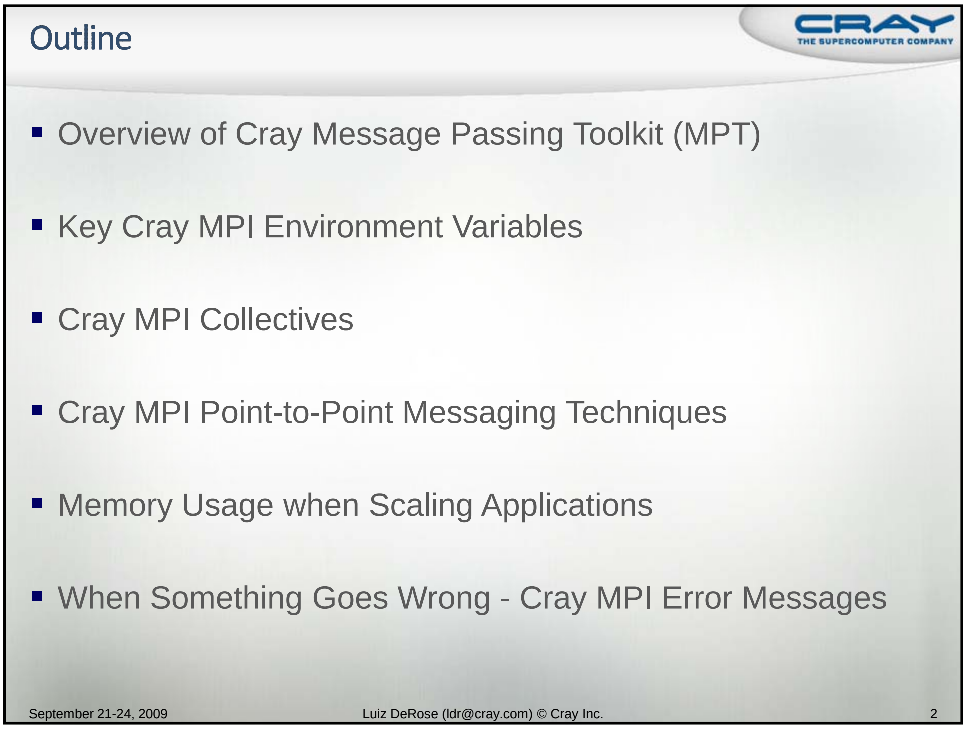
How to Make the Best Use of the Cray MPI on the Cray XT System

Luiz DeRose
Programming Environments Director
Cray Inc.
ldr@cray.com

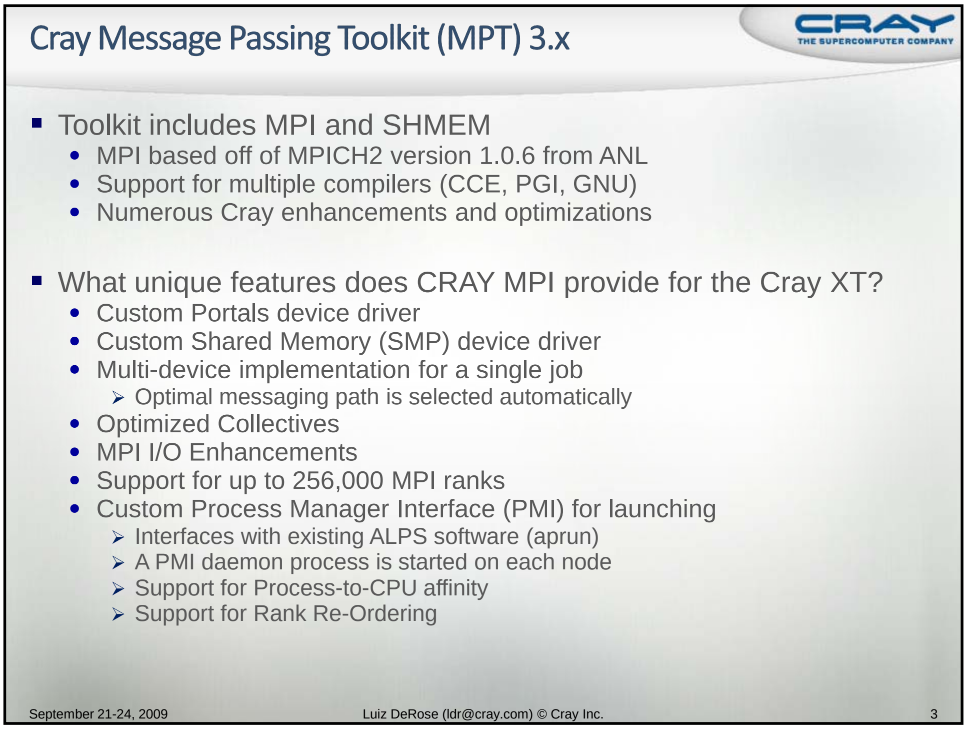
CSC, Finland

September 21-24, 2009



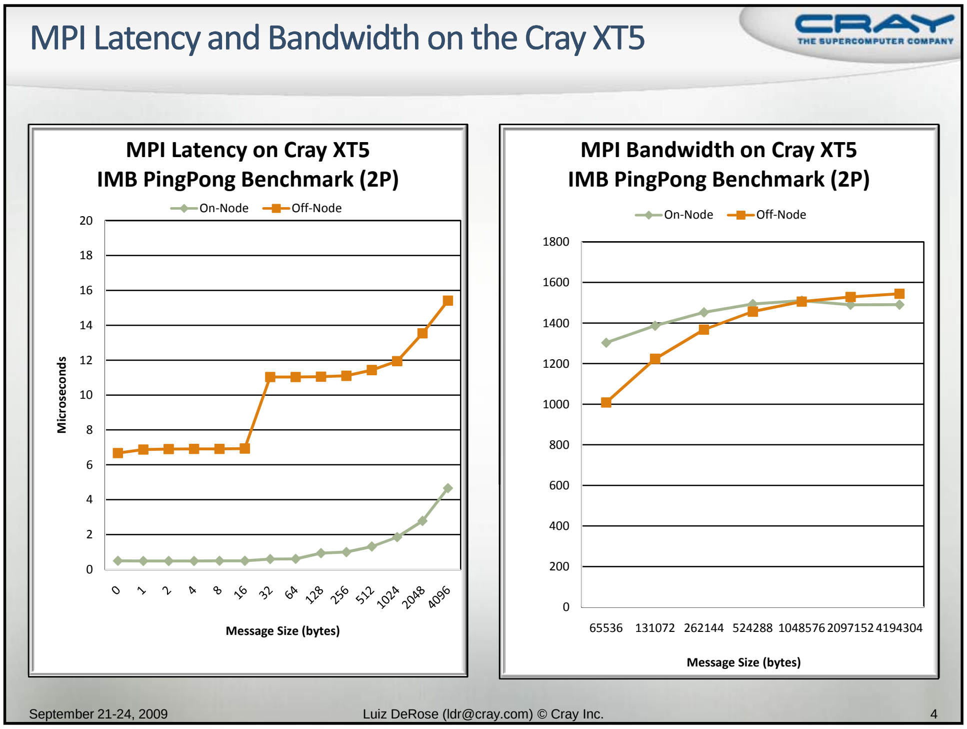
Outline

- Overview of Cray Message Passing Toolkit (MPT)
- Key Cray MPI Environment Variables
- Cray MPI Collectives
- Cray MPI Point-to-Point Messaging Techniques
- Memory Usage when Scaling Applications
- When Something Goes Wrong - Cray MPI Error Messages



Cray Message Passing Toolkit (MPT) 3.x

- Toolkit includes MPI and SHMEM
  - MPI based off of MPICH2 version 1.0.6 from ANL
  - Support for multiple compilers (CCE, PGI, GNU)
  - Numerous Cray enhancements and optimizations
- What unique features does CRAY MPI provide for the Cray XT?
  - Custom Portals device driver
  - Custom Shared Memory (SMP) device driver
  - Multi-device implementation for a single job
    - Optimal messaging path is selected automatically
  - Optimized Collectives
  - MPI I/O Enhancements
  - Support for up to 256,000 MPI ranks
  - Custom Process Manager Interface (PMI) for launching
    - Interfaces with existing ALPS software (aprun)
    - A PMI daemon process is started on each node
    - Support for Process-to-CPU affinity
    - Support for Rank Re-Ordering



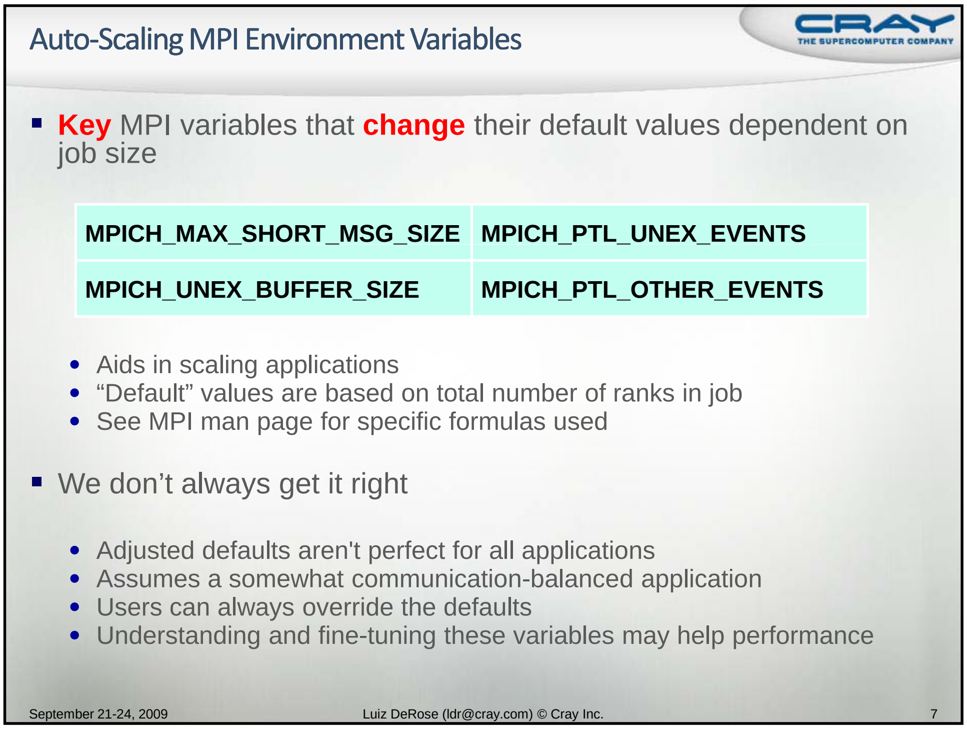
MPI Latency and Bandwidth on the Cray XT5

Key Cray MPI Environment Variables

- Why use MPI environment variables?
  - Allow users to tweak optimizations for specific application behavior
  - Flexibility to choose cutoff values for collective optimizations
  - Determine maximum size of internal MPI buffers/queues
- MPI Display Variables
  - **MPICH_VERSION_DISPLAY**
    - Displays version of Cray MPI being used
      - MPI VERSION : CRAY MPICH2 XT version 3.1.2 (ANL base 1.0.6)
      - BUILD INFO : Built Mon Feb 16 10:20:17 2009 (svn rev 7304)
    - strings ./mpi.exe | grep VERSION
  - **MPICH_ENV_DISPLAY**
    - Displays all MPI env variables and their current values
    - Helpful to determine what defaults are set to

MPICH_ENV_DISPLAY & MPICH_VERSION_DISPLAY

```
MPI VERSION : CRAY MPICH2 XT version 3.1.2-
pre (ANL base 1.0.6)
BUILD INFO  : Built Thu Feb 26  3:58:36 2009
(svn rev 7308)
PE 0: MPICH environment settings:
PE 0:   MPICH_ENV_DISPLAY           = 1
PE 0:   MPICH_VERSION_DISPLAY       = 1
PE 0:   MPICH_ABORT_ON_ERROR        = 0
PE 0:   MPICH_CPU_YIELD             = 0
PE 0:   MPICH_RANK_REORDER_METHOD   = 1
PE 0:   MPICH_RANK_REORDER_DISPLAY  = 0
PE 0:   MPICH_MAX_THREAD_SAFETY     = single
PE 0:   MPICH_MSGS_PER_PROC         = 16384
PE 0: MPICH/SMP environment settings:
PE 0:   MPICH_SMP_OFF               = 0
PE 0:   MPICH_SMPDEV_BUFS_PER_PROC  = 32
PE 0:   MPICH_SMP_SINGLE_COPY_SIZE  = 131072
PE 0:   MPICH_SMP_SINGLE_COPY_OFF   = 0
PE 0: MPICH/PORTALS environment settings:
PE 0:   MPICH_MAX_SHORT_MSG_SIZE    = 128000
PE 0:   MPICH_UNEX_BUFFER_SIZE      = 62914560
PE 0:   MPICH_PTL_UNEX_EVENTS       = 20480
PE 0:   MPICH_PTL_OTHER_EVENTS      = 2048
PE 0:   MPICH_VSHORT_OFF            = 0
PE 0:   MPICH_MAX_VSHORT_MSG_SIZE   = 1024
PE 0:   MPICH_VSHORT_BUFFERS        = 32
PE 0:   MPICH_PTL_EAGER_LONG        = 0
PE 0:   MPICH_PTL_MATCH_OFF         = 0
PE 0:   MPICH_PTL_SEND_CREDITS      = 0
PE 0: MPICH/COLLECTIVE environment settings:
PE 0:   MPICH_FAST_MEMCPY           = 0
PE 0:   MPICH_COLL_OPT_OFF          = 0
PE 0:   MPICH_COLL_SYNC             = 0
PE 0:   MPICH_BCAST_ONLY_TREE       = 1
PE 0:   MPICH_ALLTOALL_SHORT_MSG    = 1024
PE 0:   MPICH_REDUCE_SHORT_MSG      = 65536
PE 0:   MPICH_REDUCE_LARGE_MSG      = 131072
PE 0:   MPICH_ALLREDUCE_LARGE_MSG   = 262144
PE 0:   MPICH_ALLGATHER_VSHORT_MSG  = 2048
PE 0:   MPICH_ALLTOALLVW_FCSIZE     = 32
PE 0:   MPICH_ALLTOALLVW_SENDWIN    = 20
PE 0:   MPICH_ALLTOALLVW_RECVWIN    = 20
PE 0: MPICH/MPIIO environment settings:
PE 0:   MPICH_MPIIO_HINTS_DISPLAY   = 0
PE 0:   MPICH_MPIIO_CB_ALIGN        = 0
PE 0:   MPICH_MPIIO_HINTS           = NULL
```



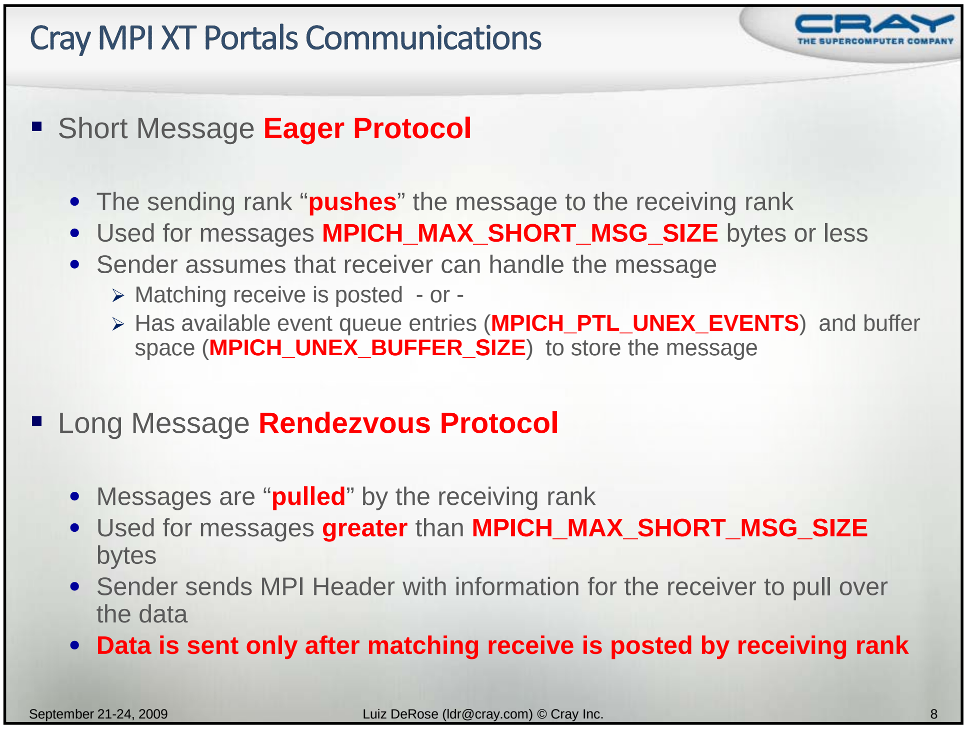
Auto-Scaling MPI Environment Variables

- **Key** MPI variables that **change** their default values dependent on job size

MPICH_MAX_SHORT_MSG_SIZE	**MPICH_PTL_UNEX_EVENTS**
MPICH_UNEX_BUFFER_SIZE	**MPICH_PTL_OTHER_EVENTS**

  - Aids in scaling applications
  - “Default” values are based on total number of ranks in job
  - See MPI man page for specific formulas used

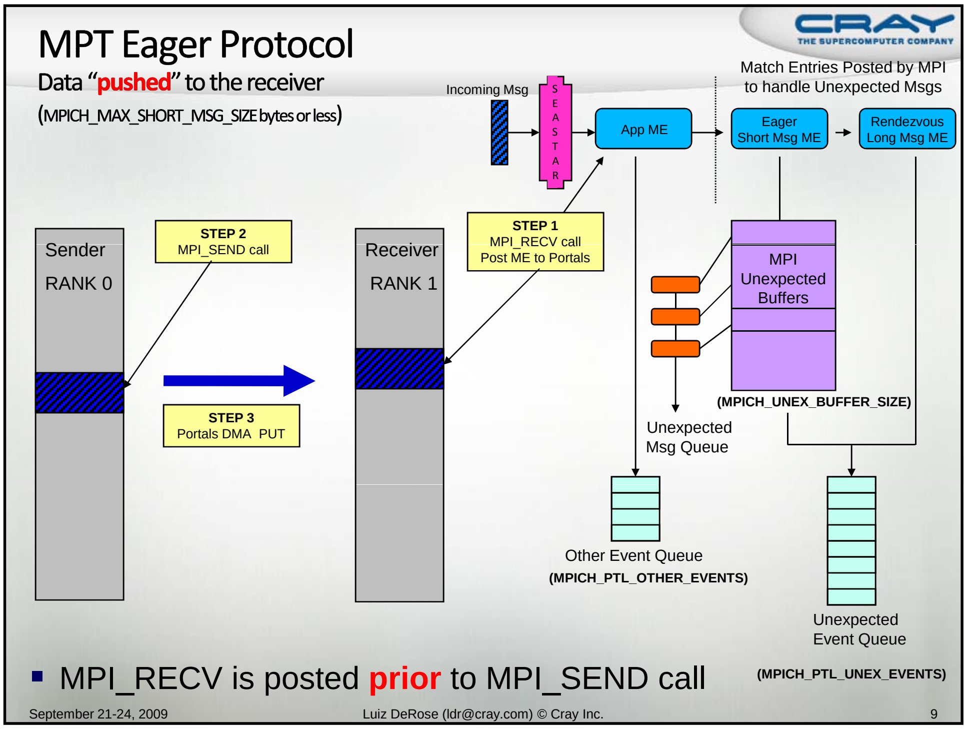
- We don’t always get it right
  - Adjusted defaults aren't perfect for all applications
  - Assumes a somewhat communication-balanced application
  - Users can always override the defaults
  - Understanding and fine-tuning these variables may help performance



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 MPI Header 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)

Incoming Msg

SEASTAR

App ME

Match Entries Posted by MPI to handle Unexpected Msgs

Eager Short Msg ME

Rendezvous Long Msg ME

Sender

RANK 0

STEP 2 MPI_SEND call

Receiver

RANK 1

STEP 1 MPI_RECV call Post ME to Portals

MPI Unexpected Buffers

STEP 3 Portals DMA PUT

(MPICH_UNEX_BUFFER_SIZE)

Unexpected Msg Queue

Other Event Queue

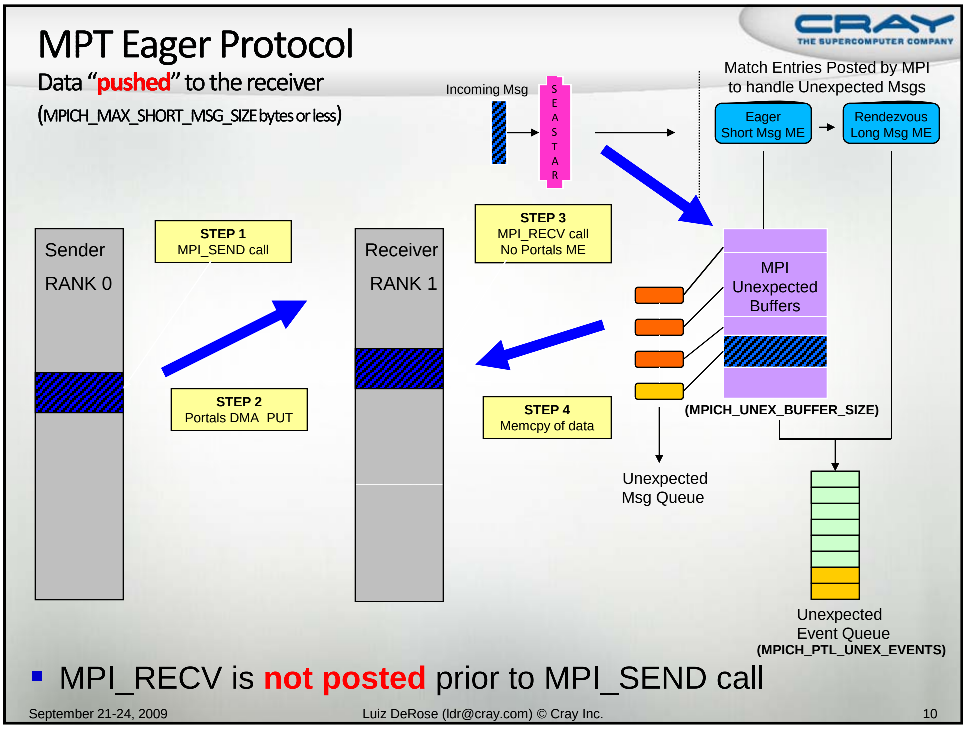
(MPICH_PTL_OTHER_EVENTS)

Unexpected Event Queue

(MPICH_PTL_UNEX_EVENTS)

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

September 21-24, 2009 Luiz DeRose (ldr@cray.com) © Cray Inc. 9



MPT Eager Protocol

Data "**pushed**" to the receiver

(MPICH_MAX_SHORT_MSG_SIZE bytes or less)

Incoming Msg

SEASTAR

Match Entries Posted by MPI to handle Unexpected Msgs

Eager Short Msg ME

Rendezvous Long Msg ME

Sender

RANK 0

STEP 1 MPI_SEND call

Receiver

RANK 1

STEP 3 MPI_RECV call No Portals ME

STEP 2 Portals DMA PUT

MPI Unexpected Buffers

STEP 4 Memcpy of data

(MPICH_UNEX_BUFFER_SIZE)

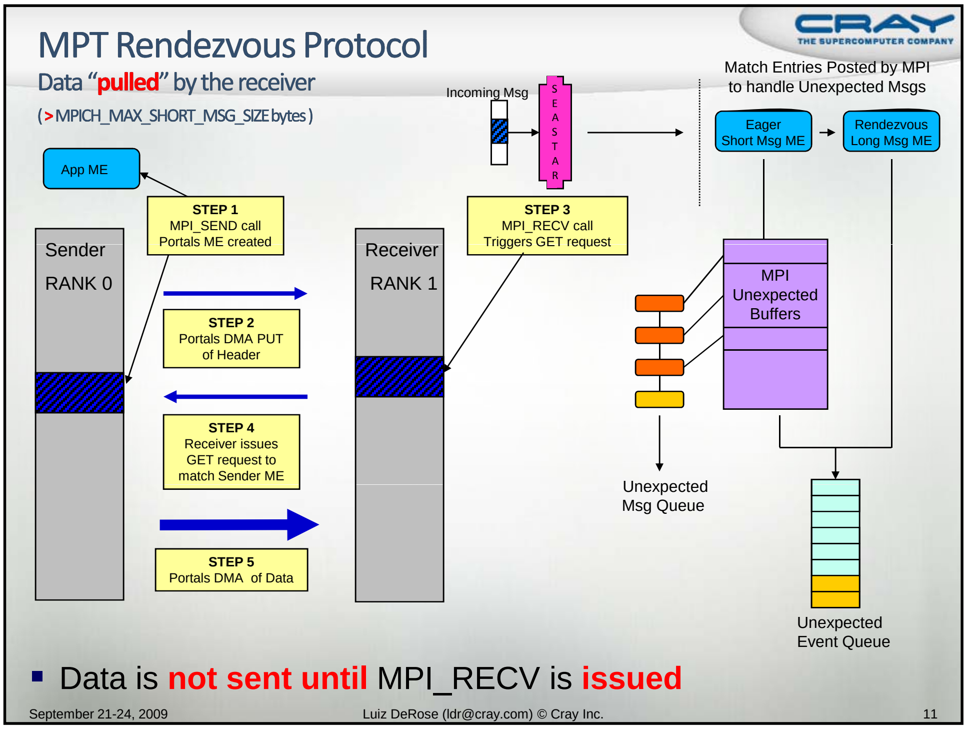
Unexpected Msg Queue

Unexpected Event Queue

(MPICH_PTL_UNEX_EVENTS)

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

September 21-24, 2009 Luiz DeRose (ldr@cray.com) © Cray Inc. 10



MPT Rendezvous Protocol

Data "**pulled**" by the receiver

(> MPICH_MAX_SHORT_MSG_SIZE bytes)

App ME

STEP 1
MPI_SEND call
Portals ME created

Sender
RANK 0

STEP 2
Portals DMA PUT
of Header

STEP 4
Receiver issues
GET request to
match Sender ME

STEP 5
Portals DMA of Data

Receiver
RANK 1

Incoming Msg

SEASTAR

STEP 3
MPI_RECV call
Triggers GET request

Match Entries Posted by MPI
to handle Unexpected Msgs

Eager
Short Msg ME

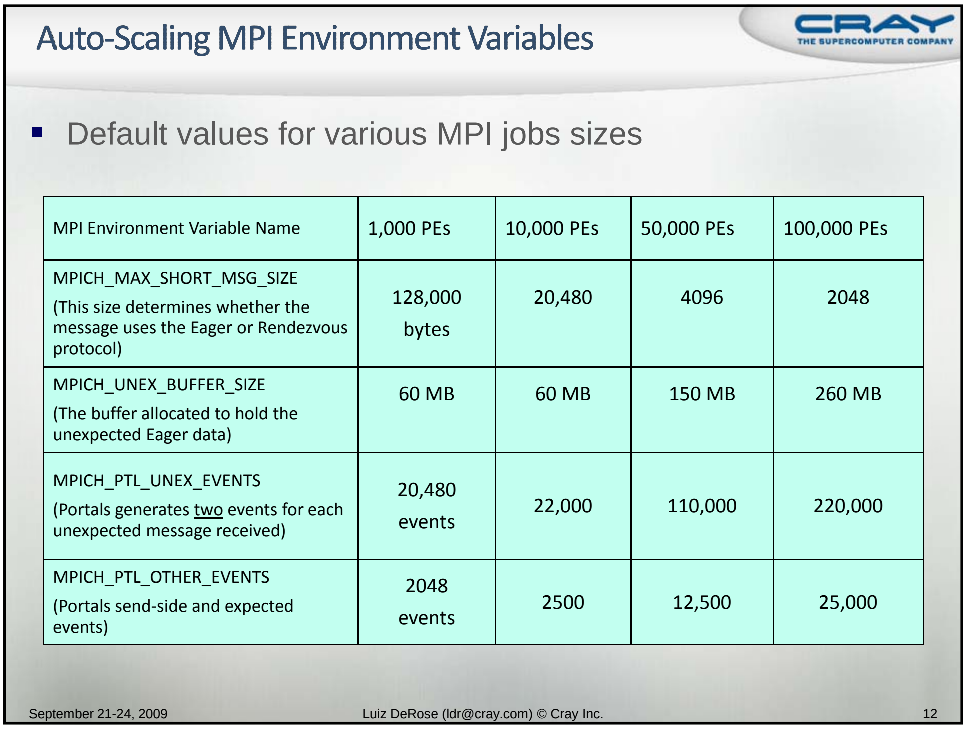
Rendezvous
Long Msg ME

MPI
Unexpected
Buffers

Unexpected
Msg Queue

Unexpected
Event Queue

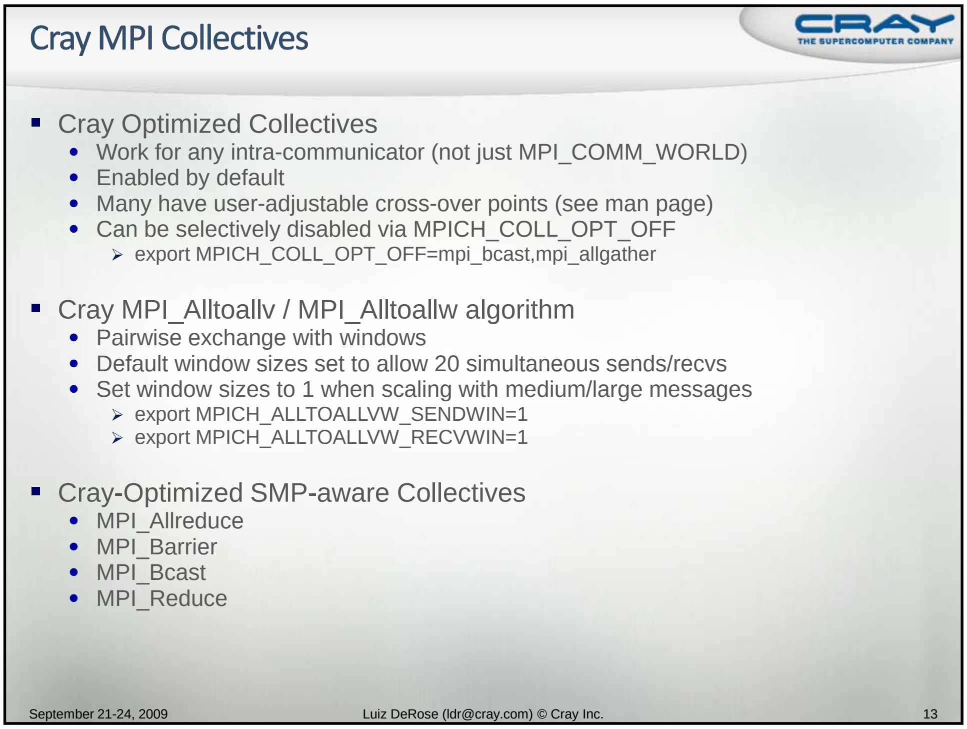
- Data is **not sent until** MPI_RECV is **issued**



Auto-Scaling MPI Environment Variables

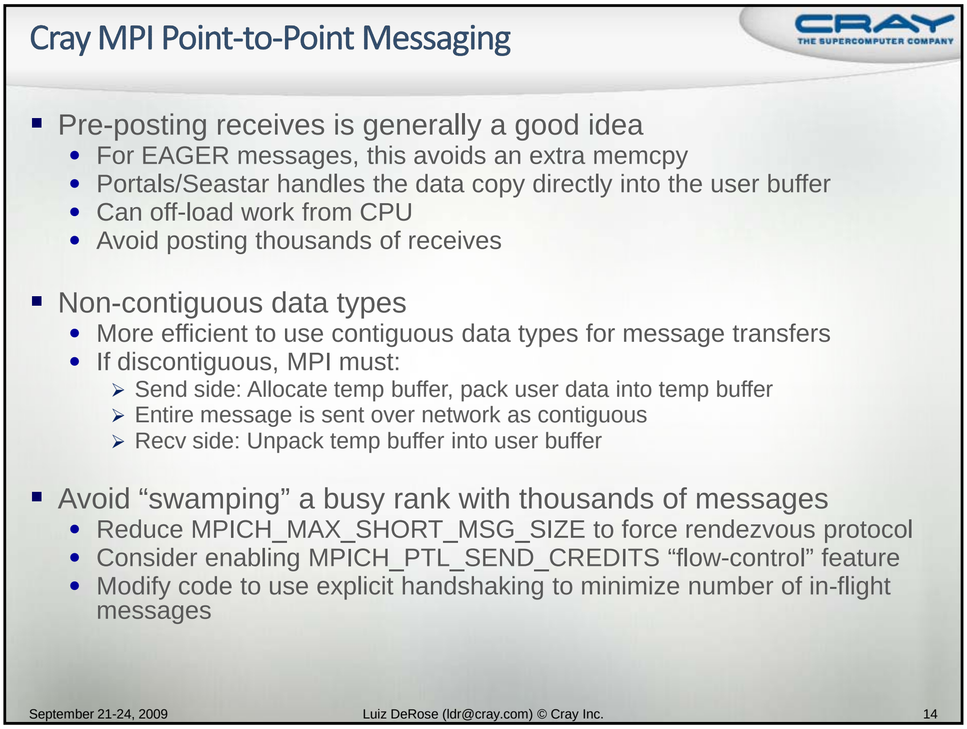
- Default values for various MPI jobs sizes

MPI Environment Variable Name	1,000 PEs	10,000 PEs	50,000 PEs	100,000 PEs
MPICH_MAX_SHORT_MSG_SIZE (This size determines whether the message uses the Eager or Rendezvous protocol)	128,000 bytes	20,480	4096	2048
MPICH_UNEX_BUFFER_SIZE (The buffer allocated to hold the unexpected Eager data)	60 MB	60 MB	150 MB	260 MB
MPICH_PTL_UNEX_EVENTS (Portals generates two events for each unexpected message received)	20,480 events	22,000	110,000	220,000
MPICH_PTL_OTHER_EVENTS (Portals send-side and expected events)	2048 events	2500	12,500	25,000



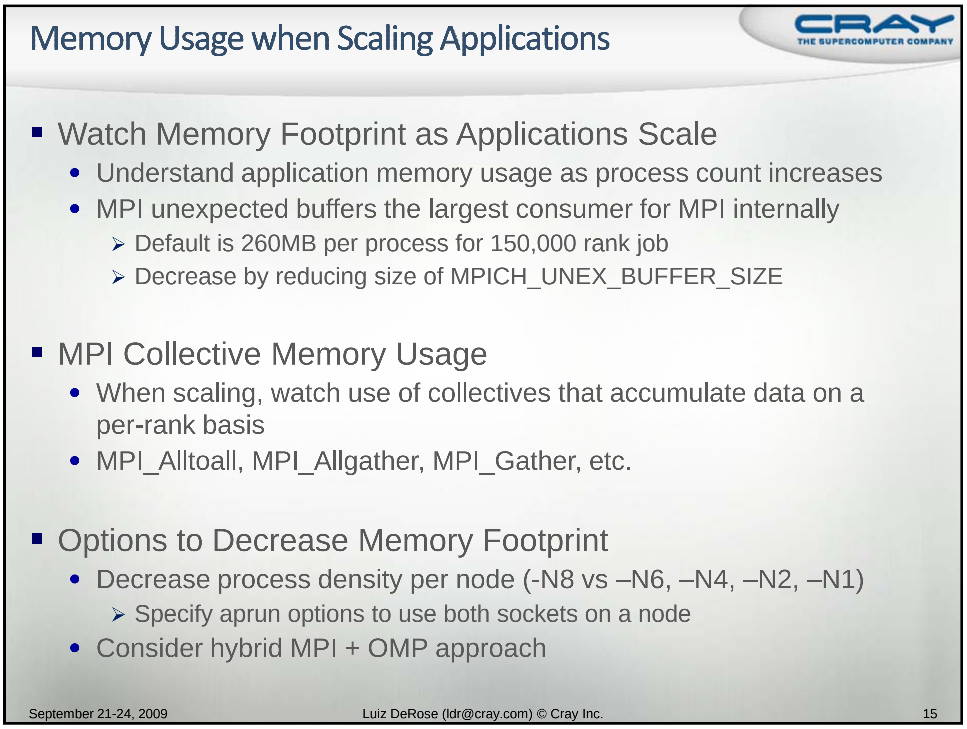
Cray MPI Collectives

- Cray Optimized Collectives
  - Work for any intra-communicator (not just MPI_COMM_WORLD)
  - Enabled by default
  - Many have user-adjustable cross-over points (see man page)
  - Can be selectively disabled via MPICH_COLL_OPT_OFF
    - export MPICH_COLL_OPT_OFF=mpi_bcast,mpi_allgather
- Cray MPI_Alltoallv / MPI_Alltoallw algorithm
  - Pairwise exchange with windows
  - Default window sizes set to allow 20 simultaneous sends/recvs
  - Set window sizes to 1 when scaling with medium/large messages
    - export MPICH_ALLTOALLVW_SENDWIN=1
    - export MPICH_ALLTOALLVW_RECVWIN=1
- Cray-Optimized SMP-aware Collectives
  - MPI_Allreduce
  - MPI_Barrier
  - MPI_Bcast
  - MPI_Reduce



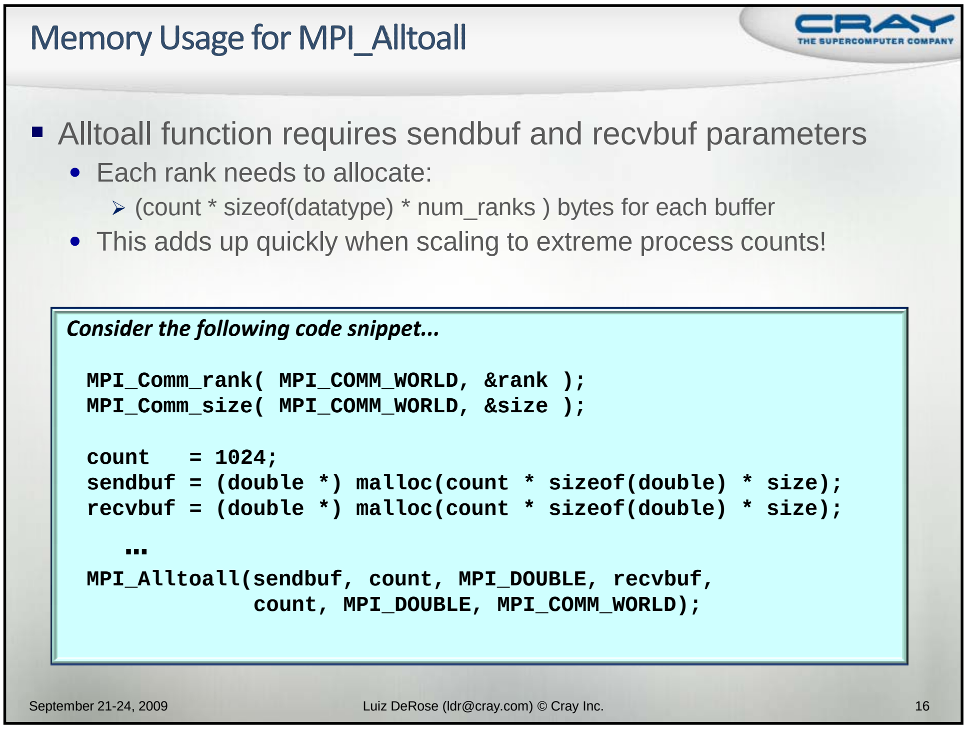
Cray MPI Point-to-Point Messaging

- Pre-posting receives is generally a good idea
  - For EAGER messages, this avoids an extra memcpy
  - Portals/Seastar handles the data copy directly into the user buffer
  - Can off-load work from CPU
  - Avoid posting thousands of receives
- Non-contiguous data types
  - More efficient to use contiguous data types for message transfers
  - If discontiguous, MPI must:
    - Send side: Allocate temp buffer, pack user data into temp buffer
    - Entire message is sent over network as contiguous
    - Recv side: Unpack temp buffer into user buffer
- Avoid “swamping” a busy rank with thousands of messages
  - Reduce MPICH_MAX_SHORT_MSG_SIZE to force rendezvous protocol
  - Consider enabling MPICH_PTL_SEND_CREDITS “flow-control” feature
  - Modify code to use explicit handshaking to minimize number of in-flight messages



Memory Usage when Scaling Applications

- Watch Memory Footprint as Applications Scale
  - Understand application memory usage as process count increases
  - MPI unexpected buffers the largest consumer for MPI internally
    - Default is 260MB per process for 150,000 rank job
    - Decrease by reducing size of MPICH_UNEX_BUFFER_SIZE
- MPI Collective Memory Usage
  - When scaling, watch use of collectives that accumulate data on a per-rank basis
  - MPI_Alltoall, MPI_Allgather, MPI_Gather, etc.
- Options to Decrease Memory Footprint
  - Decrease process density per node (-N8 vs –N6, –N4, –N2, –N1)
    - Specify aprun options to use both sockets on a node
  - Consider hybrid MPI + OMP approach



Memory Usage for MPI_Alltoall

- Alltoall function requires sendbuf and recvbuf parameters
  - Each rank needs to allocate:
    - (count * sizeof(datatype) * num_ranks) bytes for each buffer
  - This adds up quickly when scaling to extreme process counts!

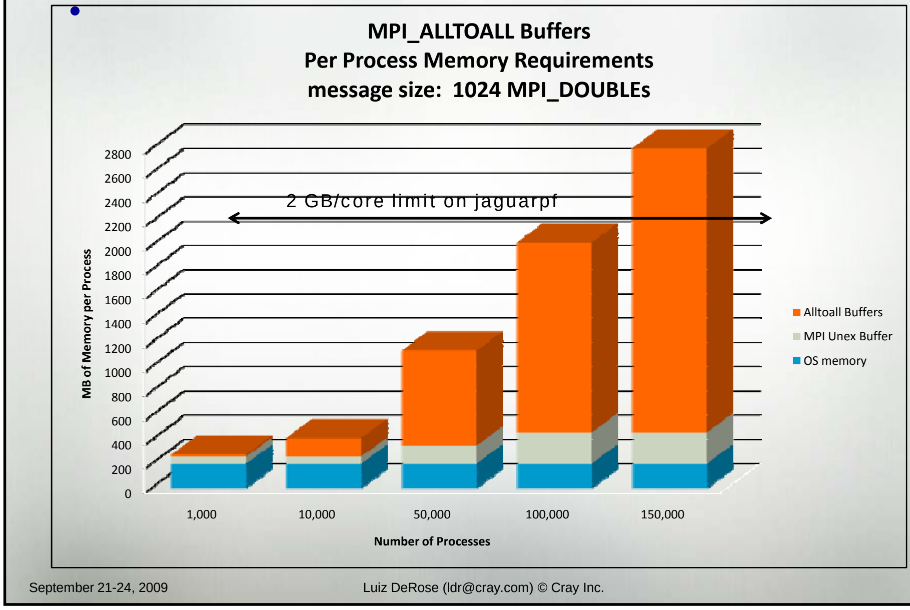
Consider the following code snippet…

```
MPI_Comm_rank( MPI_COMM_WORLD, &rank );
MPI_Comm_size( MPI_COMM_WORLD, &size );

count   = 1024;
sendbuf = (double *) malloc(count * sizeof(double) * size);
recvbuf = (double *) malloc(count * sizeof(double) * size);

   …
MPI_Alltoall(sendbuf, count, MPI_DOUBLE, recvbuf,
             count, MPI_DOUBLE, MPI_COMM_WORLD);
```

Memory Usage for Scaling MPI_Alltoall



When Something Goes Wrong - MPI Error Messages

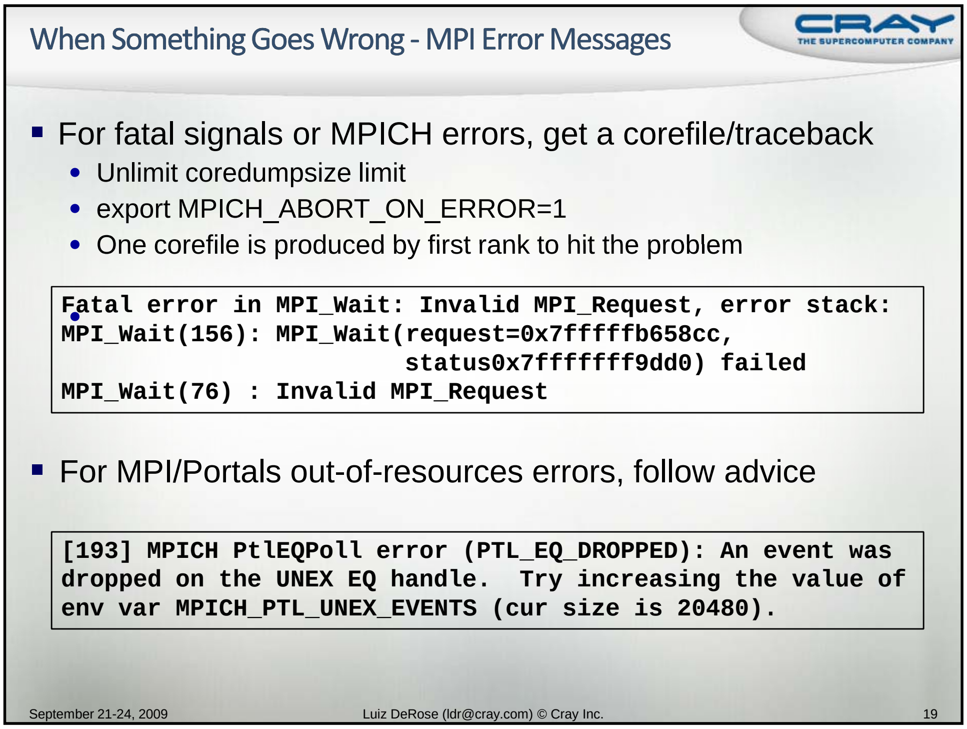
- If a rank exits abnormally, PMI daemon reports the error

```
_pmii_daemon(SIGCHLD): PE 1036 exit signal Segmentation fault
_pmii_daemon(SIGCHLD): PE 0 exit signal Killed
_pmii_daemon(SIGCHLD): PE 1 exit signal Killed
_pmii_daemon(SIGCHLD): PE 2 exit signal Killed
   ...
_pmii_daemon(SIGCHLD): PE 1035 exit signal Killed
```

- To quiet the PMI daemon, use: export PMI_QUIET=1
- Rely on single aprun error message for clues

```
 [NID 3343]Apid 250839: initiated application termination
 Application 250839 exit codes: 139
 Application 250839 exit signals: Killed
 Application 250839 resources: utime 0, stime 0
```

Subtract 128 from aprun exit code to get the fatal signal number. In this case, signal 11 is a segmentation fault. See aprun man page for more info.



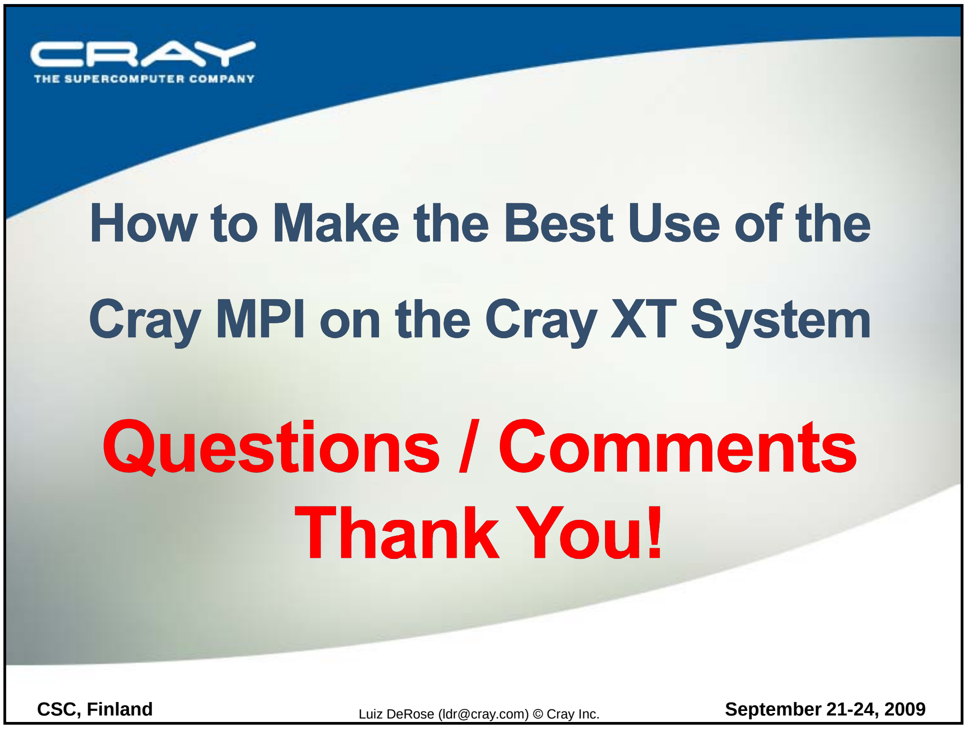
When Something Goes Wrong - MPI Error Messages

- For fatal signals or MPICH errors, get a corefile/traceback
  - Unlimit coredumpsize limit
  - export MPICH_ABORT_ON_ERROR=1
  - One corefile is produced by first rank to hit the problem

```
Fatal error in MPI_Wait: Invalid MPI_Request, error stack:
MPI_Wait(156): MPI_Wait(request=0x7fffffb658cc,
                        status0x7fffffff9dd0) failed
MPI_Wait(76) : Invalid MPI_Request
```

- For MPI/Portals out-of-resources errors, follow advice

```
[193] MPICH PtlEQPoll error (PTL_EQ_DROPPED): An event was
dropped on the UNEX EQ handle.  Try increasing the value of
env var MPICH_PTL_UNEX_EVENTS (cur size is 20480).
```



How to Make the Best Use of the Cray MPI on the Cray XT System

Questions / Comments
Thank You!