PAPI BOF

Philip J. Mucci

UTK/ORNL
SiCortex
PAPI 3.5 Release

- Platforms
- Build environment
- Features
- Bug fixes
- Bugs
New Platforms

• Linux/Intel Core, Core 2 Duo, Dual Core AMD
• Windows/AMD64
• Perfmon2 targets
  – All perfmon platforms supported for Cycles, Instructions and all Native events.
  – Supported Presets
    • MIPS5K, 20K, 24KF, 25KC
    • IA64/Montecito
    • Opteron
Montecito Support

• Working for Perfmon2 kernels
• Broken for unpatched kernels, awaiting access to hardware.
• Data address sampling working.
  – PAPI_s/profile on data space!
Retired Platforms

• Linux/Alpha, Tru64/Alpha
• AIX/ Power N (N <= 3)
Build Environment

• Full adoption of GNU configure
• Functional in cross compilation environments
  – Cray XT3
  – IBM BG/L
• Install targets standardized
PAPI Attach/Detach

• New API call to support third-party access.
• Only implemented for Linux systems
  – Perfmon2 & PerfCtr
• Requires target PID be ptraced and stopped.
• Every eventset can be attached.
Kernel-assisted Multiplexing

• Allows PAPI to use the in-kernel implementation of multiplexing.
  – Less control over algorithm
  – More control over interference
  – More precise intervals

• Platforms
  – IRIX/MIPS
  – Linux/Perfmon2
Substrate information

• New interface to substrate information
• Allows
  – Changing previously fixed behavior at runtime
  – Exporting important bits of information to tools
Linux Kernel support?

• Perfmon2 undergoing active development by Stefane E. from HP, with some help from myself and others.
• Actively being reviewed by LKML and piece by piece, is being accepted into the mainline.
• Current support: x86, x86_64, MIPS and IA64.
Perfmon 2 Architecture Summary

Perfmon2 and libpfm

- Perfmon2 provides the means to program the registers.
- It does not dictate the register contents!
- This is often even more work than getting the kernel components correct.
- Perfmon2 comes with libpfm to help.
Perfmon2

- Measurement types
  - Counting
  - Sampling
- Scopes
  - System-wide
  - Per-thread
- Views
  - First person
  - Third Person
- Integration
  - Cooperates with Oprofile

The following slides borrow heavily from Stefane Eranian's talk at OLS2006 at http://perfmon2.sourceforge.net/ols2006-perfmon2.pdf
Perfmon2 (2)

- Counters virtualized to 64-bit
- Logical view of PMD's and PMC's, not machine specific.
- System call approach rather than driver approach.
- Compatible with existing mechanisms
  - Mmap, signals, ptrace, etc...
Perfmon2 Sampling

- Traditionally sampling has been looking at the IP upon PMD interrupt, passed through to the user through a signal context.
- IA64 and PPC64 series introduced address and branch sampling.
- Perfmon2 provides access to buffered, customized sampling of any PMU resource.
Perfmon2 Multiplexing

- PAPI has had the ability to multiplex counters for a while, but it does this at user level with signals and a timer.
- Perfmon2 can do this in the kernel.
  - Much lower overhead.
  - Less pollution of user counts.
  - Provides switching based on PMD overflow or clock.
PAPI 4.0

• Multisubstrate support
  – CPU
  – Network
  – Sensor data
• Lower number of entry points into substrates
  – Allow incremental porting of features
• Remove redundant API calls
• More support for Branch and Data address sampling
  – IA64/PPC64 and Perfmon2
PAPI 4.0

• Sampling API
  – Allows event and time based sampling of PMC contents.
  – Currently this can be done with PAPI_overflow().

• 2D profiling API
  – Allows statistical profiling on 2 items
  – IP of Miss vs. Addr of Miss
  – IP of Branch vs. Branch Target
  – Miss Addr vs. Latency
Sampling API

• How programmable should it be to be useful?
  – EventSet and Events to sample
    • What about non-Events?
  – Trigger event and Interval
  – Buffer and Length
  – Function to call when buffer is full
  – Address range(s)?
2D Profiling API

• Through Perfmon2, PAPI can now profile on other things other than IP. (currently only on IA64, soon on PPC64)
  – Data address and Latency
  – Branch arcs
  – Any combination of 2 is valid
• Traditional SVR4 profil()/sprofil() API is not adequate for this.
• Sparse or dense layout? Memory intensive.
Sampling API

• Currently there is no way to read 'special' PMD's directly:
  – IP/Data address (PPC64, IA64, SiCortex)
  – Branch information (PPC64, IA64)
  – Latency information (PPC64, IA64)
  – Trace buffers (Cell)
• PAPI only refers to events, not PMD's.
• These are not events, although could be treated as meta events.
Derived events

• Move computation of derived events to higher level layers of PAPI
  – Allows computation of derived metrics on multiplexed EventSets
  – Removes common substrate code
Compatible platform integration

• Sample PAPI library for all binary compatible platforms
  – No more separate builds for PIV, K8, x86, etc.
  – Just 32 and 64-bit builds
  – Same for PPC family
• Move a lot of shared code out of substrates
Other feature additions

• For Perfmon and Perfmon2 platforms
  – Profiling and sampling on attached processes/threads.
  – Support per-CPU and System-wide counting
• Support edge detect (cycle of event counting), thresholding and range restrictions where available
• Allow profiling and multiplexing simultaneously.
Links

- http://icl.cs.utk.edu/~mucci/monitor
- http://icl.cs.utk.edu/~mucci/papiex
- http://icl.cs.utk.edu/papi
- http://perfminer.pdc.kth.se

Questions: mucci at cs.utk.edu