Software-Defined Events though PAPI for in-depth Analysis of Application Performance

The Performance API (PAPI) provides tool designers and application engineers with a consistent interface and methodology for the use of low-level performance counter hardware found across the entire system (i.e., CPU, GPUs, on/off-chip memory, interconnects, I/O system, energy/power, etc.). PAPI enables users to see, in near real-time, the relationship between software performance and hardware events across the entire system.

PROJECT SCOPE

This project builds on the latest PAPI and extends it with support for software-defined events (SDE) that originate from layers of the software stack which are currently treated as black boxes (e.g., task-based runtime systems, communication libraries, math libraries, applications).

The objective is to enable monitoring of both types of performance events (hardware-related and software-related) in a uniform way, through the same interface offered by PAPI. Therefore, third-party tools and application developers have to handle only a single hook to PAPI to access all hardware performance counters in a system, including the new software-defined events.

CASE STUDY 1: Integration of PAPI SDE in NWChem

- NWChem (v. 6.8) iterative coupled cluster model with single and double excitations (CCSD)
- We registered four SDE counters via our FORTRAN'08 interface for each CCSD kernel (e.g., sde::NWChem::t28_flop_cnt counter for sde::NWChem::t28_flop_cnt counter).
- Figure 1 shows the relative workload of different subroutines

Table 1 provides a sample of PAPI-NWChem performance metrics that are exposed per CCSD sub-kernel

<table>
<thead>
<tr>
<th>PAPI-NWChem Performance Counter</th>
<th>Counter Description</th>
<th>Type</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>sde::NWChem::t28_max_chain_length</td>
<td>Maximum number of sequential DGEMM per chain in CCSD kernel (64-bit)</td>
<td>64-bit integer</td>
<td>instantaneous</td>
</tr>
<tr>
<td>sde::NWChem::t28_dgmem_cnt</td>
<td>Total number of DGEMMs in CCSD kernel (64-bit)</td>
<td>64-bit integer</td>
<td>delta</td>
</tr>
<tr>
<td>sde::NWChem::t28_flop_cnt</td>
<td>Total number of floating point operations in CCSD kernel (64-bit)</td>
<td>64-bit integer</td>
<td>delta</td>
</tr>
</tbody>
</table>

Table 1 shows the relative workload of different subroutines.

CASE STUDY 2: Integration of PAPI SDE in PaRSEC

- PaRSEC is a runtime system for task-based execution
- PaRSEC features SDE counters (and counter groups) for monitoring performance counters.

Software-Defined Events in PAPI

Goal: Offer support for software-defined events to extend PAPI’s role as a standardizing layer for monitoring performance counters.

Vision: Enable software layers to expose events that happened inside that software layer.

Benefit: Application developers will be able to better understand the interaction of the different components and layers of their applications as well as their interaction with external libraries and runtimes.

PAPI’s New SDE API

- API for reading SDEs is the same as for reading hardware events, i.e., PAPI_start()/PAPI_stop().
- New SDE APIs are only needed inside libraries for exporting SDEs from within those libraries.
- All API functions will be available in C and FORTRAN 2008.

New SDE API calls are only needed inside libraries for exporting SDEs from within those libraries. All API functions will be available in C and FORTRAN 2008.

Want to see a schematic of the SDE model?

PROJECTS AND THIRD-PARTY TOOLS APPLYING PAPI

- PaRSEC
- Caliper
- Kokkos
- TAU
- HPCToolkit
- ScaLapack
- Vampir
- Open/Speedshop
- SwPablo
- ompP

Figure 3 shows the evolution of various PaRSEC task queues during the execution of the application SPOTFI on a 32-core machine. The values shown are for counter groups which sum the values of all corresponding queues across the 32 threads of execution.