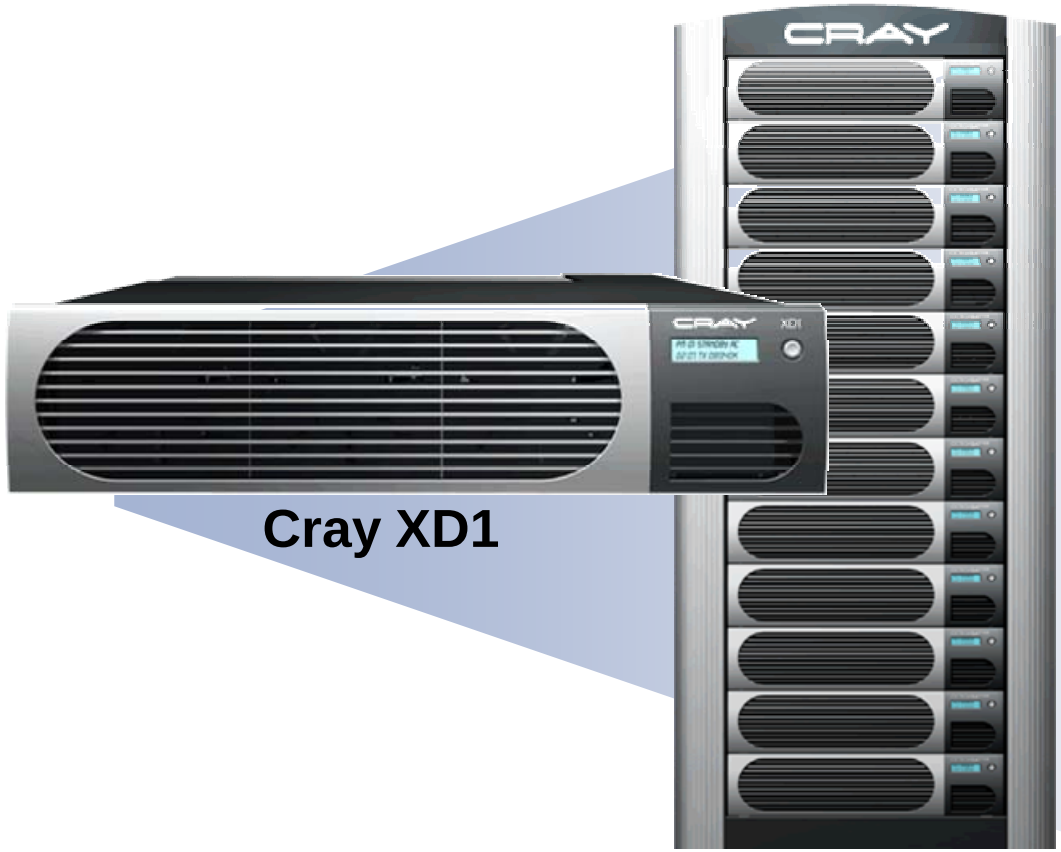
The Cray XD1

Technical Overview

Amar Shan, Senior Product Marketing Manager

The Cray XD1



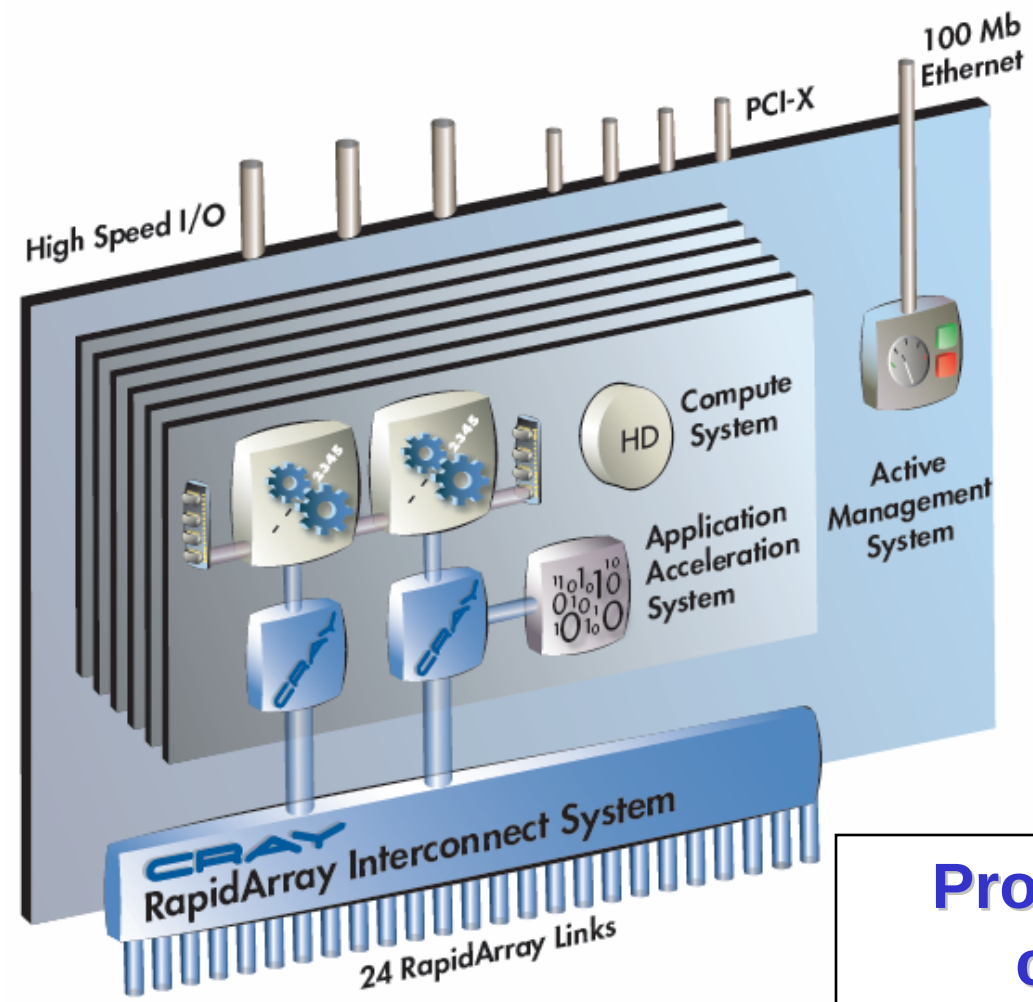
Cray XD1

- Built for price performance
- 30 times interconnect performance
- 2 times the density
- High availability
- Single system command & control

Purpose-built and optimized for high performance workloads

CUG 2004

Cray XD1 System Architecture



Compute

- 12 AMD Opteron 32/64 bit, x86 processors
- High Performance Linux

RapidArray Interconnect

- 12 communications processors
- 1 Tb/s switch fabric

Active Management

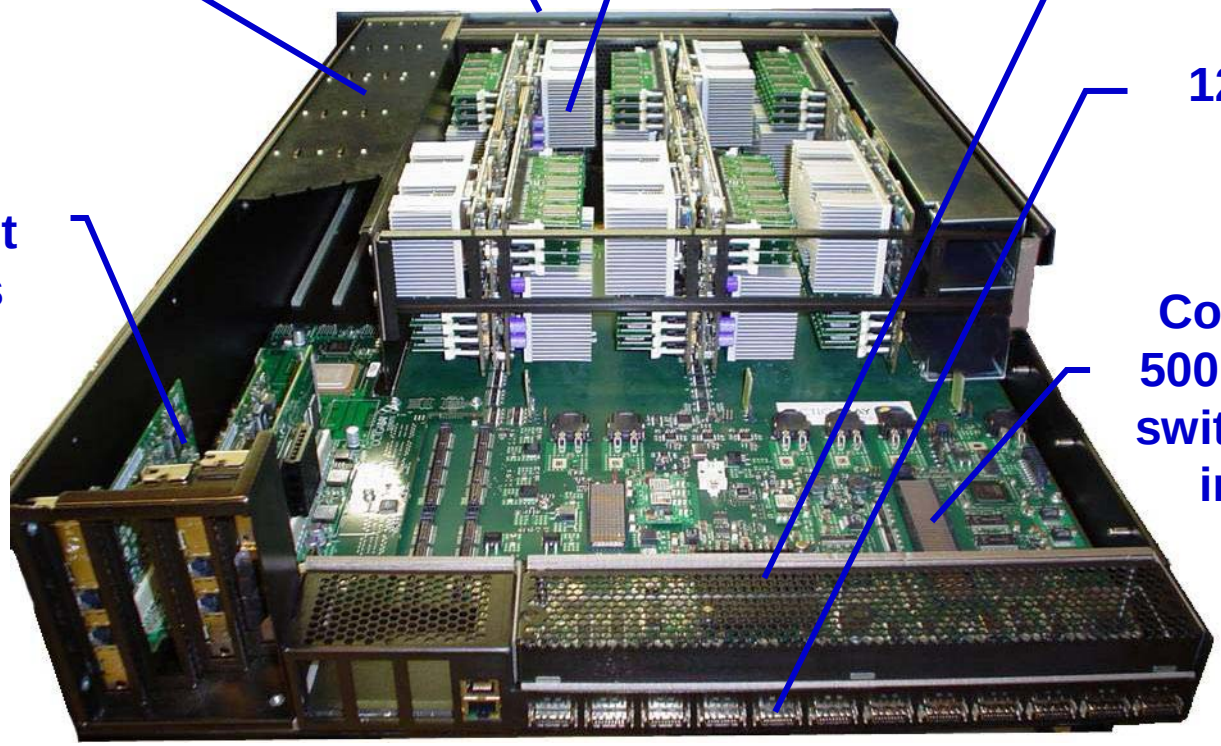
- Dedicated processor

Application Acceleration

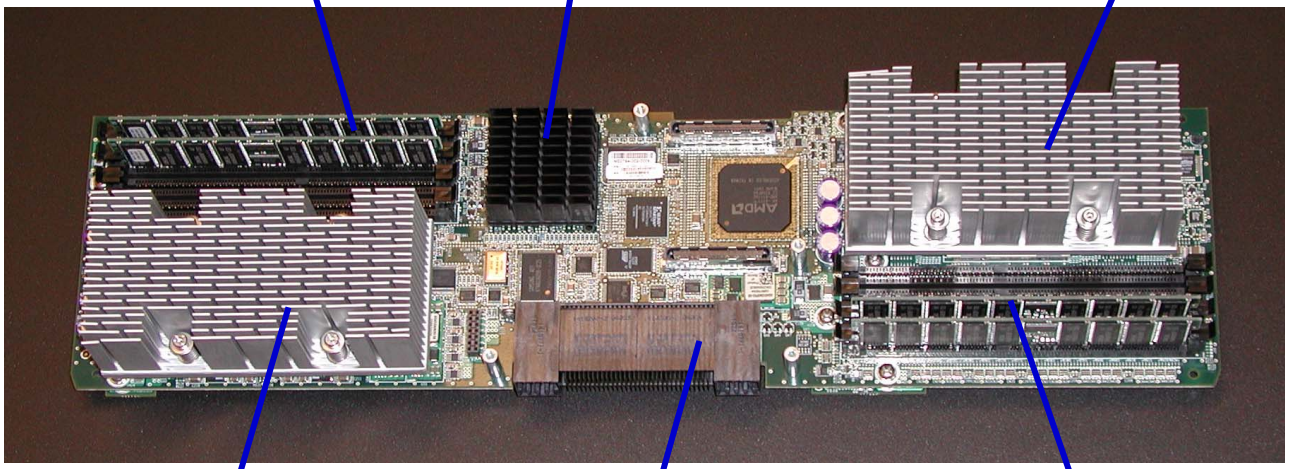
- 6 co-processors

Processors directly connected via integrated switch fabric

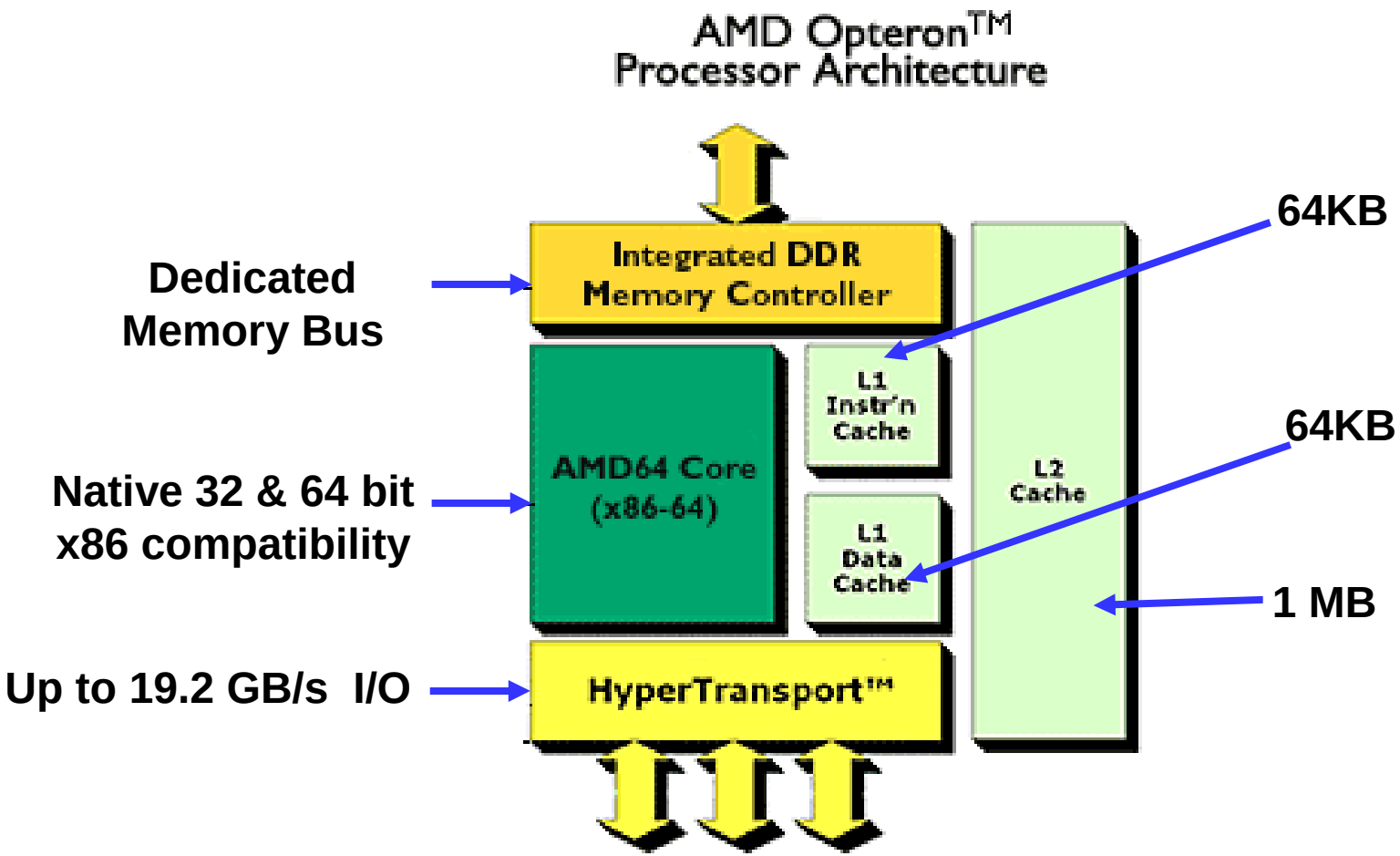
Under the Covers



Compute Blade



The AMD Opteron Processor



Cray Innovations

Cray XD1

Balanced Interconnect

Active Management

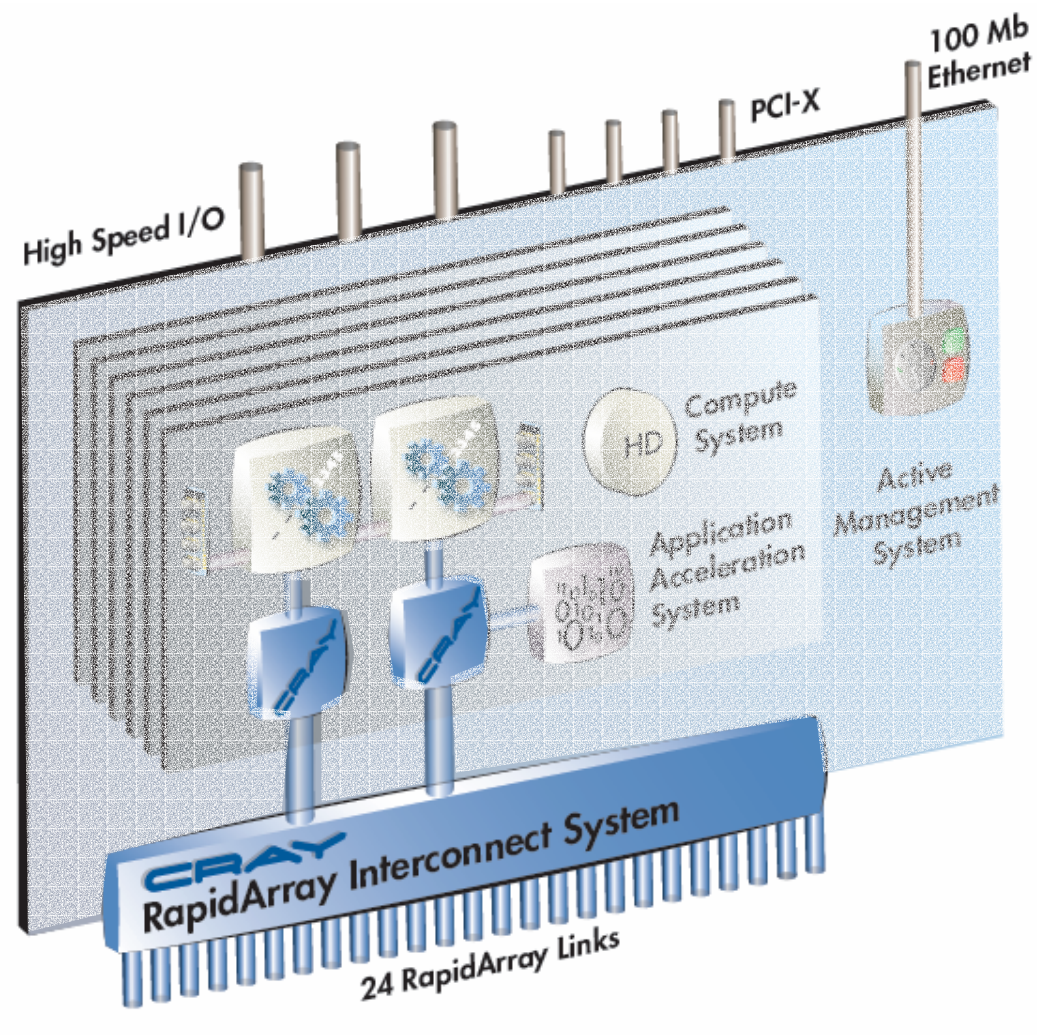
Application Acceleration

Performance and Usability

CUG 2004

Interconnect

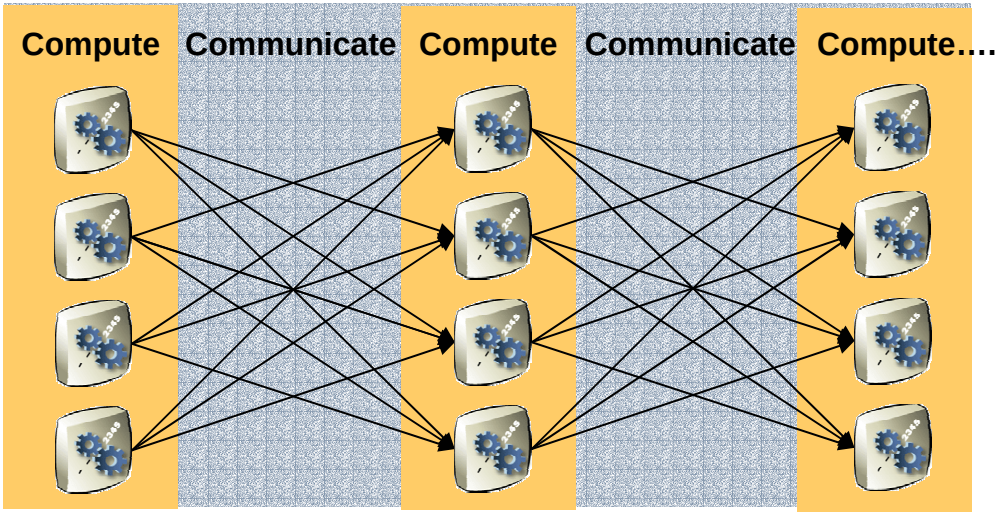
Cray XD1 Interconnect System



RapidArray

- **Interconnect processors**
- **Switch fabric**
- **Communications software**

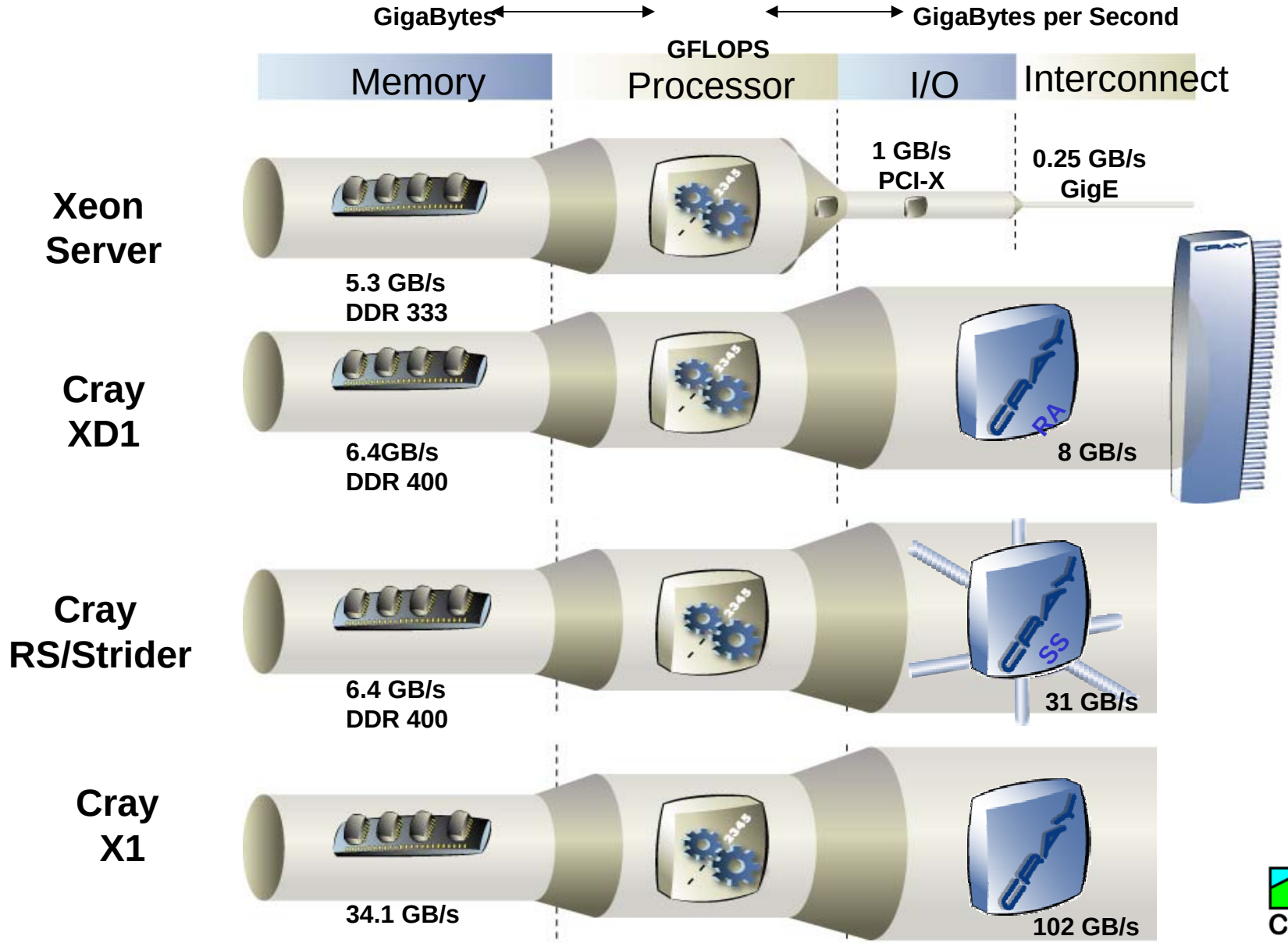
Typical HPC Application



- **HPC applications exhibit intense compute/ communicate cycles**
- **20% - 60% of time, CPUs sit idle, stalled by communications**
- **Application performance is very sensitive to latency and bandwidth**

Interconnect Drives System Performance

Removing the Communications Bottleneck



CUG 2004

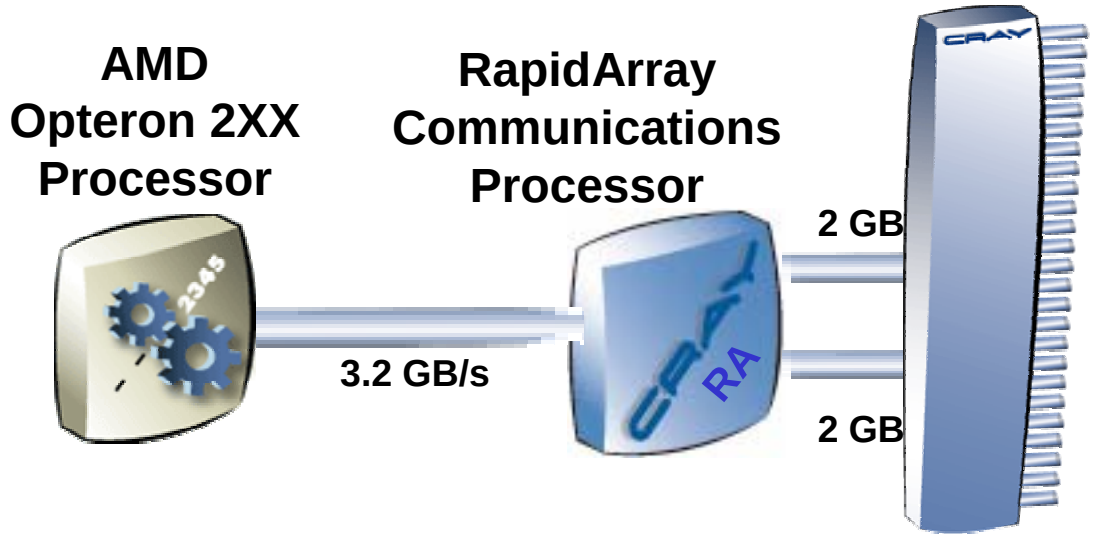
HPC Communications Optimizations

Cray Communications Libraries

- MPI 1.2 library
- TCP/IP
- PVM
- Shmem
- Global Arrays
- System-wide process & time synchronization

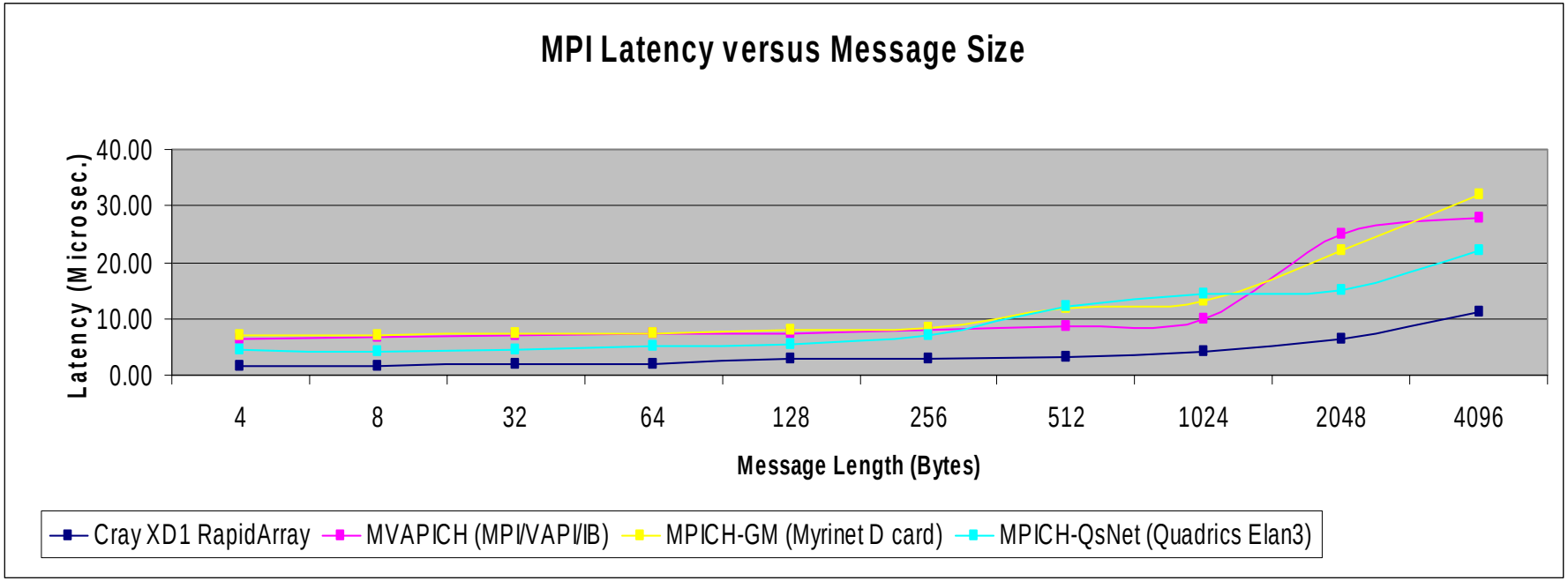
RapidArray Communications Processor

- HT/RA tunnelling with bonding
- Routing with route redundancy
- Reliable transport
- Short message latency optimization
- DMA operations
- System-wide clock synchronization



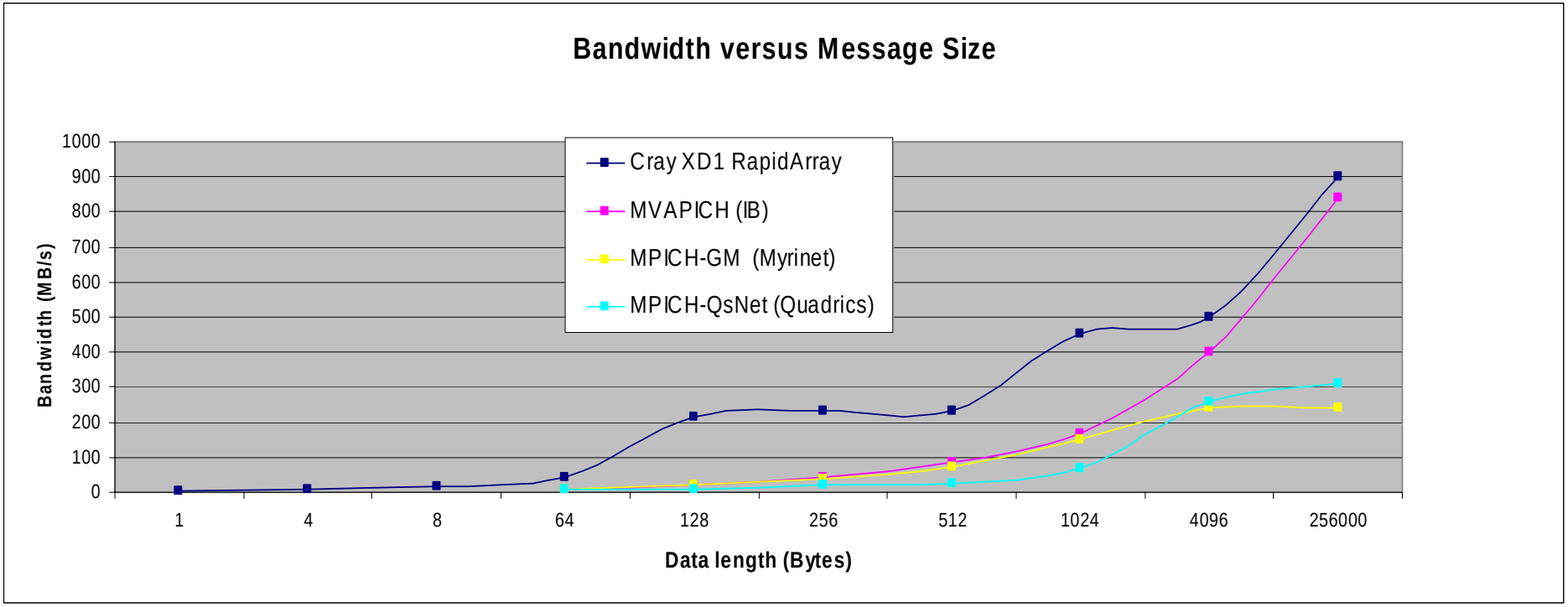
Direct Connected Processor Architecture

Interconnect Benchmarks (MPI Latency)



4 times lower latency than Myrinet (small message).
The Cray XD1 has sent 1 KB before others have sent a single byte.

Interconnect Benchmarks (MPI Throughput)



The Cray XD1 is 5X throughput of Quadrics at 1KB messages

Processor Interaction

The Case of the Missing Supercomputer Performance:
Achieving Optimal Performance on the 8,192 Processors of ASCI Q

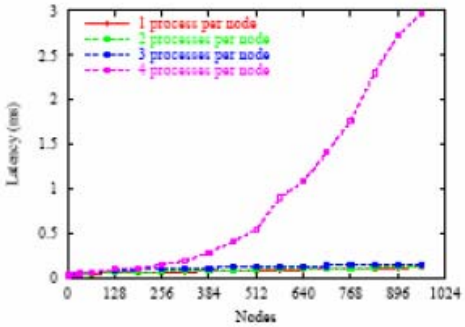
Fabrizio Petrini Darren J. Kerbyson Scott Pakin

Performance and Architecture Laboratory (PAL)
Computer and Computational Sciences (CCS) Division
Los Alamos National Laboratory
Los Alamos, New Mexico, USA

{fabrizio,djk,pakin}@lanl.gov

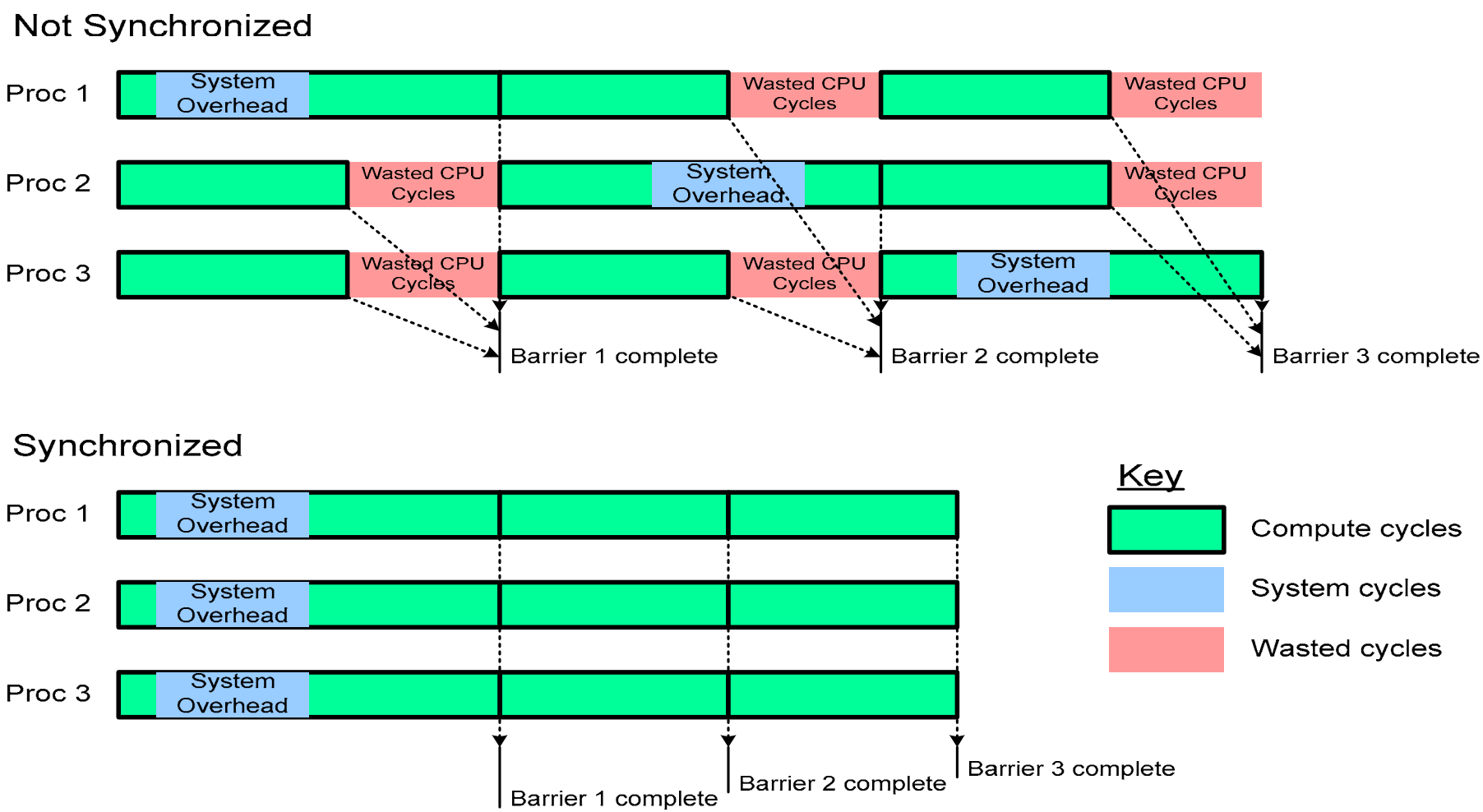
Abstract

In this paper we describe how we improved the effective performance of ASCI Q, the world's second-fastest supercomputer, to meet our expectations. Using an arsenal of performance-analysis techniques including analytical models, custom microbenchmarks, full applications, and simulators, we succeeded in observing a serious—but previously undetected—performance problem. We identified the source of the problem, eliminated the problem, and "closed the loop" by demonstrating up to a factor of 2 improvement in application performance. We present our methodology and provide insight into performance analysis that is immediately applicable to other large-scale supercomputers.



How to Double Application Performance

Synchronized Linux Scheduler



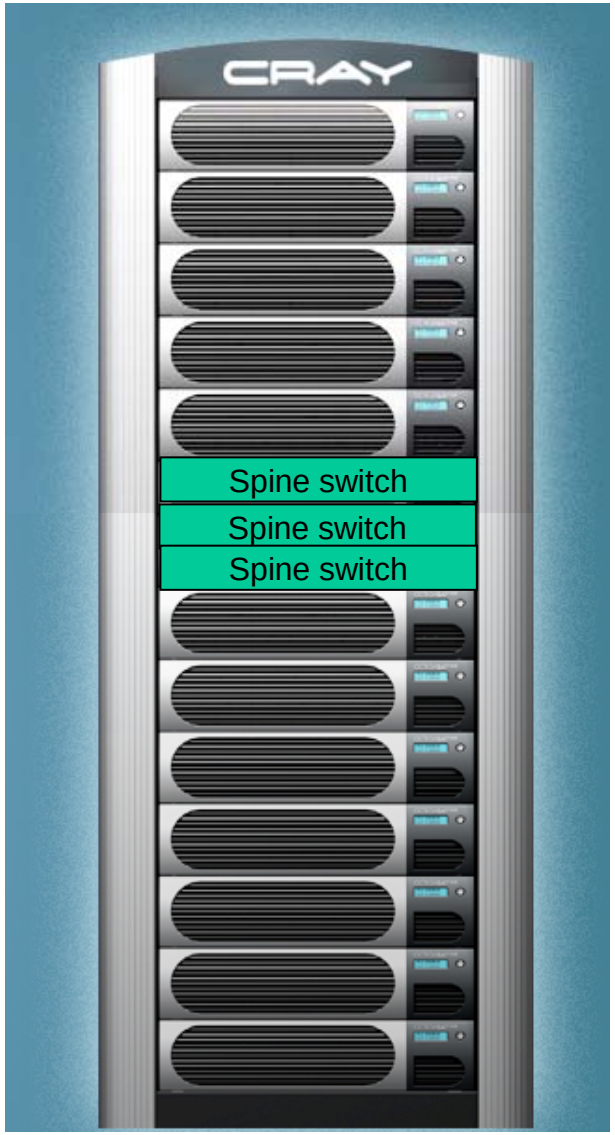
Direct Connect Topology

1 Cray XD1 Chassis
12 AMD Opteron Processors
53 GFLOPS
8 GB/s between SMPs
1.6 µsec interconnect
Integrated switching

3 Cray XD1 Chassis
36 AMD Opteron Processors
158 GFLOPS
8 GB/s between SMPs
1.8 µsec interconnect
Integrated switching

25 Cray XD1 Chassis, two racks
300 AMD Opteron Processors
1.3 TFLOPS
2 - 8 GB/s between SMPs
1.8 µsec interconnect
Integrated switching

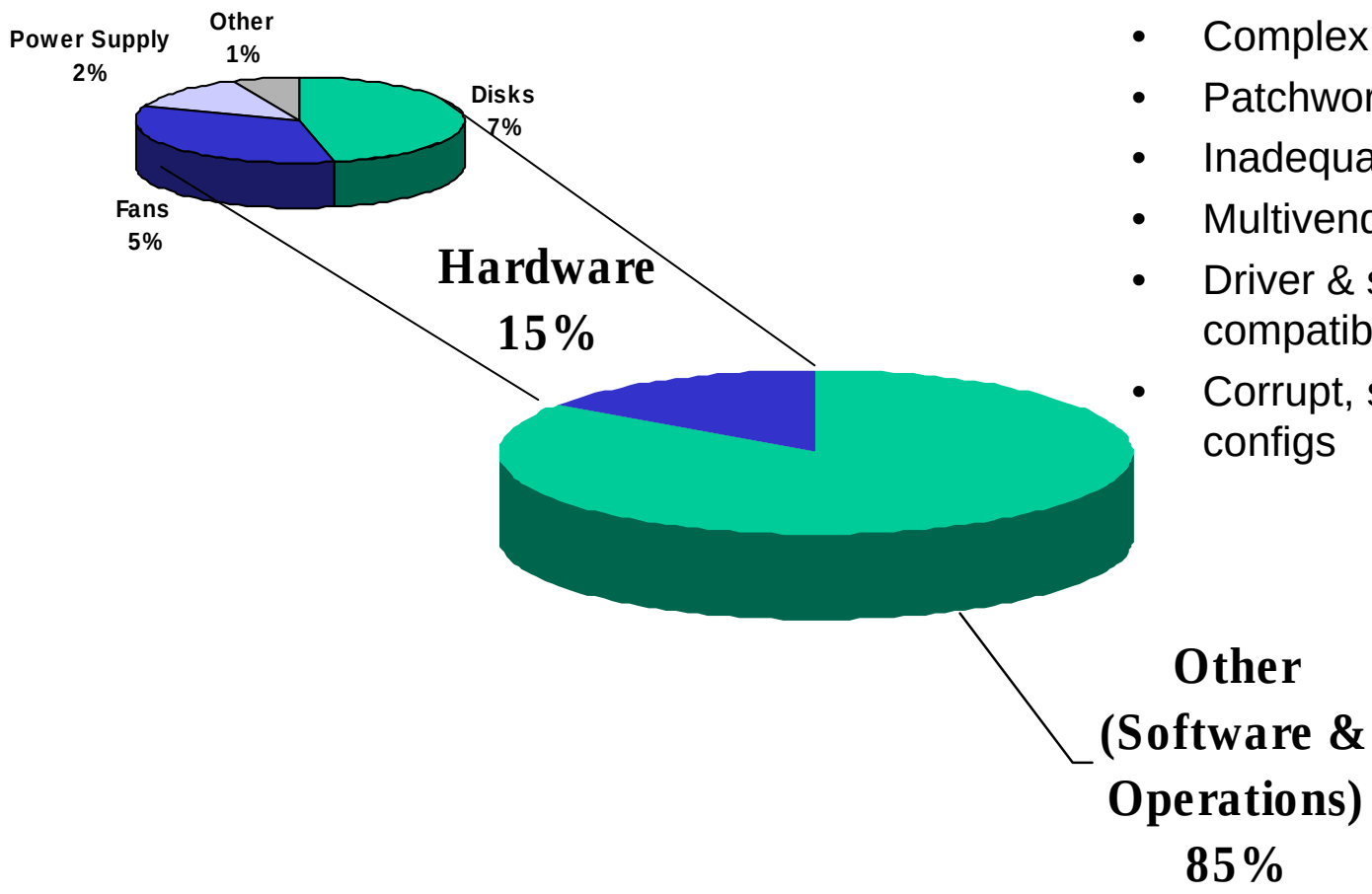
Fat Tree Topology



- **12 Cray XD1 chassis**
- **144 AMD Opteron Processors**
- **633 GFLOPS**
- **4/8 GB/s between SMPs**
- **1.9 μsec interconnect**
- **Fat tree switching, integrated first & third order**
- **6/12 RapidArray spine switches (24-ports)**

Management

Causes of System Outages

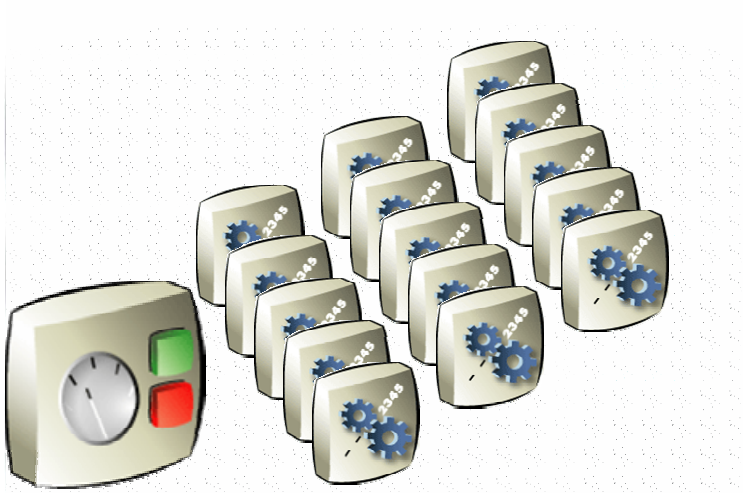


- Most problems occur during:
  - **Upgrades,**
  - **Problem diagnosis,**
  - **Configuration**
- Complex systems
- Patchwork software
- Inadequate tools
- Multivendor h/w, s/w
- Driver & software compatibility
- Corrupt, stale software & configs

Source: UC Berkeley

Active Manager System

CLI and Web Access



Active Management Software

Usability

- Single System Command and Control

Resiliency

- Dedicated management processors, real-time OS and communications fabric.
- Proactive background diagnostics with self-healing.

Automated management for exceptional reliability, availability, serviceability

FCAPS

	FUNCTION	Active Manager Features
Fault	• Maintain health of the system • Monitor, report on and automatically heal or • Allow operator intervention to resolve problems	• System monitoring • Hardware management • Alarm management
Configuration	• Define, monitor and change operational parameters of the system • Provide administrators with a real-time view of total system configuration and • Automate configuration tasks to minimize effort and inconsistencies	• Commission, upgrade, and expand system • Software management • Network configuration management • Manage users • Partition Management
Accounting	• Monitor system resources and usage • Support cost accounting or bill back	• Quotas • Usage tracking – per job, user, department • Reporting • Resource and Queue Management
Performance	Allow administrators to fine tune system operation to improve job and system performance	• Job management • File system management • System performance analysis
Security	Control access to • the system • system resources and to • specific functions within the system	• User privilege management • Data backups

Active Manager GUI: SysAdmin Portal

GUI provides quick access to status info and system functions

CUG 2004

Active Manager Command Line Interface

```
Adam Lorant@Adam ~
$ lspart
Name                  Login JobExec  State  SMPs  Load  JobsR  JobsQ  AlarmCnt
-----------------------------------------------------------------------------
Ab_Initio_Project     No    Yes      open   10    59.46 13     20
Chemistry             Yes   Yes      open   33    50.26 17     0      1
Diagnostic-Staging    No    No       N/A    3     0.84  0      0
File-Server           No    No       N/A    2     0.91  0      0
Front_End_Partition   Yes   No       open   4     1.1   0      0
Physics               No    Yes      open   8     58.2  5      44

Adam Lorant@Adam ~
$ qstat
Job Id      Job Name        Owner          CPU      State    Partition
-----------------------------------------------------------------------
157         amar_job.sh     amar           1057     running  Chemistry
63          paul_job.sh     paul           0        queued   Physics
143         pat_job.sh      pat            1802     running  Ab_Initio_Project
94          adam_job.sh     adam           0        queued   Physics
67          adam_job.sh     adam           0        queued   Physics
136         paul_job.sh     paul           1938     running  Chemistry
125         pat_job.sh      pat            2405     running  Ab_Initio_Project
86          steve_job.sh    steve          19504    running  Chemistry
167         adam_job.sh     adam           331      running  Ab_Initio_Project
145         paul_job.sh     paul           0        queued   Physics
96          john_job.sh     john           0        queued   Physics
95          steve_job.sh    steve          0        queued   Ab_Initio_Project
69          john_job.sh     john           0        queued   Physics
126         ron_job.sh      ron            0        queued   Physics
43          pat_job.sh      pat            0        queued   Physics
76          adam_job.sh     adam           0        queued   Ab_Initio_Project
144         ron_job.sh      ron            0        queued   Physics
79          pat_job.sh      pat            0        queued   Ab_Initio_Project
152         pat_job.sh      pat            0        queued   Physics
155         auji_job.sh     auji           0        queued   Ab_Initio_Project
44          ron_job.sh      ron            0        queued   Physics
117         paul_job.sh     paul           2498     running  Chemistry
90          paul_job.sh     paul           0        queued   Physics
132         steve_job.sh    steve          0        queued   Ab_Initio_P
88          pat_job.sh      pat            0        queued   Physics
149         adam_job.sh     adam           1365     running  Ab_Initio_P
58          adam_job.sh     adam           0        queued   Physics
36          paul_job.sh     paul           0        queued   Ab_Initio_P
59          steve_job.sh    steve          0        queued   Ab_Initio_P
16          pat_job.sh      pat            0        queued   Ab_Initio_P
15          john_job.sh     john           12375    running  Physics
62          ron_job.sh      ron            2797     running  Chemistry
158         adam_job.sh     adam           0        queued   Ab_Initio_Project
161         pat_job.sh      pat            483      running  Ab_Initio_Project
172         paul_job.sh     paul           205      running  Ab_Initio_Project
114         john_job.sh     john           0        queued   Physics
53          ron_job.sh      ron            0        queued   Physics
```

Administrators can access Active Manager software and Linux through the CLI

CUG 2004

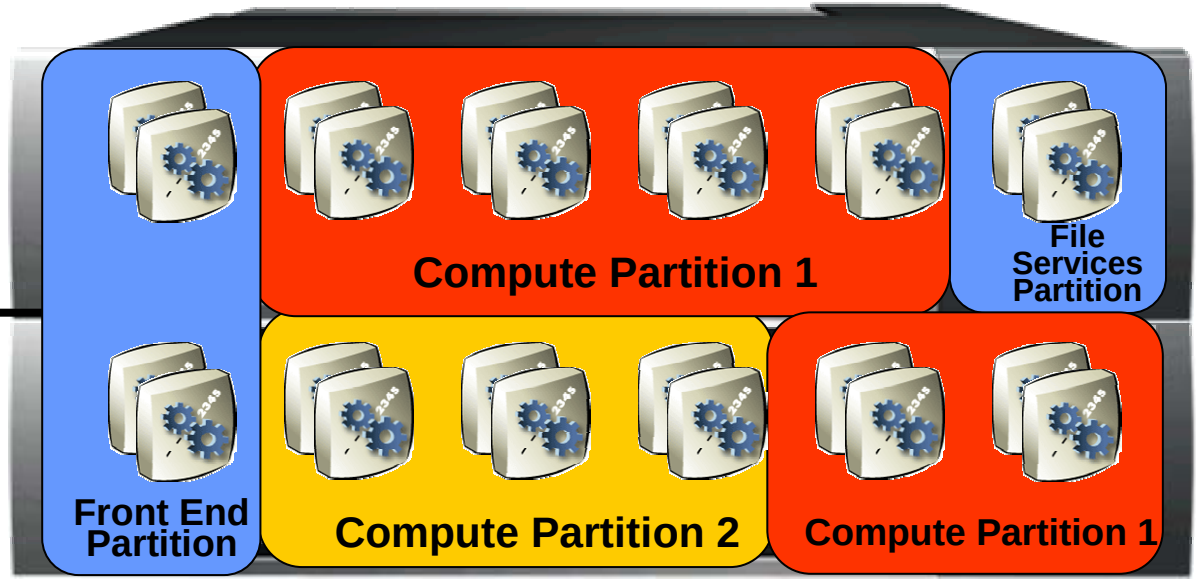
XML Integration Point

```
Adam Lorant@Adam ~
$ lspart --xml
<lspart>
  <partition name="Ab_Initio_Project" login="No" job_execution="Yes" state="open
" smp_count="10" load="61.53" running_jobs="11" queued_jobs="19" alarm_count=" "
/>
  <partition name="Chemistry" login="Yes" job_execution="Yes" state="open" smp_c
ount="33" load="64.19" running_jobs="21" queued_jobs="1" alarm_count=" "/>
  <partition name="Diagnostic-Staging" login="No" job_execution="No" state="N/A"
 smp_count="3" load="1.12" running_jobs="0" queued_jobs="0" alarm_count=" "/>
  <partition name="File-Server" login="No" job_execution="No" state="N/A" smp_co
unt="2" load="0.77" running_jobs="0" queued_jobs="0" alarm_count=" "/>
  <partition name="Front_End_Partition" login="Yes" job_execution="No" state="op
en" smp_count="4" load="1.09" running_jobs="0" queued_jobs="0" alarm_count=" "/>

  <partition name="Physics" login="No" job_execution="Yes" state="open" smp_coun
t="8" load="60" running_jobs="5" queued_jobs="43" alarm_count=" "/>
</lspart>

Adam Lorant@Adam ~
$
```

Simplifies data transfer

System Partitions

Users & Administrators



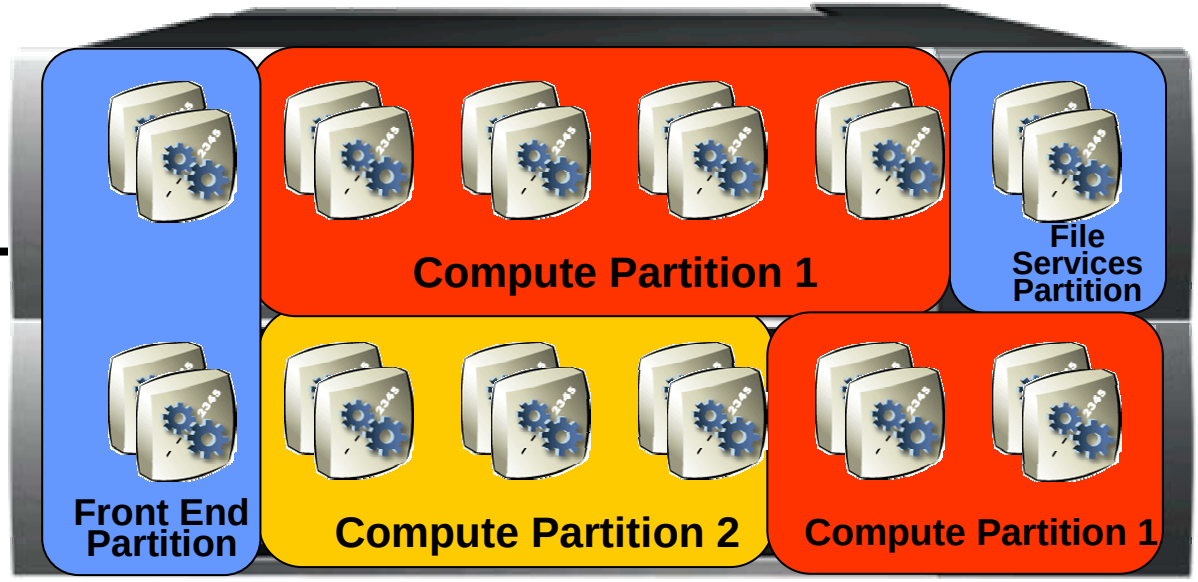
- **Front End Partition**
- **Compute Partition**
- **Service Partition**
  - **File Services**
  - **Database**
  - **DNS**

Manage multiple processors and copies of Linux as single, unified system

CUG 2004

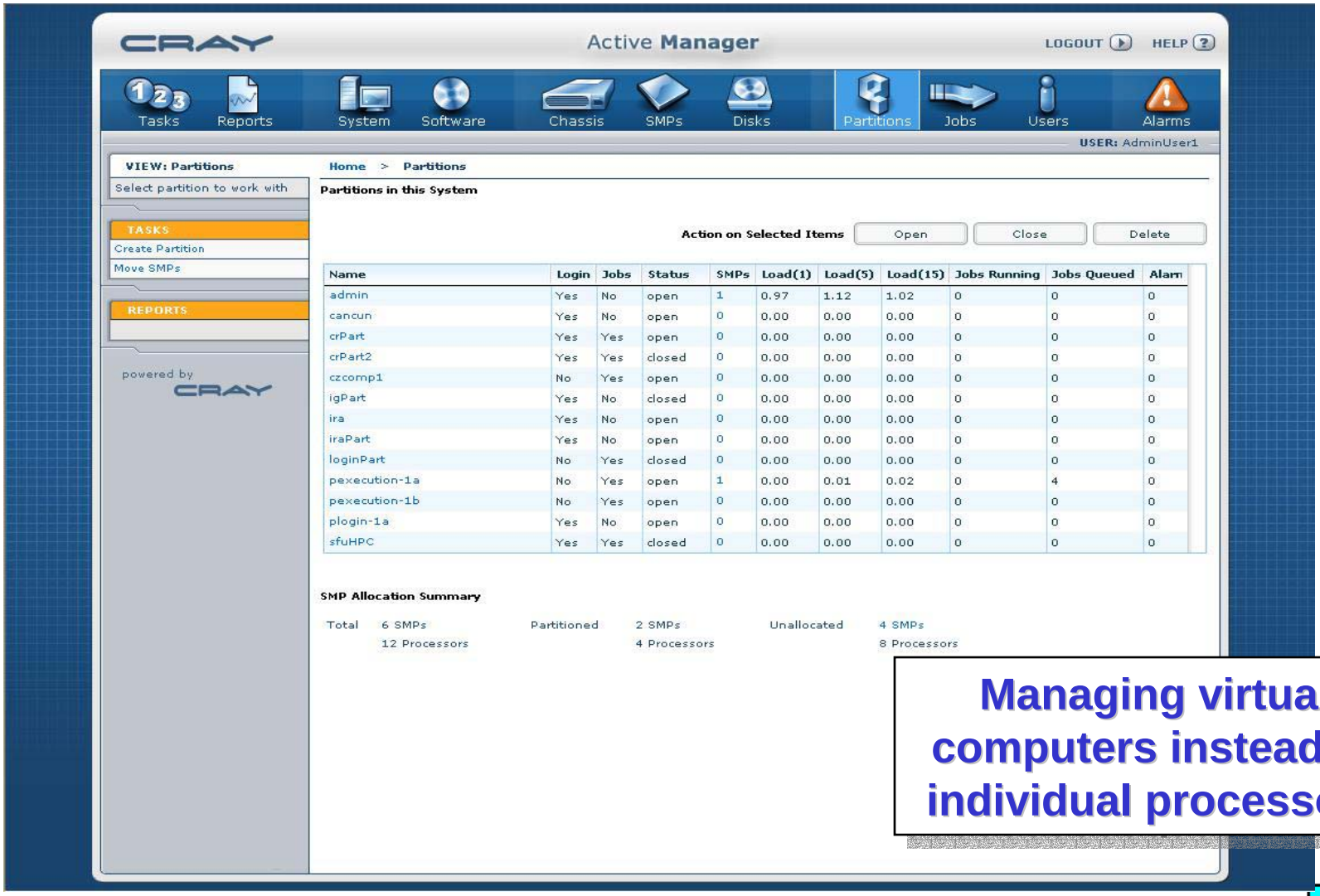
Single System Command and Control

Users & Administrators



- **Partition management**
- **Linux configuration**
- **Hardware monitoring**
- **Software upgrades**
- **File system management**
- **Data backups**
- **Network configuration**
- **Accounting & user management**
- **Security**
- **Performance analysis**
- **Resource & queue management**

Active Manager System/Partition View

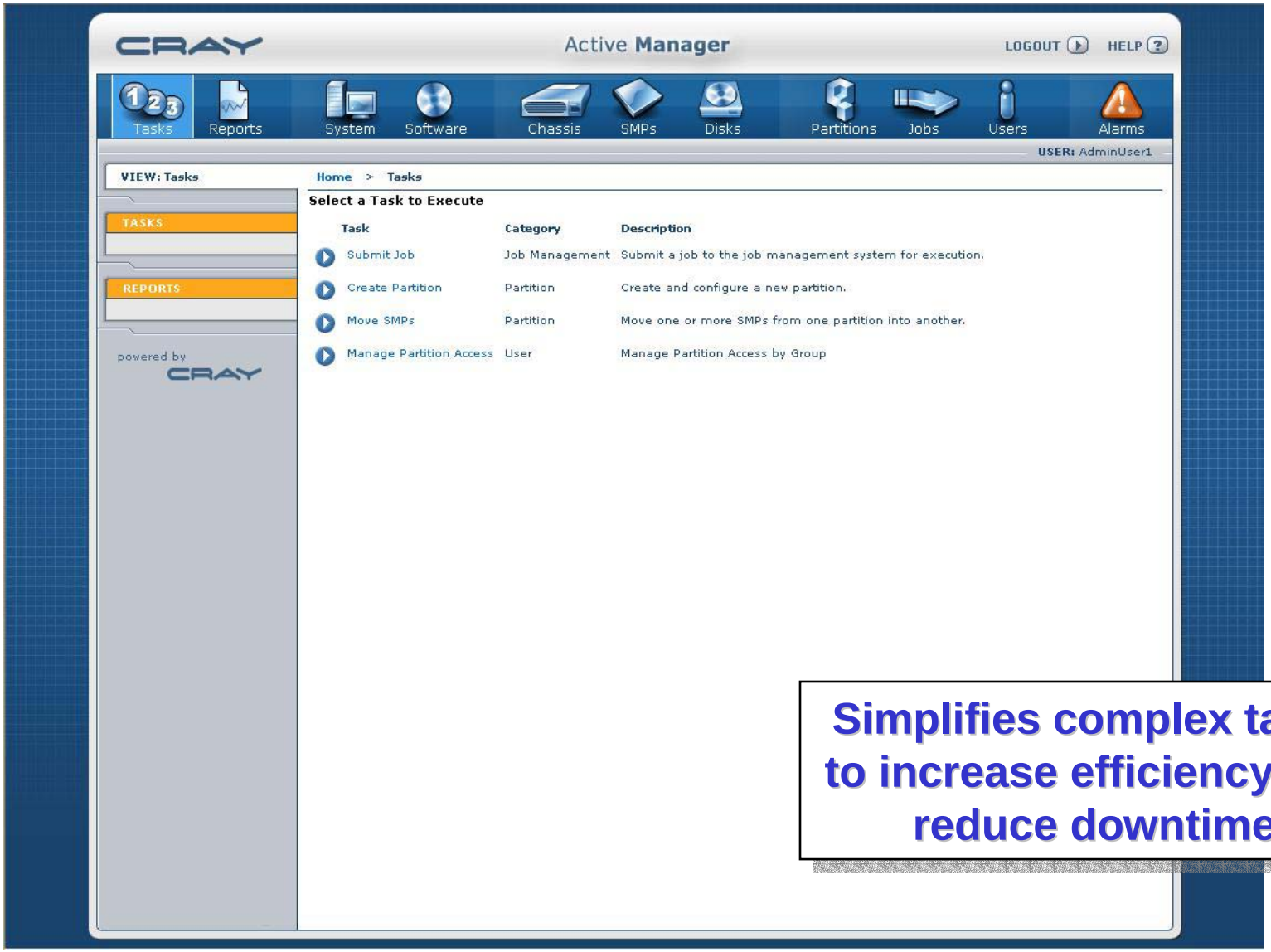


Name	Login	Jobs	Status	SMPs	Load(1)	Load(5)	Load(15)	Jobs Running	Jobs Queued	Alarm
admin	Yes	No	open	1	0.97	1.12	1.02	0	0	0
cancun	Yes	No	open	0	0.00	0.00	0.00	0	0	0
crPart	Yes	Yes	open	0	0.00	0.00	0.00	0	0	0
crPart2	Yes	Yes	closed	0	0.00	0.00	0.00	0	0	0
czcomp1	No	Yes	open	0	0.00	0.00	0.00	0	0	0
igPart	Yes	No	closed	0	0.00	0.00	0.00	0	0	0
ira	Yes	No	open	0	0.00	0.00	0.00	0	0	0
iraPart	Yes	No	open	0	0.00	0.00	0.00	0	0	0
loginPart	No	Yes	closed	0	0.00	0.00	0.00	0	0	0
pexecution-1a	No	Yes	open	1	0.00	0.01	0.02	0	4	0
pexecution-1b	No	Yes	open	0	0.00	0.00	0.00	0	0	0
plogin-1a	Yes	No	open	0	0.00	0.00	0.00	0	0	0
sfuHPC	Yes	Yes	closed	0	0.00	0.00	0.00	0	0	0

Managing virtual computers instead of individual processors

CUG 2004

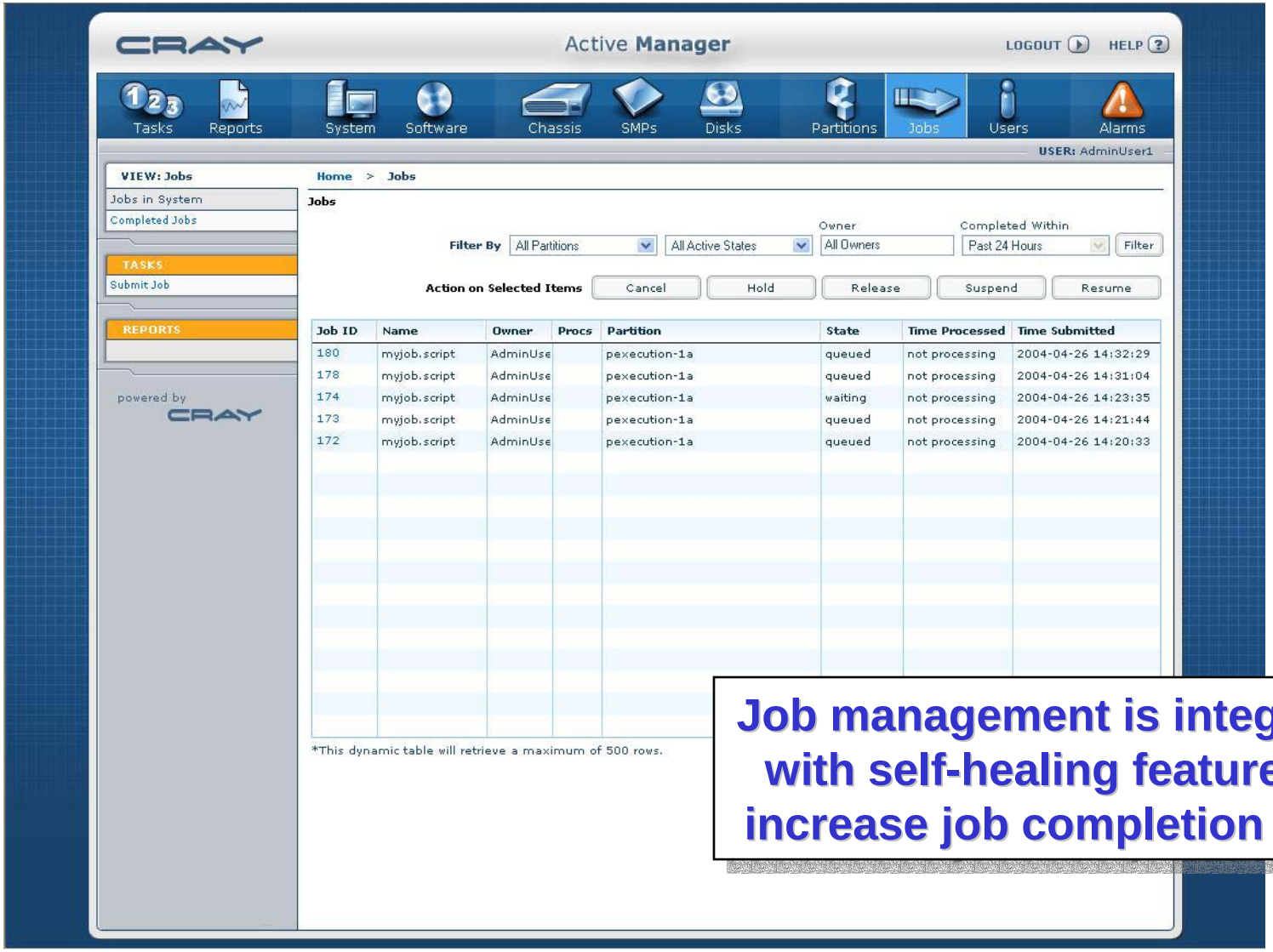
Active Manager Task Wizard



Simplifies complex tasks to increase efficiency and reduce downtime

CUG 2004

Active Manager Job Scheduler



Job management is integrated with self-healing features to increase job completion rates

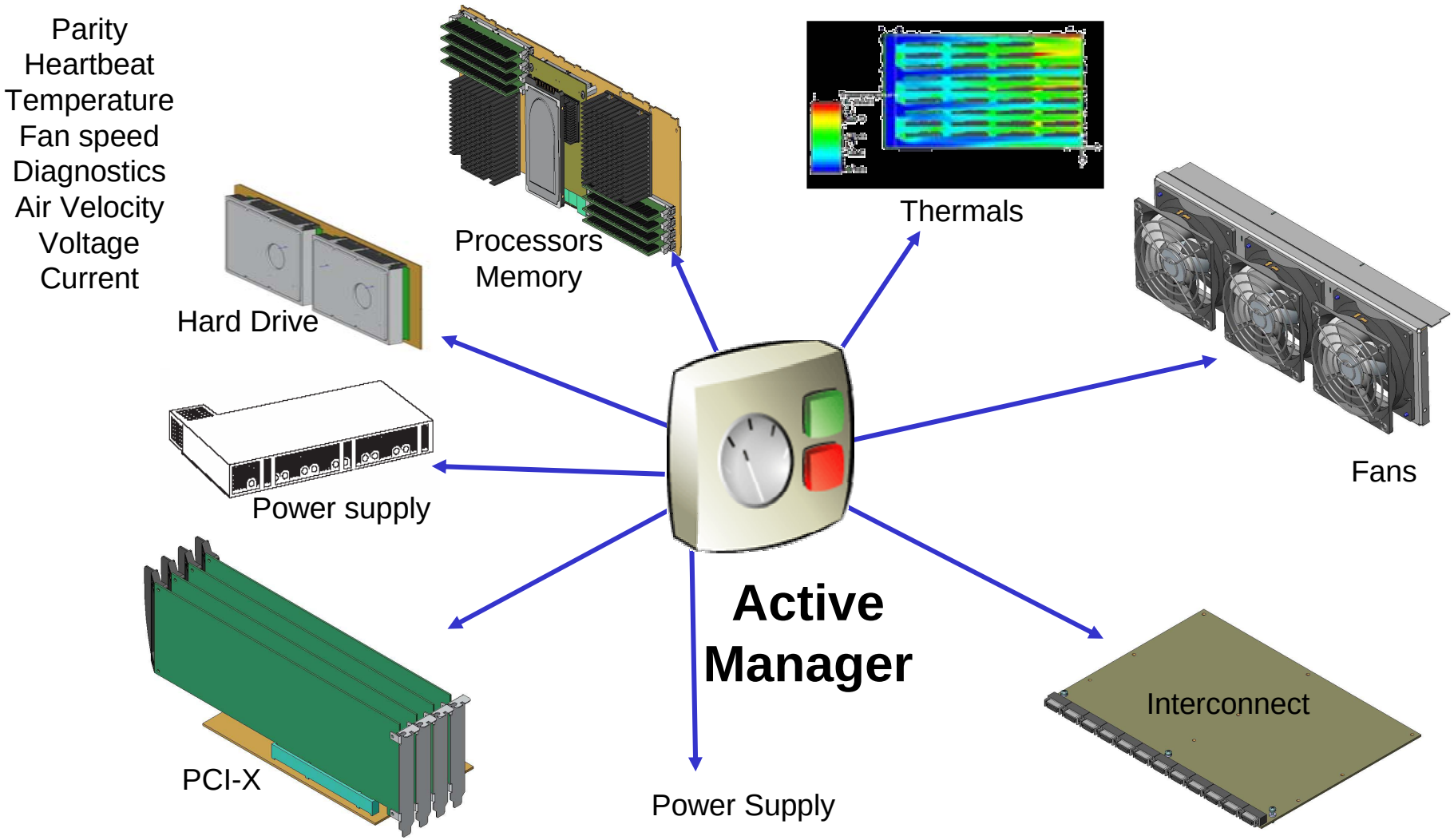
CUG 2004

Active Manager User Portal

Intuitive user access for job submission and status lets users focus on applications, not computing

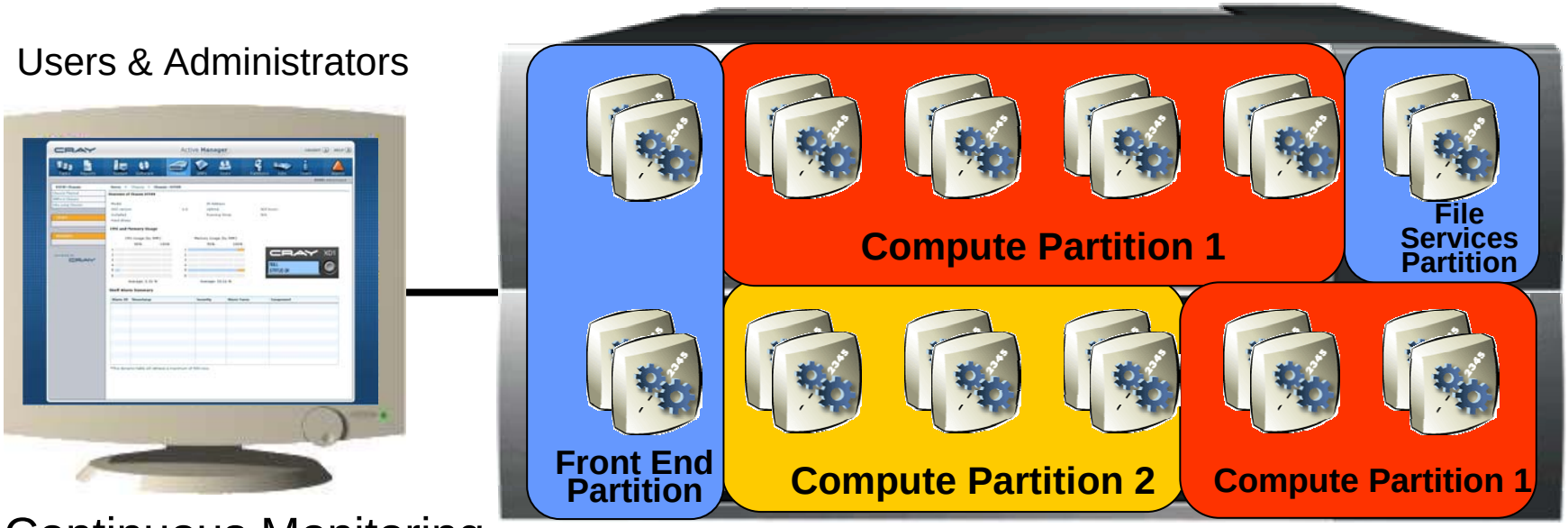
CUG 2004

Self-Monitoring



Dedicated Management Processor, OS, Fabric

Self-Healing

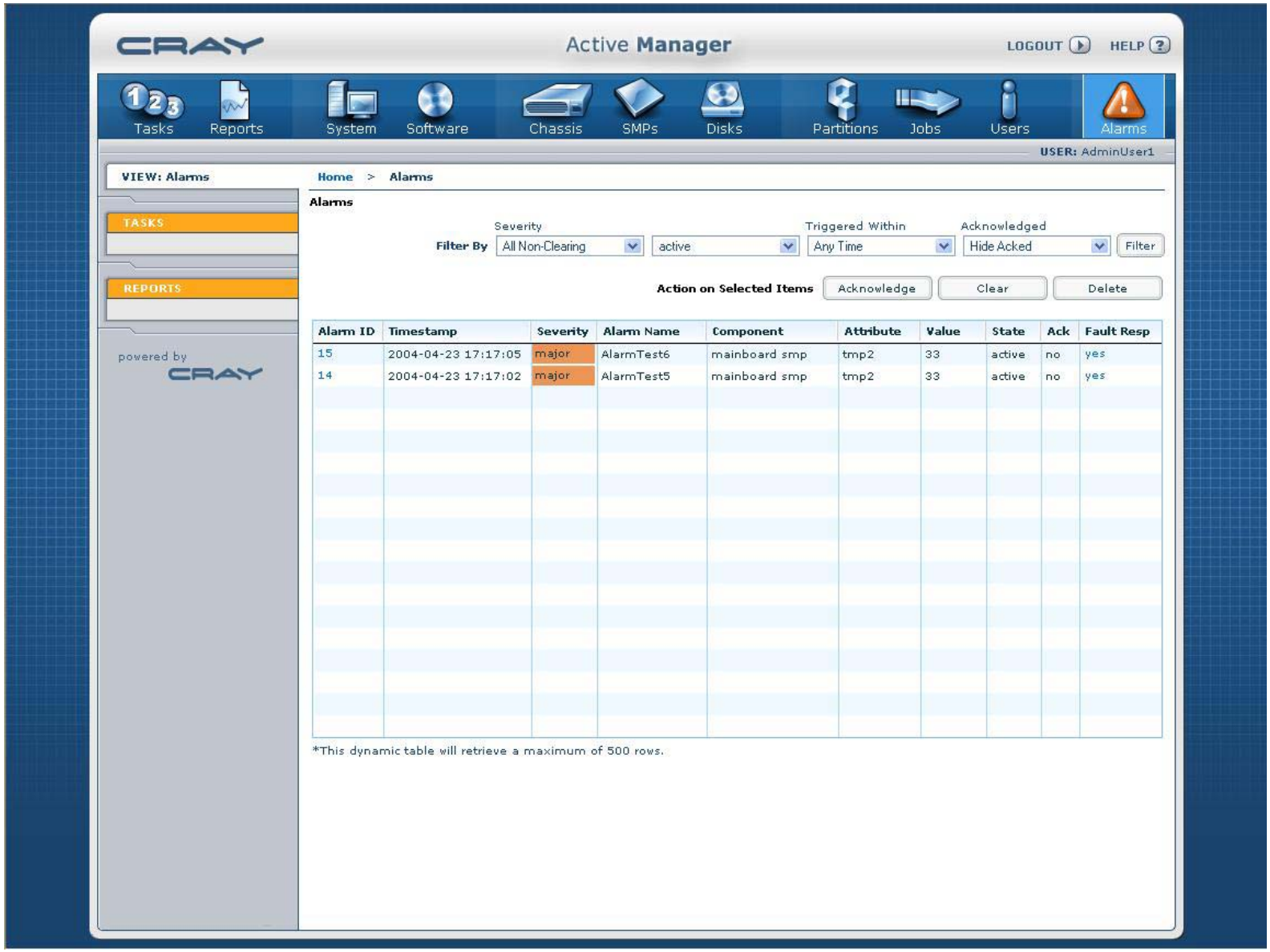


- Continuous Monitoring
- Detect (Future) Failure
- Attempt reset
- Isolate failed component
- Re-allocate resources (N+1 sparing or policy-based)

Automated Recovery Reduces MTTR from hours to minutes

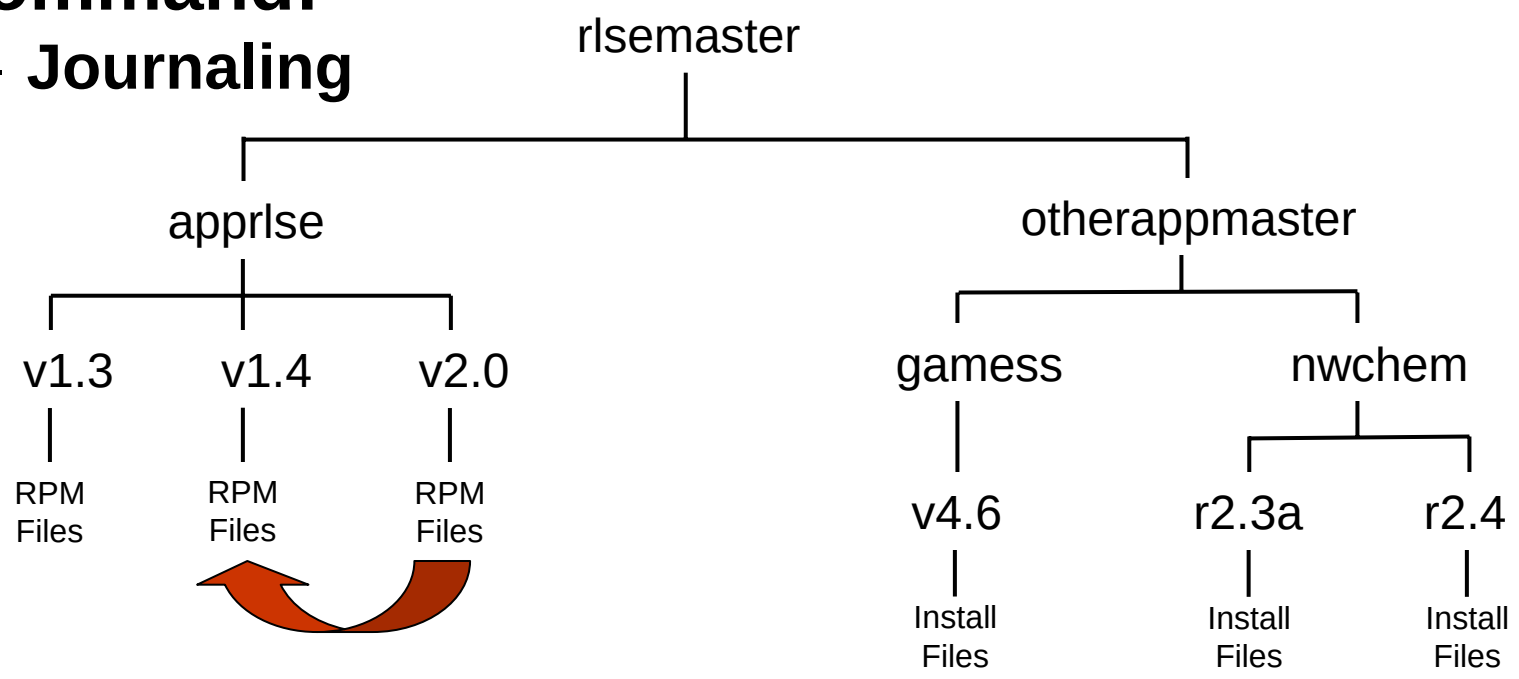
Active Manager Thermal Management

Active Manager Alarm Management



Roll back

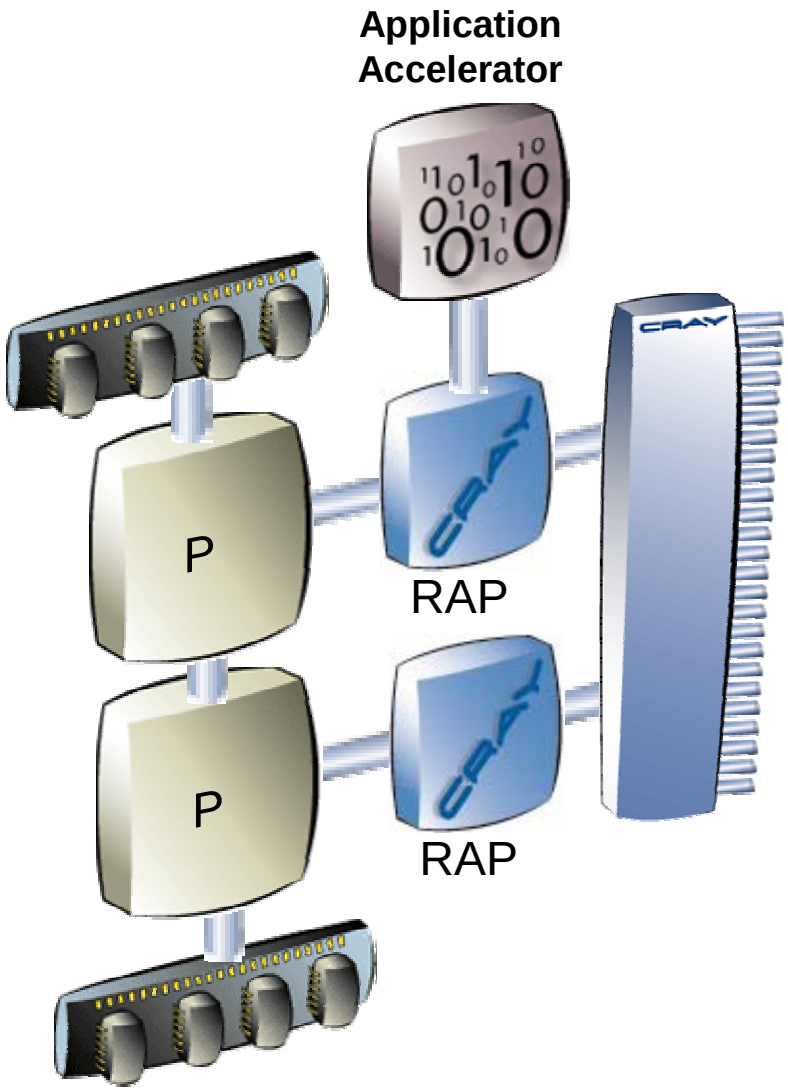
- **Undo command:**
  - **Journaling**



Quick Rollback Reduces MTTR from hours to minutes

Application Acceleration FPGA

Application Acceleration

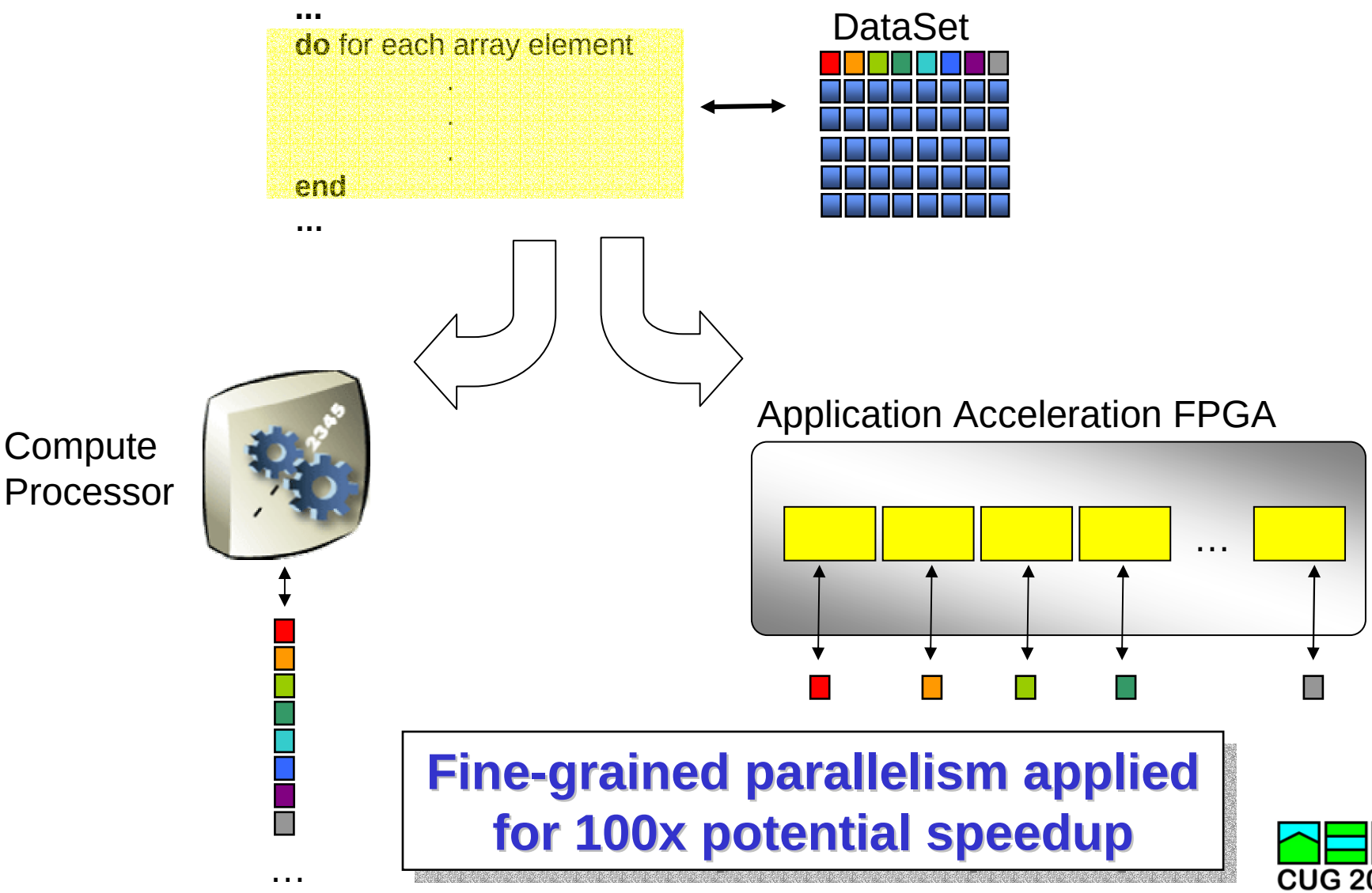


Application Acceleration

- *Reconfigurable Computing*
- Tightly coupled to Opteron
- FPGA acts like a programmable co-processor
- Well-suited for:
  - Searching, sorting, signal processing, audio/video/image manipulation, error correction, coding/decoding, packet processing, random number generation.

SuperLinear speedup for key algorithms

Application Acceleration FPGA



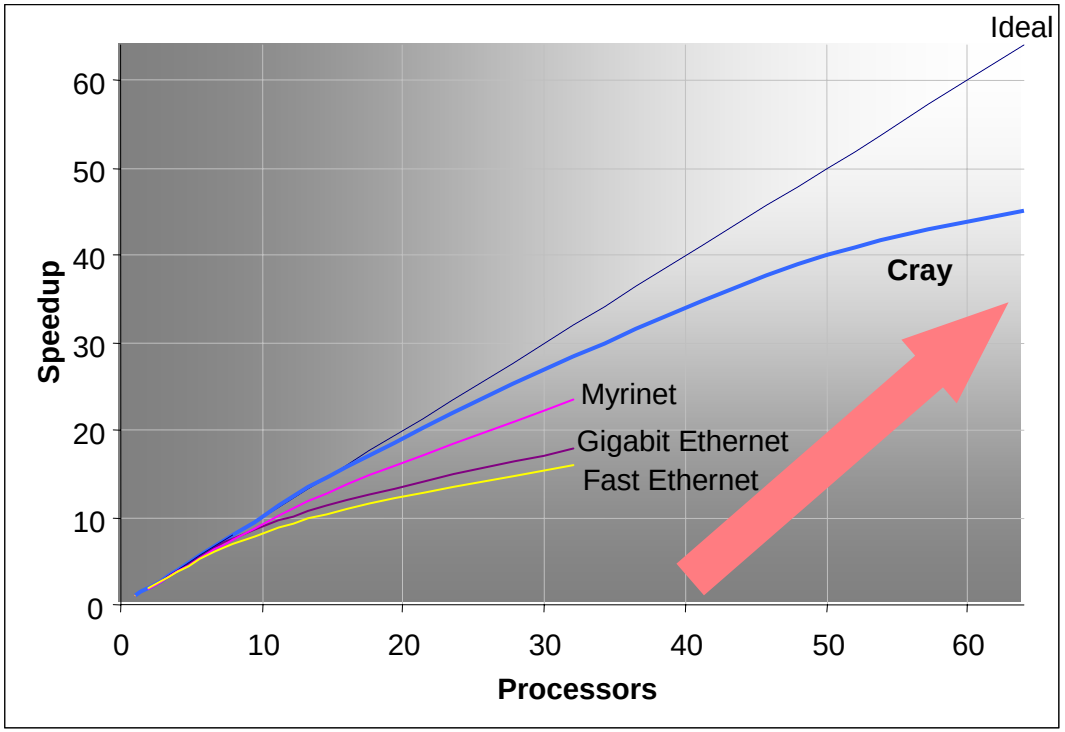
Fine-grained parallelism applied for 100x potential speedup

FPGA and Vector Processors

Vector Processors	FPGAs
• SIMD – instruction level parallelism	• MIMD – code fragments are replicated
• Mature Development Environment (C, Fortran)	• Emerging Development Tools (mostly HW tools – VHDL, Verilog)
• Integer or Floating Point	• Integer or Fixed Point
• Suited to matrix operations: BLAS, LAPACK, …	• Suited to pre-processing (signal processing, sorting/searching, error correction, coding/decoding, packet processing)

Cray XD1: Built for Performance

Faster interconnect throughput
Lower interconnect latency
System-wide process synchronization

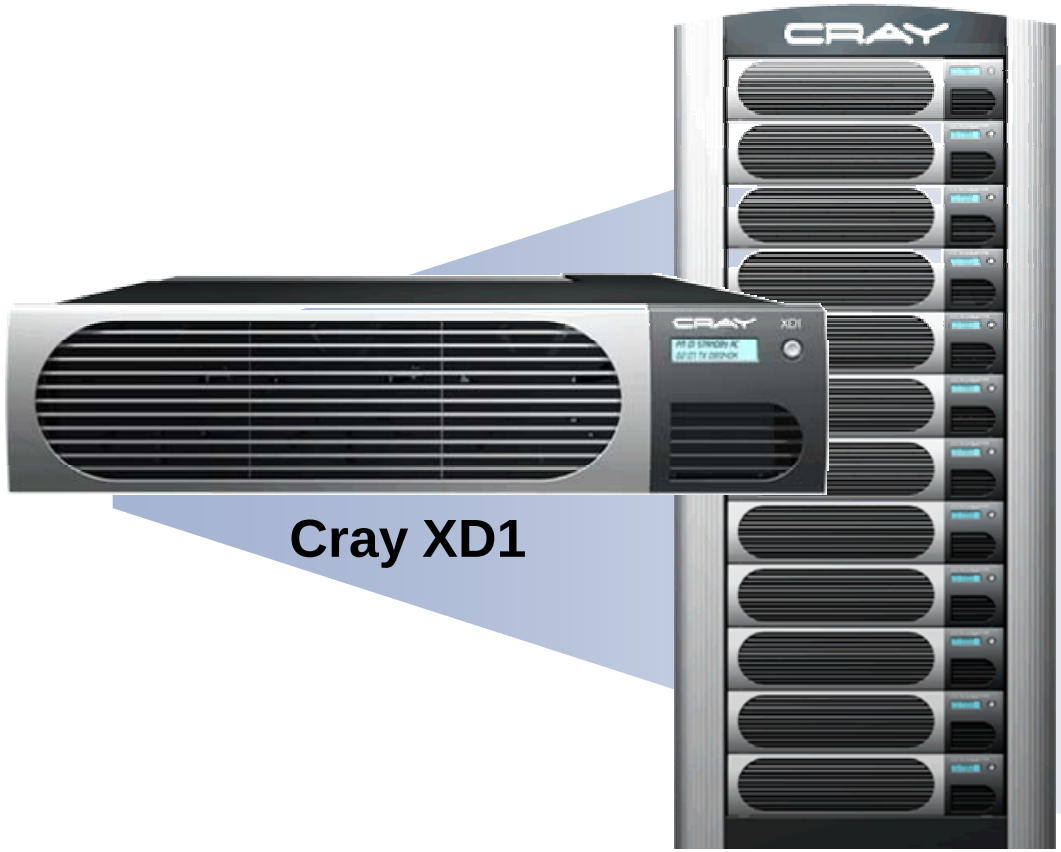


Greater Efficiency
Greater Scalability
Faster Performance

Increasing application efficiency and scalability for breakthrough performance gains

CUG 2004

The Cray XD1



Cray XD1

- Built for price/performance
  - Interconnect bandwidth/latency
  - System-wide process synchronization
  - Application Acceleration FPGAs
- Standards-based
  - 32/64-bit X86, Linux, MPI
- High resiliency
  - Self-configuring, self-monitoring, self-healing
- Single system command & control
  - Intuitive, tightly integrated management software

Purpose-built and optimized for high performance workloads

CUG 2004

Questions?

Amar Shan, Senior Product Marketing Manager

shan@cray.com

Additional Slides

Cray System Portfolio

Cray X1

- 1 to 50+ TFLOPS
- $3 M - $10 M+
- Vectorized apps
- Cray MSP/UNICOS/mp

RS/Strider

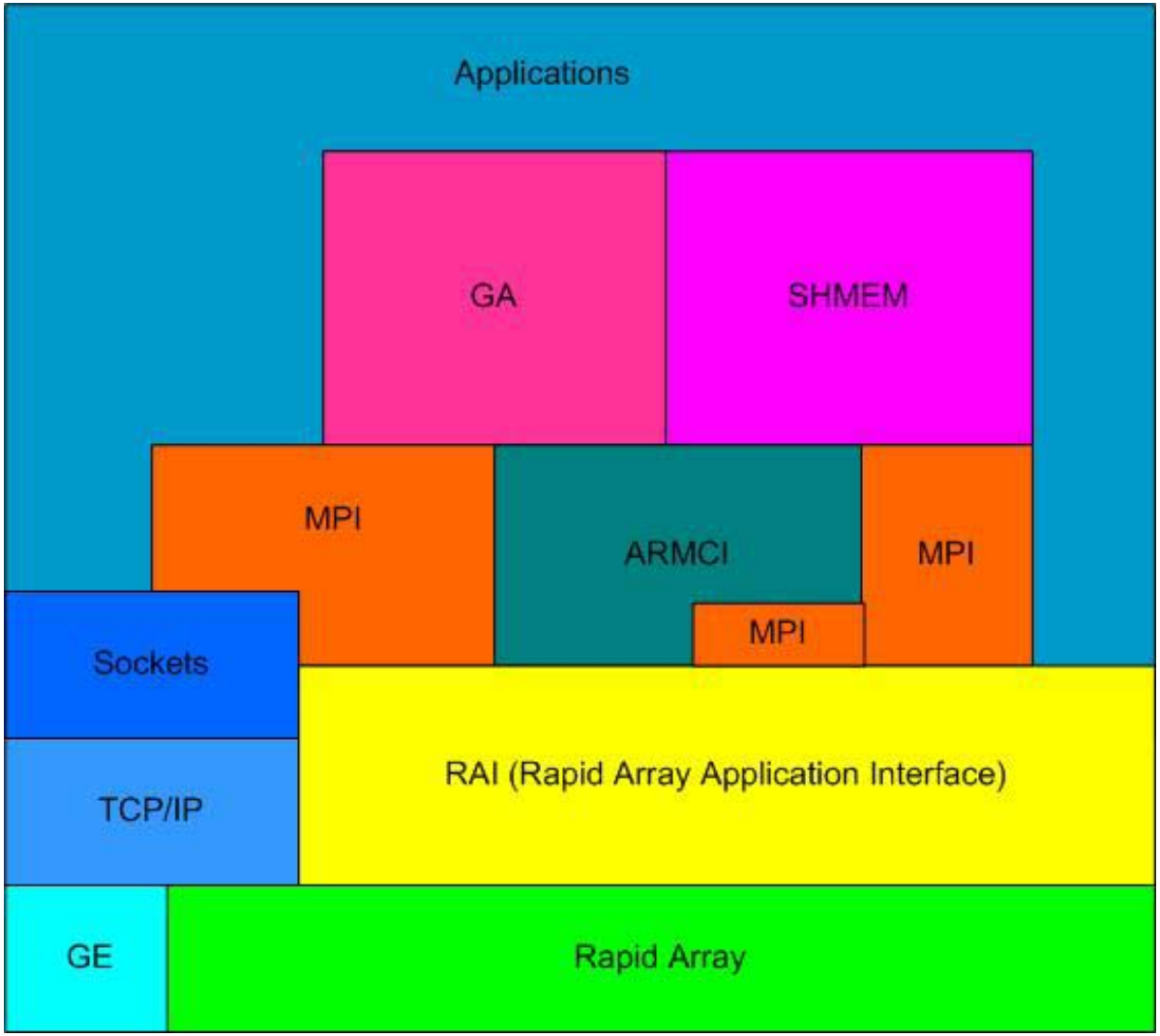
- 1 to 50+ TFLOPS
- 256 – 10,000+ processors
- $1 M - $5 M+
- Opteron/Linux/Catamount

Cray XD1

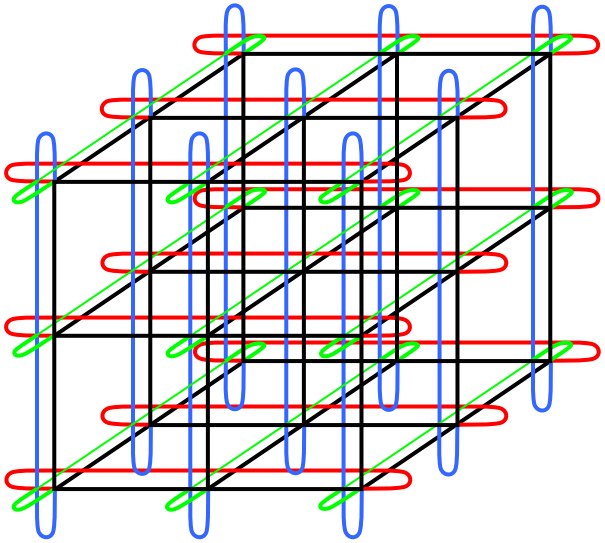
- 48 GFLOPS - 1.2+ TFLOPS
- 12 – 288+ processors
- $50 K - $2 M+
- AMD Opteron/Linux

Purpose-Built High Performance Computers

Communications Protocols



Switchless Toroid Topology



3D torus

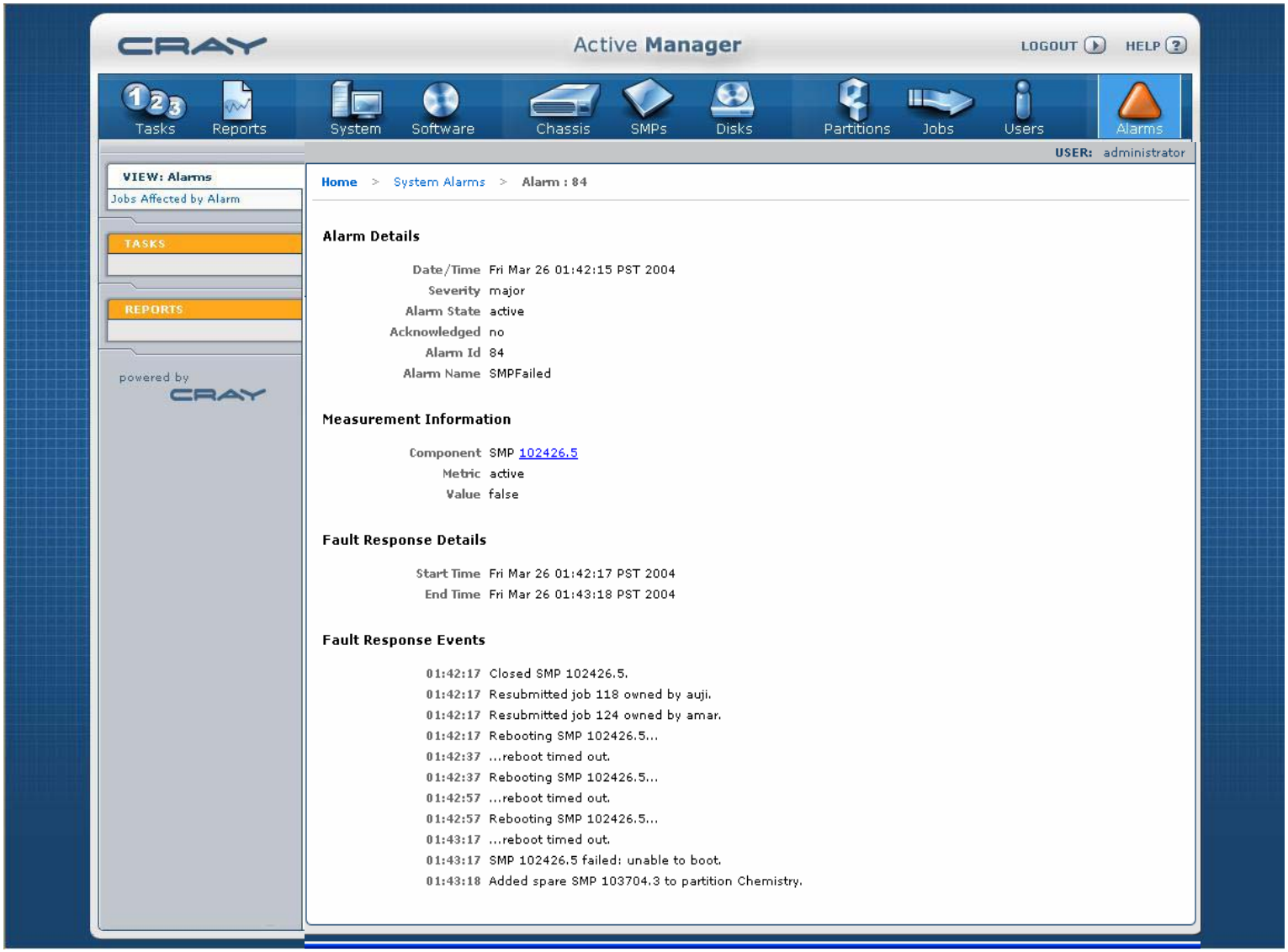
27 chassis, 324 Processors
1.4 TFLOP
3x3x3 Toroid

Nearest Neighbor
- 4-8 GB/s bandwidth per SMP
- 1.8 µsec latency

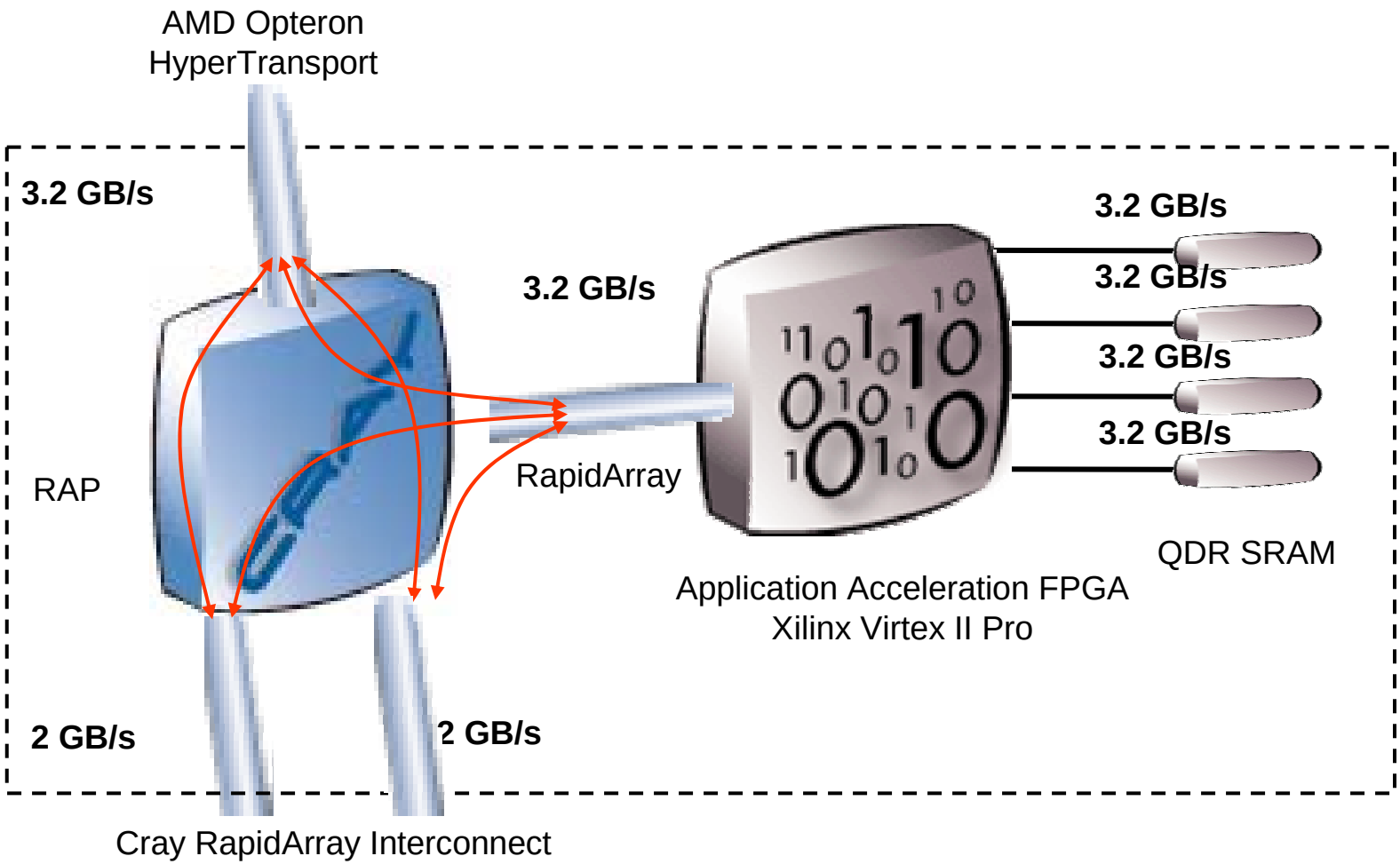
Worst case: 6 Hops, 2.8 µsec

Well-Suited for Nearest Neighbor Problems

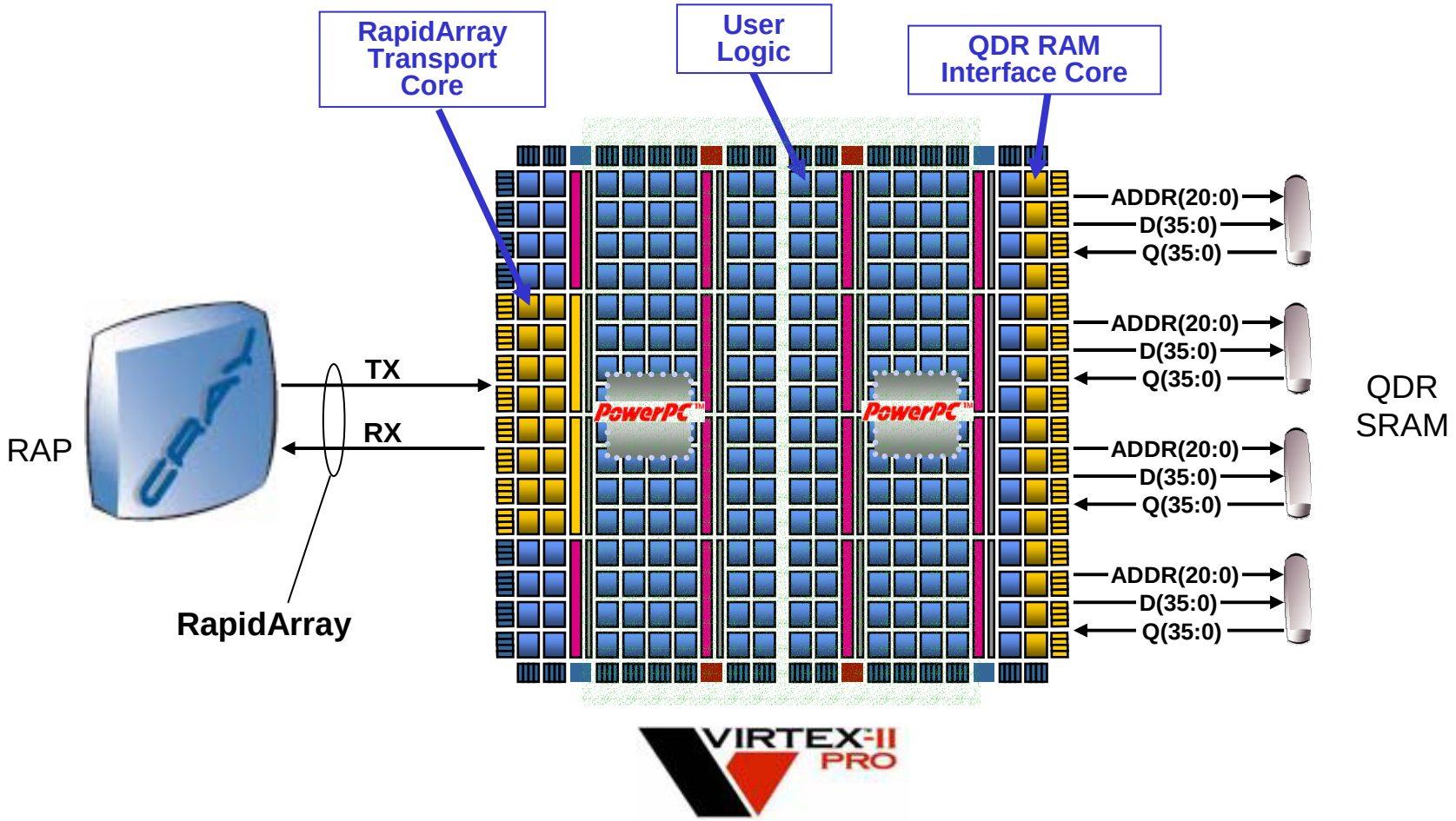
Active Manager Self Healing Policies



Application Acceleration Co-Processor



Application Acceleration Interface



- XC2VP30 running at 200 MHz.
- 4 QDR II RAM with over 400 HSTL-I I/O at 200 MHz DDR (400 MTransfers/s).
- 16 bit simplified HyperTransport I/F at 400 MHz DDR (800 MTransfers/s.)
- QDR and HT I/F take up <20 % of XC2VP30. The rest is available for user applications.

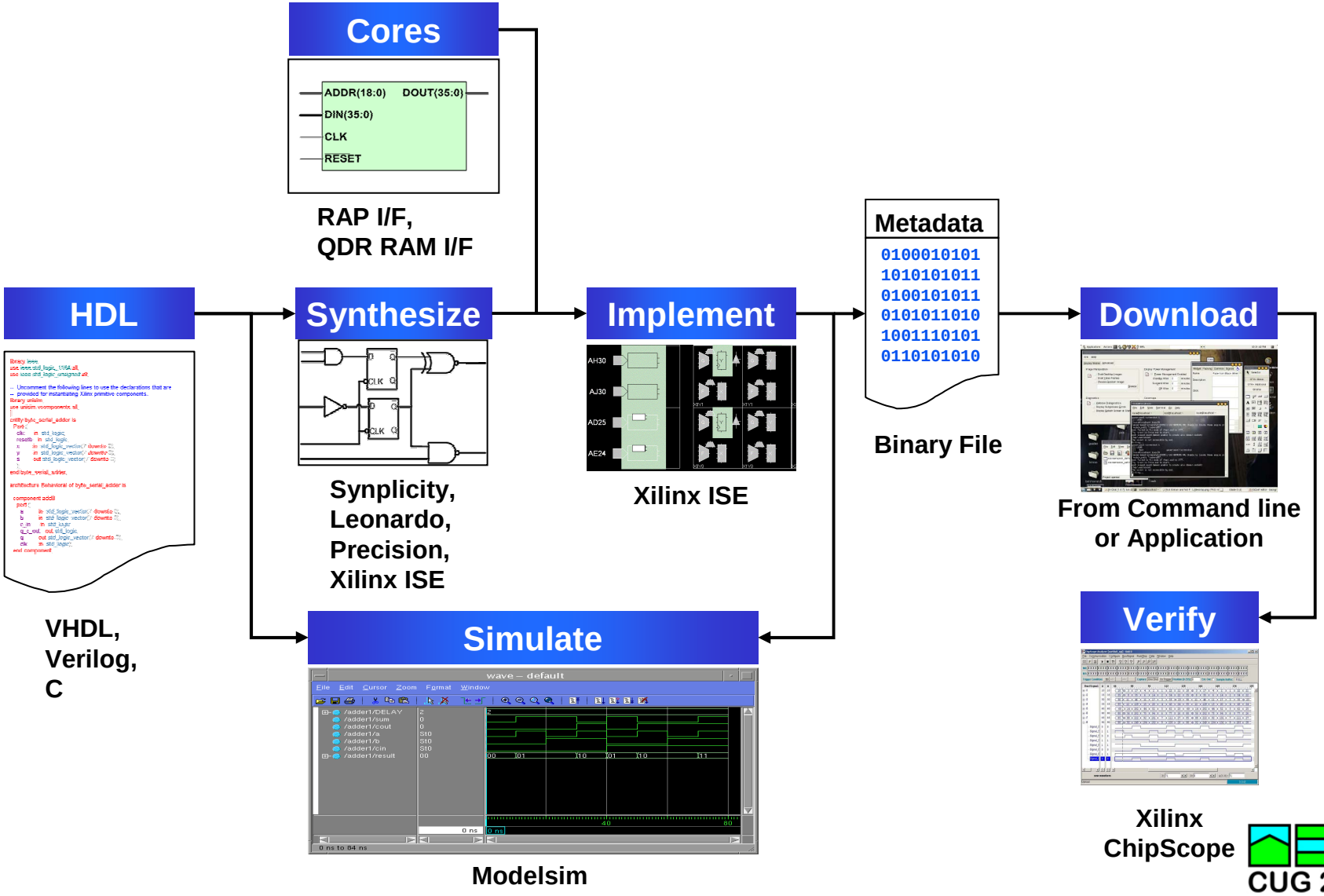
Application Acceleration Variants

A variety of Application Acceleration variants can be manufactured by populating different pin compatible FPGAs and QDR II RAMs.

FPGA	Speed	Logic Elements	PowerPC	18x18 Multipliers
XC2VP30	-6	30,816	2	136
XC2VP50	-7	53,136	2	232

RAMs	Speed	Dimensions	Quantity	Total Size
K7R643682	**200 MHz**	**1M x 36**	**4**	**16 MByte**

FPGA Development Flow



FPGA Linux API

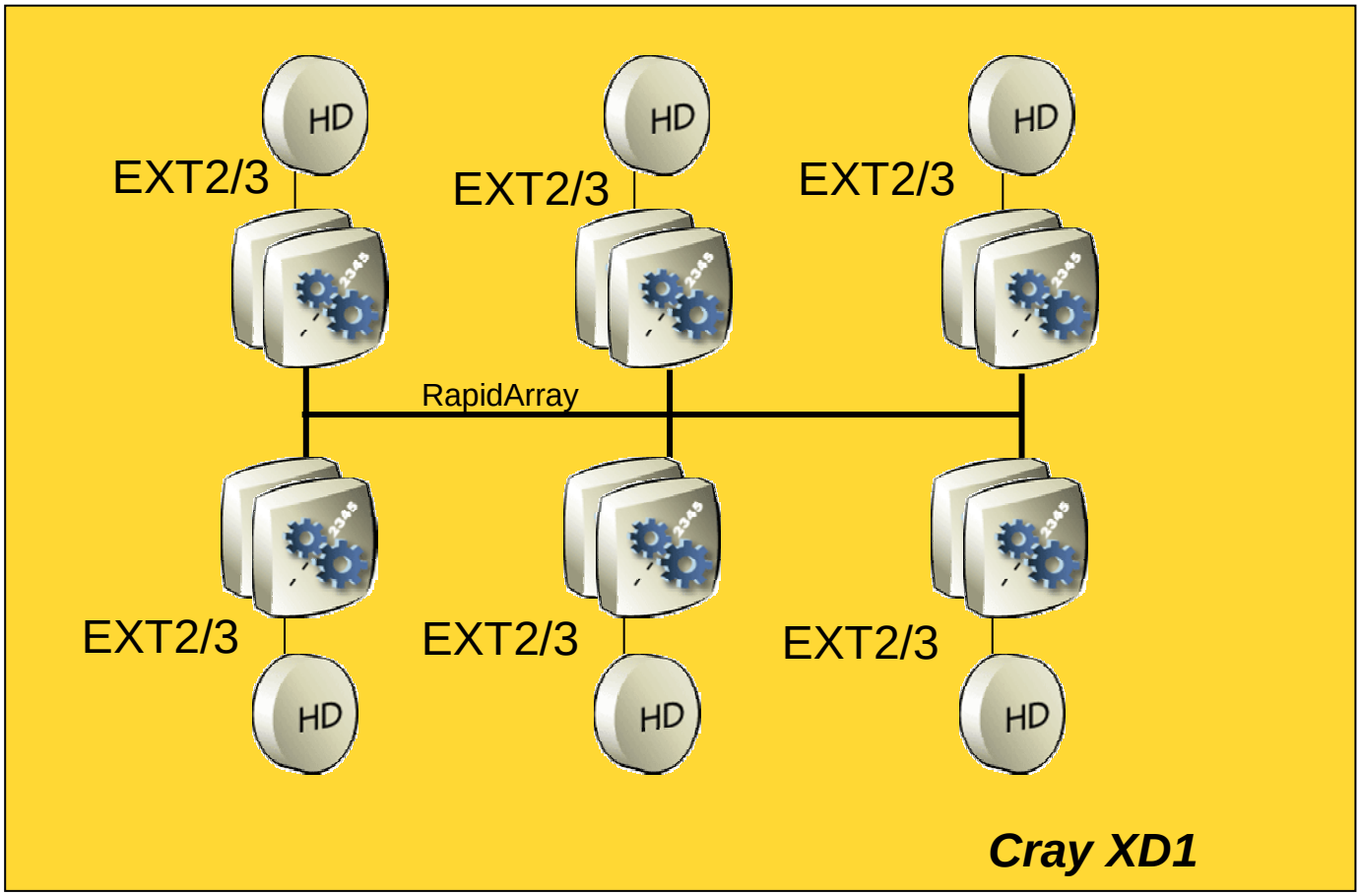
- Admininstration Commands
  - fpga_open – allocate and open fpga
  - fpga_close – close allocated fpga
  - fpga_load – load binary into fpga
- Control Commands
  - fpga_start – start fpga (release from reset)
  - fpga_stop – stop fpga
- Status Commands
  - fpga_status – get status of fpga
- a
  ram

Programmer sees get/put and message passing programming model

Storage Strategy

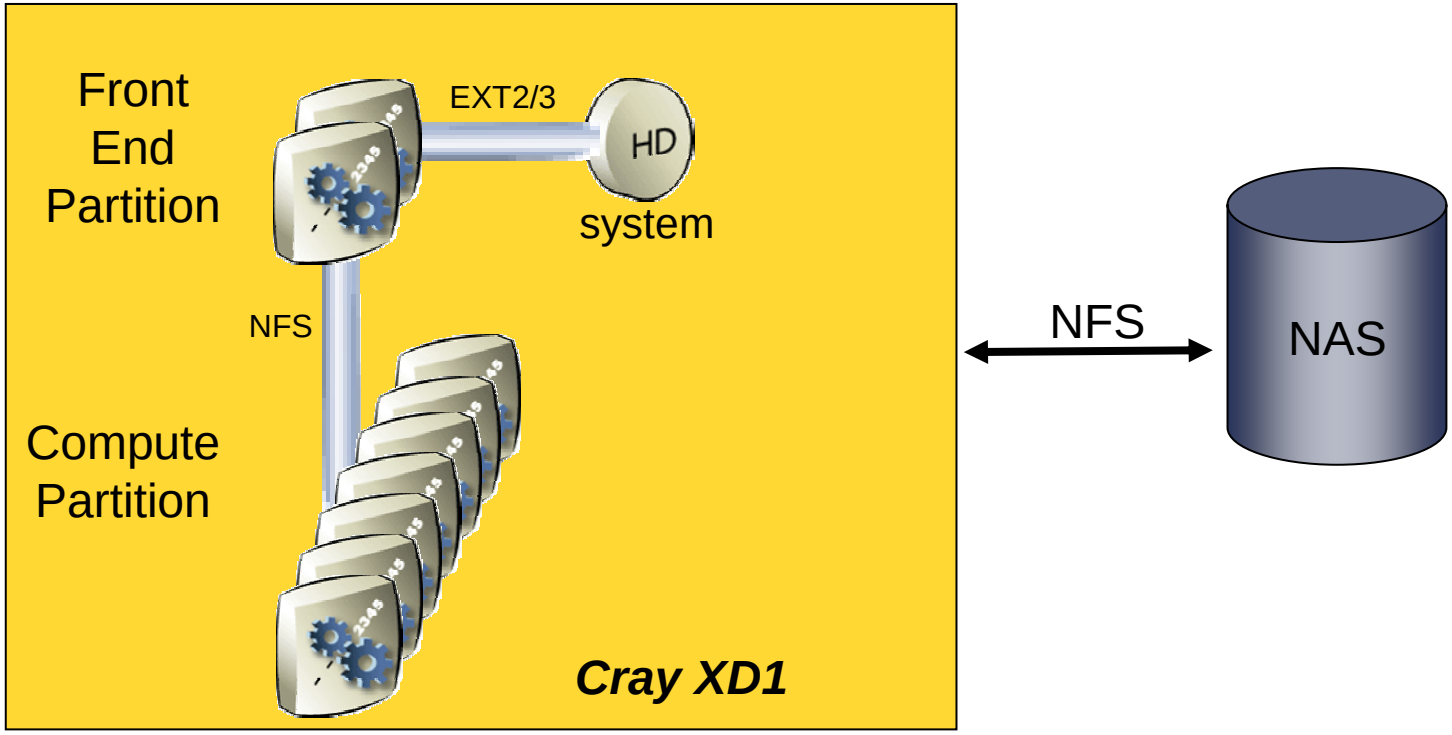
File Systems: Local Disks

One S-ATA HD per SMP; Local Linux directory per HD



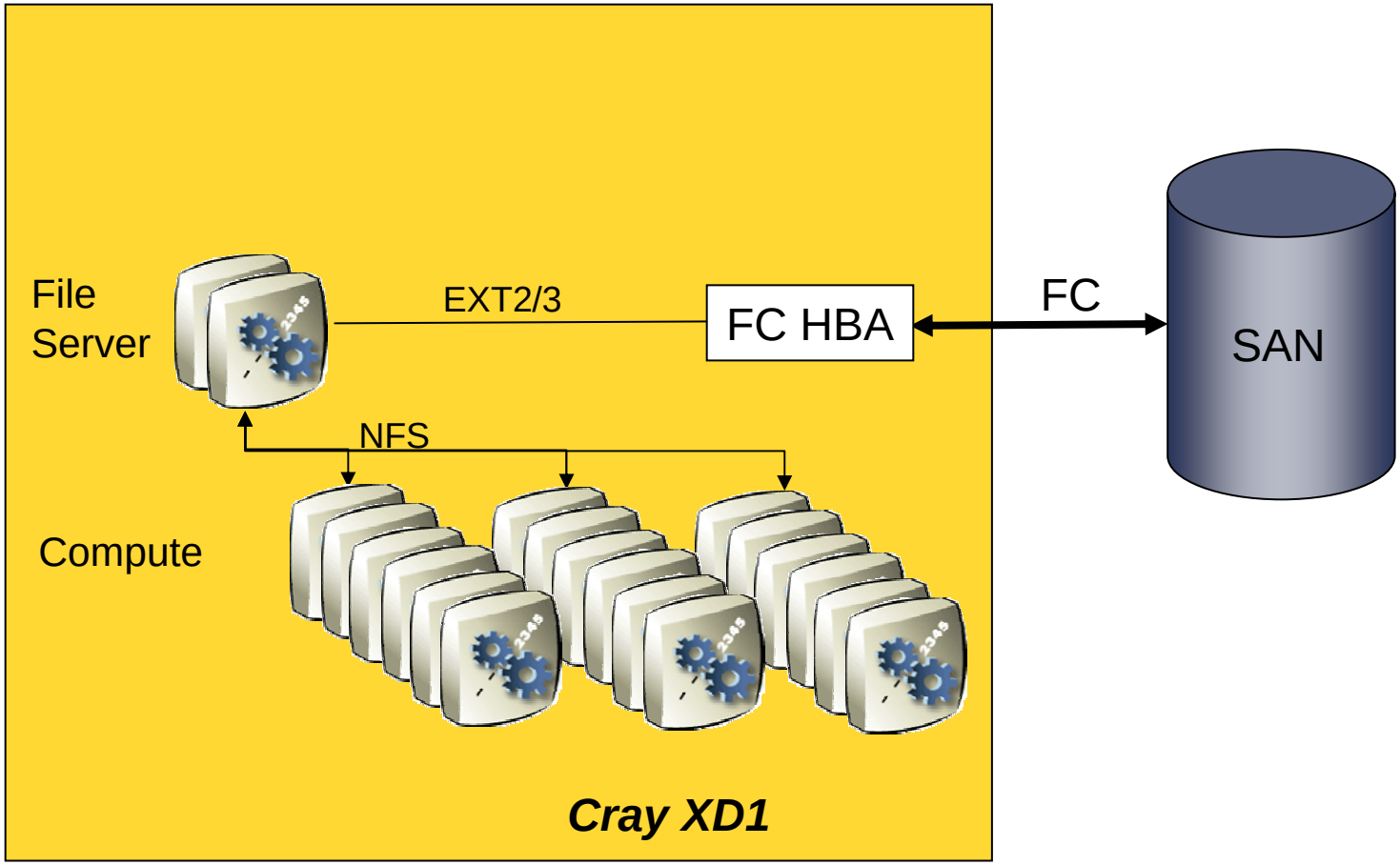
File Systems: Diskless compute nodes

Single Boot Disk for System, External NAS for User Data



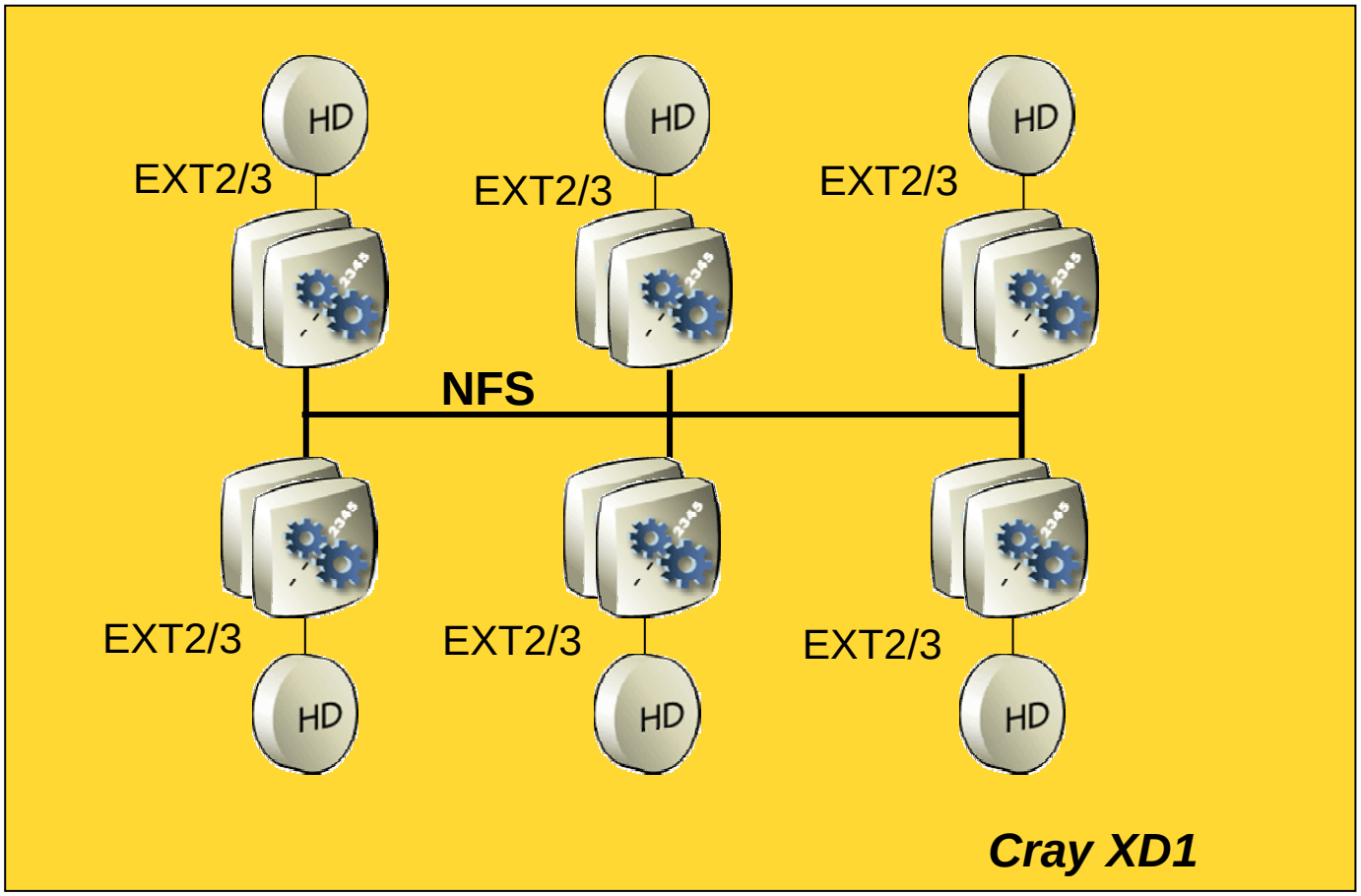
File Systems: SAN

SMP acting as a File Server for the SAN



File Systems: Local Disk per Compute Node

One disk per SMP; NFS Cross-mounting; External NAS Optional



Designed for High Availability

Fewer Failures

- **Stateless Hardware**
- **Reduced Variability**
- **Self-Configuring**

Faster Recovery

- **Self-Monitoring**
- **Self-Healing**
- **Software Rollbacks**

Today

99% Availability

- **100 minutes downtime / week**
- **1 failure / week**

Target

99.99% Availability

- **53 minutes downtime / year**
- **1 failure of 5 minutes/month**
- **No incremental cost**

One vendor
One phone call
Fully integrated and tested