

# Getting Started

*VTune*

*Intel's Visual Tuning Environment for  
Windows\* 95 and Windows NT\* Developers*

Order Number: 657001-003

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, or life sustaining 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 various processors 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.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation  
P.O. Box 7641  
Mt. Prospect IL 60056-764

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

\* Third-party brands and names are the property of their respective owners.  
Copyright © 1997. Intel Corporation. All rights reserved.

# Contents

---

<b>VTune: An Overview</b> .....	<b>1</b>
<b>Hardware Requirements</b> .....	<b>4</b>
<b>Software Requirements</b> .....	<b>4</b>
<b>Installation</b> .....	<b>6</b>
<b>Starting VTune</b> .....	<b>9</b>
<b>Getting Online Help in VTune</b> .....	<b>11</b>
<b>The VTune Project</b> .....	<b>13</b>
The Project Wizard.....	13
Tips on Managing Projects .....	14
<b>Key Features of VTune</b> .....	<b>14</b>
The Sampler.....	15
The Java* Call Graph.....	17
The Code Analyzer.....	18
The Code Coach .....	20
Static Assembly Code Analyzer .....	21
Dynamic Assembly Analyzer .....	23

# *VTune: An Overview*

---

Welcome to VTune, Intel's Visual Tuning Environment. Designed to provide an integrated tuning environment for Windows\* 95 and Windows NT\* systems, VTune helps you do the following:

## **Analyze performance through Sampling and Hotspot Analysis**

- Sample and display a system-wide view of software activity and CPU time distribution.
- Determine which modules are taking the most CPU time, including Java\* classes compiled with the Just-in-Time (JIT) compiler, executables, DLLs, and VXD's.
- Analyze the hotspots in these modules, display the source code, and determine performance problems at source and assembly code levels.

## **Profile your Java\* source code and analyze the data**

- In a compatible Java environment, profile your Java code, analyze the data, and generate a Call Graph.

## **Analyze performance through Code Analysis.**

- Perform Static Assembler Analysis of the functions or basic blocks of code in your application without executing or sampling your application.
- Get a list of functions with their respective addresses for quick access to your code.
- Get summary information associated with each function.

## **Analyze your C, Fortran, or Java source.**

- Use VTune's Coach to analyze C, Fortran, or Java source code and get high-level language optimization advice.

**Determine performance problems at the assembly code level using the Static Assembly Analyzer.**

- See how each assembly instruction is processed in the different decode units of the Pentium® Pro and Pentium II processors. (Each assembly instruction on the Pentium Pro and Pentium II processors is composed of one or more micro-ops.)
- Determine how many clocks each assembly instruction takes to execute and how many of them were incurred due to penalties, plus identify instructions that did not pair and the reasons why they did not pair on Pentium processors and Pentium processors with MMX™ technology.

**Fine-tune sections of your code using the Dynamic Assembly Analyzer.**

- Regardless of the processor in the system you are currently running, dynamically simulate a block of assembly code and discover events that degrade performance on the Pentium and Pentium with MMX technology processors, such as missed cache accesses, BTB misses, and misaligned data.

## What's new in VTune 2.5

VTune 2.5 offers the following new features to help you analyze the performance of your Java\* application:

### **Sampling and Hotspot Analysis for JIT-compiled Java modules**

In a compatible Java environment, you can sample your application for Java hotspots. The Modules Report of software activity will now include the CPU time spent in the JIT-compiled Java module, Java VM, and JIT. The Hotspots view for the Java module displays the CPU time spent in the most active classes or methods in the module.

### **Call Graph generation for Java methods**

You can use VTune to profile your Java code and generate a Call Graph, graphically displaying the threads that were created and the parent and child methods that were called during the profiling session.

### **Coach to provide tuning advice for Java source code**

If the Java source code is available, VTune's Coach now provides tuning advice for your Java source code.

# Hardware Requirements

---

Following are the minimum hardware requirements for installing VTune:

- Intel486™ microprocessor (Pentium® processor recommended)
- 16MB of RAM (32MB recommended)
- 40MB of disk space on the local or network drive you specify for VTune files. On Windows\* NT\*, VTune output files are created only on the local drive
- An additional 10MB of disk space on your local Windows System drive
- CD-ROM Drive

The additional disk space on your local Windows drive is required to install the DLLs and OCXs that VTune requires in the windows\system directory.

---

**NOTE.** *You will require more disk space if you install the rest of the software available on the VTune CD, such as the Intel Performance Library Suite, C/C++ Compiler plug-in, Fortran compiler plug-in, Intel Architecture Tutorials, and reference manuals.*

---

## Requirements for Event-Based Sampling

- Event-Based Sampling (EBS) uses internal CPU events to sample your application and is supported by VTune on Pentium Pro and Pentium II processors.

---

**NOTE.** *You must use a Pentium Pro or a Pentium II processor for Event-Based Sampling (EBS) since these processors have a built-in capability for EBS. Intel no longer distributes custom EBS sockets for Pentium processor-based systems.*

---

# Software Requirements

---

VTune runs on either Windows\* 95 or Windows NT\* (Version 3.51 or higher) operating systems. Windows 3.x and Windows NT 3.5 are not supported.

## Java\* Environments Supported

VTune 2.5 currently supports the following Java environments:

- Microsoft\* VM, Build 2228 and above. This VM is used in the Java SDK 2.0 Environment and Internet Explorer 4.0.
- Microsoft SDK 1.5 for Java with Intel JIT. Use the Switch JIT utility provided in the VTune program group to setup the Intel JIT.

---

**NOTE.** Refer to <http://developer.intel.com/design/perftool/vtune> for more up-to-date information about the different Java environments supported.

---



# *Installation*

---

Follow the instructions below to install VTune on systems running Windows\* 95 or NT\*.

## **To Install VTune on Windows 95 or Windows NT (3.51 or higher)**

1. Insert the VTune CD into your CD ROM drive.  
The AutoRun feature will execute automatically. The installation window appears prompting you to install VTune and other software available on the CD.
2. If the AutoRun feature does not execute, then execute AutoRun manually from your command prompt, e.g. `e:/autorun.exe`
3. Click on the VTune icon to install VTune.
4. Follow the instructions the setup program prompts you through.

---

### **NOTES.**

1. *You must have administrator privileges to install VTune on Windows NT.*
  2. *To do sampling on Windows NT, you must either have administrator rights or have the "Profile system performance right" assigned to your user account by the administrator. Otherwise, VTune will not be able to collect samples.*
  3. *If the User ID of the person using VTune is different from the User ID of the person who installed VTune, then the person using VTune must double-click on "Update User's VTune Registry" icon in the VTune group when they login for the first time.*
  4. *Windows NT 4.0 Beta releases are NOT supported and VTune does not support Checked Build on Windows NT.*
-

## Upgrading to VTune 2.5:

If you have VTune 2.4 or an earlier version installed on your system, follow the instructions below to install VTune 2.5.

### To upgrade to VTune 2.5:

- Run the `setup.exe` program from the VTune CD.

The `setup` program will prompt you through the installation process.

---

**NOTE.** *If you have 2.11 or earlier version installed, before you upgrade to 2.5 or later version of VTune, use the Export Project command on the VTune File menu to save the registry entries for your old projects into files with `.vts` extensions. You can also use the Registry Editor to save your project settings. In 2.4 and later versions, VTune automatically saves a project's settings into a `.vts` file.*

---

---

**NOTE.** *Use the VTune Uninstall program in the VTune program group **only** if you wish to remove all the project files (`*.vts`) you created as well as all the files VTune installed on your system. The Uninstall program, does not, however, delete VTune output files (`*.ldb` and `*.mdb`).*

---

## Installing Other Intel Architecture Software from the VTune CD

The VTune CD includes other Intel software products, such as the Intel Architecture Tutorials, online reference manuals, the C/C++ Compiler plug-in, the Fortran compiler plug-in, and the Performance Library Suite.

These products provide you with the tools to optimize your code for the Intel Architecture and to help you understand the Intel Architecture concepts and terminology used by VTune. Note, however, that VTune itself does not require these tools to be installed in order to execute.

### To install the other software on the VTune CD

1. Run the `Autorun.exe` program from the VTune CD.  
The installation window appears, prompting you to install the various software components available on the CD.
2. Click on the appropriate button to install specific software.
3. Follow the instructions the setup program prompts you through.

Refer to the Release Notes in each of these product directories for more information.

# Starting VTune

---

To start VTune, double-click on the VTune icon on the Windows\* 95/NT\* program menu or program group.

When you invoke VTune for the first time, the Assistant appears, prompting you with options about what to do next. Select the “Open the last project” option in the Assistant window to load a sample project called VTundemo. Use the Assistant's directions to run this project to familiarize yourself with the VTune project settings and sessions.

## The VTune Assistant

The VTune Assistant is designed to introduce you to the various aspects of VTune and quickly teach you how to use the interface to tune your applications. By default, the Assistant appears every time you invoke VTune. The initial Assistant window prompts you to create a new project or open an existing one. You can click on any of the icons in the Assistant window to invoke a function. Depending on your selection, the subsequent Assistant windows display tips about the specific function or VTune window you invoke.

## Related Online Help Topics

### Topics in the “Overviews” section of the Help Contents

The VTune Assistant

## Online Help Topics to Help You Get Started

Refer to the following Online Help topics for information that will help you quickly create a VTune project and start using VTune for tuning your application.

### Topics in the Help Contents

- Before you begin
- Quick Start to Time-Based Sampling (TBS)
- Quick Start to Event-Based Sampling (EBS)
- Java\* Environments Supported by VTune
- Quick Start to Sampling your Java Application
- Quick Start to Generating a Call Graph of Java Methods

### Topics in the “Overviews” section of the Help Contents

- The Project Wizard: An Overview
- The VTune Assistant: An Overview
- Java Support in VTune: An Overview
- Call Graph: An Overview
- Call List: An Overview

# Getting Online Help in VTune

---

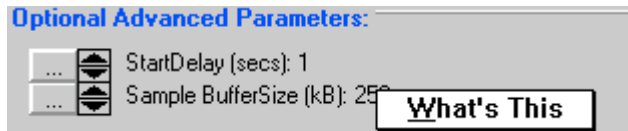
Documentation for VTune is provided in the form of Windows\* 95/NT\* Online help. VTune has extensive context-sensitive help built into the interface. Since the Online help is the main source of information on VTune, effort has been made to make the Online help comprehensive. You can access Online Help from the application in several ways:

## Use the Hints on the toolbar icons.



Select the VTune window and place your cursor on any icon in the toolbar to display the hint for that icon.

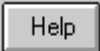
## Use the right mouse button to invoke the “What's This” help for any part of the interface.



“What's This” help provides a brief description of the option, column, icon, button, graph, or any other part of the interface you click on.

**Use the right mouse click on any column or graph in a VTune window to display the “What to do now” help topic**

The “What to do now” help topic displays a brief list of the different functions you can invoke from any VTune form, such as the Modules, Hotspot, or the Source views.

**Use the  button to display the Overview help topic for the active VTune window.**

You could also use the **F1** key.

**Use the F1 key to display help for the Menu commands**

Select the menu command and press the F1 key. VTune displays context-sensitive help for that command.

**Use the Help Contents and Index**

You can invoke the Help Contents and the Help Index from the Help menu. Use the “How to” sections of the Help Contents to display step-by-step instructions.

# The VTune Project


---

The first step towards using VTune is to define your project. VTune uses the concept of a “project” to define the configuration options set up for sampling and analyzing your application. VTune saves your project settings into a file with a `.vts` extension.



## The Project Wizard

VTune provides you with a Project Wizard to help you quickly create a project and start using VTune for tuning your applications.

### To invoke the Project Wizard:

- Click on the New Project icon  on the VTune Assistant window or on the toolbar.

VTune displays the Project Wizard which prompts you to enter the basic information required for both Time-Based Sampling and Static Code Analysis. VTune creates a project with the parameters you specified, uses default values for other settings, and saves the project with the name of the program you are analyzing.

You can now click on the Start a Monitor Session icon  on the toolbar to start sampling the hotspots in your application, or click on the View Code Analysis icon  to perform a static analysis of all the functions in your code.

## Tips on the Project Wizard

- When the Wizard prompts you for the name of the program you want to test, enter the name of an executable (EXE) that will invoke the module you are interested in, not the name of the module itself (DLL, VXD, or OCX).
- For Java\* programs, specify the name of the application (e.g. browser, Java applet viewer, etc.) you want to invoke to execute your Java class file.



## Tips on Managing Projects

- If you modified your source code and want to perform a detailed analysis of the original and the modified versions, create separate projects and maintain both sets of source code in separate directories.
- Even if you modified and recompiled your application, you can continue to use the same project if you were concerned only with system-wide CPU usage or localities of hotspots. Be aware, however, that the samples collected in earlier sessions may no longer correspond accurately to the addresses and offsets in the changed source files. If you open an earlier session, VTune detects the change in the executable and displays a warning message.
- If you wish to analyze your application's performance under different project configurations, create separate projects for each configuration. Note that you can use one project and rely on the Session IDs and comments to track the configuration changes.
- If you want to compare multiple HotSpot screens from different sampling sessions, use the same project. VTune can have only one project open at a time. All sessions you wish to simultaneously view must be in the same project.
- Project files created with previous versions of VTune (versions prior to VTune 2.4) can be opened in VTune only if they are located in the same directory as they were originally created.

## Related Online Help Topics

### Topics in the Help Contents

- Quick Start to Time-Based Sampling (TBS)
- Quick Start to Event-Based Sampling (EBS)
- Quick Start to Sampling your Java\* Application

### Topics in the “Overviews” section of the Help Contents

- The VTune Project
- The Project Wizard

# Key Features of VTune

---

## The Sampler


Use VTune's Sampler to run a "Monitor Session." VTune monitors all active software on your system, including Java class files, your Windows applications and DLLs, the operating system, device drivers, and other application software.

---

**NOTE.** *In order to have VTune display symbol information for Windows Operating System modules, it is necessary that you install all the symbols files from your Win32\* SDK or DDK. On Windows\* 95, make sure you install .sys files and, on Windows NT\*, .dbg files.*

---

### Before you invoke the Sampler:

1. Use the Project Wizard to create a project and specify the application you want to monitor.
2. Click on the Start Monitor Session icon  on the toolbar to start the monitoring and sampling process.

### During the Monitor Session, the VTune Sampler does the following:

- Executes your application
- At the specified interrupt intervals, collects a sample (32 bytes of data) of the current instruction address and places it in a buffer
- Writes the sample data to the disk when the buffer is full
- At the end of the specified sampling time, matches the collected instruction addresses with the modules
- Stores the results in a database

At the end of the Monitor Session, VTune displays the total number of samples collected for that session and asks if you would like to analyze the samples. (If you click OK at the prompt, VTune displays the Modules Report with a system-wide view of software activity.)

## Tips

- In a compatible Java\* environment, VTune also gets the names of JIT-compiled methods along with their load addresses, sizes, and debug information from the JIT/VM and writes this data also to a file.
- To display the Hotspots Report with a graphical view of the most active functions in the module, double-click on a module of interest.
- To display the source code for the hotspot, double-click on a spike representing a function of interest in the Hotspots graph.
- To invoke the Coach, double-click on a line of C, Fortran, or Java source code.
- To display the context-sensitive “What’s This” help for any part of the interface, click your right mouse button on that part of the interface.

## Related Online Help Topics

### Topics in the Help Contents

- “Symbol Files” and “Debug Symbols” in the “Before you Begin” section
- Quick Start to Time-Based Sampling (TBS)
- Quick Start to Event-Based Sampling (EBS)
- Quick Start to Sampling Java applications

### Topics in the “Overviews” section of the Help Contents

- Time-Based Sampling and Event-Based Sampling
- Monitoring a Java application
- Modules View for JIT-compiled Java methods

---

**NOTE.** *If you enabled Call Graph profiling for Java applications, during the Monitor Session, VTune will also collect profile data for generating a call graph. See the Help section “Call Graph Profiling” for more information.*

---

## Java\* Call Graph Profiling

In a VTune compatible Java environment, you can use VTune to profile your Java application and generate a call graph of Java methods.

Before you start a Monitor Session, you must enable the Call Graph Profiling option in the Project Options/Advanced tab. During the Monitor Session, in addition to collecting Hotspot Analysis data, VTune also collects Call Graph profiling data.

During Call Graph profiling, Java methods are instrumented and profiling code is added to each method so that every time a method is loaded, VTune is notified. VTune collects the profiling data and stores it in a `.prf` file. At the end of the Monitor Session, VTune creates a new entry in the Sessions view for the Call Graph session.

When you double-click on the Call Graph entry in the Sessions View, VTune analyzes the data and generates the call graph. The Call Graph window displays the following information:

- Every thread that was created.
- The parent methods that were called while a specific thread was in progress.
- The child methods that were called by the parent methods.
- The number of times a specific method was called.
- The amount of time spent in a specific method.
- The total time spent in a method and in the child methods it called.

---

**CAUTION.** *If Call Graph Profiling is enabled, the Monitor Session becomes intrusive and slows down application execution due to the instrumentation of the Java methods.*

---

## **Related Online Help Topics**


### **Topics in the “Overviews” section of the Help Contents**

- [Call Graph Profiling: An Overview](#)


### **Topics in the “How To” section of the Help Contents**

- [To enable Call Graph Profiling](#)
- [To analyze a Call Graph session and display a Call Graph of Java\\* methods](#)

## The Code Analyzer

Use VTune's Code Analyzer  to analyze the performance of your program (.obj, .exe, or .dll) without sampling.

### Before you invoke the Code Analyzer:

1. Use the Project Wizard to create a project and specify the application you want to monitor.
2. Click on the Code Analyzer icon  on the VTune toolbar to start static code analysis.

VTune analyzes each basic block and function in the program you specify, creates a database with the results, and displays the results in the Code Analysis window. It displays summary information about the performance of each function, including the name, the address, the number of instructions executed, the percentage of pairing, the total clocks incurred, and the additional clocks incurred due to penalties on Pentium<sup>®</sup> processors and Pentium processors with MMX<sup>™</sup> technology.

### Tips

- To display the source code for any function or basic block in the Code Analysis window, double-click on the function or basic block.
- VTune displays source code for any program compiled with the debug symbols turned on. The debug symbols need not be in the binary.
- If no source code is available, VTune's Static Assembly Analyzer disassembles the functions and displays the assembly code.
- To display the What's This help for each column, click your right mouse button on any column in the Code Analysis window.

### Related Online Help Topics

#### Topics in the “Overviews” section of the Help Contents

- Code Analyzer: An Overview


#### Topics in the “How To” section of the Help Contents

- Analyzing performance through static code analysis

## The Code Coach

Use VTune's Coach to get optimization advice for C, Fortran, or Java\* source code. Once you identify the active functions in your program and display the source code, you can invoke the Coach to analyze the code and display advice on how to modify the original code and improve its performance.

**To invoke the Coach, do one of the following from the source view:**

- Double-click on a line of C, Fortran, or Java code
- Or select a loop or a function and then click on the Coach icon  on the source view toolbar.

For C and Fortran source code, the Coach prompts you to specify the Makefile options, the Manual options, or a preprocessed file. It uses the Makefile or Manual options to complete the source code before it analyzes its performance, searches for optimization advice in the code, and displays the advice. If a preprocessed ( . i ) file is specified, the Coach can use that to analyze the performance of your source code.

### Tips

- Click on the Help button next to each optimization displayed by the Coach to invoke the context-sensitive help with examples of original and optimized code.

## Related Online Help Topics


**Topics in the “Overviews” section of the Help Contents**

- Code Coach: An Overview
- Interpreting CPU time in Source view

## Static Assembly Code Analyzer

Use VTune's Static Assembly Analyzer to disassemble the functions in your Windows\* 95/NT\* binary files and analyze the performance attributes of each assembly instruction at the processor level.

### To invoke the Static Assembly analyzer:

1. Double-click on a function in the Code Analysis view or on a hotspot in the Hotspot view.
2. If the source code is displayed, click on the View Assembly icon  on the source view toolbar.

VTune disassembles the function or hotspot and displays the assembly instructions annotated with performance information.

### Tips

- You can select a group of instructions and display summary information about their performance in the status bar.
- To evaluate the performance of your application on other Intel processors, select a CPU type from the pull-down menu on the Assembly view status bar.
- From the Static Assembly view, you can invoke the Dynamic Assembly Analyzer for fine-tuning a small section of your code.
- You can use the right mouse click on any column in the assembly view to invoke the “What's this column” or the “What to do now” help topics.
- To display a detailed description of the pairing and penalty issues incurred by an assembly instruction, double-click on the instruction and display the Advanced Instruction Analyzer.

---

**CAUTION.** *The percentage values displayed in the Time and Event columns in the source views actually apply to other instructions in the loop or block in question and not to the instruction next to which they are displayed. For more information, please see “Interpreting Time and Event Totals in the Assembly View” in the “Overviews” section of the Help Contents.*

---



## Related Online Help Topics

### Topics in the “Overviews” section of the Help Contents

- [Static Assembly Analysis: An Overview](#)
- [Interpreting CPU time in the Assembly view](#)

### Topics in the “How To” section of the Help Contents

- [Dynamically Analyzing Assembly Code](#)

## Dynamic Assembly Analyzer




Use VTune's Dynamic Assembly Analyzer to invoke a processor-specific simulator to dynamically analyze and fine-tune a small section of your application. You can invoke the Dynamic Analyzer from any of the Source or Static Assembly Analysis views.

---

**NOTE.** *The Dynamic Assembly Analyzer currently supports the Pentium and Pentium with MMX™ technology processors. It does not support Pentium® Pro and Pentium II processors.*

---

### To invoke the Dynamic Analyzer from the Source or Static Assembly Analysis view:

- Use the Set Simulation Entry point icon  and the Exit point icon  on the Source view toolbar to mark the range of assembly code you want to simulate and analyze.
- Click on the Start Dynamic Simulation icon  in the toolbar to invoke the Dynamic Analyzer. A window will appear in which you can set detailed parameters of the simulation before the simulation actually starts.

### During Dynamic Analysis, VTune

- Executes your application full speed to the Entry point you specify.
- Single steps to trace execution flow to determine the exact sequence of the executed instructions and their memory addresses.
- From the Entry point you specify, simulates performance on an instruction-by-instruction basis.
- Identifies performance stalls and penalties associated with the instructions such as instruction and data cache misses, BTB misses, and misaligned data references.
- Combines this information with known architectural pairing rules and restrictions and displays the information in the source window.

## **Tips**

- You can use the right mouse click on any column in the assembly view to invoke the “What's this column” or the “What to do now” help topics.
- To display a detailed description of the architectural and penalty issues incurred by an assembly instruction, double-click on the instruction to display information provided by the Advanced Instruction Analyzer.

## **Related Online Help Topics**

### **Topics in the “Overviews” section of the Help Contents**

- Dynamic Analysis: An Overview
- Clock Counts in the Dynamic View

### **Topics in the “How To” section of the Help Contents**

- Dynamically Analyzing Assembly Code