Hardware Counter Working Group: Detailed Discussion Notes
Hardware Counter Support

- Intel
  - PEBS
    - captures IP of next dynamic instruction (e.g. branch target)
  - LBR hardware support (last branch record)
    - captures last taken branch record: (src, target) pair
      branches, calls, interrupts?
    - Nehalem +: 16-deep src and target
    - can filter on certain kinds of branches (e.g. conditional)
  - Advice: must stop LBR on PMU interrupt (this is a mode)
    - Nehalem
      must not set LBR stop bit for PEBS events to stop LBR recording
      stopping LBR is handled by PEBS microcode instead
      set LBR stop bit for non-PEBS events
    - Westmere - set the bit always
  - PEBS can be used to trigger on function call
    - one can use it to perform function call argument sampling
      can capture machine state. using dwarf can get args
    - NOTE: on a function call, you want the LBR target, not the LBR source
Linux perf_events

- Uses libpfm4: convert from event string to event encoding
- perf_events provides support for RAW
  - can monitor everything
- perf_events must schedule counters
  - kernel only needs to know about ones with problems
- libpfm4 attr.type=RAW; config = ...
Linux perf_events: Kernel Revisions

- 2.6.32: thread centric signal delivery
- 2.6.35: PEBS support
  - can only extract instruction address, not rest of machine state
- 2.6.36: IBS support?
  - AMD has IBS patch; won’t be in a release before 2.6.36
  - Cray could potentially use AMD patch in their upcoming kernel refresh (expected to be 2.6.32)
Linux perf_events and PEBS

• PEBS
  — kernel activates LastBranchRecord (LBR)
  — kernel tries to recover responsible IP using PEBS addr + LBR,
    – for a branch instruction, just look at LBR to identify inst addr
    – non branch: disassembles forward in last block to find proper IP
  — when sampling calls, you typically don’t want the IP of the call
    instruction, you want the IP of the target
    – this differs from the norm!

• Nehalem+ support
  — request type x response type
    – measure on per thread basis
      useful for non-precise memory access
    – capturing off core response requires extra MSR programming
    perf_events lacks ability to use MSR pair for configuration
    hardware problem: extra MSR shared with hyperthread
      needs to be managed

• PMU sharing is a problem
  — if two processes want to use the PMU, both can’t be scheduled
    as hyperthreads on a shared core
Using PEBS with perf_events

- perf_event abstraction
  - precision flag, values 0-3
    - skid = 2, skid = 0
- PEBS enabled in 1-shot mode
- Event record type TID, IP, TSTAMP
- Enabling PEBS: set flag for precise sampling to 2-3
Linux perf_events and IBS

• **Type RAW**
  — raw_type = 1

• **Config:**
  — MSR, sampling period
    – sample period = (20 bit sampling period; bottom 4 bits 0)
      shanghai+ - now randomizing bottom 6 bits of period

• **IBS modes**
  — IBS OP - 5 reg
    – useful for cache misses
    – sample on UOP or cycles
  — IBS fetch - 3 reg
  — SAMPLE_RAW record (size x # bytes)
Linux perf_events Action Items (1)

• Randomization

• Add IBS support and expose IBS data

• Intel processor issues
  — expose PEBS machine state
  — cycle counting problem with perf_events
    – core and ref cycles have same encoding, just different counter
    – perf_events doesn’t allow one to name a register for counting
    – without support for both, can’t understand freq scaling
  — monitoring off-core response on Intel processors
    – requires MSR pair and needs special perf register scheduling code within the kernel
  — PEBS native support for IP
    – without having kernel apply LBR correction
Linux perf_events Action Items (2)

- Intel processor issues (continued)
  - expose LBR
    - feature request: perf record -lbr
    - need use cases to motivate the need to export LBR information
  
  perf report
  
  report src and dest of branches

basic block profile support to perf_events_tool

idea: use ROSE to output start and end of basic blocks

correlate them with LBR info
Driver Support for HW Counters

• Drivers only support system wide monitoring

• Available drivers
  — Intel processors
    – PTU has driver source
      system wide monitoring. need to add support for per thread
      can’t send a signal from an interrupt handler
    – likwid
  — AMD processors
    – AMD has IBS patch for perf_events (see Robert Richter)
    – likwid
Attributing Costs of Blocking

- Hypothetical processors
  - Capture ring 3 to ring 0 transition & LBR
    - capture the trajectory into the kernel
  - Capture ring 0 to ring 3
    - capture trajectory out of the kernel

- Using perf_events tracepoints
  - monitor syscalls
  - when you cross the tracepoint
    - you can count
    - you can use it in sampling mode
  - use the timestamp counter to check crossing timestamp
    - can’t use gettimeofday because that is a syscall
Analyzing Performance with Multiplexing

- Levinthal
  - multiplex over 100ms intervals
  - every group samples every phase
  - want to collect for about an hour
  - significant events will fire at 1K/s frequency
  - Westmere processors
    - 57 & 83 event sets
    - tree diagram for performance diagnosis
  - Nehalem multiplex set ~50 events
Monitoring Load Latency (Intel)

- Program minimum threshold
- Loads are sampled at random
  - capture address
  - if above threshold, increment counter
  - if overflow, capture the details
    - where did the cache line come from, ...
  - Local DRAM retired at particular address is correct

- Issue
  - fraction of events sampled depends on workload
  - is dispatch to availability latency an advantage?

- Analysis technique: when lots of cores
  - very high latency (1000+) gives highly contended locks

- Can’t normalize load latency because you don’t know how many you dropped
Steps Forward

• Reconcile PAPI events with Intel event names
  — Terpstra and Eranian to send libpfm event name encoding to Levinthal. He will validate encodings for Nehalem & Westmere
    – Levinthal: some event counters are problematic
      PAPI needs to know which
    – PTU has correct tables that can serve as a reference rather than manual “3B” (only correct version for Westmere)
      Eranian can obtain the up to date copy. (not on whatif today)

• David Levinthal to pursue DOE-wide license for Intel tools
  — like a corporate wide license
  — action item: Levinthal needs point of contact
    – David Skinner will investigate possibilities
  — Intel driver under GPL, but support packages not

• Make sure that Cray understands the limits of the kernel they adopt
  — urge them to add appropriate patches
    – IBS support patch from Robert Richter @ AMD
Steps Forward

• Multiplexing sets for Intel processors
  — Levinthal constructed event sets for CERN
    – perhaps these are more detailed than we need
    – CERN multiplex set
      standard stuff
      additional support for diagnosing IFETCH problems for OO code
  — action item: Levinthal to send to Terpstra, Eranian, M-C, Itzkowitz
  — PTU GUI: “get command” interface can be used to extract info
  — Levinthal diagnosis tree
  — CERN
    – source and asm + performance data: turn it into linked HTML
    – use the tree diagram to decide what is interesting

• Open source display of performance information
  — need open source file format to support tools
    – header gives location of load points of modules. what was recorded, etc.
    – variable length record per event.
      fixed part: (IP, thread id, timestamp, module)
      extra stuff: LBR, PEBS
Performance Counters and Virtualization

- Wanted: performance counters for guest OS
- Need to swap out performance counters when world swaps
- Issues
  - uncore event counts cannot be assigned to any thread
  - different guest OS can be hyperthreaded and sharing a core
  - counters associated with a shared resource can’t be attributed on a per thread basis
- Not a problem (believe that proper support provided)
  - Solaris zones
  - AIX hypervisor