An Introduction to Hardware Performance Analysis and PAPI

Philip J. Mucci
mucci@cs.utk.edu

SIAM Parallel Processing 2004
San Francisco, CA
“The single most important impediment to good parallel performance is still poor single-node performance.”

- William Gropp

Argonne National Lab
Performance Evaluation

• Traditionally, timing and performance evaluation has been an art:
  – Resolution of the clock
  – Issues about cache effects
  – Different systems
  – Can be cumbersome and inefficient with traditional tools

• Situation has changed
  – Today’s processors have internal counters
Rising Complexity
New Performance Science

BIG SCIENCE CODE

IMPROVE PERFORMANCE
- ROSE
- Export/Inout
- Hand Optimization
- Automatic Tuning

UNDERSTAND & PREDICT PERFORMANCE
- Mellis m.
- Dinemar.
- Bayesian Modeling
- Performance Assertions

PERFORMANCE CHARACTERIZATION OF WHOLE APPLICATION
- gprof
- homecount
- mpiP
- parallel run
- wallclock times

IDENTIFY LOCATION OF BOTTLENECKS
- Dynacal
- SvPablo
- TAU
- Source code analysis & instrumentation

DETERMINE CAUSE OF BOTTLENECKS
- Vampir
- Spillog
- SIGMA
- TAU
- Paraver
Today most high performance processors include hardware performance counters.

Some are easy to access, others not available to users.

On most platforms the APIs, if they exist, are not appropriate for the end user or well documented.

Existing performance counter APIs

- Compaq Alpha EV 6 & 6/7
- SGI MIPS R10000
- IBM Power Series
- CRAY T3E, X1
- Sun Solaris
- Pentium and AMD
- IA-64
- HP-PA RISC
- Hitachi
- Fujitsu
- NEC
- **Performance Application Programming Interface**

- The purpose of PAPI is to implement a standardized portable and efficient API to access the hardware performance monitor counters found on most modern microprocessors.

- The goal of PAPI is to facilitate the optimization of parallel and serial code performance by encouraging the development of cross-platform optimization tools.
PAPI provides 3 interfaces to the underlying counter hardware:

1. The low level interface manages hardware events in user defined groups called EventSets, and provides access to advanced features.
2. The high level interface provides the ability to start, stop and read the counters for a specified list of events.
3. Graphical and end-user tools provide facilitate data collection and visualization.
PAPI Implementation

PAPI Low Level

PAPI Machine Dependent Substrate

Kernel Extension

Operating System

Hardware Performance Counters

Portable Layer

PAPI High Level

Tools

Machine Specific Layer
High Level Interface

- Meant for application programmers wanting coarse-grained measurements
- Not tuned for efficiency
- Calls the lower level API
- Only allows PAPI Presets
- Easier to use and less setup (less additional code) than low level
Low-level Interface

- Increased efficiency and functionality over the high level PAPI interface
- Obtain information about the executable, the hardware & the memory
- Thread-safe
- Fully programmable (native events)
- Multiplexing
- Callbacks on counter overflow
- Profiling
- 54 functions
Available Performance Data

- **Cycle count**
- **Instruction count**
  - All instructions
  - Floating point
  - Integer
  - Load/store
- **Branches**
  - Taken / not taken
  - Mispredictions
- **Pipeline stalls due to**
  - Memory subsystem
  - Resource conflicts
- **Cache**
  - I/D cache misses for different levels
  - Invalidations
- **TLB**
  - Misses
  - Invalidations
Aggregate performance measures over all tasks for a 0.1 simulation-second run. Collected with PAPI on an IBM SP (Nighthawk II / 375MHz).
Parallel Ocean Program Performance
Run: x1 Data Set, 2x2 Procs, 10 Steps

<table>
<thead>
<tr>
<th>Raw Data</th>
<th>Debug</th>
<th>Optimized</th>
<th>Metric</th>
<th>Debug</th>
<th>Optimized</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPI_LD_INS</td>
<td>1.21E+011</td>
<td>2.104E+10</td>
<td>% Ld Ins</td>
<td>36.86</td>
<td>33.63</td>
</tr>
<tr>
<td>PAPI_SR_INS</td>
<td>2.02E+010</td>
<td>7.783E+09</td>
<td>% Sr Ins</td>
<td>6.17</td>
<td>12.44</td>
</tr>
<tr>
<td>PAPI_BR_INS</td>
<td>8.64E+009</td>
<td>5.043E+09</td>
<td>% Br Ins</td>
<td>2.63</td>
<td>8.06</td>
</tr>
<tr>
<td>PAPI_FP_INS</td>
<td>2.21E+010</td>
<td>2.251E+10</td>
<td>% FP Ins</td>
<td>6.75</td>
<td>35.98</td>
</tr>
<tr>
<td>PAPI_FMA_INS</td>
<td>1.04E+010</td>
<td>1.007E+10</td>
<td>% FMA Ins</td>
<td>3.16</td>
<td>16.09</td>
</tr>
<tr>
<td>PAPI_FPU_FDIV</td>
<td>2.551E+08</td>
<td></td>
<td>% FP Divide</td>
<td>0.41</td>
<td></td>
</tr>
<tr>
<td>PAPI_FPU_FSQRT</td>
<td>1.317E+08</td>
<td></td>
<td>% FP SQRT</td>
<td>0.21</td>
<td></td>
</tr>
<tr>
<td>PAPI_TOT_INS</td>
<td>3.28E+011</td>
<td>6.257E+10</td>
<td>MFLIPS</td>
<td>12.19</td>
<td>72.31</td>
</tr>
<tr>
<td>PAPI_TOT_CYC</td>
<td>3.63E+011</td>
<td>6.226E+10</td>
<td>% MFLIPS Peak</td>
<td>3.05</td>
<td>18.08</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IPC</td>
<td>0.90</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mem Opts/FLIP</td>
<td>6.38</td>
<td>1.28</td>
</tr>
<tr>
<td>PAPI_L1_LDM</td>
<td>1.03E+009</td>
<td>1.011E+09</td>
<td>% L1 Ld HR</td>
<td>99.15</td>
<td>95.19</td>
</tr>
<tr>
<td>PAPI_L1_STM</td>
<td>3.54E+008</td>
<td>3.475E+08</td>
<td>% L1 Sr HR</td>
<td>98.25</td>
<td>95.54</td>
</tr>
<tr>
<td>PAPI_L2_LDM</td>
<td>6.94E+008</td>
<td>6.894E+08</td>
<td>% L2 Ld HR</td>
<td>99.43</td>
<td>96.72</td>
</tr>
<tr>
<td>PAPI_FPU_IDL</td>
<td>1.66E+011</td>
<td>1.411E+10</td>
<td>% FPU Idle Cyc</td>
<td>45.77</td>
<td>22.66</td>
</tr>
<tr>
<td>PAPI_LSU_IDL</td>
<td>4.06E+010</td>
<td>1.483E+10</td>
<td>% LSU Idle Cyc</td>
<td>11.17</td>
<td>23.82</td>
</tr>
<tr>
<td>PAPI_MEM_RCY</td>
<td>1.03E+011</td>
<td>1.368E+10</td>
<td>% Ld Stall Cyc</td>
<td>28.28</td>
<td>21.97</td>
</tr>
<tr>
<td>PAPI_MEM_SCY</td>
<td>1.26E+011</td>
<td>2.413E+10</td>
<td>% Sr Stall Cyc</td>
<td>34.59</td>
<td>38.76</td>
</tr>
<tr>
<td>PAPI_STL_CCY</td>
<td>2.01E+011</td>
<td>3.367E+10</td>
<td>% No Ins. Cyc</td>
<td>55.25</td>
<td>54.08</td>
</tr>
</tbody>
</table>
Supported Architectures

- AMD Athlon and Opteron
- Cray T3E and X1
- HP Alpha (caveats)
- IBM POWER3, POWER4
- Intel Pentium Pro, II, III + 4, Itanium 1 + 2
- MIPS R10K, R12K, R14K
- Sun UltraSparc I, II, III
Standard PAPI Features

- Standardized Access to Performance Counters
- Standardized Performance Metrics
- Easy Access to Platform-Specific Metrics
- Multiplexed Event Measurement
- Dispatch on Overflow
- Event Profiling
- Bindings for C, Fortran, Matlab, and Java
New in PAPI 3.0

- Lower Measurement Overheads
- Overflow and Profiling on Multiple Simultaneous Events
- Easy Access to Platform-Specific Metrics
- High level API is now thread safe
- Internal timer/signal/thread abstractions
New in PAPI 3.0

- Lower Measurement Overheads
- Overflow and Profiling on Multiple Simultaneous Events
- Complete and Easy Access to All Platform-Specific Metrics
- High level API is now thread safe
- Internal timer/signal/thread abstractions
Multiple Counter Profiling

Program Text Addresses

Event Count

L1DCM
L2DCM
DTLB
New in PAPI 3.0

- Lower Measurement Overheads
- Overflow and Profiling on Multiple Simultaneous Events
- Complete and Easy Access to All Platform-Specific Metrics
- High level API is now thread safe
- Internal timer/signal/thread abstractions
Which Tool?
You must have the right tool for the job.

What are your needs? Things to consider:

- **User Interface**
  - Complex Suite
  - Quick and Dirty

- **Data Collection Mechanism**
  - Aggregate
  - Trace based
  - Statistical
The Right Performance Tool 2

• **Instrumentation Mechanism**
  – Source
  – Binary (DPCL/DynInst)
  – Library interposition

• **Data Management**
  – Performance Database
  – User (Flat file)

• **Data Visualization**
  – Run Time
  – Post Mortem
  – Serial/Parallel Display
  – ASCII
Some Tools that use PAPI

- TAU (Sameer Shende, U Oregon)
  http://www.cs.uoregon.edu/research/paracom/tau/

- SvPablo (Celso Mendes, UIUC)
  http://www-pablo.cs.uiuc.edu/Project/SVPablo/

- HPCToolkit (J. Mellor-Crummey, Rice U)
  http://hipersoft.cs.rice.edu/hpctoolkit/

- psrun (Rick Kufrin, NCSA, UIUC)
  http://www.ncsa.uiuc.edu/~rkufrin/perfsuite/psrun/

- Titanium (Dan Bonachea, UC Berkeley)
  http://www.cs.berkeley.edu/Research/Projects/titanium/

- SCALEA (Thomas Fahringer, U Innsbruck)
  http://www.par.univie.ac.at/project/calea/
Some Tools that use PAPI 2

- **KOJAK (Bernd Mohr, FZ Juelich; U Tenn)**
  [http://www.fz-juelich.de/zam/kojak](http://www.fz-juelich.de/zam/kojak)

- **Cone (Felix Wolf, U Tenn)**

- **HPMtoolkit (Luiz Derose, IBM)**

- **CUBE (Felix Wolf, U Tenn)**
Some Tools that use PAPI 3

- **ParaVer (J. Labarta, CEPBA)**
  
  http://www.cepba.upc.es/paraver

- **VAMPIR (Pallas)**
  
  http://www.pallas.com/e/products/vampir/index.htm

- **DynaProf (P. Mucci, U Tenn)**
  
  http://www.cs.utk.edu/~mucci/dynaprod
This talk:


PAPI Homepage:

- http://icl.cs.utk.edu/papi