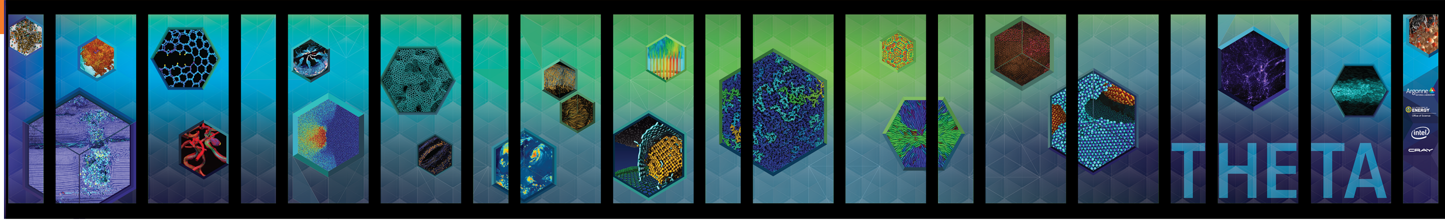
ALCF Theta/Cooley Quick Start

Part I : Theta

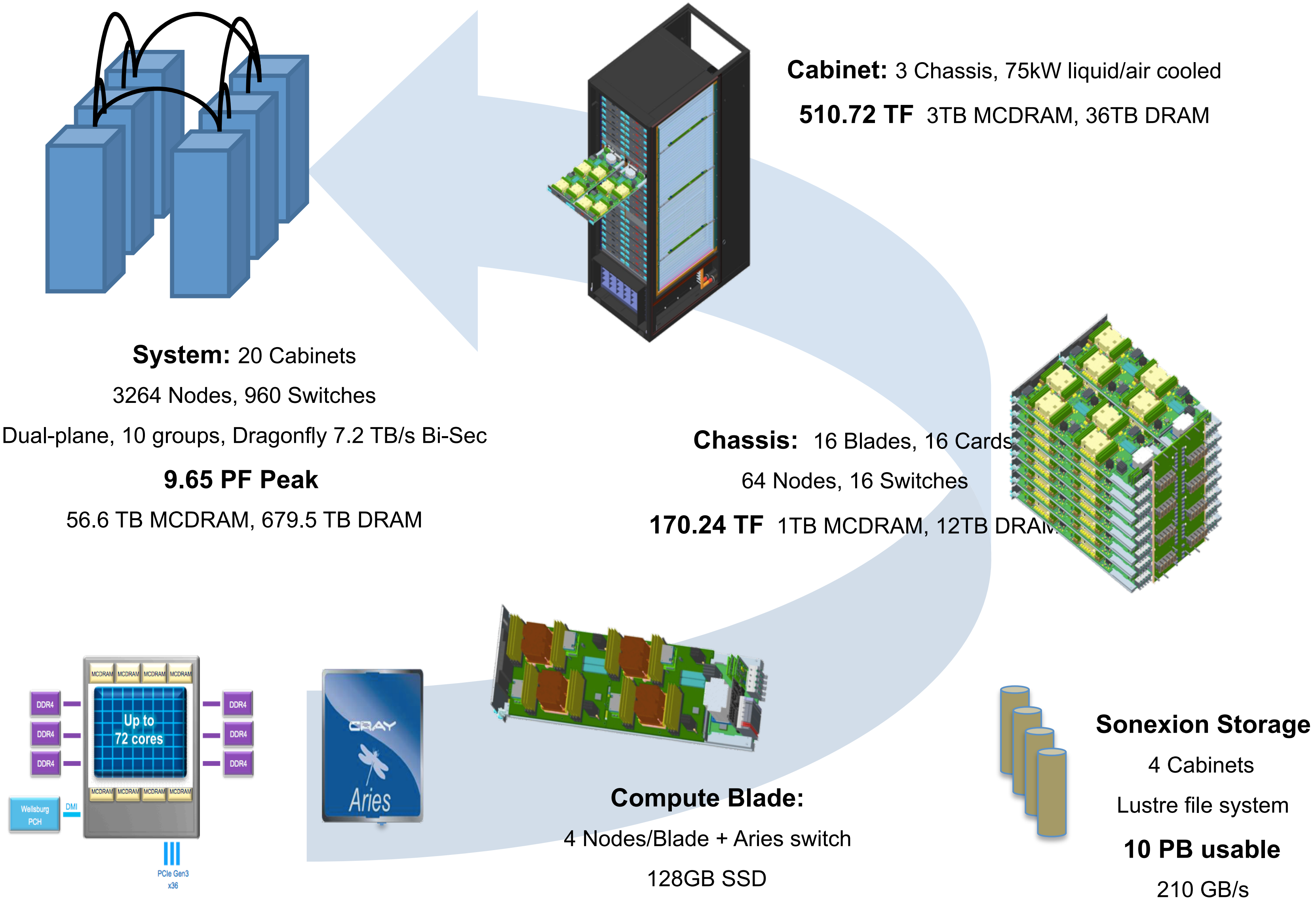
Theta



Theta is the first of two pre-exascale systems coming to Argonne

- Serves as a bridge between Mira and Aurora, transition and data analytics system
- Cray XC40 system. Runs Cray software stack
- 9.65 PF peak performance
- 3624 nodes with 2nd Generation Intel® Xeon Phi™ processor
  - codename Knights Landing, 7230 SKU 64 cores 1.3GHz
- 192GB DDR4 memory 16GB MCDRAM on each node
- 128GB SSD on each node
- Cray Aries high speed interconnect in dragonfly topology
- Initial file system: 10PB Lustre file system, 200 GB/s throughput
- Currently in T2O phase running ESP & ADSP projects
- Iota – 44 node, air cooled TD system

Theta system overview



Cabinet: 3 Chassis, 75kW liquid/air cooled

510.72 TF 3TB MCDRAM, 36TB DRAM

System: 20 Cabinets

3264 Nodes, 960 Switches

Dual-plane, 10 groups, Dragonfly 7.2 TB/s Bi-Sec

9.65 PF Peak

56.6 TB MCDRAM, 679.5 TB DRAM

Chassis: 16 Blades, 16 Cards

64 Nodes, 16 Switches

170.24 TF 1TB MCDRAM, 12TB DRAM

Sonexion Storage

4 Cabinets

Lustre file system

10 PB usable

210 GB/s

Compute Blade:

4 Nodes/Blade + Aries switch

128GB SSD

10.64 TF 64GB MCDRAM

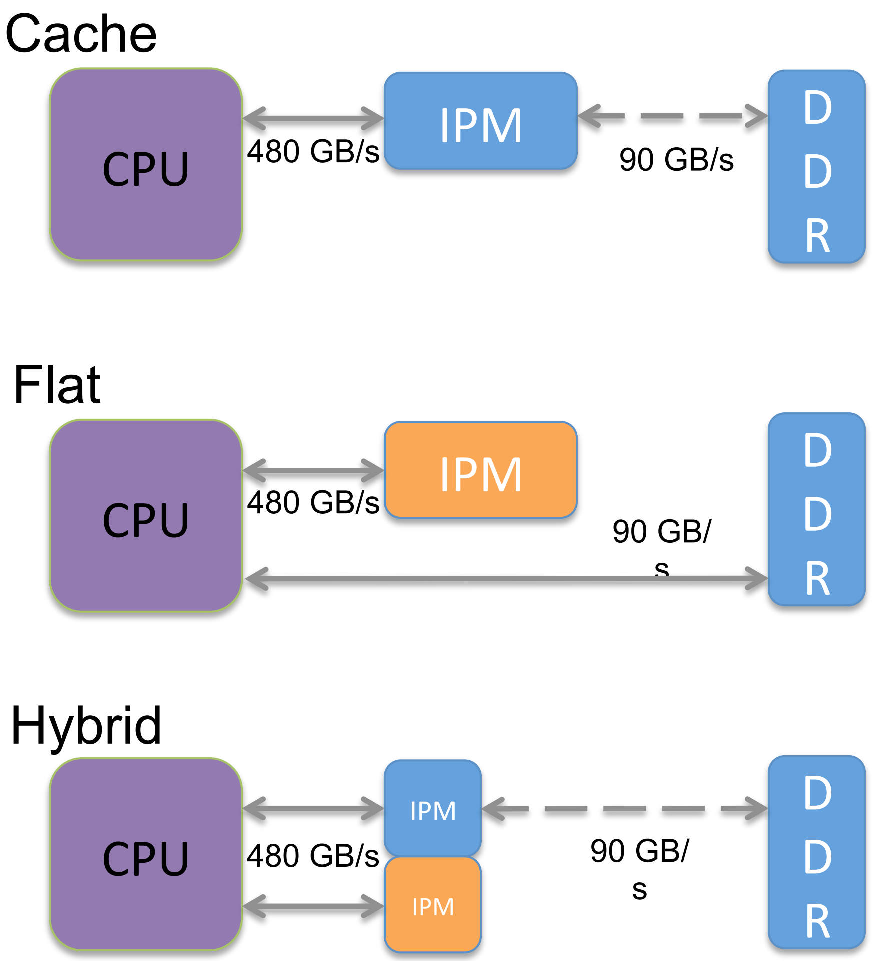
768GB DRAM

Node: KNL Socket

192 GB DDR4 (6 channels) **2.66 TF** 16GB MCDRAM

Memory Modes - IPM and DDR

Selected at node boot time



- **Two memory types**
- In Package Memory (IPM)
  - 16 GB MCDRAM
  - ~480 GB/s bandwidth
- Off Package Memory (DDR)
  - Up to 384 GB
  - ~90 GB/s bandwidth
- **One address space**
- Possibly 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

Operating System

Cray Linux Environment (CLE)

- **Login node**: SUSE Enterprise Linux based full CLE OS
- **Compute Node** Linux (CNL)
  - Subset of CLE Linux distribution
  - Reduced OS noise and jitter, <3% runtime variability
  - Provides standard Linux services and interfaces
  - Doesn't restrict services as much as a Light Weight Kernel
  - Configurable from Extreme Scaling Mode to Cluster Compatibility Mode
  - OS activity largely confined to OS cores
  - LD_PRELOAD and shared libraries supported
  - MPMD jobs supported
  - Interfaces for controlling thread placement and affinity
  - POSIX signals
  - Memory utilization information
  - Core file generation and management via ATP

Filesystems

- GPFS
  - Home directories (/home) are in /gpfs/mira-home
    - Default quota 50GiB
    - Your home directory is backed up

- Lustre
  - Project directories (/projects) are in /lus/theta-fs0/projects
    - Access controlled by unix group of your project
    - Default quota 1TiB
    - NOT backed up
  - With large I/O, be sure to consider **stripe width**

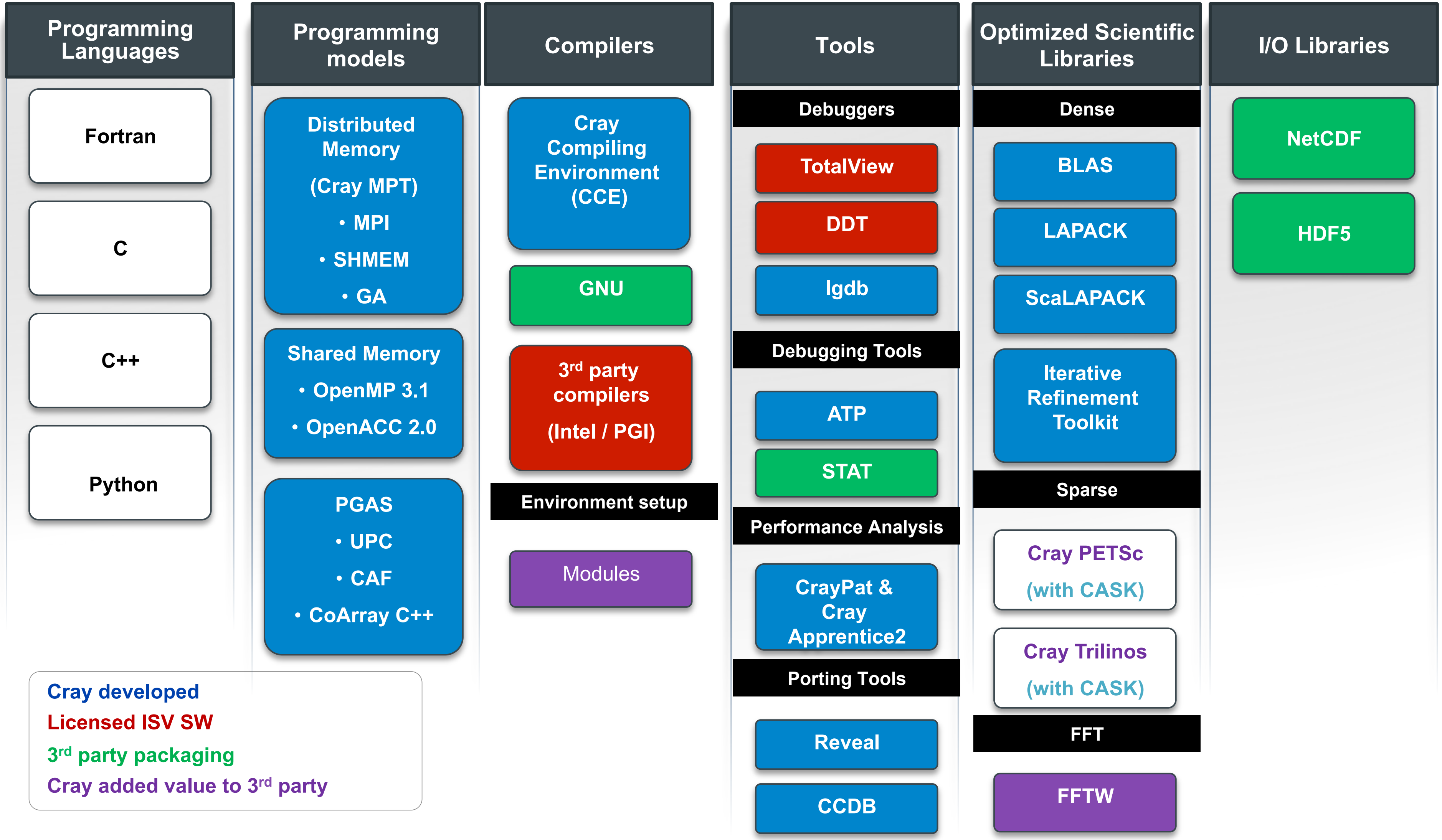
Modules

- 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*
  - *help*
  - *list ← what is currently loaded*
  - *avail*
  - *load*
  - *unload*
  - *switch|swap*
  - *use ← add a directory to MODULEPATH*
  - *display|show*

Compilers

- For all compilers (Intel, Cray, Gnu, etc):
  - **Use:** cc, CC, ftn
  - **Do not use** mpicc, MPICC, mpic++, mpif77, mpif90
    - *they do not generate code for the compute nodes*
- Selecting the compiler you want using **"module swap"** or **"module unload"** followed by **"module load"**
  - Intel
    - PrgEnv-intel *This is the default*
  - Cray
    - module swap PrgEnv-intel PrgEnv-cray
    - **NOTE:** links libsci by default
  - Gnu
    - module swap PrgEnv-intel PrgEnv-gnu
  - Clang/LLVM
    - module swap PrgEnv-intel PrgEnv-llvm

Cray Programming Environment



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 / PGI)

Environment setup

- Modules

Tools

Debuggers

- TotalView
- DDT
- lgdb

Debugging Tools

- ATP
- STAT

Performance Analysis

- CrayPat & Cray Apprentice2

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

Cray developed

Licensed ISV SW

3rd party packaging

Cray added value to 3rd party

Tools: performance, profiling, debugging

- Non-system libraries and tools are under the /soft directory, *module setup is in progress*
  - **/soft/applications** – applications
  - **/soft/compilers** – site installed compilers
    - llvm and intel beta releases
  - **/soft/debuggers** - debuggers
    - DDT
  - **/soft/libraries** - libraries
    - argobots, bolt, breakpad
  - **/soft/perftools** - performance tools
    - darshan, hpctoolkit, memlog, TAU, etc.

Theta Job script

```
#!/bin/bash
#COBALT -t 10
#COBALT –n 2
#COBALT -A Myprojectname

# Various env settings are provided by Cobalt
echo $COBALT_JOBID  $COBALT_PARTNAME  $COBALT_JOBSIZE

aprun -n 16 -N 8 -d 1 -j 1 -cc depth ./a.out
status=$?

# could do another aprun here...

exit $status
```

Aprun overview

- Options
  - -n *total_number_of_ranks*
  - -N *ranks_per_node*
  - -d *depth* [number of cpus (hyperthreads) per rank]
  - -cc **depth** [Note: **depth** is a keyword]
  - -j *hyperthreads* [cpus (hyperthreads) per compute unit (core)]

- Env settings you may need
  - -e OMP_NUM_THREADS=*nthreads*
  - -e KMP_AFFINITY=...

- See also **man aprun**

Submitting a Cobalt job

- qsub –A <project> -q <queue> -t <time> -n <nodes> ./jobscript.sh

  E.g.

  qsub –A Myprojname –q cache-quad t –t 10 –n 32 ./jobscript.sh

- If you specify your options in the script via #COBALT, then just:
  - qsub jobscript.sh
- Make sure jobscript.sh is executable
- Without "-q", submits to the queue named "default"
- Without "-A", uses environment variable COBALT_PROJ if set
  - export COBALT_PROJ=ATPESC2016
- **man qsub** for more options

Theta queues and modes

- MCDRAM and NUMA modes can only be set by the system when nodes are rebooted. *Users cannot directly reboot nodes.*
- Submit job with the --attrs flag to get the mode you need. E.g.
  - qsub −n 32 −t 60 −attrs mcdram=cache:numa=quad ./jobscript.sh
- Other mode choices
  - mcdram: cache, flat, split, equal
  - numa: quad, a2a, hemi, snc2, snc4
- Queues
  - Normal jobs use queue named "default"
  - Debugging: debug-cache-quad, debug-flat-quad
    - Note: pre-set for mcdram/numa configuration
  - "qstat −Q" lists all queues

Managing your job

- qstat – show what's in the queue
  - qstat –u <username> # Jobs only for user
  - qstat <jobid> # Status of this particular job
  - qstat –fl <jobid> # Detailed info on job
- qdel <jobid>
- showres – show reservations currently set in the system
- **man qstat** for more options

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”
  - jobid can be inserted into your string e.g. "-O myprefix_$jobid"

- **Cobalt log file**: **<prefix>.cobaltlog**
  - created by Cobalt when job is submitted, additional info written during the job
  - 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

Interactive job

- Useful for short tests or debugging
- Submit the job with –I (letter I for Interactive)
  - Default queue and default project
    - qsub –I –n 32 –t 30
  - Specify queue and project:
    - qsub –I –n 32 –t 30 –q cache-quad –A Myprojname
- Wait for job's shell prompt
  - *This is a new shell* with env settings e.g. COBALT_JOBID
  - Exit this shell to end your job
- From job's shell prompt, run just like in a script job, e.g.
  - aprun -n 512 -N 16 -d 1 -j 1 -cc depth ./a.out
- After job expires, apruns will fail. *Check* **qstat $COBALT_JOBID**

Reasons why a job may not be running yet

- Job is in state "queued"
  - There is a reservation which interferes with your job
    - **showres** shows all reservations currently in place
  - There are no available nodes for the requested queue
    - **nodelist** shows idle nodes
  - Job was submitted to a queue that is restricted from running at this time
- Job is in state "starting"
  - If no nodes are currently booted in the cache/numa mode requested (via --attrs), your job may be in the state "starting" for up to 15 minutes while the nodes are rebooted.

Core files and debugging

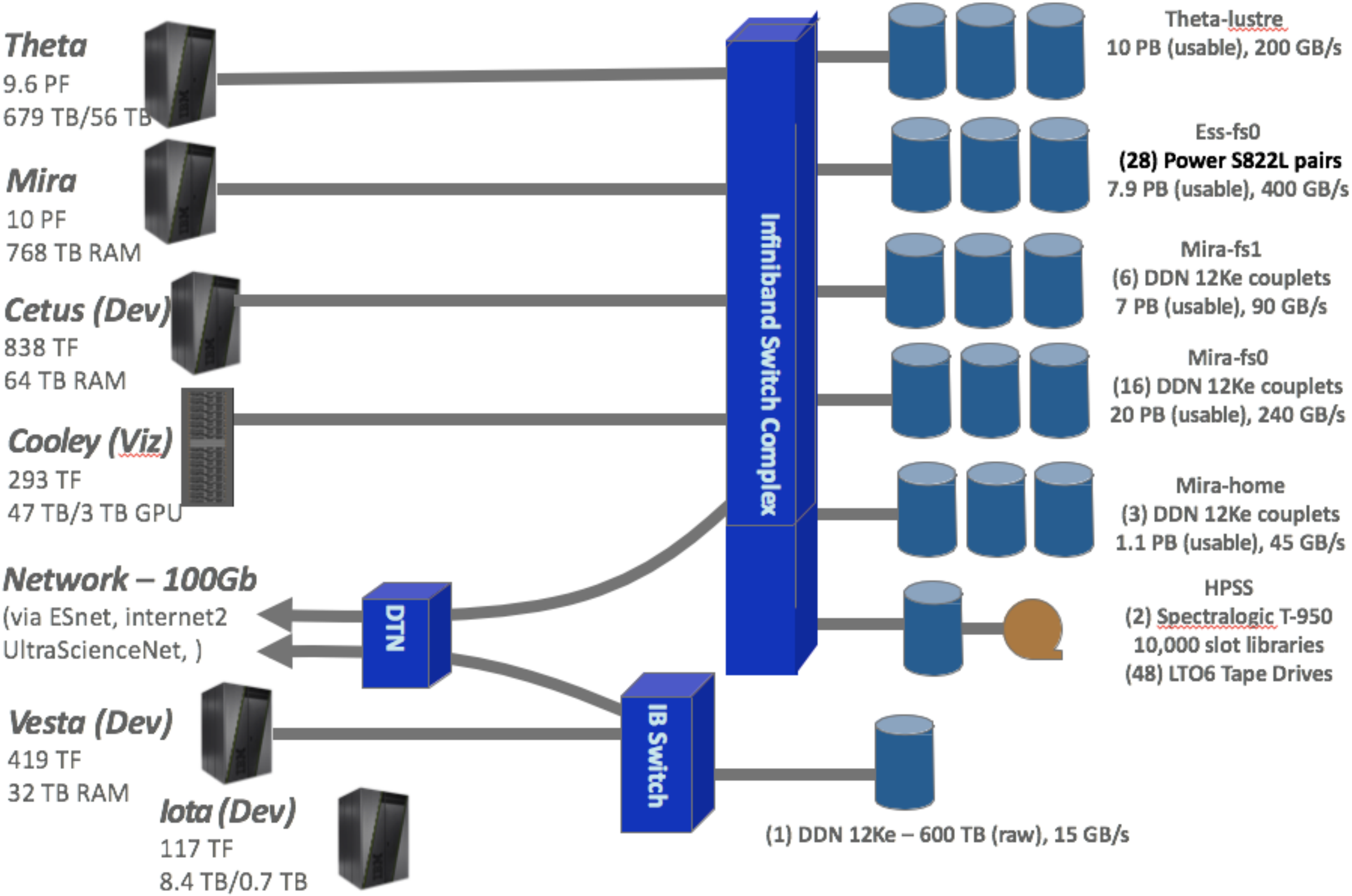
- Abnormal Termination Processing (ATP)
  - Set environment **ATP_ENABLED=1** in your job script before aprun
  - On program failure, generates a merged stack backtrace tree in file **atpMergedBT.dot**
  - View the output file with the program **stat-view** (module load stat)

- Notes on linking your program
  - PrgEnv-cray links everything necessary by default
  - PrgEnv-intel
    - Link with **-Wl,-T/opt/cray/pe/cce/8.5.2/craylibs/x86-64/2.23.1.cce.ld**

- Other debugging tools
  - You can generate STAT snapshots asynchronously
  - Full-featured debugging with DDT
  - More info at
    - https://collab.cels.anl.gov/display/ESP/ATP+and+STAT
    - https://collab.cels.anl.gov/display/ESP/Allinea+Forge+(DDT)

Part II : Cooley

ALCF Resources



Softenv

- Cooley uses **softenv** instead of **modules**
  - Softenv is similar to modules
  - Keys are read at login time to set environment variables like PATH.
  - Mira, Cetus, Vesta: ~/.soft
  - Cooley: ~/.soft.cooley
- To get started:

```
# Select latest version of mvapich2 with GNU compilers
+mvapich2
@default
# the end – do not put any keys after the @default
```

- After edits to .soft, type "**resoft**" or log out and back in again
- Type "**softenv"** to see list of all available keys

Compilers

- Choose compiler via softenv keys

- Non-MPI
  - GNU: +gcc-4.8.1 (gcc, g++, gfortran)
  - Intel: +intel-composer-xe (icc, ifort)
  - Clang: @clang (clang)

- MPI compiler wrappers
  - mvapich (mpicc, mpicxx, mpifort)
    - GNU: +mvapich2
    - Intel: +mvapich2-intel
    - Clang: @mvapich2-clang (no mpifort)
  - mpich (mpicc, mpicxx, mpif77, mpif90)
    - GNU: +mpich2-1.4.1p1
    - Intel: +mpich2-1.4.1p1-intel

Cooley Job Script

- Job script similar to Theta except mpirun instead of aprun
  - Example test.sh:

    ```
    #!/bin/sh
    NODES=`cat $COBALT_NODEFILE | wc -l`
    PROCS=$((NODES * 12))
    mpirun -f $COBALT_NODEFILE -n $PROCS myprog.exe
    ```

  - Submit on 5 nodes for 10 minutes

    ```
    qsub –q default -n 5 -t 10 –A yourprojectname ./test.sh
    ```

  - Queues
    - **default** for large/long jobs, **debug** for short/small jobs
    - use **pubnet** to get public network visibility
    - use **nox11** queues to suppress X server for CUDA jobs
  - Refer to online user guide for more info

Live demos

Wrap-up

When things go wrong… logging in

- Check to make sure it’s not a scheduled system maintenance day:
  - Login nodes on ALCF systems are often closed off during system maintenance to allow for activities that would impact users.
  - Look for reminders in the weekly maintenance announcement to users and in the pre-login banner message.
  - An all-clear email will be sent out to users 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:
  - 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 a 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?
    - 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

Theta ESP wiki (Confluence)

1. Sign up for a Confluence account:
   - Go to https://collab.cels.anl.gov and click "Sign up" at top right
   - Enter contact info, including email address. Click "Sign up”.
2. Request access to the Theta ESP wiki area within Confluence
   - E-mail to accounts@alcf.anl.gov with your Confluence username and request access to the Theta ESP wiki
   - You will be notified via email within two business days when you have been added to the Theta ESP wiki
3. Go to https://collab.cels.anl.gov/display/ESP/Theta+ESP+Wiki and log in to Confluence with your credentials from step 1.

Getting help

Online resources (24/7):

- ALCF web pages:
  - -- Theta: https://collab.cels.anl.gov/display/ESP (see slide on sign-up)
  - – Production ALCF machines (e.g. Cooley): http://www.alcf.anl.gov
  - – https://accounts.alcf.anl.gov

Contact us:

e-mail: Theta support**: theta-esp@alcf.anl.gov**

Cooley/Mira/Vesta: **support@alcf.anl.gov**

ALCF Help Desk:

Your Catalyst

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

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

News from ALCF:

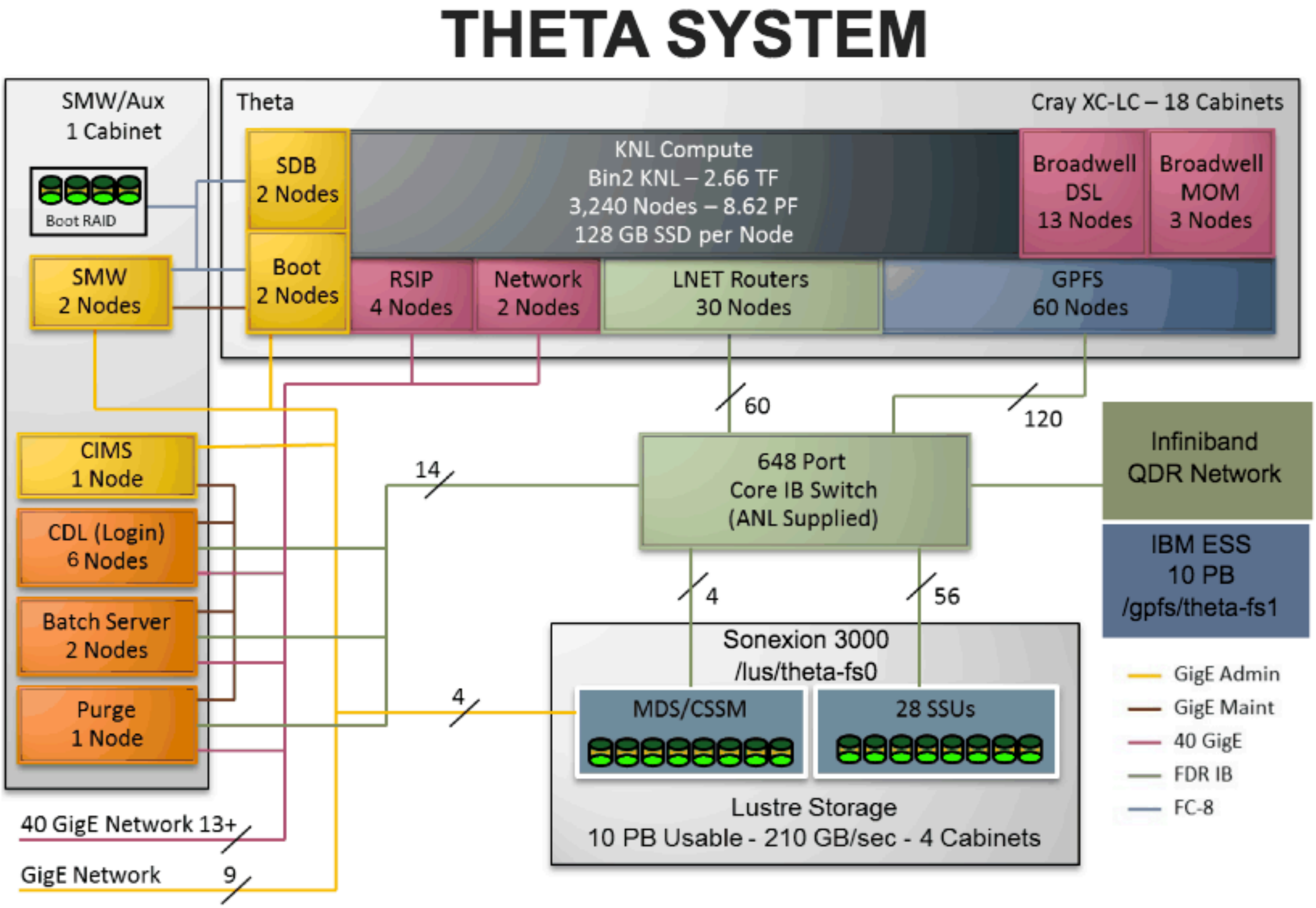
- ALCF Weekly Updates, ALCF newsletters, e-mail via *theta-notify* and *cooley-notify* lists.

Argonne Leadership Computing Facility
Theta/Cooley Quick Start

Thank you for attending!

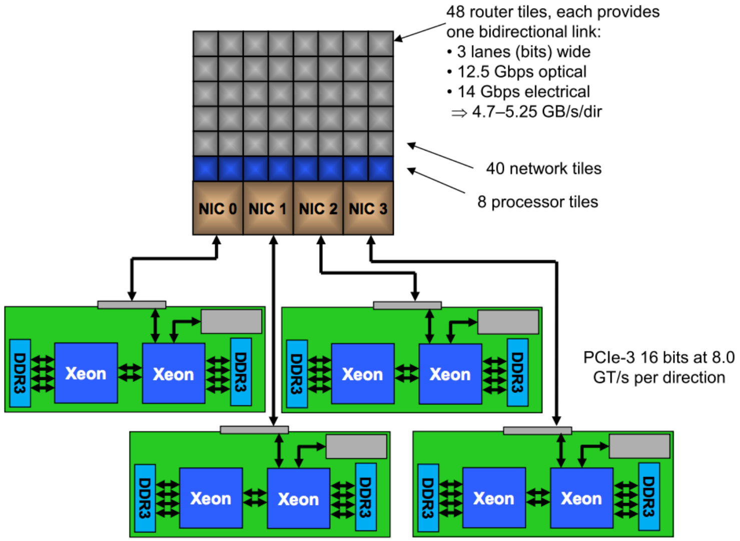
SUPPLEMENTAL TOPICS

Theta System



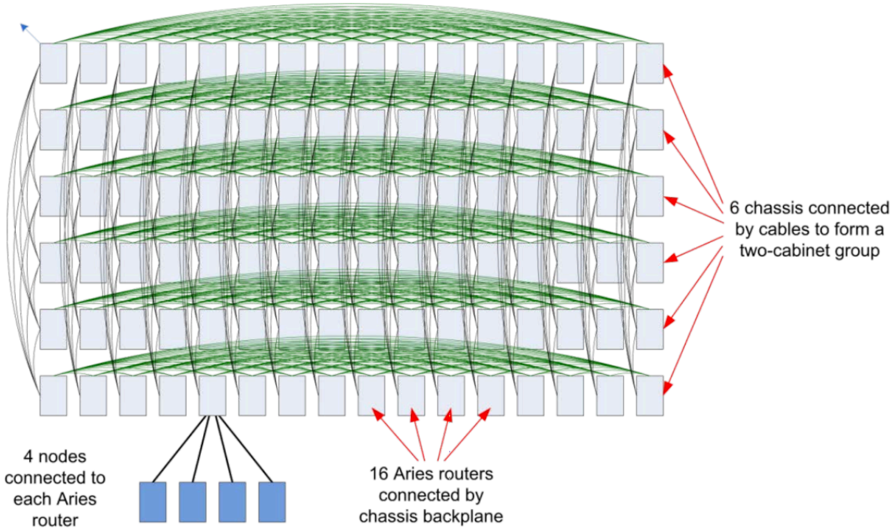
Argonne

Aries Dragonfly Network



Aries Router:

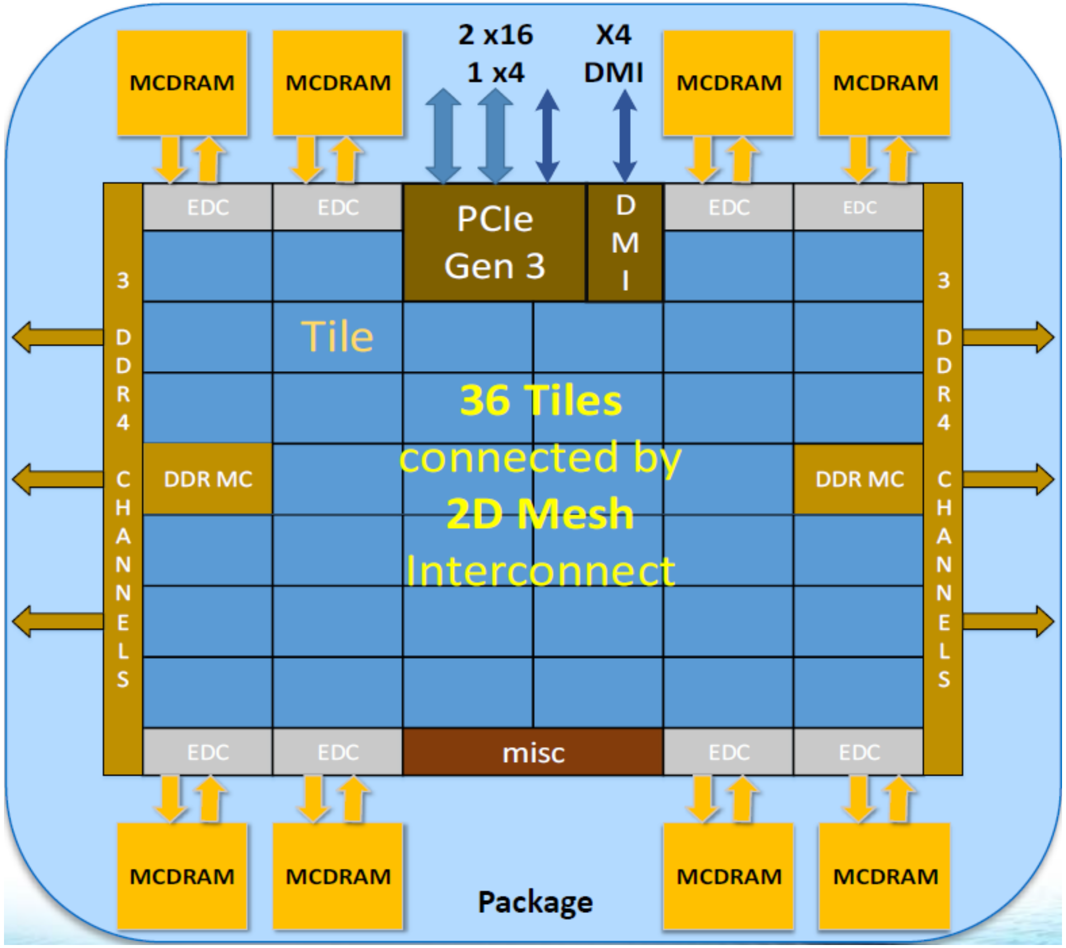
- 4 NIC's connected via PCIe
  - 40 Network tiles/links
- 4.7-5.25 GB/s/dir per link


6 chassis connected by cables to form a two-cabinet group
4 nodes connected to each Aries router
16 Aries routers connected by chassis backplane

Dragonfly topology

- 4 nodes connected to an Aries
  - 2 Local all-to-all dimensions
    - 16 all-to-all horizontal
    - 6 all-to-all vertical
  - 384 nodes in local group
- All-to-all connections between groups

Knights Landing Processor



Chip
- 683 mm²
- 14 nm process
- 8 Billion transistors

Up to 72 Cores
- 36 tiles
- 2 cores per tile
- 2.4 TF per node

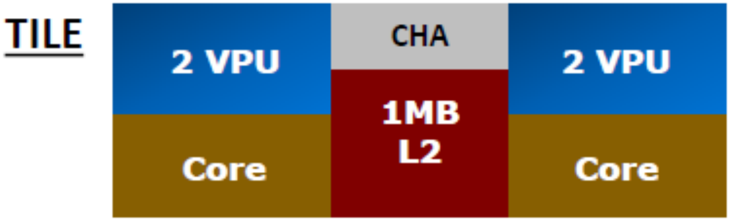
2D Mesh Interconnect
- Tiles connected by 2D mesh

On Package Memory
- 16 GB MCDRAM
- 8 Stacks
- ~450 GB/s bandwidth

6 DDR4 memory channels
- 2 controllers
- up to 384 GB external DDR4
- 90 GB/s bandwidth

On Socket Networking
- Omni-Path NIC on package
- Connected by PCIe

KNL Tile and Core

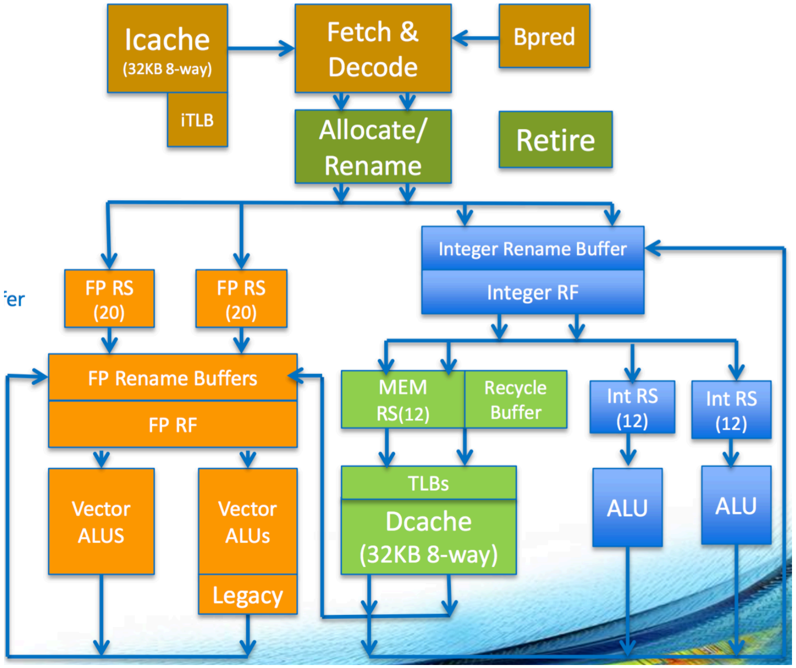


Tile

- Two CPUs
- 2 VPUs per core
- Shared 1 MB L2 cache (not global)

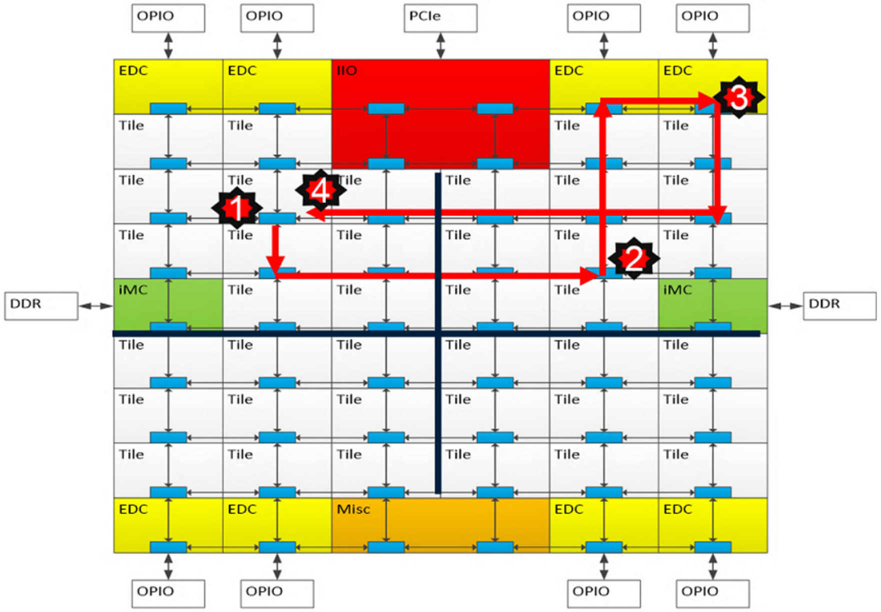
Core

- Based on Silvermont (Atom)
  - Functional units:
    - 2 Integer ALUs
    - 2 Memory units
  - 2 VPU's with AVX-512
  - Instruction Issue & Exec:
    - 2 wide decode

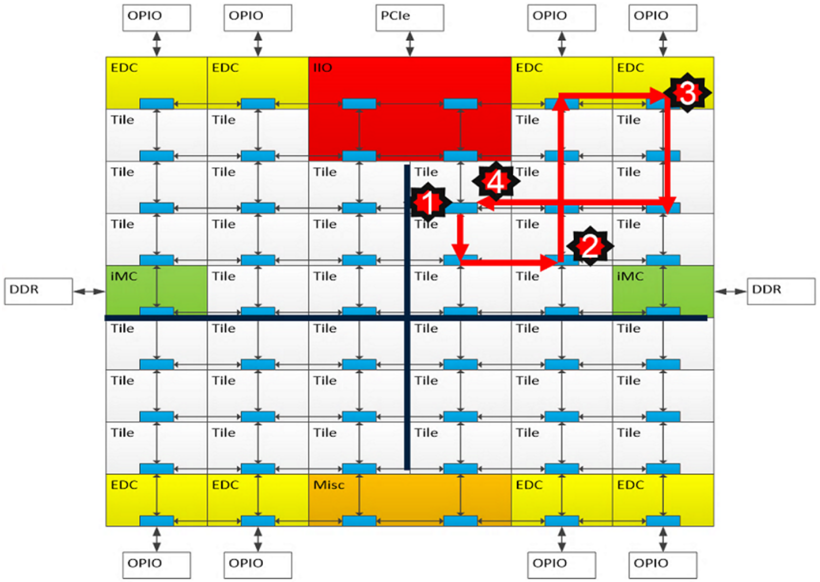


Clustering Modes

Quadrant Mode



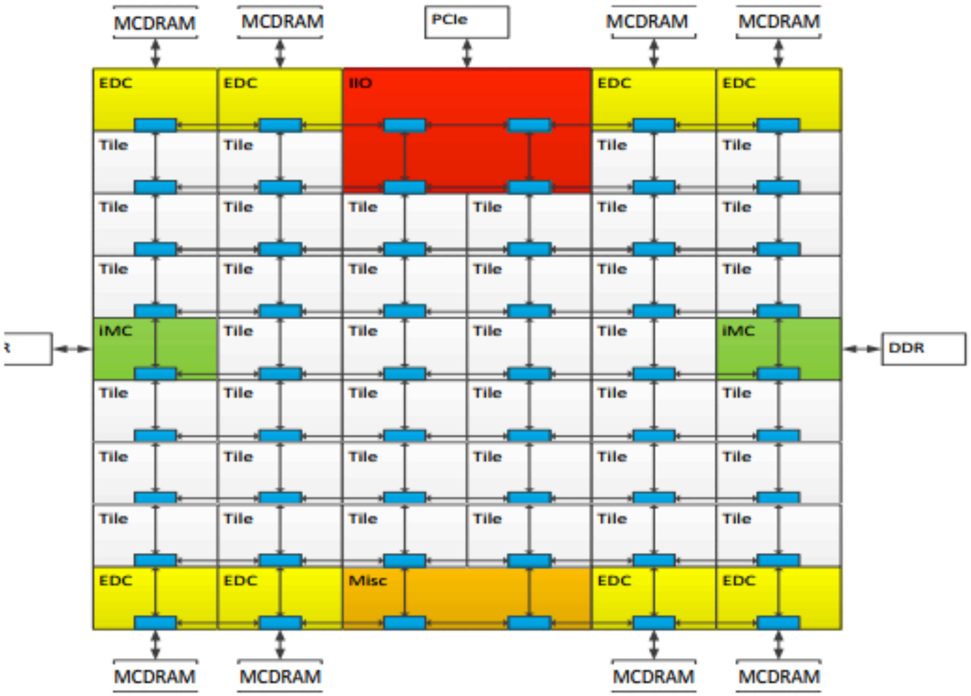
SNC-4 Mode



1. Tile has Cache Miss
2. CHA selected
3. Memory Controller
4. Memory Received

Knights Landing overview

KNL Mesh Interconnect



Mesh of Rings

- Every row and column is a (half) ring
- YX routing: Go in Y → Turn → Go in X
- Messages arbitrate at injection and on turn

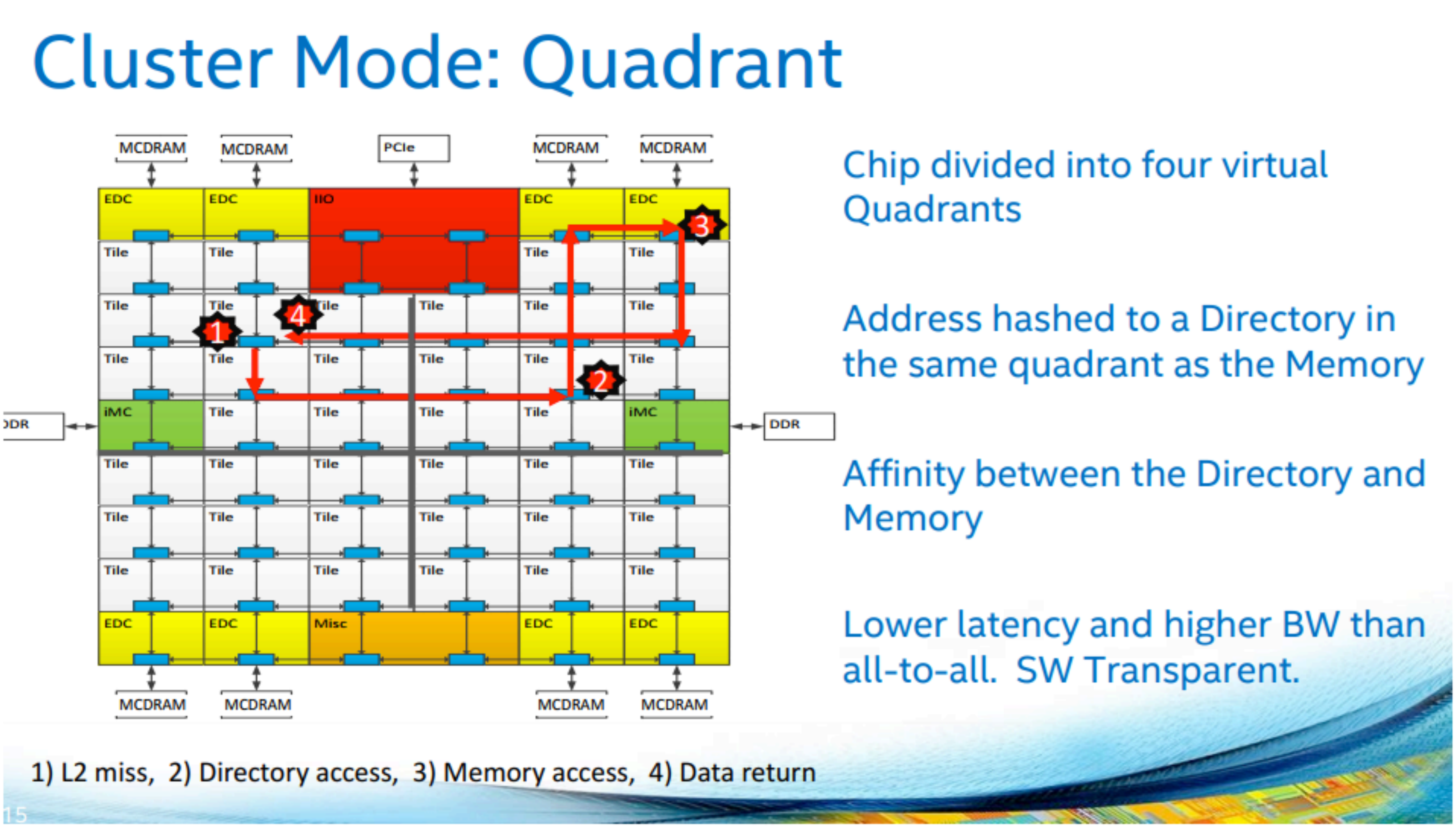
Cache Coherent Interconnect

- MESIF protocol (F = Forward)
- Distributed directory to filter snoops

Three Cluster Modes

(1) All-to-All (2) Quadrant (3) Sub-NUMA Clustering

Knights Landing overview



Cluster Mode: Quadrant

Chip divided into four virtual Quadrants

Address hashed to a Directory in the same quadrant as the Memory

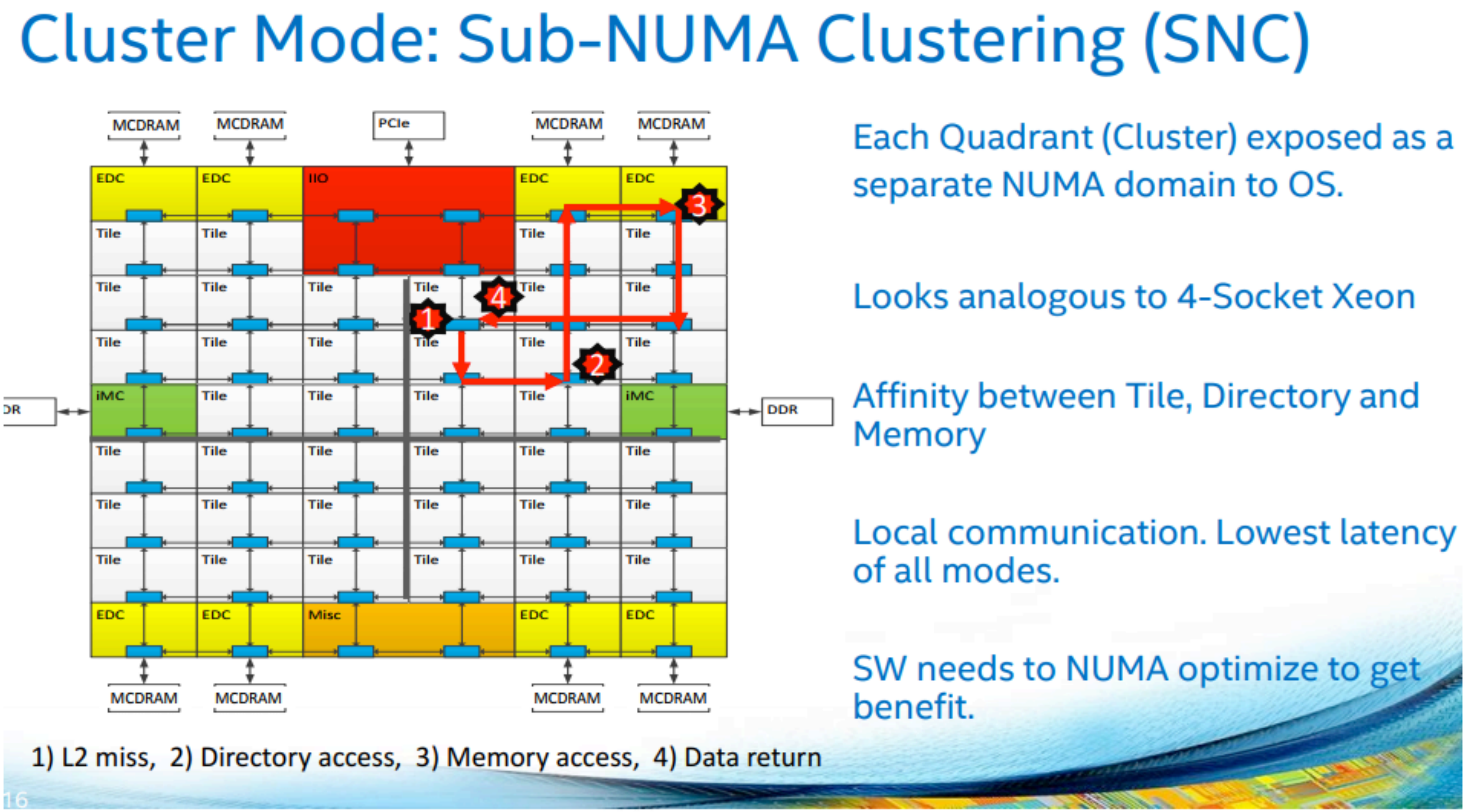
Affinity between the Directory and Memory

Lower latency and higher BW than all-to-all. SW Transparent.

1) L2 miss, 2) Directory access, 3) Memory access, 4) Data return

15

Knights Landing overview



Cluster Mode: Sub-NUMA Clustering (SNC)

Each Quadrant (Cluster) exposed as a separate NUMA domain to OS.

Looks analogous to 4-Socket Xeon

Affinity between Tile, Directory and Memory

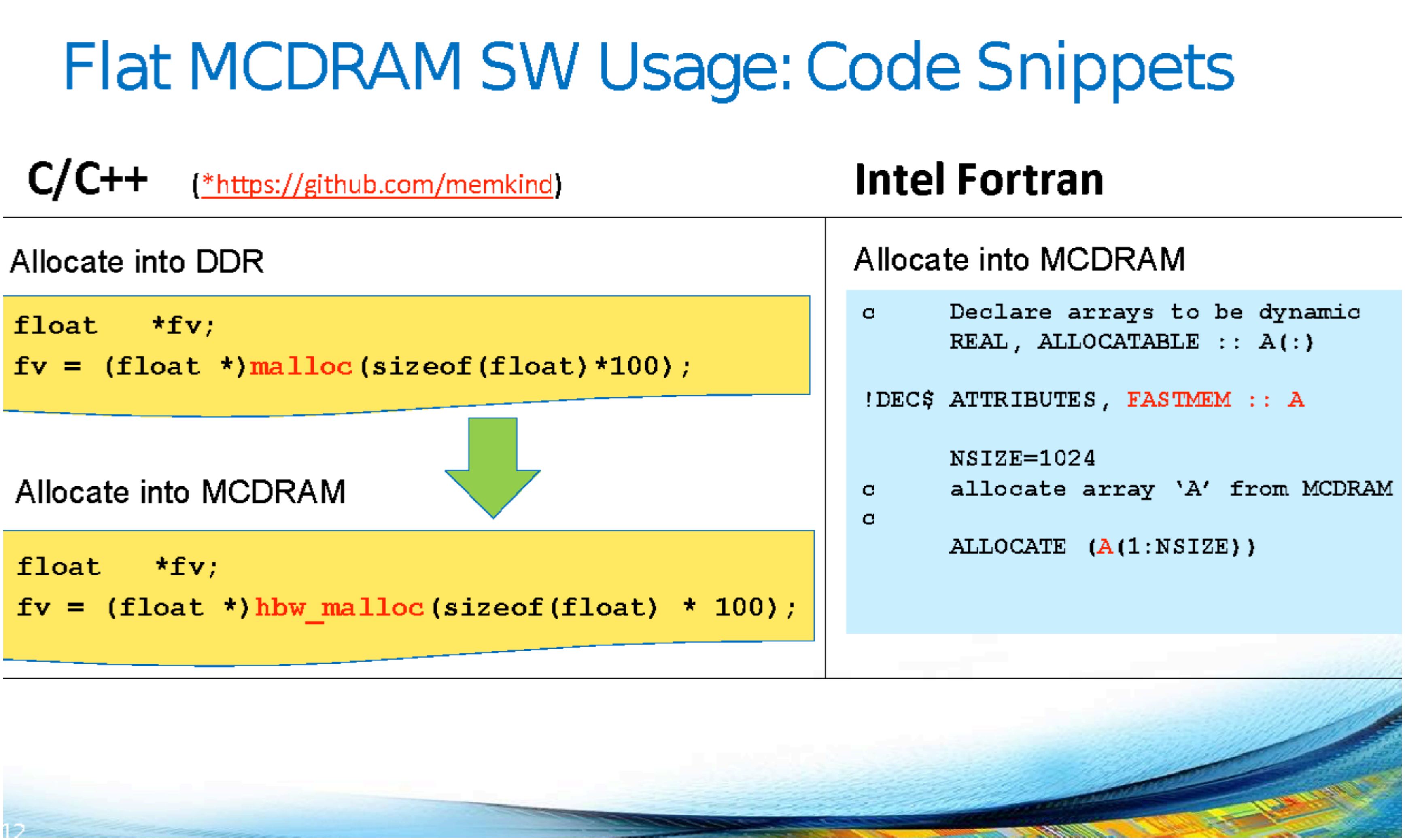
Local communication. Lowest latency of all modes.

SW needs to NUMA optimize to get benefit.

1) L2 miss, 2) Directory access, 3) Memory access, 4) Data return

16

Knights Landing overview



Flat MCDRAM SW Usage: Code Snippets

C/C++ (*https://github.com/memkind)

Allocate into DDR

```
float    *fv;
fv = (float *)malloc(sizeof(float)*100);
```

Allocate into MCDRAM

```
float    *fv;
fv = (float *)hbw_malloc(sizeof(float) * 100);
```

Intel Fortran

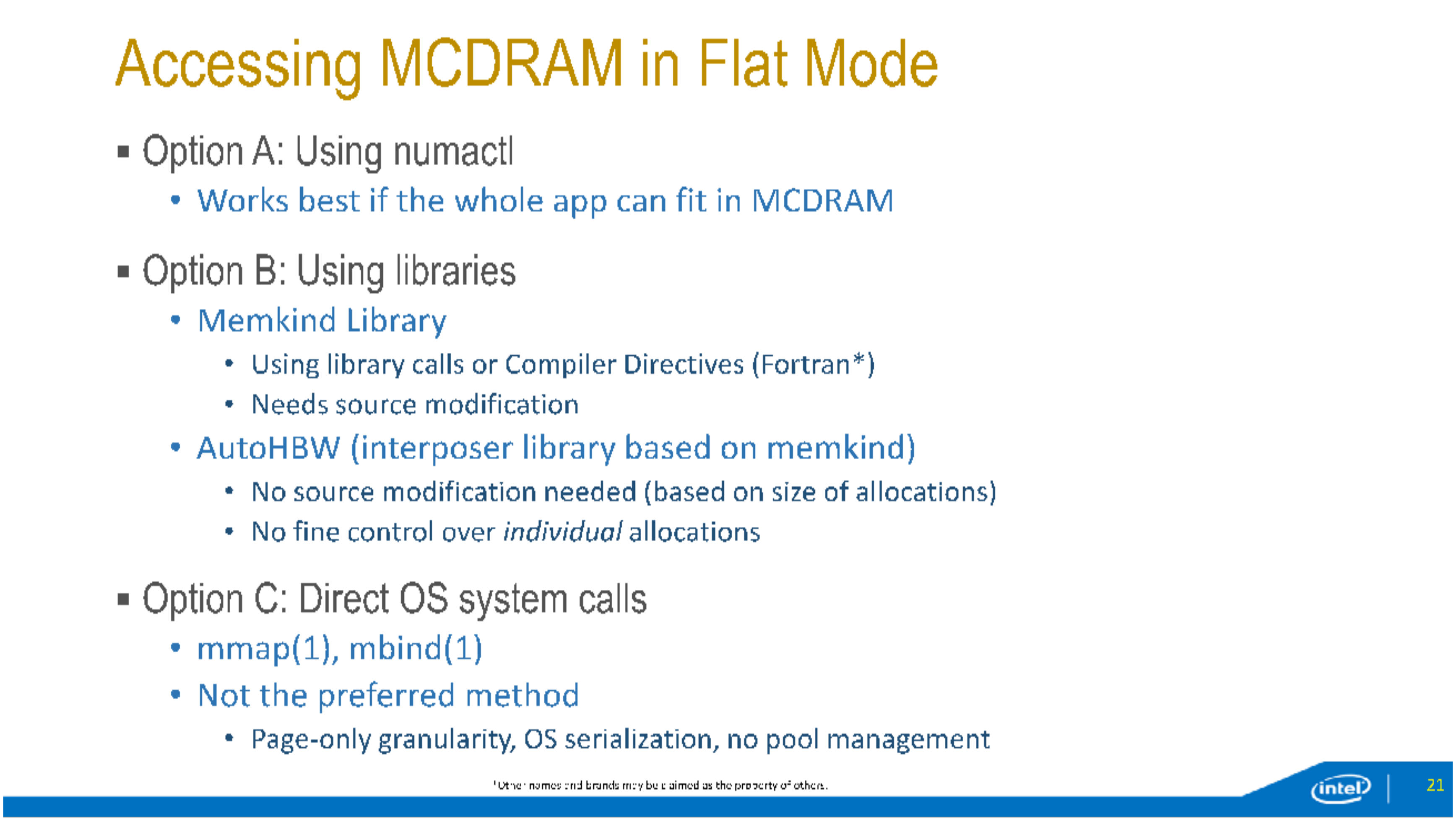
Allocate into MCDRAM

```
c       Declare arrays to be dynamic
        REAL, ALLOCATABLE :: A(:)

!DEC$ ATTRIBUTES, FASTMEM :: A

        NSIZE=1024
c       allocate array 'A' from MCDRAM
c
        ALLOCATE (A(1:NSIZE))
```

Multi-channel DRAM overview

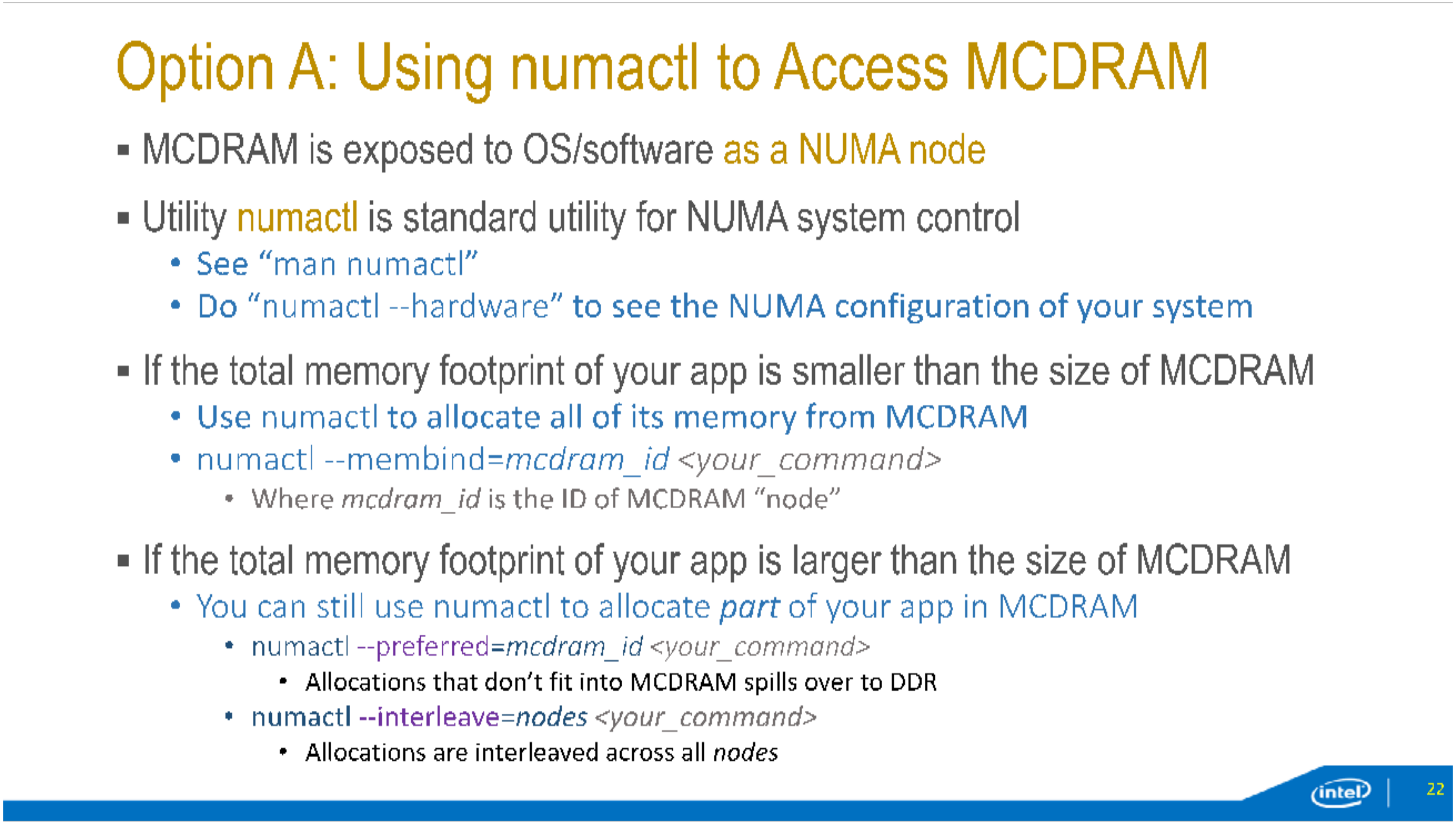


Accessing MCDRAM in Flat Mode

- Option A: Using numactl
  - Works best if the whole app can fit in MCDRAM
- Option B: Using libraries
  - Memkind Library
    - Using library calls or Compiler Directives (Fortran*)
    - Needs source modification
  - AutoHBW (interposer library based on memkind)
    - No source modification needed (based on size of allocations)
    - No fine control over *individual* allocations
- Option C: Direct OS system calls
  - mmap(1), mbind(1)
  - Not the preferred method
    - Page-only granularity, OS serialization, no pool management

*Other names and brands may be claimed as the property of others.

Multi-channel DRAM overview



Option A: Using numactl to Access MCDRAM

- MCDRAM is exposed to OS/software as a NUMA node
- Utility numactl is standard utility for NUMA system control
  - See "man numactl"
  - Do "numactl --hardware" to see the NUMA configuration of your system
- If the total memory footprint of your app is smaller than the size of MCDRAM
  - Use numactl to allocate all of its memory from MCDRAM
  - numactl --membind=*mcdram_id* <*your_command*>
    - Where *mcdram_id* is the ID of MCDRAM "node"
- If the total memory footprint of your app is larger than the size of MCDRAM
  - You can still use numactl to allocate *part* of your app in MCDRAM
    - numactl --preferred=*mcdram_id* <*your_command*>
      - Allocations that don't fit into MCDRAM spills over to DDR
    - numactl **--interleave**=*nodes* <*your_command*>
      - Allocations are interleaved across all *nodes*

22

Performance Tools

- CrayPat/Cray Apprentice2
  - Profiling, tracing, and performance visualization tool
- Cray Reveal
  - Combines performance information with Cray compiler optimization feedback
- Intel Vtune
  - Detailed processor level performance analysis utilizing sampling and hardware counters
- Intel Trace Analyzer and Collector
  - MPI profiling and tracing
- Intel Advisor
  - Provides guidance for vectorizing and threading
- PAPI
  - Library providing API to access hardware performance counters
- TAU
  - Profiling and tracing toolkit
- HPCToolkit
  - Performance measurement and analysis toolkit utilizing sampling
- Vampir/Score-P
  - Performance analysis tools providing large scale tracing and visualization
- Darshan:
  - IO characterization tool

Debugging Tools

- DDT
  - Full featured parallel debugger
- TotalView
  - Full featured parallel debugger
- Cray LGDB & CCDB
  - Parallel command line debugger with comparative debugging
- ATP (Cray Abnormal Termination tool)
  - Stack traces on exit for application failures
- STAT
  - Stack traces for hung applications
- Intel Inspector
  - Memory and thread error checking
- Valgrind
  - Memory debugging, memory leak detection, thread debugging with data race detection

Reservations

- Reservations allow exclusive use of a set of nodes for a specified group of users for a specific period of time
  - a reservation prevents other users' jobs from running on that resource
  - 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