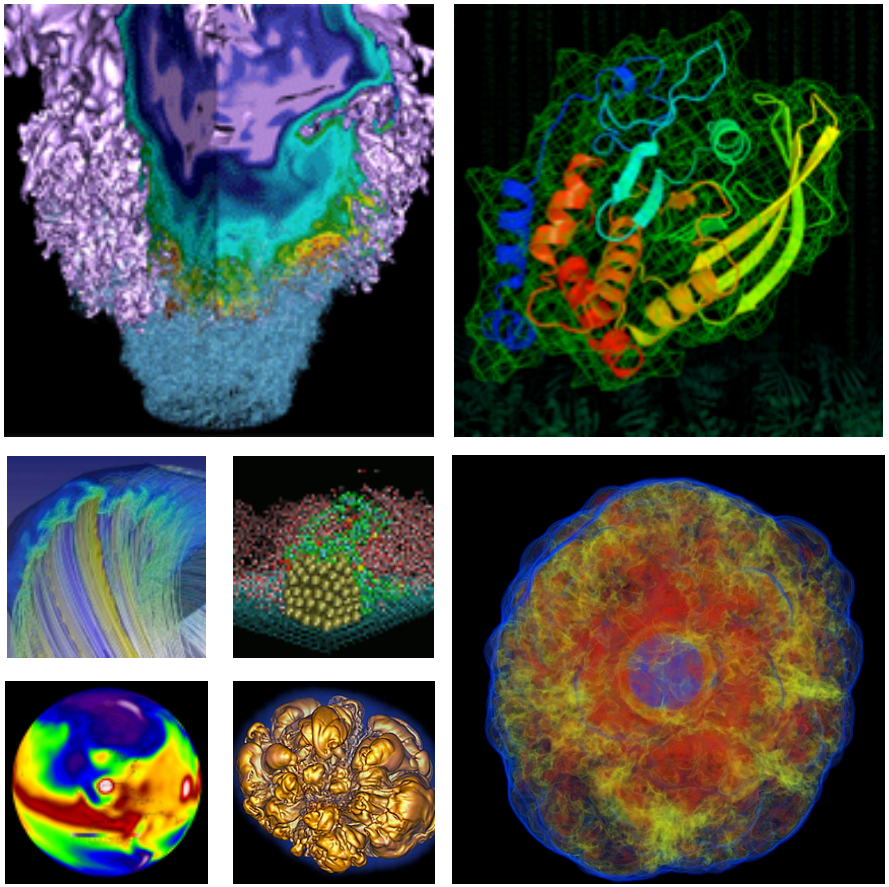
Moving from Hopper (Cray XE6) to Edison (Cray XC30)



Zhengji Zhao

NERSC User Services Group

NUG 2013, February 25, 2013

Edison-NERSC Newest HPC Machine

- **Easy to use**
  - Edison programming environment is remarkably similar to that of Hopper
- **Supports a rich set of production software applications, libraries, and tools needed by the entire NERSC workload**
  - A robust set of programming languages, models
  - A rich set of highly optimized libraries, tools and applications
  - Community and pre-packaged applications
  - Shared-object libraries and socket communication (CCM)
- **Enables effective application performance at scale, single node (high-throughput computing), and everything in between**

Edison Phase 1 Configuration

	Hopper	Edison Phase 1	Comments
Compute node	Two 12 core MagnyCours processors	Two 8 core Intel Sandy Bridge processors	Hyper-Threading is available on Edison
System size	6384 nodes, 153216 cores	664 nodes, 10624 cores	Smaller in size
Cores per node	24	16	
Clock speed	2.1 GHz	2.6GHz	Faster processor
Memory	32 GB DDR3 1300 MHz 1.3GB/core	64GB DDR3 1600 MHz 4GB/core	Faster, larger per core memory
NUMA Domain	4	2	

Edison Phase 1 configuration

	Hopper	Edison Phase 1	Comments
Interconnect	Gemini interconnect with 3D-Torus	Aries with dragonfly network topology	Next generation interconnect

	Hopper	Edison Phase 1	Comments
Login nodes	8 4 quad-core 2.4 GHz AMD Opteron 8378 processors (16 cores total); 4 8-core AMD 2.0 GHz Opteron 6128 processors (32 cores)	quad-core, quad-socket (16 total cores) 2.0 GHz Intel "Sandy Bridge" processors with 512 GB memory.	“Fatter” login nodes

Edison Phase 1 configuration

Lustre file system	Hopper	Edison Phase 1	Comments
Online Disk	2.2 PB with 70 GB/s bandwidth	1.6 PB with 36GB/s bandwidth	
IO Servers (OSSs)	52	18	
OSTs	312	72	
Storage system	LSI 7900	Cray Sonexion 1600	

Batch system	Hopper	Edison Phase 1	Comments
Torque/Moab	Internal batch server	External batch server	There are some advantages

New features and technologies available on Edison Compared to Hopper

- **Cray Aries interconnect with Dragonfly topology**
  - Scalable global bandwidth with lower network cost,
  - Adaptive routing
  - Designed to support both MPI and PGAS programing models.
- **Hyper-Threading Technology**
  - Some of the codes may get benefit from using HT
  - Benefit: time to solution vs charging
- **Cray Sonexion Storage System**
  - Cray's large-scale parallel storage solutions for HPC
- **External Batch Server**
  - Single batch server
  - Torque commands (eg., pbsnodes) and environment variables (eg., PBS_NODEFILE) work as they do on Carver.

Supported Programming Languages, Programming models, and compilers

Supported Programming Languages	Supported Programming Models	Supported Compilers
Fortran	MPI	Intel*
C, C++	OpenMP	Cray
UPC	Cray Shmem	GNU
Python, Perl, Shells	POSIX Threads	
Java	POSIX Shared Memory	
Chapel	UPC	
	Coarray Fortran	
	Chapel	

***) Intel compiler is the default compiler on Edison.**

PGI compiler is not supported on Edison.

Cray Scientific and Math Libraries

Libraries	Intel	Cray	GNU
LibSci: LAPACK, ScaLAPACK, BLACS, PBLAS		✔	✔
Third Party Scientific Libraries: MUMPS, SuperLU, ParMETIS, HYPRE, Scotch		✔	✔
Trilinos		✔	✔
FFTW	✔	✔	✔
PETSc		✔	✔
DMAPP API for Aries	✔	✔	✔
MPI-IO Library	✔	✔	✔
IO Libraries: HDF5, NetCDF, Parallel-netcdf	✔	✔	✔
Intel MKL	✔	✔	✔

More libraries, Tools, and Applications

Scalable Debuggers	Performance Tools	Libraries and Tools	Applications	Visualization tools	ACTS Libraries and tools
DDT	Craypat/ Apprentice2	Python	VASP	VisIt	
TotalView	PAPI	GSL	QUANTUM ESPRESSO		
ATP	IPM	NCAR	LAMMPS		
		ADIOS	NAMD		
			BerkelyGW		
			MOLPRO		
			G09		

Our plan is to support all commonly used software and tools available on Hopper and Edison – please request if you find anything is missing

How to compile-
Use compiler wrappers

- **Use module avail to see the available software**
- **For Cray and GNU programming environment, all Cray scientific and math libraries are available.**
  - Compile codes as you do on Hopper
- **Under Intel programming environment, some math and Cray scientific libraries are not available, noticeably libsci, use MKL for math routines**
  - ftn test_scalapack.f90 –mkl=cluster
- **There is a separate talk for compilation (Mike Stewart) to help you to make the transition from PGI to Intel.**

How to run-
Use the similar job script on Hopper

- **To run with 2 nodes:**

```
#PBS -N test
#PBS -q debug
#PBS -l mppwidth=32
#PBS -l walltime=0:30:00
#PBS -j oe

cd $PBS_O_WORKDIR
 aprun -n 32 ./a.out
```

Sample job script to run MPI+OpenMP jobs

- **To run an Intel binary a.out with 2 nodes:**

```
#!/bin/bash -l
#PBS -q debug
#PBS -l mppwidth=32
#PBS -l walltime=0:30:00
#PBS -j oe

cd $PBS_O_WORKDIR
export OMP_NUM_THREADS=8
export KMP_AFFINITY=compact
 aprun –n4 –N2 –S1 –d8 –cc numa_node –ss ./a.out
```

Process/memory/thread affinity on Edison

- Process/memory/thread affinity works fine under the Cray and GNU programming environment.
- For Intel programming environment, the Process/memory affinity works fine for pure MPI jobs, but the thread affinity does not work properly. The work around is to use the env KMP_AFFINITY now.
- If you need to run MPI_OpenMP jobs with any number of threads, please experiment with the aprun's –cc and –ss options and with env KMP_AFFINITY.
- Cray code xthi.c can be used for your test

What is Hyper-Threading (HT)?

- HT is Intel's term for its simultaneous multithreading implementation. For each physical processor core, the operating system addresses two virtual or logical cores, and schedules two threads or processes simultaneously (duplicating architectural state but sharing the execution resources).
- When the processor is stalled due to cache miss, branch misprediction, or data dependency, a HT equipped processor executes another scheduled task.
- According to Intel the first implementation only used 5% more die area than the comparable non-hyperthreaded processor, but the performance was 15–30% better.

How to run with HT

- **To run with 2 nodes:**

```
#PBS -q debug
#PBS -l mppwidth=64
#PBS -l mppnppn=32
#PBS -l walltime=0:30:00
#PBS -j oe

cd $PBS_O_WORKDIR
 aprun –j2 -n 64 ./a.out
```

Sample job script to run MPI+OpenMP jobs with HT

- **To run with 2 nodes:**

```
#!/bin/bash -l
#PBS -q debug
#PBS -l mppwidth=64
#PBS -l mppnppn=32
#PBS -l walltime=0:30:00
#PBS -j oe

cd $PBS_O_WORKDIR
export OMP_NUM_THREADS=16
export KMP_AFFINITY=compact
 aprun –j2 –n 4 –N2 –S1 –d16 –cc numa_node  -ss ./a.out
```

Edison Queues

- **Configuration is subject to changes**
- **Submit production jobs to regular queue.**
  - qstat –Qf to see the queue configuration detail
  - qstat –q to see if batch queues are enabled.
- **24 hours wall limit for now**

Open Issues

- **Intel programming environment**
  - Please report any missing items in the intel programming environment.
- **Batch system issue**
  - Opened bug with Adaptive working for a solution
- **IO performance**
  - Still don't meet the target, Cray is working on this.
- **DVS performance issue**
  - Do not do shared file IO, especially file read against the global file systems, eg., /gscratch, /project, etc.
- **Core affinity**
  - Intel binary thread affinity, work with KMP_AFFINITY to get the best thread affinity as you can.
  - In practice working with KMP_AFFINITY=disabled, and aprun … –cc numa_node or –cc none maybe preferred

Summary

- Edison has faster processors with a larger memory per core when compared to Hopper.
- The programming environment on Edison is very similar to that of Hopper, except Intel compiler is the default on Edison, and MKL is used.
- Uses same job script as on Hopper, just Edison has 16 cores per node.
- If you run with HT, Edison has 32 cores per node. Run with mppnppn=32 and aprun –j2. Experiment if HT benefits your codes.

Quick start for Hopper users

1. **ssh edison@nersc.gov -l YourUserName**
2. **ftn mycode.f90**
3. **Write job script**

```
zz217@hopper08:~> cat run.pbs
#PBS -q debug
#PBS -l mppwidth=32
#PBS -l walltime=0:05:00
#PBS -j oe

cd $PBS_O_WORKDIR
aprun -n 32 ./a.out
```

4. **qsub run.pbs**
5. **Analyze results**

National Energy Research Scientific Computing Center