ALCF System Architectures - Software and Job Submission

Christopher Knight
Catalyst Team
Outline

https://www.alcf.anl.gov/user-guides

• Mira (Blue Gene Q)
  • System Overview
  • Software & SoftEnv
  • Building your code
  • Queuing and running jobs with qsub & runjob

• Theta (KNL)
  • System Overview
  • Software & Environment Modules
  • Building your code
  • Queuing and running jobs with qsub & aprun

• Tips for troubleshooting
Mira - System Overview

https://www.alcf.anl.gov/mira

Chip
16+2 cores

Module
Single chip

Compute card
One single chip module
16 GB DDR3 Memory
Heat Spreader for H₂O Cooling

Node board
32 compute cards, optical Modules, link chips; 5D Torus

Midplane
16 node boards

Rack
1 or 2 midplanes
0, 1, 2, or 4 I/O drawers

Multi-rack system
*Mira*: 48 racks, 10 PF/s

81 inches tall
Mira - System Overview

https://www.alcf.anl.gov/user-guides/torus-network-bgq-system

- 5D torus network
  - High nearest neighbor bandwidth while increasing bisection bandwidth and reducing hops vs 3D torus
  - Machine can be partitioned into independent sub-networks
  - Hardware assists for collective and barrier functions over COMM_WORLD and rectangular sub communicators
  - Single network used for P2P, collectives, and barriers.

- Nodes have 10 links with 2 GB/s raw bandwidth each
  - Bi-directional: send & receive gives 4 GB/s
  - 90% bandwidth available to user
  - Additional 11th link for I/O
Mira - Software & Libraries

https://www.alcf.anl.gov/user-guides/software-and-libraries

- IBM system and provided libraries: /bgsys/drivers/ppcfloor
  - glibc
  - mpi
  - PAMI (Parallel Active Messaging Interface)

- Site-supported libraries: /soft/libraries
  - ESSL, PETSc, HDF5, netCDF, Parallel netCDF, Boost
  - ESSL is IBM’s optimized Engineering and Scientific Subroutine Library for BG/Q:
    - BLAS, LAPACK, FFT, sort/search, interpolation, quadrature, random number, BLACS

- Additional tuned libraries: /soft/libraries/alcf
  - BLAS, FFTW, LAPACK, PARMETIS, PARPACK, SCALAPACK, SZIP, ZLIB
Mira - Software & Libraries

https://www.alcf.anl.gov/user-guides/software-and-libraries

- Applications: /soft/applications
  - LAMMPS, NAMD, QMCPACK, CP2K, etc…

- Build Tools: /soft/buildtools
  - autotools, cmake, doxygen, etc…

- Compilers: /soft/compilers
  - IBM XL, BGCLANG, GNU

- Debuggers: /soft/debuggers
  - DDT

- Performance Tools: /soft/perftools
  - TAU, HPCToolkit, PAPI, Autoperf, Scalasca, HPCTW, etc…
Mira - SoftEnv


• A tool for managing a user’s environment
  • Sets your paths 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
  • Use .bash_profile for user-specific environment modifications

• Commands
  • List all keywords defined on the system: softenv
  • Reload initial environment from ~/.soft file: resoft
  • Temporarily modify environment: soft add|remove <keyword>
Mira - Compiler Wrappers

https://www.alcf.anl.gov/user-guides/compiling-and-linking-bq

- IBM XL cross-compilers
  - SoftEnv key: `+mpiwrapper-xl`
  - `mpixlc_r, mpixlcxx_r, mpixlf77_r, mpixlf90_r, mpixlf95_r`, etc...
  - List complete command executed by mpi wrapper: `--show`

- BGCLANG cross-compilers
  - SoftEnv key: `+mpiwrapper-bgclang`
  - `mpiclang, mpiclang++, mpiclang++11`

- GNU cross-compilers
  - SoftEnv key: `+mpiwrapper-gcc`
  - `mpicc, mpicxx, mpif77, mpif90`
Mira - SoftEnv Example


- A minimal ~/.soft file needed to build code for compute nodes
  
  > cat ~/.soft
  
  +mpiwrapper-xl
  
  @default

  > resoft
  
  > mpixlc_r -qversion

  IBM XL C/C++ for Blue Gene, V12.1
  Version: 12.01.0000.0015

- Remember, after editing ~/.soft file, run resoft command to refresh environment.
Mira - IBM XL Optimization Tips


• Optimization level
  • Best for debugging: –O0
  • Good for correctness check, baseline performance: –O2
  • Loop transformations & additional optimizations: –O3
    • Can alter program semantics unless used with –qstrict

• Tips
  • Performance can decrease at higher levels: –O4 or –O5
  • Use –qlistopt to generate a listing of all flags used in compilation
  • Use –qreport to generate a listing showing how code was optimized
  • Compiler option –g must be used to resolve code line numbers
Mira - Threading

https://www.alcf.anl.gov/user-guides/threading-bgq

- OpenMP
  - IBM XL compilers: -qsmp=omp:noauto
  - GNU: -fopenmp
  - BGCLANG: -fopenmp

- Pthreads
  - NPTL Pthreads implementation in glibc requires no modification

- Tips
  - Auto thread parallelization with -qsmp=auto not always effective
  - Number of threads controlled with runjob command
  - Each core needs at least 2 (possibly 4) threads for peak efficiency
Mira - Preparing to Submit Job

https://www.alcf.anl.gov/user-guides/allocation-accounting-sbank

- Check that you are a member of a project: projects

- Check available disk space
  - $HOME directory: myquota
  - Project directories: myprojectquotas
  - Project directories should be used for production work

- Check that your project has core-hours available
  - Use sbank command to query allocation details
  - Allocation available to project: sbank l a –p <project_name>
  - Charges against project by user: sbank l u –p <project_name> –u <user>
  - Charges on BG/Q are based on partition size, not number of nodes
Mira - Cobalt

https://www.alcf.anl.gov/user-guides/cobalt-job-control

- Resource management software on all ALCF systems
  - Similar to PBS

- Job management commands
  - Submit a job: qsub
  - Query job status: qstat
  - Delete a job: qdel
  - Alter job parameters: qalter
  - Move job to different queue: qmove
  - Place queued job (non-running) on hold: qhold
  - Release hold on job: qrls

- Examples in /soft/cobalt/examples
Mira - qsub

https://www.alcf.anl.gov/user-guides/cobalt-job-control

• Standard options
  • Project to charge: --A <project_name>
  • Queue: --q <queue>
  • Maximum walltime: --t <time_in_minutes>
  • Number of nodes: --n <number_of_nodes>
  • Number of processes: --proccount <number_of_processors>
  • Running mode: --mode <cX | script>
  • Environment variables: --env <VAR1=1:VAR2=1>
  • Prefix for output files: --O <file_prefix>
  • E-mail notifications: --M <email_address>
  • Dependencies: --dependencies <jobid1>:<jobid2>
  • Interactive job: --l or --interactive
Mira - Submitting Script Jobs

https://www.alcf.anl.gov/user-guides/cobalt-job-control

• Executable is invoked within script (bash, csh, …)

• Example #1 specifies job attributes on command-line

  > cat myscript.sh
  #!/bin/sh
  echo “Starting Cobalt job script”
  runjob --np 131072 --p 16 --block $COBALT_PARTNAME
  $COBALT_PARTNAME: <app> <app_args>

  > qsub –A <project_name> –t 10 –n 8192 –O <prefix_name> --mode script myscript.sh
  123456
Mira - Submitting Script Jobs

https://www.alcf.anl.gov/user-guides/cobalt-job-control

• Executable is invoked within script (bash, csh, …)

• Example #2 specifies job attributes within script

```bash
> cat myscript.sh
#!/bin/sh
#COBALT -A <project_name> -t 10 -n 8192 -O <prefix_name>
echo "Starting Cobalt job script"
runjob --np 131072 --p 16 --block $COBALT_PARTNAME --verbose=INFO
    --env OMP_NUM_THREADS=4 : <app> <app_args>

> qsub myscript.sh
123456
```
Mira - Cobalt Files For Submitted Job

https://www.alcf.anl.gov/user-guides/cobalt-job-control

• Cobalt will create three files per job
  • Prefix defaults to jobid if not set with qsub’s –O option

• Cobalt log file: <prefix_name>.cobaltlog
  • Create when job is submitted, additional info written while job runs
  • Contains submission information from qsub, runjob, and environment

• Job stderr file: <prefix_name>.error
  • Created at start of job
  • Contains job startup information and any output sent to standard error

• Job stdout file: <prefix_name>.output
  • Created at start of job
  • Contains content sent to standard output
Mira - Now That Your Job Is Queued

https://www.alcf.anl.gov/user-guides/cobalt-job-control

- Check status of submitted jobs

```bash
> qstat

<table>
<thead>
<tr>
<th>JobID</th>
<th>User</th>
<th>WallTime</th>
<th>Nodes</th>
<th>State</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>123456</td>
<td>user1</td>
<td>00:30:00</td>
<td>512</td>
<td>dep_fail</td>
<td>None</td>
</tr>
<tr>
<td>123457</td>
<td>user2</td>
<td>01:00:00</td>
<td>1024</td>
<td>running</td>
<td>CET-40400-73771-1024</td>
</tr>
<tr>
<td>123458</td>
<td>user3</td>
<td>00:30:00</td>
<td>2048</td>
<td>queued</td>
<td>None</td>
</tr>
<tr>
<td>123459</td>
<td>user1</td>
<td>01:00:00</td>
<td>512</td>
<td>running</td>
<td>CET-40040-73371-512</td>
</tr>
<tr>
<td>123460</td>
<td>user4</td>
<td>00:30:00</td>
<td>2048</td>
<td>user_hold</td>
<td>None</td>
</tr>
</tbody>
</table>
```
Mira - Now That Your Job Is Queued

https://www.alcf.anl.gov/user-guides/cobalt-job-control

• Additional qstat queries
  • Show more job details: `qstat -f <jobid>`
  • Show all job details: `qstat -fl <jobid>`
  • Show all jobs from user: `qstat -u <user>`
  • Show information about queues: `qstat -Q`

• Delete job from queue: `qdel <jobid>`

• Alter properties of queued job
  • Change walltime: `qalter -t <new_time> <jobid>`
  • Change number of nodes: `qalter -n <new_number_of_nodes> <jobid>`
  • Change queue: `qmove <new_queue> <jobid>`
Mira - Checking Status of Job

https://status.alcf.anl.gov/mira/activity
Mira - Optimizing For Queue Throughput

https://www.alcf.anl.gov/user-guides/job-scheduling-policy-bgq-systems

- Small (≤ 4096 nodes), long (6-12 hours) jobs redirected to prod-long queue, which is restricted to row 0

- Consider instead
  - Small (≤ 4096 nodes), short (≤ 6 hours) jobs redirected to prod-short queue, which run anywhere
  - Large (> 4096 nodes) jobs redirected to prod-capability queue, which run anywhere

- For long sequences of jobs, chain them together with dependencies
  - Dependent jobs inherit score boost from previous successful job in chain.
ANY QUESTIONS?
Theta - System Overview

https://www.alcf.anl.gov/theta

**System:** 24 Cabinets
4392 Nodes, 1152 Switches
Dual-plane, 12 groups, Dragonfly 12.1 TB/s Bi-Sec
11.7 PF Peak
70 TB MCDRAM, 843 TB DRAM

**Node:** KNL Socket
192 GB DDR4 (6 channels) **2.66 TF** 16GB MCDRAM
128 GB SSD

**Compute Blade:**
4 Nodes/Blade + Aries switch
**10.64 TF** 64GB MCDRAM
768GB DRAM

**Chassis:** 16 Blades, 16 Cards
64 Nodes, 16 Switches
**170.24 TF** 1TB MCDRAM, 12TB DRAM

**Cabinet:** 3 Chassis, 75kW liquid/air cooled
**510.72 TF** 3TB MCDRAM, 36TB DRAM

**Sonexion Storage**
4 Cabinets
Lustre file system
**10 PB usable**
210 GB/s
Two memory types
• In Package Memory (IPM)
  • 16 GB MCDRAM @ ~480 GB/s
• Off Package Memory (DDR)
  • 192 GB @ ~90 GB/s
• Single address space; multiple NUMA domains
• Memory configurations
  • Cached: DDR fully cached by IPM
  • Flat: User managed
  • Hybrid: ¼, ½, IPM used as cache
• Managing memory
  • jemalloc & memkind libraries
  • Pragmas for static memory allocations
# Theta - Cray Programming Environment

https://www.alcf.anl.gov/user-guides/software-and-libraries

## Programming Languages
- Fortran
- C
- C++
- Python

## Programming models
- Distributed Memory (Cray MPT)
  - MPI
  - SHMEM
  - GA
- Shared Memory
  - OpenMP 3.1
  - OpenACC 2.0
- PGAS
  - UPC
  - CAF
  - CoArray C++

## Compilers
- Cray Compiling Environment (CCE)
  - GNU
  - 3rd party compilers (Intel)

## Compiling Environment setup
- Modules

## Tools
- Debuggers
  - DDT
  - lgdb
- Debugging Tools
  - ATP
  - STAT
- Performance Analysis
- Cray Apprentice
- Porting Tools
  - Reveal
  - CCDB

## Optimized Scientific Libraries
- Dense
  - BLAS
- LAPACK
- ScaLAPACK
- Iterative Refinement Toolkit
- Sparse
  - Cray PETSc (with CASK)
  - Cray Trilinos (with CASK)
- FFT
  - FFTW

## I/O Libraries
- NetCDF
- HDF5
Theta - Non-system Software & Libraries

https://www.alcf.anl.gov/user-guides/software-and-libraries

• Compilers: /soft/compilers
  • llvm and intel beta releases

• Debuggers: /soft/debuggers
  • DDT

• Libraries: /soft/libraries
  • argobots, bolt, breakpad

• Performance tools: /soft/perftools
  • Darshan, HPCToolkit, memlog, TAU

• Visualization: /soft/visualization
  • Paraview, R
Theta - Modules

https://modules.sourceforge.net

- A tool for managing a user’s environment
  - Sets your PATH to access desired front-end tools
  - Your compiler version can be changed here

- Module commands
  - List available module commands: `module help`
  - List currently loaded modules: `module list`
  - List all available modules: `module avail`
  - Add module to environment: `module load <mod>`
  - Remove module from environment: `module unload <mod>`
  - Swap loaded module with new one: `module switch <mod_old> <mod_new>`
  - List information about module: `module show <mod>`
Theta - Compiler Wrappers

https://www.alcf.anl.gov/user-guides/compiling-and-linking-xc40

- For all compilers (Intel, Cray, GNU, Clang)
  - Use cc, CC, ftn
  - Do not use mpicc, mpiCC, mpic++, mpif77, mpif90, etc… as they do not generate code for compute nodes

- Select compiler you want: module swap <PrgEnv-old> <PrgEnv-new>
  - Intel (default): PrgEnv-intel
  - Cray: module swap PrgEnv-intel PrgEnv-cray
  - GNU: module swap PrgEnv-intel PrgEnv-gnu
  - Clang: module swap PrgEnv-intel PrgEnv-llvm

- Cray wrappers
  - List complete command executed: –craype-verbose
  - Disable automatic linking with libsci: –mkl
Theta - Submitting Script Jobs

https://www.alcf.anl.gov/user-guides/running-jobs-xc40

- Executable is invoked within script (bash, csh, …)

- `aprun` is used to launch executables on compute nodes

```bash
> cat myscript.sh
#!/bin/sh
#COBALT -- A <project_name> -- t 10 -- n 16 -- O <prefix_name> -- q default
#COBALT --attrs mcdram=cache:numa=quad
echo "Starting Cobalt job script"
aprun --n 1024 -- N 64 -- d 1 -- j 1 -- cc depth <app> <app_args>
```

```
> qsub myscript.sh
123456
```
Theta - aprun Overview

https://www.alcf.anl.gov/user-guides/running-jobs-xc40

• aprun options
  • Total number of MPI ranks: –n <total_number_ranks>
  • Number of MPI ranks per node: –N <number_ranks_per_node>
  • Number of hyperthreads per MPI rank (depth): –d <num_hardware_threads_per_rank>
  • Number of hyperthreads per core: –j <number_hardware_threads_per_core>
  • MPI rank and thread placement: --cc depth
  • Environment variables: –e <VAR1=1> –e <VAR2=1>
  • Core specialization: –r <number.hardware_threads>

• See also man aprun
Theta - aprun Overview

https://www.alcf.anl.gov/user-guides/running-jobs-xc40

- Theta’s KNL nodes have 32 tiles with 2 cores each (4 hardware threads per core)
- Example #1: 2 nodes, 64 ranks/node, 1 thread/rank, 1 rank/core
  - aprun –n 128 –N 64 –d 1 –j 1 --cc depth <app> <app_args>

```plaintext
nname= nid02937  rk= 0  td= 0  ht= {0}
nname= nid02937  rk= 1  td= 0  ht= {1}
nname= nid02937  rk= 2  td= 0  ht= {2}
nname= nid02937  rk= 3  td= 0  ht= {3}
nname= nid02937  rk= 4  td= 0  ht= {4}
nname= nid02937  rk= 5  td= 0  ht= {5}
```
Argonne Leadership Computing Facility

**Theta - aprun Overview**

https://www.alcf.anl.gov/user-guides/running-jobs-xc40

- Theta’s KNL nodes have 32 tiles with 2 cores each (4 hardware threads per core)
- Example #1: 2 nodes, 32 ranks/node, 4 thread/rank, 2 threads/core
  - `aprun -n 64 -N 32 -d 4 -j 2 --cc depth -e OMP_NUM_THREADS=4 <app> <app_args>`

```
nname= nid02937  rk= 0  tid= 0  ht= {0}
nname= nid02937  rk= 0  tid= 1  ht= {1}
nname= nid02937  rk= 0  tid= 2  ht= {64}
nname= nid02937  rk= 0  tid= 3  ht= {65}
nname= nid02937  rk= 1  tid= 0  ht= {2}
nname= nid02937  rk= 1  tid= 1  ht= {3}
```

...
 Theta - aprun Overview

https://www.alcf.anl.gov/user-guides/running-jobs-xc40

- Affinity
  - Use `--d` and `--cc depth` to let ALPS control affinity
  - Use `--cc none` if you want to use OpenMP (or KMP) env. variables to specify affinity

- Core specialization with `--r <number.hardware.threads>`
  - Offload OS and MPI to unused hardware threads (e.g. reduce variability)

- Allocating memory in flat mode
  - Default memory allocation in DDR (NUMA 0)
  - Only allocate memory to HBM (NUMA 1): `numactl --m 1`
  - Prefer memory allocation to HBM: `numactl --p 1`
  - Example: `aprun --n 128 --N 64 --d 1 --cc depth numactl --m 1 <app> <app_args>`
Theta - Queues

https://www.alcf.anl.gov/user-guides/job-scheduling-policy-xic40-systems

- Jobs are routed to single **default** queue
  - Nodes allocated to job will be rebooted (if needed) for requested memory mode
  - Best to always specify memory mode (e.g. --attrs mcdram=cache:numa=quad)
  - Pad requested walltime by ~30 minutes to account for possible rebooting
  - Don’t delete job if remains in “starting” for several minutes

- Wall-clock limits are function of number of requested nodes
  - minimum allocation: 8 (128) nodes, maximum walltime 2 (3) hours
  - capability jobs: >= 648 (802) nodes, maximum walltime 24 hours
  - Check website for current policies

- Two 16-node debug queues available
  - debug-cache-quad
  - debug-flat-quad
ANY QUESTIONS?
Why Hasn’t My Job Started?

• There is a reservation which delays your job from starting
  • List all reservations currently in place: `showres`

• There are no available nodes for the requested queue
  • Nodes may be down, busy running other jobs, draining next job, or reserved
  • Check queue status: `qstat`
  • Check machine status: `http://status.alcf.anl.gov`
  • Check “ALCF Weekly Updates” for training, reservation, and maintenance notices

• List idle resources
  • List status of partitions on Mira: `partlist`
  • List status of nodes on Theta: `nodelist`
Mira - Core Files and Debugging

https://www.alcf.anl.gov/user-guides/debugging-profiling

- Examining core files
  - Core files are in text format (e.g. readable with more command)
  - List call stack trace from single core file: bgq_stack <app> core.#
  - List call stack trace from multiple core files: coreprocessor.pl
    - Example: coreprocessor.pl –c=<dir> –b=<exe>
    - Can also connect to running job

- Environment variables
  - Create core dump when application exits: BG_COREDUMPONEXIT=1
  - Disables creation of any core files: BG_COREDUMPDISABLED=1

- Full-featured debugging with DDT
Theta - Core Files and Debugging

https://www.alcf.anl.gov/user-guides/debugging-profiling

- Abnormal Termination Processing (ATP)
  - Set environment variable ATP_ENABLED=1 in job script before aprun
  - Upon failure, generate merged stack backtrace tree in atpMergedBT.dot file
  - View output file with stat-view after loading with module load stat

- Notes on linking your application
  - PrgEnv-cray links everything necessary by default
  - PrgEnv-intel requires –Wl,–T/opt/cray/pc/cce/8.5.2/craylibs/x86-64/2.23.1.cce.ld

- Other debugging tools
  - You can generate STAT snapshots asynchronously
  - FullFeatured debugging with DDT
When Things Go Wrong Running…

https://www.alcf.anl.gov/user-support

• Examine core files

• Best to save all three files generated by cobalt
  • `<prefix_name>.cobaltlog`, `<prefix_name>.error`, and `<prefix_name>.output`

• Retain important information
  • Jobid, machine name, copy/location of all files, exact error message

• Contact us
  • Your ALCF contact
  • Email: support@alcf.anl.gov
  • Call the ALCF Help Desk
    • Hours: Monday-Friday, 9am-5pm CT
    • Phone: 630-252-3111 or 866-508-9181 (toll-free, US only)
HAPPY COMPUTING!