Chapter 13

Solaris Support for the UltraSPARC T2 Processor

Karsten Guthridge
Compiler Performance Engineering

Solaris for T2

- Release vehicle for T2 was S10 Update 4
- Several performance fixes did not make the release so patches are necessary for best performance
- Many generic T2 optimisations
  - Also a number of T5x20 specific optimisations
- Optimizations introduced in S10U5
  - Shared context allows the TLB to be used more efficiently by processes using shared memory
  - Many others

Solaris for T2

- Crypto support for T2 hardware
- nxge driver for 10Gig
- T2 Perf counters including counter overflow support
- Page coloring optimizations - Hashed Cache Index
- Out Of the Box Page Performance
- MPSS for non ISM shared memory
- trapstat changes
- Processor Groups optimization
- Stack address skew
- bcopy and memcpy optimizations including uiomove
- Shared context optimization

Performance Counters

- Significant changes from T1 which only had 7, much deeper insight into performance
- Counter overflow works in T2, can be used in the Workshop Analyzer
- Cpustat -h will give details of counters
- Note the format of the command in T2 is back to the standard SPARC implementation where event 1 and event 2 can be any counter

Performance Counters

- New counters of interest
  - Idle_strands – the number of cycles no strand could be picked for the pipeline
  - Instruction breakdown with Instr_ld Instr_st Instr_sw Instr_other Atomics
  - Hardware Table Walk Counters e.g. ITLB_HWTW_miss_L2
  - Crypto hash and cipher operations
    - AES_op
    - AES_busy_cycle
    - Others...

Performance Counters

- There is a new version of corestat which measures:
  - Utilization of multiple integer pipelines
  - Utilization of per core FPU
- Downloads and documentation at http://cooltools.sunsource.net/corestat/

Hash Cache Index

- 64 threads can cause a lot of conflicts in the L2 cache
- The key to reducing conflicts is an optimal placement of memory pages
  - choose of physical addresses so they don't create hot lines in the L2 cache
- T2's L2 cache hashes (XORs) the low order physical address bits required for the cache index with high order physical address bits to randomize accesses
  - This is called "Hash Cache Indexing"
- RFEs 6409758 and 6409758 implement HCI
  - Scaling improved significantly on many applications

Out of the Box Page Performance

- Page selection default configurations on T1 and T2 are now more optimal
  - T1
    - Private anon, heap, process stack, initdata segments all request 64 KB pages
    - Text, shared and anon segments request 4 MB pages
    - Intimate Shared Memory and Dynamic Intimate Shared Memory (ISM and DISM) request 256 MB pages
  - T2 (changes because of HCI)
    - Initdata segments request 64 KB pages
    - Private anon, heap, process stack, text, shared anon segments request 4 MB pages
    - ISM and DISM request 256 MB pages

Out of the Box Page Performance

- Stack, heap and anon values can be changed by using MPSS
- With RFE 6371967 anon pages created using mmap of /dev/zero will have large pages
- New set of variables to control page size – use with caution
  - `max_uheap_lpsize`
  - `max_ustack_lpsize`
  - `max_privmap_lpsize`
  - `max_shm_lpsize`
  - `max_uidata_lpsize`
  - `max_utext_lpsize`
  - `max_shm_lpsize`

MPSS for non ISM Shared Memory

- SAP allocates a lot of shared memory but cannot use ISM or DISM because of the need to use mprotect()
- SAP SHM currently on 8k which hurts Niagara performance
- RFE 4614772 adds large page support for non ISM shared memory
- Performance gains of 20% on our internal SAP tests

Stack Biasing

- Data cache is 4-way associative but on T2 we can have 8 threads sharing it
- When running multiple copies of the same binary on T2, variables on the process stack can become aligned in the L1 Data Cache
- This can cause High L1 cache misses and poor scaling
- RFE 6493685 adds a Stack Bias which slews the stack variables
- 19% performance increase seen in some internal benchmarks

Trapstat

- T2 has Hardware Table Walk (HWTW) acceleration for both Instruction and Data TLB misses
- Because TLB misses are no longer executed in software, trapstat cannot determine the number of misses
  - trapstat -T on T2 will report zero TLB activity for all page sizes in both user and kernel mode
  - Percentage time reported will also be zero
  - Performance counters need to be used to measure the TLB miss rate
- Translation Storage Buffer (TSB) misses are still handled in software so TSB results reported by trapstat are accurate

Multi-level CMT Scheduling Optimizations

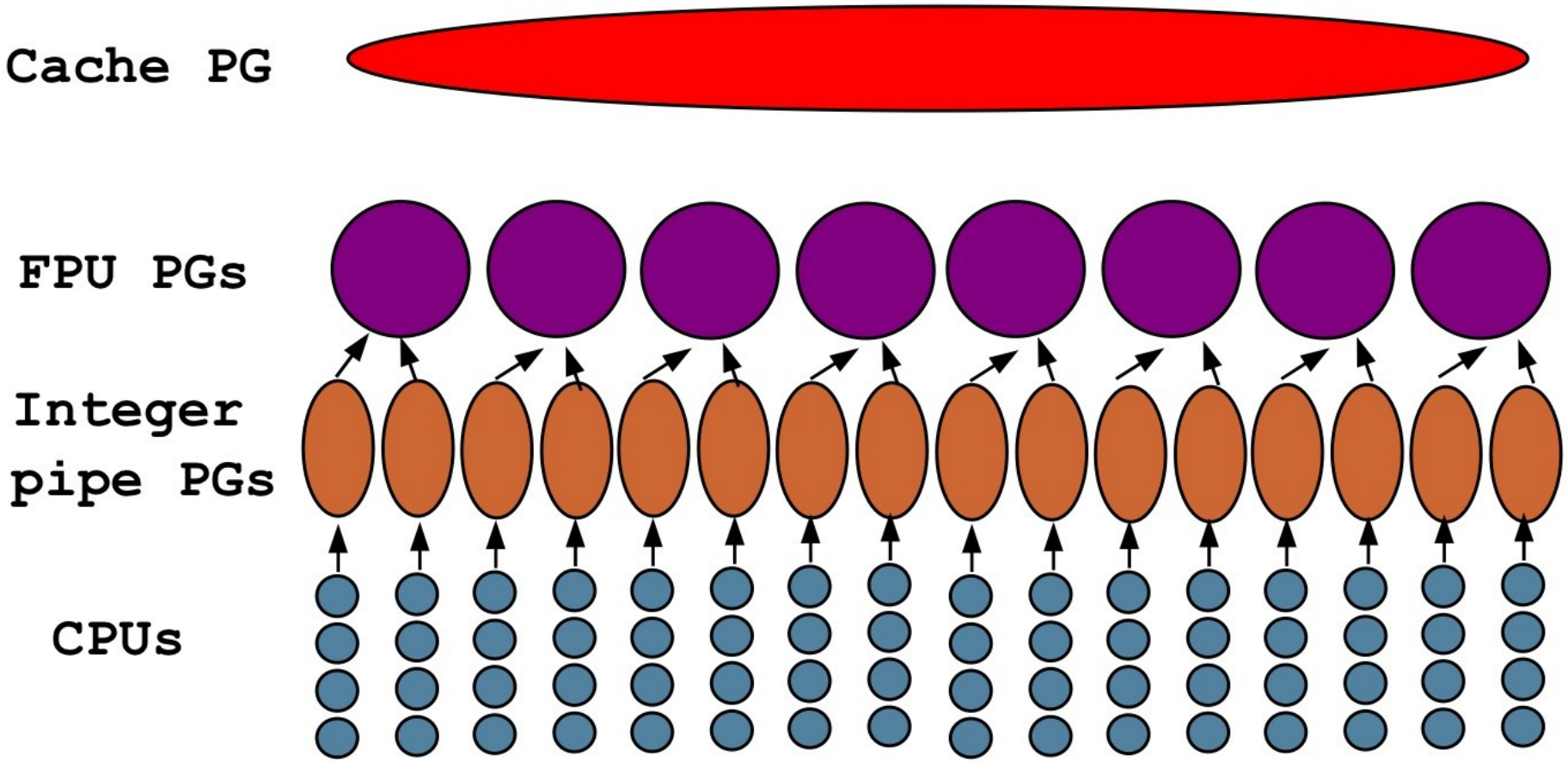
- Processor Groups
  - Abstraction introduced to capture a group of CPUs with some (hardware) sharing relationship
    - Int/FP pipelines, caches, chips, MMUs, crypto units, etc.
  - PGs are used by dispatcher to implement multi-level CMT load balancing and affinity polices
  - Replaces chip_t
- Niagara II
  - Int/FP pipelines are grouped, so threads are balanced across both levels
    - 8 threads => 1 per core (FPU), 1 per integer pipeline
    - 32 threads => 2 per core (FPU), 2 per integer pipeline
  - 7% SPECfp_rate performance improvement on T5220 (8 processes)
  - Work to characterize FPU consuming vs. non-consuming threads is underway to improve heterogeneous workload system performance

Thread Balancing

- disp.c: cmt_balance()
  - Replaces chip_balance()
  - Uses a top-down load balancing policy to balance running thread load across the levels in the load balancing lineage created by the CMT Processor Group (PG) class
  - Try to balance across the chips, then the cores, then the pipelines etc
- disp.c: disp_getbest()
  - Is more aggressive about stealing work if that CPU shares a cache with ours

CMT Processor Group Class

- Example: Niagara II



Bcopy Optimizations

- Bcopy and uiomove are key kernel routines for performance
- RFE 6492718 implements a new bcopy routine for sun4v
  - Roughly 2X as fast than current version
  - SpecWeb2005 has seen a 10% increase in performance
  - Your mileage will vary depending on amount of bcopy in a workload
- RFE 6500001 implements the same algorithm in uiomove which copies buffers to and from the kernel

Performance bugs and Patches

Patching T2-Based Systems

- A number of key performance fixes did not make it into S10U4
- Use kernel patch 127111-05 to fixe nearly all the known performance issues if you are still running S10U4
- Bottom line: make sure you have your system patched if performance is important!