Tutorial: Analyzing Disk Input/Output Waits

Intel® VTune™ Amplifier for Linux* OS

C++ Sample Application Code

Legal Information

Contents

Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors which may cause deviations from published specifications. Current characterized errata are available on request.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: Learn About Intel® Processor Numbers

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Cilk, Intel, the Intel logo, Intel Atom, Intel Core, Intel Inside, Intel NetBurst, Intel SpeedStep, Intel vPro, Intel Xeon Phi, Intel XScale, Itanium, MMX, Pentium, Thunderbolt, Ultrabook, VTune and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries.

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

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Overview


Discover how to use the **Disk Input/Output** analysis type of the Intel® VTune™ Amplifier analyze an I/O bound application and explore options to optimize work with I/O devices.

About This Tutorial	This tutorial uses the sample `diskio` application and guides you through basic workflow steps required to analyze the code for I/O waits.
Estimated Duration	10-15 minutes.
Learning Objectives	After you complete this tutorial, you should be able to: • Configure your application for Disk Input and Output analysis. • Run the Disk Input and Output analysis. • Understand basic Disk I/O metrics provided by the VTune Amplifier. • Identify I/O waits issues and their cause using the VTune Amplfiier.
More Resources	• Intel VTune Amplifier tutorials (HTML, PDF): https://software.intel.com/en-us/articles/intel-vtune-amplifier-tutorials • Intel VTune Amplifier support page: https://software.intel.com/en-us/intel-vtune-amplifier-xe-support

Navigation Quick Start

1


Intel® VTune™ Amplifier provides information on code performance for users developing serial and multithreaded applications on Windows*, Linux*, Android, and OS X* operating systems. VTune Amplifier helps you analyze algorithm choices and identify where and how your application can benefit from available hardware resources.

VTune Amplifier XE Access

VTune Amplifier installation includes shell scripts that you can run in your terminal window to set up environment variables:

1. From the installation directory, type `source amplxe-vars.sh`.

   This script sets the PATH environment variable that specifies locations of the product graphical user interface utility and command line utility.

   NOTE:

   The default installation directory is:

   - For root users: `/opt/intel/vtune_amplifier_xe_version`
   - For non-root users: `$HOME/intel/vtune_amplifier_xe_version`

2. Type `amplxe-gui` to launch the product graphical interface.

VTune Amplifier for Systems Access

VTune Amplifier installation includes shell scripts that you can run in your terminal window to set up environment variables:

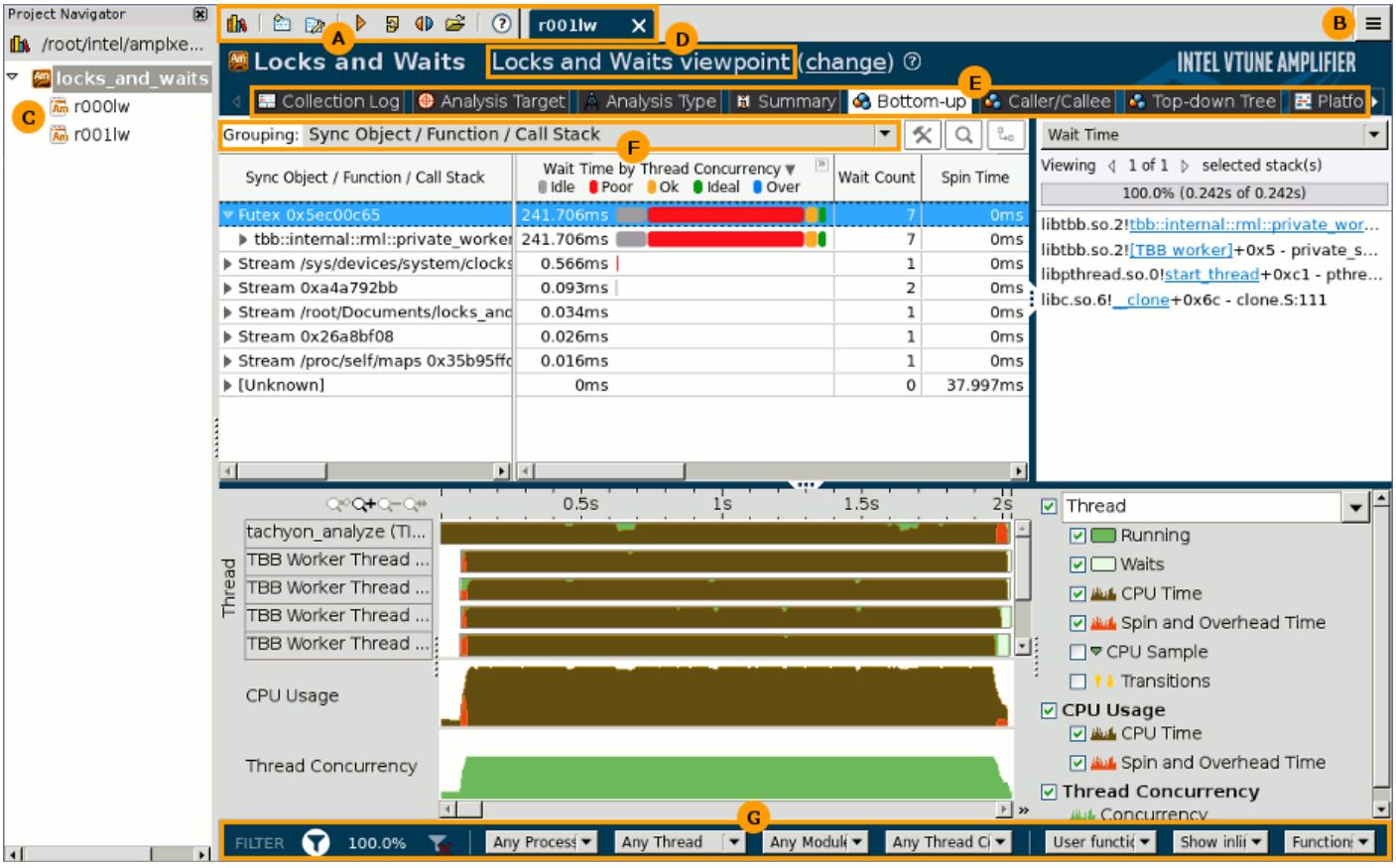
1. From the installation directory, type `source amplxe-vars.sh` for bourne or korn or bash shells, or `source amplxe-vars.csh` if you are using the C shell.

   This script sets the PATH environment variable that specifies locations of the product graphical user interface utility and command line utility.

   The default installation directory is `/opt/intel/system_studio_version/vtune_amplifier_version_for_systems`.

2. Type `amplxe-gui` to launch the product graphical interface.

VTune Amplifier GUI



A Configure and manage projects and results, and launch new analyses from the primary toolbar. Click the **Configure Project** button on this toolbar and use the **Analysis Target** tab to manage result file locations. Newly completed and opened analysis results along with result comparisons appear in the results tab for easy navigation.

B Use the VTune Amplifier menu to control result collection, define and view project properties, and set various options.

C The **Project Navigator** provides an iconic representation of your projects and analysis results. Click the **Project Navigator** button on the toolbar to enable/disable the **Project Navigator**.

D Click the **(change)** link to select a *viewpoint*, a preset configuration of windows/panes for an analysis result. For each analysis type, you can switch among several viewpoints to focus on particular performance metrics. Click the yellow question mark icon to read the viewpoint description.

E Switch between window tabs to explore the analysis type configuration options and collected data provided by the selected viewpoint.

F Use the **Grouping** drop-down menu to choose a granularity level for grouping data in the grid.

G Use the filter toolbar to filter out the result data according to the selected categories.

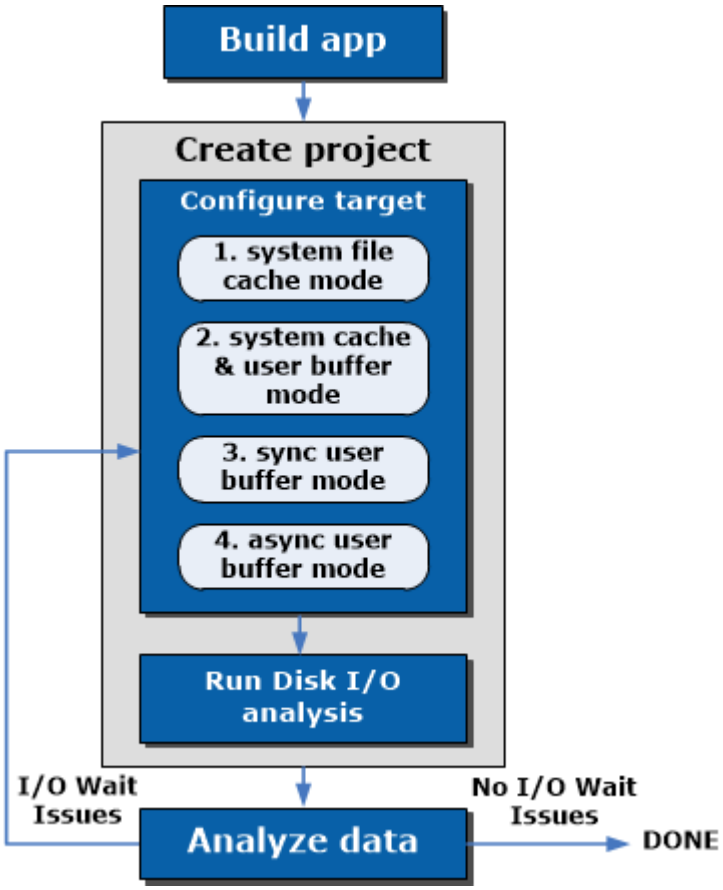
See Also

Click here for more Getting Started Tutorials

Analyzing Disk Input and Output Waits

2

You can use the Intel® VTune™ Amplifier to analyze how efficiently your application is using I/O devices by performing a series of steps in a workflow. This tutorial guides you through these workflow steps while using a sample application named `diskio`. This application works with the I/O device using the system file cache and user buffer.



1.	Prepare for analysis	Install the VTune Amplifier and run the `Makefile` to compile the sample code with the GCC* compiler.
2.	Configure and run a Disk Input and Output analysis	• Launch the VTune Amplifier as root. • Create a VTune Amplifier project. • Specify your sample `diskio` application as an analysis target. • Configure the application parameters to run the target in the system file cache mode. • Run the Disk I/O analysis.

3.	Analyze data collected in the system file cache mode	Use the Disk Input and Output viewpoint to analyze summary statistics and platform-wide metrics on the timeline and identify performance issues for the application working with the I/O device in the system file cache mode.
4.	Re-run the target in the system file cache and user buffer mode and analyze data	• Re-configure the application to run in the combined system file cache and user buffer mode. • Re-run the Disk I/O analysis. • Analyze the summary and platform-wide metrics on the timeline to identify the performance gain and analyze I/O waits.
5.	Re-run the target in the synchronous user buffer mode and analyze data	• Re-configure the application to run in the synchronous user buffer mode. • Re-run the Disk I/O analysis. • Analyze the summary and platform-wide metrics on the timeline to explore changes in the I/O waits distribution and identify more room for improvement
6.	Re-run the target in the asynchronous user buffer mode and analyze data	• Re-configure the application to run in the asynchronous user buffer mode. • Re-run the Disk I/O analysis. • Analyze the summary and platform-wide metrics on the timeline to identify the performance improvements.

See Also
Click here for more tutorials

Prepare Your App for Analysis


Before you start analyzing your application target for I/O issues, do the following:

1. Get software tools.
2. Build your application.

Get Software Tools

You need the following tools to try the tutorial steps yourself using the `diskio` sample application:

- Intel® VTune™ Amplifier, including sample applications
- `.tgz` file extraction utility
- GNU* compiler

Acquire Intel VTune Amplifier

If you do not already have access to the VTune Amplifier, you can download an evaluation copy from http://software.intel.com/en-us/articles/intel-software-evaluation-center/.

Install and Set Up VTune Amplifier Sample Applications

1. Copy the `diskio_vtune_amp_xe.tgz` file from the `<install-dir>/samples/<locale>/C++` directory to a writable directory or share on your system. The default installation path for the VTune Amplifier XE is `opt/intel/vtune_amplifier_xe_<version>`; for the VTune Amplifier for Systems the default path is:
   - For root users: `/opt/intel/system_studio_<version>/vtune_amplifier_for_systems`
   - For non-root users: `$HOME/intel/system_studio_<version>/vtune_amplifier_for_systems`
2. Extract the sample from the tgz file.

NOTE:

- Samples are non-deterministic. Your screens may vary from the screen captures shown throughout this tutorial.
- Samples are designed only to illustrate the VTune Amplifier features; they do not represent best practices for creating code.
- Disk Input and Output analysis is supported for both VTune Amplifier XE and VTune Amplifier for Systems. Steps in this tutorial are applicable to both products but screen shots are made with the VTune Amplifier XE version.

Build Your Application

1. Browse to the directory where you extracted the sample code (for example, `/home/samples/diskio`).
2. Run the `Makefile` to build the application:

```
$ make
```

The application is built with the g++ compiler with all optimizations turned on:

```
$ g++ diskio.cpp -O3 -lrt
```

3. Run the application to create a performance baseline:

```
$ time ./diskio -f <outputfile> -m <c|b|s|a>
```

where <*outputfile*> is an output file with the records created by the `diskio` application; the `-m` option supports the following operation modes for the output file:

- `c` writes data (16 bytes) asynchronously to the output file using the system file cache;
- `b` writes data (16 bytes) to the user buffer (1024 records) and then writes the buffer to a file using the system file cache;
- `s` writes data (16 bytes) synchronously to the user buffer (1024 records) and writes the buffer directly to the I/O device without using the system file cache;
- `a` writes data (16 bytes) to the user buffers and asynchronously submits them to the disk.

For example, run the application in the system cache mode:

```
$ time ./diskio -f out.txt -m c

real    0m36.403s
user    0m4.859s
sys     0m30.771s
```

36.403 seconds is your performance baseline. You can later compare this application execution time with the execution time after proposed optimizations.

Recap

You built the target in the release mode with all optimizations turned on and created a performance baseline that could be used further to identify performance improvements. Your application is ready for analysis.

Next Step

Configure and Run Disk Input/Output Analysis

Configure and Run Disk I/O Analysis

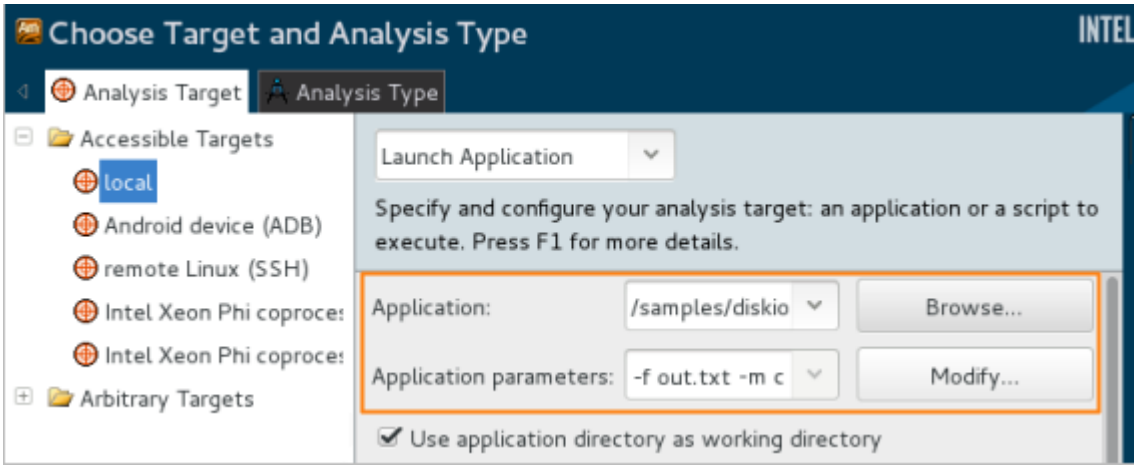
When you application is ready for analysis, create a VTune Amplifier project to specify your analysis target and run the data collection.

To run the analysis:

1. **Launch the VTune Amplifier with root privileges** : From the `<install_dir>/bin64` directory, run the `amplxe-gui` script launching the VTune Amplifier GUI.
2. **Create a VTune Amplifier project:**
   a. From the menu, select **New > Project...**.

      The **Create a Project** dialog box opens.
   b. In the dialog box, specify the project name `diskio` that will be used as the project directory name and click **Create Project**.

      VTune Amplifier creates the `diskio` project directory under the `root/intel/ampl/projects` directory and opens the **Choose Target and Analysis Type** window.
3. **Configure your analysis target:**
   a. From the left pane on the **Analysis Target** tab, select the **Accessible Targets > local** target system.
   b. From the right pane, select the **Launch Application** target type and specify your target as follows:
      - For the **Application** field, browse to: `<sample_code_dir>`, for example: `/home/samples/diskio/diskio`.
      - In the **Application parameters** field, specify the following parameters for this target: `-f out.txt -m c`.



4. **Select and run the analysis:**
   a. Click the **Choose Analysis** button on the right to select an analysis type.

      The **Analysis Type** tab is active.
   b. From the left pane, select **Platform Analysis > Disk Input/Output Analysis**.

      The right pane is updated with the predefined settings for the Disk Input/Output analysis.
   c. Click the **Start** button on the right.

NOTE:

- The Disk Input and Output analysis actively relies on the data produced by the kernel block driver system. In case your platform utilizes a non-standard block driver sub-system (for example, user-space storage drivers), disk metrics will not be available in the analysis type.
- Almost all Disk I/O metrics collected by the VTune Amplifier are system-wide (except for the I/O API calls that are attributed to a process). So when collecting the data, the VTune Amplifier automatically enables the **Analyze system-wide** target configuration option.

VTune Amplifier launches the `diskio` application in the system cache mode so that the application asynchronously writes records (16 Byte) to the output file `out.txt`, using the system file cache, and exits. VTune Amplifier finalizes the collected results and opens the analysis result in the **Disk Input/Output Analysis** viewpoint.

To make sure the performance of the application is repeatable, go through the entire tuning process on the same system with a minimal amount of other software executing.

NOTE:

This tutorial explains how to run an analysis from the VTune Amplifier graphical user interface (GUI). You can also use the VTune Amplifier command-line interface (`amplxe-cl` command) to run an analysis. If you run the sample program from the VTune Amplifier command-line interface, enter:

```
$ amplxe-cl -collect disk-io -app-working-dir /home/samples -- /home/samples/diskio -f
out.txt -m c
```

For more details, check the *Command-line Interface Support* section of the VTune Amplifier Help.

Recap

You launched the VTune Amplifier as root, created a project, specified the sample application as an analysis target and ran the Disk Input and Output analysis using the system file cache application mode.

Key Terms

- project
- target
- finalization
- viewpoint

Next Step

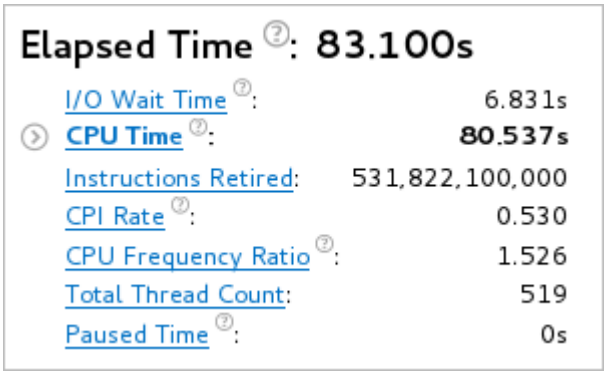
Analyze I/O Data in the System Cache Mode

Analyze I/O Data in the System Cache Mode

You ran the sample `diskio` application in the system cache mode enabled with the option `-m c`. In this mode, the application asynchronously writes records (16 Byte) to the output file, relying on system file cache.

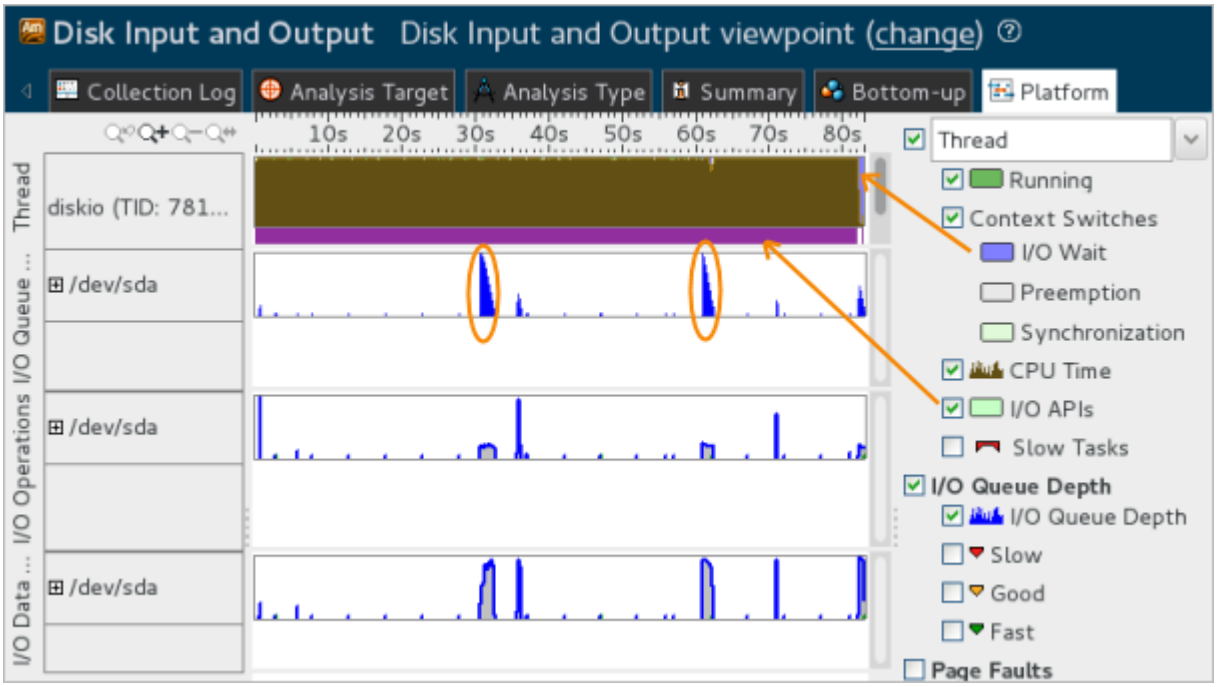
When the **Disk Input and Output** analysis is complete, the Intel® VTune™ Amplifier opens the result in the Disk Input and Output viewpoint.

For the analysis of the input/output-bound applications, the metrics summary at the top of the **Summary** window includes the **I/O Wait Time** metric that shows the amount of time when threads were in the I/O Wait state while there were idle cores on the system. For the `diskio` application in the system file cache mode, the I/O Wait time is very small, which typically means that the application is not I/O-bound.



You see that the sample application spent most of the time executing on the CPU with the CPU Time equal to 80.537 seconds.

To better understand the reason for the high CPU Time, switch to the **Platform** window:



The `diskio` thread shows the intensive usage of the CPU time (brown graph) and I/O API calls (tasks in purple). Since disk I/O operations are executed in parallel with CPU operations, the I/O overhead is small and mostly affects the last part of application execution when the rest of I/O operations are waited. The data processing in the system file cache starts only when the number of data in the cache grows big (see the peaks in the I/O Queue).

To minimize the CPU usage and effectively use the I/O device, consider running the application in a mode combining the usage of the system cache and user buffer.

Recap

You ran the Disk Input and Output analysis for the sample application interacting with the I/O device via system file cache and identified that in this mode the application has a high CPU Time overhead and does not use the I/O device effectively.

Key Terms

- viewpoint
- I/O Wait Time

- CPU Time
- I/O Queue Depth

Next Step

Analyze I/O Data in the System Cache and User Buffer Mode

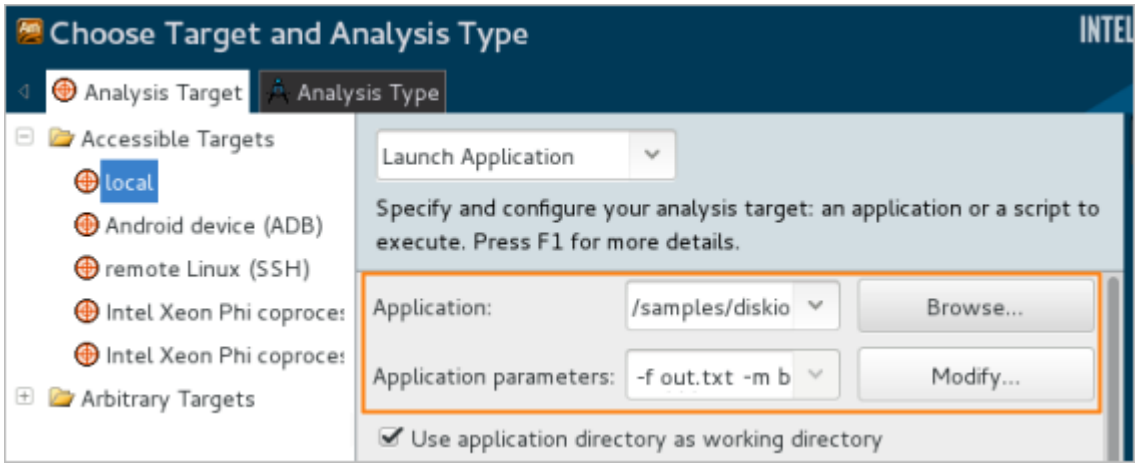
Analyze I/O Data in the System Cache and User Buffer Mode

To minimize the CPU usage and effectively use the I/O device, consider running the application in a mode combining the usage of the system file cache and user buffer.

Configure a new data collection for the existing `diskio` project as follows:

1. Click the **Configure Project** button on the toolbar.

   The **Choose Target and Analysis Type** window opens with the **Analysis Target** tab active.
2. In the **Application parameters** field, specify `-f out.txt -m b`, where `b` is the combined user buffer and system cache mode.

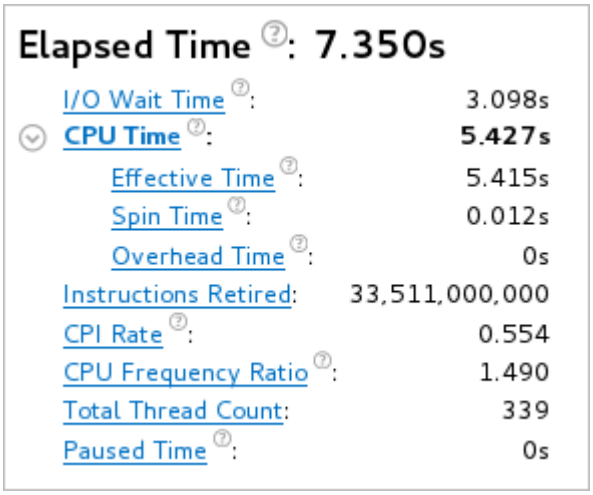


   In this mode the `diskio` application will write records (16 Byte) into the user buffer (1024 records) and then write the buffer to the `out.txt` file relying on the system file cache.
3. Click the **Choose Analysis** button on the right.

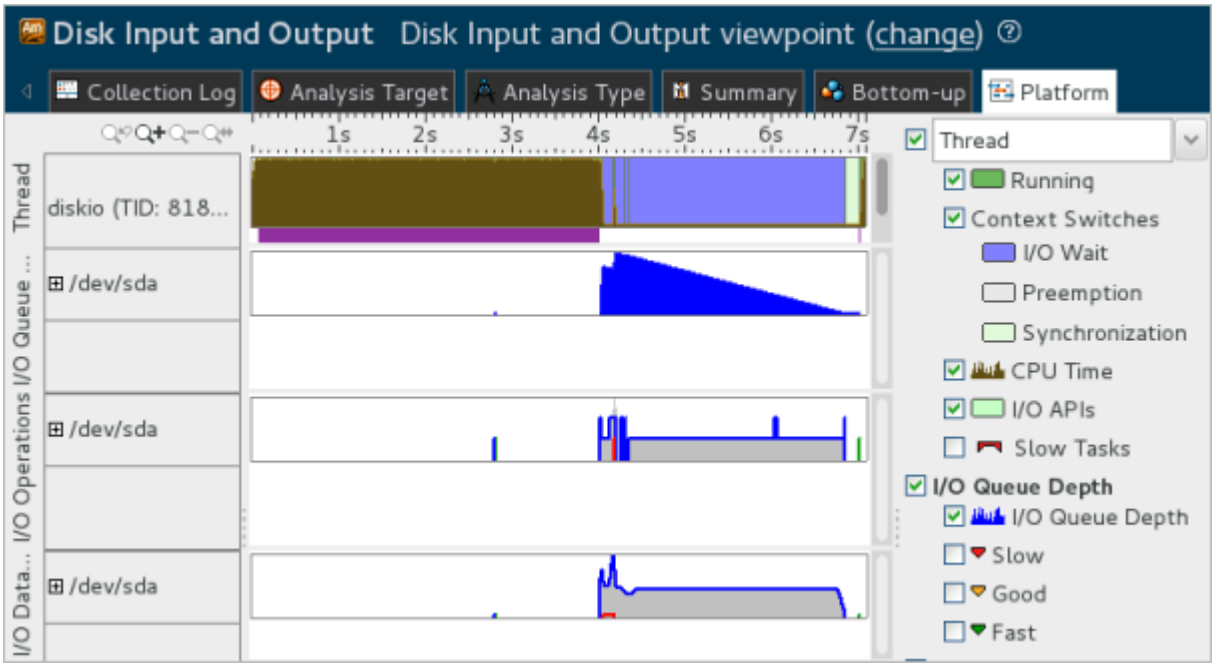
   The **Analysis Type** tab opens. The **Disk Input and Output** analysis is pre-selected.
4. Click the **Start** button on the right.

VTune Amplifier collects data and opens the result in the Disk Input and Output viewpoint with the **Summary** window active by default.

You see that the Elapsed Time of the `diskio` application in the system cache and user buffer mode has decreased significantly and is equal now to 7.350 seconds. At the same time, the **I/O Wait Time** metric has grown to almost 3 seconds and now is comparable with the CPU Time, which classifies the application as I/O bound.



To analyze the changes in the application workflow, switch to the **Platform** window:



You see that reduction of I/O API calls significantly reduced the CPU usage. But when the data is recorded to the output file, all operations are accumulated in the file cache waiting for the data to be written to the disk and creating a queue at the end of the application execution. To resolve this issue, consider writing the data to the disk directly to avoid accumulating a long queue.

Recap

You ran the Disk Input and Output analysis for the sample application in the combined system file cache and user buffer mode and identified that this mode provides about 66s of improvement in the application Elapsed Time but creates a long I/O queue and increases the I/O Wait Time.

Key Terms

- viewpoint
- I/O Wait Time
- CPU Time
- I/O Queue Depth

Next Step

Analyze I/O Data in the Synchronous User Buffer Mode

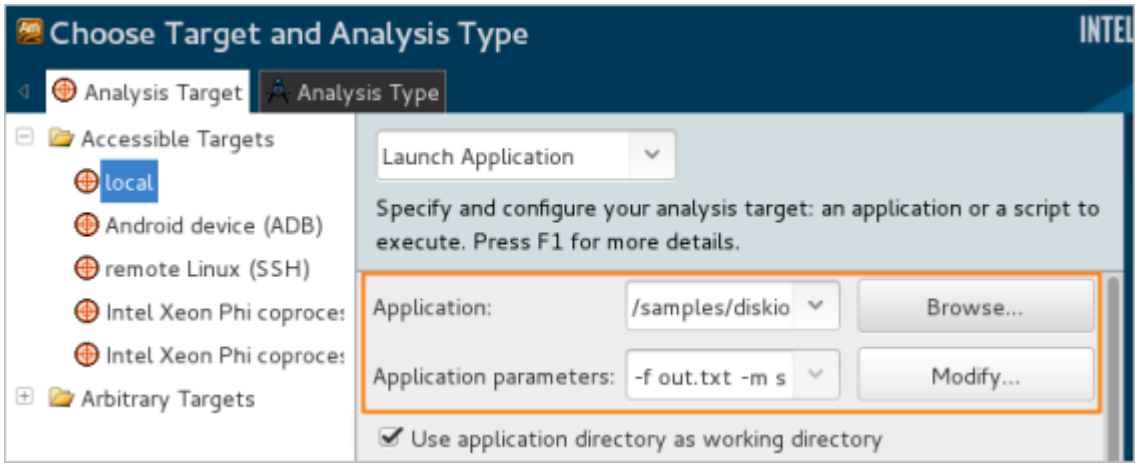
Analyze I/O Data in the Synchronous User Buffer Mode

You identified that in the mixed system file cache and user buffer mode the application runs faster and takes much less CPU Time but creates a big I/O queue and locates all I/O waits at the end of the application execution. To investigate further optimization options, run the `diskio` application in the user buffer mode.

Configure a new data collection for the existing `diskio` project as follows:

1. Click the **Configure Project** button on the toolbar.

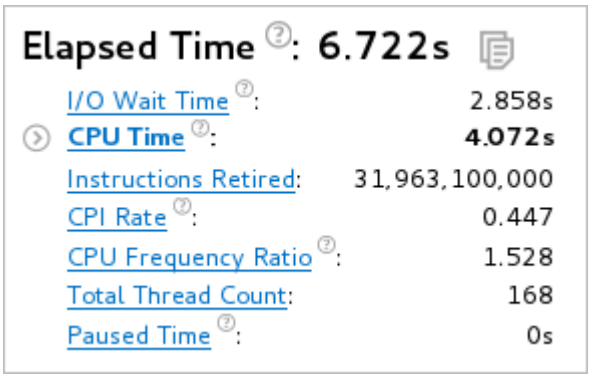
   The **Choose Target and Analysis Type** window opens with the **Analysis Target** tab active.
2. In the **Application parameters** field, specify `-f out.txt -m s`, where `s` is the synchronous user buffer mode.



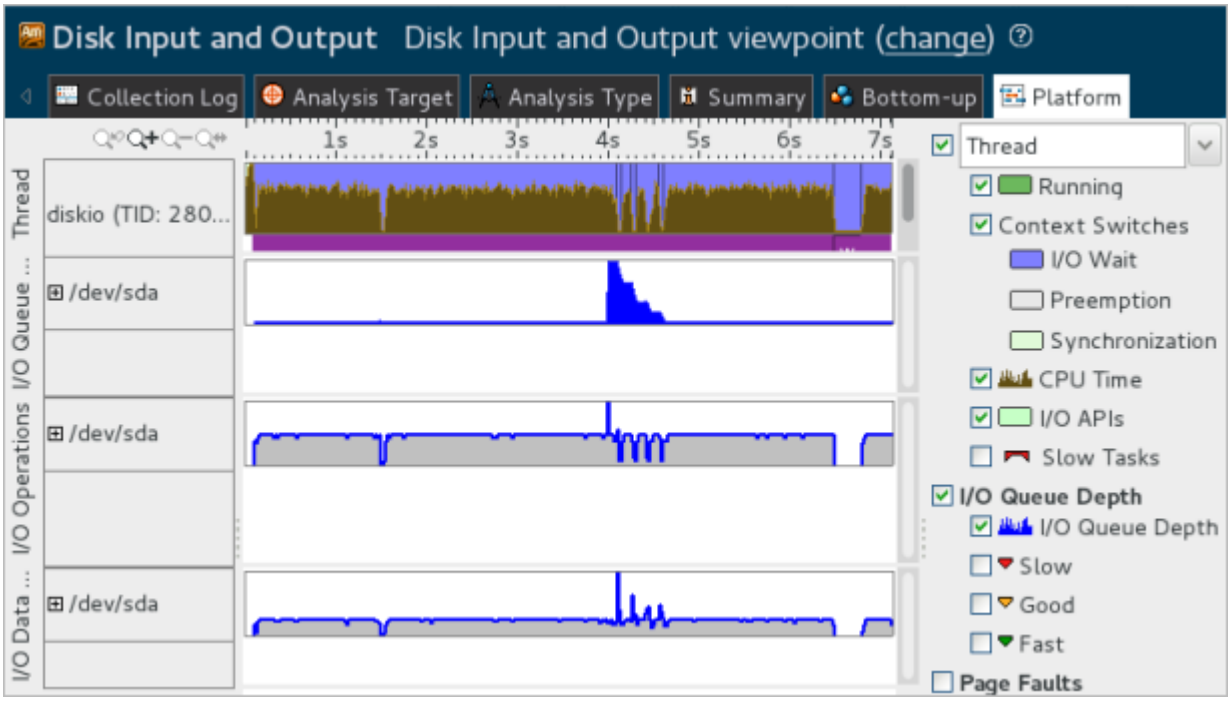
   In this mode the `diskio` application will write records (16 Byte) into the user buffer (1024 records) and then writes the buffer directly to the I/O device without relying on the system file cache.
3. Click the **Choose Analysis** button on the right.

   The Analysis Type tab opens. The **Disk Input and Output** analysis is pre-selected.
4. Click the **Start** button on the right.

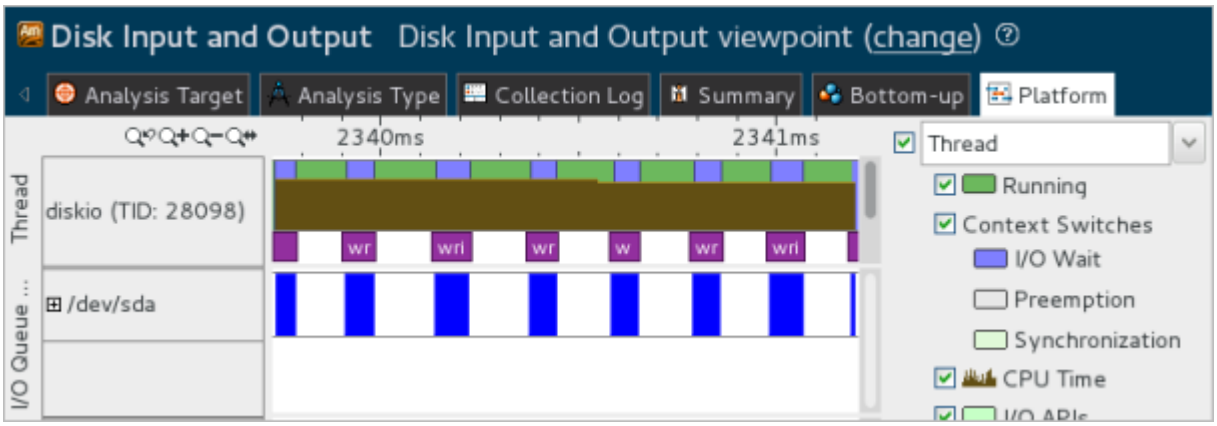
VTune Amplifier collects data and opens the result in the Disk Input and Output viewpoint with the **Summary** window active by default. The metrics summary shows that the application Elapsed time and I/O Wait Time have reduced slightly in comparison with the combined system cache and user buffer mode:



Switch to the **Platform** window to view the changes in the metrics distribution on the timeline:



The CPU Time values for the `diskio` thread in the **Thread** area are not high, which means that putting data into the buffer does not take much time. I/O queue depth is equal to 1 and the I/O device is loaded evenly since the data is written to the disk directly. You may select an area on the timeline and use the context menu options to zoom in and view parallel I/O Waits and user API tasks:



I/O Waits distributed all over the application execution time clearly indicate that the main issue now is waiting for the IO operations submitted directly to the I/O device. Explore whether using two buffers and submitting the data asynchronously to the disk can help get rid of I/O Waits on the user thread.

Recap

You ran the Disk Input and Output analysis for the sample application in the synchronous user buffer mode and identified that though in this mode the Elapsed Time and I/O Wait time have decreased a little and the I/O queue is low, there are still I/O Waits on the user thread that could be avoided.

Key Terms

- viewpoint
- I/O Wait Time
- CPU Time

- I/O Queue Depth

Next Step

Analyze I/O Data in the Asynchronous User Buffer Mode

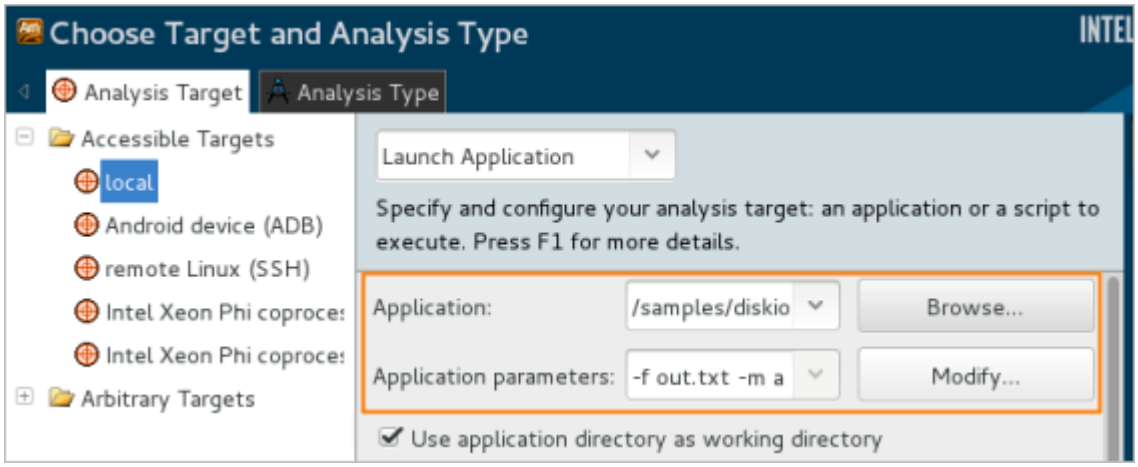
Analyze I/O Data in the Asynchronous User Buffer Mode

You identified that writing the data synchronously to the user buffer and then directly to the I/O device removes the CPU Time overhead but makes the application I/O bound. Try running the sample application in a mode using two user buffers and asynchronously submitting data to the disk.

Configure a new data collection for the existing `diskio` project as follows:

1. Click the **Configure Project** button on the toolbar.

   The **Choose Target and Analysis Type** window opens with the **Analysis Target** tab active.
2. In the **Application parameters** field, specify `-f out.txt -m a`, where `a` is the asynchronous user buffer mode.

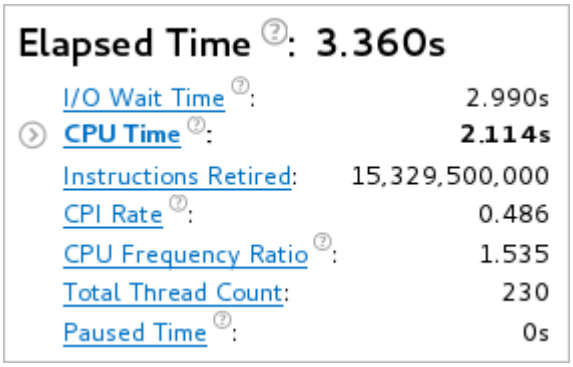


3. Click the **Choose Analysis** button on the right.

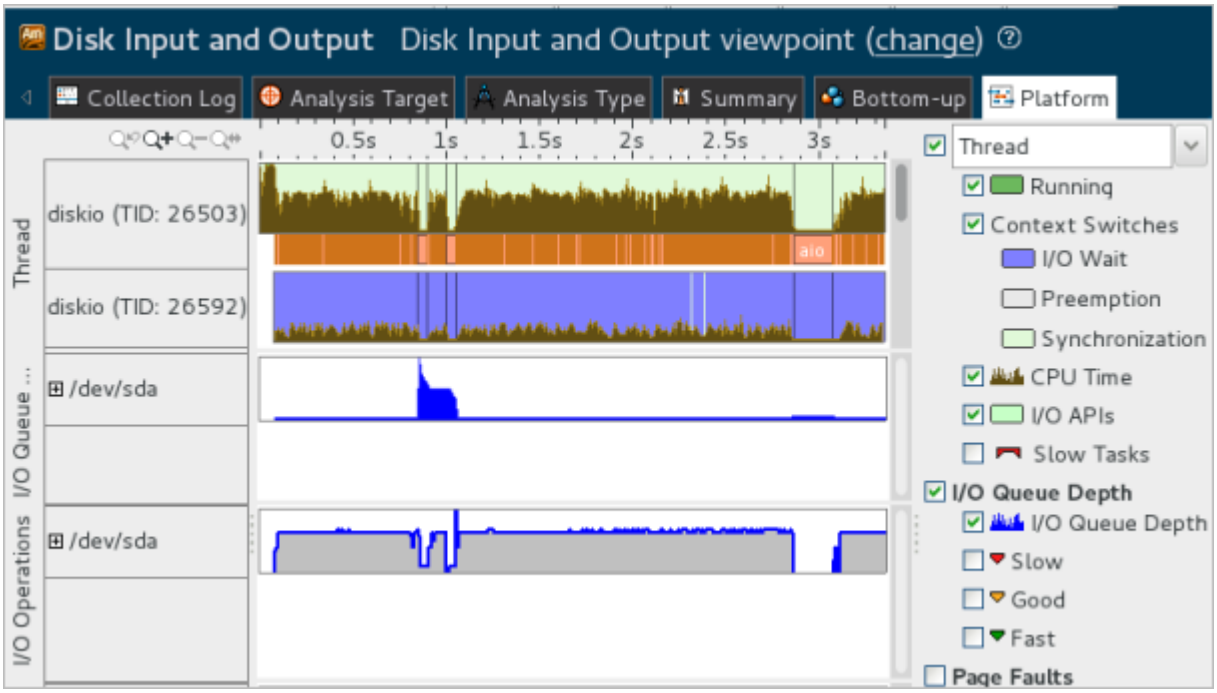
   The **Analysis Type** tab opens. The **Disk Input and Output** analysis is pre-selected.
4. Click the **Start** button on the right.

VTune Amplifier collects data and opens the result in the Disk Input and Output viewpoint with the **Summary** window active by default.

You see that the Elapsed Time of the `diskio` application in the asynchronous user buffer mode has decreased and is equal now to 3.360 seconds. The I/O Wait Time almost has not changed:



If you switch to the **Platform** window, you see two `diskio` threads where the first one is a user thread busy with the CPU activity and it has no I/O Waits anymore; and the second thread is a system thread used for I/O synchronization.



In the asynchronous mode, due to the usage of the two buffers filling the second buffer happens in parallel with writing the first buffer data to the disk. This workflow is optimal since it effectively uses both CPU and I/O resources and provides the fastest application execution.

> **NOTE:**
>
> The screen shots and execution time data provided in this tutorial may differ from the data collected on your system. For example, if your disk is slow but your CPU is powerful, the sample workload provided for filling in the buffer may be insufficient and you will see almost no difference between the synchronous and asynchronous user buffer modes.

Recap

You ran the Disk Input and Output analysis for the sample application in the asynchronous user buffer mode and identified that in this mode the application demonstrates the best performance with no I/O Waits on the user thread and fastest execution time.

Key Terms

- viewpoint
- I/O Wait Time
- CPU Time
- I/O Queue Depth

Next Step

Summary

Summary

3

You have completed the *Analyzing Disk I/O Waits* tutorial. Here are some important things to remember when using the Intel® VTune™ Amplifier to analyze an I/O bound application:

Step	Tutorial Recap	Key Tutorial Take-Aways
1. Prepare your app for analysis.	You built the target in the release mode with all optimizations turned on and created a performance baseline that could be used further to identify performance improvements. Your application is ready for analysis.	• Compile your applications with all optimizations turned on. • Create a performance baseline to compare the application versions before and after optimization.
2. Configure and run Disk Input and Output analysis.	You launched the VTune Amplifier as root, created a project, specified the sample application as an analysis target and ran the Disk Input and Output analysis using the system file cache application mode.	• For Disk I/O analysis, launch the VTune Amplifier as root. VTune Amplifier automatically sets `perf_event_paranoid` to 0. • Use the **Analysis Target** tab to choose and configure your analysis target. • Use the **Analysis Type** tab to choose, configure, and run the analysis. You can also run the analysis from command line using the `amplxe-cl` command as follows: `amplxe-cl -collect disk-io -app-working-dir <working_dir> -- <application_path> [app_parameters]`
3. Analyze I/O Data in the system cache mode.	You ran the Disk Input and Output analysis for the sample application interacting with the I/O device via system file cache and identified that in this mode the application has a high CPU Time overhead and does not use the I/O device effectively.	• Use the Disk Input and Output viewpoint to analyze I/O bound applications. • Start with the **Summary** window and focus on the top-level metrics such as I/O Wait Time, CPU Time and Elapsed Time. • Use the **Platform** window to view the metrics distribution over time. Analyze I/O API tasks and I/O Waits in the Thread area and peaks in the I/O Queue Depth. • For optimizing I/O bound applications, consider using two user buffers and asynchronously writing data to the I/O device.
4. Analyze I/O Data in the system cache and user buffer mode.	You ran the Disk Input and Output analysis for the sample application in the combined system file cache and user buffer mode and identified that this mode provides about 66s of improvement in the application Elapsed Time but creates a long I/O queue and increases the I/O Wait Time.	
5. Analyze I/O Data in the synchronous user buffer mode.	You ran the Disk Input and Output analysis for the sample application in the synchronous user buffer mode and identified that though	

Step	Tutorial Recap	Key Tutorial Take-Aways
	in this mode the Elapsed Time and I/O Wait time have decreased a little and the I/O queue is low, there are still I/O Waits on the user thread that could be avoided.	
6. Analyze I/O Data in the asynchronous user buffer mode.	You ran the Disk Input and Output analysis for the sample application in the asynchronous user buffer mode and identified that in this mode the application demonstrates the best performance with no I/O Waits on the user thread and fastest execution time.	

Next step: Prepare your own application(s) for analysis. Then use the VTune Amplifier to find inefficiencies in the I/O device usage and fix them.

See Also

Click here for more tutorials

Key Terms

4

baseline : A performance metric used as a basis for comparison of the application versions before and after optimization. Baseline should be measurable and reproducible.

CPU Time: The time during which the CPU is actively executing your application.

CPU Activity: A metric that defines a portion of time the system spent in the following states:

- Idle state - the CPU core is idle.
- Active state - the CPU core is executing a thread.
- I/O Wait - the CPU core is idle but there is a thread, blocked by an access to the disk, that could be potentially executed on this core.

Elapsed Time :The total time your target ran, calculated as follows: **Wall clock time at end of application – Wall clock time at start of application**.

finalization : A process during which the Intel® VTune™ Amplifier converts the collected data to a database, resolves symbol information, and pre-computes data to make further analysis more efficient and responsive.

hotspot: A section of code that took a long time to execute. Some hotspots may indicate bottlenecks and can be removed, while other hotspots inevitably take a long time to execute due to their nature.

I/O Data Transfer: A metric that shows the number of bytes read from or written to the storage.

I/O Queue Depth: A metric that shows the number of I/O requests submitted to the storage device. Zero requests in a queue means that there are no requests scheduled and disk is not used at all.

I/O Wait Time: A metric that shows a portion of time when threads reside in I/O wait state while there are idle cores on the system and the number of counted threads is not greater than the number of idling cores.

Page Faults: A metric that shows the number of page faults occurred on a system. This metric is useful when analyzing access to memory mapped files.

PCIe Bandwidth: A metric that represents an amount of data transferred via the PCIe bus per second. This metric is collected only on server platforms based on Intel microarchitecture code name Sandy Bridge EP and later.

project: A container for an analysis target and analysis type configuration and data collection results.

target : A *target* is an executable file you analyze using the Intel® VTune™ Amplifier.

viewpoint : A preset result tab configuration that filters out the data collected during a performance analysis and enables you to focus on specific performance problems. When you select a viewpoint, you select a set of performance metrics the VTune Amplifier shows in the windows/panes of the result tab. To select the required viewpoint, click the **(change)** link at the top of the result tab and use the drop-down menu to select a required viewpoint.