Overview

This product provides tools for Windows® software developers to create applications that run at top speeds on all Intel® IA-32 processors, Intel processors with Intel® Extended Memory 64 Technology (Intel® EM64T) and Intel Itanium® 2 processors. Optimizations include support for Streaming SIMD Extensions 2 (SSE2) in the Intel Pentium® 4 and Intel Pentium® M processors, Streaming SIMD Extensions 3 (SSE3) in the Intel Pentium 4 and Intel® Core™ processors with SSE3 support, and software pipelining in the Intel® Itanium® 2 processor. Inter-procedural optimization (IPO) and profile-guided optimization (PGO) can provide greater application performance. Intel Compilers support multi-threaded code development through autoparallelism and OpenMP® support.

The Standard Edition of this product consists of the Intel® Visual Fortran Compiler, the Intel® Debugger, integration into Microsoft® visual development environments, and code coverage and test prioritization tools. The Professional Edition includes all of the Standard Edition features and adds the IMSL® Fortran Libraries 5.0 from Visual Numerics.

Compatibility

You must recompile all Fortran sources that were compiled with Intel Fortran versions earlier than 8.0, including those that create .mod files. If you are using third-party libraries and/or .mod files, you must obtain compatible versions from the library vendor. If you encounter difficulties obtaining updated third-party libraries, please let us know through Intel® Premier Support. If you previously
used Compaq* Visual Fortran (CVF), note that the default procedure calling and naming conventions are different from CVF. A document describing CVF porting considerations is available from http://developer.intel.com/software/products/compilers/fwin/ or from Intel® Premier Support.

Product Contents

This product contains the following components:

- Intel® Fortran Compiler for 32-bit applications, Version 9.1
- Intel® Fortran Compiler for Itanium®-based applications, Version 9.1
- Intel® Fortran Compiler for Intel® EM64T-based applications, Version 9.1
- Intel® Fortran Itanium® Compiler for Itanium®-based applications, Version 9.1
- Intel® IA-32 Assembler 8.0 to produce Itanium®-based applications
- Intel® Itanium®-based Assembler 8.0 to produce Itanium®-based applications
- Intel® Debugger 9.1 (See separate Release Notes)
- Utilities
  - Intel® Compilers code-coverage tool
  - Intel® Compilers test-prioritization tool
- Integration into Microsoft* visual development environments
- IMSL* Fortran Libraries 5.0 (included in Professional Edition only)
- On-disk documentation

Note: The default installation master directory referred to as <install-dir> in this document is C:\Program Files\Intel. The Fortran 9.1 compiler is installed into the Compiler\Fortran\9.1 subfolder.

To receive technical support and product updates for the tools provided in this product you need to register, as described in the Technical Support section.

Changes in Version 9.1

The following section discusses new features and changes in the Intel Visual Fortran Compiler version 9.1 and updates to 9.1. Please see the separate release notes for the Intel Debugger.

New and Changed Command Line Options

The following is a listing of command line options that are new or have changed since the initial version 9.0 release. Please refer to the compiler documentation for more information on these options.

/allow:[no]fpp_comments
Determines how the fpp preprocessor treats Fortran end-of-line comments in preprocessor directive lines. (Default: /allow:fpp_comments)

/assume:[no]writeable-strings
Determine whether character constants go into read-only memory.
(Default: /assume:nowriteable-strings)

/fp:keyword
Controls the semantics of floating-point calculations. (default: /fp:fast)

/G2-p9000
Optimizes for Dual-Core Intel® Itanium® 2 Processor 9000 Sequence processors. (Default: off)

/QaxT
Directs the compiler to generate processor-specific code optimized for the Intel processor code named "Merom" (and compatible Intel processors) as well as generic IA-32 code (IA-32 and Intel EM64T only, default: off)

/Qftz[-]
Enables or disables the flushing to zero of floating point calculations that would underflow into the denormal range. The behavior of this option has changed for version 9.1 - please see the note below for more details.

/Qinline-factor=<n>
Specifies the percentage multiplier that should be applied to all inlining options that define upper limits. (Default: off)

/Qinline-forceinline
Specifies that an inline routine should be inlined whenever the compiler can do so. (Default: off)

/Qinline-max-per-compile=<n>
Specifies the maximum number of times inlining may be applied to an entire compilation unit. (Default: off)

/Qinline-max-per-routine=<n>
Specifies the maximum number of times the inliner may inline into a particular routine. (Default: off)

/Qinline-max-size=<n>
Specifies the lower limit for the size of what the inliner considers to be a large routine. (Default: off)

/Qinline-max-total-size=<n>
Specifies how much larger a routine can normally grow when inline expansion is performed. (Default: off)

/Qinline-min-size=<n>
Specifies the upper limit for the size of what the inliner considers to be a small routine. (Default: off)

/Qopt-mem-bandwidth
Enables or disables performance tuning and heuristics that control memory bandwidth use among processors. (Itanium-only, default 0 for serial compilations, 1 for parallel compilations)

/Qsafeseh[-]
Registers exception handlers for safe exception handling. (IA-32 only, default: on if /Qvc7.1 or /Qvc8)
/Qvc8
Specifies compatibility with Microsoft* Visual Studio 2005. (Default depends on choice made during installation)

/QxT
Directs the compiler to generate specialized and optimized processor-specific code for the Intel processor code named "Merom" and compatible Intel processors.

/watch:[no]cmd
Tells the compiler to display and execute driver tool commands. (Default: /watch:nocmd)

/watch:[no]source
Tells the compiler to display the name of the file being compiled. (Default: /watch:nosource)

Integration for Microsoft* Visual Studio* 2005


It is no longer necessary to install a separate Microsoft Platform SDK (Software Development Kit) in order to develop Intel EM64T and Itanium-based applications when using Microsoft Visual Studio 2005 Standard Edition or above or if you selected command-line compatibility with Microsoft Visual Studio 2005 when installing the compiler.

Support for Cross-Platform Development in Microsoft Visual Studio 2005

When installing Microsoft Visual Studio 2005, you have the option of installing cross-platform development tools. Choosing this option installs the necessary files (cross-platform headers, tools, libraries and so forth) within the directory tree. No separate, standalone Platform SDK is required.

If you want to reference an existing installed Platform SDK from within Visual Studio 2005, you can do so in one of two ways:

1. Open the appropriate build environment command window (Start.. All Programs.. Intel Software Development Tools.. Intel Fortran Compiler 9.1..Build Environment for <platform name>) and issue the commands:
"C:\Program Files\Microsoft Platform SDK\SetEnv.bat"
"C:\Program Files\Microsoft Visual Studio
8\Common7\IDE\devenv" /useenv

The first command invokes the environment setup procedure for the Microsoft Platform SDK you are using, and the second starts the Visual Studio development environment using the established environment. Replace the file paths shown with the correct paths for your system.

2. Manually change all pertinent pathnames in Visual Studio's Tools..Options..Projects..Directories to include the appropriate directories from your chosen Platform SDK.

Be sure to install and use the cross-platform tools in a consistent manner. It is possible to create a situation where your command line build is referencing a set of tools and libraries associated with a standalone Platform SDK while your Fortran compiler integrations reference the tools provided with Visual Studio 2005. In a case such as this, compiling from the command line and compiling from within Visual Studio will use different development environments and may produce different results.

Note: A similar situation can occur with application development with native compilers and different versions of Visual Studio. For example, your command line compiler can reference tools and libraries associated with Visual Studio 2005 while your installation of the Visual Studio compiler integrations is associated with an earlier version (for example, Visual Studio .NET 2003).

Manifests in Visual Studio 2005

Manifests are a Visual Studio 2005 feature and are XML files that describe runtime dependencies of a built application. The Intel Visual Fortran Compiler supports generation of manifest files, which are typically placed alongside the EXE or DLL with a file type of .manifest. Manifest files can also be embedded in the EXE or DLL, but this is not done if incremental linking is enabled. In such situations, you will see the following warning:

Warning: Manifest file not embedded because incremental linking property is specified. To embed manifest file, set "Linker>General>Enable Incremental Linking" to "No".

Support for Multiple Compiler Versions in Microsoft Visual Studio .NET 2002 and 2003

Expanding on support provided in version 9.0, you can, within the Microsoft Visual Studio .NET 2002 or 2003 environments, select among different versions of the Intel Fortran Compiler you may have installed on the system. To select the
version of the compiler and libraries you want to use, click on Tools..Options..Intel Fortran..General..Directories. There you will find an option to select from different Intel Fortran Compiler versions. The supported older versions are 9.0 and 8.1. This feature is not available from Visual Studio 2005 because the older compiler versions do not support Visual Studio 2005.

**Context-Sensitive Language Help in Visual Studio**

You can now request documentation on Fortran language features, including library routines, using context-sensitive help in the Visual Studio IDE. In a source window, position the cursor on a language keyword (for example, INTEGER or CONTAINS), or a library routine (for example, DATE_AND_TIME or SIGNALQQ) and press the F1 key. You will then be brought to the appropriate topic in the on-disk documentation.

**Use of /QxP and /QaxP Code Generation Switches**

The /QxP and /QaxP switches specify generation of specialized code for Intel processors supporting the Streaming SIMD Extensions 3 (SSE3), including the Intel® Core™ Duo and Solo processors.

**Constants Are Now Read-Only**

Constants, including literals and named constants (PARAMETER), are now allocated in a memory section that is protected against write access. This means that if a constant is passed as an actual argument to a procedure and the procedure tries to modify the argument, an access violation will result. For example:

```fortran
call sub (3)
...
subroutine sub (i)
i = i + 1 ! Will cause an access violation
```

The Fortran language prohibits changing the definition status of an argument associated with a constant or expression. If your application needs to do so, you can specify the /assume:noprotect_constants option. (In the visual development environment, select project property Fortran..Data..Constant Actual Arguments Can Be Changed..Yes.) This will instruct the compiler to create and pass a temporary copy of the constant actual argument. The called procedure can change this copy which will be discarded when the procedure exits.

**Uninitialized Variable Checking Now Supported in All Contexts**
As of version 9.0.030, uninitialized variable checking (/check:uninit) is supported for programs run in any context, not just when built against the debug libraries and run under the Visual Studio debugger, as was the case in earlier versions. Console applications will display the message in the console window, with traceback if /traceback was also specified. QuickWin and Windows applications will display the message in a message box. The message will be of the form:

```
forrtl: severe (193): Run-Time Check Failure. The variable 'TEST$J' is being used without being defined
```

The variable name may be displayed with an added prefix of the name of the program unit where the error occurred - TEST in this example.

Please note that uninitialized variable checking is currently limited to scalar, local variables. Arrays, COMMON variables and routine arguments are not checked.

**Rename of User-Defined Operators in USE (Fortran 2003 feature)**

A rename clause on a USE statement may now also specify a defined operator. For example:

```
USE mymod, OPERATOR(.localop.) => OPERATOR(.moduleop.)
```

**PROTECTED Attribute and Statement (Fortran 2003 feature)**

The PROTECTED attribute restricts the use of module entities. Other than within the module within which the entity has been given the PROTECTED attribute, the following restrictions apply:

- A non-pointer entity may not be defined or redefined
- A pointer entity may not have its association status changed through the pointer

If an object has the PROTECTED attribute, all of its subobjects have the PROTECTED attribute.

**INTENT Attribute for Pointer Objects (Fortran 2003 feature)**

Pointer dummy arguments may now have the INTENT attribute specified. The INTENT attribute restricts how the association status of the pointer may change - it does not restrict changes to the target of the pointer.

**MOVE_ALLOC Intrinsic Subroutine (Fortran 2003 feature)**
The **MOVE_ALLOC** intrinsic subroutine moves an allocation from one allocatable object to another.

**Syntax:** CALL MOVE_ALLOC (FROM, TO)

**Arguments:**

FROM
- May be of any type and rank and must be allocatable.

TO
- Must be type compatible with FROM, have the same rank, and be allocatable.

If TO is currently allocated, it is deallocated. Then, if FROM is allocated, TO becomes allocated with the same bounds and value identical to that of FROM. Lastly, FROM is deallocated. If TO has the TARGET attribute, any pointer associated with FROM at the time of the call to MOVE_ALLOC is now associated with TO.

The implementation of **MOVE_ALLOC** is such that the internal descriptor contents are copied from FROM to TO, so that the storage pointed to is the same. The typical use of **MOVE_ALLOC** is to provide for an efficient implementation of reallocating an variable to a larger size without copying the data twice.

**ASSUME_ALIGNED Directive**

The **ASSUME_ALIGNED** directive tells the compiler to assume that a variable's memory allocation is aligned on a particular address boundary. This can aid in certain optimizations.

**Syntax:** cDEC$ ASSUME_ALIGNED address1:n1 [ , address2:n2] ...

- address is a memory reference (variable name, array reference, etc.)
- n1 is a positive integer initialization expression with one of the following values: 1, 2, 4, 8, 16, 32, 64, 128, 256
  - address may not be a derived type component nor a variable in COMMON.
  - address may not be use or host associated.
  - If address has the POINTER attribute, or is an integer POINTER variable (using the integer POINTER extension), the presumed alignment applies to the pointer and not its target.

**ABORT Library Routine Now Exits with Non-Zero Status**
A program whose execution is terminated by a call to the 
\texttt{ABORT} library routine, 
defined in module \texttt{IFPORT}, now exits with a status of 134. In previous releases, 
the exit status was set to zero.

\textbf{EQUIVALENCE}d variables cannot be \texttt{DLLIMPORTed} or \texttt{DLLEXPORTed}

The compiler now gives an error message if a variable given the \texttt{DLLIMPORT} or 
\texttt{DLLEXPORT} attribute is named in an \texttt{EQUIVALENCE} specification. This 
combination cannot be supported by the compiler, but previous versions did not 
give an error for it. Alternatives include a \texttt{UNION} inside a \texttt{STRUCTURE} or use of 
the \texttt{TRANSFER} intrinsic.

\textbf{Buffering Now Supported for Direct-Access I/O}

You may now enable buffering for units opened for direct-access I/O. Buffering 
causes the run-time library to read or write multiple records in a single disk 
operation. This can greatly improve run-time performance for applications that 
use direct-access I/O to access records in order, but may harm performance for 
applications that access records out-of-order. Buffering can be enabled or 
disabled in the same manner as for sequential I/O, including the \texttt{BUFFERED} 
keyword for \texttt{OPEN}, the \texttt{/assume:buffered_io} compile command option, and 
the \texttt{FORT_BUFFERED} environment variable. Buffering of direct-access I/O is not 
supported when the \texttt{/fpscomp:general} option is specified.

\textbf{Fortran 2003 Feature Summary}

The Intel Fortran Compiler supports many features that are new to the latest 
revision of the Fortran standard, Fortran 2003. Additional Fortran 2003 features 
will appear in future versions. Fortran 2003 features supported by the current 
compiler include:

- The Fortran character set has been extended to contain the 8-bit ASCII 
  characters \(~ \backslash [ ] \backslash ^ { } \{ \} \) | # @
- Names of length up to 63 characters
- Statements of up to 256 lines
- Square brackets [ ] are permitted to delimit array constructors instead of (/ /
  )
- \texttt{GET\_COMMAND} intrinsic
- \texttt{GET\_COMMAND\_ARGUMENT} intrinsic
- \texttt{COMMAND\_ARGUMENT\_COUNT} intrinsic
- \texttt{GET\_ENVIRONMENT\_VARIABLE} intrinsic
- \texttt{MOVE\_ALLOC} intrinsic
- The following intrinsics take an optional \texttt{KIND=} argument: \texttt{ACHAR, COUNT, IACHAR, ICHAR, INDEX, LBOUND, LEN, LEN\_TRIM, MAXLOC, MINLOC, SCAN, SHAPE, SIZE, UBOUND, VERIFY}
- Allocatable components of derived types
- Allocatable dummy arguments
- Allocatable function results
- PROTECTED attribute and statement
- VOLATILE attribute and statement
- INTENT attribute for pointer objects
- A named PARAMETER constant may be part of a complex constant
- In all I/O statements, the following numeric values can be of any kind: UNIT=, IOSTAT=
- The following OPEN numeric values can be of any kind: RECL=
- The following READ and WRITE numeric values can be of any kind: REC=, SIZE=
- The following INQUIRE numeric values can be of any kind: NEXTREC=, NUMBER=, RECL=, SIZE=
- Recursive I/O is allowed in the case where the new I/O being started is internal I/O that does not modify any internal file other than its own
- IEEE Infinities and NaNs are displayed by formatted output as specified by Fortran 2003
- In an I/O format, the comma after a P edit descriptor is optional when followed by a repeat specifier
- Rename of user-defined operators in USE

**Intel® Array Visualizer Removed from Product**

As of version 9.1 of the Intel® C++ and Fortran Compilers for Windows, the Intel® Array Visualizer component will no longer be included in the compiler products. The most recent version of Array Visualizer will be made available as a free, unsupported download and will no longer require an Intel compiler license to use. Intel will not further maintain nor develop Array Visualizer. The Array Visualizer user forum will continue to be available at [http://softwareforums.intel.com/ids/board?board.id=Visualizer](http://softwareforums.intel.com/ids/board?board.id=Visualizer) where users can share experiences and usage tips. Please visit the forum for more information.

**Microsoft* Visual Studio* .NET 2002 Support Deprecated**

In a future release of the Intel C++ and Fortran Compilers, Intel intends to remove support for integration with Microsoft Visual Studio .NET 2002. If you are using Visual Studio .NET 2002, we recommend that you plan to upgrade to Visual Studio .NET 2003 or Visual Studio 2005 by the end of calendar year 2006.

**IMSL* Fortran Libraries 5.0 (Professional Edition Only)**

The IMSL* Fortran Libraries 5.0, provided in Intel Visual Fortran Compiler Professional Edition, have been updated for this release to include corrections for reported problems.
Please refer to the section on using IMSL in the *Using Libraries* section of the *Intel Fortran Building Applications Manual* for additional information, including configuration for use in the Microsoft IDE.

It is no longer necessary to separately run `cttsetup.bat` to define IMSL environment variables as this is done automatically when `ifortvars.bat` is run for a command line build environment session.

**System Requirements**

**Processor Terminology**

Intel compilers support three platforms: general combinations of processor and operating system type. This section explains the terms that Intel uses to describe the platforms in its documentation, installation procedures and support site.

**IA-32**

IA-32 (Intel Architecture, 32-bit) refers to systems based on 32-bit processors generally compatible with the Intel Pentium® II processor, (for example, Intel® Core™ Duo, Pentium® 4, Pentium® D, Celeron® or Intel® Xeon®), or processors from other manufacturers supporting the same instruction set, running a 32-bit operating system.

**Intel EM64T**

Intel® EM64T (Intel® Extended Memory 64 Technology) refers to systems based on IA-32 processors which have 64-bit architectural extensions, running a 64-bit operating system such as Microsoft* Windows* XP Professional x64 Edition or Microsoft Windows Server 2003 x64 Edition. Systems based on the AMD* Athlon64* and Opteron* processors running a 64-bit operating system are also supported by Intel compilers for EM64T-based applications.

**Intel Itanium®**

Refers to systems based on the Intel Itanium® 2 processor running a 64-bit operating system.

**Native and Cross-Platform Development**

The term "native" refers to building an application that will run on the same platform that it was built on; for example, building on IA-32 to run on IA-32. The term "cross-platform" or "cross-compilation" refers to building an application on a platform type different from the one on which it will be run, for example, building on IA-32 to run on Intel® Itanium®-based systems. Not all combinations of cross-platform development are supported and some combinations may require installation of optional tools and libraries.
The following table describes the supported combinations of compilation host (system on which you build the application) and application target (system on which the application runs).

<table>
<thead>
<tr>
<th>Host \ Target</th>
<th>IA-32</th>
<th>Intel® EM64T</th>
<th>Intel® Itanium®</th>
</tr>
</thead>
<tbody>
<tr>
<td>IA-32</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Intel® EM64T</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Intel® Itanium®</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Note:** The above table refers to use of the command-line build environment. Microsoft Visual Studio .NET 2002 and 2003 support development of IA-32 applications only. Cross-platform development of Itanium-based applications is supported in Microsoft Visual Studio 2005 Team System Edition only. Visual Studio 2005 does not support installation on Itanium-based systems.

**Minimum Hardware Requirements to Develop Applications**

- A system based on an IA-32 processor (minimum 450 MHz Intel® Pentium® II processor or greater - Intel® Core™ Duo or Solo, Intel® Pentium® 4 or Pentium® D or Intel® Xeon® processor recommended), or a system based on an Intel® Itanium® 2 processor, or a system based on an Intel processor with Intel EM64T, or a system based on an AMD* Athlon* or AMD Opteron* processor
- For IA-32 systems, 256 MB of RAM (512 MB recommended)
- For other systems, 512MB of RAM (1GB recommended)
- 300 MB of free hard disk space, plus an additional 300 MB during installation for download and temporary files
- 100 MB of hard disk space for the virtual memory paging file. Be sure to use at least the minimum amount of virtual memory recommended by the operating system.

**Software Requirements to Develop IA-32 Applications**

  **Note:** Microsoft Windows 98, Windows 98 SE, Windows Millennium Edition and Windows NT are not supported for product development, but are supported for application deployment.
- One of the following Microsoft development products must be installed:
  - Microsoft Visual C++® .NET® 2002 or 2003, Standard edition or above
  - Microsoft Visual Studio® .NET® 2002 or 2003, any edition, with Visual C++ component installed
Microsoft Visual Studio 2005, Standard edition or above, with Visual C++ component installed (Note: Microsoft Visual Studio 2005 Express Edition is supported for command line development only. Express Edition users are advised to also install the Microsoft Platform SDK if Win32 APIs will be used.)

• For development of IA-32 applications on Windows XP Professional x64 Edition or Windows Server 2003 x64 Edition, only the 2003 and 2005 versions of the Microsoft development environments are supported.

Software Requirements to Develop Applications for Systems with Intel EM64T or AMD Opteron Processors

• Microsoft Windows 2000, Windows XP or Windows Server 2003
• For development using the Visual Studio environment or the command line tools, Microsoft Visual Studio 2005, Standard edition or above, with Visual C++ and "X64 Compiler and Tools" components installed (Visual Studio 2005 Standard Edition does not require explicit selection of the X64 component.)
• For development using the command line tools only, Windows Server 2003 SP1 Platform SDK Follow the link for the Platform SDK Download Site and run PSDK-amd64.exe to begin the PSDK installation. Only the "Core SDK" is required.

Software Requirements to Develop Itanium-based Applications

• Microsoft Windows 2000, Windows XP or Windows Server 2003
• For development using the Visual Studio environment or the command-line tools on IA-32 or Intel EM64T systems, Microsoft Visual Studio 2005, Team System edition or above, with Visual C++ and "Itanium Compiler and Tools" components installed
• For development using the command line tools only, Windows Server 2003 SP1 Platform SDK Follow the link for the Platform SDK Download Site and run PSDK-ia64.exe to begin the PSDK installation. Only the "Core SDK" is required.

Requirements to Run Applications

• For applications built for systems with Intel EM64T: a system based on a processor with Intel EM64T or AMD Opteron processor running Windows Server 2003 x64 Edition or Windows XP Professional x64 Edition
• For applications built for Intel Itanium-based systems: a system based on an Intel Itanium 2 processor running Windows Advanced Server or Windows Server 2003 (Enterprise and Datacenter Editions)
• Running applications on systems that do not have Intel Visual Fortran Compiler installed may require installation of redistributable DLLs on the target system.

Notes:

• The above lists of processor model names are not exhaustive - other processor models correctly supporting the same instruction set as those listed are expected to work. Please contact Intel® Premier Support if you have questions regarding a specific processor model
• Some optimization options have restrictions regarding the processor type on which the application is run. Please see the documentation of these options for more information.
• Advanced optimization options or very large programs may require additional resources such as memory and disk space
• Adobe* Acrobat Reader* version 5.0 or later is required to view some of the reference documentation.

It is the responsibility of application developers to ensure that the machine instructions contained in the application are supported by the operating system and processor on which the application is to run.

Installation

Please see the separate Installation Guide for installation and configuration instructions.

Known Issues

Installation related limitations

• When Visual C++.NET is installed, if the user selects the option to update system environment variables, the user variables for the installing username may also be updated. Intel® Visual Fortran updates only the system variables, as recommended by Microsoft in its documentation for software developers. The effect of this is that if the user who installs Intel Visual Fortran then tries to build a Fortran application from a command prompt, without using the preset environment shortcut provided or invoking ifortvars.bat, default libraries, include files and modules will not be found. The workaround is to delete the user environment variables INCLUDE and LIB. This problem does not affect builds done from the IDE.
• If you uninstall the Intel Visual Fortran 8.1 integration into Microsoft Visual Studio .NET, this action will remove `deftofd.exe`, a tool for converting resource files to Fortran include files, from the 8.1 compiler's `BIN` area. This will be an issue if you use the Selected Compiler Version option in Visual Studio .NET to use the 8.1 compiler and your project uses resource files. To fix this problem, copy `deftofd.exe` from the 9.1 compiler's `IA32\BIN` folder to the corresponding 8.1 folder.

**Building Visual Studio Projects from the Command Line**

If you have a Visual Studio-created Intel Fortran project or solution and wish to initiate a build from the command line or a batch file, use `devenv.exe`, the Visual Studio launcher. Type `devenv /?` for available command line options. The Microsoft tools MSBuild and Team Build cannot currently be used to build Intel Fortran projects.

**Static, Single-Threaded Libraries Not Available in Visual Studio 2005**

In Microsoft Visual Studio 2005, the static, single-threaded Visual C++ libraries `libc.lib` and `libcd.lib` have been removed. If you specified that the Intel Fortran Compiler should use Visual Studio 2005 for command-line building, and attempt to build an application with the static, single-threaded library, which was the default in earlier versions, you will get a warning from the compiler as follows:

```
ifort: warning: option '/Qvc8' or higher used with '/ML[d]' is not supported
```

Note that `/Qvc8` was added to `ifort.cfg` at install time if you specified command line integration with Visual Studio 2005. This tells the `ifort` compiler driver that you are using Visual Studio 2005.

If you attempt to link such an application you will get an error from the linker that `libc.lib` or `libcd.lib` is not found. For example:

```
LINK : fatal error LNK1104: cannot open file 'LIBC.lib'
```

To resolve this, specify the threaded and/or DLL forms of the run-time libraries. For example:

```
/libs:static /threads
/libs:dll
/MT
```

If you do not specify otherwise, the `ifort` default for use with Visual Studio 2005 is `libs:static /threads` which is the same as `/MT`. 
This issue may also affect projects which are built from the Microsoft Visual Studio IDE. In this case, change the project property Fortran..Libraries..Use Run-Time Library to the desired new value.

**Active Platform Dropdown is Empty in Visual Studio 2005**

In Visual Studio 2005, after adding a new project platform (x64 or Itanium), the active platform dropdown list in the toolbar may be empty. As a workaround, click on the Output window to make it active, and then the dropdown list will appear.

**Security Warning in Visual Studio 2005**

In some editions of Visual Studio 2005, when you create a new Fortran project you may see the following warning displayed by the Application Wizard:

To help protect your security, your web browser has restricted this file from showing active content that could access your computer. Click here for options...

You can disregard the warning and click Next - the project will be created normally. If you wish, you can click on the warning, select Allow Blocked Content, and then click Yes when another security warning appears. This warning appears the first time a project is created in a new Visual Studio session. The cause of the warning is under investigation.

**Link Error for Itanium-based or Intel EM64T-based Systems When Using Platform SDK**

When building for Itanium-based systems or Intel EM64T-based systems using the Microsoft Platform SDK, applications may fail to link with errors such as the following:

```
LIBC.lib(a_str.obj) : error LNK2001: unresolved external symbol __security_cookie
[...]
LIBC.lib(a_str.obj) : error LNK2001: unresolved external symbol __security_check_cookie
[...]
```

This problem is due to an inconsistency within some versions of the Microsoft Platform SDK. Intel is working with Microsoft to resolve this issue. To work around the problem, link your application in one of the following ways:

- `ifort hello.f90 /MD`
- `ifort hello.f90 -link bufferoverflowu.lib`
- `ifort hello.f90 /MT bufferoverflowu.lib`
Limited Debug Information with Automatic CPU Dispatching (/Qax*)

Compilation using /Qax{W|N|B|P} results in two copies of generated code for each function: one for IA-32 generic code and one for CPU specific code. The symbol for each function then refers to an Auto CPU Dispatch routine that decides at run-time which one of the generated code sections to execute. Debugger breakpoints that are set on these functions by name cause the application to stop in the dispatch routine. This may cause unexpected behavior when debugging. This issue may be addressed in a future version of the Intel Debugger and Compilers.

Cannot Debug or View Traceback for IA-32 Programs Built with /Oy-

Compilation using /Oy- specifies that the IA-32 EBP register be used as a general purpose register, eliminating its use as a frame pointer. Debuggers and traceback handlers may not be able to properly unwind through a stack that contains a call to a function that is compiled in this manner.

POSIX* Library routine PXFGETPPID Not Supported on Windows NT 4.0

Applications which use the POSIX* library routine PXFGETPPID will not run on Windows NT 4.0. Windows 2000 or later is required for such applications.

Extra Floating Point Exception Visible When Debugging

On IA-32 systems, for programs compiled with the -fpe:0 option that encounter a floating point exception, one extra floating overflow exception will be raised deliberately by the Fortran runtime library to determine which version of the Microsoft libraries is linked to the user's program. This extra exception is raised only once per program process. When in the debugger, users should not be alarmed to see an expected underflow exception followed by an unexpected overflow exception.

Technical Support

Your feedback is very important to us. To receive technical support for the tools provided in this product and technical information including FAQ's and product updates, you need to be registered for an Intel® Premier Support account on our secure web site, https://premier.intel.com. Please register at https://registrationcenter.intel.com/.

- Registering for support varies for release products or pre-release products (alpha, beta, etc) - only released products have support web pages on http://support.intel.com/.
If you are having trouble registering or are unable to access your Intel®
Premier Support account, please let Intel know of the problem at
https://registrationcenter.intel.com/support.

Note: If your distributor provides technical support for this product, please
contact them for support rather than Intel.

For information about the Intel® Visual Fortran Compiler's Users Forums, FAQs,
tips and tricks, and other support information, please visit:
http://support.intel.com/support/performance/tools/fortran/windows/. For general
support information please visit http://www.intel.com/software/products/support/.

Submitting Issues

Steps to submit an issue:

2. Log in to the site. Note that your username and password are case-
sensitive.
3. Click on the "Go" button next to the "Product" drop-down list.
4. Click on the "Submit Issue" link in the left navigation bar.
5. Choose "Development Environment (tools, SDV, EAP)" from the
"Product Type" drop-down list.
6. If this is a software or license-related issue, choose "Intel(R) Fortran
Compiler for Windows*" from the "Product Name" drop-down list.
7. Enter your question and complete the fields in the windows that follow to
successfully submit the issue.

Note: Please notify your support representative prior to submitting source code
where access needs to be restricted to certain countries to determine if this
request can be accommodated.

Guidelines for problem report or product suggestion:

1. Describe your difficulty or suggestion.
   For problem reports please be as specific as possible, so that we may
   reproduce the problem. For compiler problem reports, please include the
   compiler options and a small test case if possible.
2. Describe your system configuration information.
   You can obtain the Package ID information as follows: Select
   Start..All Programs..Intel(R) Software Development
   Tools..Intel(R) Fortran Compiler 9.1..Build
   Environment for IA-32 Applications (substitute EM64T-based
   or Itanium(R)-based as applicable). Type the command:
   ifort /what
   and copy the "Package ID" (e.g. w_fc_p_9.1.xxx) from the output into
the corresponding Intel® Premier Support field. Please include any other specific information that may be relevant to helping us to reproduce and address your concern.

3. If you were not able to install the compiler or cannot get the Package ID, enter the filename you downloaded as the package ID.

Resolved Issues

Please review <package ID>_README.TXT (e.g. w_fc_p_9.1.xxx_README), available for download from Intel® Premier Support, https://premier.intel.com, to see which issues have been resolved in the latest version of the compiler.

Compiler Error Source Reducer (CESR)

Compiler Error Source Reducer (CESR) is a set of utilities which are useful individually or collectively in gathering, reducing, pinpointing, protecting, documenting, and bundling test cases comprised of C/C++ or Fortran source code. It can be helpful if you have a large application for which you want to extract a small test case for submission to Intel® Premier Support. CESR can be downloaded from Intel® Premier Support File Downloads - search for text CESR in the file description. CESR is unsupported and should be considered experimental, but we welcome your feedback on it through Intel® Premier Support. CESR requires prior installation of Python 2.2 or newer.

Documentation

You can view the Intel compiler and related HTML-based documentation with a web browser that supports the Compiled HTML Help (.CHM) format, which provides full navigation, index look-up, search, and hyperlink capabilities. If your browser does not support opening .CHM files, you can open them directly by double-clicking on the file names in Windows Explorer in the Docs directory.

The documentation index is provided for easy access of all documents. The Document index is available from the Intel® Visual Fortran Compiler program folder and is located at: <install-dir>Compiler\Fortran\9.1\Docs\Doc_Index.htm. For this release, the online help has been reorganized as described in the Doc_Index.htm file and now includes a Getting Started guide as well as a separate Installation Guide. The Intel® Debugger Manual is available from the Intel® Debugger program folder.

Additional Information

Related Products and Services
Information on Intel software development products is available at 

Some of the related products include:

- The **Intel® Software College** provides training for developers on leading-edge software development technologies. Training consists of online and instructor-led courses covering all Intel architectures, platforms, tools, and technologies.
- The **Intel® VTune™ Performance Analyzer** enables you to evaluate how your application is utilizing the CPU and helps you determine if there are modifications you can make to improve your application's performance.
- The **Intel® C++ and Fortran Compilers** are an important part of making software run at top speeds with full support for the latest Intel IA-32 and Itanium® processors.
- The **Intel® Performance Library Suite** provides a set of routines optimized for various Intel processors. The **Intel® Math Kernel Library**, which provides developers of scientific and engineering software with a set of linear algebra, fast Fourier transforms and vector math functions optimized for the latest Intel Pentium® and Intel Itanium processors. The **Intel® Integrated Performance Primitives** consists of cross-platform tools to build high performance software for several Intel architectures and several operating systems.

**Disclaimer and Legal Information**

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice.
Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The software described in this document may contain software defects which may cause the product to deviate from published specifications. Current characterized software defects are available on request.

This document as well as the software described in it is furnished under license and may only be used or copied in accordance with the terms of the license. The information in this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Intel Corporation. Intel Corporation assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document. Except as permitted by such license, no part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the express written consent of Intel Corporation.


* Other names and brands may be claimed as the property of others.

Copyright © 2006, Intel Corporation.