

ASCI-RED Intro Course

Patrick Fay

Computational Scientist, Intel Corp.

pfay@lanl.gov

Topics

- ◆ ASCI-RED LAN(s) Overview
 - System Overview
 - Filesystems
 - Accounts
 - Access
 - File Transfer
 - Getting Help
- ◆ janus Overview
- ◆ OS descriptions
- ◆ Interacting with the Compute Partition
- ◆ How to compile and run hello_world.[fc]
- ◆ Programming model
- ◆ Languages
- ◆ IO
- ◆ Message-passing
- ◆ Tools
- ◆ Performance Tips for the PPRO
- ◆ Getting to the 2nd processor
- ◆ How ASCI-RED differs from other Unix systems

ASCI-RED LAN(s) Overview

◆ System Overview - 2 LANs (black,red)

— black (open LAN):

- janus.sandia.gov

Parallel supercomputer using P6 (Pentium Pro)

System Overview:

<http://mephisto.ca.sandia.gov/TFLOP/sc96/index.html>

Used for running parallel applications (only)

- sasn100.sandia.gov

Sun Ultrasparc server

Used for code compilation, pre/post processing.

- smss

archival tape storage (new NSS)

SMSS Overview:

http://www.sandia.gov/SMSS/smss_index.html

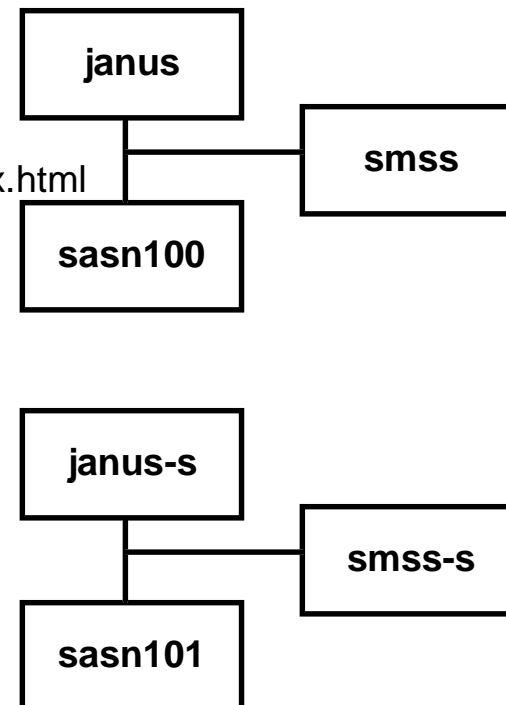
— red (classified LAN)

Mirrors black side. email janus-help@sandia.gov
if it doesn't

- janus-s.sandia.gov

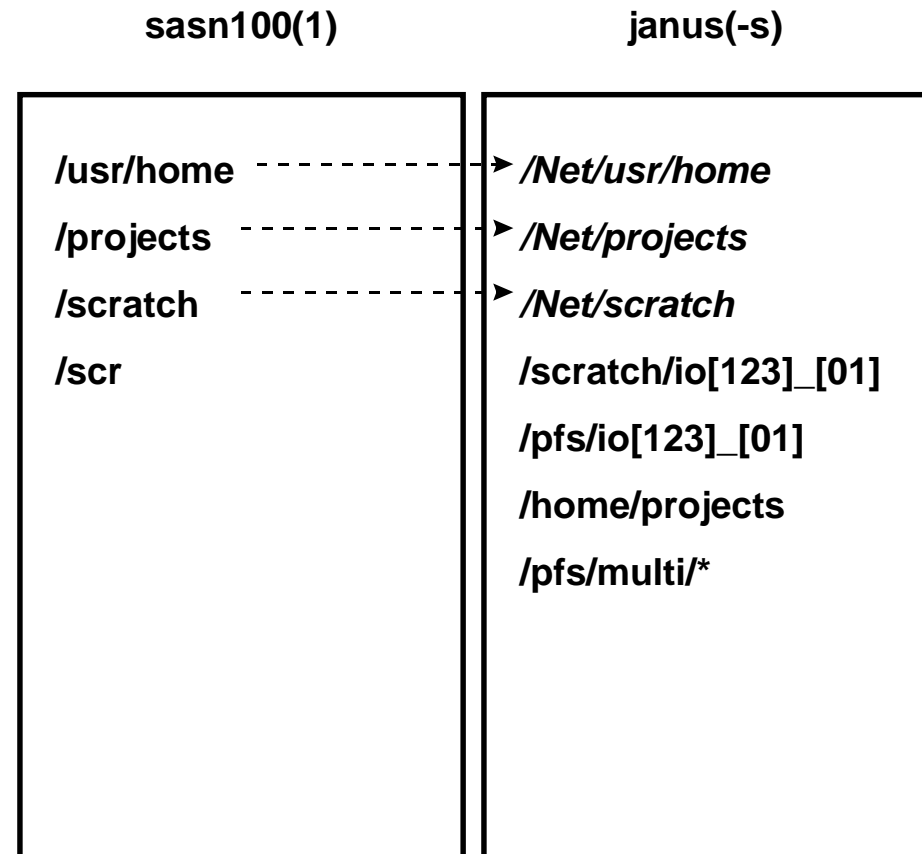
- sasn101.sandia.gov, sun server

- smss-s, tape storage



Filesystem Diagram

- ◆ home areas are on sasn100(1)
 - Don't do parallel IO to your home. Slow...
 - limited disk space
- ◆ Large codes on /projects
 - email
 - janus-help@sandia.gov to get an area
- ◆ IO to local disks
 - /scratch, /pfs
 - aggressive purging policy
- ◆ DFS
 - to/from kerberized DCE cell, server
- ◆ Where to put your data:
 1. /scr area on sasn100
 2. Archival Storage (smss)
 3. Your own LAN



ASCI-RED Lan Overview (cont)

- ◆ Obtaining an account

- who: ASCI work only
- how:

See for instructions: <http://www.acl.lanl.gov/~pfay/teraflop/gaf2.html>

Follow the above instructions for an unclassified account (and classified account).

Contact David Gardner drgardn@cs.sandia.gov for classified account

ASCI-RED LAN Access

- ◆ Access: ssh and xterm

- Must have gotten dce account and ASCI-RED account
- Workstation must have kerberos/dce software
- Kerberos supported (I have sgi versions of ssh/scp/ftp)

```
.kent.pfay,/users/pfay {101} > kinit -f pfay@dce.lanl.gov
```

```
Password for pfay@dce.lanl.gov:
```

```
.kent.pfay,/users/pfay {102} > ssh janus.sandia.gov
```

```
Last login: Sat Feb 15 15:09:23 1997 from kent.acl.lanl.go
```

```
janus.sandia.gov -- the two-headed Intel TFLOP Supercomputer
```

```
*****
```

```
janus >
```

- ssh to do 'secure telnet' to ASCI-RED

```
% which ssh #see me for sgi versions of ssh/kinit
```

```
/Net/local/bin/ssh
```

```
% ssh sasn100.sandia.gov
```

```
your dce.lanl.gov password:
```

```
....
```

```
sasn100% xterm &
```

ASCI-RED LAN Overview (Cont.)

◆ File Transfer

— scp from/to <local-lan> to/from ASCI-RED lan

%**which scp** #see Pat Fay for sgi versions of scp

`/Net/local/bin/scp`

`scp <local_filename> sasn100.sandia.gov:<pathname>`

`your_dce_password:` #your dce password, note that you won't be prompted for a password if
you have an up-to-date version of kinit,scp,ssh

— ftp from/to ASCI-RED lan to/from local lan

`.kent.pfay,/users/pfay { 103} > ftp sasn100.sandia.gov`

Connected to sasn100.sandia.gov.

220 sasn100 FTP server (Version 5.60) ready.

334 Using authentication type GSSAPI; ADAT must follow
GSSAPI accepted as authentication type

GSSAPI error messages due to not running gssftpd daemon

GSSAPI authentication succeeded

Name (sasn100.sandia.gov:pfay): **(hit enter)**

232 GSSAPI user pfay@dce.lanl.gov is authorized as pfay

230 User pfay logged in.

Remote system type is UNIX.

Using binary mode to transfer files.

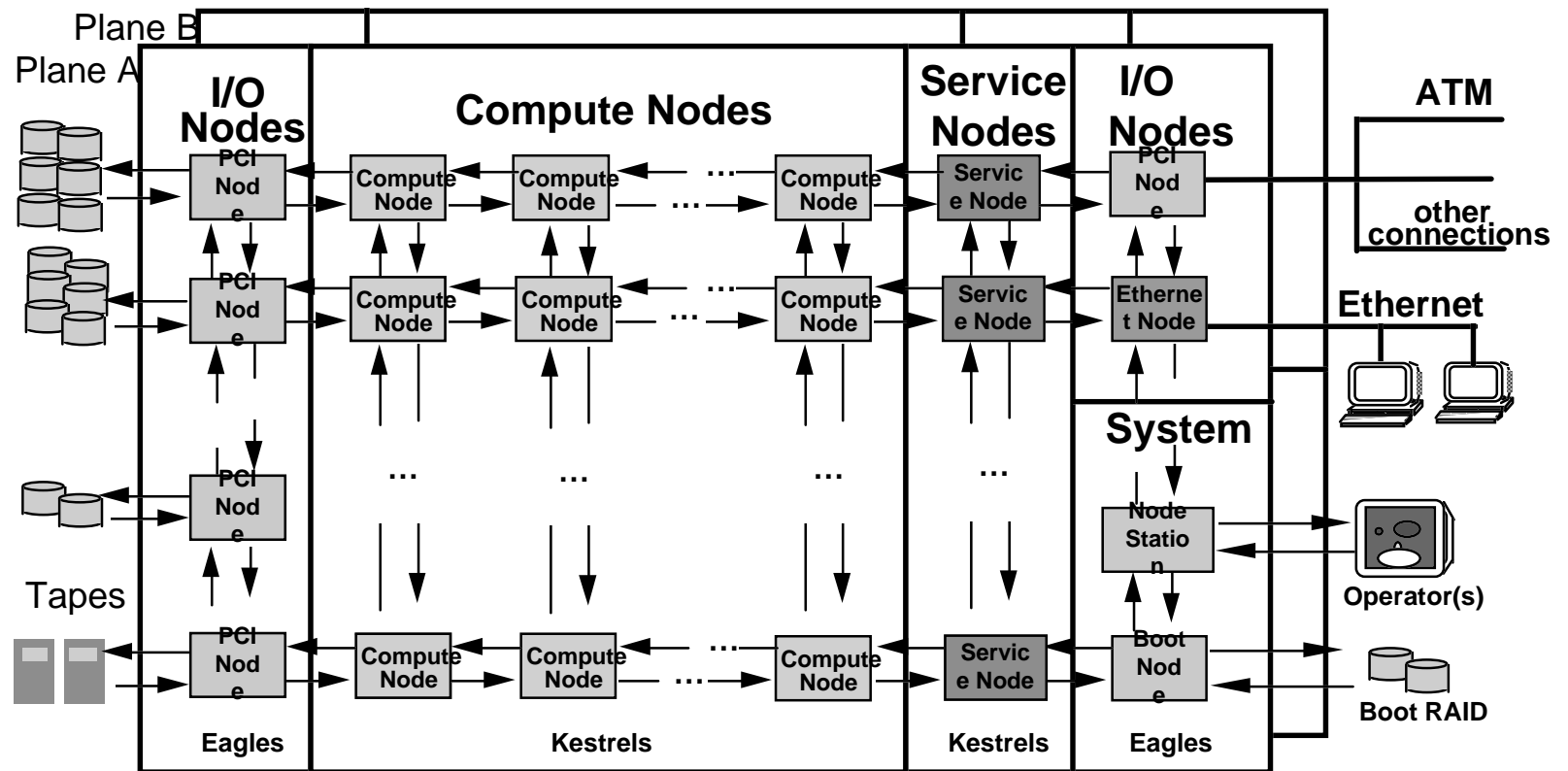
`ftp> passive` #Necessary due to not running gssftpd daemon on kent.

Passive mode off.

Getting Help

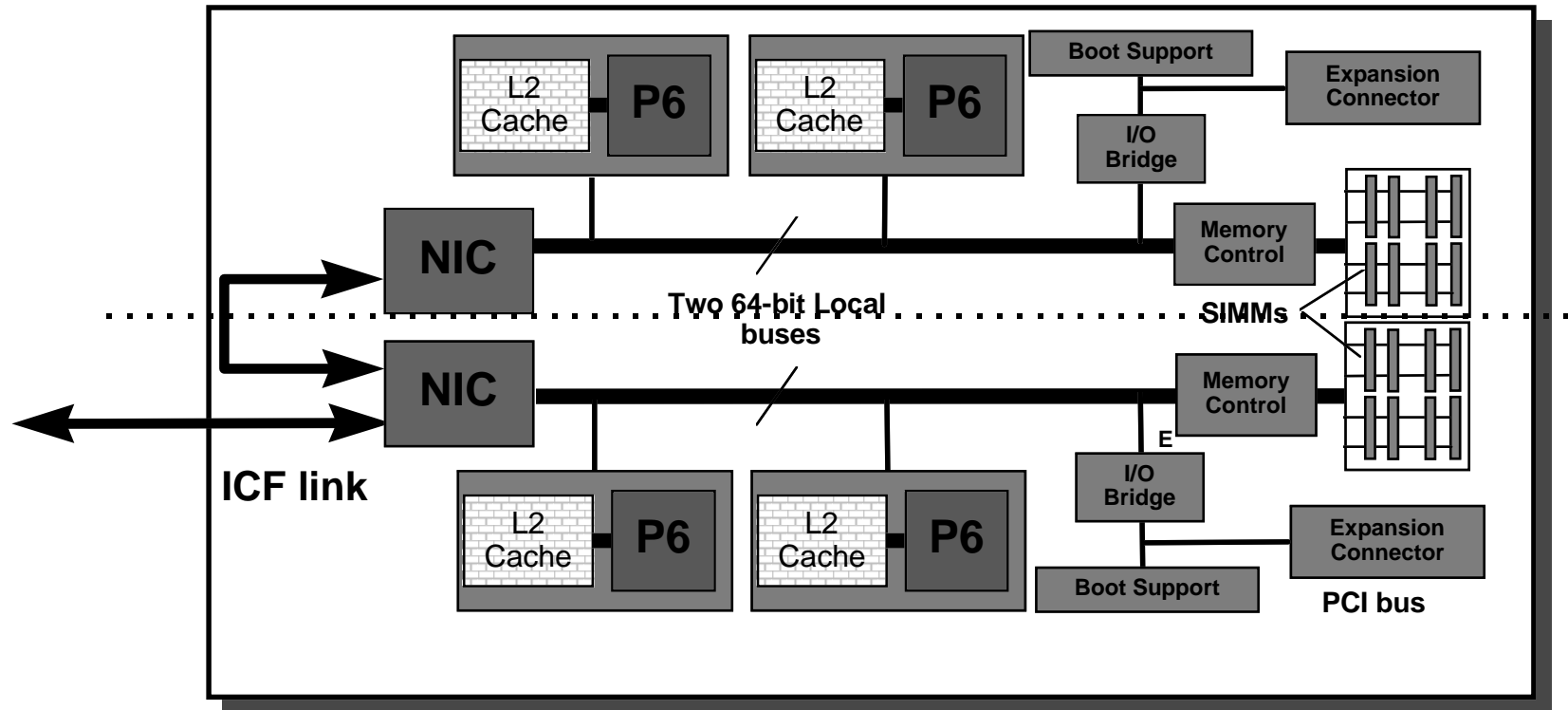
- ◆ ssh, scp, ftp on your workstations: your local sysadmin or Pat Fay
- ◆ Local sysadmin needs to install/config kerberos/dce software
- ◆ Usage problems: janus-help@sandia.gov
- ◆ Management/dedicated time request: janus-managers@sandia.gov
- ◆ mailing lists: janus-admin@sandia.gov
- ◆ Responses via:
 - janus-isn-users: classified or both
 - janus-irn-users: unclassified only
 - janus-users: all users
 - janus-info: users + other interested.
- ◆ News on sasn100
 - sasn100% **news janus-dedicate** or **news -a** for all topics
- ◆ Online information
 - General References
 - <http://mephisto.ca.sandia.gov/TFLOP/sc96/index.html>
 - <http://www.acl.lanl.gov/~pfay/teraflop>

janus Hardware View



- ◆ 2 planes
- ◆ Service nodes provide shell access, single-system image

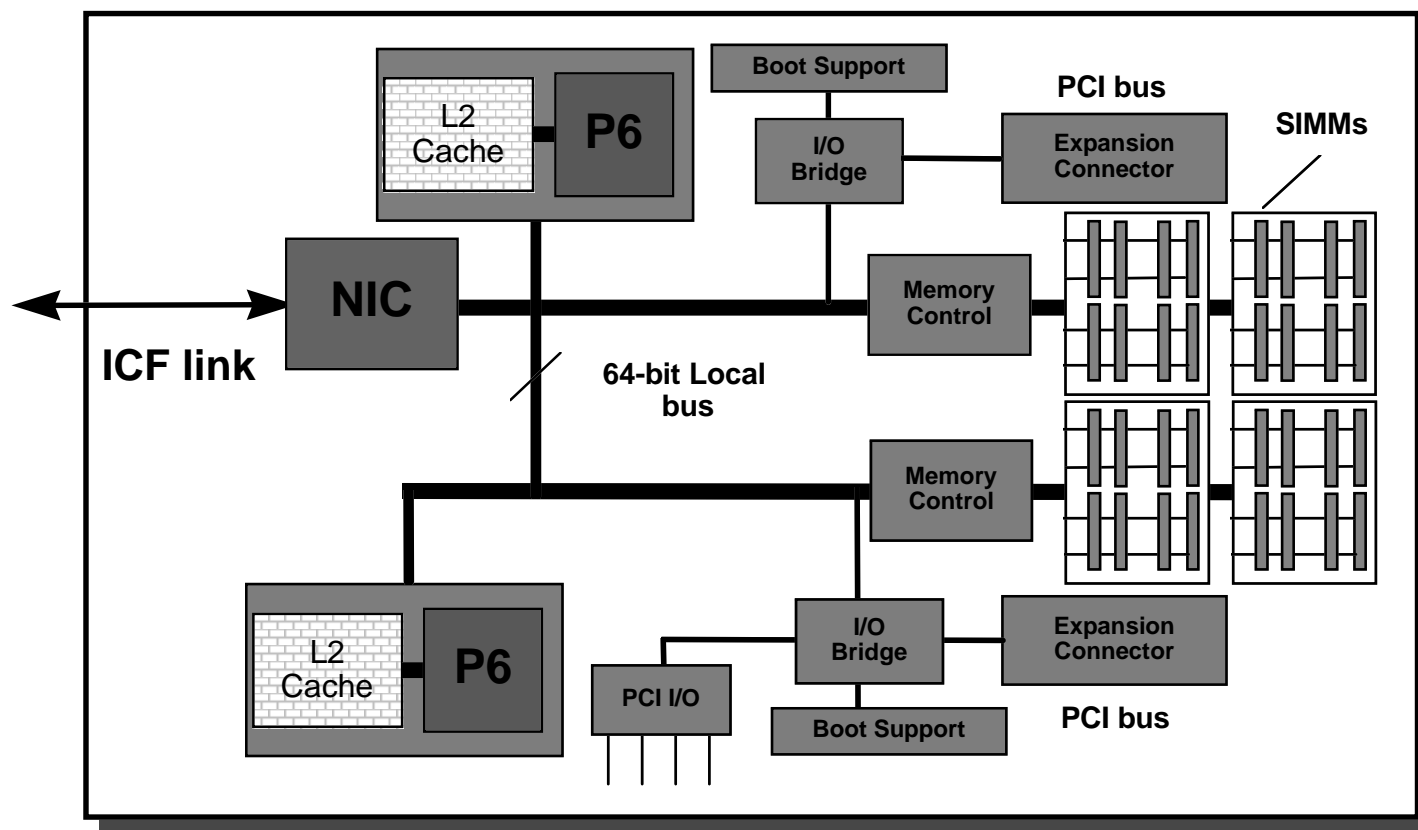
Kestrel Board: Compute & Service Nodes



■ Commodity components

■ Scalability components

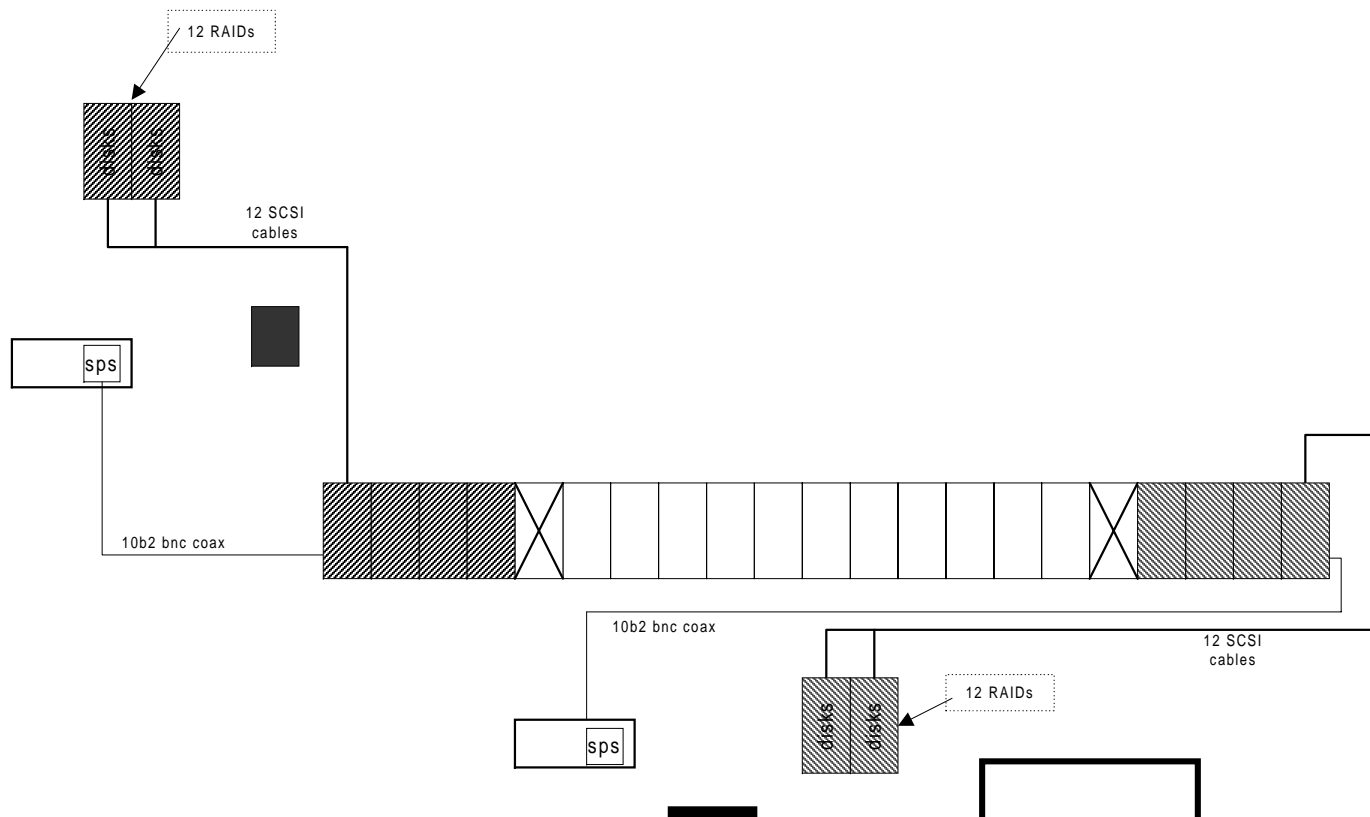
Eagle Board: System & I/O Nodes



■ Commodity components

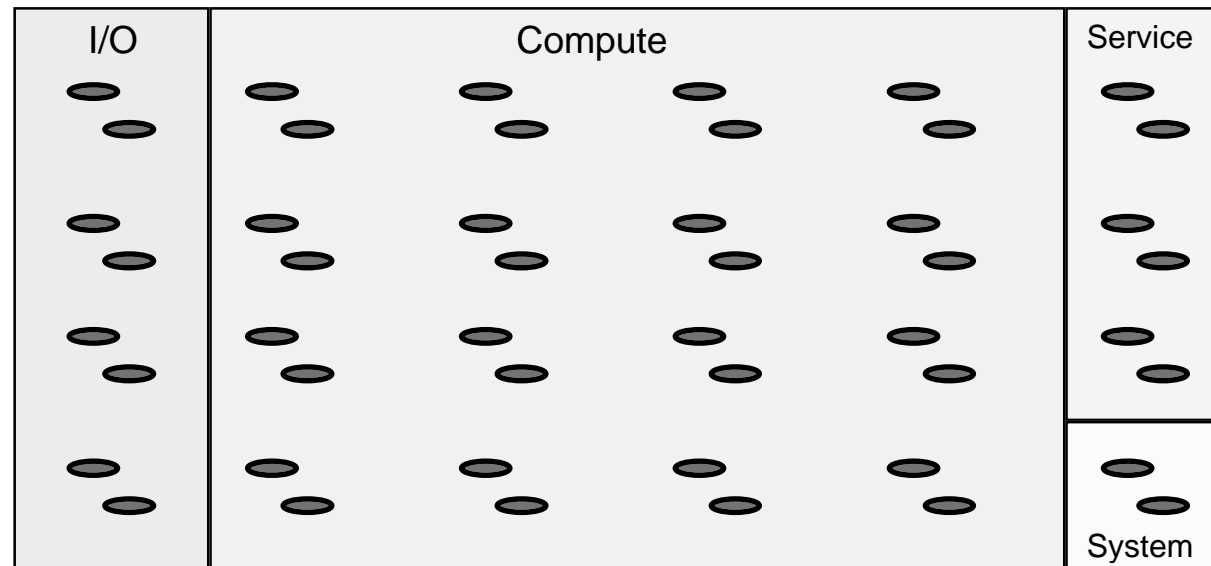
■ Scalability components

Physical Layout



janus applications view

- ◆ 4 partitions:
 - Service partition: Supports interactive users and prog. builds.
 - Compute partition: Where the parallel applications run.
 - I/O partition: I/O to RAIDs, networking, etc.
 - System partition: Booting, admin.



OS(es)

- ◆ Compute Partition: Minimum services required for computation.
 - Cougar: A small and efficient, light Weight OS.
 - Program Loading
 - Memory Management
 - Message Passing Support
 - Signal and Exit handling
 - Run time support for supported languages (C, Fortran, C++, HPF)
 - See <http://www.acl.lanl.gov/~pfay/teraflop/> for PUMA/Cougar references
- ◆ Service/IO partitions: A common, full featured Unix.
 - The “Teraflops Operating System”: A distributed OSF Unix
 - Boot and configuration support
 - system administration
 - user logins
 - user commands and services
 - development tools

Interacting with the Compute Partition

◆ yod -- launches parallel applications

— yod: App Usage: yod [{-D <option>}] [-comm <size>] [-stack <size>] [-heap <size>] [-help] [-proc <mode=(0 single) (1 coproc) (2 assym)>] [-retry] [-sz <size> | -size <size>] [-fyod <num-fyods>] <file> <args>

yod -sz 4 -stack 2M a.out > a.out.lst &

◆ showmesh

```
janus ~/tst 107 > showmesh
```

Current mesh allocation for janus (last booted Wed Dec 31 17:00:00 1969)

```

          1          2          3          4          5
01234567890123456789012345678901234567890123456789
+-----+
0|                                     .....DD|
60|                                     .....EDaa|
120|                                     .....D|
180|                                     .....DF..|
240|                                     .....|
300|                                     .....|
360|                                     .....|
420|                                     .....|
+-----+
```

Legend:

B	boot node	D	OSF disk node (RAID)		
	non-existant node	S	OSF service node		
R	reserved node	H	OSF HiPPI node		
F	other OSF node	A	OSF DAT (digital audio tape) node		
.	free Puma node		free Puma NQS node		
Job ID	User	Base OS	Size	Start	Partition_name/Command line
---	---	---	---	---	---
a 20	pfay	118 PUMA	4	Feb 15 16:07:15	/cougar/bin/yod.real -sz 4 a.out

Environment Setup

- ◆ Server
 - TFLOPS_XDEV=/usr/local/intel/tflop/current
 - Add to path: \$(TFLOPS_XDEV)/tflops/bin.solaris
 - see ~pfay/.cshrc on janus/sasn100 or my teraflop web page
<http://www.acl.lanl.gov/~pfay/teraflop/cshrc.html>
 - cicc, cif77, ciCC
- ◆ Service Node
 - Compilers are in /bin
 - cicc, cif77, ciCC
- ◆ Cross-compilers available on local LAN if purchase license
 - Environment variable will vary.
 - sun4 or solaris only.

How to compile/run hello_world.[fc]

◆ Fortran

```
include 'mpif.h'
integer*4 ierr,num_pe,my_pe
real*8 x
call mpi_init(ierr)
call mpi_comm_size(MPI_COMM_WORLD,num_pe,ierr)
call mpi_comm_rank(MPI_COMM_WORLD,my_pe,ierr)
call mpi_barrier(MPI_COMM_WORLD,ierr)
print *, 'hello from node ', my_pe, ' of ', num_pe
call mpi_finalize(ierr)
call exit
end
```

◆ Compile

cif77 -o mine mine.f -lmpi

◆ C

```
#include <nx.h>
void main()
{
    if (mynode() == 0)
        printf("The number of nodes %d\n",
            numnodes());
}
```

◆ Compile

cicc -o mine mine.f

◆ Execute

%yod -sz 32 mine

%yod -sz 32 -comm 6M -stack 2M progname -f file1 file2 file3

-comm is only for NX message passing

see 'yod -help' for default stack size

Break

"Talk about Blue Mountain"

Programming Model

- ◆ Distributed Memory, MIMD architecture.
- ◆ Explicit Parallelism:
 - Decompose data structures into small chunks and distribute them among the nodes of the machine.
 - On each node, use standard programming languages for local computation.
 - Pass messages between nodes to update distributed data structures.
- ◆ Implicit Parallelism: HPF (not available yet on ASCI-RED)
 - Provide information about which data structures to distribute.
 - Let the compiler manage data distribution and concurrency.

Message Passing

- ◆ Two Message-Passing Libraries
 - MPI:
 - The 1.1 specification.
 - 1-sided communication
 - NX: The native Paragon message passing library.
 - The Sunmos compatibility version of NX will be supported.
- ◆ Both libraries provide a full featured message passing environment
 - Synchronous and Asynchronous communication.
 - Broadcast
 - Global operations (global sum, max, min, etc).

Programming Languages

- ◆ The following languages will be supported
 - Fortran77
 - C
 - C++
 - HPF
 - Fortran90
- ◆ The compilers written by PGI.
- ◆ Libraries
 - libc.a
 - libm.a
 - libkmath.a (blas)
 - libperfmon.a
 - libdbmalloc.a
 - libmpi.a
- ◆ Signals (some)
- ◆ NagF90 being ported now

IO

◆ PFS

- Optimized for large block transfers
 - C: read/write
 - Fortran: cread/cwrite
 - (Note: Sandia Paragon was using fread/fwrite)
- 64 KB stripe sizes currently -- Expect to increase
- singly-striped vs. Multi

◆ UFS

- /scratch
- Standard Network and Unix File Systems

Moving Target--still being configured

Tools

- ◆ Debugging
 - Parallel Scalable Debugger.
 - MQM: Message Queue Manager.
 - Light weight Corefile browser (Ptools).
- ◆ Performance analysis
 - Compiler driven code instrumentation.
 - User callable library for control of NIC and P6 counters.
- ◆ Resource Management
 - MACS: resource accounting.
 - NQS: Tool for scheduling parallel batch jobs.
 - Queues: (Day/Night)*(Full/End-only)*(SNL/LANL/LLNL)
 - interactive during prime time

Performance Tips for the Pentium Pro

- ◆ Improve Branch Prediction (static algorithm)
 - Make forward branches usually not taken
 - Make backward branches usually taken
 - Compiler predicts if tests “true”

```
if <condition> {  
    no performance hit  
} else {  
    performance hit  
}
```
- ◆ Use the cache
- ◆ Avoid floating point to integer conversions
- ◆ Avoid misaligned data (dynamic alloc), usually done by compiler
- ◆ Avoid divides

Reference URL: <http://www.intel.com/design/pro/applnotes>

See App. Note 526

What about the Second Processor?

- ◆ There are three possible modes for the second processor:
 - ignore the second processor.
 - use it as a communication co-processor.
 - use it to run an additional application thread.
- ◆ How to make use of the second processor in the application
 - Math Libraries
 - Compiler directives
 - COP interface from Cougar.
- ◆ math libraries - just link in the right libraries and ??? argument to yod
- ◆ Use the -Mconcur compiler switch and source code directives.
- ◆ The Compiler's functionality will go well beyond the MP-Paragon:
 - Parallel Loops - cyclic, block and with/without barrier
 - Private data directives
 - Parallel Sections
 - Critical Sections
 - Single user Section
 - Barriers

Unique aspects

- ◆ No virtual Memory
- ◆ No Posix threads
- ◆ No sockets from parallel applications; must go through filesystem