



MEMORY AND THREADING ERROR CHECKER

Intel® Parallel Inspector 2011

Product Brief

Intel® Parallel Inspector 2011



“Intel® Parallel Inspector and Intel® Parallel Amplifier greatly simplified the task of finding hotspots and memory leaks. We were pleased with the 2X overall performance improvement and the elimination of several previously unidentified memory leaks.”

*Vlad Romashko
Software Development Manager
OpenCascade S.A.S.*

Increase Application Reliability and Quality by Catching Memory and Threading Errors Early

Intel® Parallel Inspector 2011, a dynamic analysis tool for serial and parallel applications, combines memory and threading error checking in one powerful tool. It helps increase the application reliability and quality of Microsoft Visual Studio* C/C++ applications. Using dynamic instrumentation, Intel Parallel Inspector makes it easier to test code more often, without the need to use special test builds or compilers.

- Find memory and threading errors in serial and parallel code with one easy-to-use tool.
- Deliver reliable, higher quality applications, and increase customer satisfaction.
- Give both experts and novices greater insight into threaded code behavior.
- Easily find latent bugs in threaded programs.
- Reduce development and support costs, enhance productivity, and speed time-to-market.

Memory and Thread Checking in One Easy-to-Use Tool

Both memory and thread checking are fully integrated into Microsoft Visual Studio with an easy-to-use interface. Intel Parallel Inspector provides root-cause analysis of crash-causing threading and memory defects. This analysis, combined with problem set analysis that summarizes related bugs, makes this a comprehensive tool for finding memory and threading errors.

Dynamic Instrumentation That Works on Standard Builds and Binaries

Intel Parallel Inspector utilizes dynamic instrumentation to acquire test data and doesn't require special builds, add-ins, or compilers. Since it only instruments the code that's executed, analysis can run in less time and work on larger applications. It can even find errors in binaries without having the source code.

A Memory and Thread Dynamic Analysis Tool

Intel Parallel Inspector is a comprehensive dynamic analysis tool for serial and parallel code, making it easier and faster to find memory and thread errors. Intel Parallel Inspector detects memory leaks, invalid memory read/write issues, dangling pointers, use of uninitialized data, and data races and deadlocks.

Excellent Value

Intel Parallel Inspector's analysis feature helps developers deliver reliable software, while reducing development cost and speeding time-to-market. Included in Intel® Parallel Studio, Intel Parallel Inspector is a comprehensive tool suite for developing, debugging, verifying, and tuning threaded C/C++ applications.

ID	Problem	Sources	Modules	Object Size	State
P1	Uninitialized memory access	main.cpp	worstcodeever.exe		Not fixed
P2	Uninitialized memory access	main.cpp	worstcodeever.exe		Not fixed
P3	Mismatched allocation/deallocation	main.cpp	worstcodeever.exe		Not fixed
P4	Mismatched allocation/deallocation	main.cpp	worstcodeever.exe		Not fixed
P5	Invalid memory access	main.cpp	worstcodeever.exe		Fixed
P6	Invalid memory access	main.cpp	worstcodeever.exe		Not fixed
P7	Invalid memory access	main.cpp	worstcodeever.exe		Not fixed
P8	Invalid memory access	main.cpp	worstcodeever.exe		Not fixed
P9	Memory leak	main.cpp	worstcodeever.exe	5	Not fixed
P10	Memory leak	main.cpp	worstcodeever.exe	12	Not fixed

Quickly finds memory errors, including leaks and corruptions, in single and multithreaded applications. This decreases support costs by identifying problems before an application ships.

ID	Problem	Sources	Modules	Object Size	State
P1	Data race	main.cpp	worstcodeever.exe		Not fixed
P2	Lock hierarchy violation	main.cpp	worstcodeever.exe		Not fixed

ID	Description	Source	Function	Module	Object Size	State
X10	Allocation site	main.cpp:176	DeadLock	worstcodeever.exe		Information
X13	Allocation site	main.cpp:174	DeadLock	worstcodeever.exe		Information

Accurately pinpoint latent threading errors including deadlocks and data races. This helps reduce stalls and crashes due to threading errors not found by debuggers and other tools.

Analysis completed successfully **Interpret Result**

Event Log → Sources

Time	Description	Modules	Sources
11:06:05	Error:Invalid memory access	worstcodeever.exe	main.cpp:52
11:06:06	Error:Invalid memory access	worstcodeever.exe	main.cpp:54
11:06:06	Error:Uninitialized memory access	worstcodeever.exe	main.cpp:51; main.cpp:56
11:06:06	Error:Uninitialized memory access	worstcodeever.exe	main.cpp:51; main.cpp:57
11:06:06	Error:Invalid memory access	worstcodeever.exe	main.cpp:61
11:06:06	Error:Mismatched allocation/deallo...	worstcodeever.exe	main.cpp:64; main.cpp:66
11:06:06	Error:Mismatched allocation/deallo...	worstcodeever.exe	main.cpp:63; main.cpp:67
11:06:06	Error:Invalid memory access	worstcodeever.exe	main.cpp:79
11:06:06	Error:Memory leak	worstcodeever.exe	main.cpp:76
11:06:06	Error:Memory leak	worstcodeever.exe	main.cpp:192

Clicking the **Interpret Result** button intuitively guides you by grouping related issues together. When you fix one problem, Intel Parallel Inspector shows you all of the related locations where the same fix needs to be applied.

Build | Debug | Tools | Window | Co

- Build Solution F7
- Rebuild Solution Ctrl+Alt+F7
- Clean Solution
- Build worstcodeever
- Rebuild worstcodeever
- Clean worstcodeever
- Project Only
- Profile Guided Optimization
- Batch Build...
- Configuration Manager...
- Compile Ctrl+F7

Intel® Parallel Inspector works on standard debug builds and even binaries. No special compilers, add-ins, or special builds are needed.

Configure Analysis

Intel® Parallel Inspector

Analysis type: Memory Errors Always show this dialog box before running memory error analyses

2x-20x | 10x-40x | 20x-80x | 40x-160x

Does my target leak memory?

Does my target have memory access problems?

Where are the memory access problems?

Where are all the memory problems Inspector can find?

Analysis Time Overhead

Simple analysis configuration enables you to control the depth of analysis vs. collection time.

- L1 analysis finds memory leaks and deadlocks.
- L2 analysis identifies the existence of a problem.
- L3 analysis provides root cause information to fix problems.
- L4 provides the most comprehensive level of problem identification and detail.

Overview → Sources ← Details

Focus Observation: main.cpp:86 - Write

```

85 for (int i=0; i<TotalBlueTroops; i++)
86     BantamBridge+=Blue;
87     return NULL;
88 }
    
```

Related Observation: main.cpp:92 - Write

```

91 for (int i=0; i<TotalGreyTroops; i++)
92     BantamBridge+=Gray;
93     return NULL;
94 }
    
```

Click on an identified problem to reveal the source code to go directly to the offending code to make changes quickly.

Private suppressions:

- Delete problems
- Delete problems
- Mark problems
- Do not use suppressions

Result suppression allows you to mark or delete identified issues that are not relevant.

Features

- Intel Parallel Inspector is fully integrated with Microsoft Visual Studio*.
- Find memory errors in single and multithreaded applications.
- Memory checking includes uninitialized load detection, use of invalid memory references, mismatched memory allocation and deallocation, memory leaks detection, stack memory checks, and stack trace with controllable stack trace depth.
- Find threading errors.
- Benefit from data race detection, deadlock detection, depth- configurable call stack analysis, diagnostic guidance, built-in knowledge of Intel® Threading Building Blocks, OpenMP*, and Windows* threads.
- Intel Parallel Inspector works with any standard debug build.
- No special test builds or compilers are required, making it easier to test code more often.
- Dynamic instrumentation enables testing code without the source; test larger applications because less memory is needed since only executed code is instrumented.

System Requirements

- Microsoft Visual Studio 2005*, 2008*, or 2010* (except the Express Edition)
- For the latest system requirements, go to: www.intel.com/software/products/systemrequirements/.

Compatibility

- Compilers: Microsoft Visual C++* Compiler 2005, 2008, 2010 and Intel® C++ Compiler
- Threading methodologies: Intel® Threading Building Blocks, Intel® Cilk™ Plus, OpenMP*, Windows* threads
- Processors: Designed for and tested on Intel® IA-32 and Intel® 64 processors including Intel® Core™2 and Core™ i7 processors. It can be used on compatible processors, although proprietary instructions may cause it to function incorrectly. Please note that Intel® Parallel Composer (compiler and libraries) supports Intel IA-32, Intel 64, and all compatible processors.

Support

Purchase of Intel® Parallel Studio products include Premium Support service which allows you to submit questions, access to product updates, and technical documentation.

For more information, go to <http://software.intel.com/sites/support/>.

Download a Trial Version Today

Evaluation copy available at: www.intel.com/software/products/ParallelStudio/

Optimization Notice

Intel® compilers, associated libraries and associated development tools may include or utilize options that optimize for instruction sets that are available in both Intel® and non-Intel microprocessors (for example SIMD instruction sets), but do not optimize equally for non-Intel microprocessors. In addition, certain compiler options for Intel compilers, including some that are not specific to Intel micro-architecture, are reserved for Intel microprocessors. For a detailed description of Intel compiler options, including the instruction sets and specific microprocessors they implicate, please refer to the “Intel® Compiler User and Reference Guides” under “Compiler Options.” Many library routines that are part of Intel® compiler products are more highly optimized for Intel microprocessors than for other microprocessors. While the compilers and libraries in Intel® compiler products offer optimizations for both Intel and Intel-compatible microprocessors, depending on the options you select, your code and other factors, you likely will get extra performance on Intel microprocessors.

Intel® compilers, associated libraries and associated development tools may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include Intel® Streaming SIMD Extensions 2 (Intel® SSE2), Intel® Streaming SIMD Extensions 3 (Intel® SSE3), and Supplemental Streaming SIMD Extensions 3 (Intel® SSSE3) instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors.

While Intel believes our compilers and libraries are excellent choices to assist in obtaining the best performance on Intel® and non-Intel microprocessors, Intel recommends that you evaluate other compilers and libraries to determine which best meet your requirements. We hope to win your business by striving to offer the best performance of any compiler or library; please let us know if you find we do not.

Notice revision #20101101

The Ultimate All-in-One Performance Toolkit—Intel® Parallel Studio 2011

Designed for today's serial applications and tomorrow's software innovators

Intel brings simplified threading to Microsoft Visual Studio* C++ developers with a complete productivity solution designed to optimize serial and new threaded applications for multicore and scale for manycore.

INNOVATIVE THREADING ASSISTANT

Intel® Parallel Advisor 2011: Demystify and speed threaded application design.

COMPILER AND THREADED LIBRARIES

Intel® Parallel Composer 2011: Develop effective applications with a C/C++ compiler and advanced threaded libraries.

MEMORY AND THREADING ERROR CHECKER

Intel® Parallel Inspector 2011: Ensure application reliability with proactive parallel memory and threading error checking.

THREADING AND PERFORMANCE PROFILER

Intel® Parallel Amplifier 2011: Quickly find bottlenecks and tune threaded applications for scalable multicore performance.

