CRAY
THE SUPERCOMPUTER COMPANY

# An Introduction to the Cray Systems at HECToR

DEISA Workshop at EPCC

15th September 2010

Cray Centre of Excellence for HECToR

# Overview

- The Cray XT Architecture
- The Environment
- Compilers
- Libraries
- Communications
- Running Applications
- I/O
- Top 10 tips for tuning

# The Cray XT Architecture

# Cray Systems at HECToR

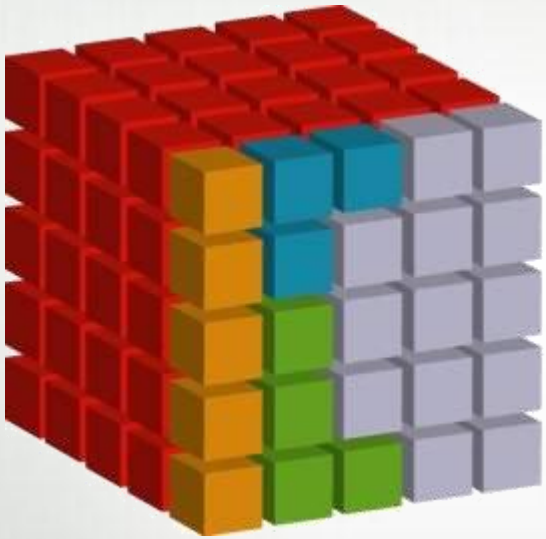
- Cray XT4 Compute Nodes
- 3072 x 2.3 GHz Quad Core Opteron Nodes
- 12,238 Cores
- 8GB per node – 2GB per Core

- Cray X2 Compute Nodes
- 28 x 4 Cray X2 Compute nodes
- 32 GB per node 8GB per Core


- Cray XT6 Compute Nodes
- 1856 x 2.1 GHz Dual 12 Core Opteron Nodes
- 44,544 Cores
- 32GB per node – 1.33 GB per Core

# Scalable Software Architecture: Cray Linux Environment (CLE)

*"Primum non nocere"*



**Compute PE**

**Login PE**

**Network PE**

**System PE**

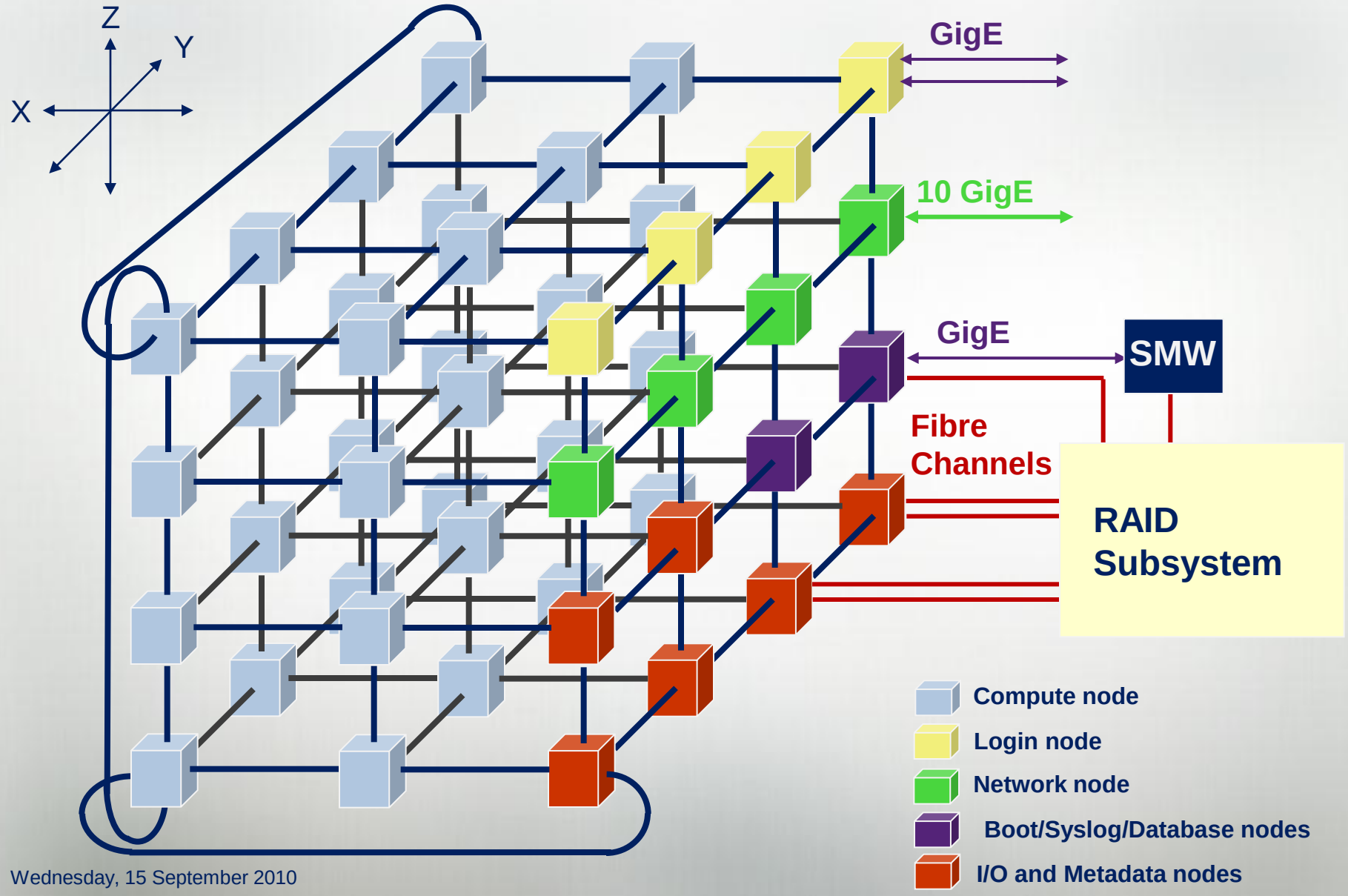
**I/O PE**

**Service Partition**

***Specialized Linux nodes***

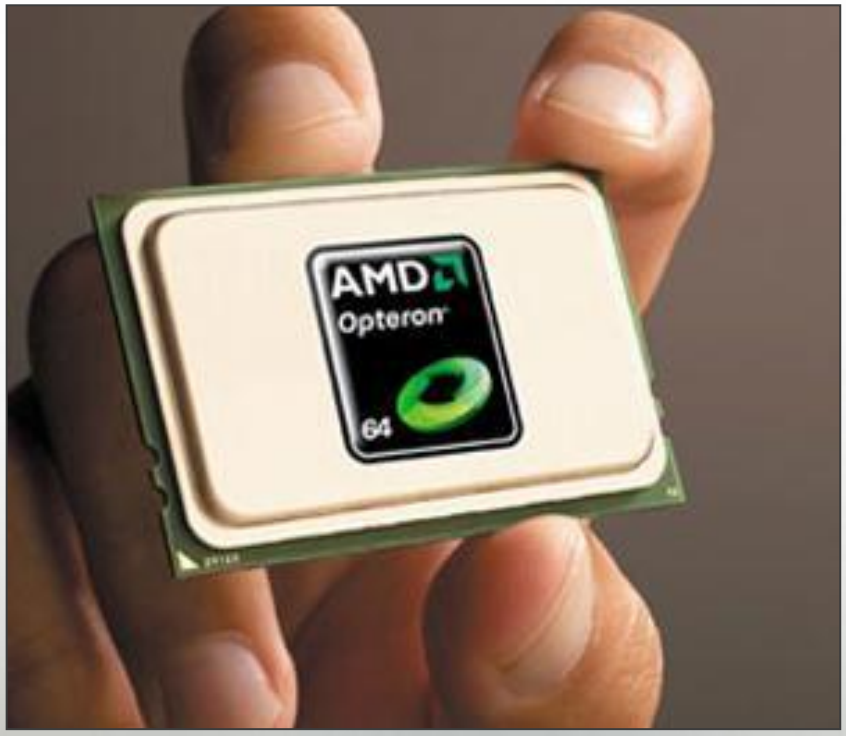
- Microkernel on Compute PEs, full featured Linux on Service PEs.
- Service PEs specialize by function
- Software Architecture eliminates OS "Jitter"
- Software Architecture enables reproducible run times
- Large machines boot in under 30 minutes, including filesystem

# XT System Configuration Example



# XT Processors Properties

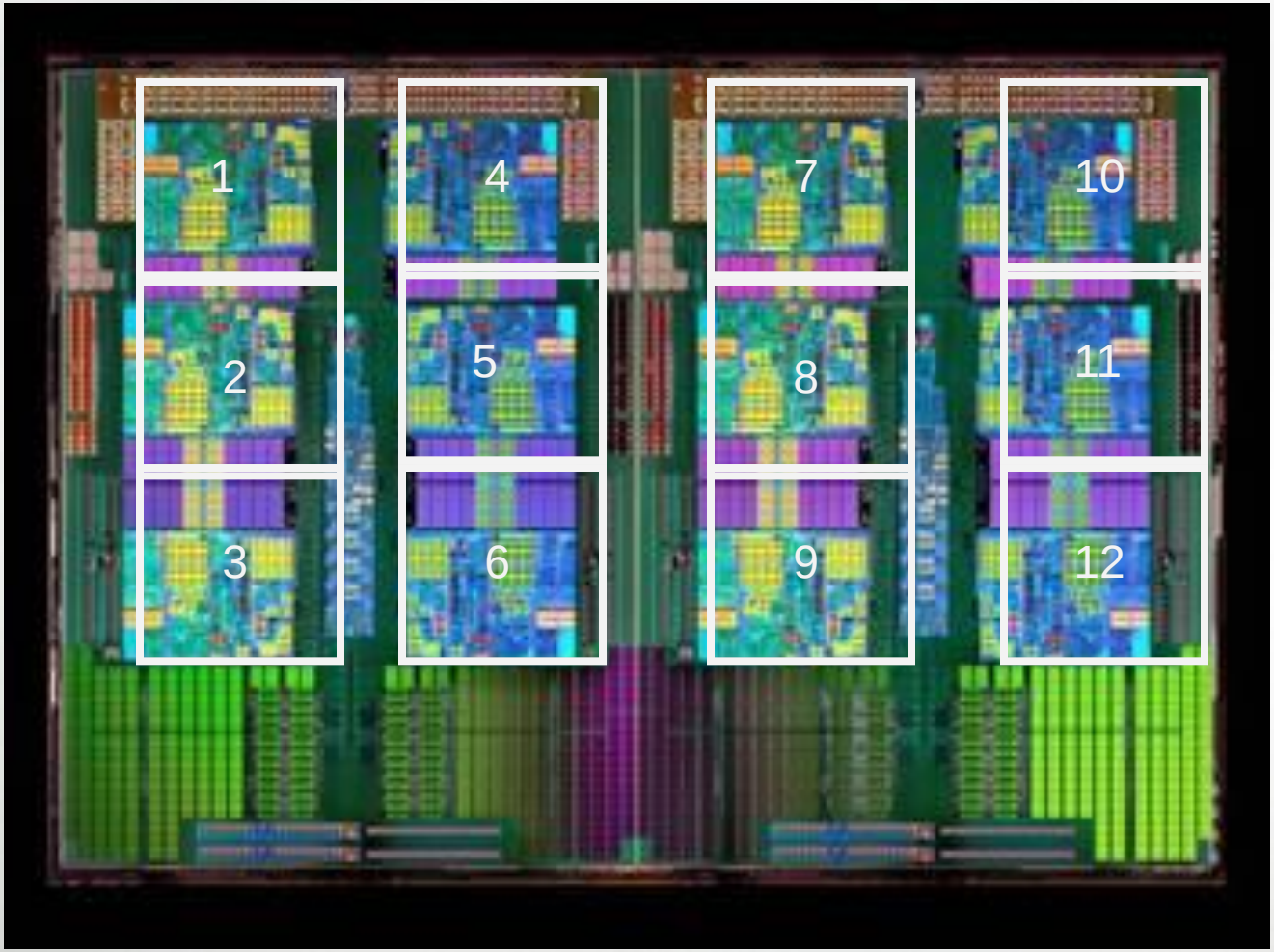
| Processor | Cores | Frequency | Peak (Gflops) | Bandwidth (GB/sec) | Balance (bytes/flop) |
|---|---|---|---|---|---|
| **Barcelona (XT4)** | 4 | 2.3 | 36.8 | 12.8 | 0.34 |
| **MC-12 (XT6)** | 12 | 2.1 | 100.8 | 42.6 | 0.42 |



# x86 64-bit Architecture Evolution

| | 2003 | 2005 | 2007 | 2008 | 2009 | 2010 |
|---|---|---|---|---|---|---|
| | **AMD Opteron™** | **AMD Opteron™** | **"Barcelona"** | **"Shanghai"** | **"Istanbul"** | **"Magny-Cours"** |
| **Mfg. Process** | 130nm SOI | 90nm SOI | 65nm SOI | 45nm SOI | 45nm SOI | 45nm SOI |
| **CPU Core** | K8  | K8  | Greyhound  | Greyhound+  | Greyhound+  | Greyhound+  |
| **L2/L3** | 1MB/0 | 1MB/0 | 512kB/2MB | 512kB/6MB | 512kB/6MB | 512kB/12MB |
| **Hyper Transport™ Technology** | 3x 1.6GT/.s | 3x 1.6GT/.s | 3x 2GT/s | 3x 4.0GT/s | 3x 4.8GT/s | 4x 6.4GT/s |
| **Memory** | 2x DDR1 300 | 2x DDR1 400 | 2x DDR2 667 | 2x DDR2 800 | 2x DDR2 800 | 4x DDR3 1333 |

# AMD Opteron™ 6000 Series Processors – "Magny Cours"




12 cores
1.7-2.2Ghz
105.6Gflops

Power (ACP)
80Watts

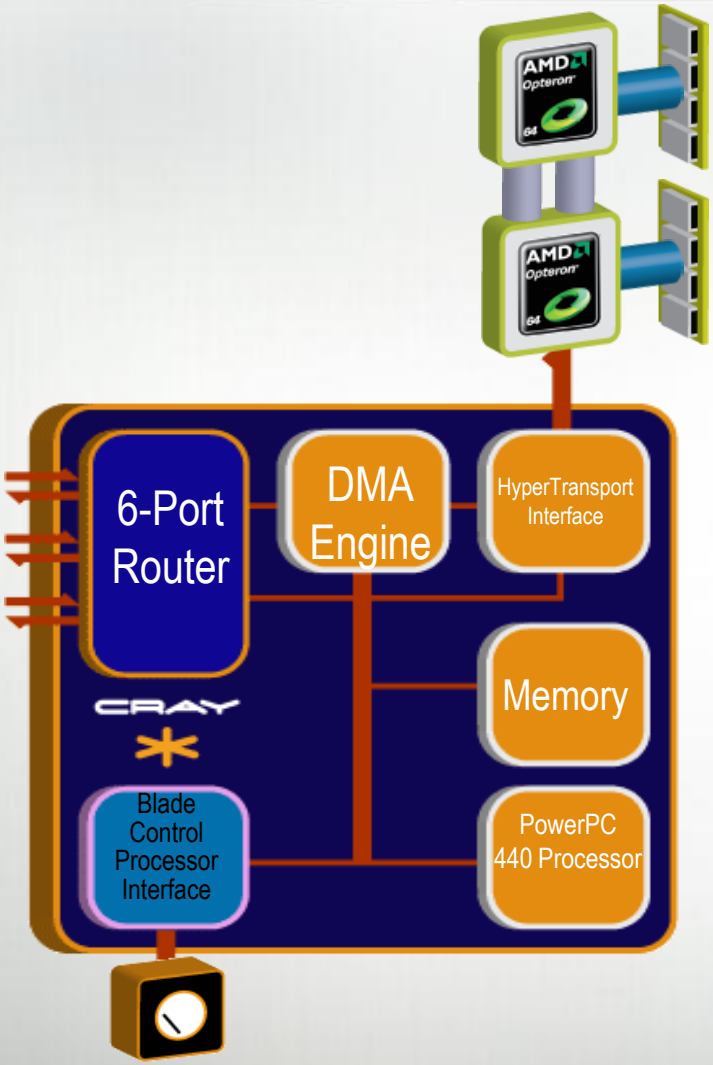
Stream
27.5GB/s

Cache
12x 64KB L1
12x 512KB L2
12MB L3

# XT6 Node Details: 24-core Magny Cours


- 2 Multi-Chip Modules, 4 Opteron Dies
- 8 Channels of DDR3 Bandwidth to 8 DIMMs
- 24 (or 16) Computational Cores, 24 MB of L3 cache
- Dies are fully connected with HT3
- Snoop Filter Feature Allows 4 Die SMP to scale well

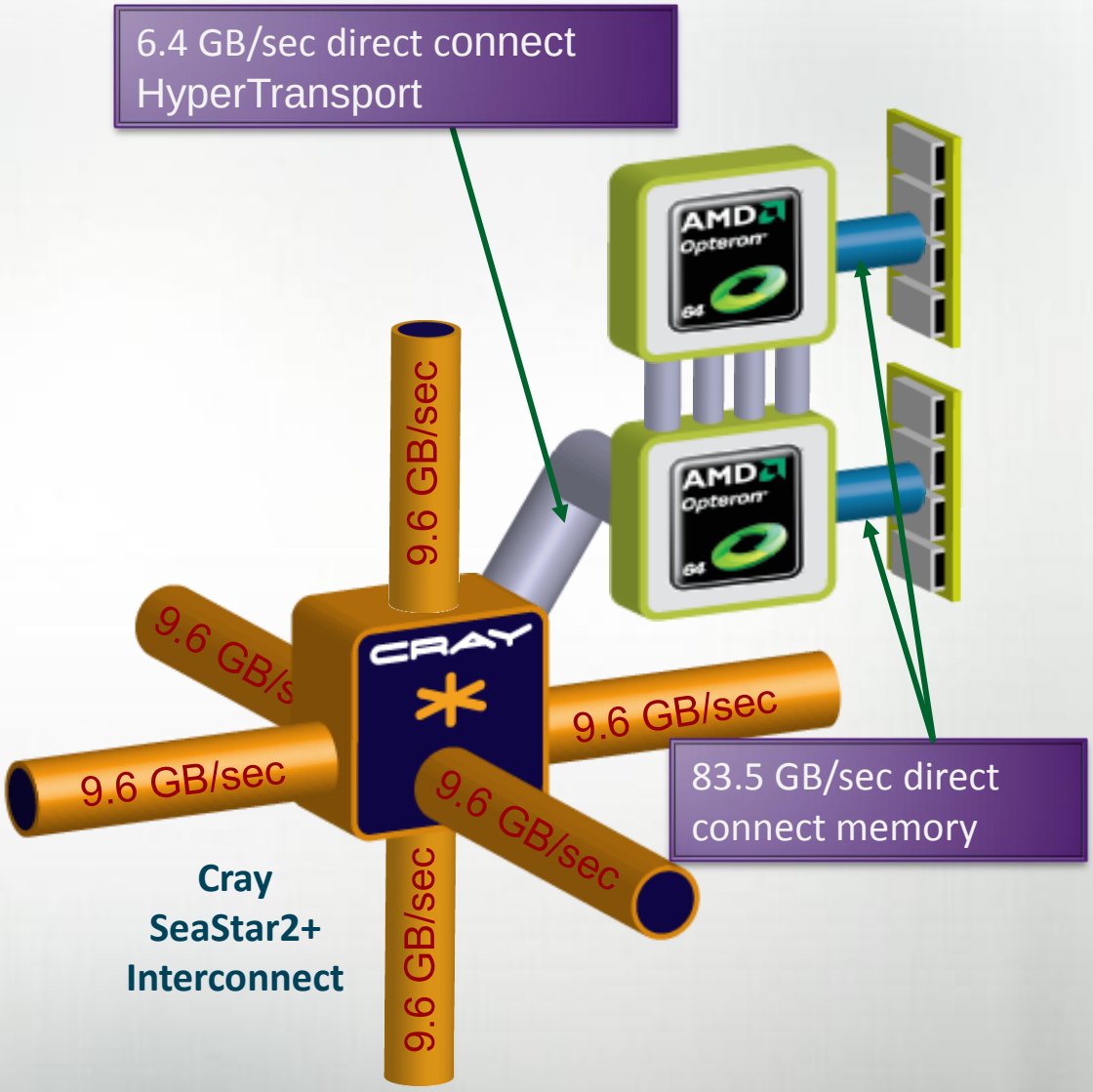
# Cray SeaStar2+ Interconnect




- **Cray XT systems ship with the SeaStar2+ interconnect**
- Custom ASIC
- **Integrated NIC / Router**
- MPI offload engine
- Connectionless Protocol
- Link Level Reliability
- Proven scalability to 225,000 cores

# Cray XT6 (Or XT6m) Node

| Characteristics | |
|---|---|
| Number of Cores | 24 |
| Peak Performance MC-12 (2.2) | 211 Gflops/sec |
| Memory Size | 32GB per node |
| Memory Bandwidth | 83.5 GB/sec |



# The Environment

# Cray XT programming environment

- Four compilers available for x86
  - PGI, Pathscale, Cray and GNU
- Optimized libraries
  - 64 bit AMD Core Math library (ACML)
  - Scilib: Level 1,2,3 of BLAS, LAPACK, FFT, ScaLAPACK, BLACS
  - FFTW, netCDF, HDF5, PETSc
  - MPI-2 message passing library for communication between nodes
  - SHMEM one-sided communication library
- PBSPro batch system
- Performance tools: CrayPat, Apprentice2, PAPI
- Totalview debugger

# Cray XT programming environment is SIMPLE

- Edit and compile MPI program (no need to specify include files or libraries)

```
$ vi mysrc.f90
$ ftn -o myexe mysrc.f90
```

- Edit PBSPro job file (myjob.job)

```
#PBS -N myjob
#PBS -l mppwidth=256
#PBS -j oe
#PBS -A y03
cd $PBS_O_WORKDIR
aprun -n 256 ./myexe
```

- Run the job (output will be myjob.oxxxxx)

```
$ qsub myjob.job
```

# Modifying your environment

- How to access the different compilers and libraries
- **`module`** tool used on Cray XT to handle different versions of packages

  (compiler, tools,...):

  e.g.: **`module load compiler1`**

  e.g.: **`module switch compiler1 compiler2`**

  e.g.: **`module load totalview`**

  e.g.: **`module unload totalview`**

takes care of changing of PATH, MANPATH, LM_LICENSE_FILE,....

- User do not need to set these environment variable in their shell startup files, makefiles,....

# module list

```
xt4tds nid00004:~> module list
Currently Loaded Modulefiles:
  1) modules/3.1.6                          12) Base-opts/2.2.48B
  2) pbs/10.1.0.91350                       13) pgi/10.6.0
  3) xtpe-target-cnl                        14) totalview-support/1.0.6
  4) xt-service/2.2.48B                     15) xt-totalview/8.7.0
  5) xt-os/2.2.48B                          16) xt-libsci/10.4.6
  6) xt-boot/2.2.48B                        17) pmi/1.0-1.0000.7901.22.1.ss
  7) xt-lustre-ss/2.2UP02_PS12_1.6.5        18) xt-mpt/5.0.1
  8) cray/job/1.5.5-0.1_2.0202.19481.53.6   19) xt-pe/2.2.48B
  9) cray/csa/3.0.0-1_2.0202.19602.75.1     20) xt-asyncpe/4.1
 10) cray/account/1.0.0-2.0202.19482.49.3   21) PrgEnv-pgi/2.2.48B
 11) cray/projdb/1.0.0-1.0202.19483.52.1
```

- Current default versions
  - Unicos/lc (CLE) 2.2.48B
  - PGI 10.6.0
  - Libsci 10.4.6
  - PBS 10.1.0.91350

# module show

```
ted@xt4tds nid00004:~> module show pgi
-------------------------------------------------------------------
/opt/modulefiles/pgi/10.6.0:

setenv          PGI_VERSION 10.6
setenv          PGI_VERS_STR 10.6.0
setenv          PGI_PATH /opt/pgi/10.6.0
setenv          PGI /opt/pgi/10.6.0
prepend-path    PGROUPD_LICENSE_FILE /opt/pgi/license.dat
prepend-path    PATH /opt/pgi/10.6.0/linux86-64/10.6/bin
prepend-path    MANPATH /opt/pgi/10.6.0/linux86-64/10.6/man
prepend-path    LD_LIBRARY_PATH /opt/pgi/10.6.0/linux86-64/10.6/lib
prepend-path    LD_LIBRARY_PATH /opt/pgi/10.6.0/linux86-64/10.6/libso
module-whatis   PGI compiler for use on XTs
-------------------------------------------------------------------
```

- Provides details about the module settings and files
  - Environment variables defined
  - Location of files

# module avail

```
ted@xt4tds nid00004:~> module avail

---------------------------------- /opt/cray/modulefiles -----------------------------
atp/1.0.1                    xt-mpt/2.1.56HD              xt-mpt/3.4.1
atp/1.0.2(default)           xt-mpt/2.1.56HDA             xt-mpt/3.4.2
perftools/5.1.0(default)     xt-mpt/2.1.56HDB             xt-mpt/3.5.0
pmi/1.0-1.0000.7628.10.2.ss  xt-mpt/3.0.0                 xt-mpt/3.5.1
pmi/1.0-1.0000.7901.22.1.ss  xt-mpt/3.0.0.8               xt-mpt/4.0.2
stat/1.0.0(default)          xt-mpt/3.0.1                 xt-mpt/4.0.3
trilinos/10.0.0              xt-mpt/3.0.3                 xt-mpt/4.1.0.1
trilinos/10.0.1              xt-mpt/3.0.4                 xt-mpt/4.1.1
trilinos/10.2.0(default)     xt-mpt/3.1.0                 xt-mpt/5.0.0
xt-mpich2/5.0.0              xt-mpt/3.1.1                 xt-mpt/5.0.1(default)
xt-mpich2/5.0.1(default)     xt-mpt/3.1.2                 xt-shmem/5.0.0
xt-mpt/2.1.36HD              xt-mpt/3.2.0                 xt-shmem/5.0.1(default)
xt-mpt/2.1.41HD              xt-mpt/3.3.0
xt-mpt/2.1.50HD              xt-mpt/3.4.0
```

- Provides info about the available modules
  - /opt/modulefiles — Cray provided modules
  - /opt/cray/modulefile — More Cray provided modules
  - /opt/modules/packages — Modules provided by HECToR

# More advanced module commands

- Module system will generally prevent incompatible modules from loading together
- Multiple versions of each module may be available
  - Only one version can be loaded at each time
  - Default version is specified by the system administrator
  - Version specified after backslash
    - `module unload xt-mpt/3.1.2`
    - `module load xt-mpt/5.0.0`
  - Swap command unloads the loads
    - `module swap xt-mpt xt-mpt/5.0.0`
    - `module swap PrgEnv-pgi PrgEnv-cray`
- Always worth checking the latest versions of compilers and libraries when optimising applications

# Compilers

# Cray XT Compilers

- PGI
  - Loaded by default
  - man pages: pgf90, pgcc, pgCC
  - PGI User's Guide (Chapter 2) http://www.pgroup.com/doc/pgiug.pdf
- GNU
  - module swap PrgEnv-pgi PrgEnv-gnu
  - man pages: gfortran, gcc, g++
- Pathscale
  - module swap PrgEnv-pgi PrgEnv-pathscale
  - man pages: eko, pathf90, pathcc
- Cray
  - man pages: crayftn, intro_directives, intro_pragmas
  - craydoc

    http://docs.cray.com/cgi-bin/craydoc.cgi?mode=View;id=S-3901-71;idx=books_search;this_sort=;q=3901;type=books;title=Cray%20Fortran%20Reference%20Manual
  - Free bonus: assign, explain

# Using compiler drivers to create executables

- When the PrgEnv is loaded the compiler drivers are also loaded
  - By default PGI compiler under compiler drivers
  - the compiler drivers also take care of loading appropriate libraries (-lmpich, -lsci, -lacml, -lpapi)

- Available drivers (also for linking of MPI applications):
  - Fortran 90/95 programs: **`ftn`**
  - Fortran 77 programs: **`f77`**
  - C programs: **`cc`**
  - C++ programs: **`CC`**

- Cross compiling environment
  - Compiling on a Linux service node
  - Generating an executable for a CLE compute node
  - Do not use standard compler names (pgf90, mpicc,...) unless you want a Linux executable for the service node
  - Extra Information message in verbose mode (-v)

```
ftn: INFO: linux target is being used
```

# Choosing the right target

- Two types of Opteron on the XTs at HECToR
  - Barcelona – XT4
  - Magny-Cours – XT6
- Load modules to target either compute node type:
  - **`module load xtpe-barcelona`** – Targets XT4
  - **`module load xtpe-mc12`** – Targets XT6
- Affects compiler options and optimised libraries (like libsci)
  - E.g. Routines optimised for the appropriate caches sizes

# Cray XT Compilers Rosetta Stone

| Feature | PGI | Pathscale | Cray |
|---|---|---|---|
| Listing | -Mlist | -LIST:=ON | -ra |
| Diagnostic | -Minfo -Mneginfo | -LNO:simd_verbose=ON | (produced by -ra) |
| Free format | -Mfree | -freeform | -f free |
| Preprocessing | -Mpreprocess | -cpp -ftpp | -eZ -F |
| Suggested Optimization | -fast | -O3 –OPT:Ofast | (default) |
| Aggressive Optimization | -Mipa=fast,inline | -Ofast | -O3 |
| Byte swap | -byteswapio | -byteswapio | -hbyteswapio |
| OpenMP recognition | -mp=nonuma | -mp | (default) |
| Automatic parallelization | -Mconcur | -apo | n.a. (yet) |

Default 64 bit variables - ftn –default64
Dynamic Linking (static default) – ftn –dynamic

# PGI programming environment

- **Overall Options**
  - -Mlist — creates a listing file
  - -Wl,-M — generates a loader map (to stdout)
  - -Minfo — info about optimizations performed
  - -Mneginfo — info on why certain optimizations are not performed
- **Preprocessor Options**
  - -Mpreprocess — runs the preprocessor on Fortran files (default on .F, .F90, or .fpp files)
- **Optimisation Options**
  - -fast — chooses generally optimal flags for the target platform
  - -Mipa=fast,inline — Inter Procedural Analysis
  - -Minline=levels:n — number of levels of inlining

# PGI programming environment

- **Language Options**
  - -Mfree — process Fortran source using free form specifications
  - -Mnomain — useful for using the ftn driver to link programs with the main program written in C or C++ and one or more subroutines written in Fortran
  - -i8, -r8 — treat INTEGER and REAL variables as 64-bit
  - -Mbyteswapio — big-endian files in Fortran; Cray XT is little endian
- **Parallelization Options**
  - -mp=nonuma — recognize OpenMP directives
  - -Mconcur — automatic parallelization

man pages: pgf90, pgcc, pgCC

PGI User's Guide (Chapter 2) http://www.pgroup.com/doc/pgiug.pdf

# GNU programming environment

- Module available

  **module swap PrgEnv-pgi PrgEnv-gnu**

- Default compiler version is 4.4.3

man pages: gfortran, gcc, g++

# Pathscale programming environment

- Module available

  **module swap PrgEnv-pgi PrgEnv-pathscale**

- **Overall Options**
  - -LIST:=ON — creates a listing file
  - -LNO:simd_verbose=ON — info about vectorizations performed
- **Preprocessor Options**
  - -cpp -ftpp — runs the preprocessor on C or FORTRAN files
- **Optimisation Options**
  - -O3 –OPT:Ofast — chooses generally optimal flags
  - -Ofast — aggressive optimization
  - -ipa — Inter Procedural Analysis

# Pathscale programming environment

- **Language Options**
  - -freeform process Fortran source using free form specifications
  - -i8, -r8 treat INTEGER and REAL variables as 64-bit
  - -byteswapio big-endian files in Fortran; XT4 is little endian
- **Parallelization Options**
  - -mp recognize OpenMP directives
  - -apo automatic parallelization

man pages: eko, pathf90, pathcc

# Cray Compiler Environment (CCE): Brief History of Time

- Cray has a long tradition of high performance compilers
  - Vectorization
  - Parallelization
  - Code transformation
  - More…
- Began internal investigation leveraging an open source compiler called LLVM
- Initial results and progress better than expected
- Decided to move forward with Cray X86 compiler
- Initial x86 release, 7.0, in December 2008
- 7.2.5 the latest release (many improvments to PGAS support)

# Cray Opteron Compiler: Current Capabilities

- Fortran, C and C++ support
- Excellent Vectorization
  - Vectorize more loops than other compilers
- OpenMP - 3.0 standard
- PGAS: Functional UPC and CAF available today.
- Excellent Cache optimizations
  - Automatic Blocking
  - Automatic Management of what stays in cache
- Prefetching, Interchange, Fusion, and much more…

# Cray Opteron Compiler: Future Capabilities

- Enhanced C++ Front end
- More Automatic Parallelization
  - Modernized version of Cray X1 streaming capability
  - Interacts with OMP directives
- Optimized PGAS for Gemini
- Improved Vectorization
- Improve Cache optimizations

# Cray compiler flags

- Module available

  **module swap PrgEnv-pgi PrgEnv-cray**

- **Overall Options**
  - -ra creates a listing file with optimization info
- **Preprocessor Options**
  - -eZ runs the preprocessor on Fortran files
  - -F enables macro expansion throughout the source file
- **Optimisation Options**
  - -O2 optimal flags [ enabled by default ]
  - -O3 aggressive optimization
  - -O ipa<n> inlining, n=0-5

# Cray compiler flags

- **Language Options**
  - -f free — process Fortran source using free form specifications
  - -s real64 — treat REAL variables as 64-bit
  - -s integer64 — treat INTEGER variables as 64-bit
  - -hbyteswapio — big-endian files in Fortran; XT4 is little endian
    this is a link time option
- **Parallelization Options**
  - -O omp — Recognize OpenMP directives [ enabled by default ]
  - -O thread<n> — n=0-3, aggressive parallelization, default n=2

man pages: crayftn, intro_directives, intro_pragmas, assign

http://docs.cray.com/cgi-bin/craydoc.cgi?mode=View;id=S-3901-71;idx=books_search;this_sort=;q=3901;type=books;title=Cray%20Fortran%20Reference%20Manual

# Cray programming environment: explain

- Displays the explanation for an error message: compile, run time
- Compiler error

```
ftn-1615 crayftn: WARNING TEST, File = fail.F90, Line = 24, Column = 16
  Procedure "ADD" is defined at line 1 (fail.F90).  Dummy argument "B" is an array argument.
   This argument is scalar.
```

```
>explain ftn-1615
Warning : Procedure "%s" is defined at line %s.  Dummy argument "%s" is an
array argument.  This argument is scalar.

The scope of a global name is the entire compilation, so a global (or
external) name must be defined and referenced consistently throughout the
compilation.  The compiler has found that a reference or definition of this
global name differs with another reference or definition for this name.

The compiler is comparing two definitions or a definition and a reference to
the listed procedure.  If the compiler is comparing two definitions, then the
compiler has found that in one definition, a dummy argument is an array
argument, but the corresponding dummy argument in the second definition is a
scalar argument.  The arguments must be the same.  If the compiler is
comparing a reference with its definition, then it has found an array dummy
argument associated with a scalar actual argument or vice versa.  Again, the
arguments must either both be scalar or both be arrays.  (Note:  In a
reference, an array element is considered a scalar.)
```

# Cray programming environment: explain

- I/O runtime error

```
Open IOSTAT=5016
```

```
rosa> explain lib-5016
An EOF or EOD has been encountered unexpectedly.
The file terminated unexpectedly, perhaps without the proper end-of-file
record or end-of-data control information.  The file specification may be
incorrect, or the file may be corrupted.
Ensure that the file specification is correct.  If the file is corrupted,
create the data again, if possible.
See the man pages for assign(1) and asgcmd(1).
The error class is UNRECOVERABLE (issued by the run time library).
```

# Libraries

# Libsci - Cray Scientific Library

- Libsci provides cray optimised versions of common libraries
  - BLAS – Basic Linear Algebra Subroutines
  - LAPACK – Linear Algebra Routines
  - ScaLAPACK – Parallel Linear Algebra Routines
  - BLACS – Basic Liner Algebra Communication Subprograms
  - IRT – Iterative Refinement Toolkit
  - SuperLU – Sparse Solver Routines
  - CRAFFT – Cray Adaptive Fast Fourier Transform
- **`module load xt-libsci`**
- Load appropriate Opteron module
  - **`xtpe-barcelona, xtpe-mc12`**

# Libsci - Threaded

- Libsci uses OpenMP threads as of **`xt-libsci/10.4.2`** and **`asyncpe/3.9`**
  - Can still be used for serial applications as default **`OMP_NUM_THREADS=1`**
  - Can be used inside parallel regions without spawning additional threads
  - Requires the appropriate **`xtpe-<barcelona|mc12>`** module to be loaded

# Other Cray optimised libraries

- Once loaded all headers and libraries included automatically by **`ftn`**, **`cc`** or **`CC`**
- FFTW - Fastest Fourier Transform in the West
  - Version 2 – **`module load fftw/2.1.5.2`**
  - Version 3 – **`module load fftw/3.2.2.1`**
- PETSc – Portable Extensible Toolkit for Scientific Computation
  - Real – **`module load petsc/3.0.00`**
  - Complex – **`module load petsc-complex/3.0.00`**
- ACML – AMD Core Math Libraries
  - **`module load acml`**
- Trilinos
  - **`module load trilinos`**

# Cray provided libraries

- Netcdf
  - Version 3 – **`module load netcdf/3.6.2`**
  - Version 4 – **`module load netcdf/4.0.1.3`**
  - Parallel – **`module load netcdf-hdf5parallel`**
- Hdf5
  - Serial – **`module load hdf5/1.8.4.1`**
  - Parallel – **`module load hdf5-parallel/1.8.4.1`**

# Communications

# Communications

- Cray XTs support multiple communication paradigms
  - Message passing
    - MPI
  - Single Sided
    - SHMEM
  - Partitioned Global Address Space (PGAS)
    - Coarray Fortran
    - Unified Parallel C (UPC)
  - Shared memory (on node)
    - OpenMP
    - pthreads

# Message Passing Toolkit - features

- Message Passing Toolkit (MPT) based on MPICH-2
- Automatically-tuned default values for MPICH environment variables
  - Several of the MPICH environment variable default values are now dependent on the total number of processes in the job
  - The new default values are displayed setting **`MPICH_ENV_DISPLAY`**
- Optimized collectives
  - Cray version of MPI collective routines are available and enabled by default (**`MPI_Allreduce,MPI_Bcast,MPI_Alltoallv`**)
  - The original MPICH-2 algorithm can be selected with the **`MPICH_COLL_OPT_OFF`** environment variable
- SMP aware global communication
  - MPI_Bcast, MPI_Reduce
- MPI-IO performance improvements for collective buffering on MPI collective writes

# Some MPI environment variables

| Environment Variable | Description |
|---|---|
| **MPICH_MAX_SHORT_MSG_SIZE** | Maximum message size (bytes) to use EAGER protocol where sender sends message header+data to receiver.<br>Default: 2048*(100000/num_ranks )<br>Min: 1024; Max: 128000 |
| **MPICH_UNEX_BUFFER_SIZE** | Global size of the three unexpected message buffers maintained by each process.<br>Default: (num_ranks * 3000)<br>Min: 60M; Max: 260M |
| **MPICH_PTL_UNEX_EVENTS** | Size of the event queue associated with short message buffers. Basically maximum number of short messages that can be outstanding at a given process.<br>Default: max(num_ranks * 2.2, 20480) |

# Some MPI environment variables

| Environment Variable | Description |
|---|---|
| **MPICH_PTL_MATCH_OFF** | If set, disables registration of receive requests with portals. Setting this allows MPI to perform the message matching for the portals device.<br>It may be beneficial to set this variable when an application exhausts portals internal resources and for latency-sensitive applications.<br>Default: Not enabled |
| **MPICH_MSGS_PER_PROC** | Specifies the initial number of internal message headers allocated by MPI. If additional message headers are required during program execution, MPI dynamically allocates more message headers in quantities of **MPICH_MSGS_PER_PROC**.<br>Default: 16384 |

See **man mpi** for more details and configuration options

# MPI-IO Collective Buffering

- MPI-IO performance improvements for collective buffering on MPI collective writes
- This optimization is enabled by default
  - MPI-IO hint romio_cb_write to "enable“
  - Environment variable **`MPICH_MPIIO_CB_ALIGN`** to 2.

- For more info: mpi man page sections:
  - **`MPICH_MPIIO_CB_ALIGN`**
  - **`MPICH_MPIIO_HINTS`** (collective buffering)

- New MPI-IO White Paper
  ftp://ftp.cray.com/pub/pe/download/MPI-IO_White_Paper.pdf

# Running Applications

# Running an application on the Cray XT

- Cray XTs at HECToR operate in Batch mode only
- Queue Management by PBSPro
  - Manages and schedules resources (like the compute nodes)
  - Job control through standard **`qsub`**, **`qstat`**, **`qcancel`**
    - Interprets **`#PBS`** comments in Batch Scripts
- ALPS - Application Level Placement Scheduler
  - Responsible for launching jobs on the compute nodes
- **`aprun`** is the ALPS application launcher
  - It must be used to run application on the XT compute nodes
  - If **`aprun`** is not used, the application is launched on the login node (and likely fails)

# Running an application on the Cray XT4

Remember to set the corresponding PBS option in the job header

- Assuming a XT4 system (4 cores per node)
- Pure MPI application, using all the available cores in a node
  - **nrank** MPI tasks, **nrank** cores, **nrank/4** nodes

```
$ aprun –n <nrank>
```

- Pure MPI application, using only 1 core per node
  - **nrank** MPI tasks, **4*nrank** cores allocated, **nrank** nodes allocated
  - Can be done to give all the memory on the node to the MPI tasks

```
$ aprun –N 1 –n <nrank>
```

- Hybrid MPI/OpenMP application
  - **nrank** MPI tasks, 4 OMP threads each, **4*nrank** cores, **nrank** nodes
  - need to set **OMP_NUM_THREADS**

```
$ export OMP_NUM_THREADS=4
$ aprun –N 1 –d 8 –n <nrank>
```

# Running an application on the Cray XT6

PBS options must allocate all nodes used in their entirity

- Assuming a XT6 system (24 cores per node)
- Pure MPI application, using all the available cores in a node
  - **nrank** MPI tasks, **nrank** cores, **nrank/24** nodes

```
$ aprun –n <nrank>
```

- Underpopulating nodes – Node consists of 4 NUMA nodes
  - Pure MPI application 12 ranks per node => 3 ranks per numa_node

```
$ aprun –n <nrank> -N 12 –S 3
```

- Hybrid MPI/OpenMP application
  - **nrank** tasks, 6 OMP threads each, **6*nrank** cores, **nrank/4** nodes
  - 1 rank per numa node => 6 threads per numa node
  - need to set **OMP_NUM_THREADS**

```
$ export OMP_NUM_THREADS=6
$ aprun –n <nrank> –N 4 –s 1 –d 6
```

# aprun CPU Affinity control

- CPU affinity options enable to bind a PE or thread to a particular CPU or a subset of CPUs on a node
- CNL can dynamically distribute work by allowing PEs and threads to migrate from one CPU to another within a node
- In some cases, moving PEs or threads from CPU to CPU increases cache and translation lookaside buffer (TLB) misses and therefore reduces performance
- **aprun** CPU affinity option:
  - **-cc cpu_list** | keyword
  - suggested settings: **-cc cpu**
  - The cpu keyword (the default) binds each PE to a CPU within the assigned NUMA node
- Pathscale compiler provide its own control of cpu affinity: this should be disabled to avoid interference with ALPS
  - export **PSC_OMP_AFFINITY=FALSE**

# Running an application on the Cray XT - MPMD

- aprun supports MPMD – Multiple Program Multiple Data
- Launching several executables on the same MPI_COMM_WORLD

```
$ aprun -n 128 exe1 : -n 64 exe2 : -n 64 exe3
```

# aprun Memory Affinity control

- Cray XT6 systems use dual-socket 12 core compute nodes
- Remote-NUMA-node memory references, such as a process running on NUMA node 0 accessing NUMA node 1 memory, can adversely affect performance.
- aprun memory affinity options:
  - -S pes_per_numa_node #PEs to allocate per NUMA node
  - -sl list_of_numa_nodes list of NUMA nodes to use [0,1]
  - -sn numa_nodes_per_node #NUMA nodes to consider [1,2]
  - -ss strict memory containment per NUMA node
    a PE can allocate only the memory local to its assigned NUMA node
  - Suggested settings: -ss

# Running a batch application with PBSPro

- The number of required nodes and cores is determined by the parameters specified in the job header

```
#PBS -l mppwidth=256
#PBS -l mppnppn=4
```

- The job is submitted by the qsub command
- At the end of the exection output and error files are returned to submission directory
- PBS environment variable: $PBS_O_WORKDIR

  Set to the directory from which the job has been submitted

# Allocating entire nodes

**`Claim exceeds reservation's node-count`**

- On the XT4 this is usually caused by mismatching aprun and PBS commands
  - The aprun command is attempting to use cores that it hasn‘t been allocated
- On the XT6 this can occur when spreading ranks or threads over numa_nodes
  - Solution always requst the entire node from PBS (charging is on a per nodes basis)
  - Always set PBS
    - `mppnppn=24`
    - `mppwidth=24*ceiling(nranks/ranks_per_node)`

# Other PBSPro parameters

- **`#PBS -N job_name`**

  the job name is used to determine the name of job output and error files
- **`#PBS -l walltime=hh:mm:ss`**

  Maximum job elapsed time

  should be indicated whenever possible: this allows PBS to determine best scheduling startegy
- **`#PBS -j oe`**

  job error and output files are merged in a single file
- **`#PBS -q queue`**

  request execution on a specific queue: usually not needed
- **`#PBS -A account`**

  budget/account the run should be charged to

# Running out of Memory

**`OOM killer terminated this process.`**

- There is no paging on the XT (there are no local disks)
- Applications are limited to the amount of physical memory on the node. This includes
  - All memory allocated by the applications process
  - All memory used by the system libraries (e.g. MPI)
- Applications are killed when they exceed this memory limit
  - 8 GB per node on the XT4 (2GB per core)
  - 32GB per node on the XT6 (1.33GB per core)
- Try running with a smaller problem size or running the same problem over more MPI ranks.

# MPI Multi-core vs MPI Single-core vs Hybrid MPI+OpenMP

All the examples allocate 64 XT4 nodes

**QUAD CORE (most common)**

```
#PBS -N DCjob
#PBS -l mppwidth=256
#PBS -j oe

aprun -n256 myfile.exe
```

**SINGLE CORE**

```
#PBS -N SCjob
#PBS -l mppwidth=64
#PBS -l mppnppn=1
#PBS -j oe

aprun -N1 -n64 myfile.exe
```

**Hybrid MPI + OpenMP**

```
#PBS -N OMPjob
#PBS -l mppwidth=64
#PBS -l mppnppn=1
#PBS -l mppdepth=4
#PBS -j oe

export OMP_NUM_THREADS=4
aprun -N1 -d4 -n64 myfile.exe
```

# Watching a launched job on the Cray XT

- **`xtnodestat`**
  - Shows XT nodes allocation and aprun processes
  - Both interactive and PBS
- **`apstat`**
  - Shows aprun processes status
  - **`apstat`** overview
  - **`apstat -a <id>`** info about all the applications or a specific one
  - **`apstat -n`** info about the status of the nodes
- PBS **`qstat`** command
  - shows batch PBS jobs
  - **`qstat -r`** check running jobs
  - **`qstat -n`** check running and queued jobs
  - **`qstat -s <job_id>`** reports comments about the job

# xtnodestat output

```
     C0-0     C0-1     C1-0     C1-1     C2-0     C2-1     C3-0     C3-1
  n3 aaSadeSe gggSgaaS aaaaaahh SZgZSgZa ggmmmmmm nppppppp qrrssggg wwxxxxxx
  n2 dX ade e ggg gga  aaaaaahh  ggZ gaa ggmmmmmm nppppppp qrrssggg wwxxxxZx
  n1 dd dae e ggg gga  aaaaaahh  ggg gaa ggmmmmmm nnpppppp qrrsssgg wwwxxxxx
c2n0 ddSddeSe gggSggaS aaaaaahh SgggSggZ ggmmmmmm nnpppppp qrrrssgg wwwxxxxx
  n3 bSdadSdd aaSfaaSg aaaaaaaa aaaZaaZg kkgggggg nnnnnnnn qqqqqqqq vvvvvwww
  n2 b cad dd aa faa g aaaaaaaa aaaaaaag kkgggggg onnnnnnn qqqqqqqq vvvvvwww
  n1 b caa dd aa aaa a aaaaaaaa aaaaaZag kkkggggg onnnnnnn qqqqqqqq vvvvvwww
c1n0 aScdaSdd aaSafaSa aaaaaaaa aaaaaaag akkggggg onnnnnnn qqqqqqqq vvvvvwww
  n3 SaabSbba fSfaaSaa aagSaaaS ijjZaaaa Zaaaaaaa mmmmnnoo ppppkkqq ssttgggv
  n2  aab aba e faa aa aag aaa  ijjaaaaa aaaaaaaa mmmmmnno ppppkkqq gstttguv
  n1  bab abb e faa aa aag gaa  hjjaaaaa aaaaaZaa mmmmmnno ppppkkqq gssttgug
c0n0 SaZbSaab eSffaSaa aaaSgaaS Zjjjaaaa aaaaaaaa mmmmmnno pppppkkq gssttgug
    s01234567 01234567 01234567 01234567 01234567 01234567 01234567 01234567
```

```
Job ID     User       Size   Age              command line
--- ------ --------   -----  ---------  --------------------
a   178935 jc590      216    1h36m      cp2k.popt
b   178770 bac        14     5h21m      orca025_LIM2_
c   178855 dxa        3      2h48m      vasp
d   178866 kjelfs     22     2h40m      cp2k.popt
e   178998 jdha       11     0h02m      shelf_ahp1
f   178999 andreww    9      0h00m      castep
g   178910 js615      128    1h55m      neci.x
h   178994 andreww    9      0h11m      castep
i   178958 miammann   2      0h57m      vasp5.k
j   178997 andreww    9      0h03m      castep
k   178938 acc        75     1h35m      orca0083_1784
m   178789 sar00046   43     4h45m      lbe.2b
n   178813 sar00046   43     3h50m      lbe
o   178804 sga        8      4h21m      opa
p   178817 sar00046   43     3h48m      lbe.2b
q   178821 sar00046   43     3h47m      lbe.2b
r   178970 andreww    9      0h42m      castep
s   178973 lois       16     0h40m      N512L70.exe
t   178990 pkr22      15     0h15m      hyd-5.22.LINU
u   178853 dxa        3      2h51m      vasp
v   178859 kjelfs     22     2h44m      cp2k.popt
w   178861 kjelfs     22     2h42m      cp2k.popt
x   178863 kjelfs     22     2h41m      cp2k.popt
y   178981 eyoung     6      0h25m      shelf
```

# The ideal job

```
#PBS -N goodjob
#PBS -l mppwidth=1024
#PBS -l mppnppn=1
#PBS -l mppdepth=4
#PBS -l walltime=01:00:00
#PBS -j oe

cd $PBS_O_WORKDIR

export MPICH_ENV_DISPLAY=1
#export MPICH_PTL_MATCH_OFF=1
#export MPICH_MSGS_PER_PROC=64000

export MALLOC_MMAP_MAX_=0
export MALLOC_TRIM_THRESHOLD_=536870912

export OMP_NUM_THREADS=4
#export PSC_OMP_AFFINITY=FALSE

aprun -N1 -d4 -n 1024 -cc cpu -ss ./goodexe
```

# Documentation

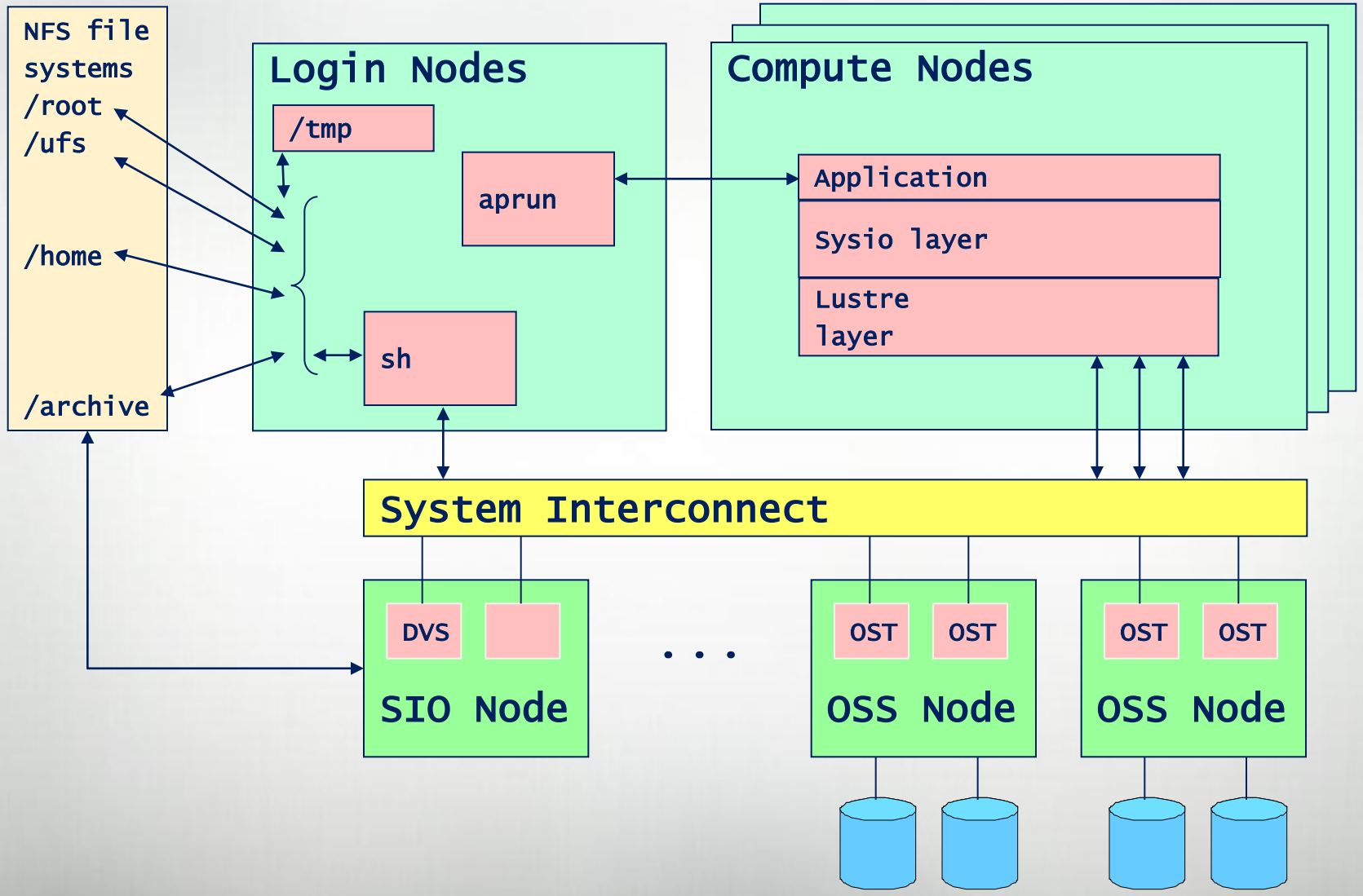
- Cray docs site
  - http://docs.cray.com
- Starting point for Cray XT info
  - http://docs.cray.com/cgi-bin/craydoc.cgi?mode=SiteMap;f=xt3_sitemap
- Twitter
  - http://twitter.com/craydocs

# I/O

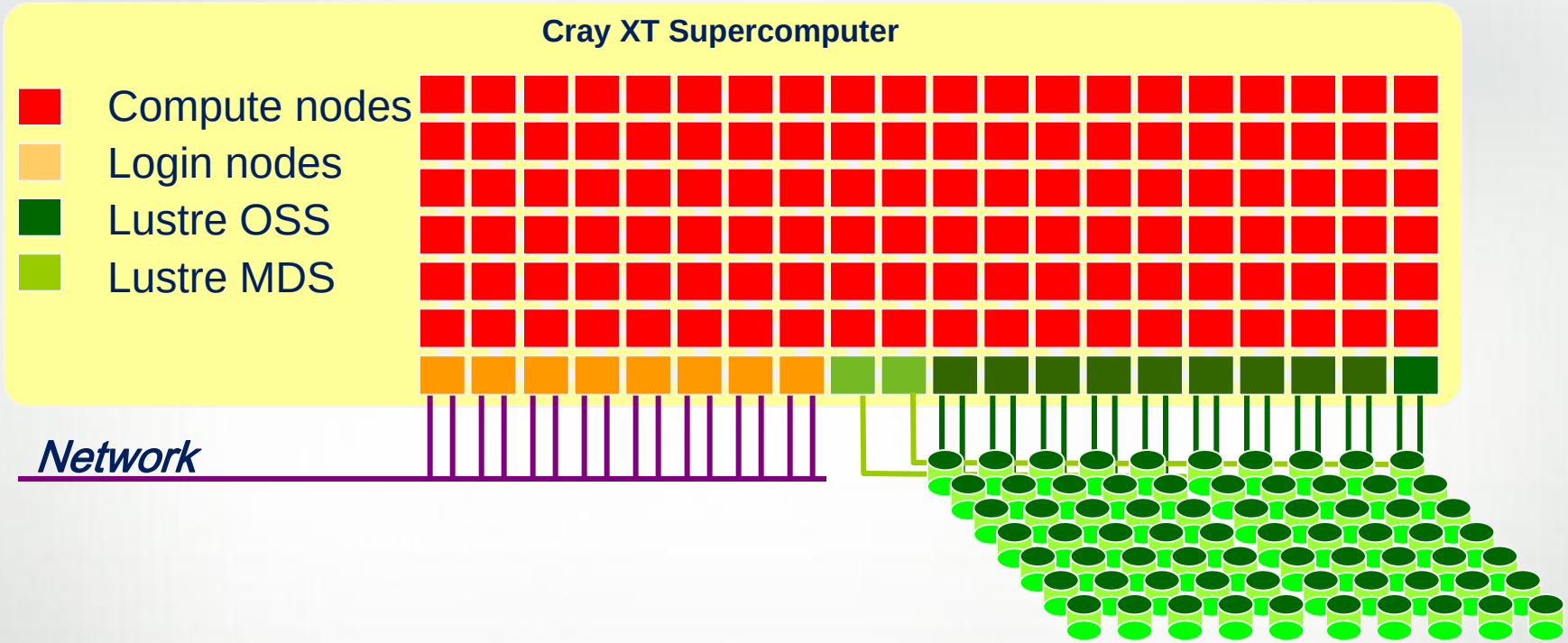
# Cray XT I/O architecture

- All I/O is offloaded to service nodes
- Lustre - /work
  - High performance parallel I/O file system
  - Direct data transfer between compute nodes and files
- DVS
  - Virtualization service
  - Allows compute nodes to access NFS mounted on service node
  - Allows linking with shared libraries ( for more detail see http://docs.cray.com/books/S-0012-22/S-0012-22.pdf)
- No local disks
- /tmp is a memory resident file system, on all login and compute nodes

# Cray XT I/O architecture



# The Storage Environment




Lustre

high performance

parallel filesystem

- A scalable cluster file system for Linux
  - Developed by Cluster File Systems -> Sun -> Oracle.
  - Name derives from “Linux Cluster”
  - The Lustre file system consists of software subsystems, storage, and an associated network
- **MDS** – metadata server
  - Handles information about files and directories
- **OSS** – Object Storage Server
  - The hardware entity
  - The server node
  - Support multiple OSTs
- **OST** – Object Storage Target
  - The software entity
  - This is the software interface to the backend volume

# Lustre File Striping

- Stripes defines the number of OSTs to write the file across
  - Can be set on a per file or directory basis

- CRAY recommends that the default be set to
  - not striping across all OSTs, but
  - set default stripe count of one to four

- But not always the best for application performance.
  As a general rule of thumbs :
  - If you have one large file: stripe over all OSTs
  - If you have a large number of files (~2 times #OSTs): turn off striping

# Lustre lfs command

- lfs is a lustre utility that can be used to create a file with a specific striping pattern, displays file striping patterns, and find file locations
- The most used options are :
  - setstripe
  - getstripe
  - df
- For help execute lfs without any arguments

```
$ lfs
lfs > help
Available commands are:
        setstripe
        find
        getstripe
        check
        ……..
```

# lfs setstripe

- Sets the stripe for a file or a directory
- lfs setstripe <file|dir> <-s size> <-i start> <-c count>
  - stripe size: Number of bytes on each OST (0 filesystem default)
  - stripe start: OST index of first stripe (-1 filesystem default)
  - stripe count: Number of OSTs to stripe over (0 default, -1 all)
- Comments
  - The stripes of a file is given when the file is created. It is not possible to change it afterwards.
  - If needed, use lfs to create an empty file with the stripes you want (like the touch command)
- XT4 /scratch configuration:
  - 72 OSTs
  - Default count: 4
  - Default size: 1MByte

# Lustre striping hints

- For maximum aggregate performance: Keep all OSTs occupied
- Many clients, many files: Don't stripe
  - If number of clients and/or number of files >> number of OSTs: Better to put each object (file) on only a single OST.
- Many clients, one file: Do stripe
  - When multiple processes are all accessing one large file: Better to stripe that single file over all of the available OSTs.
- Some clients, few large files: Do stripe
  - When a few processes access large files in large chunks: Stripe over enough OSTs to keep the OSTs busy on both write and read paths.

# Top 10 Tips for Tuning

# 10. Load the proper xtpe-<arch>

And try every available compiler!

# What is xtpe-arch

```
:) module show xtpe-istanbul
-------------------------------------------------------------------
/opt/c                                    /xtpe-istanbul:

conf
conflict        xtpe-quadcore
conflict        xtpe-shanghai
conflict        xtpe-mc8
conflict        xtpe-mc12
prepend-path    PE_PRODUCT_LIST XTPE_ISTANBUL
setenv          XTPE_ISTANBUL_ENABLED ON
setenv          INTEL_PRE_COMPILE_OPTS  -msse3
setenv          PATHSCALE_PRE_COMPILE_OPTS  -

se
-------------------------------------------------------------------
```

**It'd probably be a really bad idea to load two architectures at once.**

**I should build for the right compute-node architecture.**

**Oh yeah, let's link in the tuned math libraries for this architecture too.**

# Starting Points for Each Compiler

- PGI
  - `-fast -Mipa=fast(,safe)`
  - If you can be flexible with precision, also try `-Mfprelaxed`
  - Compiler feedback: `-Minfo=all -Mneginfo`
  - `man pgf90; man pgcc; man pgCC;` or `pgf90 -help`
- Cray
  - <none, turned on by default>
  - Compiler feedback: `-rm` (Fortran) `-hlist=m` (C)
  - If you know you don't want OpenMP: `-xomp` or `-Othread0`
  - `man crayftn; man craycc ; man crayCC`
- Pathscale
  - `-Ofast` Note: this is a little looser with precision than other compilers
  - Compiler feedback: `-LNO:simd_verbose=ON`
  - `man eko` ("Every Known Optimization")
- GNU
  - `-O2 / -O3`
  - Compiler feedback: `good luck`
  - `man gfortran; man gcc; man g++`

# 9. Try MPICH_PTL_MATCH_OFF For Latency-Sensitive Codes.

# What is MPICH_PTL_MATCH_OFF?

- If set => Disables Portals matching
  - Matching happens on the Opteron
  - Requires extra copy for EAGER protocol
- Reduces MPI_Recv Overhead
  - Helpful for latency-sensitive application
    - Large # of small messages
    - Small message collectives (<1024 bytes)
- When can this be slower?
  - When extra copy time longer than post-to-Portals time
  - Lots of pre-posted Receives can slow it down
  - For medium to larger messages (16k-128k range)

# 8. Set MPICH_FAST_MEMCPY

# What is MPICH_FAST_MEMCPY?

- If set, enables an optimized memcpy routine in MPI. The optimized routine is used for local memory copies in the point-to-point and collective MPI operations.
  - This can help performance of some collectives that send large (256K and greater) messages.
    - Collectives are almost always faster
    - Speedup varies by message size
    - Example: If message sizes are known to be greater than 1 megabyte, then an optimized memcpy can be used that works well for larges sizes, but may not work well for smaller sizes.
  - Default is not enabled (because there are a few cases that experience performance degradation)
- Ex: PHASTA at 2048 processes:  reduction from 262s to 195s

# 7. Pre-post receives

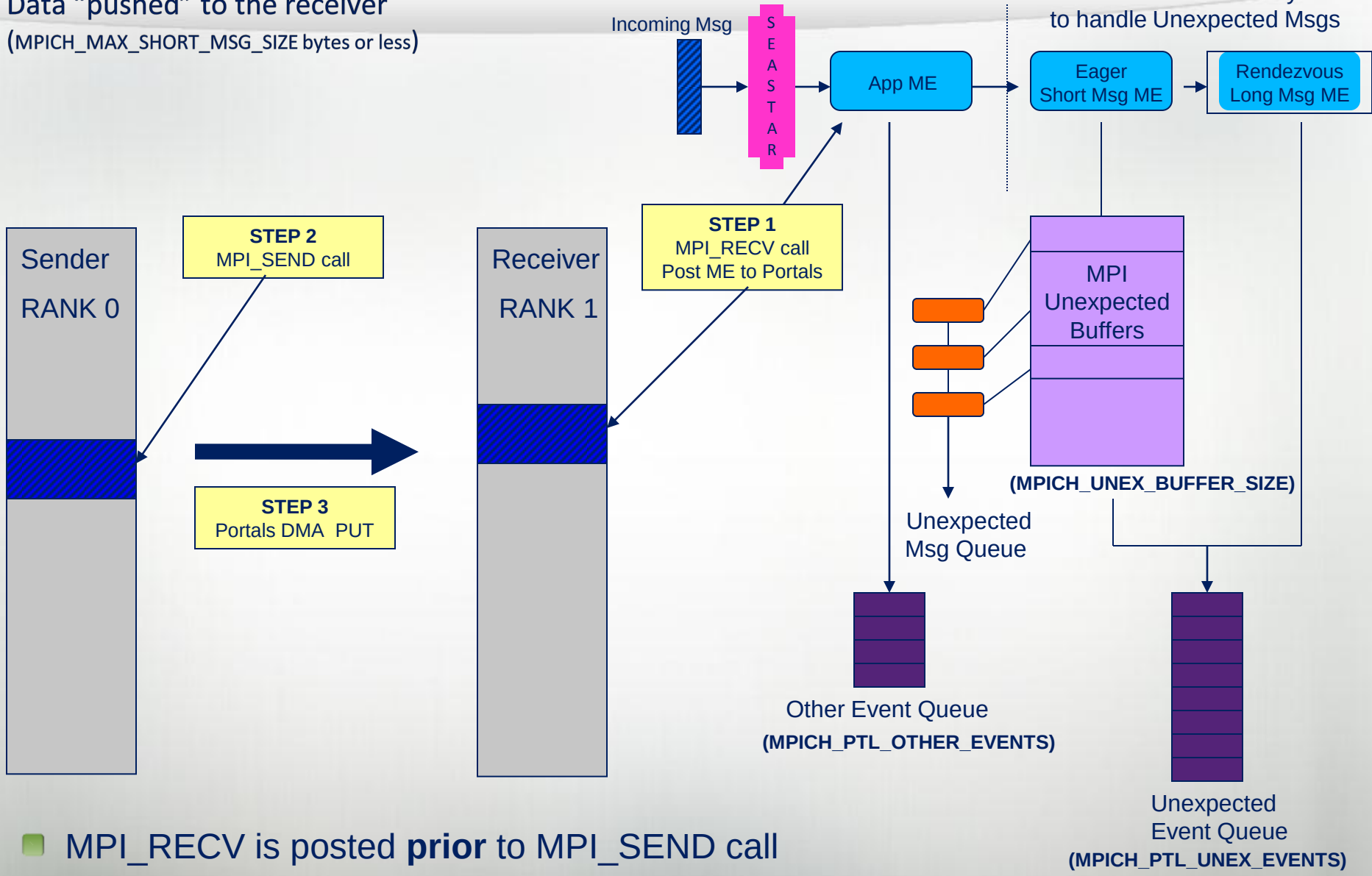
# Cray MPI XT Portals Communications

- Short Message **Eager** Protocol
  - The sending rank "pushes" the message to the receiving rank
  - Used for messages **MPICH_MAX_SHORT_MSG_SIZE** bytes or less
  - Sender assumes that receiver can handle the message
    - Matching receive is posted - or -
    - Has available event queue entries (**MPICH_PTL_UNEX_EVENTS**) and buffer space (**MPICH_UNEX_BUFFER_SIZE**) to store the message
- Long Message **Rendezvous** Protocol
  - Messages are "pulled" by the receiving rank
  - Used for messages greater than **MPICH_MAX_SHORT_MSG_SIZE** bytes
  - Sender sends small header packet with information for the receiver to pull over the data
  - Data is sent only after matching receive is posted by receiving rank

# MPT Eager Protocol

## Data "pushed" to the receiver

(MPICH_MAX_SHORT_MSG_SIZE bytes or less)

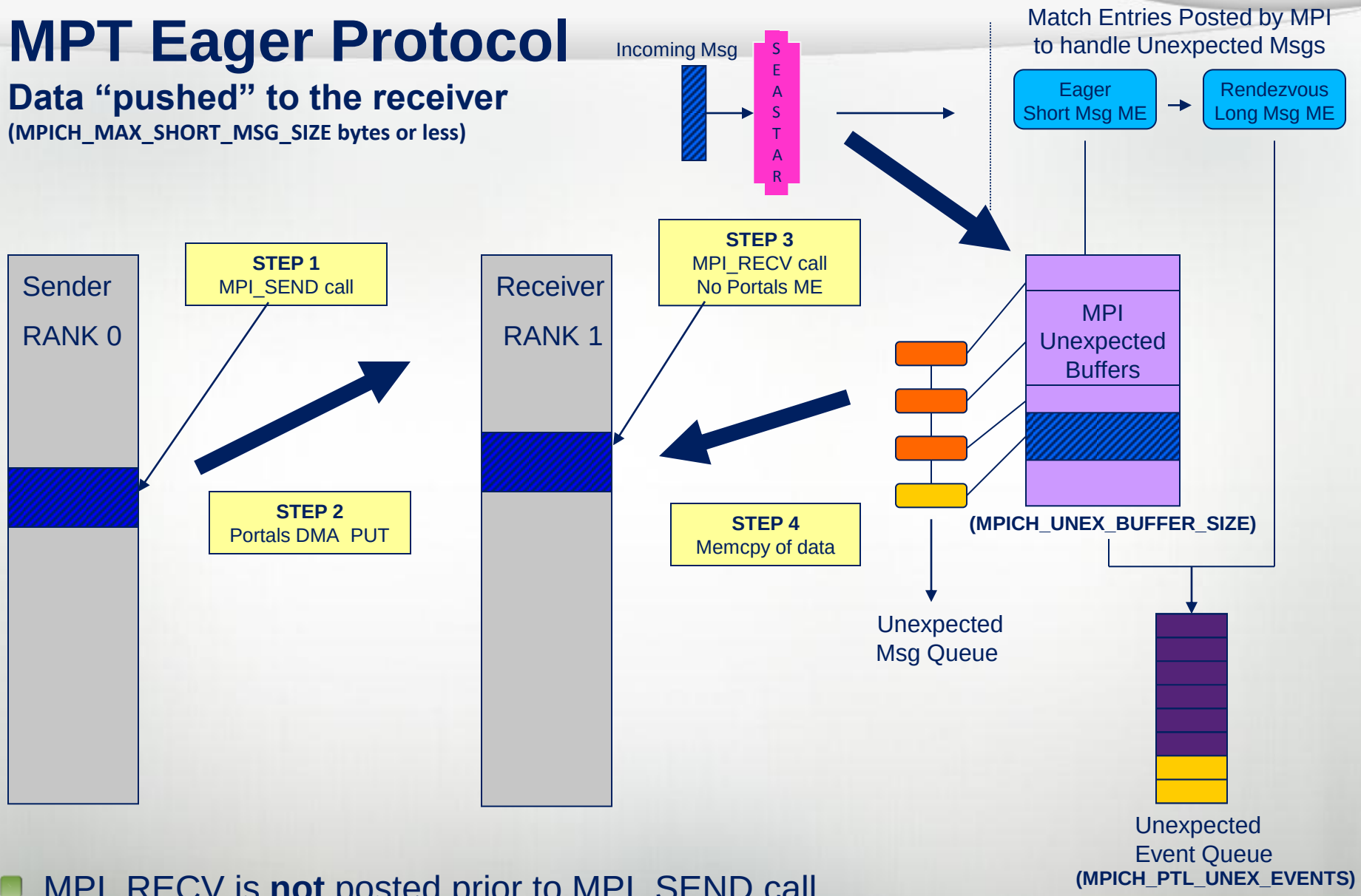



- MPI_RECV is posted **prior** to MPI_SEND call

# MPT Eager Protocol

## Data "pushed" to the receiver

**(MPICH_MAX_SHORT_MSG_SIZE bytes or less)**

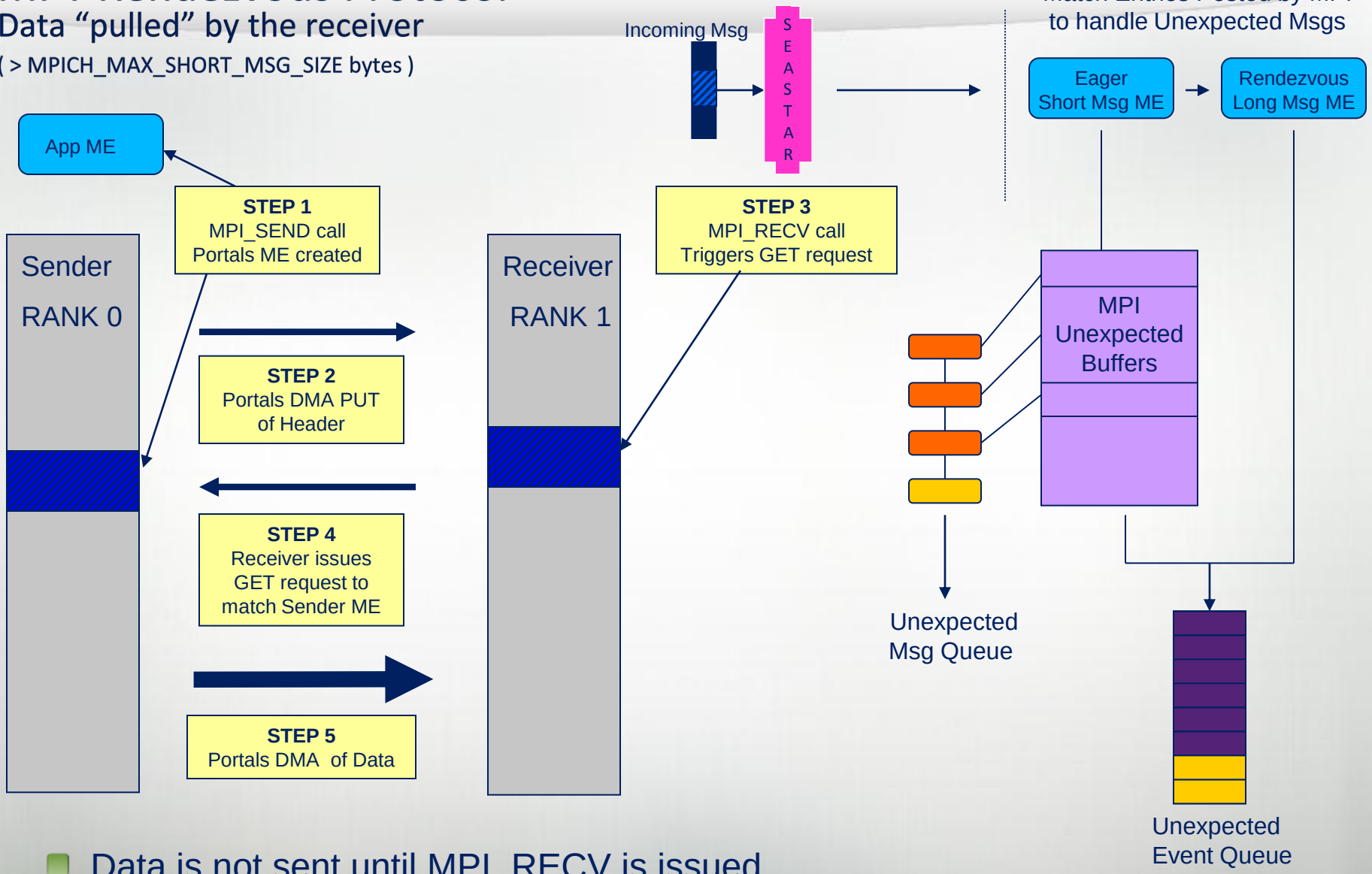



- MPI_RECV is **not** posted prior to MPI_SEND call

# MPT Rendezvous Protocol

## Data "pulled" by the receiver

( > MPICH_MAX_SHORT_MSG_SIZE bytes )




- Data is not sent until MPI_RECV is issued

# 6. Tweak MPICH_MAX_SHORT_MSG_SIZE

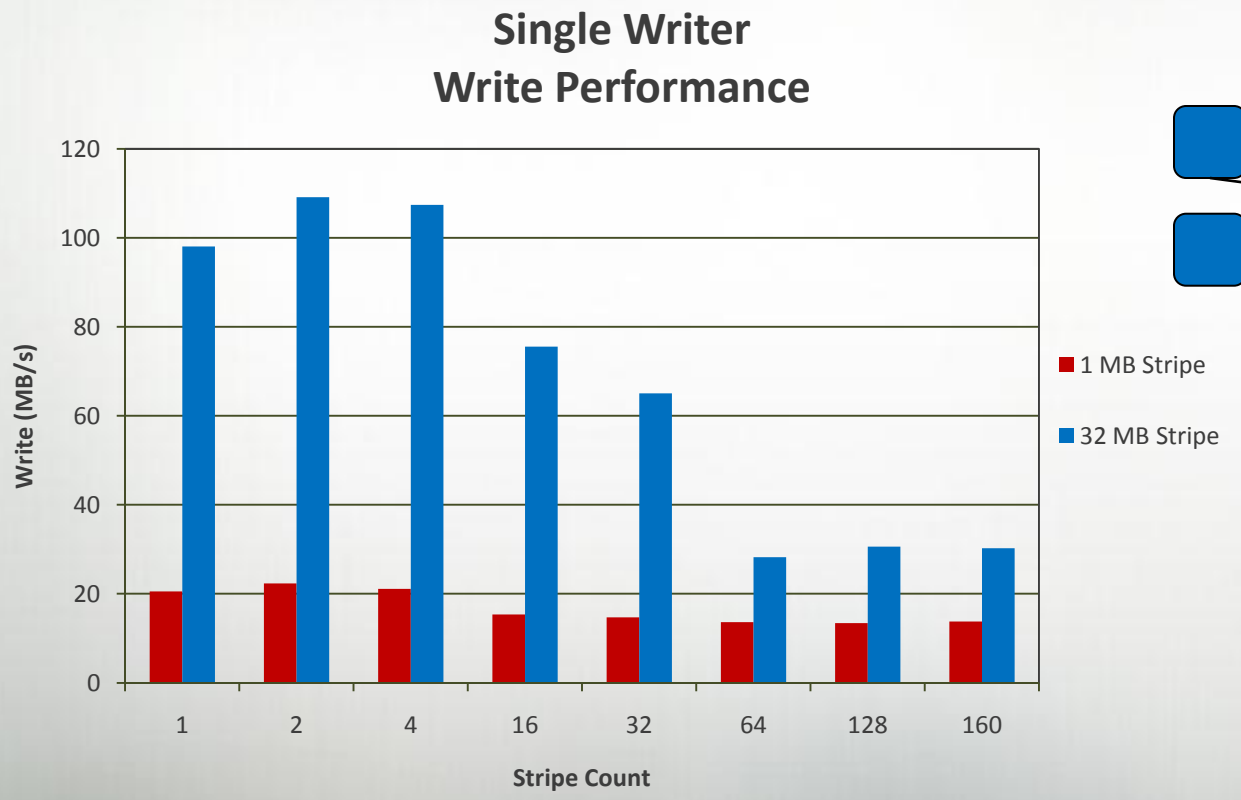
# Adjusting MPICH_MAX_SHORT_MSG_SIZE

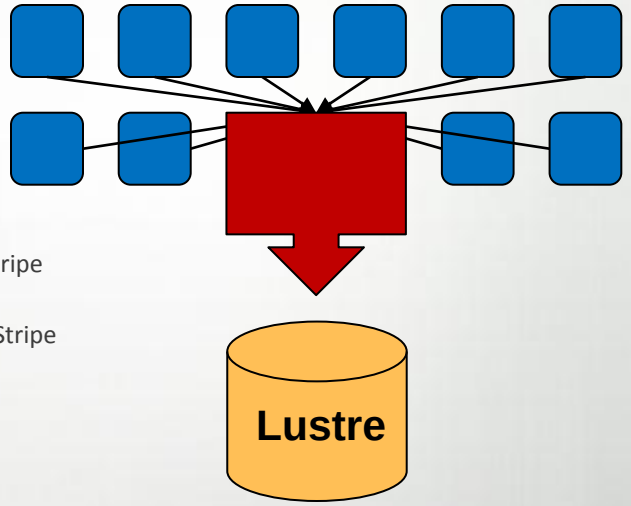
- Controls message sending protocol
  - Message sizes <= MSG_SIZE: Use EAGER
  - Message sizes > MSG_SIZE: Use RENDEZVOUS
  - Increasing this variable may require that MPICH_UNEX_BUFFER_SIZE be increased
- Increase MPICH_MAX_SHORT_MSG_SIZE if App sends large messages and receives are pre-posted
  - Can reduce messaging overhead via EAGER protocol
  - Can reduce network contention
  - Can speed up the application
- Decrease MPICH_MAX_SHORT_MSG_SIZE if:
  - App sends lots of smaller messages and receives not pre-posted, exhausting unexpected buffer space

# 5. Stripe lustre directories.

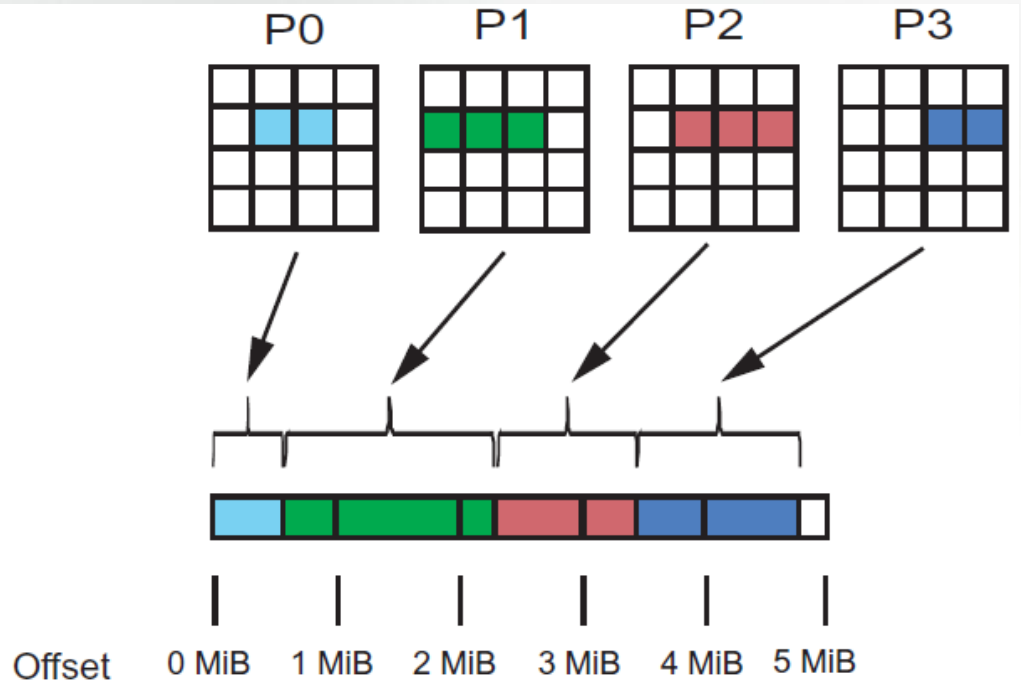
# Single writer performance and Lustre

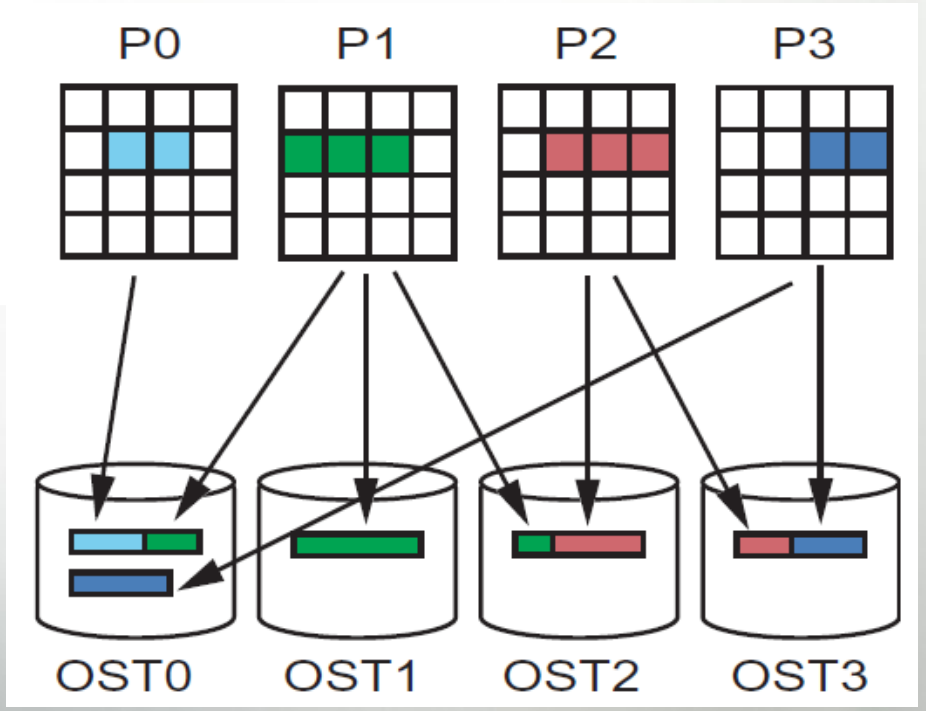
- 32 MB per OST (32 MB – 5 GB) and 32 MB Transfer Size
  - Unable to take advantage of file system parallelism
  - Access to multiple disks adds overhead which hurts performance



# File Striping: Physical and Logical Views



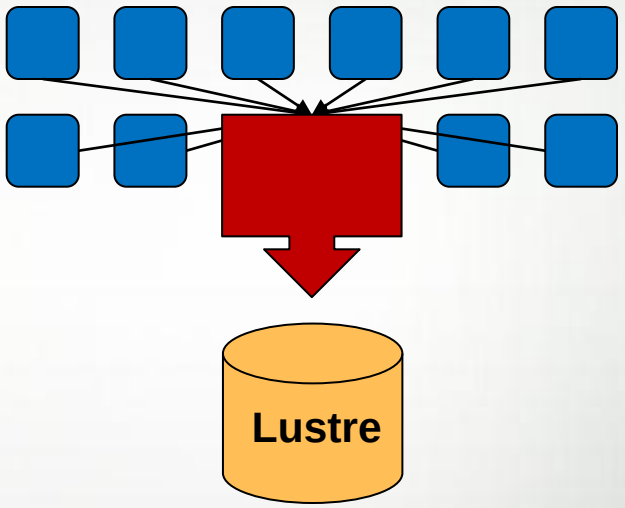



# Lustre: Important Information

- Use the lfs command, libLUT, or MPIIO hints to adjust your stripe count and possibly size
  - lfs setstripe -c -1 -s 4M <file or directory> (160 OSTs, 4MB stripe)
  - lfs setstripe -c 1 -s 16M <file or directory> (1 OST, 16M stripe)
  - export MPICH_MPIIO_HINTS='*: striping_factor=160'
- Files inherit striping information from the parent directory, this **cannot** be changed once the file is written
  - Set the striping before copying in files

# Standard Output and Error

- Standard Output and Error streams are effectively serial I/O.
- All STDIN, STDOUT, and STDERR I/O serialize through aprun
- Disable debugging messages when running in production mode.
  - “Hello, I’m task 32000!”
  - “Task 64000, made it through loop.”



# 4. Tune malloc.

# GNU Malloc

- GNU malloc library
  - malloc, calloc, realloc, free calls
    - Fortran dynamic variables
- Malloc library system calls
  - Mmap, munmap =>for larger allocations
  - Brk, sbrk => increase/decrease heap
- Malloc library optimized for low system memory use
  - Can result in system calls/minor page faults

# Improving GNU Malloc

- Detecting “bad” malloc behavior
  - Profile data => “excessive system time”
- Correcting “bad” malloc behavior
  - Eliminate mmap use by malloc
  - Increase threshold to release heap memory
- Use environment variables to alter malloc
  - MALLOC_MMAP_MAX_ = 0
  - MALLOC_TRIM_THRESHOLD_ = 536870912
- Possible downsides
  - Heap fragmentation
  - User process may call mmap directly
  - User process may launch other processes
- PGI’s –Msmartalloc does something similar for you at compile time

# 3. Touch your memory to improve locality

# Memory Allocation: Make it local

- Linux has a “first touch policy” for memory allocation
  - *alloc functions don’t actually allocate your memory
  - Memory gets allocated when “touched”
- Problem: A code can allocate more memory than available
  - Linux assumed “swap space,” we don’t have any
  - Applications won’t fail from over-allocation until the memory is finally touched
- Problem: Memory will be put on the core of the “touching” thread
  - Only a problem if thread 0 allocates all memory for a node
- Solution: Always initialize your memory immediately after allocating it
  - If you over-allocate, it will fail immediately, rather than a strange place in your code
  - If every thread touches its own memory, it will be allocated on the proper socket

# 2. Try different MPI Rank Orders

# Rank Placement

- The default ordering can be changed using the following environment variable:

  MPICH_RANK_REORDER_METHOD

- These are the different values that you can set it to:

  0: Round-robin placement – Sequential ranks are placed on the next node in the list. Placement starts over with the first node upon reaching the end of the list.

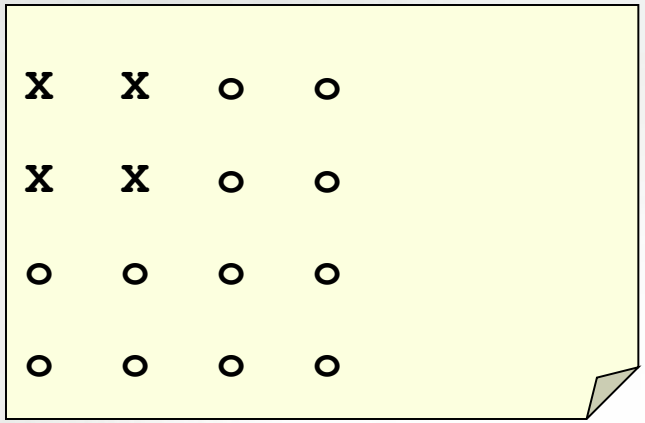
  1: SMP-style placement – Sequential ranks fill up each node before moving to the next.

  2: Folded rank placement – Similar to round-robin placement except that each pass over the node list is in the opposite direction of the previous pass.

  3: Custom ordering. The ordering is specified in a file named MPICH_RANK_ORDER.

- When is this useful?
  - Point-to-point communication consumes a significant fraction of program time and a load imbalance detected
  - Also shown to help for collectives (alltoall) on subcommunicators (GYRO)
  - Spread out IO across nodes (POP)

# Rank order choices




- Nodes marked X heavily use a shared resource
- If memory bandwidth, scatter the X's
- If network bandwidth to others, again scatter
- If network bandwidth among themselves, concentrate

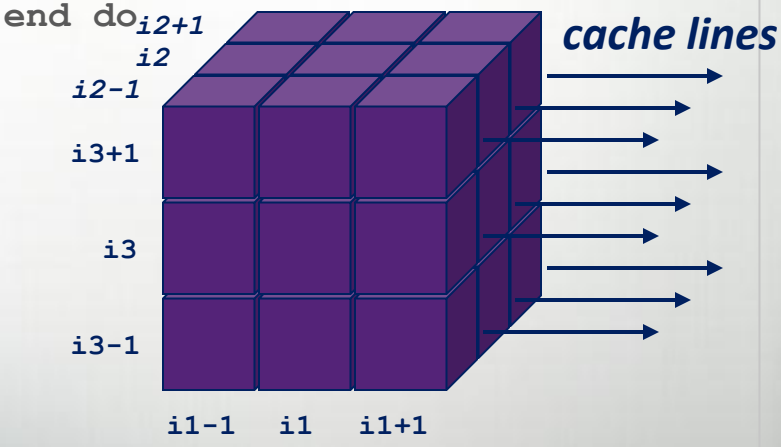
- Check out pat_report, grid_order, and mgrid_order for generating custom rank orders based on:
  - Measured data
  - Communication patterns
  - Data decomposition

# 1. Optimize for Cache.

# A Real-Life Example: NPB MG

- Multigrid PDE solver
- Class D, 64 MPI ranks
  - Global grid is 1024 × 1024 × 1024
  - Local grid is 258 × 258 × 258
- Two similar loop nests account for >50% of run time
- 27-point 3D stencil
  - There is good data reuse along leading dimension, even without blocking

```
do i3 = 2, 257
   do i2 = 2, 257
      do i1 = 2, 257
!        update u(i1,i2,i3)
!        using 27-point stencil
      end do
   end do
end do
```



# I'm Doing It Wrong

- Block the inner two loops
- Creates blocks extending along **i3** direction

```
do I2BLOCK = 2, 257, BS2
   do I1BLOCK = 2, 257, BS1
      do i3 = 2, 257
         do i2 = I2BLOCK,                    &
                 min(I2BLOCK+BS2-1, 257)
            do i1 = I1BLOCK,                 &
                    min(I1BLOCK+BS1-1, 257)
!              update u(i1,i2,i3)
!              using 27-point stencil
            end do
         end do
      end do
   end do
end do
```

| Block size | Mop/s/process |
|---|---|
| unblocked | 531.50 |
| 16 × 16 | 279.89 |
| 22 × 22 | 321.26 |
| 28 × 28 | 358.96 |
| 34 × 34 | 385.33 |
| 40 × 40 | 408.53 |
| 46 × 46 | 443.94 |
| 52 × 52 | 468.58 |
| 58 × 58 | 470.32 |
| 64 × 64 | 512.03 |
| 70 × 70 | 506.92 |

# Improved Version

- Block the outer two loops
- Preserves spatial locality along **i1** direction

```
do I3BLOCK = 2, 257, BS3
   do I2BLOCK = 2, 257, BS2
      do i3 = I3BLOCK,                    &
               min(I3BLOCK+BS3-1, 257)
         do i2 = I2BLOCK,                 &
                  min(I2BLOCK+BS2-1, 257)
            do i1 = 2, 257
!              update u(i1,i2,i3)
!              using 27-point stencil
            end do
         end do
      end do
   end do
end do
```

| Block size | Mop/s/process |
|---|---|
| unblocked | 531.50 |
| 16 × 16 | 674.76 |
| 22 × 22 | 680.16 |
| 28 × 28 | 688.64 |
| 34 × 34 | 683.84 |
| 40 × 40 | 698.47 |
| 46 × 46 | 689.14 |
| 52 × 52 | 706.62 |
| 58 × 58 | 692.57 |
| 64 × 64 | 703.40 |
| 70 × 70 | 693.87 |


CRAY
THE SUPERCOMPUTER COMPANY

# Using configure script on CLE

- Compiling on Linux, generating code for compute nodes => cross-compiling
- Setting variables:

```
setenv CC cc
setenv CXX CC
setenv F77 ftn
setenv F90 ftn
```

- Configure <options>
  - cross-compile to prevent execution of CLE executable on Linux node

    `--host=x86_64-unknown-linux-gnu`
  - build only statically linked executables

    `--enable-static`