



Intel® VTune™ Analyzer Beta Support For The Itanium® Processor

- Paul Cohen
- Product Marketing Manager
- Developer Products
Division
- Intel® Corporation
- Gelato* April 2006



Disclaimer

- Information in this document is provided in connection with Intel® products. No license, express or implied, by estoppels or otherwise, to any intellectual property rights is granted by this document, and Intel® makes no representations as to the necessity of obtaining licenses from any third party. Except as provided in the 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 Intel® intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications.
- The information in this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Intel®. Intel assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document.
- Except as permitted by such license, no part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the express written consent of Intel®.
- The Intel® product may contain design defects or errors known as errata which may cause the product to deviate from published specifications.

Eclipse* 3.1 Native on Itanium® Processors

VTune(TM) Performance Tools - Call Graph Results [localhost] - Fri Aug 20

File Edit Refactor Navigate Search Project Tuning Run Windows

Start Help

Related Topics

About Call Graph

The Call Graph view. The lower section of the view provides a pictorial view of program flow to help you quickly identify critical functions and call sequences. The upper section displays the function information in a table format. Select a function in either section to highlight it in both sections.

[Secrets of the Call Graph View](#)

[About Call Graph Views](#)

Go To:

[All Topics](#) [Search](#) [Bookmarks](#)

Process: /opt/intel/vtune/

Function	Calls
test_oror1	1,000,000
divd_rout	500,000
test_if	100,000
test_if1	100,000

Graph View

is the call graph of a sample project and provides a short explanation of several of the view elements:

Process	Self Time	Total Wait ...	Class	Module Path	Full Name
1	2,098	0	0	/lib/tls/	setlocale
232	53	0	0	/lib/tls/	__libc_malloc
53	1	0	0	/lib/tls/	__cfree
1	1	0	0	/lib/tls/	__textdomain
0	0	0	0	/lib/tls/	__ctype_get_m...
11	11	0	0	/lib/tls/	__bindtextdomain
301	301	0	0	/lib/tls/	__overflow
15	15	0	0	/lib/tls/	getopt_long

Call Graph

Thread_0(40211A80)

libc.start_main → main → is_selinux_enabled → setlocale → print_dir → exit

Function Summary

Function: is_selinux_enabled
Module: /lib64/libselinux.so.1
Source:
Total Time: 2.326 ms
Self Time: 2.098 ms
Total Wait Time: 0 ms
Self Wait Time: 0 ms
Calls: 1

Call List

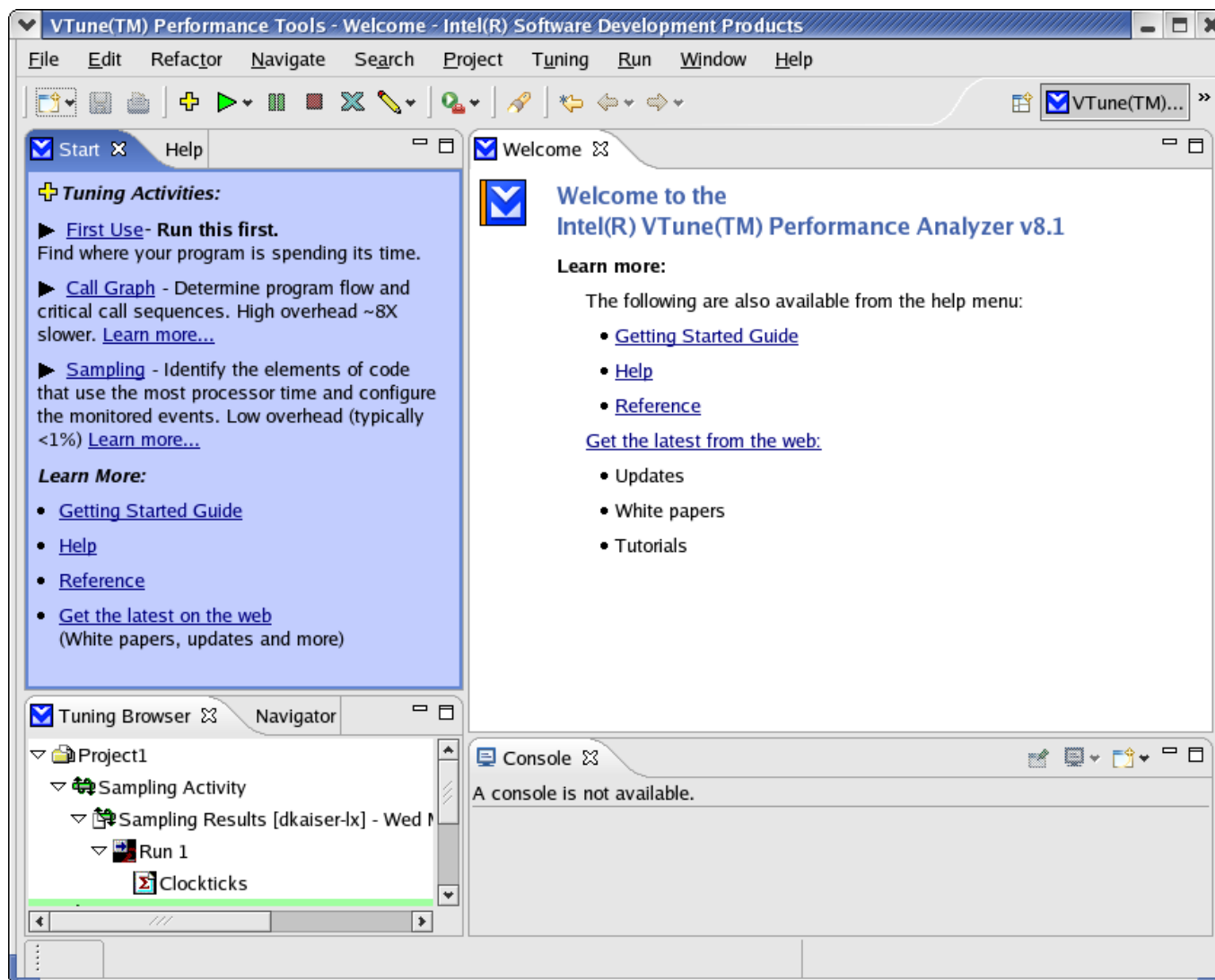
Graph Call list

Callouts:

- Choose the process. Tip: choose process with the largest size
- Click to see the call information in a table format
- Click to see the graph in a small window.
- The red arrows indicate the critical path
- Hover over a function to see its tool tip information. Click the function to select it in the graph and the Function Summary

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

VTune™ Analyzer Initial Screen



Linux* 2.6 Kernel Support

Distribution	Kernel
Red Hat* Enterprise Linux* 3.0 Update 4	2.4.21-27.EL
Red Hat* Enterprise Linux* 4.0	2.6.9-5.EL
SuSE* Linux* Enterprise Server (SLES) 9.0 Service Pack 1	2.6.5-7.139
SuSE* Linux*	9.2 2.6.8-24
SuSE* Linux* Enterprise Server (SLES) 10	
SGI Pro Pack* 3.0	
SGI Pro Pack* 4.0	

- Auto Driver Update

- On install if driver not found we look on the web
- If we can't find driver and kernel source is available we will automatically build driver
- During startup we check compatibility of driver and kernel in case of kernel update
- Manual process to download drivers for off-line machines available

- More supported distributions out of the box

- See Website for details

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



Per CPU Local Buffers

- Allows more efficient, lower overhead way of gathering sampling data on NuMA systems.
- Most useful for >>16 CPU systems
- Buffers are locally allocated on each node, via OS

CPU masking

- Allows system administrator or user to specify exactly which CPUs will be used to gather performance data.
- Collect only the data you need
- This greatly reduces the amount of data you need to collect
- Eclipse* or command line based
- Lots of options
 - All processors
 - Only those in your allocation
 - Only the processors you specify
 - Exact CPU by number
 - Number of CPU's by relative position within allocated group

Compiler Optimization Report Integration

The screenshot shows an IDE with a source code editor and a console window. The source code is as follows:

```

2
3
4 static double a[1000][1000], b[1000][1000], c[1000][1000], t[1], s[1];
5 int i,j,k;
6 dummy(t,a,b,c,s);
7 for( i=0; i<1000; i++) {
8     for( j=0; j< 1000; j++) {
9         for( k=0; k<1000; k++) {
10            c[i][j] = c[i][j] + a[i][k] * b[k][j];
11        }
12    }
13    dummy(t,a,b,c,s);
14    for( i=0; i<1000; i++) {
15        for( k=0; k<1000; k++) {
16            for( j=0; j< 1000; j++) {
17                c[i][j] = c[i][j] + a[i][k] * b[k][j];
18            }
19        }
20    }

```

The table below shows the IA64 and CPU cycle counts for the code blocks:

Line Number	Source	IA64...	CPU_CYCLES
7	for(i=0; i<1000; i++) {		
8	for(j=0; j< 1000; j++) {	0.13%	0.08%
9	for(k=0; k<1000; k++) {	10.60%	7.72%
10	c[i][j] = c[i][j] + a[i][k] * b[k][j];	35.45%	24.85%
11	}		
12	}		
14	for(i=0; i<1000; i++) {	0.04%	
15	for(k=0; k<1000; k++) {	0.31%	0.34%
16	for(j=0; j< 1000; j++) {	11.45%	7.30%
17	c[i][j] = c[i][j] + a[i][k] * b[k][j];	34.57%	25.36%
18	}		
19	}		

The console window shows the optimization report for the loop at line 10:

```

/home/ewmoore/gelato/mgO3.txt
-----
Loop at line 10: interchanged

Resource II = 2
Recurrence II = 1
Minimum II = 2
Scheduled II = 2

Estimated GCS II = 11

Percent of Resource II needed by ar
Percent of Resource II needed by m
Percent of Resource II needed by fl

Number of stages in the software pi

SWP (9-10): main
-----
Loop at line 10: interchanged

Resource II = 4
Recurrence II = 1
Minimum II = 4
Scheduled II = 4

Estimated GCS II = 11

```

- A single click tells you that the compiler didn't optimize your critical loop because of an assumed vector dependency.

Optimization Report Reveals Sub-optimal Assumptions

- Even the best optimizing compiler doesn't always get it right.
- A single click tells you that the compiler didn't optimize your critical loop because of an assumed vector dependency.
- You insert a pragma telling it to ignore the dependency which results in a 20% improvement in run time.

Multi-User Call Graph

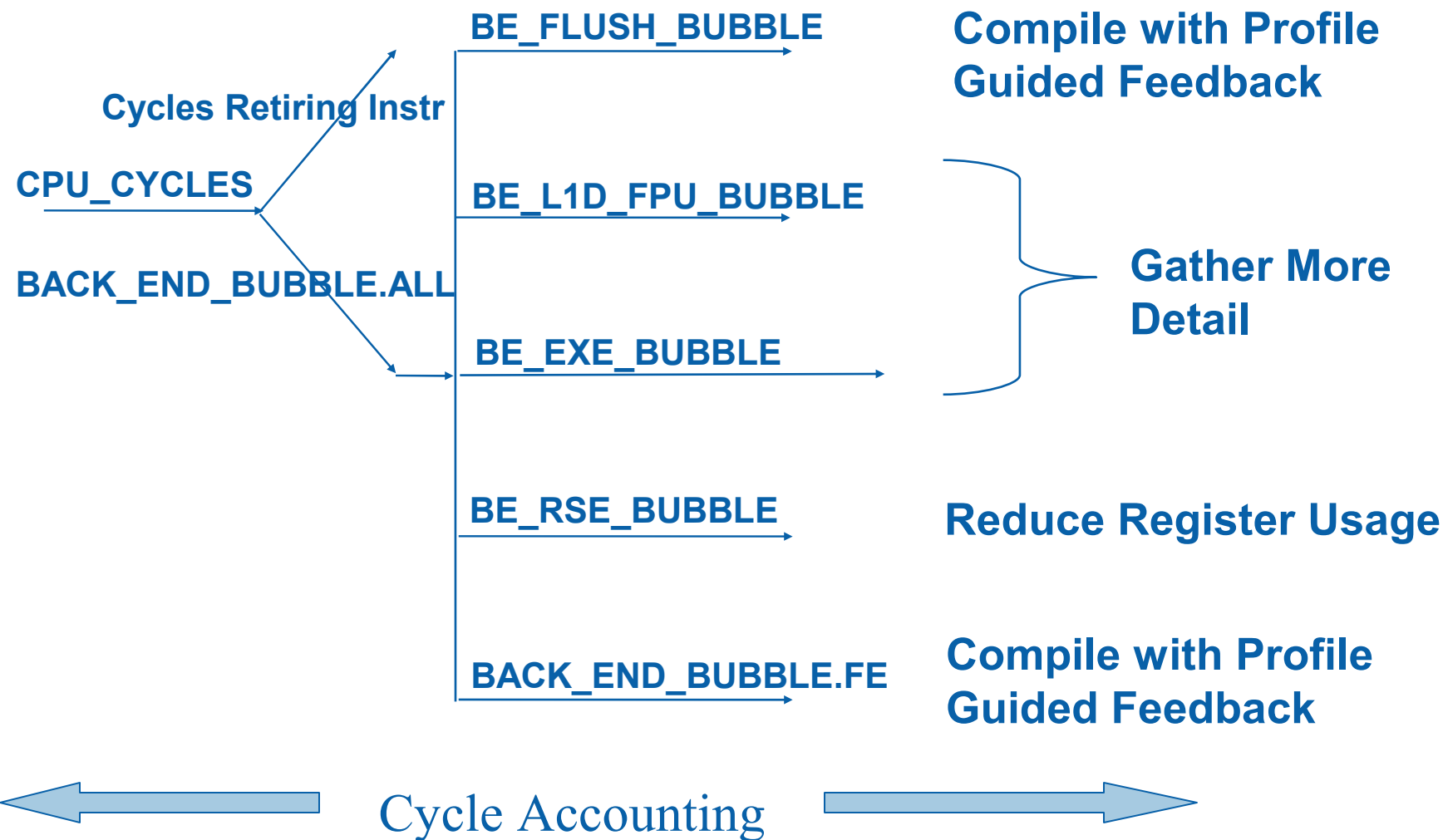
- Allows multiple VTune™ Analyzer users to run Call Graph on the same machine at once
 - Needed for large systems that developers share
 - Especially needed in HPC or Financial Enterprise markets
 - Eliminates need for scheduling a system for 1 person's VTune™ Analyzer use
 - Some customers have users in diverse locations that share systems and it can be difficult to coordinate VTune™ Analyzer usage

Performance Analysis for Itanium® 2-based Hardware

- Cycle accounting identifies processor pipeline stalls
 - Performance Monitoring Units can determine exactly how CPU cycles are used
- VTune™ Analyzer determines where (IP) cycles are being consumed but also exactly why
 - Exact cause and location of execution inefficiency can be identified

Unique Features of the Itanium® 2 microarchitecture

Follow the Cycle Accounting Tree



What is Missing?

- Many different instructions can cause a single type of architectural event to increment.
 - L3 miss can result from lfetch (good) or ld (bad)
- This can cause incorrect interpretation of VTune™ Analyzer data
- Finding source instruction is rarely possible
 - Event skid (imprecise IP) is unavoidable
- Problem is amplified at O3 optimization
 - Aggressive instruction scheduling
 - Prefetching

Event Skid and Optimization can Make Interpretation Very Difficult

Constraining Performance Monitoring Events

- The Performance Monitoring Events can be constrained to only increment on particular
 - Instruction type (OpCode matching)
 - Instruction Pointer range (IP matching)
 - Virtual Address Range (Data Address matching)
 - Or any combination of the above
 - default is no constraint = collect all events

Unique Features of the Itanium® 2 microarchitecture



OpCode Matching

- Problem: How to fix the cache misses that matter to me?
- Solution: Use OpCode matching to increment memory hierarchy events (cache misses etc) coincident with the following:
 - Integer memory ops
 - FP loads/FP stores
 - Prefetch
- Problem: Locating poorly aligned memory accesses
- Solution: Use OpCode matching to find 1 and 2 byte memory ops coincident with memory alignment events
- Problem: How to estimate FP scoreboard dependency stalls
- Solution: Find reciprocal approximations coincident with dependency stall events

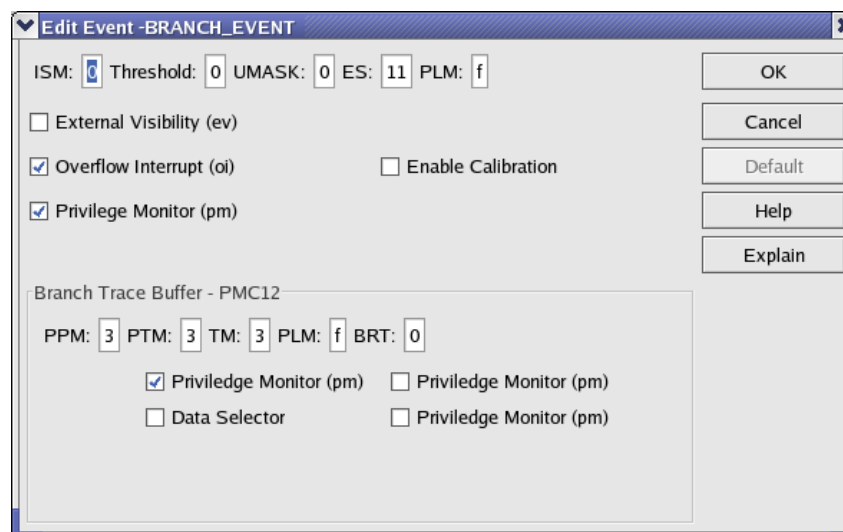
Long Latency Loads

- Problem: Long latency loads frequently lead to data access stalls.
- Partial Solution: Use O3 compilation. This usually prefetches the data adequately but it can fail for many reasons:
 - Short inner loop
 - Threads fighting over the same cache line
 - Multi level gathers
- Solution: Take data with the DEAR_Latency_GT_32/64 event. The source lines with very large numbers are the culprits.

Event Editing

- Allows expert users to fine tune the definition of an EBS event
 - Capture only kernel or User mode events
 - Ex: User may not want to see OS (kernel mode) cache misses
 - Specify type of branch events to be captured as opposed to all branch events
 - Procedure returns
 - Taken conditional branches
 - Not taken conditional branches

Event Editing Example



- User can select User or O.S. mode (plm) for all events
- For Branch Event the branch type (brt), privilege level for the branches captured (plm) and so on



Last But Not Least

- Stay tuned for surprise during James Reinders Keynote





Beta Info

- Intel Booth or vtune_beta@intel.com
- Your feedback help us set direction

