

Intel® Parallel Inspector Release Notes

Installation Guide and Release Notes
Document number: 320754-002US

Contents:

[Introduction](#)
[What's New](#)
[System Requirements](#)
[Installation Notes](#)
[Issues and Limitations](#)
[Attributions](#)
[Disclaimer and Legal Information](#)

1 Introduction

The Intel® Parallel Inspector is a multithreading error checking tool for Microsoft Visual Studio* C/C++ developers. The tool detects challenging threading and memory errors and provides guidance to help ensure application reliability. Unlike other error checkers on the market, Inspector is an easy, comprehensive method to pinpoint latent multithreading and memory errors.

This document provides system requirements, installation instructions, issues and limitations, and legal information.

To learn more about this product's:

- Documentation at: `<install-dir>\documentation\<locale>\documentation_inspector.htm`.
- Technical support, including answers to questions not addressed in the installed tool. Visit the technical support forum at: <http://software.intel.com/sites/support/>

Please remember to register your tool at <https://registrationcenter.intel.com/> by providing your email address. This helps Intel recognize you as a valued customer in the support forum.

2 What's New

Update 2:

- Supports analysis on Microsoft Windows 7* operating system.
- Can detect deadlocks involving the critical construct and locking API in OpenMP* programs if they are using an Intel OpenMP* runtime library.

See <http://software.intel.com/en-us/intel-parallel-inspector/> or the What's New section in the help.

3 System Requirements

For an explanation of architecture names, see <http://software.intel.com/en-us/articles/intel-architecture-platform-terminology/>

- A PC based on an IA-32 or Intel® 64 architecture processor supporting the Intel® Streaming SIMD Extensions 2 (Intel® SSE2) instructions (Intel® Pentium® 4 processor or later, or compatible non-Intel processor)
 - Incompatible or proprietary instructions in non-Intel processors may cause the analysis capabilities of this tool to function incorrectly. Any attempt to analyze code not supported by Intel® processors may lead to failures in this tool.
 - For the best experience, a multi-core or multi-processor system is recommended.
- 2GB RAM
- 4GB free disk space for all tool features and architectures
- Microsoft Windows 7*, Microsoft Windows XP*, Microsoft Windows Vista*, Microsoft Windows Server* 2003 or Microsoft Windows Server* 2008, 32-bit or "x64" editions - embedded editions not supported
- Microsoft Visual Studio* 2005 or 2008 Standard Edition (or higher edition) with C++ component installed[1]
- Application coding requirements
 - Programming Language: C or C++ (native, not managed code) [1]
 - Threading methodologies supported by the analysis tool:
 - Intel® Threading Building Blocks
 - Win32* Threads
 - OpenMP*[1]
 - Intel's C/C++ Parallel Language Extensions
- Adobe* Reader* 7.0 or later to read installed documentation,

Notes:

1. Inspector supports analysis of applications built with Intel® Parallel Composer, Intel® C++ Compiler Professional Edition version 10.0 or higher, and/or Microsoft Visual C++*

2005 or 2008. Applications that use OpenMP* technology and are built with the Microsoft* compiler must link to the OpenMP* compatibility library as supplied by an Intel® compiler.

4 Installation Notes

If you are installing Inspector for the first time, please be sure to have the product serial number available so you can type it in during installation.

Updates of Inspector will uninstall your current version, and will use the existing valid license on the system.

Default Installation Folders

The default top-level installation folder for Inspector is:

```
C:\Program Files\Intel\Parallel Studio\Inspector
```

If you are installing on a system with a non-English language version of the Windows* operating system, the name of the `Program Files` folder may be different. On Intel® 64 architecture systems, the folder name is `Program Files (X86)` or the equivalent.

Changing, Updating and Removing the Tool

To remove, modify, or repair Inspector:

2. Open the Control Panel.
3. Select the **Add or Remove Programs** applet.
4. Select **Intel Parallel Inspector**.
5. Click the **Change** button.

5 Issues and Limitations

Installation

- Inspector may not install correctly if installation of another tool that integrates with Microsoft Visual Studio* software is in progress.

General Intel® Parallel Inspector Issues

- Intel does not guarantee this software tool will detect or report every memory and threading error in an application.
 - Not all logic errors are detectable.
 - Heuristics used to eliminate false positives may hide real issues.
 - Highly correlated events will be grouped into a single problem.
- Inspector can be used on most native binaries, both debug and released versions. However, applications compiled with full debug information result in the most complete information being reported that allows easy viewing of the source code relevant to problems detected. The Microsoft Visual Studio* default debug configuration has these preferred compiler settings: `/ZI`, `/Od`, `/MDd`.
- If no symbols are found for a module in which a problem is detected, Inspector displays the call stack and observation source code of the first location where it can find symbols. If it cannot find any location in the call stack with symbols, it displays the module name and relative virtual address (RVA) for the location. To ensure full symbolic information is available, compile with `/ZI` or `/Zi` [Debug Information Format=Program Database (for Edit & Continue)] and link with `/DEBUG` (Generate Debug Information=Yes).
- Inspector will analyze only one process in an application. This will be the initial process created by the execution of the targeted application. This means that an application launched by a script will result in the analysis of the script, not the process which the script starts.
- Applications that crash when run outside Inspector may crash or hang the Inspector runtime analysis engine. For example, a corrupt return address on an application call stack crashes the runtime analysis engine. If a crash occurs, problems detected prior to that time can be viewed, but memory leaks will not be reported.
- Inspector uses a socket to communicate between the graphical user interface and the runtime analysis engine. Preventing an application from opening a socket prevents Inspector from analyzing the application.
- Inspector may report an incorrect call stack following an interruption of normal call flow, such as when an exception is thrown and caught. While Inspector recognizes and attempts to correct result data when this situation occurs, it is possible for a threading or memory problem to be reported before the call stack is fully corrected.

- Inspector Help may be unavailable in Microsoft Visual Studio* software if the language for non-Unicode programs does not match the operating system language: for example, the Japanese Windows* operating system with English language set for non-Unicode programs. Workaround: Configure the language for non-Unicode programs to match the operating system language (choose **Control Panel > Regional and Language Options >Tab: Advanced**).

Threading Error Analysis

- Parallel Inspector may report false positives and false negatives when analyzing applications that call Microsoft Windows* thread pool APIs (first introduced in Microsoft Windows Vista*) or User-Mode scheduling(UMS) APIs (first introduced in Microsoft Windows 7*).
- Inspector does not detect deadlocks or potential deadlocks created with some types of locks via Intel's C/C++ parallel extension (`__critical`) provided by the Intel® Parallel Composer, some types of locks in Intel® Threading Building Blocks (`spin_mutex`, `spin_rw_mutex`) or non-exclusive ownership synchronization objects involved, for example, condition variables, semaphores and events etc
- Inspector may not detect threading issues on data accessed in the C runtime library (like `memmove` and `memcpy`).
- Inspector does not detect data races or deadlock/potential deadlocks when multiple processes are involved.
- Inspector does not capture the main thread creation site if the `.pdb` symbol file is not in the location specified within the `.exe` or `.dll` executable file or in the location containing the `.exe` or `.dll` executable file.
- Inspector may report false positives if you don't use the dynamic version of the Microsoft* C runtime library specified by setting the runtime library compiler option to `/MD` or `/MDd`.
- Inspector may report false positives for analyzed applications using customized synchronization primitives.
- To reduce false positives and false negatives when analyzing applications using Intel® Threading Building Blocks, include `"TBB_USE_THREADING_TOOLS"` in the list of compilation Preprocessor Definitions (`/D TBB_USE_THREADING_TOOLS`).

- Synchronization, function calls and memory loads/stores that take place before Inspector takes control of the program are not visible to Inspector. Missing these events can potentially cause the tool to report false positives in the application. This situation can occur if these constructs occur in DIIMain.

Memory Error Analysis

- On 64-bit version of Windows 7* operating system, Parallel Inspector may show incorrect call stacks associated with memory leaks detected by “check for memory errors level 1” analysis type. Any stack frames corresponding to functions in libraries/executables that call `LoadLibrary()` will be missing in call stacks associated with memory leaks. As a workaround, analyze your application using “check for memory errors level 2” or higher analysis type.
- Inspector does not report memory leaks when using the low (mi1) analysis setting if the application under analysis circumvents the normal termination flow and does not call `ExitProcess()` (which is a call normally made by the runtime library when the application’s main function ends). You can work around this problem and obtain a report of memory leaks by using any of the higher memory analysis settings (mi2, mi3, and mi4).
- Inspector does not report memory as leaked if a pointer to the memory is available in the application memory space at the time the application exits, because the application has the ability to free this memory. For example, if an application allocates a block of memory and stores a pointer to the memory in a global variable, this memory is not included in a list of reported memory leaks. Only memory that has no pointer to it is considered leaked.
- Inspector may report false positives when the analyzed application uses custom memory allocators.
- In some circumstances, Inspector does not record the deallocation of memory freed during application shutdown. For example, Inspector may not record the event if memory is freed from the destructor of an object that is located in global memory, and that destructor does not execute until late in the shutdown process. This can lead to such memory being reported as a memory leak.
- The behavior of Inspector is unknown and could lead to abnormal termination of the analysis if the semantics of standard C runtime allocators are changed (the application is using non-standard versions) such that the memory returned by the allocator is initialized.
- Inspector may report mismatched allocation/deallocation for an array that appears correct with an allocation of `new type[]` and a matching `delete[]` if the code uses `#include <new.h>`. This occurs because the underlying implementation brought in by this include file may not actually use a matched deallocation in order to support backwards compatibility. Codes that use `#include <new.h>` are non-conforming C++ applications. They can be made to conform by using `#include <new>`, which eliminates this problem. Otherwise, you can suppress it.

Command-line

- The help available in PDF format at `install-dir\documentation\en\help_inspector.pdf` does not contain help on the command line. The help for the command line is available from the Microsoft Visual Studio* Help menu, choose **Intel Parallel Studio > Parallel Studio Help > Inspector Help**.
- Options that are put in a file and passed to `insp-cl` using the `-option-file` are not allowed the same syntax alternatives as when entering them on the command line. The restrictions are as follows:
 - Must put a newline char after the final line in the file, otherwise the final character gets duplicated
 - Must use '=' between option name and its value(s)

To obtain more detailed information, please refer to Technical Support.

6 Attributions

wxWindows Library

This tool includes wxWindows software which can be downloaded from <http://www.wxwidgets.org/downloads>.

wxWindows Library Licence, Version 3.1

=====

Copyright (C) 1998-2005 Julian Smart, Robert Roebling et al

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

WXWINDOWS LIBRARY LICENCE TERMS AND CONDITIONS FOR COPYING,
DISTRIBUTION AND MODIFICATION

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public Licence as published by the Free Software Foundation; either version 2 of the Licence, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public Licence for more details.

You should have received a copy of the GNU Library General Public Licence along with this software, usually in a file named COPYING.LIB. If not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.

EXCEPTION NOTICE

1. As a special exception, the copyright holders of this library give permission for additional uses of the text contained in this release of the library as licenced under the wxWindows Library Licence, applying either version 3.1 of the Licence, or (at your option) any later version of the Licence as published by the copyright holders of version

3.1 of the Licence document.

2. The exception is that you may use, copy, link, modify and distribute under your own terms, binary object code versions of works based on the Library.

3. If you copy code from files distributed under the terms of the GNU General Public Licence or the GNU Library General Public Licence into a copy of this library, as this licence permits, the exception does not apply to the code that you add in this way. To avoid misleading anyone as to the status of such modified files, you must delete this exception notice from such code and/or adjust the licensing conditions notice accordingly.

4. If you write modifications of your own for this library, it is your choice whether to permit this exception to apply to your modifications. If you do not wish that, you must delete the exception notice from such code and/or adjust the licensing conditions notice accordingly

Libxml2

Except where otherwise noted in the source code (e.g. the files hash.c,list.c and the trio files, which are covered by a similar license but with different Copyright notices) all the files are:

Copyright (C) 1998-2003 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHERIN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

Boost

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF

MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

7 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.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

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 information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting Intel's Web Site. 64-bit computing on Intel architecture requires a computer system with a processor, chipset, BIOS, operating system, device drivers and applications enabled for Intel® 64 architecture. Performance will vary depending on your hardware and software configurations. Consult with your system vendor for more information.

Intel and Pentium are trademarks of Intel Corporation in the U.S. and other countries.

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

Copyright © 2009, Intel Corporation. All rights reserved.