消息。

有关概要文件定向反馈的详细信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using profile-directed feedback。

有关列表文件的其他信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler listings。

对 showpdf 报告的增强

除了当前提供的块计数器和调用计数器概要分析信息之外，您还可以使用 **showpdf** 实用程序来查看高速缓存不命中概要分析和值概要分析信息。只能使用 XML 格式来显示值概要分析和高速缓存不命中概要分析信息。然而，可以使用文本或 XML 格式来显示所有其他类型的概要分析信息。在此发行版中，概要文件定向反馈 (PDF) 信息保存在两个文件中。一个文件是 **-qpdf1** 阶段期间生成的 PDF 映射文件，另一个文件是执行所产生的应用程序期间生成的 PDF 文件。可以运行 **showpdf** 实用程序来显示这两个文件中包含的 PDF 信息。有关更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Viewing profiling information with showpdf。

新增和增强的诊断选项

下表中的条目描述新增或已更改的编译器选项和伪指令，它们使您可以对编译器列表进行控制。

此处提供的信息是一个简短的概述。有关这些编译器选项以及其他与性能相关的编译器选项的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″Listings, messages and compiler information″。

表 4. 与列表相关的编译器选项和伪指令

选项/伪指令	描述
-qlistfmt	增强了 **-qlistfmt** 选项，以生成 HTML 报告以及 XML 报告，这些包含有关下列方面的信息：编译器执行的优化以及未执行的优化。 此选项的缺省行为已更改。现在，如果您未指定内容的特定类型，那么该选项将生成所有可用内容，而非不生成任何内容。

内置函数

本节描述自 XL C/C++ 以来 V12.1 新增的内置函数。

GCC 原子内存访问内置函数（IBM 扩展）

此发行版中新增了用于进行原子内存访问的 XL C/C++ 内置函数，这些函数的行为对应于 GNU 编译器集合 (GCC) 所提供的函数的行为。在具有多个线程的程序中，您可以使用这些函数以原子方式安全地修改一个线程中的数据，而不会与其他线程发生冲突。

有关 XL C/C++ 提供的内置函数的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler built-in functions。

编译器选项和编译指示伪指令

本节描述 V12.1 中新增或已更改的编译器选项和编译指示伪指令，

可以在命令行上指定编译器选项。还可以通过应用程序源文件中嵌入的编译指示伪指令来修改编译器行为。请参阅 *XL C/C++ Compiler Reference*，以获得这些编译器选项和其他编译器选项的详细描述和用法信息。

新增或已更改的编译器选项

-g 已将 **-g** 选项扩张为具有新级别，以改进对已优化的程序的调试。

-qhaltonmsg

以前仅受 C++ 编译器支持的 **-qhaltonmsg** 选项现在受 XL C 支持。如果生成了指定的错误消息，那么此选项将在生成任何对象文件、可执行文件或汇编程序源文件之前停止编译。还增加了负数形式的 **-qnohaltonmsg**。

-qinclude

增加了否定形式的 **-qnoinclude**，以忽略先前指定的 **-qinclude** 选项。

-qinfo **-qinfo=all** 现在对除了 **als** 和 **ppt** 之外的所有组启用所有诊断消息

-qinitauto

增强了 **-qinitauto** 选项，以便能够对自动变量执行字初始化。

-qkeyword

C++11 新的子选项 **-q[no]keyword=constexpr** 启用或禁用 `constexpr` 关键字。

-qlanglvl

增加或更新了下列子选项:

C++11 **-qlanglvl=autotypededuction**

此子选项现在除了可以启用自动类型推导功能之外，还可以启用跟踪返回类型功能。

C++ **-qlanglvl=c1xnoreturn**

此子选项启用对 `_Noreturn` 函数说明符的支持。

C++ **-qlanglvl=complexinit**

此子选项控制是否允许对复杂类型进行初始化。

C++ IBM **-qlanglvl=compatrvaluebinding**

此子选项指示编译器允许使用非常量左值引用，以在不需要初始化方法的情况下绑定至用户定义的类型的右值。

C++11 **-qlanglvl=constexpr**

此子选项启用一般化的常量表达式功能，以扩展允许在常量表达式中使用的表达式。

注: 在 XL C/C++ V12.1 中，此功能是 C++11 标准中所定义的相应功能的部分实现。

C++11 **-qlanglvl=explicitconversionoperators**

此子选项启用显式转换运算符功能，该功能允许您通过用户定义的转换函数来阻止不需要的隐式转换。

C11 **-qlanglvl=extc1x**

此子选项启用当前受支持的所有 C11 功能以及其他特定于实现的语言扩展。

C++11 **-qlanglvl=referencecollapsing**

此子选项启用引用折叠功能，借助该功能，您可以使用 `decltype` 说明符、`typedef` 名称或模板类型参数来形成对引用类型的引用。

C++11 **-qlanglvl=rightanglebracket**

此子选项启用右尖括号功能，该功能将取消连续的右尖括号之间必须有空格的要求。

C++11 **-qlanglvl=rvaluereferences**

此子选项启用右值引用功能。

C++11 **-qlanglvl=scopedenum**

此子选项启用有作用域的枚举功能，借助该功能，您可以在不提供枚举符的情况下声明有作用域的枚举类型或枚举。

C++ IBM **-qlanglvl=tempsaslocals**

此子选项延长临时变量的生存期，以降低迁移难度。

IBM **-qlanglvl=textafterendif**

此子选项禁止显示当您将代码从允许 `#endif` 或 `#else` 后面存在额外文本的编译器移植到 IBM XL C/C++ 编译器时发出的警告消息。

有关新 C++11 功能的更多信息，请参阅 第 19 页的『C++11 功能』。

有关 C11 功能的更多信息，请参阅 第 21 页的『C11 功能』。

-qlistfmt

增强了 **-qlistfmt** 选项以生成 HTML 报告以及 XML 报告，这些报告包含有关下列方面的信息: 编译器执行的优化以及未执行的优化。

-qlistfmt 的缺省行为已更改。在此发行版中，如果您未指定内容的特定类型，那么该选项将生成所有可用内容，而非不生成任何内容。

-qoptfile

新的选项 **-qoptfile** 指定一个文件，该文件包含要用于编译的其他命令行选项的列表。

-qpic **-qpic=large** 现在启用大型 TOC 访问并在目录大于 64 KB 时阻止 TOC 溢出情况出现。

-qshowpdf

缺省值已从 **-qnoshowpdf** 更改为 **-qshowpdf**。

新增或已更改的编译指示伪指令

#pragma ibm independent_loop

增加了 **independent_loop** 编译指示。它显式地声明所选循环的迭代是独立的并且可以并行地执行这些迭代。

#pragma ibm iterations

增加了 **iterations** 编译指示。它指定所选循环的循环迭代大概次数。

#pragma ibm max_iterations

增加了 **max_iterations** 编译指示。它指定所选循环的循环迭代大概最大次数。

#pragma ibm min_iterations

增加了 **min_iterations** 编译指示。它指定所选循环的循环迭代大概最小次数。

#pragma simd_level

增加了 **simd_level** 编译指示。它控制各个循环的向量指令的编译器代码生成。

V11.1 中新增的增强功能

本节描述添加到 V11.1 中的编译器的功能部件和增强功能。

支持 POWER7 处理器

XL C/C++ for Linux V11.1 支持 POWER7 处理器。

为了支持 POWER7 处理器而引入的新功能和增强功能分为下列四个类别:

- 向量标量扩展数据类型和内部函数
- POWER7 处理器的 MASS 库
- 用于 POWER7 处理器的内置函数
- POWER7 处理器的编译器选项

向量标量扩展数据类型和内部函数

编译器的此发行版支持 POWER7 处理器中的向量标量扩展 (VSX) 指令集。引入了新的数据类型和内部函数以支持 VSX 指令。借助 VSX 内部函数和原始向量多媒体扩展 (VMX) 内部函数，可以有效地处理应用程序中的向量操作。

有关 VSX 数据类型和内部函数的更多信息，请参阅 *XL C/C++ Language Reference* 中的 Vector types 和 *XL C/C++ Compiler Reference* 中的 Vector built-in functions。

POWER7 处理器的 Mathematical Acceleration Subsystem (MASS) 库

向量库

向量 MASS 库 **libmassvp7.a** 包含已针对 POWER7 体系结构进行调整的向量函数。可以在 32 位方式或 64 位方式下使用这些函数。

POWER7 处理器处理器包含支持先前 Power® 处理器的函数（单精度或双精度）。

在单精度和双精度函数组中添加了下列新函数:

- exp2
- exp2m1

- log21p
- log2

有关向量库的更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using the vector libraries。

SIMD 库

MASS SIMD 库 **libmass_simdp7.a** 包含一组已加速的常用数学内部函数，它们的性能优于相应的标准系统库函数。

有关 SIMD 库的更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using the SIMD library for POWER7。

POWER7 硬件内在机制

添加了新硬件内在机制以支持下列 POWER7 处理器功能:

- 新的 POWER7 预取扩展和高速缓存控制
- 新的 POWER7 硬件指令

有关更多信息，请参阅第 37 页的『内置函数』。

POWER7 处理器的新编译器选项

新的 arch 和 tune 编译器选项

-qarch 编译器选项指定生成代码所针对的处理器体系结构。**-qtune** 编译器选项调整指令选择、调度和其他与体系结构有关的性能增强功能，以在特定的硬件体系结构中以最优状态运行。

-qarch=pwr7 生成对象代码，该对象代码包含将在 POWER7 硬件平台上运行的指令。如果指定了 **-qtune=pwr7**，那么将针对 POWER7 硬件平台对优化进行调整。

有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qarch 和 *XL C/C++ Compiler Reference* 中的 -qtune。

C++11 功能

XL C/C++ V11.1 引入了对 C++11 的选定功能的支持，这是新增的 C++ 编程语言标准。

注: IBM 支持 C++11（在批准之前称为 C++0x）的所选功能。IBM 将继续开发和实现此标准的功能部件。此语言级别的实现基于 IBM 对该标准的解释。在 IBM 完成所有 C++11 功能（包括支持新的 C++11 标准库）的实现之前，该实现可能会随发行版的不同而改变。IBM 在实现新的 C++11 功能时，不会尝试在源代码、二进制代码或列表以及其他编译器接口方面保持与先前发行版的兼容性。

XL C/C++ V11.1 中引入了以下功能:

- 自动类型推导
- C99 `long long`
- C++11 中采用的 C99 预处理器功能
- Decltype
- 委托构造函数

- 显式实例化声明
- 扩展的友元声明
- 内联名称空间定义
- 静态断言
- 可变参数模板

可以使用 **-qlanglvl=extended0x** 选项来启用大多数 C++ 功能和当前支持的所有 C++11 功能。有关详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qprefetch。

自动类型推导

使用自动类型推导功能，您不再需要在声明变量时指定类型。这是因为自动类型推导功能会根据自动变量的初始化方法表达式的类型将推导自动变量的类型这个任务委托给编译器。

可以使用 **-qlanglvl=autotypededuction** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 The auto type specifier (C++11)。

C99 `long long`

C++ 编译器可以使用 C99 `long long` 功能，此功能改进了 C 与 C++ 语言之间的源代码兼容性。

可以使用 **-qlanglvl=c99longlong** 选项来启用 C99 `long long` 功能。

IBM 在启用此功能后，如果十进制整数文字不具有包含 `u` 或 `U` 的后缀，从而无法以 `long long int` 类型来表示，那么您可以通过指定 `-qlanglvl=[no]extendedintegersafe` 选项来决定是否使用 `unsigned long long int` 类型来表示该文字。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Integer literals。

C++11 中采用的 C99 预处理器功能

借助于 C++11 中采用的若干 C99 预处理器功能，C 和 C++ 编译器提供了一个更通用的预处理器接口，它可以轻松将 C 源文件移植到 C++ 编译器，可以消除 C 与 C++ 预处理器之间的语义差别，并可以避免预处理器兼容性问题或相异的预处理器行为。

可以使用 **-qlanglvl=c99preprocessor** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 C99 preprocessor features adopted in C++11。

Decltype

借助于 decltype 功能，您可以获取一种类型，它基于可能取决于类型的表达式所产生的类型。

可以使用 **-qlanglvl=decltype** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 The decltype(expression) type specifier (C++11)。

委托构造函数

借助于委托构造函数功能，您可以在一个构造函数中构造公共初始化值，这使程序更具有可读性并更易于维护。

可以使用 **-qlanglvl=delegatingctors** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Delegating constructors (C++11)。

显式实例化声明

借助于显式实例化声明功能，您可以禁止对模板特例化或其成员进行隐式实例化。

可以使用单个子选项 **-qlanglvl=externtemplate** 或者组选项 **-qlanglvl=extended** 或 **-qlanglvl=extended0x** 来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Explicit instantiation (C++ only)。

扩展的友元声明

扩展的友元声明功能放宽了用于控制友元声明的语法规则，如下所示:

- 可以将模板参数、`typedef` 名称和基本类型声明为友元。
- 友元声明上下文中的类关键字在 C++11 中不再是必要的。

可以使用 **-qlanglvl=extendedfriend** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Friends (C++ only)。

内联名称空间定义

内联名称空间定义是具有初始 `inline` 关键字的名称空间定义。您可以定义或特例化内联名称空间的成员，就好像这些成员属于包含内联名称空间的外层名称空间。

可以使用 **-qlanglvl=inlinenamespace** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Inline namespace definitions (C++11)。

静态断言

静态断言功能具有下列优点:

- 库可以在编译时检测一般用法错误。
- C++ 标准库的实现可以检测和诊断一般用法错误，从而提高可用性。

可以使用 static_assert 声明在编译时检查重要的程序不变量。

可以使用 **-qlanglvl=static_assert** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 static_assert declaration (C++11)。

可变参数模板

借助于可变参数模板功能，您可以定义具有任何数目（其中包括零）的参数的类或函数模板。

可以使用 **-qlanglvl=variadic[templates]** 选项来启用此功能。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Variadic templates (C++11)。

XL C/C++ Compiler Reference **中的相关信息**


-qlanglvl

性能和优化

其他功能部件和增强功能可以帮助进行性能调整和应用程序优化。

对 -qpdf 的增强

对 **-qpdf** 选项的使用由两个步骤组成。首先，使用 **-qpdf1** 选项对程序进行编译，并通过一组典型数据来运行该程序以生成概要分析数据。然后，使用 **-qpdf2** 选项再次对程序进行编译，以根据概要分析数据对该程序进行优化。

在前发行版中，如果您修改源文件并使用 **-qpdf2** 选项进行编译，那么编译会停止并发生错误。从 XL C/C++ for Linux V11.1 开始，编译器将发出一系列警告，但编译不会停止。这使您可以在修改源文件之后继续使用概要分析数据。

对 **-qpdf** 选项增加了新的子选项。可以使用这些新的子选项对性能改进进行更多控制以及扩展 **-qpdf** 以支持多回合概要分析、高速缓存不命中概要分析和扩展的值概要分析。

新的 **-qpdf** 子选项如下:

level
支持多回合概要分析、单回合概要分析、高速缓存不命中概要分析、值概要分析、块计数器概要分析和调用计数器概要分析。可以使用 **-qpdf1=level=0|1|2** 对程序进行编译，以指定所产生的应用程序要生成的概要分析信息类型。

exename
根据 **-o** 选项所指定的输出文件的名称来指定所生成的 PDF 文件的名称。

defname
将 PDF 文件恢复至其缺省文件名称。

有关这些子选项的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qpdf1, -qpdf2。

有关编译器优化的报告

对列表报告的许多功能进行了增强，以便为您提供有关编译器如何优化代码的更多信息。可以使用此信息来从编译器的优化功能中获取更多的优点。有关这些所增强报告的更多详细信息，请参阅第 32 页的『新的诊断报告』。

与性能相关的编译器选项和伪指令

下表中的条目描述新增或已更改的编译器选项和伪指令。

此处提供的信息是一个简短的概述。有关这些选项、伪指令以及其他与性能相关的编译器选项的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″Optimization and tuning options″。

表 5. 与性能相关的编译器选项和伪指令

-qinline=level=*number*	对 -qinline 添加了一个新选项，以将相对于缺省值 5 的内联相对值告诉编译器。*number* 是范围 0 到 10 内的整数值，指示要使用的内联级别。有关详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qinline。
-qpdf	**-qpdf** 提供子选项，以让您更灵活地控制不同的 PDF 优化。有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qpdf1, -qpdf2 部分。
-qprefetch	给 **-qprefetch** 添加了新的增强功能，以便自动将预取指令插入到有可能提高代码性能的位置: `-qprefetch=assistthread`。有关详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 -qprefetch。

有关性能调整和程序优化的其他信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Optimizing your applications。

新的诊断报告

新的诊断报告可以帮助您确定改进代码性能的时机。

XML 格式的编译器报告

现在，可获取有关下列内容的 XML 格式的信息: 编译器之前能够执行的优化以及未执行的优化。此信息可用来在调整应用程序时减少编程工作量，对于高性能应用程序更是如此。

来自编译器的信息以 XML 1.0 格式生成。该报告通过 XML 模式进行定义，由您创建的用于读取和分析结果的工具能够轻易使用该报告。提供了一个样式表 `xlstyle.xsl` 以使报告显示为可读的格式，以便具有支持 XSLT 的浏览器的任何用户都可以读取。

在此发行版中，报告中提供了下列四个优化类别:

- 内联
- 循环变换
- 数据重组
- 概要定向反馈信息

可以使用新的 **-qlistfmt** 选项及其相关联子选项来生成新的 XML 1.0 报告。

有关此报告以及如何进行使用的详细信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 ″Using reports to diagnose optimization opportunities″。

对概要分析报告的增强

对列表文件增加了新节以帮助对程序进行分析。将 **-qreport** 选项与 **-qpdf2** 选项配合使用时，可以将下列各节添加至列表文件中标题为 PDF Report 的节中:

循环迭代计数

对于程序中大多数循环，会计算给定的一组输入数据的最频繁循环迭代计数和平均迭代计数。仅当在优化级别为 -O5 时编译该程序，此信息才可用。

块和调用计数

报告的此部分包括程序的调用结构以及每个所调用函数的相应执行次数。它还包括每个函数的块信息。对于非用户定义的函数，仅提供执行次数。块和调用覆盖总计数以及用户函数列表（按执行次数递减顺序排序）显示在此报告部分的末尾。此外，块计数信息显示在列表文件中每个伪码块的开头。

高速缓存不命中

报告的此部分显示在单个表中。它报告某些函数的高速缓存不命中数目以及有关这些函数的其他信息，例如: 高速缓存级别、高速缓存不命中率、行号、文件名和内存引用。

注: 必须使用 **-qpdf1=level=2** 选项来获取此报告。
还可通过在运行时期间使用 **PDF_PM_EVENT** 环境变量来选择要进行概要分析的高速缓存级别。

有关概要文件定向反馈的详细信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Using profile-directed feedback。

有关列表文件的其他信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler listings。

数据重组报告

编译器可以在列表文件中生成下列信息:

- 数据重组（有关编译器如何重组程序变量数据的摘要）
- 编译器插入的数据预取指令的位置

要生成数据重组信息，请随 **-qreport** 指定优化级别 **-qipa=level=2** 或 **-O5**。在 IPA 链接传递期间，程序变量数据的数据重组消息被添加至列表文件的数据重组部分（标签为 DATA REORGANIZATION SECTION）。重组包括以下各项:

- 数组分割
- 数组转置
- 内存分配合并
- 数组交叉
- 数组结合

要生成有关数据预取插入位置的信息，请使用优化级别 **-qhot** 或任何其他表示随 **-qreport** 使用 **-qhot** 的选项。此信息出现在列表文件的 LOOP TRANSFORMATION SECTION中。

其他循环分析

新的子选项已添加至 **-qhot** 以添加更为主动的循环分析。随 **-qsmp** 和 **-qreport** 使用 **-qhot=level=2** 会添加有关循环嵌套的信息，在这些嵌套循环上，已对列表文件的 LOOP TRANSFORMATION SECTION 执行主动的循环分析。此信息还可出现在使用 **-qlistfmt** 选项创建的 XML 列表文件中。

新增和增强的诊断选项

下表中的条目描述新增或已更改的编译器选项和伪指令，它们使您可以对编译器列表进行控制。

此处提供的信息是一个简短的概述。有关这些编译器选项以及其他与性能相关的编译器选项的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″Listings, messages and compiler information″。

表 6. 与列表相关的编译器选项和伪指令

选项/伪指令	描述
-qlistfmt	生成 XML 1.0 格式的报告，该报告包含有关下列方面的信息：编译器执行的优化以及未执行的优化。该报告包含有关内联、循环变换、数据重组和概要文件定向反馈的信息。
-qreport	当与 **-qpdf2** 配合使用时，列表现在包含 PDF 报告部分。当与 **-qipa=level=2** 或 **-O5** 配合使用时，列表文件中的另一个新部分是数据重组。
-qskipsrc	确定编译器跳过的源代码语句是否显示在列表文件的 SOURCE 部分中。

使用情况跟踪和报告工具

使用情况跟踪和报告功能是轻量级的简单机制，用于跟踪组织中的编译器使用情况。缺省情况下禁用此功能。可以使用此功能来检测组织对编译器的使用是否超过了编译器许可证权利。

当使用情况跟踪已启用时，对编译器的每次调用都会记录在编译器使用情况文件中。可以运行使用情况报告工具来根据一个或多个此类文件生成报告，以了解组织内编译器的总体使用情况。可以使用 **urt** 命令来控制生成报告的方式。报告会特别指出使用编译器的并发用户的数目。

使用情况跟踪和报告功能易于设置和管理，并且使用情况跟踪不会影响编译器的使用或性能。

有关使用情况跟踪和报告功能的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 Tracking and reporting compiler usage。

新增或已更改的编译器选项和伪指令

本节描述了 XL C/C++ V11.1 中新增和已更改的编译器选项和伪指令。

可以在命令行上指定编译器选项。还可以通过应用程序源文件中嵌入的编译指示伪指令来修改编译器行为。请参阅 *XL C/C++ Compiler Reference*，以获得这些编译器选项和其他编译器选项的详细描述和用法信息。

表 7. 新增或已更改的编译器选项和伪指令

选项或伪指令	描述
-qarch	已对 **-qarch** 添加新的子选项，指定 **-qarch=pwr7** 会生成对象代码，该对象代码包含在 POWER7 硬件平台上运行的指令。
-qassert	**-qassert** 是 XL C/C++ 的新选项。它用来提供有关可帮助微调优化的文件的特征的信息。
-qfunctrace	跟踪编译单元中函数或仅特定的函数列表的入口点和出口点。
-qhot	已对 **-qhot** 添加新的子选项。**-qhot** 编译器选项对于手工调优是很有用的选择，提供了优化循环和数组语言的时机。 仅当启用了 **-qstrict=nolibrary** 时，**-qhot=fastmath** 选项才允许用 XLOPT 库中的可用数学例程来替换数学例程。**-qhot=nofastmath** 会禁用由 XLOPT 库执行的数学例程替换。缺省情况下，如果指定了 -qhot，那么无论热级别如何，都会启用 -qhot=fastmath。
-qinline	尝试内联函数而不是生成对这些函数的调用，以提高性能。
-qkeepinlines	新的子选项 **exports** 已添加至 **-qkeepinlines** 选项。可使用 **-qkeepinlines=exports** 来确保编译器保存符号列表及其定义（来自通过编译器的较低版本编译的共享对象文件）。

表 7. 新增或已更改的编译器选项和伪指令 (续)

选项或伪指令	描述
-qlanglvl	C++11 新的子选项已添加至 **-qlanglvl**: • **-qlanglvl=autotypedededuction**: 控制是否启用自动类型推导功能。根据自动变量的初始化方法表达式的类型，此功能可用来将该变量的类型推导任务委派给编译器。 • **-qlanglvl=c99longlong**: 控制是否启用 C99 `long long` 功能。此功能用于提高 C 和 C++ 语言之间的源代码兼容性。 • **-qlanglvl=c99preprocessor**: 控制是否启用 C++11 中采用的 C99 预处理器功能。此功能可用来为 C 和 C++ 编译器提供更通用的预处理器接口。 • **-qlanglvl=decltype**: 控制是否启用 decltype 功能。此功能可用于获取基于可能取决于类型的表达式所产生的类型。 • **-qlanglvl=delegatingctors**: 控制是否启用委派构造函数功能。此功能可以用来在一个构造函数中集中进行公共初始化。 • **-qlanglvl=extendedfriend**: 控制是否启用扩展友元声明功能。此功能可以用来接受其他格式的非函数友元声明。 • IBM **-qlanglvl=extendedintegersafe**: 控制是否可将 `unsigned long long int` 用作十进制整数文字的类型，这些十进制整数文字不具有包含 `u` 或 `U` 的后缀并且无法以 `long long int` 类型表示。仅当指定了 **-qlanglvl=c99longlong** 选项时，此选项才生效。 • **-qlanglvl=externtemplate**: 控制是否启用显式实例化声明功能。此功能可用来禁止隐式实例化模板特例化或其成员。 • **-qlanglvl=inlinenamespace**: 控制是否启用内联名称空间定义功能。此功能可用来定义内联名称空间的成员并使其特例化，就如它们也是封闭名称空间的成员一样。 • **-qlanglvl=static_assert**: 控制是否启用静态断言功能。此功能可用来生成编译时断言，当发生故障时，会为这些断言发出严重错误消息。 • **-qlanglvl=variadic[templates]**: 控制是否启用可变参数模板功能。此功能可用来定义具有任何数目（其中包括零）的参数的类或函数模板。
-qlibmpi	根据消息传递接口 (MPI) 函数的已知行为对代码进行调整。
-qlistfmt	生成 XML 1.0 格式的报告，该报告包含有关下列方面的信息: 编译器执行的某些优化、内联时某些未执行的优化、循环变换、概要文件定向反馈和数据重组。
-qpdf1,-qpdf2	对 **-qpdf1,-qpdf2** 增加了新的子选项。

表 7. 新增或已更改的编译器选项和伪指令 (续)

选项或伪指令	描述
-qprefetch	已对 **-qprefetch** 添加新的子选项。当使用生成高速缓存不命中率高的应用程序时，可使用 **-qprefetch=assistthread** 来利用辅助线程以进行数据预取。
-qrestrict（仅限于 C）	可以使用 **-qrestrict** 将以下事实通知编译器：任何其他指针都不能访问函数参数指针已寻址的同一内存。
-qsaveopt|-qnosaveopt	增强了现有 **-qsaveopt** 选项，以便还包含用户的配置文件名称和配置文件中指定的选项。
-qstackprotect	使应用程序不包含覆盖或损坏堆栈的恶意代码或编程错误。
-qstaticlink	控制共享和非共享运行时库如何链接到应用程序中。
-qstrict	已对 **-qstrict** 选项添加新子选项，以允许对违反严格程序语义的优化和变换进行更多控制。 **-qstrict=vectorprecision** 禁用循环中的向量化，这可能会在向量化迭代中产生与非向量化迭代不同的结果。
-qtune	已对 **-qtune** 添加新的子选项。如果指定 **-qtune=pwr7**，那么会针对 POWER7 硬件平台对优化进行调整。

表 8. 建议不要使用的伪指令和选项

选项或伪指令	描述
-Q	建议不要使用此选项，已将它替换为 **-qinline**。
-qenablevmx	建议不要使用此选项，已将它替换为 **-qsimd=auto** 选项。
-qhot=simd | nosimd	建议不要使用 **-qhot=simd | nosimd**，在将来的发行版中可能将其除去。可以使用 **-qsimd**。
-qinfo=private	建议不要使用 **-qinfo=private**，已将它替换为 **-qreport**。
-qinfo=reduction	建议不要使用 **-qinfo=reduction**，已将它替换为 **-qreport**。
-qipa=inline | noinline	建议不要使用 **-qipa=inline | noinline**，在将来的发行版中可能将其除去。可以使用 **-qinline**。
-qipa=clonearch | noclonearch	**-qipa=clonearch | noclonearch** 不再受支持。可以使用 **-qtune=balanced**。
-qipa=clonearch | noclonearch	**-qipa=cloneproc | nocloneproc** 不再受支持。可以使用 **-qtune=balanced**。

内置函数

本节列示了 XL C/C++ V11.1 的新增内置函数。

有关 XL C/C++ 提供的内置函数的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler built-in functions。

VSX 内置函数

向量标量扩展 (VSX) 是 POWER7 处理器的新增功能。

有关 VSX 内置函数的更多信息，请参阅 Vector built-in functions。

POWER7 预取扩展和高速缓存控制

POWER7 处理器具有支持存储流预取和预取深度控制的高速缓存控制和流预取扩展功能。XL C/C++ 提供下列新的内置函数，使程序员可以直接访问这些指令:

- `__protected_stream_stride`
- `__transient_protected_stream_count_depth`
- `__unlimited_protected_stream_depth`
- `__transient_unlimited_protected_stream_depth`
- `__partial_dcbt`
- `__dcbtt`
- `__dcbtstt`
- `__dcbflp`

编译器可以在对代码进行优化时自动插入内置函数。您可以通过 **-qnoprefetch** 禁止自动使用这些指令。

有关这些伪指令的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 built-in functions。

POWER7 硬件内置函数

此发行版中新增了对应于每个新的 POWER7 硬件指令的 XL C/C++ 内置函数。借助这些函数，您可以在代码中直接处理特定的硬件指令，这可以提高应用程序的性能。

- `__bpermd`
- `__cbcdtd`
- `__cdtbcd`
- `__load8r`
- `__store8r`
- `__divde`
- `__divdeu`
- `__cmpb`
- `__divwe`
- `__divweu`
- `__addg6s`

转换函数

这些新函数在 Declets 与二进制编码的十进制数之间进行转换。

- `__cbcdtd`
- `__cdtbcd`

比较函数

此新函数对字节进行比较。

- __cmpb

十进制浮点函数

此新函数添加和生成由 6 个数字构成的组。

- __addg6s

V10.1 中新增的增强功能

本节描述添加到 V10.1 中的编译器的功能部件和增强功能。

C++11 功能

XL C/C++ V10.1 引进了对 C++ 编程语言（特别是 C++11，批准之前又称为 C++0x）的新版本标准的支持。在发行时不会正式采用该标准，但开始支持其部分功能。

注: IBM 支持 C++11（在批准之前称为 C++0x）的所选功能。IBM 将继续开发和实现此标准的功能部件。此语言级别的实现基于 IBM 对该标准的解释。在 IBM 完成所有 C++11 功能（包括支持新的 C++11 标准库）的实现之前，该实现可能会随发行版的不同而改变。IBM 在实现新的 C++11 功能时，不会尝试在源代码、二进制代码或列表以及其他编译器接口方面保持与先前发行版的兼容性。

具体地说，在此发行版中:

- 我们添加了新的语言级别
- 我们为与 long long 数据类型的算术转换引进了新的整数提升规则
- C++ 预处理器现在支持 C99 功能

新的语言级别 - extended0x

当调用 C++ 编译器时，缺省 **-qlanglvl** 编译器选项仍是 **extended**。

在此发行版中，新的子选项已添加至 **-qlanglvl** 选项。**-qlanglvl=extended0x** 用于允许用户对 C++11 的、当前受 XL C/C++ 支持的任何功能部件的早期实现进行试验。

C++ 环境下的 C99 long long

在使用 XL C/C++ 此发行版 V10.1 的情况下，当使用整数字面值数据类型执行某些算术运算时，编译器行为会改变。具体地说，整数提升规则已更改。

先前，在 C++（并且作为对 C89 的扩展）中，当使用 **-qlonglong** 进行编译时，无后缀整数将提升为它所适合的此列表中的第一个类型:

```
int
long int
unsigned long int
long long int
unsigned long long
```

从此发行版开始，当使用 **-qlanglvl=extended0x** 进行编译时，编译器立即将无后缀整数提升为它所适合的此列表中的第一个类型:

```
int
long int
long long int
unsigned long long
```

注: 与 C99 标准在 C 编译器中的实现相似，如果值不符合 `long long` 类型，但符合 `unsigned long long`，那么 C++ 将允许从 `long long` 提升至 `unsigned long long`。在此情况下，将生成一条消息。

已添加宏 `__C99_LLONG`，以便与 C99 兼容。在 **-qlanglvl=extended0x** 的情况下，此宏定义为 `1`；在其他情况下，则取消定义。

有关更多信息，请参阅 *XL C/C++ Language Reference* 中的 Integral and floating-point promotions。

预处理器更改

对 C++ 预处理器所作的下列更改使其更容易将代码从 C 移植至 C++:

- 常规字符串文字现在可与宽字符串文字并置。
- `#line <整数>` 预处理器伪指令具有较大的上限。对于 C++，它已经从 32767 增大至 2147483647。
- C++ 现在支持 `_Pragma` 运算符。
- 这些宏现在适用于 C++ 和 C:
  - __C99_MACRO_WITH_VA_ARGS（也可与 **-qlanglvl=extended** 配合使用）
  - __C99_MAX_LINE_NUMBER（也可与 **-qlanglvl=extended** 配合使用）
  - __C99_PRAGMA_OPERATOR
  - __C99_MIXED_STRING_CONCAT

注: 除另有说明之外，否则，仅当使用 **-qlanglvl=extended0x** 进行编译时，才提供这些 C++ 预处理器更改。

有关 XL C/C++ 支持的语言标准的其他信息，请参阅 *XL C/C++ Language Reference* 中的 Language levels and extensions。

与其他 XL C/C++ 语言相关的更新

本节描述了 V10.1 中引入的语言相关的更改。

向量数据类型

向量数据类型现在可以使用一些能与基本数据类型配合使用的运算符，例如:

- 一元运算符
- 二目运算符
- 关系运算符

OpenMP 3.0

IBM XL C/C++ for Linux V10.1 支持 OpenMP API V3.0 规范。 XL C/C++ 实现基于 IBM 对 OpenMP Application Program Interface Draft 3.0 Public Comment 的解释。

版本 2.5 与版本 3.0 之间的主要差别是:

- 增加任务级别并行化。新的 OpenMP 构造 TASK 和 TASKWAIT 使用户能够对非常规算法（例如，现有 OpenMP 构造无法处理的指针跟踪或递归算法）进行并行化。
- for 循环现在可以包含 unsigned int 和 pointer 类型以及 signed int 的 *var* 值。
- 堆栈大小控制。现在可以使用新的环境变量 OMP_STACKSIZE 控制由 OMP 运行时库所创建线程的堆栈大小。
- 用户可以使用新的环境变量 OMP_WAIT_POLICY 和 OMP_SET_POLICY 来暗示等待线程的预期行为。
- 存储器复用。已除去对 PRIVATE 子句的一些限制。出现在并行构造的 reduction 子句中的列表项现在也可以出现在工作共享构造上 private 子句中。
- 安排。新的 SCHEDULE 属性 auto 允许编译器和运行时系统控制安排。
- 具有 STATIC 时间表的连续循环构造现在可以使用 nowait。
- 嵌套支持 - COLLAPSE 子句已添加至 DO、FOR、PARALLEL FOR 和 PARALLEL DO 伪指令以允许并行化精确循环嵌套。这意味着可以使嵌套中的多个循环并行化。
- THREADPRIVATE 伪指令现在可适用于类作用域以及文件和块作用域的变量。
- 规范格式的迭代器循环（其中包括具有随机存取迭代器的循环）的并行化。

有关更多信息，请参阅:

- *XL C/C++ Optimization and Programming Guide* 中的 Using OpenMP directives
- www.openmp.org

性能和优化

XL C/C++ V10.1 包括一些功能和增强功能，用于帮助对应用程序进行性能调整和优化。

对 -qstrict 的增强

许多子选项已经被添加至 **-qstrict** 选项，以允许对违反严格程序语义的优化和变换进行更细颗粒度控制。在前发行版中，**-qstrict** 选项禁止所有违反严格程序语义的变换。如果您使用不带子选项的 **-qstrict**，那么它仍然会执行该行为。相似地，在前发行版中，**-qnostrict** 已允许那些可更改程序语义的变换。由于较高级别的优化可能需要放宽严格程序语义，所以添加子选项会放宽所选规则，以便获得特定的较快代码的优点，而无须关闭所有语义验证。

可以分别使用 16 个新子选项，也可以使用子选项组。以下是子选项组的列表:

all 禁用所有语义更改变换，其中包括由其他子选项控制的那些变换。

ieeefp 控制单独的操作是否符合 IEEE 754 语义。

order 控制是否可采用违反程序语言语义的方式对单个操作重新排序。

precision
控制可影响程序结果精度的优化和变换。

exceptions

控制可影响程序生成的运行时异常的优化和变换。

有关这些子选项的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″-qstrict″。

与性能相关的编译器选项和伪指令

下表中的条目描述新增或已更改的编译器选项和伪指令。

此处提供的信息是一个简短的概述。有关这些编译器选项以及其他与性能相关的编译器选项的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″Optimization and tuning options″。

表 9. 与性能相关的编译器选项和伪指令

选项/伪指令	描述
-qstrict	已经添加许多新的子选项，以便您能够对放宽的程序语义规则进行更多控制，从而获得一些性能益处。
-qfloat	一些 **-qfloat** 子选项受 **-qstrict** 的新子选项影响。
-qreport	列表现在包含有关为每个循环创建多少个流以及哪些循环由于 non-stride-one 引用而无法 SIMD 向量化的信息。您可以使用此信息来提高应用程序的性能。
-qsmp	当 **-qsmp=omp** 生效时，OpenMP API 3.0 的其他功能现在可用。有关更多信息，请参阅 第 41 页的『OpenMP 3.0』。

有关性能调整和程序优化的其他信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Optimizing your applications。

编译器选项和编译指示伪指令

本节描述了 XL C/C++ V10.1 中新增和已更改的编译器选项和伪指令。

可以在命令行上指定编译器选项。还可以通过应用程序源文件中嵌入的编译指示伪指令来修改编译器行为。请参阅 *XL C/C++ Compiler Reference*，以获得这些编译器选项和其他编译器选项的详细描述和用法信息。

表 10. 新增或已更改的编译器选项和伪指令

选项或伪指令	描述
-qstrict	许多子选项已经被添加至 **-qstrict** 选项，以允许对违反严格程序语义的优化和变换进行更多控制。有关更多信息，请参阅第 31 页的『性能和优化』。
-qshowmacros	当与 **-E** 选项一起使用时，**-qshowmacros** 选项会用宏定义替换预处理的输出。提供了子选项用于更准确地控制预定义宏和用户定义的宏的发出。
-qreport	当与启用自动并行化或向量化的编译器选项配合使用时，**-qreport** 选项现在报告循环中的流数量，并且在循环由于 non-stride-one 引用而无法 SIMD 向量化时生成信息。
-qsmp	当 **-qsmp=omp** 生效时，OpenMP API 3.0 的其他功能现在可用。有关更多信息，请参阅 第 41 页的『OpenMP 3.0』。
-qtimestamps	此选项可用于从已生成的二元中除去时间戳记。

表 10. 新增或已更改的编译器选项和伪指令 (续)

选项或伪指令	描述
-qtls	线程本地存储器支持已经增强为包括 `__attribute__((tls-model ("string")))`，其中 *string* 是 `local-exec`、`initial-exec`、`local-dynamic` 或 `global-dynamic`。
-qinfo	子选项 `als` 和 `noals` 已添加至 **qinfo** 选项以报告（或不报告）ANSI 别名判别规则的可能违例。

预定义宏

本节提供了有关此发行版中添加的预定义宏的相关信息。

在 XL C/C++ V10.1 中添加了四个新宏:

_ILP32 __ILP32__
仅当对一个目标进行编译时，才可定义为 1，其中该目标的 `long int`，`int` 和指针使用的都是 32 位。否则，不能对它定义。

_LP64 __LP64__
仅当对一个目标进行编译时，才可定义为 1，其中该目标的 `long int` 和指针使用的都是 64 位而 `int` 使用的是 32 位。否则，不能对它定义。

编译器不再支持 `__C99_COMPLEX_HEADER__` 宏。

要获取 XL C/C++ 预定义宏的完整列表，请参阅 *XL C/C++ Compiler Reference* 中的 Compiler predefined macros。

第 4 章 设置和定制 XL C/C++

有关 XL C/C++ 的完整先决条件和安装信息，请参阅 *XL C/C++ Installation Guide* 中的 ″Before installing XL C/C++″。

使用定制编译器配置文件

您可以通过修改缺省配置文件或创建自己的配置文件来定制编译器设置和选项。.

可以进行下列选择来定制编译器设置:

- XL C/C++ 编译器安装进程会创建缺省编译器配置文件。可以直接修改此配置文件，以根据特定需要添加缺省选项。但是，如果稍后对该编译器应用更新，那么必须将所有修改重新应用于新安装的配置文件。
- 可以创建您自己的定制配置文件，用于覆盖或补充缺省配置文件。编译器可识别和解析您在定制配置文件以及缺省配置文件中指定的编译器设置。 编译器更新（以后可能会影响缺省配置文件中的设置）不会影响定制配置文件中的设置。

有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Using custom compiler configuration files。

配置编译器使用情况跟踪和报告

除了编译器配置文件，对于使用情况跟踪和报告功能，还有一个单独的配置文件。缺省情况下，会禁用使用情况跟踪，但可以通过在此配置文件中修改条目来启用该功能。通过使用此文件，还可配置使用情况跟踪的其他各个方面。

虽然编译器配置文件不同于使用情况跟踪配置文件，但是它包含指定使用情况跟踪配置文件的位置的条目，以便编译器可找到此文件。

有关如何配置使用情况跟踪和报告功能的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 Tracking and reporting compiler usage。

© Copyright IBM Corp. 1996, 2014

第 5 章 使用 XL C/C++ 开发应用程序

C/C++ 应用程序开发就是编辑、编译、链接和运行过程的重复循环。缺省情况下，编译和链接将合并为一个步骤。

注意事项:

1. 必须先确保正确安装和配置 XL C/C++，然后才能使用编译器。有关更多信息，请参阅 *XL C/C++ Installation Guide*。
2. 要了解有关编写 C/C++ 程序的信息，请参阅 *XL C/C++ Language Reference*。

编译器阶段

典型的编译器调用按顺序执行下列某些或全部活动。为了获得链接时优化，将在编译期间多次执行某些活动。当每个编译组件运行时，结果按顺序发送至下一个步骤。

1. 预处理源文件
2. 编译，根据所指定的编译器选项，它可能包含下列阶段:
   a. 前端语法分析和语义分析
   b. 高级优化
   c. 低级优化
   d. 寄存器分配
   e. 最终的汇编
3. 在预处理组合件（**.s**）文件和未预处理的汇编程序（**.S**）文件之后对它们进行汇编
4. 对象链接以创建可执行的应用程序

要查看编译器如何单步执行这些阶段，请在编译应用程序时指定 **-v** 编译器选项。要查看编译器在每个阶段所花费的时间量，请指定 **-qphsinfo**。

编辑 C/C++ 源文件

要创建 C/C++ 源程序，您可以使用任何系统可用的文本编辑器。

源程序必须使用被认可的文件名后缀来保存。请参阅 第 49 页的『XL C/C++ 输入和输出文件』，以获得 XL C/C++ 可识别的后缀列表。

要使 C 或 C++ 源程序成为有效的程序，它必须符合在 *XL C/C++ Language Reference* 中指定的语言定义。

使用 XL C/C++ 进行编译

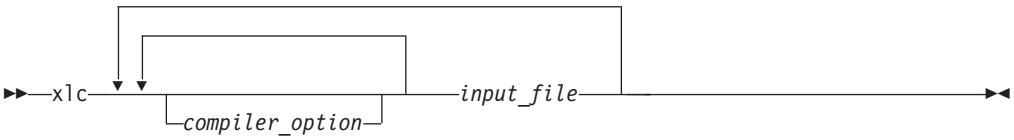
XL C/C++ 是命令行编译器。您可以根据特定 C/C++ 应用程序的需要选择调用命令和选项。

© Copyright IBM Corp. 1996, 2014

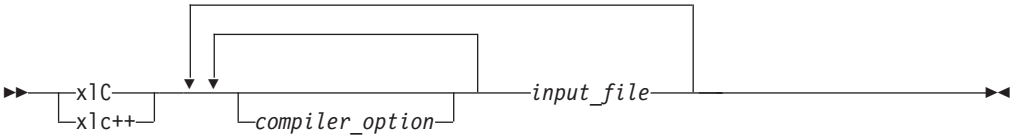
调用编译器

编译器调用命令执行所有必要的步骤来编译 C 或 C++ 源文件、汇编任何 **.s** 和 **.S** 文件并将对象文件和库链接至可执行程序。

要编译 C 源程序，请使用以下基本调用语法:



要编译 C++ 源程序，请使用以下基本调用语法:



对于大多数应用程序，应使用 **xlc**、**xlc++**、 或线程安全等效项进行编译。您可以使用 **xlc++** 来编译 C 或 C++ 程序源代码，但是，如果使用 **xlc** 来编译 C++ 文件，那么可能导致链接或运行时错误，这是因为当 C 编译器调用链接程序时不会指定 C++ 代码所需要的库。

另外，还提供了其他调用命令以满足专门的编译需要，目的主要是为 C 或 C++ 语言的不同级别和扩展提供显式编译支持。请参阅 *XL C/C++ Compiler Reference* 中的 中的 ″Invoking the compiler″，以获取有关可供您使用的编译器调用命令的更多信息，其中包括旨在帮助开发人员从 GNU 编译环境迁移到 XL C/C++ 的特殊调用。

编译并行化 XL C/C++ 应用程序

XL C/C++ 提供线程安全编译器调用命令以编译要用于多处理器环境中的并行应用程序。

这些调用类似于其相应的基本编译器调用，只是它们会将已编译的对象链接和绑定至线程安全组件和库。一般 XL C/C++ 线程安全编译器调用如下所示:

- xlC_r
- xlc++_r
- xlc_r

XL C/C++ 提供其他线程安全调用来满足特定的编译需求。有关更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″Invoking the compiler″。

注: 单独使用这些命令中的任何一个并不表示并行化。为了编译器能够识别 OpenMP 伪指令并激活并行化，您还必须指定 **-qsmp** 编译器选项。相应地，您仅应该将 **-qsmp** 选项与这些线程安全调用命令的其中一个一起指定。当您指定 **-qsmp** 时，驱动程序链接将指向配置文件的活动节中 smp 库行上所指定的库。

有关并行应用程序的更多信息，请参阅 ″Parallelizing your programs″。

指定编译器选项

编译器选项执行各种功能，如设置编译器特征、描述要生成的对象代码、控制发出的诊断消息和执行某些预处理器功能。

您可以采用以下方法的其中一种或任何组合指定编译器选项:

- 在命令行上使用命令行编译器选项
- 在源代码中使用伪指令语句
- 在 makefile 中
- 在编译器配置文件内找到的节中

还可以将编译器选项传递至链接程序、汇编程序和预处理器。

有关编译器选项和及其用法的更多信息，请参阅 ″Compiler options reference″ 。

编译器选项的优先级顺序

当指定了多个编译器选项时，有可能发生选项冲突和不兼容。为了以一致的方式解决这些冲突，编译器对大多数选项通常应用以下一般优先级顺序:

1. 源文件中的伪指令语句*覆盖*命令行设置
2. 命令行编译器选项设置*覆盖*配置文件设置
3. 配置文件设置*覆盖*缺省设置

通常，如果调用编译器时在命令行上多次指定了相同的编译器选项，那么最后指定的那个选项起作用。

注: 某些编译器选项（例如，**-l** 选项）并不按照以上所述的优先级顺序。编译器在搜索命令行上用 **-l** 指定的目录之前，首先搜索在 xlc.cfg 文件中用 **-l** 指定的任何目录。**-l** 选项是累积的而不是优先的。具有累积行为的其他选项是 **-R** 和 **-l**（小写 L）。

通过 gxlc 和 gxlc++ 来复用 GNU C/C++ 编译器选项

XL C/C++ 包括各种帮助您从 GNU C/C++ 编译器过渡到 XL C/C++ 的功能，其中包括 gxlc 和 gxlc++ 命令。

每个 gxlc 和 gxlc++ 实用程序都接受 GNU C 或 C++ 编译器选项并将它们转换成相当的 XL C/C++ 选项。这两个实用程序都使用 XL C/C++ 选项来创建 **xlc** 或 **xlc++** 调用命令，然后使用该命令来调用编译器。提供这些实用程序是为了帮助您复用为先前用 GNU C/C++ 开发的应用程序创建的 makefile。然而，为了充分利用 XL C/C++ 的功能，您可使用 XL C/C++ 调用命令及其相关选项。

gxlc 和 gxlc++ 的操作由 `gxlc.cfg` 配置文件控制。此文件中显示具有 XL C/C++ 对应选项的 GNU C/C++ 选项。并非每个 GNU 选项都具有对应的 XL C/C++ 选项。gxlc 和 gxlc++ 会对未转换的输入选项返回警告。

gxlc 和 gxlc++ 选项映射是可修改的。有关使用 gxlc 或 gxlc++ 配置文件的信息，请参阅 *XL C/C++ Compiler Reference* 中的 Reusing GNU C/C++ compiler options with gxlc and gxlc++ 。

XL C/C++ 输入和输出文件

XL C/C++ 可识别这些文件类型。

有关这些文件类型和编译器使用的其他文件类型的详细信息，请参阅 *XL C/C++ Compiler Reference* 中的 Types of input files 和 *XL C/C++ Compiler Reference* 中的 Types of output files 。

表 11. 输入文件类型

文件扩展名	描述
.c	C 源文件
.C、.cc、.cp、.cpp、.cxx 和 .c++	C++ 源文件
.i	预处理的源文件
.o	对象文件
.s	汇编程序文件
.S	未预处理的汇编程序文件
.so	共享对象或库文件

表 12. 输出文件类型

文件扩展名	描述
a.out	用于编译器创建的可执行文件的缺省名称
.d	生成依赖性文件
.i	预处理的源文件
.lst	列表文件
.o	对象文件
.s	汇编程序文件
.so	共享对象或库文件

将已编译的应用程序与 XL C/C++ 链接在一起

缺省情况下，不需要执行任何特殊操作来链接 XL C/C++ 程序。编译器调用命令自动调用链接程序来生成可执行输出文件。

例如，可使用以下命令来编译 `file1.C` 和 `file3.C` 以生成对象文件 `file1.o` 和 `file3.o`。所有对象文件（其中包括 `file2.o`）都将提交给链接程序以生成一个可执行文件。

```
xlc++ file1.C file2.o file3.C
```

分步进行编译和链接

要生成稍后可以链接的对象文件，使用 **-c** 选项。

```
xlc++ -c file1.C              # Produce one object file (file1.o)
xlc++ -c file2.C file3.C      # Or multiple object files (file1.o, file3.o)
xlc++ file1.o file2.o file3.o  # Link object files with default libraries
```

有关编译和链接程序的更多信息，请参阅:

- *XL C/C++ Compiler Reference* 中的 "Linking"
- *XL C/C++ Optimization and Programming Guide* 中的 "Constructing a library"

动态和静态链接

XL C/C++ 允许程序利用操作系统工具进行动态和静态链接。

动态链接意味着在程序首次运行时找到并装入某些外部例程的代码。当编译使用共享库的程序时，缺省情况下，共享库自动链接至程序。如果多个程序使用共享库中的例程，那么动态链接的程序将占用较少的磁盘空间和虚拟内存。在链接期间，它们不需要任何特殊的预防措施来避免与库例程的命名冲突。如果几个程序同时使用相同的共享例程，那么它们的性能旨在超过静态链接的程序。通过使用动态链接，您可以升级共享库中的例程而不需要重新链接。

因为该链接形式是缺省值，所以您不需要其他选项来启用该链接。

静态链接意味着由程序调用的所有例程的代码成为可执行文件的一部分。

可以将静态链接的程序移至系统并在系统上运行它们，而不需要 XL C/C++ 运行时库。如果它们执行许多对库例程的调用或调用许多小例程，那么它们的性能可能超过动态链接的程序。如果您要避免与库例程产生命名冲突，那么在为程序中的数据对象和例程选择名称时它们确实需要一些预防措施。如果分别在不同级别的操作系统编译和运行它们，那么它们也可能不生效。

运行已编译的应用程序

在对程序进行编译和链接之后，可以在命令行上运行生成的可执行文件。

由 XL C/C++ 编译器生成的程序可执行文件的缺省文件名是 **a.out**。可以使用 **-o** 编译器选项来选择另一名称。

因为您可能偶尔会执行错误的命令，所以应避免对程序可执行文件指定与系统命令或 shell 命令相同的名称（如 `test` 或 `cp`）。如果您决定使用与系统命令或 shell 命令相同的名称来命名程序可执行文件，那么应该通过指定该可执行文件所在目录的路径名（如 `./test`）来执行程序。

要运行程序，请在命令行上输入程序可执行文件的名称和任何运行时自变量。

取消执行

要暂挂正在运行的程序，当程序处于前台时按 **Ctrl+Z** 键。使用 **fg** 命令来恢复运行。

要取消正在运行的程序，当程序处于前台时按 **Ctrl+C** 键。

设置运行时选项

您可以使用环境变量设置，控制某些运行时选项以及使用 XL C/C++ 编译器所创建的应用程序的行为。某些环境变量不控制实际的运行时行为，但可以影响应用程序的运行方式。

有关环境变量和它们在运行时如何影响应用程序的更多信息，请参阅*XL C/C++ Installation Guide*。

在其他系统上运行已编译的应用程序

如果要在没有安装编译器的另一个系统上运行用 XL C/C++ 编译器开发的应用程序，那么您将需要在该系统上安装运行时环境或者以静态方式链接应用程序。

您可以从 XL C/C++ for Linux 支持页面获取最新的 XL C/C++ Runtime Environment PTF 图像以及许可发放和使用信息。

XL C/C++ 编译器诊断辅助

当编译应用程序遇到问题时，XL C/C++ 会发出诊断消息。您可以使用这些消息和编译器输出列表中提供的其他信息来帮助识别和更正这类问题。

有关可帮助解决应用程序问题的列表、诊断和相关编译器选项的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的下列主题:

- Compiler messages and listings
- Error checking and debugging options
- Listings, messages, and compiler information options

调试已编译的应用程序

您可以使用符号调试器来调试采用 XL C/C++ 编译的应用程序。

在编译时，您可使用 **-g** 或 **-qlinedebug** 选项以指示 XL C/C++ 编译器把调试信息包括在已编译的输出中。对于 **-g**，还可以使用其他级别来实现调试功能与编译器优化之间的平衡。有关调试选项的更多信息，请参阅 *XL C/C++ Compiler Reference* 中的 ″Error checking and debugging″。

然后，可以使用 **gdb** 或其他任何符号调试器，以单步调试和检查已编译的应用程序的行为。

已优化的应用程序在调试时，将会给您带来一些特殊的挑战。当调试高度优化的应用程序时，您应该考虑使用 **-qoptdebug** 编译器选项。有关优化代码的更多信息，请参阅 *XL C/C++ Optimization and Programming Guide* 中的 Optimizing your applications。

确定使用哪一个级别的 XL C/C++

要显示正在使用的 XL C/C++ 的版本和发行版级别，请使用 **-qversion** 编译器选项调用编译器。

例如，要获取详细的版本信息，请输入以下命令:

```
xlc++-qversion=verbose
```

声明

本信息是为在美国提供的产品和服务编写的。IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务信息，请向您当地 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

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

有关双字节（DBCS）信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: INTERNATIONAL BUSINESS MACHINES CORPORATION“按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证，因此本条款可能不适用于您。

本信息中可能包含有技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本资料中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的: （i）允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及（ii）允许对已经交换的信息进行相互使用，请与下列地址联系:

© Copyright IBM Corp. 1996, 2014

Lab Director
IBM Canada Ltd. Laboratory
8200 Warden Avenue
Markham, Ontario L6G 1C7
Canada

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息包含在日常业务操作中使用的数据和报告的示例。为了尽可能全面地作举例说明，这些示例包含个人、公司、品牌和产品的名称。所有这些名称都是虚构的，与实际商业企业所用的名称和地址的任何雷同纯属巧合。

版权许可:

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。为了开发、使用、营销或分发符合此操作平台（编写样本程序的操作平台）的应用程序编程接口的应用程序，您可以用任何方式复制、修改和分发这些样本程序，而不必向 IBM 公司付款。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。用户如果是为了按照 IBM 应用程序编程接口开发、使用、经销或分发应用程序，则可以任何形式复制、修改和分发这些样本程序，而无须向 IBM 付费。

凡这些样本程序的每份拷贝或其任何部分或任何衍生产品，都必须包括如下版权声明:

©（贵公司的名称）（年）。此部分代码是根据 IBM 公司的样本程序衍生出来的。© Copyright IBM Corp. 1998, 2014 年.

此软件和文档在部分程度上基于加州大学董事会颁发许可的 Fourth Berkeley 软件分发。我们感谢下列机构在此产品的开发中所发挥的巨大作用: Berkeley 校区的电子工程系和计算机科学系。

商标和服务标记

IBM、IBM 徽标和 ibm.com 是 International Business Machines Corp. 在全球许多管辖区域注册的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公司的商标。Web 站点 “Copyright and trademark information” (http://www.ibm.com/legal/copytrade.shtml) 提供了 IBM 商标的最新列表。

Adobe 和 Adobe 徽标是 Adobe Systems Incorporated 在美国和/或其他国家或地区的注册商标或商标。

Linux 是 Linus Torvalds 在美国和/或其他国家或地区的注册商标。

UNIX 是 Open Group 在美国和其他国家或地区的注册商标。

索引

[B]

编辑源文件 47
编译
  活动顺序 47
  SMP 程序 48
编译器
  调用 48
  控制行为 49
  运行 48
编译器伪指令
  新增或已更改 16, 25, 35
编译器选项
  冲突和不兼容 49
  规范方法 49
  新增或已更改 16, 25, 35
并行化 7, 41

[C]

程序
  运行 51

[D]

代码优化 6
调试 52
调试器支持 52
  符号 8
  输出列表 52
调试信息，生成 52
调试已编译的应用程序 52
调用编译器 48
调用程序 51
调用命令 48
定制
  与 GNU 兼容 3
动态链接 51
对象文件 50
  创建 50
  链接 50
多处理器系统 7, 41

[F]

符号调试器支持 8

[G]

工具 5
  配置文件实用程序 5
  新安装配置实用程序 5
  cleanpdf 实用程序 5
  gxlc 和 gxlc++ 实用程序 5
  mergepdf 实用程序 6
  new_install 实用程序 5
  resetpdf 实用程序 6
  showpdf 实用程序 6
  xlc_configure 5
共享对象文件 50
共享内存并行化 7, 41
归档文件 50

[H]

汇编程序
  源（.S）文件 50
  源（.s）文件 50

[J]

基本示例，描述的 viii
静态链接 51

[K]

可执行文件 50
库 50

[L]

链接
  动态 51
  静态 51
链接过程 50
列表 50

[N]

内置函数 12, 25, 37

[Q]

迁移 19
  源代码 49

[S]

实用程序 5
  cleanpdf 5
  gxlc 和 gxlc++ 5
  mergepdf 6
  new_install 5
  resetpdf 6
  showpdf 6
  xlc_configure 5
输出文件 50
输入文件 50

[W]

文件
  编辑源代码 47
  输出 50
  输入 50
问题确定 52

[X]

性能 18
  优化变换 6

[Y]

优化 18
  程序 6
语言标准 2
语言支持 2
源级别调试支持 8
源文件 50
运行编译器 48
运行时
  库 50
运行时环境 52
运行时选项 51

[Z]

执行程序 51
执行链接程序 50

[数字]

64 位环境 7

© Copyright IBM Corp. 1996, 2014

C

C11 11
C1X 11
  _Static_assert 21
C++11 28
  跟踪返回类型 19
  静态断言 28
  可变参数模板 28
  扩展的友元声明 28
  内联名称空间定义 28
  缺省函数和已删除的函数 10
  委托构造函数 28
  显式实例化声明 28
  显式转换运算符 19
  引用折叠 19
  有作用域的枚举 19
  右值引用 19
  自动类型推导 28
  C99 long long 28
  C++11 中采用的 C99 预处理器功能 28
  decltype 28

G

GNU
  兼容性 3

M

mod 文件 50

O

OMP 伪指令 41
OpenMP 7

S

SMP
  程序，编译 48
SMP 程序 7

X

XL C/C++ 的级别，确定 52
xlc.cfg 文件 49
xlc_configure 5

[特别字符]

.a 文件 50
.c 和 .C 文件 50
.i 文件 50
.lst 文件 50
.mod 文件 50
.o 文件 50
.S 文件 50
.s 文件 50
.so 文件 50

程序号: 5765-J08; 5725-C73

Printed in China

S151-2064-00