新增功能

Sun™ Studio 10

Sun Microsystems, Inc.
www.sun.com

文件号码 819-1597-10
2005 年 1 月，修订版 A

请将有关本文档的意见和建议提交至： http://www.sun.com/hwdocs/feedback

请回收

Adobe PostScript

目录

开始之前

*《新增功能》*介绍了 Sun Studio 10 软件发行版本和 Sun™ Studio 9 软件发行版本的新增功能，其中包括 C、 C++ 和 Fortran 编译器、库以及工具中的新增功能。

印刷约定

表 P-1 字样约定

字样	含义	示例
`AaBbCc123`	命令、文件和目录的名称；计算机屏幕输出	编辑您的 `.login` 文件。 使用 `ls -a` 列出所有文件。 `% You have mail.`
`AaBbCc123`	输入的内容，以便与计算机屏幕输出相区别	`% su` `Password:`
AaBbCc123	书名、新词或术语以及要强调的词	请阅读 *《用户指南》*的第 6 章。 这些称作*类*选项。 您*必须*是超级用户才能执行此操作。
AaBbCc123	命令行占位符文本；用实际名称或值替换	要删除文件，请键入 **`rm`** *filename*。

表 **P-2** 代码约定

代码符号	含义	表示法	代码示例
[ ]	括号包含可选参数。	O[*n*]	-O4, -O
{ }	大括号包含所需选项的选项集合。	d{y|n}	-dy
|	分隔变量的“|”或“-”符号，只能选择其一。	B{dynamic|static}	-Bstatic
:	与逗号一样，冒号有时可用于分隔参数。	R*dir*[:*dir*]	-R/local/libs:/U/a
…	省略号表示一系列省略。	-xinline=*f1*[,…*fn*]	-xinline=alpha,dos

Shell 提示符

Shell	提示符
C shell	*machine-name*%
C shell 超级用户	*machine-name*#
Bourne shell 和 Korn shell	$
Bourne shell 和 Korn shell 的超级用户	#

受支持的平台

此 Sun Studio 发行版本支持使用如下 SPARC® 和 x86 系列处理器架构的系统：UltraSPARC®、SPARC64、AMD64、Pentium 和 Xeon EM64T。对于您正在使用的 Solaris 操作系统版本所支持的系统，可从位于下列位置的硬件兼容性列表中获得：`http://www.sun.com/bigadmin/hcl`。这些文档中给出了平台类型间所有实现的区别。

在本文档中，术语“x86”指采用兼容 AMD64 或 Intel Xeon/Pentium 产品系列的处理器的 64 位和 32 位系统。有关受支持的系统，请参阅硬件兼容性列表。

访问 Sun Studio 软件和手册页

Sun Studio 软件及其手册页没有安装到标准的 /usr/bin/ 和 /usr/share/man 目录中。要访问软件，必须正确设置 PATH 环境变量 （请参阅 第 7 页 “访问软件”）。要访问手册页，必须正确设置 MANPATH 环境变量 （请参阅第 8 页 “访问手册页”）。

关于 PATH 变量的更多信息，请参阅 csh(1)、sh(1)、ksh(1) 和 bash(1)) 手册页。关于 MANPATH 变量的更多信息，请参阅 man(1) 手册页。关于设置 PATH 变量和 MANPATH 变量以访问此发行版本的更多信息，请参阅安装指南或询问系统管理员。

注 – 本部分的信息假设 Sun Studio 软件分别安装在 Solaris 平台和 Linux 平台上的 /opt 目录和 /opt/sun 目录中。如果软件没有安装在缺省目录中，请询问系统管理员以获取系统中的等效路径。

访问软件

使用下列步骤来决定是否需要更改 PATH 变量以访问软件。

要决定是否需要设置 PATH 环境变量

1. **通过在命令提示符后输入下列内容以显示 PATH 变量的当前值。**

```
% echo $PATH
```

2. **在 Solaris 平台上，查看输出中是否有包含 /opt/SUNWspro/bin 的路径字符串。在 Linux 平台上，查看输出中是否有包含 /opt/sun/sunstudio10/bin 的字符串路径。**

   如果找到该路径，您的 PATH 变量已经设置好，可以访问编译器和工具了。如果没有找到该路径，按照下一步中的说明来设置 PATH 环境变量。

要设置 PATH 环境变量以访问编译器和工具

- **在 Solaris 平台上，将下列内容增加到 PATH 环境变量。如果已安装 Forte Developer 软件、 Sun ONE Studio 软件或 Sun Studio 软件的其他发行版本，则将以下路径增加到这些安装的路径之前。**

  /opt/SUNWspro/bin

- **在 Linux 平台上，将下列内容增加到 PATH 环境变量。**
  `/opt/sun/sunstudio10/bin`

访问手册页

使用下列步骤来决定是否需要更改 MANPATH 变量以访问手册页。

要决定是否需要设置 MANPATH 环境变量

1. **通过在命令提示符后输入下列内容以请求 dbx 手册页。**

```
% man dbx
```

2. **如果有输出的话，请查看输出。**

   如果 dbx(1) 手册页无法找到或者显示的手册页不是用于安装软件的当前版本，请按照下一步中的说明来设置 MANPATH 环境变量。

要设置 MANPATH 环境变量以访问手册页

- **在 Solaris 平台上，将下列内容增加到 MANPATH 环境变量。**
  `/opt/SUNWspro/`
- **在 Linux 平台上，将下列内容增加到 MANPATH 环境变量。**

  `/opt/sun/sunstudio10`

访问集成开发环境

Sun Studio 9 集成开发环境 (IDE) 提供了创建、编辑、生成、调试和分析 C、 C++ 或 Fortran 应用程序性能的模块。

启动 IDE 的命令是 sunstudio。有关该命令的详细信息，请参阅 sunstudio(1) 手册页。

IDE 是否能够正确操作取决于 IDE 能否找到核心平台。 sunstudio 命令会在以下两个位置查找核心平台：

- 该命令首先分别查找 Solaris 平台和 Linux 平台上的缺省安装目录 /opt/netbeans/3.5V 和 /opt/sun/netbeans/3.5V。

- 如果该命令在缺省目录未找到核心平台，则它将假设包含 IDE 的目录和包含核心平台的目录均安装在同一位置上。例如，在 Solaris 平台上，如果包含 IDE 的目录的路径是 `/foo/SUNWspro`，则该命令将在 `/foo/netbeans/3.5V` 中查找核心平台。在 Linux 平台上，如果包含 IDE 的目录的路径是 `/foo/sunstudio10`，则该命令将在 `/foo/netbeans/3.5V` 中查找核心平台。

如果核心平台未安装在 `sunstudio` 命令查找它的任一位置上，则客户端系统上的每个用户必须将环境变量 `SPRO_NETBEANS_HOME` 设置为安装核心平台的位置 (*/installation_directory*`/netbeans/3.5V`)。

在 Solaris 平台上，IDE 的每个用户也必须将 */installation_directory*`/SUNWspro/bin` 增加到其他任何 Forte Developer 软件、Sun ONE Studio 软件或 Sun Studio 软件发行版本前面的 `$PATH` 中。在 Linux 平台上， IDE 的每个用户也必须将 */installation_directory*`/sunstudio10/bin` 增加到其他任何 Sun Studio 软件发行版本前面的 `$PATH` 中。

路径 */installation_directory*`/netbeans/3.5V/bin` 不可增加到用户的 `$PATH` 中。

访问 Sun Studio 文档

您可以访问下列位置的文档：

- 可以通过在本地系统或网络上随软件一起安装的文档索引获取文档，在 Solaris 平台和 Linux 平台上的位置分别为 `file:/opt/SUNWspro/docs/index.html` 和 `file:/opt/sun/sunstudio10/docs/index.html`。

  如果软件未安装在 Solaris 平台上的 `/opt` 目录中或 Linux 平台上的 `/opt/sun` 目录中，请询问系统管理员以获取系统中的等效路径。

- 大多数的手册都可以从 `docs.sun.com`sm Web 站点上获得。下列书目只能从您所安装的软件中找到：
  - *《标准 C++ 库类参考》*
  - *《标准 C++ 库用户指南》*
  - *《Tools.h++ 类库参考》*
  - *《Tools.h++ 用户指南》*
- 发行说明可以从 `docs.sun.com` Web 站点上获得。
- 在 IDE 中通过 “帮助”菜单或窗口和对话框上的 “帮助”按钮可以访问 IDE 所有组件的联机帮助。

您可以通过 Internet 在 `docs.sun.com` Web 站点 (`http://docs.sun.com`) 上阅读、打印和购买 Sun Microsystems 的各种手册。如果找不到手册，请参阅和软件一起安装在本地系统或网络中的文档索引。

注 – Sun 对本文档中提到的第三方 Web 站点的可用性不承担任何责任。对于此类站点或资源中的（或通过它们获得的）任何内容、广告、产品或其他材料， Sun 并不表示认可，也不承担任何责任。对于因使用或依靠此类站点或资源中的（或通过它们获得的）任何内容、产品或服务而造成的或连带产生的实际或名义损坏或损失， Sun 概不负责，也不承担任何责任。

使用易读格式的文档

该文档以易读格式提供，以方便残障用户使用辅助技术进行阅读。您还可以按照下表所描述的信息找到文档的易读版本。如果软件没有安装在 `/opt` 目录中，请询问系统管理员以获取系统中的等效路径。

文档类型	易读版本的格式和位置
手册 （第三方手册除外）	HTML，位于 `http://docs.sun.com`
第三方手册： • *《标准 C++ 库类参考》* • *《标准 C++ 库用户指南》* • *《Tools.h++ 类库参考》* • *《Tools.h++ 用户指南》*	HTML，位于安装的软件中的文档索引 `file:/opt/SUNWspro/docs/index.html`
自述文件和手册页	HTML，位于安装的软件上的文档索引中，在 Solaris 平台和 Linux 平台上的位置分别为 `file:/opt/SUNWspro/docs/index.html` 和 `file:/opt/sun/sunstudio10/docs/index.html`。
联机帮助	通过 IDE 中的 “帮助”菜单可以使用 HTML
发行说明	HTML，位于 `http://docs.sun.com`

相关文档

下表描述的相关文档可以在 `file:/opt/SUNWspro/docs/index.html` 和 `http://docs.sun.com` 上获得。如果软件没有安装在 `/opt` 目录中，请询问系统管理员以获取系统中的等效路径。

文档标题	描述
使用 dbx 调试程序	描述如何使用 `dbx` 命令行调试程序来调试用 C、C++、Fortran 和 Java™ 编程语言编写的程序。
Fortran 编程指南	描述如何在 Solaris™ 环境中编写高效 Fortran 代码；输入 / 输出、库、性能、调试和并行处理。
Fortran 库参考	详细说明 Fortran 库和内部例程
Fortran 用户指南	描述 `f95` 编译器的编译时环境和命令行选项。还包括关于将以前的 `f77` 程序迁移到 `f95` 的说明。
C 用户指南	描述 `cc` 编译器的编译时环境和命令行选项。
C++ 用户指南	描述 `CC` 编译器的编译时环境和命令行选项。
性能分析器	描述如何使用收集器和性能分析器来执行大范围性能数据的统计分析，以及跟踪各种系统调用，并在函数、源代码行和指令级将这些数据与程序结构相关联。

访问相关的 Solaris 文档

下表描述了可从 `docs.sun.com` Web 站点上获得的相关文档。

文档集合	文档标题	描述
Solaris 参考手册集合	请参阅手册页部分的标题。	提供关于 Solaris™ 操作环境的信息。
Solaris 软件开发人员集合	*链接程序和库指南*	描述了 Solaris™ 链接编辑器和运行时链接程序的操作。
Solaris 软件开发人员集合	*多线程编程指南*	涵盖 POSIX® 和 Solaris™ 线程 API、使用同步对象进行程序设计、编译多线程程序和多线程程序的查找工具。

开发人员资源

访问 `http://developers.sun.com/prodtech/cc` 以查找以下经常更新的资源：

- 关于编程技术和最佳方法的文章
- 短小编程提示的知识库
- 编译器和工具组件的文档以及与软件同时安装的文档的更正
- 支持等级信息
- 用户论坛
- 可下载代码样例
- 新技术预览

您可以在 `http://developers.sun.com` 上找到开发人员的额外资源。

与 Sun 技术支持联系

如果您有关于本产品的技术问题而本文档未予以解答，请访问：

`http://www.sun.com/service/contacting`

Sun 欢迎您提出意见和建议

Sun 致力于提高文档质量，并欢迎您提出宝贵的意见和建议。请将您的意见发送至以下 URL

`http://www.sun.com/hwdocs/feedback`

请在您的意见中提供文档的文件号码。例如，本文档的文件号码是 819-1597-10。

当您提供意见和建议时，可能需要在表单中提供文档英文版本的标题和文件号码。本文档英文版本的文件号码和标题是：819-0488-10、 Sun Studio 10 What's new。

第 1 章

Sun Studio 10 新增功能与增强特性

Sun™ Studio 10 将代替 Sun™ Studio 9。Sun Studio 10 发行版中的新增功能包括对以下编译器、库及工具的更新：

- C 编译器
- C++ 编译器
- Fortran 编译器
- Sun 性能库
- 分布式 `make` 实用程序， `dmake`
- `dbx` 命令行调试程序
- 性能分析工具
- 集成开发环境 (IDE)
- 文档

在大多数章节中，我们提供了一个表，其中列出了该组件的新增功能。该表分两列，左边一列提供了功能的简短说明，而右边一列显示了更详细的说明。

注 – 要查找本章中介绍的 Sun Studio 10 文档，请参阅随产品软件一起安装的、位于 `/opt/SUNWspro/docs/index.html` 的文档索引。如果软件不是安装在 `/opt` 目录中，请与系统管理员联系以获取系统或网络上的等效路径。

C 编译器

表 1-1 C 编译器新特性

特性	描述
OpenMP 并行编程 API	运行 Solaris 操作系统的基于 32 位 和 64 位 x86 系统上已经启用 API。
新增 `-xarch` 选项	`-xarch=amd64` 指定 64 位 AMD 指令集编译。指定 `-xarch=amd64` 时， C 编译器预定义 `__amd64` 和 `__x86_64`。
新增 `-xtarget` 选项	`-xtarget=opteron` 指定 32 位 AMD 编译中的 `-xarch`、`-xchip` 和 `-xcache` 设置。
基于 x86 系统的新增 `-xregs` 标志	仅适用 x86 的新标志 `-xregs` 选项，`-xregs=[no%]frameptr`，让您将帧指针寄存器作为未分配的调用方保存寄存器使用以提高应用程序的运行时性能。
`lint` 的新增 `-Xarch`=amd64 选项	C 实用程序 `lint` 现在接受新增选项 `-Xarch=amd64`。更多信息，请参阅 `lint`(1) 手册页。
基于 x86 系统的 `-xarch=generic64`	现有 `-xarch=generic64` 选项现在除了支持传统 SPARC 平台还支持 x86 平台。
基于 x86 系统的 `-xipo`	`-xipo` 选项现在已经可以在基于 x86 的系统上使用。

注 – 必须在命令行指定 `-fast` 和 `-xtarget` 右侧的 `-xarch=amd64` 来生成 64 位代码。例如，指定 `cc -fast -xarch=amd64` 或 `cc -xtarget=opteron -xarch=amd64`。新增 `-xtarget=opteron` 选项不会自动生成 64 位代码。该选项将扩展成 `-xarch=sse2`、`-xchip=opteron` 和 `-xcache=64/64/2:1024/64/16`，产生 32 位代码。 `-fast` 选项同样产生 32 位代码，因为该选项是定义 `-xtarget=native` 的宏。

C++ 编译器

表 1-2 C++ 编译器新特性

特性	描述
OpenMP 并行编程 API	运行 Solaris 操作系统的基于 32 位和 64 位 x86 系统上已经启用 API。
新增 `-xarch` 选项	`-xarch=amd64` 指定 64 位 AMD 指令集编译。指定 `-xarch=amd64` 时， C++ 编译器预定义 `__amd64` 和 `__x86_64`。
新增 `-xtarget` 选项	`-xtarget=opteron` 指定 32 位 AMD 编译中的 `-xarch`、 `-xchip` 和 `-xcache` 设置。
基于 x86 系统的新增 `-xregs` 标志	仅适用 x86 的新标志 `-xregs` 选项， `-xregs=[no%]frameptr`，让您将帧指针寄存器作为未分配调用方保存寄存器使用来提高应用程序的运行时性能。
基于 x86 系统的 `-xarch=generic64`	现有 `-xarch=generic64` 选项现在除了支持传统 SPARC 平台还支持 x86 平台。
基于 x86 系统的 `-xipo`	`-xipo` 选项现在已经可以在基于 x86 的系统上使用。
模板 - 模板参数	可以指定带参数的模板定义，参数就是模板本身，而不是类型或值。请回想一下，在类型上实例化的模板本身就是类型。例如，请参阅第 15 页 “模板 - 模板参数示例”。
访问嵌套类规则	缺省模式下，本发行版本的 C++ 编译器允许嵌套类同样访问成员函数的成员类。有关更多信息，请参阅第 16 页 “嵌套类访问规则”。

注 – 必须在命令行指定 `-fast` 和 `-xtarget` 右侧的 `-xarch=amd64` 来生成 64 位代码。例如，指定 `CC -fast -xarch=amd64` 或 `CC-xtarget=opteron -xarch=amd64`。新增 `-xtarget=opteron` 选项不会自动生成 64 位代码。该选项将扩展成 `-xarch=sse2`, `-xchip=opteron` 和 `-xcache=64/64/2:1024/64/16`，产生 32 位代码。`-fast` 选项同样生成 32 位代码，因为该选项是定义 `-xtarget=native` 的宏。

模板 - 模板参数示例

本节提供了两个代码示例，一个示例不使用模板 - 模板参数，另一个示例使用模板 - 模板参数。

本示例不使用模板 - 模板参数，因为 MyClass<int> 是一个类型。

```
template<typename T> class MyClass { ... };
std::list< MyClass<int> > x;
```

本示例中，类模板 C 具有类模板的参数，对象 x 是将类模板 A 做为其参数来使用的 C 的一个实例。 c 的成员 y 具有 A<int> 类型。

```
// ordinary class template
template<typename T> class A {
    T x;
};
// class template having a template parameter
template < template<typename U> class V > class C {
    V<int> y;
// instantiate C on template
C<A> x;
```

嵌套类访问规则

缺省标准模式下， C++ 编译器现在允许嵌套类访问封装类的私有成员。

C++ 标准声称嵌套类不能对封装类的成员进行特殊访问。然而，大多数人认为该限制是不合理的，因为成员函数可以访问私有成员，那么成员类也应该可以访问。下面的示例中，函数 foo 试图访问类 outer 的私有成员。根据 C++ 标准，除非在友元函数中声明，否则函数不能访问私有成员：

```
class outer {
    int i; // private in outer
    class inner {
        int foo(outer* p) {
            return p->i; // invalid
        }
    };
};
```

C++ 委员会正在将更改应用到访问规则中，使其同样可以访问成员函数的成员类。许多编译器在预期更改语言规则时已经实现了该规则。

要恢复旧编译器的行为，禁止访问，请使用编译器选项 -features=no%nestedaccess。缺省值为 -features=nestedaccess。

Fortran 编译器

表 1-3 Fortran 编译器新特性

特性	描述
OpenMP 并行编程 API	运行 Solaris 操作系统的基于 32 位和 64 位 x86 系统上已经启用 API。
新增 `-xarch` 选项	`-xarch=amd64` 指定 64 位 AMD 指令集编译。指定 `-xarch=amd64` 时， Fortran 编译器现在预定义 `__amd64` 和 `__x86_64`。
新增 `-xtarget` 选项	`-xtarget=opteron` 指定 32 位 AMD 编译的 `-xarch`、 `-xchip` 和 `-xcache` 设置。
基于 x86 系统的 `-xarch=generic64`	现有 `-xarch=generic64` 选项现在除了支持传统 SPARC 平台还支持 x86 平台。
基于 x86 系统的 `-xipo`	`-xipo` 现在已经可以在基于 x86 的系统上使用。
Big-endian 和 Little-endian 平台间的二进制（未格式化）文件共享	在基于 SPARCA 和 x86 的系统间移动时，新的编译器标志 `-xfilebyteorder` 支持二进制 I/O 文件。标志识别未格式化 I/O 文件的字节顺序和字节对齐。有关详细信息，请参阅第 17 页“Big-endian 和 Little-endian 平台间的二进制文件共享”。

注 – 必须在命令行指定 `-fast` 和 `-xtarget` 右侧的 `-xarch=amd64` 来生成 64 位代码。例如，指定 `f95 -fast -xarch=amd64` 或 `f95 -xtarget=opteron -xarch=amd64`。新增 `-xtarget=opteron` 选项不会自动生成 64 位代码。该选项将扩展成 `-xarch=sse2`、 `-xchip=opteron` 和 `-xcache=64/64/2:1024/64/16`，产生 32 位代码。 `-fast` 选项也产生 32 位代码，因为该选项是定义 `-xtarget=native` 的宏。

Big-endian 和 Little-endian 平台间的二进制文件共享

在基于 SPARCA 和 x86 的系统间移动时，新的编译器标志 `-xfilebyteorder` 支持二进制 I/O 文件。标志识别未格式化 I/O 文件的字节顺序和字节对齐。

标志的语法是：

`-xfilebyteorder={[little`*max_align*`:%all,`*unitno*`,`*filename*`}],[big`*max_align*`:{%all,`*unitno*`,`*filename*`}],[native:{%all,`*unitno*`,`*filename*`}]}`:

`max_align`	目标平台的最大字节对齐。值为 `1`、`2`、`4`、`8` 和 `16`。对齐适用于 Fortran VAX 结构和 Fortran 95 派生类型，这些派生类型使用依赖于平台的对齐来获得与 C 结构的兼容性。
`little`*max_align*`:{%all,`*unitno*`,`*filename*`}`	文件或单元编号的列表，文件或单元编号是最大字节对齐为 *max_align* 的系统上使用的 Little-endian 文件。例如，`little4` 描述了 32 位 x86 文件，而 `little16` 描述了 64 位 x86 文件。
`big`*max_align*`:{%all,`*unitno*`,`*filename*`}`	文件或单元编号的列表，文件或单元编号是最大字节对齐为 *max_align* 的系统上使用的 Big-endian 文件。
`native:{%all,`*unitno*`,`*filename*`}`	文件或单元编号的列表，文件或单元编号是编译处理器系统上使用的相同字节顺序和对齐的本机文件。
`%all`	指定所有文件和逻辑单元，以 “`SCRATCH`” 打开的或在该选项中显式命名的除外。可以用来描述该标志非显式列出的缺省文件。`%all` 只能出现一次。
unitno	该应用程序打开的 Fortran 逻辑单元编号。
filename	该应用程序打开的 Fortran 文件名称。

该选项不适用于 `STATUS=scratch` 时打开的文件。这些文件的 I/O 操作始终带有本机处理器的字节顺序和字节对齐。

如果在编译器命令行未指定 -xfilebyteorder，则第一个缺省值是 `-xfilebyteorder=native:%all`。必须至少指定该选项的一个参数。即，至少出现 `little:`、`big:` 或 `native:` 参数之一。

该标志未显式声明的文件假定为本地文件。例如，`-xfilebyteorder=little4:zfile.out` 时编辑声明的 `zfile.out` 是 Little-endian 32 位 x86 文件 （符合 4 字节最大数据对齐规则），其他的文件是本机文件。

如果指定给文件的字节顺序与本机处理器的字节顺序相同，而指定的对齐却与本机处理器不同，即使没有字节交换，也会使用适当的填充。例如，可能发生在这种情况下，`-xarch=amd64` 时对 64 位 x86 进行编译，且指定 `-xfilebyteorder=little4:`*filename*。

Big-endian 和 Little-endian 平台间共享的数据记录中的声明类型必须具有相同大小。例如，`-xtypemap=integer:64`、`real:64`、`double:64` 时编译，可执行 x86 生成的文件不能读取 `-xyptemap=integer:64`、`real:64`、`double:128` 时编译，可执行 SPARC 生成的文件，因为缺省双精度数据类型的大小不同。

共享的 I/O 文件不能包含 VAX `UNION/MAP` 数据结构，因为编译器无法知道如何解释 `UNION` 数据。声明带有 `-xfilebyteorder` 标志，包含 `UNION` 数据的文件将产生运行时错误。

命令行调试程序 dbx

表 1-4 `dbx` 新特性

特性	描述
AMD64 体系结构支持	64 位 `dbx` 现在支持 AMD64 体系结构。

基于 SPARC 系统的 Sun Studio 软件中，基于 x86 系统的 Sun Studio 软件包含两个 `dbx` 二进制， 32 位 `dbx` 仅能调试 32 位程序， 64 位 `dbx` 可以调试 32 位和 64 位程序。

启动 `dbx` 时，决定执行哪一个二进制。64 位 Solaris 操作系统上，缺省值是 64 位 `dbx`。

OpenMP API

表 1-5 OpenMP API 新特性

特性	描述
运行 Solaris 10 操作系统的基于 x86 系统的可用性。	基于 SPARC 系统的 Solaris 系统上已经可用的 OpenMP API 特性同样可以用在运行 Solaris 10 操作系统基于 32 位或 64 位 x86 系统上的 Sun Studio 编译器。
`libmtsk`	多任务库 `libmtsk` 现在是一个共享库，并且是 Solaris 10 操作系统的一部分。
嵌套并行操作	本发行版本支持嵌套并行操作。缺省时处于禁用状态，需要设置 `OMP_NESTED` 环境变量，使运行时调用 `omp_set_nested()` 函数来启用该操作。启用嵌套并行操作后，不能忽略在并行区域中对大多数 `omp_` 函数的调用。调整并行环境 （例如， `omp_set_num_threads()` 或 `omp_set_dynamic()`）的调用仅影响线程遇到的相同或内部嵌套级的后续并行区域。
线程的缺省行为	现在线程的缺省行为是 SLEEP。早期的缺省行为是 SPIN。要恢复早期的缺省行为，请使用 `SUNW_MP_THR_IDLE=SPIN`。
`SUNW_MP_NUM_POOL_THREADS` 环境变量	`SUNW_MP_NUM_POOL_THREADS` 指定线程池的大小 （最大线程数）。线程池仅包含非用户的线程 `libmtsk` 库创建的线程。不包含用户线程，例如主线程。将 `SUNW_MP_NUM_POOL_THREADS` 设置成 0 会强制线程池清空，一个线程将执行所有的并行区域。指定的值必须为非负整数。缺省值是 1023。该环境变量可以防止单一进程创建过多线程 （可能会发生），例如，带有递归嵌套并行区域的线程。
`SUNW_MP_MAX_NESTED_LEVELS` 环境变量	`SUNW_MP_MAX_NESTED_LEVELS` 指定活动并行区域的最大深度。具有超过 `SUNW_MP_MAX_NESTED_LEVELS` 活动嵌套深度的任何并行区域都将被单一线程执行。其值必须是正整数。缺省值是 4。最外层并行区域的深度级是 1。
`SUNW_MP_GUIDED_WEIGHT` 环境变量	`SUNW_MP_GUIDED_WEIGHT` 设置 `libmtsk` 所使用的加权值来循环 `GUIDED` 调度。 `libmtsk` 使用以下公式来计算 `GUIDED` 循环的块大小： *chunk_size=num_unassigned_iterations/*(*weight*num_threads*) 其中，*num_unassigned_iterations* 是循环中未分配给任何线程的迭代数， *weight* 是浮点常量 （本发行版本是 2.0，早期版本是 1.0），*num_threads* 是用于执行循环的线程数。为 `SUNW_MP_GUIDED_WEIGHT` 指定的值必须是正非零浮点常量。`libmtsk` 将使用该值做为 `GUIDED` 块大小计算中的加权。

区间运算

本发行版本没有新的区间运算的特性。

Sun 性能库

表 1-6 Sun 性能库新特性

特性	描述
64 位 Solaris 操作系统支持	本发行版本的 Sun 性能库包含对基于 x86 系统的 64 位 Solaris 操作系统的支持。

64 位 x86 版的 Sun 性能库与 SPARC v9 版在功能上相同，但以下项除外：

- 四精度例程（`dqdoti`、`dqdota`）不可用。
- 区间 BLAS 例程不可用。
- 带有 64 位整型参数的例程不可用。例如，`DAXPY()` 可用，而 `DAXPY_64()` 不可用。

要与高性能 amd64 优化的库进行链接，请使用 `-xarch=amd64` 标志。例如：

```
f95 -xarch=amd64 example.f -xlic_lib=sunperf
```

dmake

表 1-7 dmake 新特性

特性	描述
新的 DMAKE_OUTPUT_MODE 环境变量	新的环境变量或 make 程序的描述文件宏 DMAKE_OUTPUT_MODE，允许更改日志文件的格式。缺省情况下，或将 DMAKE_OUTPUT_MODE 设置为 TXT1 时，dmake 将向日志文件打印附加的系统信息，并重复带有输出的命令。将 DMAKE_OUTPUT_MODE 设置为 TXT2 时，系统信息将被省略，且命令从不重复。有关详细信息，请参阅 dmake(1) 手册页的 ENVIRONMENT/MACROS 一节。（请注意，手册页中的环境变量说明不正确； DMAKE_OUTPUT_MODE 的正确值是 TXT1 和 TXT2。）
Unix2003 遵循性	可以通过设置 DMAKE_COMPAT_MODE=POSIX 来强制 Unix2003 的遵循性。
网格引擎支持	通过设置 DMAKE_MODE=grid 来指定网格引擎支持。
系统重载控制	使用 DMAKE_ADJUST_MAX_JOBS 控制系统重载。
提高内存的使用情况	本发行版本中包括了提高内存的使用情况。

性能分析工具

表 1-8 性能分析工具新特性

特性	描述
对实验格式的更改	已对实验格式进行更改。现在日志有一个输入，给出了目标的大小（以位计算）。同时，版本由 9.1 更改为 9.2，因此旧工具无法读取新实验，但使用 Sun Studio 10 工具可以读取旧实验。
er_kernel 实用程序	新的 er_kernel 实用程序现在只适用于 Solaris 10 操作系统。 DTrace 权限要求使用该 er_kernel 实用程序。
增强的性能衡量标准精度	性能分析器和 er_print 实用程序的衡量标准精确度已由小数点后一位增加到两位。
直接编辑实验说明文件	性能分析器中已添加了对实验说明文件的直接编辑功能。
显示函数名称的新选项	现在在性能分析器和 er_print 命令中可以使用新的选项来显示函数名称。

表 1-8 性能分析工具新特性

特性	描述
增强的衡量标准选项	性能分析器中的衡量标准选项已得到增强。您可以一次性选择或清除所有衡量标准的显示。
收集器 GUI 的变更	用于以下后续进程的菜单已移动到 “收集实验” 标签。除了 on 和 off 选项，现在菜单还支持所有选项和扩展硬件计数器的溢出分析特性。
硬件计数器溢出分析的增强功能	硬件计数器溢出分析已增强到与大多数处理器一起使用，包括基于 x86 的处理器。在 `dbx` 和性能分析器 GUI 中使用 `collect -h` 命令、`collector hwprofile` 命令时可以使用增强功能。
新的 `appendfile` 选项	`appendfile` 选项已被添加到 `er_print` 实用程序中。该选项允许将 `er_print` 实用程序的输出添加到现有文件的末尾。
`er_src` 实用程序缺省行为的变更	`er_src` 实用程序的缺省行为已被更改为以下命令的相同行为：`er_src -source all -1 object`。
J2SE 技术的位置	目前性能分析器和 `collect` 实用程序使用 J2SE 技术的缺省位置 （产品安装程序安装它的位置）。
新的 `collect -J java_args` 选项	`collect -J java_args` 选项提供一种方法，来向正用于分析的 Java 安装传递标志参数。
暂停和恢复期间的抽样行为变更	抽样数据产生在暂停之前，恢复之后，但不在收集器暂停的时候。
JVM 函数的伪函数	Java 模式下 Java 虚拟机 （JVM）* 函数的伪函数名称由 `<JVM-Overhead>` 更改为 `<JVM-System>`。
`<Unknown>` 子类型	Java 函数 `<Unknown>` 子类型的名称已更改为更易于理解。
`.er.rc` 文件路径	进程 `.er.rc` 文件的路径目前在性能分析器以及 `er_print` 和 `er_src` 实用程序中分别显示在“错误 / 警告日志”窗口和 `stderr` 中。
`JDK_1_4_2_HOME` 环境变量	用于定义 Java 路径以进行数据收集的 `JDK_1_4_2_HOME` 环境变量，现已废弃。
堆分析	由于无法在 JVM 1.5 中得到支持，现已废弃对 Java 程序的堆分析。
`collect -J` 扩展选项	`collect` 实用程序将接受 on 或 off 的值，以及到 Java 安装的路径，以用于分析。

* 术语 “Java 虚拟机” 和 “JVM” 指 Java™ 平台的虚拟机。

集成开发环境 (IDE)

表 1-9 IDE 新特性

特性	描述
脚本执行功能	现在您可以直接从 IDE 执行脚本。
Linux 操作系统上的 ss_attach	ss_attach 特性现在可用于运行在 Linux 操作系统上的 Sun Studio 软件中

文档

有关更新 Sun Studio 10 文档的信息，请参阅开发人员门户网站中的最新消息页，位于 `http://developers.sun.com/prodtech/cc/support_index.html`。

第 2 章

Sun Studio 9 新增功能与增强特性

Sun™ Studio 9 将代替 Sun™ Studio 8。Sun Studio 9 发行版中的新增功能包括对以下编译器、库及工具的更新：

- C 编译器
- C++ 编译器
- Fortran 编译器
- Sun 性能库
- 分布式 `make` 实用程序， `dmake`
- `dbx` 命令行调试程序
- 性能分析工具
- 集成开发环境 (IDE)
- 文档

在大多数章节中，我们提供了一个表，其中列出了该组件的新增功能。该表分两列，左边一列提供了功能的简短说明，而右边一列显示了更详细的说明。

注 – 要查找本章中介绍的 Sun Studio 9 文档，请参阅随产品软件一起安装的、位于 `/opt/SUNWspro/docs/index.html` 的文档索引。如果软件不是安装在 `/opt` 目录中，请与系统管理员联系以获取系统或网络上的等效路径。

C 编译器

本节列出了该发行版的 C 编译器的新增功能。下表中包括下列新特性：

- 表 2-1 一般增强特性
- 表 2-2 增强的硬件平台支持
- 表 2-3 改进的性能和优化选项
- 表 2-4 通过 Lint 实用程序进行新的安全检查

有关本节中引用的特定编译器选项的详细信息，请参阅《*C 用户指南*》或 cc(1) 手册页。

表 2-1 列出了 C 编译器的一般增强特性。

表 2-1 C 编译器的一般增强特性

特性	描述
其他 C99 功能的实现	该发行版增加了对以下 ISO/IEC 9899:1999（在本文中简称 C99）功能的支持。下表仅详细列出此发行版中实现的 C99 功能，它是所有实现的 C99 功能的子集。有关 C 编译器的早期和当前发行版中实现的所有 C99 功能的完整列表，请参阅《*C 用户指南*》。我们为此 Sun Studio 9 发行版中支持的每个新项列出了 C99 标准中各小节的编号。 • 5.2.4.2.2：支持 `FLT_EVAL_METHOD` 宏。该宏和新的 `-fleteval` 编译选项一起，确定了编译器是否将浮点表达式作为 long double 进行计算，或是否根据表达式中类型与常量的组合来对其进行计算。 • 6.4.3: 支持四位和八位通用字符名 (UCN)，它们可用于标识符、字符常量和字符串文字中，以指定 C 基本字符集中不存在的字符。UCN \Unnnnnnnn 指定了其八位短标识符（根据 ISO/IEC 10646 的规定）为 nnnnnnnn 的字符。同样，通用字符名 \unnnn 指定了其四位短标识符为 nnnn （其八位短标识符为 0000nnnn）的字符。 • 6.7.4：支持内联函数和外部内联函数 • 6.7.8: 支持指定的初始化函数，它为初始化稀疏数组和结构提供了一种方法，这在数值和系统编程中很常见。
通过新的 -features 编译选项改进了与旧二进制的兼容性	现在，您无需更改旧二进制的行为即可将旧的 C 和 C++ 二进制（C/C++ 5.6 之前的版本）与新的 C 和 C++ 二进制链接在一起。如果希望新的二进制与包含外部内联函数的旧 C 和 C++ 二进制之间相兼容，请使用 -features=no%extinl 编译选项。 要获得符合标准的行为，必须使用当前编译器对旧代码重新进行编译。
较大的从属线程缺省栈大小	现在，从属线程的缺省栈大小更大一些。所有从属线程的栈大小都相同。缺省情况下，对于 32 位应用程序，栈大小为 4 兆字节；对于 64 位应用程序，栈大小为 8 兆字节。该大小使用环境变量 `STACKSIZE` 进行设置。

表 2-1 C 编译器的一般增强特性 *(续)*

特性	描述
改进的 `-xprofile` (SPARC®)	`-xprofile` 选项进行了以下改进： • 支持文件配置共享库 • 使用 `-xprofile=collect -mt` 收集线程安全配置文件 • 改进了对单个配置文件目录中多个程序或共享库文件配置的支持。 通过设置 `-xprofile=use`，编译器能够在包含具有非唯一基名的多个对象文件数据的配置文件目录中查找配置文件数据。如果编译器无法找到对象文件的配置文件数据，它将提供一个新选项 `-xprofile_pathmap=`*collect-prefix*:*use-prefix*。
支持 UTF-16 字符串文字：`-xustr`	如果需要支持使用 ISO10646 UTF-16 字符串文字的国际化应用程序，请指定 `-xustr=ascii_utf16_ushort`。换句话说，如果您的代码包含由 16 位字符组成的字符串文字，请使用此选项。如果不使用该选项，则编译器既不会生成、也无法识别 16 位字符的字符串文字。此选项使系统能够将 U"*ASCII_string*" 字符串文字识别为无符号短类型数组。因为这样的字符串还不属于任何标准，所以该选项的作用是使非标准 C 得以识别。
自动生成的预编译头文件	本发行版的 C 编译器扩展了预编译头文件工具，使其包含在编译器的某一部分自动生成预编译头文件的功能。您还可以手动生成预编译头文件，但是，如果您对编译器的新增功能感兴趣的话，请参阅 `cc`(1) 手册页中 `-xpch` 选项的相关说明，以了解更多信息。另请参阅 `CCadmin`(1) 手册页。

表 2-2 列出了 C 编译器中支持快速编译的新增功能。

表 2-2 增强的硬件平台支持

特性	描述
支持 SPARC® 平台的更多标志	现在， `-xchip` 和 `-xtarget` 选项支持 ultra3i 和 ultra4 作为其值，因此您可以生成针对 UltraSPARC IIIi 和 UltraSPARC IV 处理器进行了优化的应用程序。
支持 x86 平台的更多标志	C 编译器支持 `-xarch`、`-xtarget` 和 `-xchip` 编译选项的新标志用于将在 x86 平台上运行的代码。这些新标志经过精心设计，可充分利用 Pentium 3 和 Pentium 4 芯片以及 Solaris™ 软件对 x86 平台上 `sse` 和 `sse2` 指令的支持。新标志如下所示： • `-xchip=pentium3` 对 Pentium 3 处理器进行优化 • `-xchip=pentium4` 对 Pentium 4 处理器进行优化 • `-xarch=sse` 将 sse 指令集添加到 pentium_pro 指令集体系结构中 • `-xarch=sse2` 将 sse2 指令集添加到 sse 允许的指令集中 • `-xtarget=pentium3` 设置了 -xarch=sse、 -xchip=pentium3 和 -xcache=16/32/4:256/32/4 • `-xtarget=pentium4` 设置了 -xarch=sse2、 -xchip=pentium4 和 -xcache=8/64/4:256/128/8 您可以按照下面这些原则来确定哪种选项组合最适合您的编译要求： • 如果您要构建在采用 Solaris 9 更新版 6 或更高版本的 Pentium 3 或 Pentium 4 机器上运行的应用程序，请根据情况使用 `-xtarget=pentium3` 或 `-xtarget=pentium4` 进行编译。 • 如果您要构建在采用 Solaris 9 更新版 5 或更低版本的 Pentium 3 或 Pentium 4 机器上运行的应用程序，请设置 `-xarch=pentium_pro`（而并非您所认为的 pentium3 或 pentium4），因为 Solaris 9 更新版 5 或更低版本的操作系统不支持 sse 和 sse2 指令。使用 `-xtarget=pentium3` 或 `-xtarget=pentium4` 将 `-xchip` 和 `-xcache` 设置为使用相同的值，具体取决于目标计算机。 • 如果在目标计算机上进行构建，则指定 `-fast`、`-xarch=native` 或 `-xtarget=native` 将自动扩展至上述相应的 `-xchip`、`-xarch` 和 `-xtarget` 标志设置。

表 2-3 列出了支持增强性能的 C 编译器的新特性。

表 2-3 改进的性能和优化选项

特性	描述
编译器选项的全新缺省设置及其扩展	以下编译选项的缺省设置已更改： • SPARC® 平台上的 `-xarch`：v8plus。新的缺省设置可以使几乎所有当前使用的计算机的运行时性能更佳。但是，专门用于在 UltraSPARC 之前的计算机上进行部署的应用程序将不再使用该缺省选项来执行；此时应使用 `-xarch=v8` 进行编译，以确保应用程序能够在 UltraSPARC 之前的计算机上执行。 • SPARC® 平台上的 `-xcode`: `abs44`（用于 v9）和 `abs32`（用于 v8）。 • SPARC® 平台上的 `-xmemalign`：`8i`（用于 v8）和 `8s`（用于 v9） • SPARC® 平台上的 `-xprefetch`：`auto,explicit`。此项更改将对本来具有非线性内存访问模式的应用程序造成负面影响。要禁用此更改，请指定 `-xprefetch=no%auto,no%explicit`。 以下选项和宏的扩展已更改： • 现在，`-fast` 选项将在其扩展中包括新的选项 `-xlibmopt`（如下所示）。 • `-O` 宏现在扩展为 `-xO3` 而不是 `-xO2`。更改缺省设置可以使运行时性能更佳。但是，对于依赖于所有自动被视为 volatile 的变量的程序，`-xO3` 可能不适用。可能依赖于此假定的典型程序是那些实现其自己的同步基元的设备驱动程序以及早期多线程应用程序。其解决办法是使用 `-xO2` 进行编译，而不使用 `-O`。
新的优化编译选项	新编译选项如下所示： • `-xlibmopt` 和 `-xnolibmopt`：使用 `-xlibmopt` 选项，编译器可以使用优化的数学例程库。使用 `-xlibmopt` 选项时，必须通过指定 `-fround=nearest` 来使用缺省舍入模式。数学例程库的性能得到优化，通常生成速度较快的代码。这样生成的代码可能与普通数学库生成的代码稍有不同。不同之处通常在最后一位上。 通过在命令行上指定新的 `-xnolibmopt` 选项，您可以显式关闭此库。 • `-xipo_archive`：新的 `-xipo_archive` 选项使编译器能够通过使用 `-xipo` 进行编译以及在生成可执行文件之前保存在归档库 (.a) 中的对象文件来优化传递给链接程序的对象文件。库中包含的、在编译进程中进行了优化的任何对象文件都将被其已优化的版本替换。
新的优化编译选项（续）	• `-xprefetch_auto_type`：使用新的 `-xprefetch_auto_type` 选项能够以与生成直接内存访问预取相同的方式来生成由选项 -xprefetch_level=[1|2|3] 指示的间接循环预取。 诸如 `-xdepend`、`-xrestrict` 和 `-xalias_level` 等选项可以改进 `-xprefetch_auto_type` 的优化优势。它们会影响计算间接预取候选项的主动性，进而影响自动间接预取插入的主动性，因为它们有助于更好地消除内存别名信息的歧义。

表 2-4 介绍了 lint 实用程序中包括的新的安全检查功能。

表 2-4 通过 Lint 实用程序进行新的安全检查

特性	描述
用于 lint 的新 -errsecurity 选项	Sun Studio 9 发行版的 lint 实用程序采用了新的安全检查工具。在编译前可以使用新的 -errsecurity 选项来检查您的代码，以确保安全。 -errsecurity[={core | standard | extended | %none}] lint -errsecurity=core 检查源代码结构，这些结构几乎总是不安全或难以验证。此级别的检查包括： • 使用变量格式字符串和 printf() 以及 scanf()）系列函数 • 在 scanf() 函数中使用无界字符串 (%s) 格式 • 使用没有安全用法的函数：gets()、cftime()、ascftime()、creat() • 错误使用 open() 和 O_CREAT 将在此级别生成警告的源代码当作错误。应更改有问题的源代码。所有情况下，采用更安全简单的替代代码。 lint -errsecurity=standard 包括核心级别的所有检查以及可能安全的结构，但存在更好的可选方法。当检查新编写的代码时推荐采用此级别检查。此级别的其他检查包括： • 使用除 strlcpy （）外的字符串复制函数 • 使用弱随机数函数 • 使用不安全的函数生成临时文件 • 使用 fopen() 创建文件 • 使用调用 shell 的函数 使用新的或大幅修改的代码替换在此级别生成警告的源代码。消除传统代码的这些警告对应用程序造成不稳定的风险。

表 2-4 通过 Lint 实用程序进行新的安全检查 *(续)*

特性	描述
用于 `lint` 的新 `-errsecurity` 选项（续）	`lint -errsecurity=extended` 包含一套最全面的检查，包括核心级别和标准级别中的所有内容。此外，生成的许多有关构造的警告在某些情况下是不安全的。此级别的检查可用作检查代码的辅助措施，但无需将这些检查用作判断源代码是否可接受的标准检查。此级别的其他检查包括： • 在循环中调用 `getc()` 或 `fgetc()` • 使用易于产生路径名争用情况的函数 • 使用 `exec()` 函数系列 • `stat()` 和其他函数之间的争用情况 查看在这一级别产生警告的源代码，以确定可能的安全性问题是否仍然存在。 如果未指定 `-errsecurity` 的设置，则编译器会将其设置为 `-errsecury=%none`。如果指定了不带参数的 `-errsecurity`，则编译器会将其设置为 `-errsecurity=standard`。

C++ 编译器

本节列出了此发行版本中 C++ 编译器的新特性。下表中包括下列新特性：

- 表 2-5 一般增强特性
- 表 2-6 增强的硬件平台支持
- 表 2-7 新的和增强的优化选项

有关在本节中引用的特定编译器选项的更多信息，请参阅 *《C++ 用户指南》* 或 `CC(1)` 手册页。

表 2-5 列出了 C++ 编译器 （5.6 版本）的一般增强特性。

表 2-5 C++ 编译器的一般增强特性

特性	描述
外部链接的内联函数	C++ 标准规定，除非已声明为静态，否则内联 (`inline`) 函数类似于非内联函数，具有外部链接。C++ 5.6 首次为内联函数赋予了外部链接 （在缺省情况下）。如果必须在行外生成内联函数 （例如，如果需要其地址），则仅将一个副本链接到最终程序中。以前，需要副本的每个对象文件具有自己的带有本地链接的副本。 **外部内联**函数的实现可与早期编译器版本创建的二进制文件兼容，也就是说，程序行为与过去一样，均符合标准。旧的二进制可能拥有内联函数的多个局部副本，但新代码最多只能拥有**外部内联**函数的一个副本。 **外部内联**函数的实现可与内联函数的 C99 版兼容，该内联函数使用了此发行版中包含的 C 5.6 编译器。也就是说，按照 C 和 C++ 的外部内联函数规则，可以同时在 C 和 C++ 文件中定义相同的内联函数，并且最终程序中将只出现外部函数的一个副本。
增强的 UTF-16 支持	版本 5.5 的 C++ 编译器引入了对 UTF-16 字符串文字的支持。此发行版扩展了对使用语法 U"x" 的 UTF-16 字符串文字的支持，该语法类似于字符串的 U"x" 语法。要识别 UTF-16 字符文字，需要使用相同的 `-xustr` 选项。 本发行版还支持 UTF-16 字符和字符串文字中的数值转义，此支持与普通字符和字符串文字中的数值转义类似。例如： U"ab\123ef" // 字符的八进制表示 U"\x456" // 字符的十六进制表示 有关详细信息，请参阅 C++ 手册页 CC(1) 中有关 -xustr 的说明。
自动生成的预编译头文件	该版本的 C++ 编译器扩展了预编译头文件功能，以便使编译器的某个部分能够自动生成预编译头文件。您还可以手动生成预编译头文件，但是，如果您对编译器的新增功能感兴趣的话，请参阅 CC(1) 手册页中 `-xpch` 选项的相关说明，以了解更多信息。另请参阅 CCadmin(1) 手册页。

表 2-6 列出了支持更快速编译的 C++ 编译器新特性。

表 2-6 增强的硬件平台支持

特性	描述
支持 SPARC® 平台的更多标志	现在， `-xchip` 和 `-xtarget` 选项支持 ultra3i 和 ultra4 作为其值，因此您可以生成针对 UltraSPARC IIIi 和 UltraSPARC IV 处理器进行了优化的应用程序。
支持 x86 平台的更多标志	C 编译器支持 `-xarch`、 `-xtarget` 和 `-xchip` 编译选项的新标志用于将在 x86 平台上运行的代码。这些新标志经过精心设计，可充分利用 Pentium 3 和 Pentium 4 芯片以及 Solaris™ 软件对 x86 平台上 `sse` 和 `sse2` 指令的支持。新标志如下所示： • `-xchip=pentium3` 对 Pentium 3 处理器进行优化 • `-xchip=pentium4` 对 Pentium 4 处理器进行优化 • `-xarch=sse` 将 sse 指令集添加到 pentium_pro 指令集体系结构中 • `-xarch=sse2` 将 sse2 指令集添加到 sse 允许的指令集中 • `-xtarget=pentium3` 设置了 -xarch=sse、 -xchip=pentium3 和 -xcache=16/32/4:256/32/4 • `-xtarget=pentium4` 设置了 -xarch=sse2、 -xchip=pentium4 和 -xcache=8/64/4:256/128/8 您可以按照下面这些原则来确定哪种选项组合最适合您的编译要求： • 如果您要构建在采用 Solaris 9 更新版 6 的 Pentium 3 或 Pentium 4 机器上运行的应用程序，请根据情况使用 `-xtarget=pentium3` 或 `-xtarget=pentium4` 进行编译。 • 如果您要构建在采用 Solaris 9 更新版 5 或更低版本的 Pentium 3 或 Pentium 4 机器上运行的应用程序，请设置 `-xarch=pentium_pro`（而并非您所认为的 pentium3 或 pentium4），因为 Solaris 9 更新版 5 或更低版本的操作系统不支持 sse 和 sse2 指令。使用 `-xtarget=pentium3` 或 `-xtarget=pentium4` 时，请将 `-xchip` 和 `-xcache` 设置为使用相同的值，具体取决于目标机器。 • 如果在目标机器上进行构建，则指定 `-fast`、`-xarch=native` 或 `-xtarget=native` 将自动扩展至上述相应的 `-xchip`、`-xarch` 和 `-xtarget` 标志设置。

表 2-7 列出了支持更轻松移植的 C++ 编译器的新特性：

表 2-7 新的和增强的优化选项

特性	描述
编译器选项的全新缺省设置及其扩展	以下编译选项的缺省设置已更改： • SPARC® 平台上的 `-xarch`：v8plus。新的缺省设置可以使几乎所有当前使用的计算机的运行时性能更佳。但是，专门用于在 UltraSPARC 之前的计算机上进行部署的应用程序将不再使用该缺省选项来执行；此时应使用 `-xarch=v8` 进行编译，以确保应用程序能够在 UltraSPARC 之前的计算机上执行。 • SPARC® 平台上的 `-xcode`: `abs44`（用于 v9）和 `abs32`（用于 v8）。 • SPARC® 平台上的 `-xmemalign`：`8i`（用于 v8）和 `8s`（用于 v9） • SPARC® 平台上的 `-xprefetch`：`auto`、`explicit`。此项更改将对本来具有非线性内存访问模式的应用程序造成负面影响。要禁用此更改，请指定 `-xprefetch=no%auto,no%explicit`。 以下宏的扩展已更改： • `-O` 宏现在扩展为 `-xO3` 而不是 `-xO2`。更改缺省设置可以使运行时性能更佳。但是，对于依赖于所有自动被视为 volatile 的变量的程序，`-xO3` 可能不适用。可能依赖于此假定的典型程序是那些实现其自己的同步基元的设备驱动程序以及早期多线程应用程序。其解决办法是使用 `-xO2` 进行编译，而不使用 `-O`。
新的循环优化编译选项	现在，C++ 编译器支持以下选项，以优化其计算可并行进行的循环。只有当您指定 -xO3 或更高的优化级别时，这些选项才有效。 • `-xautopar` • `-xvector` • `-xdepend` 有关详细信息，请参阅 C++ 手册页 CC(1) 中有关 `-xautopar`、`-xvector` 和 `-xdepend` 的说明。
新的特定函数优化级别控制	您可以将 `#pragma opt` 指令与命令行选项 `-xmaxopt` 组合在一起，以指定编译器对个别函数应用的优化级别。需要减少特定函数的优化级别时，可以利用这种组合用法，例如，要避免增加代码（如消除堆栈帧），或提高特定函数的优化级别。
循环的预取优化	`-xprefetch_auto_type`：使用新的 `-xprefetch_auto_type` 选项能够以与生成直接内存访问预取相同的方式来生成由选项 -xprefetch_level=[1|2|3] 指示的间接循环预取。 诸如 `-xdepend`、`-xrestrict` 和 `-xalias_level` 等选项可以改进 `-xprefetch_auto_type` 的优化优势。它们会影响计算间接预取候选项的主动性，进而影响自动间接预取插入的主动性，因为它们有助于更好地消除内存别名信息的歧义

表 2-7 新的和增强的优化选项 *(续)*

特性	描述
限定指针优化	C++ 不支持 C99 中引入的 `restrict` 关键字。但现在 C++ 编译器接受 C 编译器选项 `-xrestrict`。 该选项作出了有关编译中函数的断言，即指针类型的函数参数并非指相同或重叠的对象。此选项对 C++ 比对 C 稍微更危险一些，因为该断言对 C++ 标准库中的某些函数来说是不正确的。

Fortran 编译器

表 2-8 列出了此发行版的 Fortran 编译器的新功能和增强功能，其中包括以下几个方面：

- Solaris™ 操作系统 x86 平台上 f95 的新编译功能
- 改进的运行时性能
- 新的 Fortran 2003 命令行内部函数
- 已更改的 `f95` 编译器命令行选项缺省设置
- 缺省 SPARC® 体系结构中所做的更改
- OpenMP 库的增强
- 新的 `f95` 编译器命令行选项

有关在本节中引用的特定编译器选项的更多信息，请参阅 *《Fortran 用户指南》*或 `f95(1)` 手册页。

表 2-8 Fortran 编译器的新功能和增强特性

特性	描述
Solaris 操作系统 x86 平台上 f95 的新编译功能	使用值为 `generic`、`native`、`386`、`486`、`pentium`、`pentium_pro`、`pentium3` 或 `pentium4` 的 `-xtarget` 进行编译，以便在 Solaris x86 平台上生成可执行文件。 x86 平台上的缺省值为 -xtarget=generic 以下 `f95` 功能尚未在 x86 平台上实现，并且仅可在 SPARC® 平台上使用： • 区间运算 （编译器选项 `-xia` 和 `-xinterval`） • 四精度 （128 位）算法 • IEEE 内模块 IEEE_EXCEPTIONS、 IEEE_ARITHMETIC 和 IEEE_FEATURES • sun_io_handler 模块 • 诸如 `-autopar`、`-parallel`、`-explitipar` 和 `openmp` 等并行化选项。 以下 `f95` 命令行选项仅可在 x86 平台上使用，而不能在 SPARC® 平台上使用： • `-fprecision`、 `-fstore`、 `-nofstore` 以下 f95 命令行选项仅可在 SPARC® 平台上使用，而不能在 x86 平台上使用： • `-xcode`、 `-xmemalign`、 `-xprefetch`、 `-xcheck`、 `-xia`、 `-xinterval`、 `-xipo`、 `-xjobs`、 `-xlang`、 `-xlinkopt`、 `-xloopinfo`、 `-xpagesize`、 `-xprofile_ircache`、 `-xreduction`、 `-xvector`、 `-depend`、 `-openmp`、 `-parallel`、 e–-autopar`、 `-explicitpar`、 `-vpara`、 `-XlistMP` 此外，在 x86 平台上， `-fast` 选项可扩展为包括已添加的选项 `-nofstore`。

表 2-8 Fortran 编译器的新功能和增强特性 *(续)*

特性	描述
Solaris 操作系统 x86 平台上 f95 的新编译功能 （续）	Fortran 编译器支持 `-xarch`、`-xtarget` 和 `-xchip` 编译选项的新标志用于将在 x86 平台上运行的代码。这些新标志经过精心设计，可充分利用 Pentium 3 和 Pentium 4 芯片以及 Solaris™ 软件对 x86 平台上 `sse` 和 `sse2` 指令的支持。新标志如下所示： • `-xchip=pentium3` 对 Pentium 3 处理器进行优化 • `-xchip=pentium4` 对 Pentium 4 处理器进行优化 • `-xarch=sse` 将 sse 指令集添加到 pentium_pro 指令集体系结构中 • `-xarch=sse2` 将 sse2 指令集添加到 sse 允许的指令集中 • `-xtarget=pentium3` 设置了 -xarch=sse、 -xchip=pentium3 和 -xcache=16/32/4:256/32/4 • `-xtarget=pentium4` 设置了 -xarch=sse2、 -xchip=pentium4 和 -xcache=8/64/4:256/128/8 • `-fns` 仅可在 pentium3 或 pentium4 处理器上启用。当 `-xarch` 不是 `sse` 或 `sse2` 时，将忽略 `-fns=yes`。否则，对于 SSE 和 SSE2 浮点指令， `-fns=yes` 意味着下溢将清零 (FTZ)，并且非规格化操作数将按零处理 (DAZ)。`-fns=yes` 不会影响传统的 x86 浮点指令。例如，对 long double 操作数或结果进行浮点运算可以利用传统的 x86 浮点指令，并且这些运算不会受 `-fns=yes` 的影响。 *x86* **的特殊注意事项：** 使用 `-xarch=sse` 或 `-xarch=sse2` 进行编译以便在 Solaris™ x86 SSE/SSE2 平台上运行的程序必须只能在支持 SSE/SSE2 的平台上运行。在不支持 SSE/SSE2 的平台上运行此类程序会导致发生段故障或错误的结果，并且不会显示任何显式警告消息。以后可能会提供 OS 和编译器的补丁程序，以避免在不支持 SSE/SSE2 的平台上执行 SSE /SSE2 编译的二进制。支持 SSE/SSE2 的 x86 平台包括在 Pentium 4 兼容处理器上运行的 Solaris 9 更新版 6。 此外，这一警告消息还扩展到了采用 `.il` 内联汇编语言函数或 `__asm()` 汇编程序代码 （利用 SSE/SSE2 指令）的程序。 在尝试运行为目标运行时平台编译的二进制时，请与您的系统管理员联系，以确定这些平台是否支持 SSE/SSE2。
改进的运行时性能	采用此发行版本后，大多数应用程序的运行时性能应有显著提高。为获得最佳效果，请用较高的优化级别 `-xO4` 或 `-xO5` 进行编译。在这些优化级别上，编译器现在可以内联包含的过程，以及那些带假定形式参数、可分配参数或指针参数的过程。

表 2-8 Fortran 编译器的新功能和增强特性 *(续)*

特性	描述
新的 Fortran 2003 命令行内部函数	Fortran 2003 标准草案引入了三个新的内部函数，来处理命令行参数和环境变量。此发行版的 f95 编译器已实现了这些内部函数。新增的内部函数包括： • GET_COMMAND(command, length, status) 以命令的形式返回调用该程序的整个命令行。 • GET_COMMAND_ARGUMENT(number, value, length, status) 以值的形式返回命令行参数。 • GET_ENVIRONMENT_VARIABLE(name, value, length, status, trim_name) 返回环境变量的值。
已更改的命令行选项缺省设置	此发行版的 f95 更改了以下命令行选项缺省设置。 • `-xprefetch` 缺省值为 `-xprefetch=no%auto,explicit`。 • `-xmemalign` 的缺省值为 `-xmemalign=8i`，`-xarch=v9` 和 `v9a` 除外，其缺省值为 `-xmemalign=8f`。
缺省 SPARC® 体系结构中所做的更改	缺省 SPARC® 体系结构不再是 V7。此 Sun Studio 9 发行版中限制了对 `-xarch=v7` 的支持。新的缺省值为 `V8PLUS` (UltraSPARC)。编译时使用 `-xarch=v7` 将被视为 `-xarch=v8`，因为 Solaris 8 OS 仅支持 `-xarch=v8` 或更高版本。
OpenMP 库的增强	OpenMP 库在以下几个方面得到了增强： • `OMP_NUM_THREADS` 和多任务库的最大线程数量已从 128 增至 256。 • 此发行版 Fortran 95 编译器的共享内存并行编程的 OpenMP API 实现在并行区域中采用了自动变量作用域功能。有关详细信息，请参阅 《OpenMP API 用户指南》 。（对于此发行版， OpenMP 仅在 SPARC® 平台上实现。）

表 2-8 Fortran 编译器的新功能和增强特性 *(续)*

特性	描述
新的 f95 编译器命令行选项	以下 f95 命令行选项在此发行版中为新选项。有关详细信息，请参阅 f95(1) 手册页。 • `-xipo_archive={ none | readonly | writeback }` 允许交叉文件优化，以包括归档 (.a) 库。（仅适用于 SPARC®） • `-xipo_archive=none` 未处理任何归档文件。 • `-xipo_archive=readonly` 编译器可以通过使用 -xipo 进行编译、在生成可执行文件之前保存在归档库 (.a) 中的对象文件来优化传递给链接程序的对象文件。 • `-xipo_archive=writeback` 编译器可以通过使用 -xipo 进行编译、在生成可执行文件之前保存在归档库 (.a) 中的对象文件来优化传递给链接程序的对象文件。库中包含的、在编译进程中进行了优化的任何对象文件都将被其已优化的版本替换。 如果未指定 -xipo 的设置，则编译器会将其设置为 `-xipo_archive=none`。 • `-xprefetch_auto_type=[no%]indirect_array_access` 生成间接预取，以间接访问数据数组。（仅适用于 SPARC®） • `[no%]indirect_array_access` 以与生成直接内存访问预取相同的方式来生成 [不生成] 由选项 `-xprefetch_level=[1|2|3]` 指示的间接循环预取。 如果未指定 `-xprefetch_auto_type` 的设置，则编译器会将其设置为 `-xprefetch_auto_type=[no%]indirect_array_access`。 `-xprefetch` 选项仅可在 SPARC® 平台上使用 诸如 `-xdepend`、`-xrestrict` 和 `-xalias_level` 等选项可影响计算间接预取候选项的主动性，进而影响自动间接预取插入的主动性，因为它们可以更好地消除内存别名信息的歧义。 • `-xprofile_pathmap=collect_prefix:use_prefix` 设置配置文件数据文件的路径映射。当文件配置的目录不是以前通过 `-xprofile=collect` 进行编译时所使用的目录时，请将 `-xprofile_pathmap` 选项与 `-xprofile=use` 选项一起使用。

命令行调试程序 dbx

以下新功能已添加到 Sun Studio 9 发行版的 `dbx` 中：

- 在 Linux 平台上支持 gcc 和 g++ 编译器
- 在 Solaris™ 操作系统 (x86 Platform Edition) 上支持 Fortran

区间运算

在此发行版本中没有新的区间运算功能。

Sun 性能库

Sun Performance Library™ 是一组经过优化、用于解决线性代数和其他数值密集型问题的高速数学子例程。Sun 性能库是建立在可从 Netlib (`http://www.netlib.org`) 获得的公共域应用程序集合之上。将这些例程增强并捆绑后即为 Sun 性能库。

表 2-9 列出了本发行版的 Sun 性能库新特性。有关更多信息，请参阅 *《Sun 性能库用户指南》*和 3p 手册页。

表 2-9 Sun 性能库新特性

特性	描述

为 x86 发行的 Sun 性能库	此发行版的 Sun 性能库包括 Solaris/x86 平台的库。存在两种版本：
	• 对支持 SSE2 指令集的系统使用这些指令的高性能版本。
	• 适用于不支持 SSE2 的系统的兼容版本。
	x86 版的 Sun 性能库与 SPARC® 版在功能上相同，但以下项除外：
	• 四精度例程 （`dqdoti`、`dqdota`）不可用
	• 区间 BLAS 例程不可用
	• x86 库为单线程
	• 仅可使用 32 位寻址功能
	• 可移植库性能特性在 Solaris/x86 上不可用
	SSE2 支持需要以下版本的 Solaris/x86：
	• Solaris 10 试用版 48 （或更高版本）
	• Solaris 9 试用版 6 更新版 5 （或更高版本）
	要与高性能 SSE2 优化的库进行链接，请使用 -xarch=sse2 标志。例如：
	`f95 -xarch=sse2 example.f -xlic_lib=sunperf`
	或
	`cc -xarch=sse2 example.c -xlic_lib=sunperf`

dmake

dmake 是一个命令行工具，与 make(1) 兼容。 dmake 能够以分布、并行或串行模式生成目标。如果使用标准 make(1) 实用程序，在对 make 程序的描述文件修改的情况下对 dmake 的转变很小。 dmake 是 make 实用程序的一个超集。对于嵌套 make，如果一个顶级的 make 程序的描述文件称为 make，则需要使用 $(MAKE)。 dmake 将分析 make 程序的描述文件并决定能够并行生成哪些目标，以及在您设置的众多主机上分布这些目标的试用版本。有关其他详细信息，请参阅 dmake(1) 手册页。表 2-10 列出了 Sun Studio 9 发行版中 dmake 的新特性。

表 2-10 dmake 新特性

特性	描述
dmake 中针对 Solaris 所做的性能、可靠性和可用性方面的改进	make 程序的描述文件分析器比原来的版本快 10 倍，而比 GNU make 快 3 倍。试用版本运行更快速，并且更稳定。此外，日志文件也更具有可读性。
Linux dmake 实现	在串行、并行和分布式模式下为 Linux 试用版实现完整的 dmake 功能。因此，无需对 make 程序的描述文件做重大变更即可在 Linux 上生成 Solaris™ 应用程序。一个试用版本可以同时分发给 Linux 和 Solaris™ 系统。

性能分析工具

表 2-11 列出了 Sun Studio 9 发行版性能分析工具中的新数据集合和演示特性。有关详细信息，请参阅以下手册页：

- analyzer(1)
- collect(1)
- collector(1)
- er_print(1)
- er_src(1)
- libcollector(3)

表 2-11 列出了 Sun Studio 9 性能分析器中的新特性和增强特性。

表 2-11 性能分析工具新特性

特性	描述
新的 Linux 分发	除了 Solaris™ 上的 Sun Studio 9 之外，性能分析器现在还可用于 Linux 上的 Sun Studio 9 中。支持以下 Linux 操作系统： • Java™ Desktop System 1.0 • SuSE Linux Enterprise Server 8 • RedHat Enterprise Linux 3 这两种操作系统上可用的实用程序相同，但有一点除外，即 er_kernel 不包括在 Linux 分发中。Linux 上的 collect 命令更有限。仅可使用基于时钟的分析和堆跟踪；有关详细信息，请参阅 collect 手册页。在 Linux 下可以对多线程应用程序进行分析，但在 RedHat 版的 Linux 操作系统下进行分析时，可以看到当前存在的高数据差异。
数据空间分析	现在，针对 SPARC® 平台的 C 程序可以进行数据空间分析。数据空间分析是一个数据集合，在其中根据导致事件（而不仅仅是之中发生与内存相关的事件的指令）的数据对象引用报告与内存相关的事件，例如缓存缺失。 数据空间分析信息的分析可显示在命令行上，也可显示在分析器 GUI 中，如下所示： • `er_print` 命令具有三个与数据空间分析相关的新选项：`data_objects`、`data_osingle` 和 `data_olayout` • 现在，该分析器包括两个与数据空间分析相关的新标签，即“数据对象”和“数据布局”。如果数据空间配置文件出现在实验中，则这些标签将自动显示。
后续进程	后续进程的记录已得到增强，以包括记录所有后续进程的功能，而不仅仅是使用 fork 和 exec 命令创建的进程及其变异进程。为了支持增强的功能， collect `-F` 命令现在又有了新的选项： `collect -F all`。 使用 `-F all`（并非 `-F on`）处理后续进程，例如系统呼叫均使用代码字母“c”命名。 使用命令行实用程序 `er_print` 或在分析器 GUI 中可显式选择后续进程的数据以便进行显示。 有关详细信息，请参阅 `collect`(1) 手册页。
数据收集输出重定向	collect 命令具有一个新的选项，即 `collect -O` *file*，它可以将所有输出从 `collect` 重定向到已指定的 *file*。该命令不会重定向来自派生目标的输出。
增强的分析器命令行参数	现在，分析器命令（启动脚本）可接受对较长的参数使用双短线，特别是 `--jdkhome` 和 `--fontsize`。
分析器 API 共享库的新软件包	分析器 API 的共享库已置于单独的软件包中，以便对其单独并免费进行分发。

表 2-11 性能分析工具新特性 *(续)*

特性	描述
支持 `collect` 命令的说明文件	`collect` 命令具有新的命令行选项 `collect -C` *comment*。将 *comment* 添加到该实验的说明文件中。可以应用多达 10 个 `-C` 参数。
实验预览和实验标题中的说明	实验预览和实验标题将显示该实验中任何说明文件的内容
增强的源码和反汇编显示	带注释的源码和反汇编改进了替代源上下文中的代码处理。以红色斜体字显示的索引行将指示从另一个文件插入代码的位置。借助 “源码” 标签，在索引行上单击鼠标将打开替代源文件中的 “源码” 窗口。
增强的 `er_src` 命令	现在，命令行实用程序 `er_src` 可以显示函数列表、处理 Java `.class` 文件并显示来自替代源上下文中的源码和反汇编。
Java™ 方法签名	Java™ 长名称格式可显示完整的方法签名，而不仅仅是函数名称本身。
进行堆跟踪时 `mmap` 调用包括的内容	在进行堆跟踪时，对 `mmap` 的调用将按照内存分配来处理。

集成开发环境 (IDE)

以下新功能已添加到 Sun Studio 9 发行版的 IDE 中：

- 新的 `ss_attach` 功能使您能够在其开始执行时即捕获程序，并附加了 dbx 调试器以立即开始对其进行调试，而不是在运行进程后再连接到调试器。
- 源码编辑器中的 “快速浏览” 组合框使您能够导航到类方法、函数、 #define 或源文件的其他元素。

文档

本节介绍了 Sun Studio 9 文档的新特性。

- *OpenMP API 用户指南*已进行扩展，以包括两章新的内容。第 5 章介绍了通过 Fortran 95 __AUTO 子句进行自动数据作用域设置。第 6 章说明了 OpenMP 程序的性能，并提供了一些用于改进性能的方法的一般建议。
- *《C 用户指南》*包含两部分新的附录：*附录 A “按功能分组的编译器选项”* 和 *附录 D “C99 实现定义的信息”*。附录 A 的内容原来在选项参考一章的开头，但现在单独划分成一个附录，以增强其明显程度。

- *性能分析器*手册新增了一章内容，标题为“了解带注释的源码和反汇编数据”。本章介绍了各种不同的注释，如索引行、编译器注释、特殊行 （如外联函数），并说明了如何识别注释与最初源码之间的显示差异。
- Sun 开发人员门户网站上提供了有关性能分析器的教程，网址是：
  `http://developers.sun.com/prodtech/cc`
- *Sun WorkShop to Sun Studio Migration* 帮助集具有有关文件比较和文件合并的新主题。
- 新的*编译器与工具*帮助集介绍了 Sun Studio 发行版中包括的编译器和工具。每个主题均简单说明了某个组件，并列出了该组件的文档。