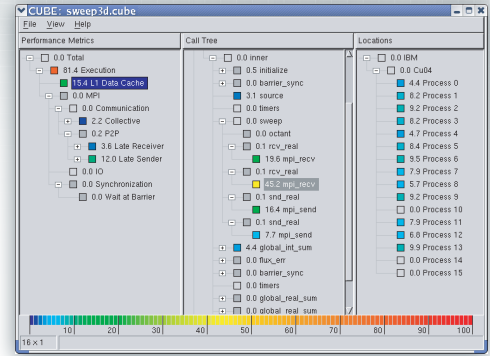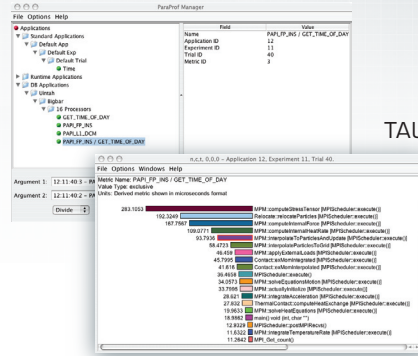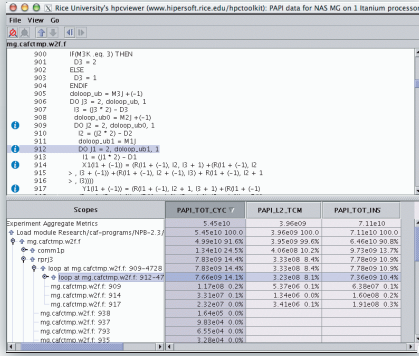# PAPI

R&D100 2001 WINNER

PERFORMANCE APPLICATION PROGRAMMING INTERFACE


HPCVIEW

TAU

KOJAK

PERFORMANCE ANALYSIS TOOLS

STANDARD PAPI FEATURES

Standardized Performance Metrics
Easy Access to Platform-Specific Metrics
Multiplexed Event Measurement
Dispatch on Overflow
Overflow & Profiling on Multiple
  Simultaneous Events
Bindings for C, Fortran, Matlab and Java

INTRODUCING PAPI 4

Multiple Simultaneous Substrates
Enhanced Installation Features
Better Software Componentization
Easier 3rd Party Substrate Development

PAPI

VENDOR HW/ KERNEL EXTENSION

OS/KERNEL

HARDWARE SUBSTRATES
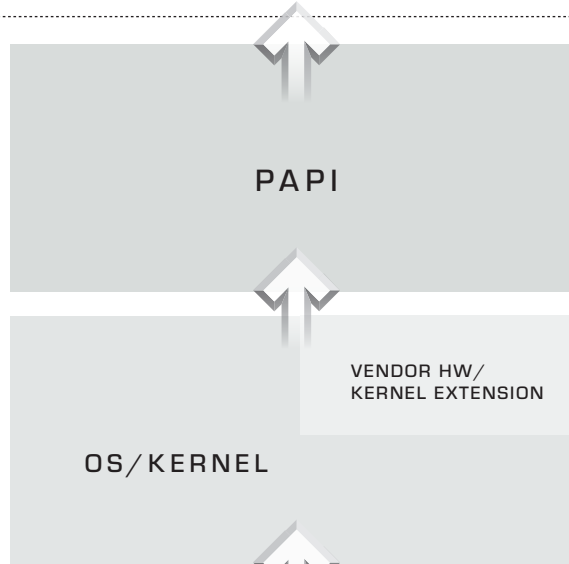
SUPPORTED ARCHITECTURES

AMD Athlon, Athlon64, Opteron
Cray X1, XD1,XT3
IBM POWER3, 4, 5, BlueGene
Intel Pentium III, 4, EMT64, D
Intel Itanium 1, 2
MIPS R10K, R12K, R14K
Sun UltraSparc I, II, III
SGI Altix
See web site for OS and installation details

SOME TOOLS THAT USE PAPI

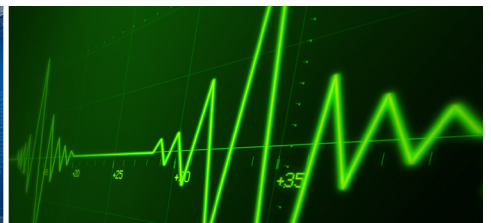|		
ADAPTOR	PerfSuite	TAU
HPCToolkit	SCALEA	Titanium
KOJAK	SvPablo	VProf
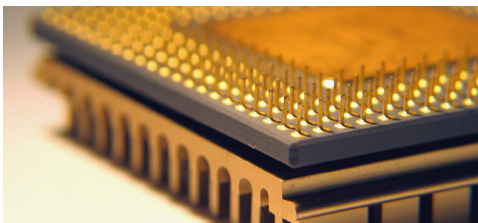
CURRENT CPU

NETWORK COMMUNICATIONS

TEMPERATURE/SYSTEMS DIAGNOSTICS

PAPI

R&D100 2001 WINNER

PERFORMANCE APPLICATION PROGRAMMING INTERFACE

The purpose of the PAPI (Performance Application Programming Interface) project is to define a standardized, easy to use interface that provides access to the hardware performance counters on most major processor platforms, thereby providing application developers the information they need to tune their software on different platforms. The goal is to make it easy for users to gain access to the counters to aid in performance analysis, modeling, and tuning.

The PAPI library provides two interfaces to the underlying hardware: a high level interface for the acquisition of simple measurements, and a fully programmable, low level interface directed towards users with more sophisticated needs. The high level interface simply provides the capability to start, stop, and read the counters for a specified list of events. The target audience for the high level interface is application engineers and benchmarking teams looking to quickly and simply acquire some rudimentary performance measurements. The tool developer will likely find the high level interface too restrictive. The low level interface provides the more sophisticated functionality of user callbacks on counter overflow and SV4 compatible profiling based on any counter event, regardless of whether or not the operating system supports it. These features provide the necessary basis for any source-level performance analysis software. Thus, for any architecture with even the most basic access to hardware performance counters, PAPI provides the foundation for truly portable source level performance analysis tools based on real processor statistics.

PAPI aims to provide the tool designer and application engineer with a consistent interface and methodology for use of the performance counter hardware found in most major microprocessor lines. The main motivation for this interface is the increasing divergence of application performance from near peak performance of most machines in the HPC marketplace. This performance gap is largely due to differences in memory and communication bandwidth at different levels of the memory hierarchy. Users need a compact set of robust and useful tools to quickly diagnose and analyze processor specific performance metrics. To address this problem, PAPI focuses on developing a reusable, portable, and functionality oriented infrastructure for performance tool design and advanced program analysis.

Although the PAPI library can be used directly by application developers, most application scientists will prefer to use one of the many performance analysis tools built on top of it. Examples of these include SvPablo from the University of North Carolina, TAU from the University of Oregon, HPCToolkit from Rice University, PerfSuite from the National Center for Supercomputing Applications, and KOJAK from Forschungszentrum Juelich and University of Tennessee.

PAPI and the data that it provides have proven of great usefulness in the analysis of the nation's most important scientific applications. For example, Sandia National Laboratory uses PAPI to tune quantum chemistry simulations. Both Lawrence Livermore and Los Alamos National Laboratories use PAPI to tune and model codes used in monitoring the safety of the nuclear stockpile. Researchers associated with the Department of Energy (DOE) Performance Evaluation Research Center use PAPI to analyze many of the codes in the DOE SciDAC program, codes that span domains from extraterrestrial gas dynamics to high resolution global weather simulations. The Department of Defense (DoD) High Performance Computing Modernization Program has used PAPI for profiling of application benchmark codes. PAPI is installed and in use on many of the National Science Foundation (NSF) TeraGrid machines. PAPI's usefulness is not limited to just the United States, as it is currently being used for application analysis all over world from Denmark to Vietnam to Brazil.

New to PAPI 4 is a restructured architecture that promotes independent development of the hardware dependent substrate layers. This architecture also makes it possible to simultaneously access multiple substrates to support the measurement of counters on a variety of hardware components such as thermal sensors, network interfaces, memory components and more.

PAPI 4 is now available as a technology prerelease. It and other available releases support a wide variety of architectures including: IBM POWER 3, 4 and 5, IBM Blue Gene, Cray T3E, X1,XD1 and XT3, Intel Pentium and Itanium, Sun UltraSparc, MIPS R10K, R12K, R14K, AMD Athlon, Athlon-64 and Opteron, and HP Alpha. For further details and implementation specifics visit the PAPI website at http://icl.cs.utk.edu/papi/.