Intel® VTune™ Amplifier XE
and Intel® MPI

Dr.-Ing. Michael Klemm
Senior Application Engineer
Software and Services Group
Legal Disclaimer

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 go to: http://www.intel.com/design/literature.htm
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
Agenda

- Quick Introduction to VTune Amplifier XE
- Analysis of Multi-threaded MPI Applications
Agenda

• Quick Introduction to VTune Amplifier XE
• Analysis of Multi-threaded MPI Applications
Intel® VTune™ Amplifier XE Evolution

Tune
Analyze and optimize performance issues

VTune™ Amplifier XE
Linux OS & Windows OS* GUI/CLI support
Intel® VTune™ Amplifier XE
Quick Overview

- **Fast, Accurate Performance Profiles**
  - Hotspot (Statistical call tree)
  - Hardware-Event Based Sampling (EBS)

- **Thread Profiling**
  - Visualize thread interactions on timeline
  - Balance workloads

- **Easy set-up**
  - Pre-defined performance profiles
  - Use a normal production build

- **Compatible**
  - Microsoft*, GCC*, Intel compilers
  - C/C++, Fortran, Assembly, C#,.NET
  - Latest Intel® Architecture Processors and compatible processors¹

- **Windows OS* or Linux OS***
  - Visual Studio* integration (Windows)
  - Standalone user i/f and command line
  - 32 and 64-bit

¹ IA32 and Intel® 64 architectures. Many features work with compatible processors. Event based sampling requires a genuine Intel® Processor.
Intel® VTune™ Amplifier XE
Performance Profiler

Where is my application...

**Spending Time?**

- Focus tuning on functions taking time
- See call stacks
- See time on source

<table>
<thead>
<tr>
<th>Function - Call Stack</th>
<th>CPU Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithm_2</td>
<td>3.560s</td>
</tr>
<tr>
<td>do_xform</td>
<td>3.560s</td>
</tr>
<tr>
<td>algorithm_1</td>
<td>1.412s</td>
</tr>
<tr>
<td>BaseThreadInitThunk</td>
<td>0.000s</td>
</tr>
</tbody>
</table>

**Wasting Time?**

- See cache misses on your source
- See functions sorted by # of cache misses

```
475 float rx, ry, rz = ...
476 float param1 = (A)
477 float param2 = (A)
478 bool neg = (rz < 0)
```

**Waiting Too Long?**

- See locks by wait time
- Red/Green for CPU utilization during wait

<table>
<thead>
<tr>
<th>Line</th>
<th>MEM_LOAD... LLC_MISS</th>
</tr>
</thead>
<tbody>
<tr>
<td>5000</td>
<td>30,000</td>
</tr>
</tbody>
</table>

Key requirements

- Low overhead
- No special recompiles
- Easy to use
Application Level Analysis: Hotspots

Hottest Call Stack

Hottest Functions

Quickly identify what is important
Application Level Analysis
Concurrency Analysis

Frame is a time step or iteration

Thread active
Thread waiting
Thread transitions

CPU Usage
Thread Concurrency
Frame Rate
Software & Services Group, Developer Products Division

Copyright© 2011, Intel Corporation. All rights reserved. *Other brands and names are the property of their respective owners.
# Hardware-Event Based Sampling (EBS)
## EBS Made Easier

<table>
<thead>
<tr>
<th>System Wide Event Based Sampling (EBS)</th>
<th>Every Intel® Processor has an on chip Performance Monitoring Unit (PMU).</th>
</tr>
</thead>
<tbody>
<tr>
<td>uses the on chip PMU to count performance events like cache misses, clock ticks and instructions retired.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predefined EBS Profiles</th>
<th><img src="image" alt="Predefined EBS Profiles" /></th>
</tr>
</thead>
<tbody>
<tr>
<td>Easy EBS setup for newer processors. No memorizing complex event names. Profiles vary by microarchitecture. (Full custom profiles also available)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Opportunities Highlighted</th>
<th><img src="image" alt="Opportunities Highlighted" /></th>
</tr>
</thead>
<tbody>
<tr>
<td>General Exploration turns the cell pink when it suspects a tuning opportunity is present. Hover gives suggestions.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pinpoint tuning opportunities</th>
<th><img src="image" alt="Pinpoint tuning opportunities" /></th>
</tr>
</thead>
<tbody>
<tr>
<td>See opportunities like cache misses. View results on the timeline, in the grid view or on your source.</td>
<td></td>
</tr>
</tbody>
</table>
New in VTune™ Amplifier XE: Pre-Configured Profiles!

The Intel® Microarchitecture Codename Sandy Bridge: General Exploration profile should be used for a top-level analysis of potential issues. It is the subject of this guide.

All the events required are pre-configured – no research needed! Simply click **Start** to run the analysis.
The Old Way vs. The New Way

The Old Way: To see if there is an issue with branch misprediction, multiply event value (86,400,000) by 20 cycles, then divide by CPU_CLK_UNHALTED.THREAD (5,214,000,000). Then compare the resulting value to a threshold. If it is too high, investigate.

The New Way: Look at the Branch Mispredict metric, and see if any cells are pink. If so, investigate.
Command Line Interface

- `amplxe-cl` is the command line.
  - **Windows:** `C:\Program Files\Intel\Inspector XE\bin[32|64]\amplxe-cl.exe`
  - **Linux:** `/opt/intel/inspector_xe/bin[32|64]/amplxe-cl`
- To get detailed help:
  - `amplxecl -help`
- Get Command Line from GUI
  - Command examples:
    1. `amplxe-cl -collect-list`
    2. `amplxe-cl -knob-list=hotspots`
    3. `amplxe-cl -collect=hotspot - myapp.exe [MyParams]`
    4. `amplxe-cl -report hotspots`

More Help is available with the Online Documentation
Remote Data Collection
Conveniently analyze data collected on remote systems

1. Setup the experiment using GUI locally
2. Copy command line instructions to paste buffer
3. Open remote shell on target machine
4. Paste command line, run collection
5. Copy result file to your local system
6. Open file using local GUI

- Minimal “performance footprint” during collection
- Easy setup using GUI
- Easy analysis of results
Compare Results Quickly - Sort By Difference

- Quickly identify cause of regressions.
  - Run a command line analysis daily
  - Identify the function responsible so you know who to alert
- Compare 2 optimizations – What improved?
- Compare 2 systems – What didn’t speed up as much?
Agenda

- Quick Introduction to VTune Amplifier XE
- Analysis of Multi-threaded MPI Applications
Nesting Multiple Levels of Parallelism

Distributed parallelism
- Explicit coordination through message-passing
- Example: Intel® MPI Library, Intel® Trace Analyzer and Collector (ITAC)

Thread-level parallelism
- Data parallelism and/or tasking
- Examples: OpenMP®, ITAC, VTune™ Amplifier XE

Instruction-level parallelism
- Examples: SIMD, VTune™ Amplifier XE
MPI Analysis
Hybrid program: 2 MPI processes + 12 Threads per process
Hybrid Analysis

• Beyond the inter-process level of MPI parallelism, the processes that make up the programs on a modern cluster often also use fork-join threading through OpenMP* and Intel® TBB

• VTune™ Amplifier XE performance analyzer and the Intel Inspector XE checker can be used to analyze the performance and correctness of an MPI program

• But: it needs some manual work!
Running an Analysis in MPI Applications

• VTune Amplifier XE cannot (yet) analyze MPI applications.

• Limitations:
  – System-wide analysis can only use PMU, no instrumentation
    – But monitors events in all MPI processes on the local node
  – Process-attached analysis also only uses PMU events
    – Analysis focuses on only one process
  – MPI runtime is treated as “system library” and thus can easily be excluded from analysis

• Use Intel Trace Collector and Analyzer for a full-fledged MPI analysis.
Running an Analysis in MPI Applications

• VTune Amplifier XE can be run through mpirun/mpiexec:
  
  mpirun –np N amplxe-cl -collect analysis your_app options

  – Launches a CLI analysis for each MPI process w/ automatic
    finalization after the application finishes

  – To get the command, use the “Get Command Line” button
    in the GUI.

  – For each MPI process, one analysis result will be created.

  – (Planned feature: merge analysis results into global result.)

• MPI job configuration files can be used to restrict
  analysis to subsets of MPI processes

  – see next slide for an example
Running an Analysis in MPI Applications

• Configuration file:

```bash
# config.txt: MPI configuration file
-host h1 -n 12 your_app
-host h2 -n 11 your_app
-host h2 -n 1 amplxe-cl -collect hotspots -r res your_app
```

• Run the job:

```bash
mpirun -configfile config.txt
```

• VTune Amplifier XE will only be invoked for MPI process with rank 23.
  - Result will be placed in directory “res”.

Software & Services Group, Developer Products Division
Copyright © 2011, Intel Corporation. All rights reserved. Other brands and names are the property of their respective owners.
Hybrid Analysis
Pre-view Collected Data

• Once the results are collected, the user can open any of them in the standalone GUI or generate a command line report
  – Use inspxe-cl –help report or amplxe-cl –help report to see the options available for generating reports.

• Here is an example of viewing the text report for functions and modules after a VTune Amplifier XE analysis:

$ amplxe-cl -R hotspots -q -format text –r r003hs
  –Function Module CPU Time
  -------------- ----------- ---------------
  –F a.out 6.070
  –Main a.out 2.990

$ amplxe-cl -R hotspots -q -format text -group-by module –r r003hs
  –Module CPU Time
  ----------- ---------------
  –a.out 9.060
Hybrid Analysis
Visualize results in VTune™ Amplifier XE

• Linux: $ amplxe-gui

• Windows: navigate to the result directory and double click on icon 🍳
Hybrid Analysis (hotspots)
Hybrid program: 2 MPI processes + 12 Threads per process (1/2)

OpenMP regions with routines called inside

OpenMP threads shown together with MPI (dapl) service threads
MPI Analysis (hotspots)
Hybrid program: 2 MPI processes + 12 Threads per process (2/2)

MPI functions shown with the call stack