

# ARCHER2 Technology

Andy Turner, EPCC

22 April 2020

[www.archer2.ac.uk](http://www.archer2.ac.uk)



|epcc|

# Reusing this material



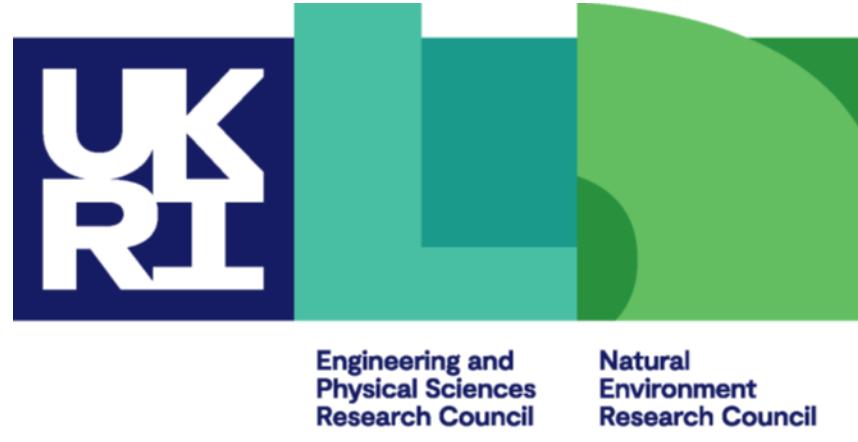
This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

This means you are free to copy and redistribute the material and adapt and build on the material under the following terms: You must give appropriate credit, provide a link to the license and indicate if changes were made. If you adapt or build on the material you must distribute your work under the same license as the original.

Note that this presentation contains images owned by others. Please seek their permission before reusing these images.

# Partners



THE UNIVERSITY  
*of* EDINBURGH



- **Hardware details**
  - Node types and architecture
  - Interconnect
  - Storage
  - Comparisons to ARCHER
- **Software details**
  - Cray-provided software environment
  - EPCC CSE-provided software environment
  - Comparisons to ARCHER
- **Details useful for:**
  - Users who plan to use pre-installed software on the service
  - Users who plan to develop and compile software on the service



# Hardware Overview



epcc

# System Overview

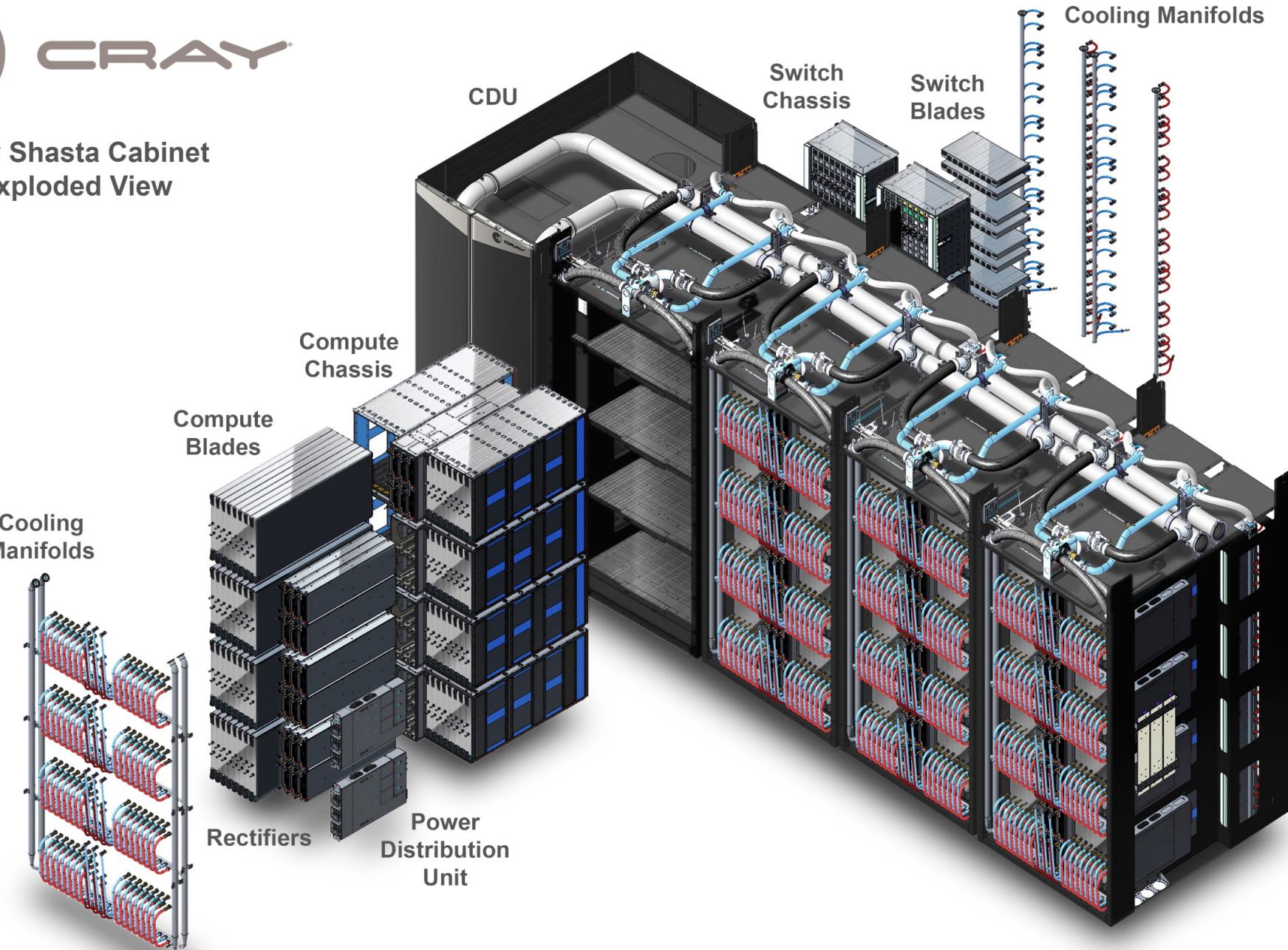
- Peak performance: ~28 PFlops
  - CASTEP: 11x ARCHER
  - CP2K: 9x ARCHER
  - GROMACS: 13x ARCHER
  - OpenSBLI: 9x ARCHER
  - Met Office UM: 18x ARCHER
- 5,848 compute nodes in 23 Shasta Mountain cabinets
  - 748,544 AMD cores
- 14.5 PB Lustre (4 file systems)
- 1.1 PB solid state burst buffer
- 1 PB home (backed up) storage
- Cray Slingshot interconnect





CRAY®

## Cray Shasta Cabinet Exploded View

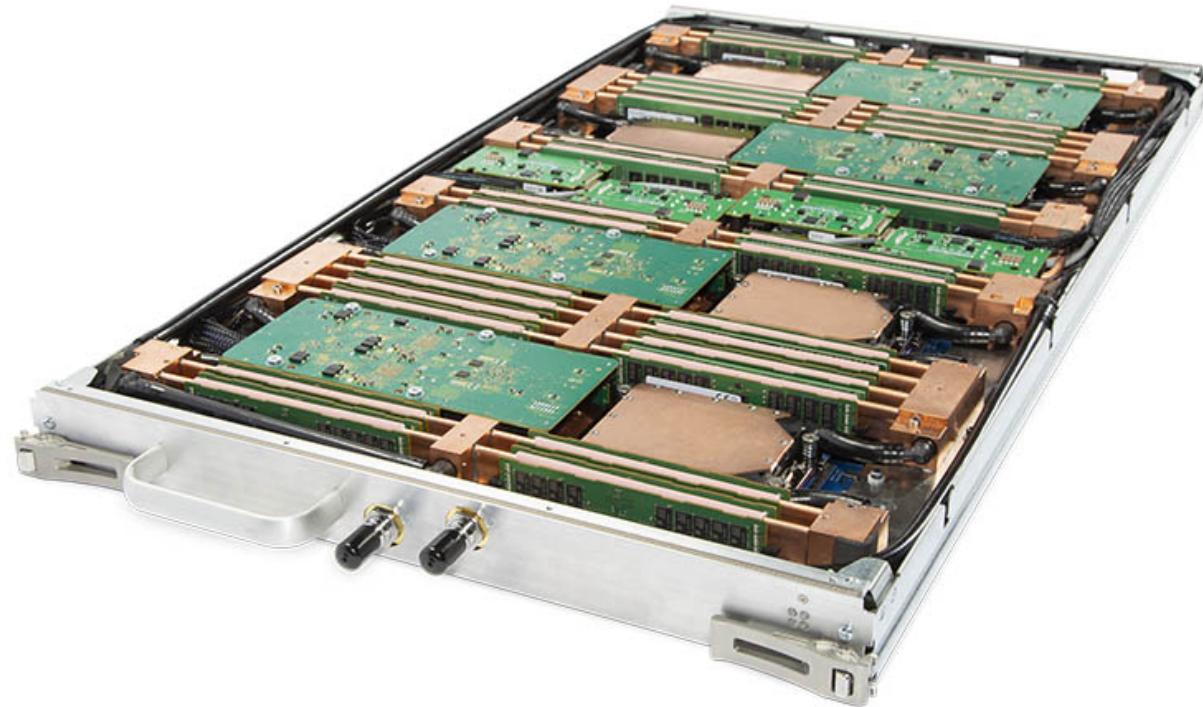


# Node types visible to users

- Login nodes, each:
  - 2x AMD EPYC Zen2 (Rome) 7742 2.25 GHz, 64-core; 512 GB DDR 3200
  - Same processors as compute nodes so no cross-compile required
- Compute nodes, each:
  - 2x AMD EPYC 7742 2.25 GHz, 64-core; 256 or 512 GB DDR 3200
  - More details coming up...
- Data analysis nodes (e.g. pre- and post-processing), each:
  - 2x AMD EPYC 7742 2.25 GHz, 64-core; 512 GB DDR 3200
- Collaboration system
  - 2 nodes each with 8 AMD GPUs
  - Very little known about this at the moment...
- (c.f. ARCHER: Unlike ARCHER, all nodes have the same architecture and instruction set)

# Compute Nodes: AMD EPYC 7742

- Dual socket AMD EPYC (Rome) 7742, 2.25GHz, 64 core
  - 128 cores per node (256 SMT)
  - 8 memory channels per socket
  - DDR 3200MHz
  - Maximum memory bandwidth 381.4 GB/s per node
- 5556 Standard Nodes: 256 GiB
- 292 High Memory Nodes: 512 GiB



# Comparison with ARCHER

	ARCHER2	ARCHER	Ratio
Processors	2x AMD EPYC 7742	2x Intel Xeon E5-2697 v2	
Cores per socket (node)	64 (128)	12 (24)	5.3:1
Cores per NUMA region	8-64	12	
Max. SP GFLOP/s per socket (node)	4608 (9216)	518 (1037)	384.0:1
Max. SP GFLOP/s per core	72	43	1.7:1
Memory per node	256/512 GB	64/128 GB	4.0:1
Memory per core	2.0/4.0 GB	2.7/5.3 GB	0.7:1
Memory channels per socket	8	4	2:1
Memory	DDR 3200	DDR 1866	
Max. memory BW per socket (node)	190.7 (381.4) GB/s	59.7 (119.4) GiB/s	3.2:1
Max. memory BW per core	3.0 GiB/s	5.0 GiB/s	0.6:1

# Comparison with ARCHER (cache and NUMA)

ARCHER2	ARCHER	Ratio
L1 data cache (per core)	32 KiB	32 KiB
L2 cache (per core)	512 KiB	256 KiB
L3 cache (shared)	32 MiB (shared by 8 cores) 256 MiB per socket	30 MiB (shared by 12 cores) 30 MiB per socket

- NUMA (and hence cache structure) is more complex on EPYC Zen2
  - Each socket made up of 8-core *chiplets* with a shared L3 cache
  - 8 chiplets connected together using I/O and memory hub chip
  - The I/O and memory hub allows the socket to be configured to present as different numbers of NUMA domains:
    - 8 NUMA domains per socket: matches the underlying hardware
    - ...other combinations...
    - 1 NUMA domain per socket: I/O and memory hub manages cache
    - Not sure yet what the configuration will be on ARCHER2

# Compute nodes vs ARCHER: Summary



- Much more compute power and memory per node...
- ...but the balance is different
  - Many more cores per node
  - Less memory and memory bandwidth on a per-core basis
  - Different cache structure
- Need to stop thinking about cores as the key resource indicator
  - Think about the whole node and how you can best exploit it
  - Potentially under-populate the node (i.e. not use all cores)
  - Explore use of multithreading per process
- Different interconnect may also change characteristics...

# Interconnect: Cray Slingshot



- Ethernet-compatible, high-performance interconnect
- 2x 100 Gbps bi-directional interfaces per node: 200 Gbps bi-directional BW per node
  - c.f. current ARCHER Aries network supports over 120 Gbps bi-directional BW per node with one network interface per node
- Diameter 3 dragonfly topology
- Supports dynamic routing and quality of service features (e.g. to manage congestion control)

- 4x Cray ClusterStor L300 Lustre file systems, each with 3.6 PB usable storage (/work file systems)
  - No backup
- 1x Cray ClusterStor E1000F solid state storage, 1.1 PB usable
  - No backup
- NetApp FAS8200 providing 1 PB /home file systems
  - Backed-up for disaster recovery

	ARCHER2	ARCHER
/work (Lustre)	14.5 PB	4.4 PB
/home	1 PB	218 TB
Solid state	1.1 PB	N/A



# Software Overview



epcc

- OS: Cray Linux Environment
- Cray Developer Environment (CDE)
  - Software modules (similar to current ARCHER setup)
  - SLURM scheduler and job launcher
  - Compilers
  - MPI and SHMEM
  - Optimised scientific and numeric libraries
  - Optimised parallel Python and R
  - Parallel profiler and debugger
- Singularity containers

- Standard SLURM environment
  - srun parallel launcher (c.f. ARCHER: srun instead of aprun on ARCHER2)
  - Script runs on compute node (c.f. ARCHER: no job launcher nodes on ARCHER2)

```
#!/bin/bash

#SBATCH -J my_job
#SBATCH --ntasks=512
#SBATCH --ntasks-per-node=128
#SBATCH --cores-per-task=1
#SBATCH --time=2:0:0

# Load modules, etc.

# srun to launch the executable using parallel options from sbatch
srun my_parallel_app
```

- Three environments provided by CDE
  - Cray Compiler Environment (CCE)
    - Cray Fortran compiler (not Flang-based)
    - Cray C/C++ compiler based on Clang
  - AMD Optimizing Compiler Collection (AOCC)
    - Flang Fortran compiler
    - Clang C/C++ compiler
  - GNU Compiler Collection (GCC)
    - Gfortran Fortran compiler
    - gcc/g++ C/C++ compilers
  - (c.f. ARCHER: no Intel compilers on ARCHER2)
- All Cray-provided libraries and tools work with these environments
- Dynamic linking by default
  - (c.f. ARCHER: static linking by default on ARCHER)

- Cray Message Passing Toolkit
  - MPI and OpenSHMEM
- Cray LibSci:
  - BLAS and LAPACK
  - CBLAS and LAPACKE
  - BLACS and ScaLAPACK
- FFTW 3
- HDF5
- NetCDF
- (c.f. ARCHER: no PETSc, no Trilinos, no TPSL provided by Cray on ARCHER2)

- Cray high-performance Python (2 and 3) distribution:
  - **numpy, scipy** - built against Cray LibSci
  - **mpi4py** - built against Cray MPT
  - **dask**
  - Can use `pip install --user` to build on top of the distribution
- Cray R distribution:
  - Less details available
  - Includes standard packages and “parallel” package
- (c.f. ARCHER: no Cray-provided Python and R environments on ARCHER)

- Debugging:
  - **gdb4hpc**: HPC version of command-line GDB debugging tool
  - **valgrind4hpc**: HPC version of memory debugging tool
  - **stat**: merged stack and backtrace analysis tool
  - **atp**: scalable core file generator and analysis tool
  - **Cray Comparative Debugging**: side-by-side debugging of two versions of the same application
  - (c.f. ARCHER: no Arm Forge DDT available on ARCHER2)
- Profiling
  - **Cray Performance Analysis Toolkit (CrayPAT)**
  - **PAPI**: support in CrayPAT
  - **Cray Reveal**: run time performance stats combined with source code visualisation (CCE only)
  - (c.f. ARCHER: no Arm Forge MAP available on ARCHER2)

# Singularity Containers

- Singularity will be available
  - See: <https://sylabs.io/>
- Support containers for single node use and multi-node using MPI
- Potential applications:
  - Custom environments for particular applications/workflows
  - Support for legacy and custom versions of software
  - Capturing environments to enhance reproducibility
  - Ease of install of software with complex dependencies
- (c.f. ARCHER: no containers available on ARCHER)

- Modelling and simulation research software
  - e.g. VASP, CASTEP, OpenFOAM, GROMACS
- Additional scientific and numeric libraries
  - e.g. PETSc, Boost, Eigen
- Data analysis tools and software
  - e.g. NCO, CDO, (VisiData!)
- Other software as required
- Initial set based on analysis of use on ARCHER and consultation with UKRI scientific consortia



# Summary



epcc

# Summary

- ARCHER2 is based on new technology
  - New cabinets - Cray Shasta
  - New processors - AMD EPYC Zen2 (Rome)
  - New interconnect - Cray Slingshot
- Compute nodes have much more floating-point power and total memory available compared to ARCHER
  - Efficient use will require investigation of what balance of cores used per node and number of processes/threads to use
  - EPCC ARCHER2 CSE service will provide advice and assistance with this
- A solid-state storage component will be available
  - potentially improve I/O performance for some applications
- Broad research software stack will be available
  - Including support for Python, R and containers
  - Environment will work in a similar way to current ARCHER setup



[www.archer2.ac.uk](http://www.archer2.ac.uk)