



Hardware Performance Monitoring: Sun's Perspective

Marty Itzkowitz

Project Lead,

Sun Studio™ Performance Tools



Introduction

- Users of HW Performance Monitoring
- HW Performance Monitoring Tools
 - Sun's Profiling Tools
 - Application and Kernel
 - Sun's Monitoring Tools
 - Application, System, and Kernel
 - HW Counter Libraries and APIs
- HW Counter Requirements

Users and their Objectives

- Users — various types
 - End users
 - End user developers
 - ISV developers
 - Compiler writers
 - Field engineers and system tuners
 - System SW developers
 - Chip designers
- Objectives for all:
 - What can I change to make things faster?

End User Tools

- Monitoring Tools
 - Both application and system
 - General overview of behavior
 - Curiosity, more than anything else:
 - End Users have few knobs to turn
 - Can change configurations:
 - Memory
 - Processors
 - Disk controllers
 - Domain partitioning
 - Application migration

End User Developer Tools

- Monitoring tools
 - Show a quick overview, system performance
 - Get data on what to profile
- Profiling Tools
 - Memory Performance: Cache, TLB
 - vs. Functions/source-lines/instructions
 - vs. Data types
 - vs. Cachelines, memory buses, controllers, pages, ...
 - Floating-point Performance
 - Report data against user's programming model
- Library APIs
 - To instrument code regions

ISV Developer Tools

- Very much like end user developers
 - But less likely to use library APIs

Compiler Writer Tools

- Monitoring and profiling
 - Compiler itself
 - Compiler-generated code
- Generated instrumentation
 - Using APIs
- Feedback-directed optimization
 - Memory profiling to inform better layout, striding
 - Branch-mispredict profiles
 - Other opportunities?

Field Engineer/System Tuner Tools

- Monitoring and profiling
 - Both user applications and system
 - Configuration management (like end users)
 - Memory
 - Processors
 - Disk controllers
 - Domain partitioning

System SW Developer Tools

- System-monitoring tools
 - Observability into OS
- Kernel profiling tools
- Runtime monitoring
 - Data to support page placement and migration
 - Understand scheduling
 - Particularly on CMT systems
- Issue: conflict with other uses of HW
 - System-level tools lock out user-level tools

Chip-designer Tools

- Not too much interest in current chips
 - Care about long-term development
 - Sometimes exploit tools on existing systems to inform next-generation designs
 - Long design cycles make this problematic
 - Leapfrog?
 - Mostly use simulators, not old-generation chips
 - Do use current HW monitoring to validate traces
- Unanswerable questions =>
 - Better HW monitoring in next chip to get answers
 - Or so we hope

Sun's Profiling Tools

- Application profilers
 - Sun Studio—`collect` /Analyzer
- Kernel profilers
 - Sun Studio—`er_kernel`/Analyzer
- No multiplexing of HW counters
 - Hard to get valid statistics with multiplexing
 - Implies multiple runs
 - If more counters of interest than counter registers

collect/analyzer

- collect -h <ctr>,<interval>, ...
 - As many counters as the HW allows
 - Automagic register assignment
- Memory counters
 - Prefix with + to record actual instruction, VA, PA
 - US-III,IV
 - Counter skid => backtracking to get data, which may fail
 - Newer chips will fix that
- Can record multiple experiments
 - Aggregate them for presentation
 - In lieu of multiplexing

Application Profiling-Analyzer

Memory Performance Profiling

Performance Analyzer [test.memory.1.er, ...]

File View Timeline Help

Find Text:

Functions Callers-Callees Source Lines Disassembly PCs Timeline LeakList Statistics Experiments

User CPU	CPU Cycles	D\$ and E\$ Stall Cycles	E\$ Stall Cycles	Name
(sec.)	(sec.)	(sec.)	(sec.)	
41.539	31.950	0.068	13.671	<Total>
11.728	7.233	0.016	3.976	dgemv_g1_
9.947	5.433	0.012	4.045	dgemv_opt1_
8.236	7.900	0.009	0.002	load_arrays_
3.853	3.789	0.001	0.990	dgemv_g2_
1.481	1.456	0.002	0.945	dgemv_opt2_
1.111	1.089	0.002	0.	__mt_WaitForWork_
1.091	1.022	0.001	0.949	dgemv_p1_ -- MP doall from line 12 [_\$d1A12.dgemv_p1_]
1.061	1.022	0.001	0.951	dgemv_p2_ -- MP doall from line 31 [_\$d1B31.dgemv_p2_]
0.971	0.933	0.001	0.896	dgemv_hi2_
0.951	0.922	0.001	0.896	dgemv_hil_
0.380	0.378	0.	0.	__mt_EndOfTask_Barrier_
0.360	0.356	0.	0.	barrier_ -- MP doall from line 20 [_\$d1A20.barrier_]
0.360	0.356	0.	0.	barrier_ -- MP doall from line 45 [_\$d1B45.barrier_]
0.010	0.	0.	0.001	_private_close
0.	0.	0.	0.	@plt
0.	0.	0.	0.001	MAIN_
0.	0.	0.	0.	__mt_MasterFunction_
0.	0.	0.	0.	__mt_SlaveFunction_
0.	0.	0.	0.	__mt_run_my_job_

Summary Event Legend Leak

Data for Selected Object:

Name: dgemv_g1_
 PC Address: 2:0x00008538
 Size: 1276
 Source File: /home/martyi/workarea/demos/cachetest
 Object File: /home/martyi/workarea/demos/cachetest
 Load Object: <cachetest>
 Mangled Name:
 Aliases:

Process Times (sec.) / Counts

	Exclusive	Counts
User CPU:	11.728 (28.2%)	11
Wall:	11.958 (29.0%)	11
Total LWP:	11.958 (27.2%)	11
System CPU:	0.210 (17.4%)	0
Wait CPU:	0.020 (33.3%)	0
User Lock:	0. (0.%)	0
Text Page Fault:	0. (0.%)	0
Data Page Fault:	0. (0.%)	0
Other Wait:	0. (0.%)	0

Application Profiling-Analyzer

Disassembly Display

Performance Analyzer [test.1.er]

File View Timeline Help

Find Text:

Functions Callers-Callees Source Lines Disassembly PCs DataLayout DataObjects MemObj Timeline LeakList Statistics Experiments

User CPU (sec.)	D\$ and E\$ Stall Cycles (sec.)	Sync Wait (sec.)	Sync Wait Count	Source File: ./mttest.c	Object File: ./mttest.o	Load Object: <mttest>
0.	0.	0.	0	[1281]	145e4: ld	[%o2 + 208], %o5
0.	0.	0.	0	[1281]	145e8: ldd	[%o5 + 8], %f8 (No type information)
0.	0.	0.	0	[1278]	145ec: nop	
0.	0.	0.	0	[1278]	145f0: nop	
0.	0.	0.	0	[1281]	145f4*: <branch target>	<=====<<<
0.	11.318	0.	0	[1281]	145f4: ld	[%o0], %f0 (structure:workStruct_t -).(float sum_ctr)
23.867	0.	0.	0	[1281]	145f8: inc	%g5
1.281	0.	0.	0	[1281]	145fc: cmp	%g5, %o1
0.	0.	0.	0	[1281]	14600: fstod	%f0, %f4
3.893	0.	0.	0	[1281]	14604: faddd	%f4, %f8, %f6
0.	0.	0.	0	[1281]	14608: fdtos	%f6, %f2
5.624	0.	0.	0	[1281]	1460c: st	%f2, [%o0] (structure:workStruct_t -).(float sum_ctr)
0.700	0.	0.	0	[1278]	14610: nop	
0.	0.	0.	0	[1278]	14614: nop	
0.	0.	0.	0	[1281]	14618: bl,pt	%icc,0x145f4
1.081	0.	0.	0	[1281]	1461c: nop	
0.	0.	0.	0	[1281]	14620: retl	
0.	0.	0.	0	[1281]	14624: nop	

1282. }

Application Profiling-Analyzer

Data Objects Display

Performance Analyzer [mcf.1.er]

File View Timeline Help

Find Text: _____

Functions Callers-Callees Source Lines Disassembly PCs DataLayout DataObjects MemObj Timeline

E\$ Stall Cycles (sec.) (%)	E\$ Read Misses (%)	Name
309.307 100.00	1 600 644 190 100.00	<Total>
169.701 54.87	952 528 575 59.51	{structure:arc -}
133.257 43.08	628 018 840 39.24	{structure:node -}
79.401 25.67	476 814 304 29.79	{structure:arc -}.{cost_t=long cost}
67.323 21.77	341 310 239 21.32	{structure:node -}.{long orientation}
67.212 21.73	411 812 354 25.73	{structure:arc -}.{long ident}
29.645 9.58	127 203 816 7.95	{structure:node -}.{pointer+structure:node child}
20.167 6.52	87 302 619 5.45	{structure:node -}.{cost_t=long potential}
13.611 4.40	13 700 411 0.86	{structure:arc -}.{pointer+node_t=structure:node tail}
8.345 2.70	49 601 488 3.10	{structure:arc -}.{flow_t=long flow}
6.267 2.03	20 800 624 1.30	{structure:node -}.{pointer+structure:node pred}
6.260 2.02	19 096 745 1.19	<Unknown>
5.444 1.76	26 000 780 1.62	{structure:node -}.{pointer+structure:arc basic_arc}
3.456 1.12	17 400 522 1.09	(No type information)
3.133 1.01	21 100 633 1.32	{structure:node -}.{long time}
1.367 0.44	0 0	(Backtracking traversed a branch target)
0.944 0.31	0 0	(Memory-referencing instruction did not specify a valid VA)
0.578 0.19	600 018 0.04	{structure:arc -}.{pointer+node_t=structure:node head}
0.522 0.17	0 0	{structure:arc -}.{cost_t=long org_cost}

Summary Event Legend Leak

Data for Selected Object:

Data Object: {structure:arc -}

Scope: (Global)

Type: structure:arc

Member of:

Offset:

Size: 64

Elements: 8

Offset	Size	Name
0	8	{structure:arc -}.{point
8	8	{structure:arc -}.{point
16	8	{structure:arc -}.{point
24	8	{structure:arc -}.{point
32	8	{structure:arc -}.{cost

List:

Process Times (sec.) / Counts

Data-derived

E\$ Stall Cycles: 169.701 (54.87%)

" count: 152731075441

E\$ Read Misses: 952528575 (59.51%)

Application Profiling–Analyzer

Data Layout Display

Performance Analyzer [test.1.er]

File View Timeline Help

Find Text: _____

Functions Callers-Callees Source Lines Disassembly PCs DataLayout DataObjects MemObj Timelit

E\$ Read Misses	E\$ Stall Cycles (sec.)	Name * +offset .element
144 039 383	24.211	<Total>
143 800 000	23.974	{structure:foo -}
300 000	0.042	/ +0 .(pointer+structure:foo fnext)
300 000	0.166	+4 .(pointer+structure:foo fright)
0	0.	+8 .(pointer+structure:foo fleft)
0	0.037	+12 .(int fstat)
143 200 000	23.674	+16 .(int fcode)
0	0.056	+20 .(int fval)
0	0.	+24 .(int fileft)
0	0.	\ +28 .(int finext)
0	0.	/ +32 .(int firight)
0	0.	\ +36 .(array[28]:char funused)
0	0.079	<Scalars>
0	0.079	(int iter)
239 383	0.158	<Unknown>
0	0.071	(Backtracking traversed a branch target)

Summary Event Legend Leak

Data for Selected Object:

Data Object: <Total>

Scope: (Global)

Type: (Synthetic)

Member of:

Offset:

Size: 0

Elements:

Process Times (sec.) / Counts

Data-derived

E\$ Read Misses:	144039383 (100.00%)
E\$ Stall Cycles:	24.211 (100.00%)
" count:	21790043527

Application Profiling-Analyzer

Memory Objects Display

Performance Analyzer [mcf.1.er]

File View Timeline Help

Find Text:

Functions Callers-Callees Source Lines Disassembly PCs DataLayout DataObjects MemObj Timeline

Display Mode: Text Graphical

Object Type: US3-L2-Cache US3-L2-Cache

E\$ Stall Cycles (sec.)	E\$ Stall Cycles (%)	E\$ Read Misses	E\$ Read Misses (%)	Name
309.307	100.00	1 600 644 190	100.00	<Total>
3.271	1.06	2 396 244	0.15	<Unknown>
0.178	0.06	100 003	0.01	US3-L2 Cache Line 10024 (0x2728)
0.167	0.05	100 003	0.01	US3-L2 Cache Line 16378 (0x3ffa)
0.167	0.05	100 003	0.01	US3-L2 Cache Line 2092 (0x82c)
0.133	0.04	400 012	0.02	US3-L2 Cache Line 8185 (0x1ff9)
0.133	0.04	100 003	0.01	US3-L2 Cache Line 7384 (0x1cd8)
0.122	0.04	300 009	0.02	US3-L2 Cache Line 6252 (0x186c)
0.111	0.04	100 003	0.01	US3-L2 Cache Line 2532 (0x9e4)
0.111	0.04	0	0.	US3-L2 Cache Line 3147 (0xc4b)
0.111	0.04	400 012	0.02	US3-L2 Cache Line 9108 (0x2394)
0.111	0.04	0	0.	US3-L2 Cache Line 5972 (0x1754)
0.111	0.04	500 015	0.03	US3-L2 Cache Line 8249 (0x2039)
0.111	0.04	400 012	0.02	US3-L2 Cache Line 3502 (0xdae)
0.111	0.04	400 012	0.02	US3-L2 Cache Line 8186 (0x1ffa)
0.111	0.04	100 003	0.01	US3-L2 Cache Line 3690 (0xe6a)
0.100	0.03	0	0.	US3-L2 Cache Line 3496 (0xda8)
0.100	0.03	400 012	0.02	US3-L2 Cache Line 8416 (0x20e0)

Summary Event Legend Leak

Data for Selected Object:

Memory Objects: US3-L2 Cache Line 10024 (0x2728)

Process Times (sec.) / Counts

Data-derived

E\$ Stall Cycles:	0.178 (0.06%)
" count:	159999973
E\$ Read Misses:	100003 (0.01%)

Application Profiling-Analyzer

Floating-point Performance Display

Performance Analyzer [test.flops.1.er]

File View Timeline Help

Find Text:

Functions Callers-Callees Source Lines Disassembly PCs Timeline LeakList Statistics Experiments

User CPU (sec.)	FP Adds	FP Mults	Name
40.839	384 317 824	287 000 371	<Total>
11.598	36 000 108	35 000 105	dgemv_g1_
9.667	36 000 108	36 000 108	dgemv_opt1_
8.076	0	0	load_arrays_
3.823	36 000 108	36 000 108	dgemv_g2_
1.461	36 000 123	36 000 144	dgemv_opt2_
1.121	0	0	__mt_WaitForWork_
1.051	36 000 195	35 000 224	dgemv_p1_ -- MP doall from line 12 [\$_d1A12.dgemv_p1_]
1.051	36 000 216	36 000 224	dgemv_p2_ -- MP doall from line 31 [\$_d1B31.dgemv_p2_]
0.971	36 000 204	36 000 228	dgemv_hil_
0.931	36 000 245	36 000 229	dgemv_hi2_
0.370	0	0	__mt_EndOfTask_Barrier_
0.360	32 000 128	0	barrier_ -- MP doall from line 20 [\$_d1A20.barrier_]
0.360	64 000 320	0	barrier_ -- MP doall from line 45 [\$_d1B45.barrier_]
0.	0	0	MAIN_
0.	0	0	__f90_allocate2
0.	0	0	__mt_Barrier_Init_
0.	0	0	__mt_MasterFunction_
0.	0	0	__mt_NewUserThread_
0.	0	0	__mt_SlaveFunction_

Summary Event Legend Leak

Data for Selected Object:

Name: dgemv_g1_

PC Address: 2:0x00008538

Size: 1276

Source File: /home/martyi/workarea/demos/cachetest

Object File: /home/martyi/workarea/demos/cachetest

Load Object: <cachetest>

Mangled Name:

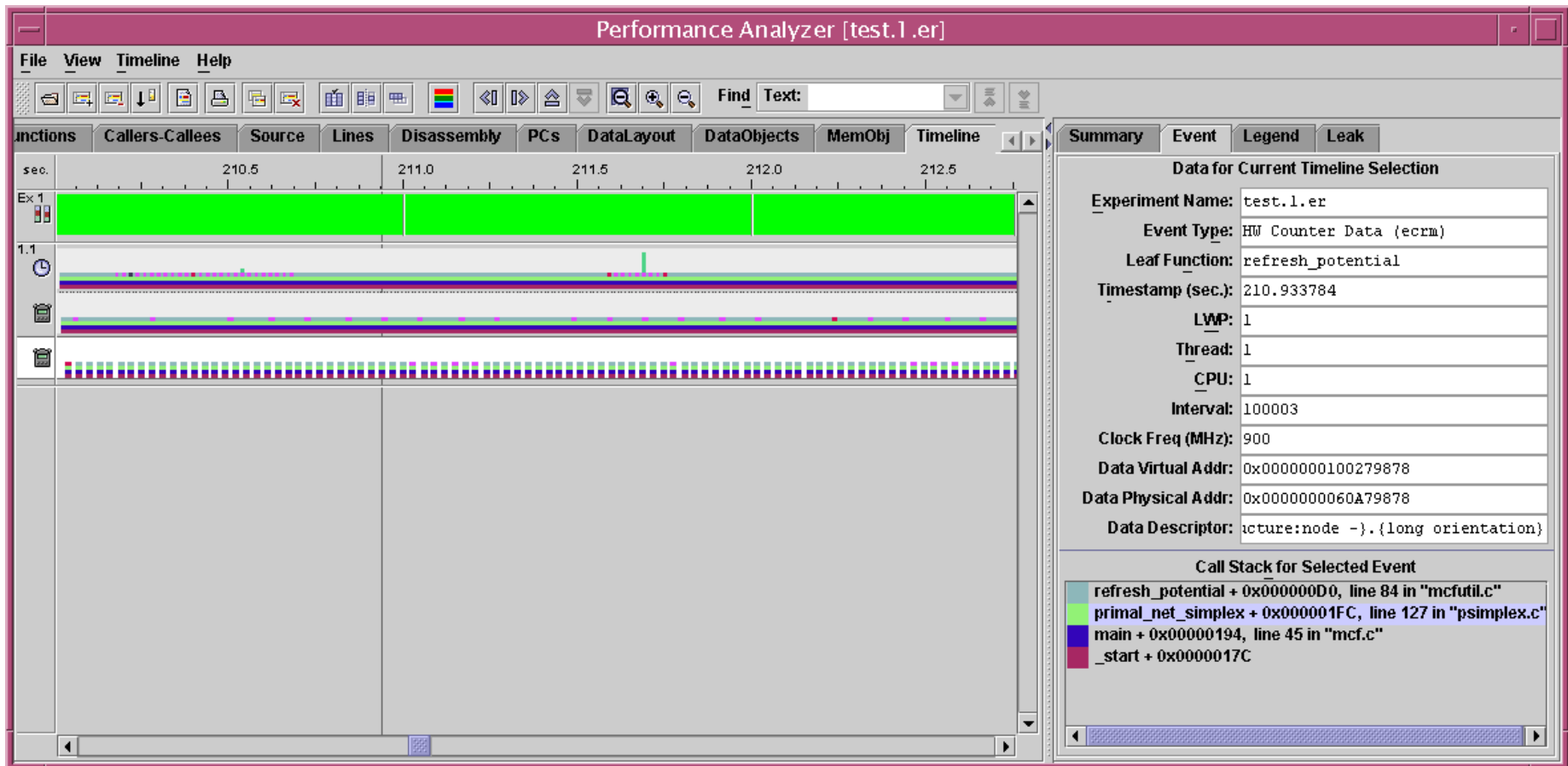
Aliases:

Process Times (sec.) / Counts

	Exclusive	Inclusive
User CPU:	11.598 (28.4%)	11.598 (28.4%)
Wall:	11.598 (29.2%)	11.598 (29.2%)
Total LWP:	11.598 (27.4%)	11.598 (27.4%)
System CPU:	0. (0.%)	0. (0.%)
Wait CPU:	0. (0.%)	0. (0.%)
User Lock:	0. (0.%)	0. (0.%)
Text Page Fault:	0. (0.%)	0. (0.%)
Data Page Fault:	0. (0.%)	0. (0.%)
Other Wait:	0. (0.%)	0. (0.%)

Application Profiling-Analyzer

Timeline Display



The screenshot shows the Performance Analyzer interface for a test named 'test.1.er'. The main window displays a timeline from 210.5 to 212.5 seconds. A prominent green bar indicates a selected event. Below the timeline, there are several horizontal bars representing different system metrics or threads.

The right-hand panel, titled 'Summary', provides detailed information about the current selection:

Data for Current Timeline Selection	
Experiment Name:	test.1.er
Event Type:	HW Counter Data (ecrm)
Leaf Function:	refresh_potential
Timestamp (sec.):	210.933784
LWP:	1
Thread:	1
CPU:	1
Interval:	100003
Clock Freq (MHz):	900
Data Virtual Addr:	0x00000000100279878
Data Physical Addr:	0x0000000060A79878
Data Descriptor:	structure:node -.}{long orientation}

Below the summary, the 'Call Stack for Selected Event' is displayed:

- refresh_potential + 0x000000D0, line 84 in "mcfutil.c"
- primal_net_simplex + 0x000001FC, line 127 in "psimplex.c"
- main + 0x00000194, line 45 in "mcf.c"
- _start + 0x0000017C

Kernel Profiling-Analyzer

- Similar invocation to **collect**
 - Can use loadable driver (internal tool)
 - For clock- or HWC-profiling
 - Use DTrace (Solaris 10)
 - Clock-profiling now
 - HWC-profiling coming
- Experiment read with Analyzer
 - Same as user profiling
- Can record simultaneous user/kernel

Sun's Monitoring Tools

- Application monitoring tools
 - **cputrack**
 - **ripc2** (built on top of **cputrack**) (internal tool)
 - **bw** (internal tool)
- System monitoring tools
 - **cpustat**
 - **busstat**
- Allow/exploit multiplexing of HW counters
 - Valid statistics much more likely than for profiling
 - Use to see which counters to profile against

Application Monitoring-~~qtrack~~ qtrack output — periodic, multiplexed

```

U.1U3 1110U 1 tick 117322342 16629 # picU=Cycle_cnt,pic1=DTLB_miss,sys
0.206 11100 1 tick 59287053 29285156 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
0.304 11100 1 tick 544973 5320111 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
0.403 11100 1 tick 259749 8 # pic0=Dispatch0_br_target,pic1=Re_PC_miss,sys
0.503 11100 1 tick 97217 452879 # pic0=Dispatch0_2nd_br,pic1=Dispatch0_mispred,sys
0.603 11100 1 tick 396249 2962149 # pic0=Dispatch_rs_mispred,pic1=Re_RAW_miss,sys
0.703 11100 1 tick 79686301 5287 # pic0=Rstall_storeQ,pic1=Rstall_FP_use,sys
0.872 11100 1 tick 3100459 18 # pic0=Rstall_IU_use,pic1=Re_FPU_bypass,sys
0.932 11100 1 tick 67885860 176 # pic0=Cycle_cnt,pic1=DTLB_miss,sys
1.052 11100 1 tick 148401091 2241460 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
1.112 11100 1 tick 62640 131944 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
1.222 11100 1 tick 359 8 # pic0=Dispatch0_br_target,pic1=Re_PC_miss,sys
1.352 11100 1 tick 180 2989 # pic0=Dispatch0_2nd_br,pic1=Dispatch0_mispred,sys
1.412 11100 1 tick 633 89865 # pic0=Dispatch_rs_mispred,pic1=Re_RAW_miss,sys
1.532 11100 1 tick 297242 35764030 # pic0=Rstall_storeQ,pic1=Rstall_FP_use,sys
1.612 11100 1 tick 1359398 8 # pic0=Rstall_IU_use,pic1=Re_FPU_bypass,sys
1.712 11100 1 tick 116030624 20 # pic0=Cycle_cnt,pic1=DTLB_miss,sys
1.832 11100 1 tick 150258502 7114364 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
1.922 11100 1 tick 44256 3965119 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
2.032 11100 1 tick 16478 4 # pic0=Dispatch0_br_target,pic1=Re_PC_miss,sys
2.132 11100 1 tick 3102 32512 # pic0=Dispatch0_2nd_br,pic1=Dispatch0_mispred,sys
2.212 11100 1 tick 1843 3160 # pic0=Dispatch_rs_mispred,pic1=Re_RAW_miss,sys
2.342 11100 1 tick 15692 1659570 # pic0=Rstall_storeQ,pic1=Rstall_FP_use,sys
2.492 11100 1 tick 128468 4 # pic0=Rstall_IU_use,pic1=Re_FPU_bypass,sys
2.522 11100 1 tick 32731512 43 # pic0=Cycle_cnt,pic1=DTLB_miss,sys
2.612 11100 1 tick 134166672 26865067 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
2.732 11100 1 tick 75839 4082663 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
2.822 11100 1 tick 14306 6 # pic0=Dispatch0_br_target,pic1=Re_PC_miss,sys
2.912 11100 1 tick 2664 29214 # pic0=Dispatch0_2nd_br,pic1=Dispatch0_mispred,sys
3.062 11100 1 tick 1691 16143 # pic0=Dispatch_rs_mispred,pic1=Re_RAW_miss,sys
3.122 11100 1 tick 10685 738362 # pic0=Rstall_storeQ,pic1=Rstall_FP_use,sys
3.212 11100 1 tick 102127 8 # pic0=Rstall_IU_use,pic1=Re_FPU_bypass,sys
3.342 11100 1 tick 152910322 68 # pic0=Cycle_cnt,pic1=DTLB_miss,sys
3.452 11100 1 tick 166866481 31695212 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
3.512 11100 1 tick 48366 2928255 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
3.662 11100 1 tick 20517 6 # pic0=Dispatch0_br_target,pic1=Re_PC_miss,sys
3.722 11100 1 tick 1838 19269 # pic0=Dispatch0_2nd_br,pic1=Dispatch0_mispred,sys
3.812 11100 1 tick 1781 21546 # pic0=Dispatch_rs_mispred,pic1=Re_RAW_miss,sys
3.912 11100 1 tick 17800 1243820 # pic0=Rstall_storeQ,pic1=Rstall_FP_use,sys
4.022 11100 1 tick 122989 8 # pic0=Rstall_IU_use,pic1=Re_FPU_bypass,sys
4.132 11100 1 tick 128806419 53 # pic0=Cycle_cnt,pic1=DTLB_miss,sys

```

Application Monitoring-ripc2

Aggregated ~~output~~ **output**

```

Application stall information
Application information
-----
Ultra-III          est-
                   ticks      sec      %
-----
DO_IC_miss         2933194905      2.782    0.3%
DO_br_targ_calc    709057035       0.673    0.1%
DO_2nd_br          9375570         0.009    0.0%
DO_mispred         422139615       0.400    0.0%
D_rs_mispred       24932760        0.024    0.0%
Rs_storeQ          83294492895     79.015   9.1%
Rs_FP_use          57868519605     54.895   6.3%
Rs_IU_use          9970661385      9.458    1.1%
Re_FPU_bypass      2220            0.000    0.0%
Re_RAW_miss        2686694685     2.549    0.3%
Re_DC_miss         464061620820    440.218  50.6%
Re_EC_miss         142546870395    135.223  15.5% (in DC miss)
Re_PC_miss         4470            0.000    0.0%
DTLB_miss          74348460        5.995    0.7%
total              628300315065    596.018  68.5%
-----
time              917205835095    870.080  100.0%
instr             505747594035
IPC               0.551 (instr/time)
Grouping         1.751 (instr/(time-total))
-----
unfinished fpop    0
-----
Ultra-III          events  evnt/instr  %
-----
IC_ref            189100726920    0.206    100.0%
IC_miss           68170380        0.000    0.0% of IC_ref
DC_rd             188104677270    0.205    100.0%
DC_rd_miss        8327843370      0.009    4.4% of DC_rd
DC_wr             63370469025     0.069    100.0%
DC_wr_miss        41307834105     0.045    65.2% of DC_wr
EC_ref            236891795940    0.258    100.0%
EC_misses         8228401335      0.009    3.5% of EC_ref
EC_rd_miss        557599905       0.001    6.7% of DC_rd_miss
EC_ic_miss        16296120        0.000    23.9% of IC_miss
ITLB              29400           0.000    0.0% of instructions
FP inst          A= 97990594365 M=100439497380 39.2% of instructions
-----

```

Measurements of which processor events contributed to stall time.

Stall times

Floating point traps

Events per instruction

System Monitoring-cpustat

System-wide version of cputrack

- CPU counters

```
# cpustat -c pic0=Instr_cnt,pic1=Re_DC_miss,sys -c pic0=Dispatch0_IC_miss,pic1=Re_EC
  time cpu event      pic0      pic1
  5.009  1  tick 2059035807  33974649 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
  5.010  0  tick 1551600712  22778438 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 10.009  0  tick 457169390   37188376 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 10.010  1  tick 328284013   41892660 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 15.009  1  tick 1451190423  157834999 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 15.010  0  tick 1462169361  341406198 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 20.009  1  tick 570276705   226545705 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 20.010  0  tick 586545226   126403413 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 25.009  0  tick 1279091800  122489122 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 25.010  1  tick 1850824519  363689893 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 30.009  1  tick 493026173   128362677 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 30.010  0  tick 586294847   81941837 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 35.009  0  tick 1946205173  229090921 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 35.010  1  tick 1351275940  175492857 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 40.009  1  tick 555076554   74295003 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 40.010  0  tick 570543507  101949954 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 45.009  0  tick 1881264848  286191421 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 45.010  1  tick 1318075599  133867092 # pic0=Instr_cnt,pic1=Re_DC_miss,sys
 50.009  0  tick 577788523  111924098 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
 50.010  1  tick 646882743  155645759 # pic0=Dispatch0_IC_miss,pic1=Re_EC_miss,sys
```


System Monitoring-cpustat

System-wide version of `cputrack`

- Memory counters

```
# cpustat -c MC_reads_0,MC_writes_0,sys -c MC_reads_1,MC_writes_1,sys -c MC_reads
time cpu event pic0 pic1 # pic0=MC_reads_0,pic1=MC_writes_0,sys
5.010 0 tick 379284 435924 # pic0=MC_reads_0,pic1=MC_writes_0,sys
5.010 1 tick 2 2 # pic0=MC_reads_0,pic1=MC_writes_0,sys
10.010 1 tick 2 2 # pic0=MC_reads_1,pic1=MC_writes_1,sys
10.010 0 tick 83775 2800104 # pic0=MC_reads_1,pic1=MC_writes_1,sys
15.010 1 tick 2 2 # pic0=MC_reads_2,pic1=MC_writes_2,sys
15.010 0 tick 416247 480723 # pic0=MC_reads_2,pic1=MC_writes_2,sys
20.010 0 tick 179454 2909166 # pic0=MC_reads_3,pic1=MC_writes_3,sys
20.010 1 tick 2 2 # pic0=MC_reads_3,pic1=MC_writes_3,sys
25.010 0 tick 2157225 1011898 # pic0=MC_reads_0,pic1=MC_writes_0,sys
25.010 1 tick 2 2 # pic0=MC_reads_0,pic1=MC_writes_0,sys
30.010 0 tick 385880 3055634 # pic0=MC_reads_1,pic1=MC_writes_1,sys
30.010 1 tick 2 2 # pic0=MC_reads_1,pic1=MC_writes_1,sys
35.010 1 tick 2 2 # pic0=MC_reads_2,pic1=MC_writes_2,sys
35.010 0 tick 304115 426714 # pic0=MC_reads_2,pic1=MC_writes_2,sys
40.010 0 tick 656979 3392755 # pic0=MC_reads_3,pic1=MC_writes_3,sys
40.011 1 tick 2 2 # pic0=MC_reads_3,pic1=MC_writes_3,sys
45.010 0 tick 1668996 746562 # pic0=MC_reads_0,pic1=MC_writes_0,sys
45.010 1 tick 2 2 # pic0=MC_reads_0,pic1=MC_writes_0,sys
50.010 1 tick 2 2 # pic0=MC_reads_1,pic1=MC_writes_1,sys
50.010 0 tick 738173 3366560 # pic0=MC_reads_1,pic1=MC_writes_1,sys
55.010 1 tick 2 2 # pic0=MC_reads_2,pic1=MC_writes_2,sys
55.010 0 tick 1449829 728740 # pic0=MC_reads_2,pic1=MC_writes_2,sys
60.010 1 tick 2 2 # pic0=MC_reads_3,pic1=MC_writes_3,sys
60.012 0 tick 891850 3072477 # pic0=MC_reads_3,pic1=MC_writes_3,sys
65.010 1 tick 2 2 # pic0=MC_reads_0,pic1=MC_writes_0,sys
65.010 0 tick 1042867 575276 # pic0=MC_reads_0,pic1=MC_writes_0,sys
```

Application Monitoring-bw

Show memory bandwidth of application

- Based on **cpustat**
 - Read memory counters, message output

```
nubbins% bw <target>
```

```
-----  
Read  memory bandwidth: 1023.58642578125 MB/sec (total bytes = 17172930560)  
Write memory bandwidth: 998.464149475098 MB/sec (total bytes = 16751448704)  
Total memory bandwidth: 2022.05057525635 MB/sec (total bytes = 33924379264)  
Elapsed time      : 16 secs  
-----
```

System Monitoring-busstat

Bus performance

```
# busstat -w pcis0
time dev      event0          pic0          event1        pic1
1    pcis0      dvma_stream_rd 200           dvma_stream_rd 200
2    pcis0      dvma_stream_rd 98            dvma_stream_rd 98
3    pcis0      dvma_stream_rd 197           dvma_stream_rd 197
4    pcis0      dvma_stream_rd 93            dvma_stream_rd 93
5    pcis0      dvma_stream_rd 90            dvma_stream_rd 90
6    pcis0      dvma_stream_rd 89            dvma_stream_rd 89
7    pcis0      dvma_stream_rd 312           dvma_stream_rd 312
8    pcis0      dvma_stream_rd 92            dvma_stream_rd 92
9    pcis0      dvma_stream_rd 100           dvma_stream_rd 100
10   pcis0      dvma_stream_rd 9204          dvma_stream_rd 9204
11   pcis0      dvma_stream_rd 2688          dvma_stream_rd 2688
12   pcis0      dvma_stream_rd 22237         dvma_stream_rd 22237
13   pcis0      dvma_stream_rd 98            dvma_stream_rd 98
14   pcis0      dvma_stream_rd 111           dvma_stream_rd 111
15   pcis0      dvma_stream_rd 116           dvma_stream_rd 116
16   pcis0      dvma_stream_rd 104           dvma_stream_rd 104
17   pcis0      dvma_stream_rd 515           dvma_stream_rd 515
18   pcis0      dvma_stream_rd 1330          dvma_stream_rd 1330
19   pcis0      dvma_stream_rd 1252          dvma_stream_rd 1252
20   pcis0      dvma_stream_rd 1172          dvma_stream_rd 1172
21   pcis0      dvma_stream_rd 189           dvma_stream_rd 189
22   pcis0      dvma_stream_rd 100           dvma_stream_rd 100
23   pcis0      dvma_stream_rd 90            dvma_stream_rd 90
24   pcis0      dvma_stream_rd 108           dvma_stream_rd 108
25   pcis0      dvma_stream_rd 100           dvma_stream_rd 100
26   pcis0      dvma stream rd 989           dvma stream rd 989
```

HW Counter Library APIs

- Typical use by users
 - Read before and after block of code
 - Compute deltas
- API needs
 - Select counters, preset and read values
 - System call access
 - Direct user-mode reading?
 - Should be lower overhead than system call
- Profiling APIs
 - Specify counters, intervals for each
 - Usually for tool developers, not end users

HW-Counter Requirements, I

- Documented behavior
 - Actual behavior, not design intent or schematics
 - *Verified behavior!*
 - If they're not verified, they don't work
- Counters for all critical resources
 - Interesting events
 - Stall-cycle counts, not just event counts
 - Measures the *cost* of events
- Wide counter registers
 - Minimize need for OS to “widen” them to 64-bits
- As many registers as possible

HW-Counter Requirements, II

- Counter contexts
 - Counters virtualizable to user context
 - *i.e.*, Kernel save and restore on context switch
 - Overflow attributable to user context
 - System-wide – for system monitoring
- CMT Issues
 - Independent counter sets for each HW thread
 - Counters for resource conflict among threads
 - Measure number of active threads (binning?)
- Wide-instruction chips
 - Counters for how many instructions in each cycle

HW-Counter Requirements, III

- Counter interface – kernel driver
 - Read and write entire width of counter
 - Specify arbitrary interval for profiling
 - Specify arbitrary method for sampling

HW-Counter Requirements, IV

- Profiling-interrupt infrastructure
 - Easily attributable to specific counter
 - Delivered as precisely as feasible
 - *i.e.*, interrupt is synchronous to the context triggering it
 - If interrupt is deferred, allow SW to wait deterministically
 - Delivered with VA/PA (memory-related counters)
 - User needs to know which data objects are getting misses
- Instruction-sampling infrastructure
 - *c.f.*, Profile-me
 - Detailed information on sampled instruction

HW-Counter Constraints, I

- Everything is a tradeoff ...
 - Complexity, alas, is *somewhere*
 - SW wants it in HW
 - HW wants in in SW
- Area constraints
 - Lead to pressure to reduce number of counters
 - But SW wants infinite number (approximately)
 - Lead to pressure to reduce width of counters
 - But SW prefers to never deal with rollover
 - But still need comparator to trigger profile interrupt
 - Are especially problematic for CMT

HW-Counter Constraints, II

- Routing constraints
 - Lead to delays due to long wires
 - Imply difficulty with counter placement
 - May make precise, synchronous delivery difficult
- Complex pipelines
 - Make precise, synchronous delivery difficult
- Mapping events to registers
 - SW wants any event to map to any register

Acknowledgements

(Alphabetic)

Russ Blaine, SW, Solaris

Darryl Gove, SW, CPE

Adam Talcott, HW, SSG

Mario Wolczko, Sun Labs



Marty Itzkowitz

marty.itzkowitz@sun.com

