



ALCF Getting Started Videoconference January 2014

Agenda

■ Part I

- Blue Gene/Q hardware overview
- Building your code
- Considerations before you run
- Queuing and running

■ Part II

- After your job is submitted
- Potential problems
- Performance tuning
- Backups, Disk Storage and Tape Archival
- Getting Help

■ Hands-on session





Part I



Section:

Blue Gene/Q hardware overview

ALCF Resources

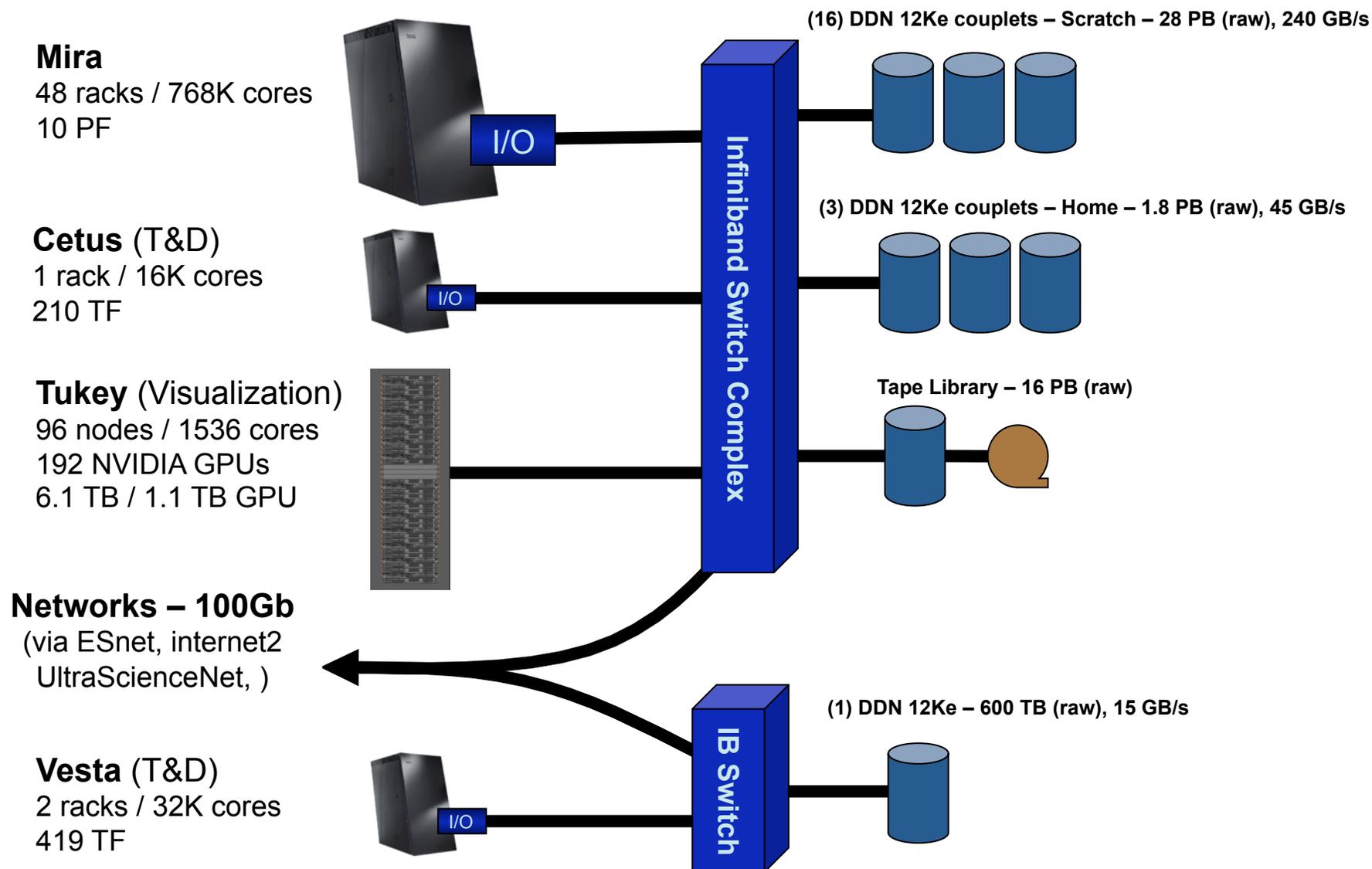
- **Mira (Production)** – IBM BG/Q system
 - 49,152 nodes / 786,432 cores
 - 768 TB of memory
 - Peak flop rate: 10 PF
 - Linpack flop rate: 8.1 PF
- **Cetus (Test & Devel.)** – IBM BG/Q system
 - 1,024 nodes / 16,384 cores
 - 16 TB of memory
 - 210 TF peak flop rate
- **Vesta (Test & Devel.)** – IBM BG/Q system
 - 2,048 nodes / 32,768 cores
 - 32 TB of memory
 - 419 TF peak flop rate
- **Tukey (Visualization)** – NVIDIA system
 - 96 nodes / 1536 x86 cores / 192 M2070 GPUs
 - 6.1 TB x86 memory / 1.1 TB GPU memory
 - Peak flop rate: 220 TF



- **Storage**
 - Scratch: 28.8 PB raw capacity, 240 GB/s bw (GPFS)
 - Home: 1.8 PB raw capacity, 45 GB/s bw (GPFS)
 - Storage upgrade planned in 2015



ALCF Resources



Blue Gene series features

- **Low speed, low power**
 - Embedded PowerPC core with custom SIMD floating point extensions
 - Low frequency (**L** – 700 MHz, **P** – 850 MHz, **Q** – 1.6 GHz)
- **Massive parallelism**
 - Many cores (**L** – 208k, **P** – 288k, **Q** – 1.5M)
- **Fast communication network(s)**
 - Torus network (**L** & **P** – 3D, **Q** – 5D)
- **Balance**
 - Processor, network, and memory speeds are well balanced
- **Minimal system overhead**
 - Simple lightweight OS (CNK) minimizes noise
- **Standard Programming Models**
 - Fortran, C, C++ & Python languages supported
 - Provides MPI, OpenMP, and Pthreads parallel programming models
- **System-on-a-Chip (SoC) & Custom designed ASIC (Application Specific Integrated Circuit)**
 - All node components on one chip, except for memory
 - Reduces system complexity and power, improves price / performance
- **High Reliability**
 - Sophisticated RAS (Reliability, Availability, and Serviceability)
- **Dense packaging**
 - 1024 nodes per rack

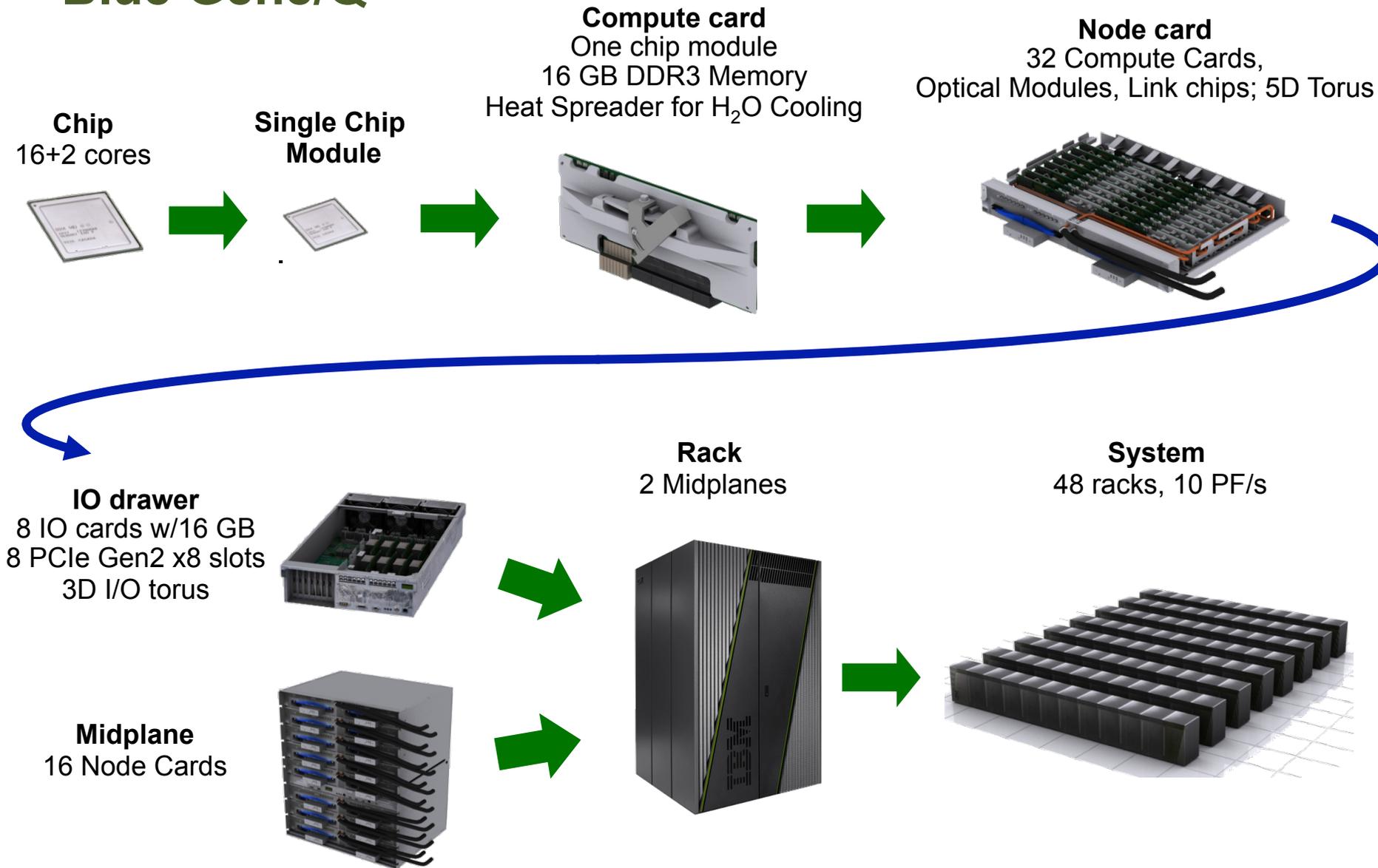


Evolution from P to Q

Design Parameters	BG/P	BG/Q	Difference
Cores / Node	4	16	4x
Hardware Threads / Core	1	4	4x
Concurrency / Rack	4,096	65,536	16x
Clock Speed (GHz)	0.85	1.6	1.9x
Flop / Clock / Core	4	8	2x
Flop / Node (GF)	13.6	204.8	15x
RAM / core (GB)	0.5 or 1	1	2x or 1x
Mem. BW/Node (GB/sec)	13.6	42.6	3x
Latency (MPI zero-length, nearest-neighbor node)	2.6 μ s	2.2 μ s	~15% less
Bisection BW (32 racks)	1.39TB/s	13.1TB/s	9.42x
Network	3D Torus + Collectives	5D Torus	Smaller diameter
GFlops/Watt	0.77	2.10	3x
Instruction Set	32 bit PowerPC + DH	64 bit PowerPC + QPX	New vector instructions
Programming Models	MPI + OpenMP	MPI + OpenMP	
Cooling	Air	Water	

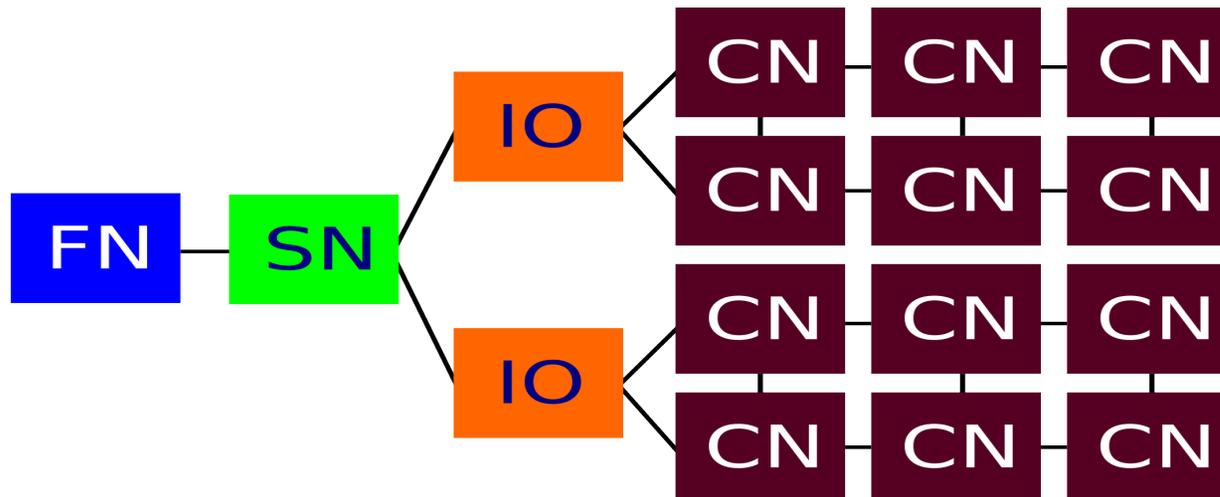


Blue Gene/Q



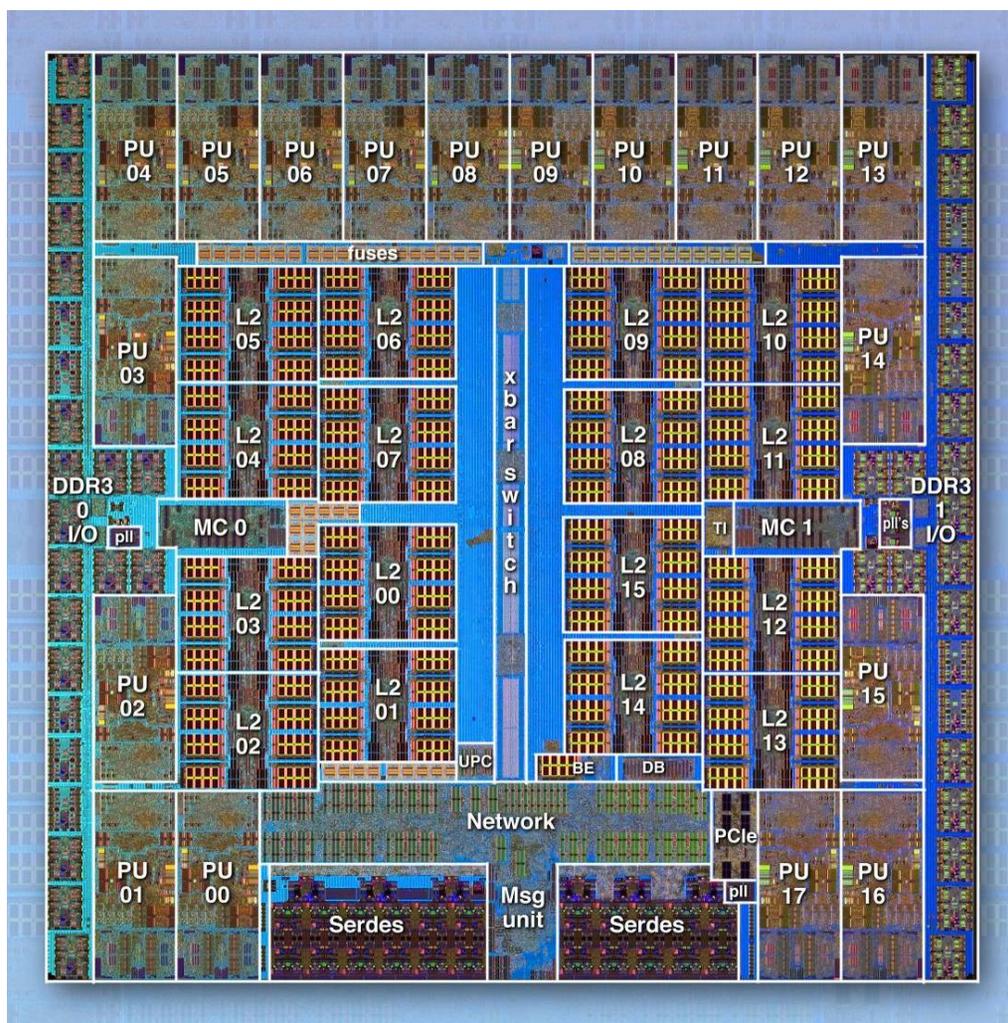
Blue Gene/Q System Components

- **Front-end nodes** – dedicated for user's to login, compile programs, submit jobs, query job status, debug applications. **RedHat Linux OS.**
- **Service nodes** – perform partitioning, monitoring, synchronization and other system management services. Users do not run on service nodes directly.
- **I/O nodes** – provide a number of Linux/Unix typical services, such as files, sockets, process launching, signals, debugging; run Linux.
- **Compute nodes** – run user applications, use simple **compute node kernel (CNK)** operating system, ships I/O-related system calls to I/O nodes.



Blue Gene/Q Compute Chip

System-on-a-Chip design: integrates processors, memory and networking logic into a single chip

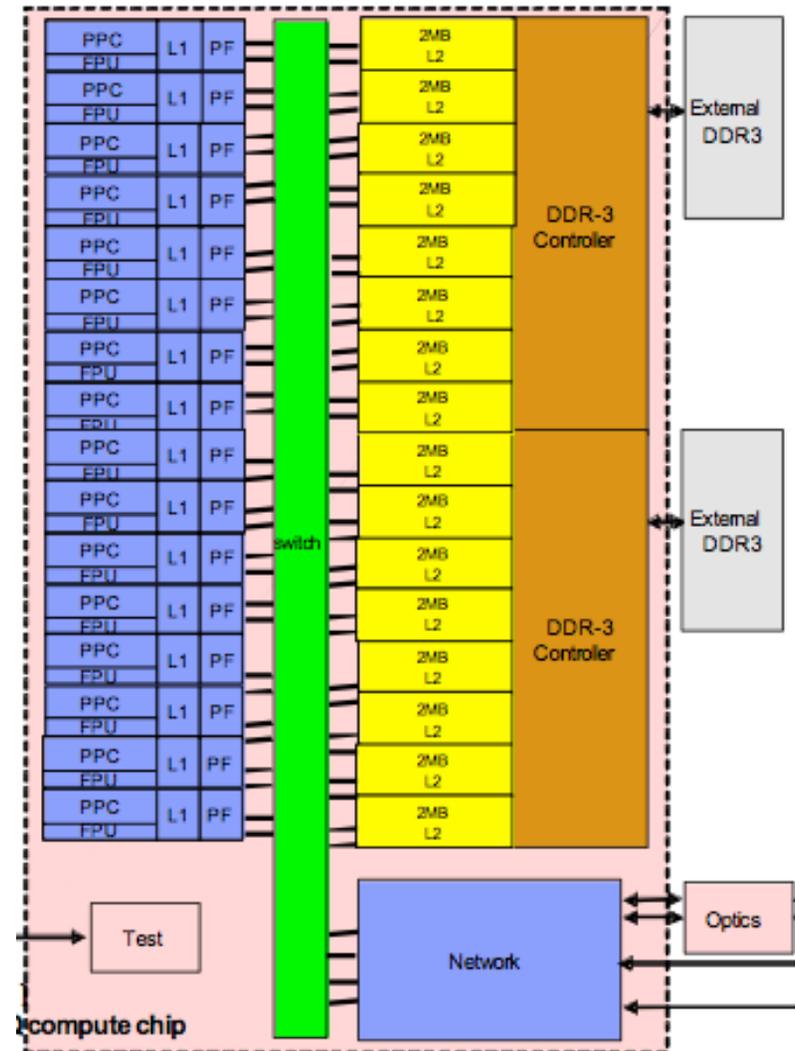


- **360 mm² Cu-45 technology (SOI)**
 - ~ 1.47 B transistors
- **16 user + 1 service processors**
 - 16 compute cores
 - 17th core for system functions (OS, RAS)
 - plus 1 redundant processor
 - all processors are symmetric
 - L1 I/D cache = 16kB/16kB
 - L1 prefetch engines (L1P)
- **Crossbar switch**
 - Connects cores via L1P to L2 slices
 - Aggregate read rate of 409.6 GB/s
- **Central shared L2 cache**
 - 32 MB eDRAM
 - 16 slices
- **Dual memory controller**
 - 16 GB external DDR3 memory
 - 42.6 GB/s bandwidth
- **Chip-to-chip networking**
 - Router logic integrated into BQC chip
 - DMA, remote put/get, collective operations
 - 11 network ports
- **External IO**
 - PCIe Gen2 interface



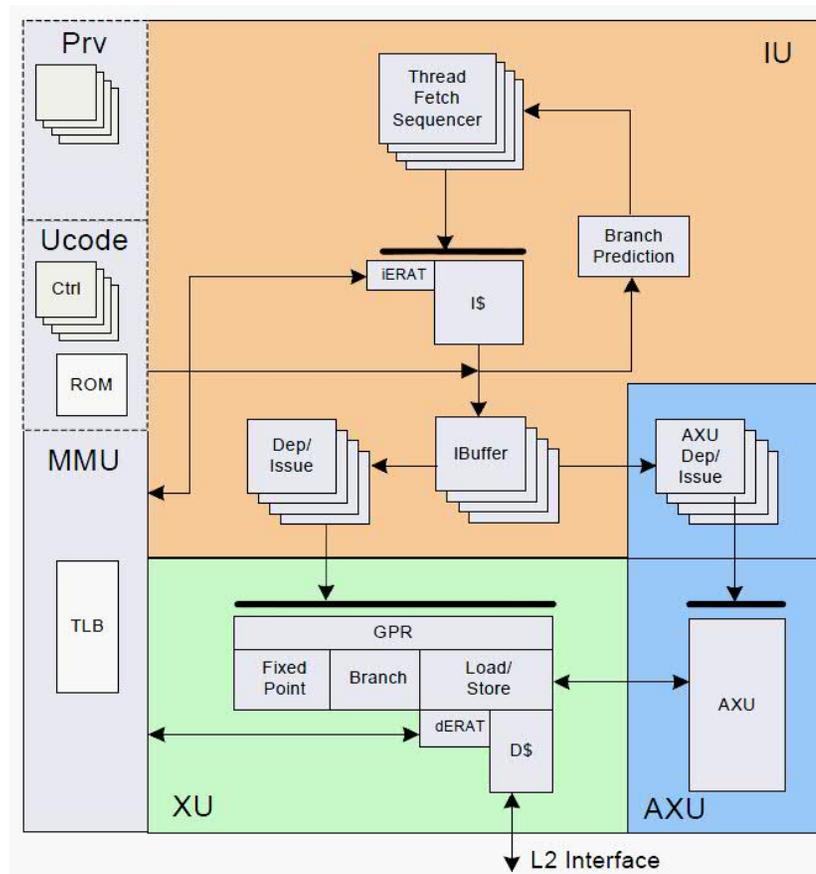
Blue Gene/Q Chip

- Design based on simple, power efficient PowerPC core:
 - Full PowerPC compliant 64-bit CPU, PowerISA v.206
 - A2 core also used in IBM PowerEn chip
 - Runs at 1.6 GHz
- Unique BG/Q ASIC with special features:
 - 4-wide SIMD floating point unit (QPX)
 - Transactional Memory & Speculative Execution
 - Fast memory based atomic operations
 - Stream and list based prefetching
 - WakeUp Unit
 - Universal Performance Counters (UPC)



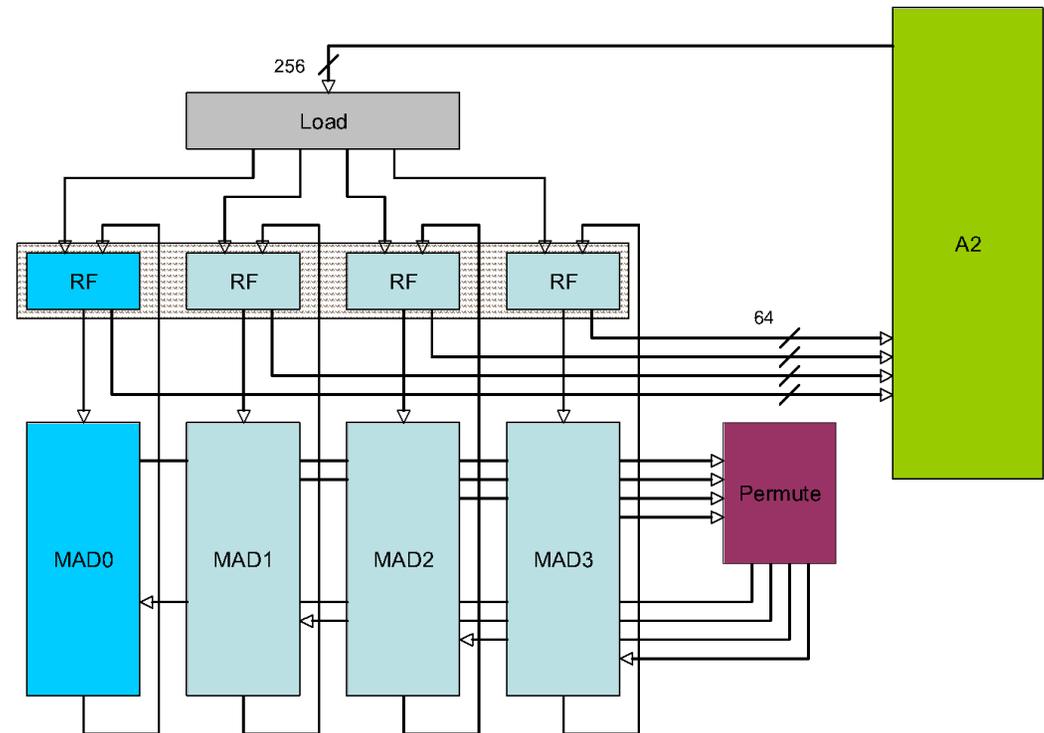
Blue Gene/Q Core

- In-order execution
- 4-way Simultaneous Multi-Threading
- 32 64-bit integer registers, 32 256-bit FP registers
- Dynamic branch prediction
- Functional Units:
 - IU – instructions fetch and decode
 - XU – Branch, Integer, Load/Store instructions
 - AXU – Floating point instructions
 - Standard PowerPC instructions
 - QPX 4 wide SIMD
 - MMU – memory management (TLB)
- Instruction Issue:
 - 2-way concurrent issue 1 XU + 1 AXU
 - A given thread may only issue 1 instruction per cycle
 - Two threads may issue up to two instructions every cycle, one instruction per cycle per thread.

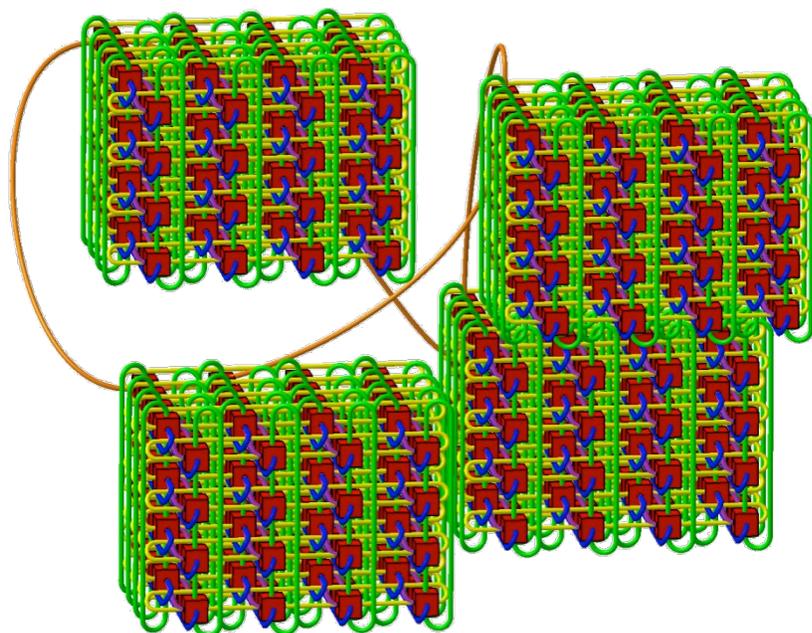


Quad Processing eXtension (QPX) Overview

- Unique 4 wide double precision SIMD instructions extending standard PowerISA with:
 - Full set of arithmetic functions
 - Load/store instructions
 - Permute instructions to reorganize data
- 4 wide FMA instructions deliver 8 flops/cycle in each instruction.
- FPU operates on:
 - Standard scale PowerPC FP instructions (slot 0)
 - 4 wide SIMD instructions
 - 2 wide complex arithmetic SIMD arithmetic
- Standard 64-bit floating point registers are extended to 256 bits
- Attached to AXU port of A2 core – A2 issues one instruction/cycle to AXU
- 6-stage floating point instruction pipeline
- Compiler can generate QPX instructions
- Intrinsic functions mapping to QPX instructions allow easy QPX programming



Blue Gene/Q network



Network Performance

All-to-all: 97% of peak

Bisection: > 93% of peak

Nearest-neighbor: 98% of peak

Collective: FP reductions at 94.6% of peak

On chip per hop latency ~ 40 ns

Allreduce hardware latency on 96k nodes ~ 6.5 μ s

Barrier hardware latency on 96k nodes ~ 6.3 μ s

▪ 5D torus network:

- 5D torus achieves high nearest neighbor bandwidth while increasing bisectional bandwidth and reducing hops
- Allows machine to be partitioned into independent sub machines. No impact from concurrently running codes.
- Hardware assists for collective & barrier functions over COMM_WORLD and rectangular sub communicators
- Half rack (midplane) is 4x4x4x2 torus
- Last dimension is always 2

▪ Nodes have 10 links with 2 GB/s raw bandwidth each

- Bi-directional: send + receive gives 4 GB/s
- 90% of bandwidth (1.8 GB/s) available to user

▪ No separate Collectives or Barrier network:

- Single network used for point-to-point, collectives, and barrier operations

▪ Additional 11th link for communication to IO nodes

▪ Hardware latency

- Nearest: 80 ns
- Farthest: 3 μ s (96-rack 20PF system, 31 hops)

Partition dimensions on Blue Gene/Q systems

Mira

Nodes	A	B	C	D	E
512	4	4	4	4	2
1024	4	4	4	8	2
2048	4	4	4	16	2
4096	4/8	4	8/4	16	2
8192	4	4	16	16	2
12288	8	4	12	16	2
16384	4/8	8/4	16	16	2
24576	4	12	16	16	2
32768	8	8	16	16	2
49152	8	12	16	16	2

Cetus

Nodes	A	B	C	D	E
128	2	2	4	4	2
256	4	2	4	4	2
512	4	4	4	4	2
1024	4	4	4	8	2

Vesta

Nodes	A	B	C	D	E
32	2	2	2	2	2
64	2	2	4	2	2
128	2	2	4	4	2
256	4	2	4	4	2
512	4	4	4	4	2
1024	4	4	4/8	8/4	2
2048(*)	4	4	8	8	2

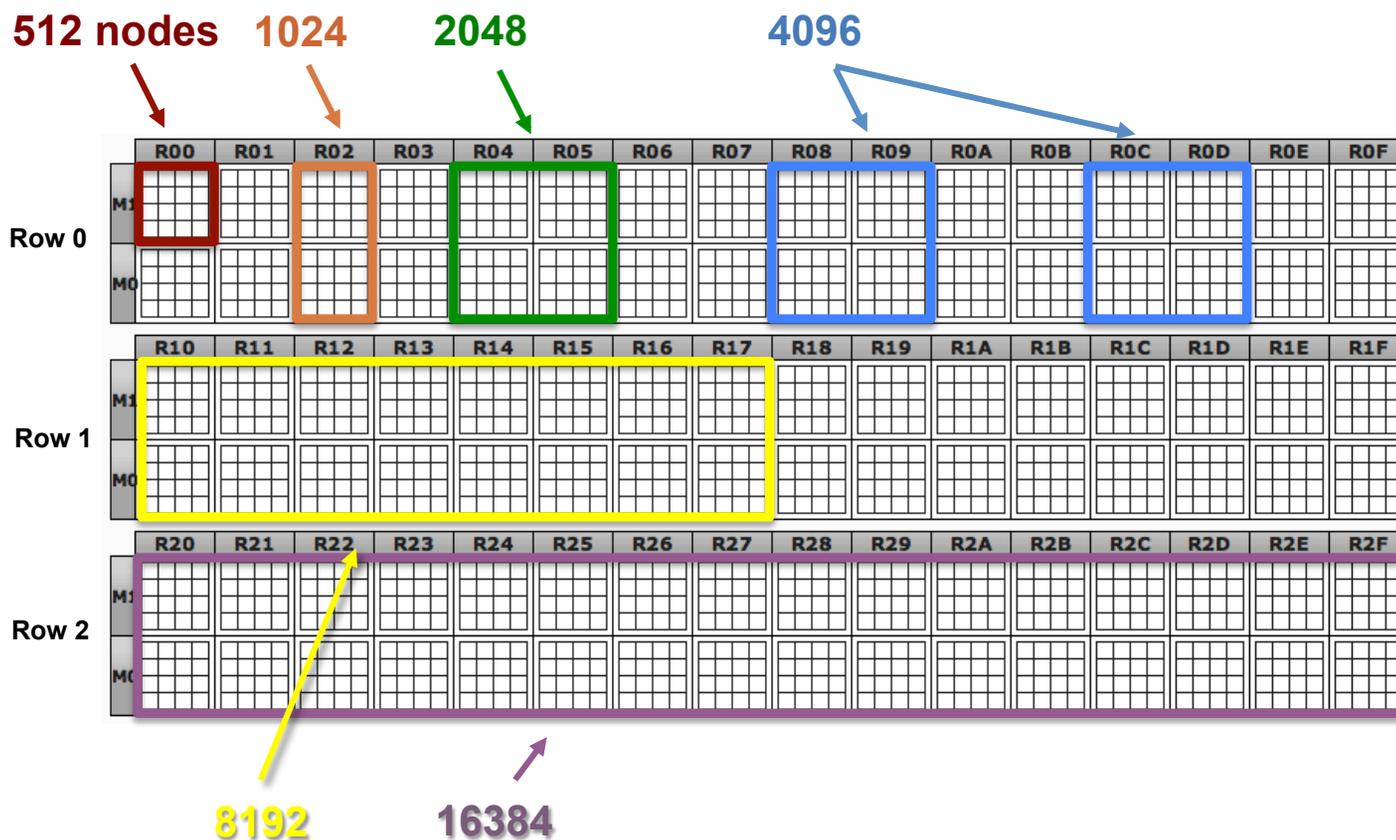
Command: partlist

<http://www.alcf.anl.gov/user-guides/machine-partitions>

(*) Partition not active.



Mira multiple rack partitions (“blocks”)



The number of large block sizes possible is:

# of nodes	# of blocks
49152	1
32768	3
24576	2
16384	9
12288	12
8192	6
4096	12
2048	24
1024	64
512	96

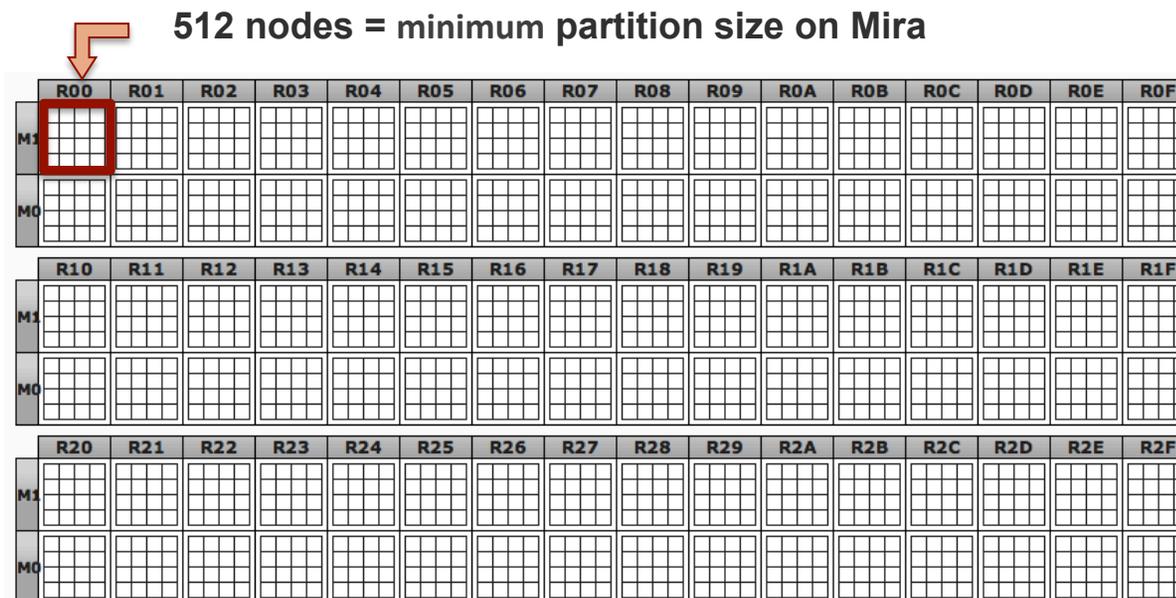
<http://status.alcf.anl.gov/mira/activity> (beta, a.k.a. The Gronkulator)

partlist will show you if a large free block is busy due to a wiring dependency

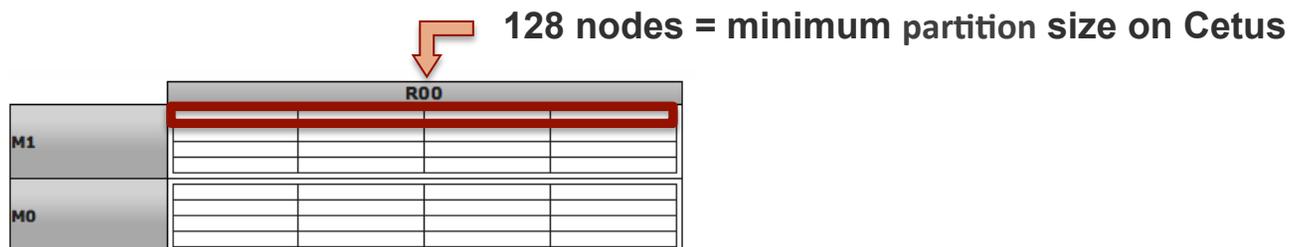


Minimum partition sizes on Blue Gene/Q systems

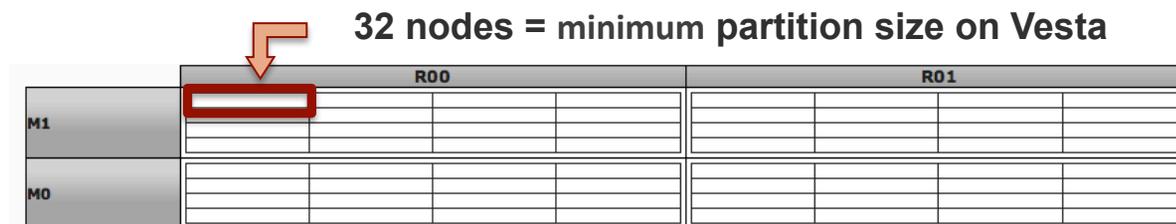
Mira
48 racks



Cetus
1 rack



Vesta
2 racks





Section:

Building your code

SoftEnv

- A tool for managing a user's environment
 - Sets your PATH to access desired front-end tools
 - *Your compiler version can be changed here*
- Settings:
 - Maintained in the file ~/.soft
 - Add/remove keywords from ~/.soft to change environment
 - ***Make sure @default is at the very end***
- Commands:
 - **softenv**
 - A list of all keywords defined on the systems
 - **resoft**
 - Reloads initial environment from ~/.soft file
 - **soft add|remove keyword**
 - Temporarily modify environment by adding/removing keywords

<http://www.mcs.anl.gov/hs/software/systems/softenv/softenv-intro.html>



Use Compiler Wrappers

- **IBM XL cross-compilers:**
 - Soft key: `+mpiwrapper-xl`
 - Non-thread-safe: `mpixlc`, `mpixlcxx`, `mpixlf77`, `mpixlf90`, `mpixlf95`, `mpixlf2003`, etc.
 - Thread-safe (add `_r` suffix): `mpixlc_r`, `mpixlcxx_r`, `mpixlf77_r`, etc.
 - “-show” option: shows complete command used to invoke compiler. E.g.:
> `mpixlc -show`
- **GNU cross-compilers:**
 - Soft key: `+mpiwrapper-gcc`
 - `mpicc`, `mpicxx`, `mpif77`, `mpif90`
- **CLANG cross-compilers:**
 - Soft key: `+mpiwrapper-bgclang`
 - `mpiclang`, `mpiclang++`, `mpiclang++11`

<http://www.alcf.anl.gov/user-guides/software-and-libraries>



IBM XL Optimization Settings Options

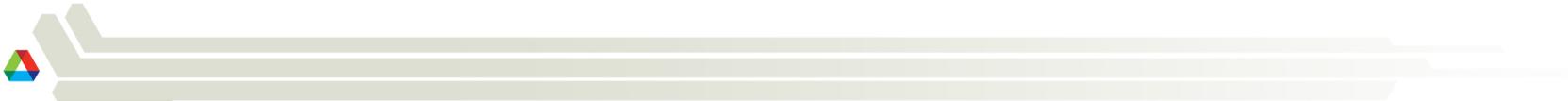
Level	Implies	Description
-O0	-qstrict -qfloat=noftint:norsqrt:rngchk -qstrict_induction	Preserves program semantics, minimal optimization, best for debugging
-O2 (or -O)	-qstrict -qfloat=noftint:norsqrt:rngchk -qnostrict_induction -qmaxmem=8192	Preserves program semantics, eliminates redundant code, basic loop optimization
-O3	-qnostrict -qfloat=fltint:rsqrt:norngchk -qnostrict_induction -qmaxmem=-1 -qhot=level=0	High order loop analysis and transformations, better loop scheduling, inlining, in depth memory access analysis, <i>can alter program semantics</i>
-O4	<i>All -O3 options plus</i> -qhot=level=1 -qhot=vector -qipa=level=1	Additional loop analysis, basic interprocedural optimization, <i>can alter program semantics</i>
-O5	<i>All -O4 options plus</i> -qipa=level=2	Advanced interprocedural analysis (IPA), <i>can alter program semantics</i>



IBM XL Hierarchy Optimization Levels

- Suggested set of optimization levels from least to most optimization:
 - -O0 # best level for use with a debugger
 - -O2 # good level for verifying correctness, baseline perf
 - -O2 -qmaxmem=-1 -qhot=level=0
 - -O3 -qstrict (preserves program semantics)
 - -O3
 - -O3 -qhot=level=1
 - -O4
 - -O5
- Tips:
 - **-qlistopt** generates a listing with all flags used in compilation
 - **-qreport** produces a listing, shows how code was optimized
 - Performance can decrease at higher levels of optimization, especially at -O4 or -O5
 - May specify different optimization levels for different routines/files
 - The compiler option ‘-g’ must be used to resolve the code line numbers in the debugger.





Threading

- **OpenMP** is supported
 - IBM XL compilers: `-qsmp=omp:noauto`
 - GNU: `-fopenmp`
 - BGCLANG: `-fopenmp`
- **Pthreads** is supported
 - NPTL Pthreads implementation in glibc requires no modifications
- Compiler **auto thread parallelization** is available
 - use `-qsmp=auto`
 - not always effective
- The runjob mode will determine maximum total number of threads (including the master thread)
 - `runjob --ranks-per-node` (or for non-script jobs, `qsub --mode`)
 - Maximum 4 threads per core
 - Each core needs at least 2 (possibly more) threads for peak efficiency



OpenMP

- Shared-memory parallelism is supported within a single node
- Hybrid programming model
 - MPI at outer level, across compute nodes
 - OpenMP at inner level, within a compute node
- **For XL compilers, thread-safe compiler version should be used** (mpixlc_r etc.) with any threaded application (either OMP or Pthreads)
- OpenMP standard directives are supported (version 3.1):
 - parallel, for, parallel for, sections, parallel sections, critical, single
 - #pragma omp <rest of pragma> for C/C++
 - !\$OMP <rest of directive> for Fortran
- Compiler functions
 - omp_get_num_procs, omp_get_num_threads
 - omp_get_thread_num, omp_set_num_threads
- Number of OpenMP threads
 - set using environment variable OMP_NUM_THREADS
 - must be exported to the compute nodes using runjob --envs (or for non-script jobs, qsub --env)



Software & Libraries on Blue Gene/Q systems

- ALCF supports two sets of libraries:
 - IBM system and provided libraries: [/bgsys/drivers/ppcfloor](#)
 - glibc
 - mpi
 - PAMI (Parallel Active Messaging Interface)
 - Site supported libraries and programs: [/soft/libraries](#)
 - PETSc, HDF5, netCDF, Parallel netCDF, Boost
 - Additional tuned libraries in [/soft/libraries/alcf](#) subdirectory
 - BLAS, CBLAS, FFTW2, FFTW3, LAPACK, METIS, PARMETIS, PARPACK, SCALAPACK, SILO, SZIP, ZLIB

For a complete list visit:

<http://www.alcf.anl.gov/user-guides/software-and-libraries>





Section:

Considerations before you run

Data Transfer to/from ALCF

The Blue Gene/Q will connect to other research institutions using a total of 100 Gbits/s of public network connectivity

Data Transfer Utilities

- **HSI** allows users to transfer data to and from HPSS.
- **GridFTP** (for large transfers)
 - Other site must accept our Certificate Authority (CA).
 - CRYPTOCARD access available.
- **sftp** and **scp** (for “small” transfers)
 - Are available for local are transfers of small files but they are not recommended for use with large data transfers due to poor performance and excess resource utilization on the login nodes.

Data Transfer Service

- **Globus** (for large transfers)
 - **Globus** addresses the challenges faced by researchers in moving, sharing, and archiving large volumes of data among distributed sites.
 - ALCF Blue Gene/Q endpoints: [alcf#dtn_mira](#)
 - Ask your laboratory or university system admin. if your institution has an endpoint.
 - **Globus Connect Personal** to share and transfer files to/from a local machine.



<http://www.alcf.anl.gov/user-guides/data-transfer>

Table of Blue Gene/Q file systems at ALCF

Name	Accessible from	Type	Path	Production	Backed-up	Uses
vesta-home	Vesta	GPFS	/home or /gpfs/vesta-home	Yes	No	General use
vesta-fs0	Vesta	GPFS	/projects	Yes	No	Intensive job output, large files
mira-home	Mira Cetus Tukey	GPFS	/home or /gpfs/mira-home	Yes	Yes	General use
mira-fs0	Mira Cetus Tukey	GPFS	/projects	Yes	No	Intensive job output, large files

<http://www.alcf.anl.gov/user-guides/bgq-file-systems>



Allocation Management

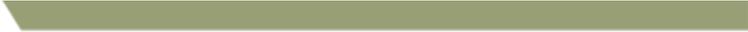
- Every user must be assigned to at least one Project:
 - Use **'projects'** command to query.
- Projects are then given allocations:
 - Allocations have an amount, start, and end date, and are tracked separately; Charges will cross allocations automatically. The allocation with the earliest end date will be charged first, until it runs out, then the next, and so on.
- Use **'cbank'** command to query allocation, balance:
 - `cbank charges -p <projectname>` # list all charges against a particular project
 - `cbank allocations -p <projectname>` # list all active allocations for a particular project
 - Other useful options:
 - `-u <user>` : show info for specific user(s)
 - `-a <YYYY-MM-DD>` : show info after date (inclusive)
 - `-b <YYYY-MM-DD>` : show info before date (exclusive)
 - `--help`

Note: cbank is updated once a day, at midnight (CDT).

- Charges are based on the partition size, **NOT the number of nodes or cores used!**

<http://www.alcf.anl.gov/user-guides/querying-allocations-using-cbank>





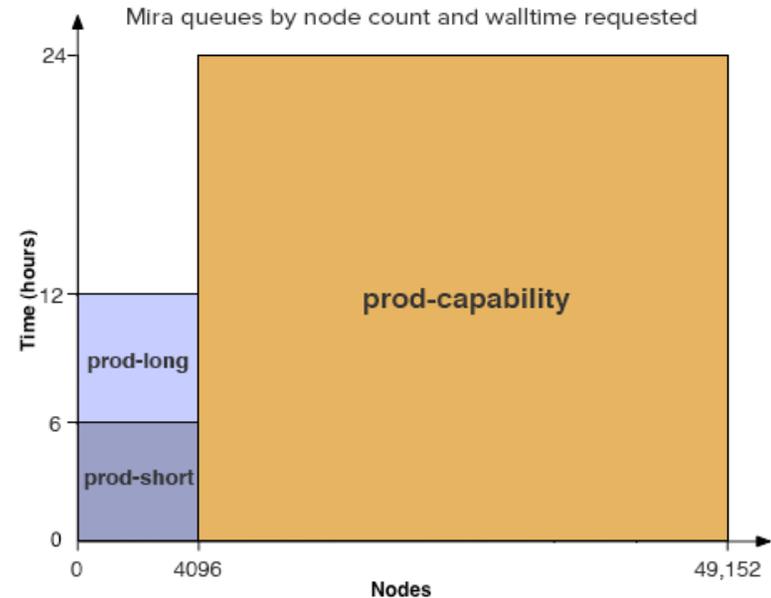
Section:

Queuing and Running

Mira Job Scheduling

- **Restrictions in queues**
 - 'prod-long' restricted to the row 0.
 - 'prod-short', 'prod-capability' can run in the full machine.
- **I/O to Compute node ratio 1:128**

<http://www.alcf.anl.gov/user-guides/job-scheduling-policy-bgg-systems>



User Queue	Underlying Queue	Nodes	Wall-clock Time (hours)	Maximum Running per User	Maximum Queued per User
prod	prod-short	512 - 4096	0 - ≤6	5	20
	prod-long	512 - 4096	>6 - 12	5	20
	prod-capability	4097 - 49152	0 - 24	5	20
	backfill (*)	512 - 8192	0 - 6	5	20
prod-1024-torus	prod-1024-torus	1024	0 - 12	5	16
prod-32768-torus	prod-32768-torus	32768	0 - 24	1	20

(*) This queue is automatically selected if a project's allocation is negative.



Cetus Job Scheduling

User Queue	Partition Sizes in Nodes	Wall-clock Time (hours)	Maximum Running per User	Maximum Queued per User
default, low	128, 256, 512, 1024	0 - 1	5	20

Cetus scheduling is designed to support application testing and debugging not production work.

- **I/O to Compute node ratio 1:128**

Vesta Job Scheduling

User Queue	Partition Sizes in Nodes	Wall-clock Time (hours)	Maximum Running per User	Maximum Queued Node-hours
default	32, 64, 128, 256, 512, 1024	0 - 2	5	1024
singles	32, 64, 128, 256, 512, 1024	0 - 2	1	1024
low	32, 64, 128, 256, 512, 1024	0 - 2	None	4096

- **I/O to Compute node ratio 1:32**

<http://www.alcf.anl.gov/user-guides/job-scheduling-policy-bggq-systems>



Mira job boot times

- Each time a job is submitted using a standard qsub command, all nodes in a partition are rebooted.
- Boot times depend on the size of the partition:

Nodes in Partition	Boot time (minutes)
≤ 2048	1
4096	1.5
8192	3
16384	4
32768	6
49152	7

The scheduler will attempt to boot the block up to three times if the boot procedure fails, so it might take as much as three times as long under rare circumstances.



Cobalt resource manager and job scheduler

- Cobalt is a resource management used on all ALCF systems
 - *Similar to PBS but not the same*
- Job management commands:
 - `qsub`: submit a job
 - `qstat`: query a job status
 - `qdel`: delete a job
 - `qalter`: alter batched job parameters
 - `qmove`: move job to different queue
 - `qhold`: place queued (non-running) job on hold
 - `qrls`: release hold on job
 - `qavail`: list current backfill slots available for a particular partition size.
- For reservations:
 - `showres`: show current and future reservations
 - `userres`: release reservation for other users



qsub options

- Syntax:

```
qsub [-d] [-v] -A <project name> -q <queue> --cwd <working directory>  
      --env envvar1=value1:envvar2=value2 --kernel <kernel profile>  
      -K <kernel options> -O <outputprefix> -t time <in minutes>  
      -e <error file path> -o <output file path> -i <input file path>  
      -n <number of nodes> -h --proccount <processor count>  
      --mode <mode> -M <email> --dependencies <jobid1>:<jobid2> <command> <args>
```

- Standard options:

-A project	project to charge
-q queue	queue
-t <time_in_minutes>	required runtime
-n <number_of_nodes>	number of nodes
--proccount <number_of_cores>	number of CPUs
--mode <script interactive cX>	running mode
--env VAR1=1:VAR2=1 <command> <args>	environment variables command with arguments
-O <output_file_prefix>	prefix for output files (default <code>jobid</code>)
-M <email_address>	e-mail notification of job start, end
--dependencies <jobid1>:<jobid2>	set the dependencies for the job being submitted
-l or --interactive	run and interactive command

Further options and details may be found in the man pages (> `man qsub`) or may be found at:

<http://trac.mcs.anl.gov/projects/cobalt/wiki/CommandReference>



qsub: Examples of submitting a job

- Despite being redundant, we recommend to always specify the number of nodes, the number of processes (MPI ranks), and the mode of your run.
- **Basic Method:** `qsub -A myproject -t 10 -n 512 --mode c1 Hello`
 - submits a job to the default queue
 - will run no longer than 10 minutes or when executable stops
 - will use 512 nodes with one MPI process per node
- **Script Method:** `qsub myscript.sh`

```
#!/bin/bash
#COBALT -A myproject -t 10 -n 1024 -O My_Run
runjob --block $COBALT_PARTNAME --np 16384 -p 16 : My_Exe My_Arg
```

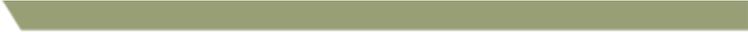
 - submits a job to the default queue and run no longer than 10 minutes
 - will use 1024 nodes, running 16K MPI processes, 16 on each node
 - will run program My_Exe with argument My_Arg
 - will create My_Run.output as stdout and My_Run.error as stderr files
 - See "man runjob" for other options



Advanced runs using script mode

- Multiple (consecutive) runs in a single job
- Multiple simultaneous runs in a single job
- Combinations of the above
- See:
 - <http://www.alcf.anl.gov/user-guides/cobalt-job-control>
 - <http://trac.mcs.anl.gov/projects/cobalt/wiki/BGQUserComputeBlockControl>





Part II



Section:

After your job is submitted

qstat: Show Status of a Batch Job(s)

- `qstat` # list all jobs

```
JobID  User  WallTime  Nodes  State  Location
=====
301295 smith  00:10:00  16    queued  None
```

- About jobs

- JobID is needed to kill the job or alter the job parameters
- Common states: `queued`, `running`, `user_hold`, `maxrun_hold`, `dep_hold`, `dep_fail`

- `qstat -f <jobid>` # show more job details
- `qstat -fl <jobid>` # show all job details
- `qstat -u <username>` # show all jobs from <username>
- `qstat -Q`
 - Instead of jobs, this shows information about the queues
 - Will show all available queues and their limits
 - Includes special queues, which we use to handle reservations



Machine status web page

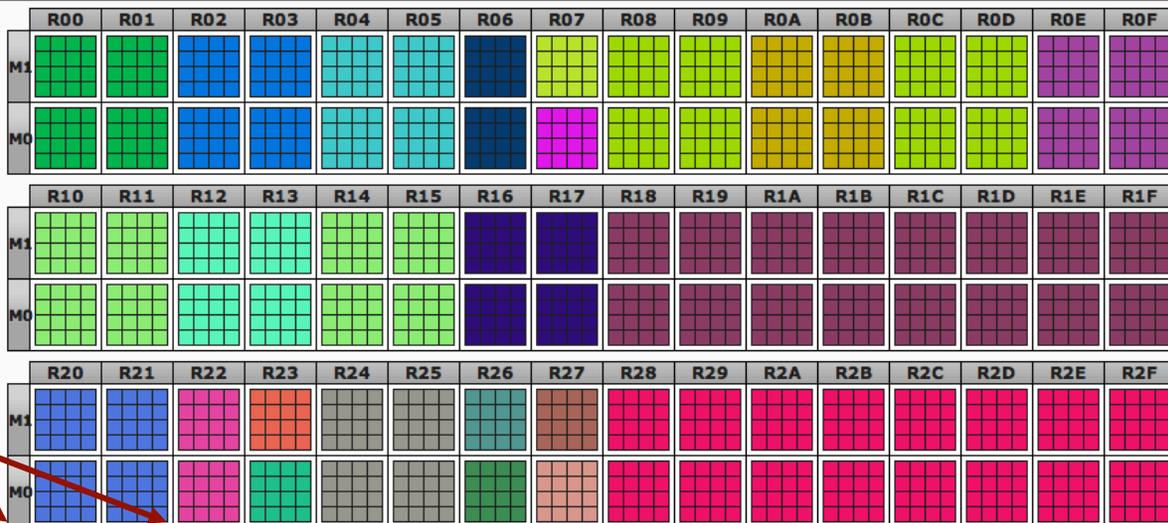


Leadership
Computing
Facility

Mira Activity

4096 nodes

Home Mira Activity



Running Jobs
Queued Jobs
Reservations

Job Id	Project	Score	Walltime	Queued Time	Queue	Nodes	Mode
Total Queued Jobs: 129							
191934	ClimEndStation	5000.1	01:30:00	00:26:28	prod-short	1816	script
190629	AbInitioC12_esp	1601.9	07:00:00	3d 20:07:53	prod-capability	8192	c4
190755	THModeling	1226.6	12:00:00	3d 12:08:00	prod-capability	8192	c32
190542	ClimEndStation	815.1	1d 00:00:00	4d 01:27:00	prod-capability	8192	script
189440	PolyNanoTrans	787.2	1d 00:00:00	6d 21:44:02	prod-capability	8192	c1
189442	PolyNanoTrans	785.7	1d 00:00:00	6d 21:37:15	prod-capability	8192	c1
189443	PolyNanoTrans	785.2	1d 00:00:00	6d 21:34:52	prod-capability	8192	c1

<http://status.alcf.anl.gov/mira/activity> (beta, a.k.a. The Gronkulator)

Machine status web page



Leadership
Computing
Facility

Cetus Activity

Home Cetus Activity

	R00			
M1				
M0				

Running Jobs Queued Jobs Reservations

Total Running Jobs: 3

Job Id	Project	Run Time	Walltime	Location	Queue	Nodes	Mode
192038	WallModJet	00:31:19	01:00:00	CET-00040-33371-512	default	512	c4
192067	NucStructReact	00:16:35	00:30:00	CET-20000-31331-128	default	32	c4
192071	THModeling	00:00:22	01:00:00	CET-00000-11331-128	default	128	c4

<http://status.alcf.anl.gov/cetus/activity>



Leadership
Computing
Facility

Vesta Activity

Home Vesta Activity

	R00		R01	
M1				
M0				

Running Jobs Queued Jobs Reservations

Total Running Jobs: 3

Job Id	Project	Run Time	Walltime	Location	Queue	Nodes	Mode
148459	Excited_States_CNTs	01:06:53	02:00:00	VST-02440-13751-64	default	40	c16
148460	Performance	01:04:27	02:00:00	VST-22460-33571-32	default	32	script
148461	Excited_States_CNTs	00:56:10	02:00:00	VST-02660-13771-32	default	32	c16

<http://status.alcf.anl.gov/vesta/activity>



Machine status web page

Home Tukey Activity

Rack 0	Rack 1	Rack 2	Rack 3

Running Jobs Queued Jobs Reservations

Total Running Jobs: 4

Job Id	Project	Run Time	Walltime	Location
84656	visualization	04:15:51	06:00:00	vs32
84815	CFDAnisotropic_esp	01:58:16	02:00:00	vs1,[vs3-4],vs6,vs8,[vs10-11],[vs13-14],[vs19-20],[vs22-24],vs26,vs28
84976	visualization	00:05:20	00:10:00	vs2,vs5,vs7,vs12,[vs17-18],vs21,vs25,[vs29-31],[vs38-39],[vs41-42],vs44
84977	MADNESS_MPQC_esp	00:04:59	00:10:00	vs16,vs51

<http://status.alcf.anl.gov/tukey/activity>

Cobalt files for a job

- Cobalt will create 3 files per job, the basename `<prefix>` defaults to the jobid, but can be set with “qsub -O myprefix”.
- Cobalt log file: `<prefix>.cobaltlog`
 - first file created by Cobalt after a job is submitted
 - contains submission information from qsub command, runjob, and environment variables
- Job stderr file: `<prefix>.error`
 - created at the start of a job
 - contains job startup information and any content sent to standard error while the user program is running
- Job stdout file: `<prefix>.output`
 - contains any content sent to standard output by user program



qdel: Kill a Job

- `qdel <jobid1> <jobid2>`
 - delete the job from a queue
 - terminated a running job



qalter, qmove: Alter Parameters of a Job

- Allows to alter the parameters of queued jobs without resubmitting
 - *Most parameters may only be changed before the run starts*
- Usage: `qalter` [options] <jobid1> <jobid2> ...
- Example:
 - > `qalter -t 60 123 124 125`
(changes wall time of jobs 123, 124 and 125 to 60 minutes)
- Type '`qalter -help`' to see full list of options
- `qalter` cannot change the queue; use `qmove` instead:
 - > `qmove <destination_queue> <jobid>`



qhold, qrls: Holding and Releasing

- **qhold** - Hold a submitted job (will not run until released)
`qhold <jobid1> <jobid2>`
- To submit directly into the hold state, use `qsub -h`
- **qrls** - Release a held job (in the *user_hold* state)
`qrls <jobid1> <jobid2>`
- Jobs in the *dep_hold* state may be released by removing the dependency
`qrls --dependencies <jobid>`
or `qalter --dependencies none <jobid>`
- Jobs in the *admin_hold* state may only be released by a system administrator



Possibilities why a job is not running yet

- There is a reservation, which interferes with your job
 - `showres` shows all reservations currently in place
- There are no available partitions for the requested queue
 - `partlist` shows all partitions marked as functional
 - `partlist` shows the assignment of each partition to a queue

Name	Queue	State	//
=====			//
...			//
MIR-04800-37B71-1-1024	prod-short:backfill	busy	//
MIR-04880-37BF1-1-1024	prod-short:backfill	blocked (MIR-048C0-37BF1-512)	//
MIR-04C00-37F71-1-1024	prod-short:backfill	blocked (MIR-04C00-37F31-512)	//
MIR-04C80-37FF1-1-1024	prod-short:backfill	idle	//
...			//

- Job submitted to a queue which is restricted to run at this time



Optimizing for queue throughput

- Small ($\leq 4K$), long ($6h < \text{time} < 12h$) jobs submitted to `prod` will be redirected to `prod-long`, which is restricted to row 0.
- Consider instead:
 - Small ($\leq 4K$), short ($\geq 6h$) jobs in `prod` queue will be redirected to `prod-short`, which can run anywhere.
 - Large ($> 4K$) jobs in `prod` queue will be redirected to `prod-capability`, which can run anywhere.
- Shotgun approach
 - If your code is amenable, submit a mix of job sizes and lengths.
- Check for drain windows
 - “`partlist | grep idle`”
 - full partlist output:

Name	Queue	State	Backfill	Geometry
...				
MIR-04800-37B71-1-1024	prod-short:backfill	busy	-	4x4x4x8x2
MIR-04880-37BF1-1-1024	prod-short:backfill	blocked (MIR-048C0-37BF1-512)	-	4x4x4x8x2
MIR-04C00-37F71-1-1024	prod-short:backfill	blocked (MIR-04C00-37F31-512)	-	4x4x4x8x2
MIR-04C80-37FF1-1-1024	prod-short:backfill	idle	0:49	4x4x4x8x2
...				

*In this case, a job submitted for 1024 nodes can run immediately if its time is < 49 minutes
(might need to be a few minutes shorter to allow for scheduling delay)*



Section:

Potential Problems

When things go wrong... Logging in

- Check to make sure it's not maintenance:
 - Often login nodes on both BG/Q and data analytics systems are closed off during maintenance to allow for activities that would impact users
 - There should be a mention in the weekly maintenance announcement and the pre-login banner message
 - An all-clear will be sent out at the close of maintenance
- Remember that CRYPTOCard passwords:
 - Require a pin at the start
 - Are all hexadecimal characters (0-9, A-F). *Letters are all **UPPER CASE**.*
- On failed login, try in this order:
 - Just try typing PIN+password again (without generating new password).
 - Try a different ALCF host to rule out login node issues (e.g. maintenance)
 - Push CRYPTOCard button to generate new password and try that
 - Walk through the unlock and resync steps at:
<http://www.alcf.anl.gov/user-guides/using-cryptocards#troubleshooting-your-cryptocard>
 - *Still can't login in?*
 - Connect with **ssh -vvv** and record the output, your IP address, hostname, and the time that you attempted to connect.
 - Send this information in your e-mail to support@alcf.anl.gov



When things go wrong... running

- Cobalt jobs, by default, produce three files (*.cobaltlog, *.error, *.output)
- Only *.cobaltlog is generated at submit time, the others at runtime
- Boot status (successful or not) written to *.cobaltlog
- After booting, the *.error file will have a non-zero size:
 - *Note: If your script job redirects the stderr of cobalt-mpirun, it will not end up in the job's .error file*
- If you think there is an issue, it's best to save all three files:
 - Send the jobid, machine name and a copy of the files to support@alcf.anl.gov



When things go wrong... running

- You'll see **RAS** events appear in your .error file it's not always the sign of trouble:
 - RAS stands for Reliability, Availability, and Serviceability
- Few are a sign of a serious issue, most are system noise:
 - Messages have a severity associated with them:
 - INFO
 - WARN
 - ERROR
 - FATAL
 - Only **FATAL** RAS events will lead to the termination of your application
 - ***ERROR may degrade performance but do NOT kill your job.***
 - Still worth watching as they may be the sign of an application performance issue
- If you run exits abnormally, the system will list the last RAS event encountered in the run. ***This RAS event did not necessarily cause the run to die.***



Core Files

- Jobs experiencing fatal errors will generally produce a core file for each process
- Examining core files:
 - Core files are in text format, readable with the “more” command
 - `bgq_stack` command provides call stack trace from a core file:
 - Ex: `bgq_stack <corefile>`
 - Command line interface (CLI)
 - Can only examine one core file at a time

<http://www.alcf.anl.gov/user-guides/bgqstack>
 - `coreprocessor.pl` command provides call stack trace from multiple cores:
 - Ex: `coreprocessor.pl -c=<directory_with_core_files> -b=a.out`
 - CLI and GUI. GUI interface requires X11 forwarding (`ssh -X mira.alcf.anl.gov`)
 - Provides information from multiple core files

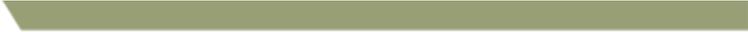
<http://www.alcf.anl.gov/user-guides/coreprocessor>
- Environment variables control core dump behavior:
 - `BG_COREDUMPONEXIT=1`: creates a core dump when the application exits
 - `BG_COREDUMPDISABLED=1`: disables creation of any core files



Reservations

- Reservations allow exclusive use of a partition for a specified group of users for a specific period of time
 - A reservation blocks other users jobs from running on that partition
 - Often used for system maintenance or debugging
 - **R.pm** (preventive maintenance), **R.hw*** or **R.sw*** (addressing HW or SW issues)
 - Reservations are sometimes idle, but still block other users jobs from running on a partition
 - Should be the exception not the rule
- Requesting
 - See: <http://www.alcf.anl.gov/user-guides/reservations>
 - Email reservation requests to **support@alcf.anl.gov**
 - View reservations with **showres**
 - Release reservations with **userres**
- When working with others in a reservation, these qsub options are useful:
 - **--run_users <user1>:<user2>:...** All users in this list can control this job
 - **--run_project <projectname>** All users in this project can control this job





Section:

Performance Tuning



Tools: Performance, Profiling, Debugging ...

- Non-system libraries and tools are under the /soft directory:
 - **/soft/applications** – Applications
 - LAMMPS, NAMD, QMCPACK, etc.
 - **/soft/buildtools** - Build tools
 - autotools, cmake, doxygen, etc.
 - **/soft/compilers** - IBM Compiler versions
 - **/soft/debuggers** - Debuggers
 - DDT, Totalview
 - **/soft/libraries** - Libraries
 - ESSL, PETSC, HDF5, etc.
 - **/soft/perftools** – Performance Tools
 - TAU, HPCToolkit, PAPI, OpenSpeedshop, Scalasca, HPM, etc.

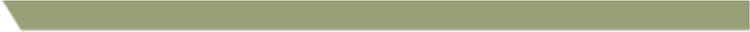


MPI Mapping

- A mapping defines the assignment of MPI ranks to BG/Q processors
- Default mapping is *ABCDET*
 - (*ABCDE*) are 5D torus coordinates, *T* is a CPU number
 - Right most letter of the mapping increases first as processes are distributed (T then E)
- Mappings may be specified by user using the `RUNJOB_MAPPING` environment variable:
 - With a mapping string:
 - `qsub --env RUNJOB_MAPPING=TEDACB --mode c32...`
 - String may be any permutation of ABCDET
 - E dimension of torus is always of size 2
 - With a mapping file:
 - `qsub --env RUNJOB_MAPPING=<FileName> --mode c32...`
 - mapfile: each line contains 6 coordinates to place the task, first line for task 0, second line for task 1...
 - allows for use of any desired mapping
 - file must contain one line per process and not contain conflicts (no verification)
 - use high-performance toolkits to determine communication pattern

<http://www.alcf.anl.gov/user-guides/machine-partitions>





Section:

Backups, Disk Storage and Tape Archival

Backups, Disk Storage and Tape Archival

▪ Backups

- On-disk snapshots of **/home** directories are done nightly
 - Check `~/.snapshots` if you delete a file accidentally
- **Only home directories** are backed up to tape
- *Data directories are not backed up:*

▪ Disk Storage

- **/home** directories:
 - Default of **100 GB** for Mira and **50 GB** for Vesta
 - Check your quota with the `'myquota'` command
- **/projects** directories:
 - Default of **1000 GB** for Mira and **500 GB** for Vesta
 - Check the quota in your projects with the `'myprojectquotas'` command
- See <http://www.alcf.anl.gov/user-guides/data-policy#data-storage-systems>

▪ Manual Data Archiving to Tape (HPSS)

- HSI is an interactive client
- GridFTP access to HPSS is available
- See <http://www.alcf.anl.gov/user-guides/using-hpss>



Getting Help

Online resources (24/7):

- ALCF web pages:
 - <http://www.alcf.anl.gov>
 - <http://www.alcf.anl.gov/user-guides>
 - <https://accounts.alcf.anl.gov>
- Mira/Cetus/Vesta/Tukey status (a.k.a. The Gronkulator):
 - <http://status.alcf.anl.gov/{mira,cetus,vesta,tukey}/activity>

Contact us:



e-mail: support@alcf.anl.gov



ALCF Help Desk:

Hours: Monday – Friday, 9 a.m. – 5 p.m. (Central time)

Phone: **630-252-3111** or **866-508-9181** (toll-free, US only)



Your Catalyst

News from ALCF:

- ALCF Weekly Updates, ALCF newsletters, *{mira,cetus,vesta,tukey}-notify* lists, etc.



Section:

Hands-on Session

GSW14 Hands-on session

- Log into Cetus or Vesta:

> ssh username@cetus.alcf.anl.gov or > ssh username@vesta.alcf.anl.gov

<http://www.alcf.anl.gov/user-guides/connect-log>

- Select MPI Wrapper Scripts:

- The default user environment does not provide a set of MPI compiler wrappers. A wrapper may be selected by using the appropriate SoftEnv key.
- For the following examples, ensure that you have the +mpiwrapper-xl key uncommented and that it is located before the @default line.

```
> cat ~/.soft
```

```
+mpiwrapper-xl
```

```
@default
```

```
> resoft
```

Note: after editing your ~/.soft file, run the command “resoft” to refresh your environment.

<http://www.alcf.anl.gov/user-guides/overview-how-compile-and-link>



GSW14 Hands-on session

- GSW14 **queue** and **project** name: `-q R.gsw14 -A ALCF_Getting_Started`
- **Compilation** example:
 - Create directory, copy files, and compile program:
 - > `mkdir -p ~/GSW14/Hands-on/Example_0`
 - > `cp /gpfs/mira-home/yuri/Hands-on/* ~/GSW14/Hands-on/Example_0`
 - or > `cp /gpfs/vesta-home/yuri/Hands-on/* ~/GSW14/Hands-on/Example_0`
 - > `cd ~/GSW14/Hands-on/Example_0`
 - > `ls`
 - > `cat hello_mpi.cpp`
 - > `cat Makefile`
 - > `make`
 - Submit job and check output:
 - > `cat run`
 - Run:
 - > `./run`
 - > `qstat -u <username>`
 - > `cat hello_mpi.output`



GSW14 Hands-on session

- Example of **subblocks**:

- Copy directory:

- > cp -r /soft/cobalt/examples/Example_1 ~/GSW14/Hands-on/

- > cd ~/GSW14/Hands-on/Example_1

- Open the README file and follow the instructions to submit a 1-rack job with two 512 blocks.

***NOTE:** remember to adapt the number of nodes and the block sizes provided in this example to the min./max. partition sizes available in the machine where you want to run the test (see slides 17 & 19 for reference).*

- Example of **interactive job** submission:

- Copy directory:

- > cp -r /soft/cobalt/examples/Example_2 ~/GSW14/Hands-on/

- > cd ~/GSW14/Hands-on/Example_2

- Open the README file and follow the instructions to submit an interactive job.

- Example of **python job** submission:

- Copy directory:

- > cp -r /soft/cobalt/examples/Example_3 ~/GSW14/Hands-on/

- > cd ~/GSW14/Hands-on/Example_3

- Open the README file and follow the instructions to submit an python job.

