The Arm Technology Ecosystem: Current Products and Future Outlook

Dan Ernst, PhD

Advanced Technology
Cray, Inc.

Why is an Ecosystem Important?

- **An Ecosystem is a collection of common material**
  - Developed jointly and shared
  - Developed commercially and sold

- **Design once and reuse is fundamental**
  - Incur most costs only once
  - Spread most of those costs out amongst many partners

- **<u>Greatly</u> lowers the barrier to entry**
  - Creates opportunities where it was previously too expensive

The Old Ways

1. Decide you want to build hardware to address a certain market
2. Design an ISA
3. Design/Build Hardware
4. Write a (good) compiler
5. Write an OS
   - (or get your customer to do it!)
6. Port every piece of software your customer may want to run
7. Go to Market!
8. Go back to step 1

Copyright 2018 Cray Inc

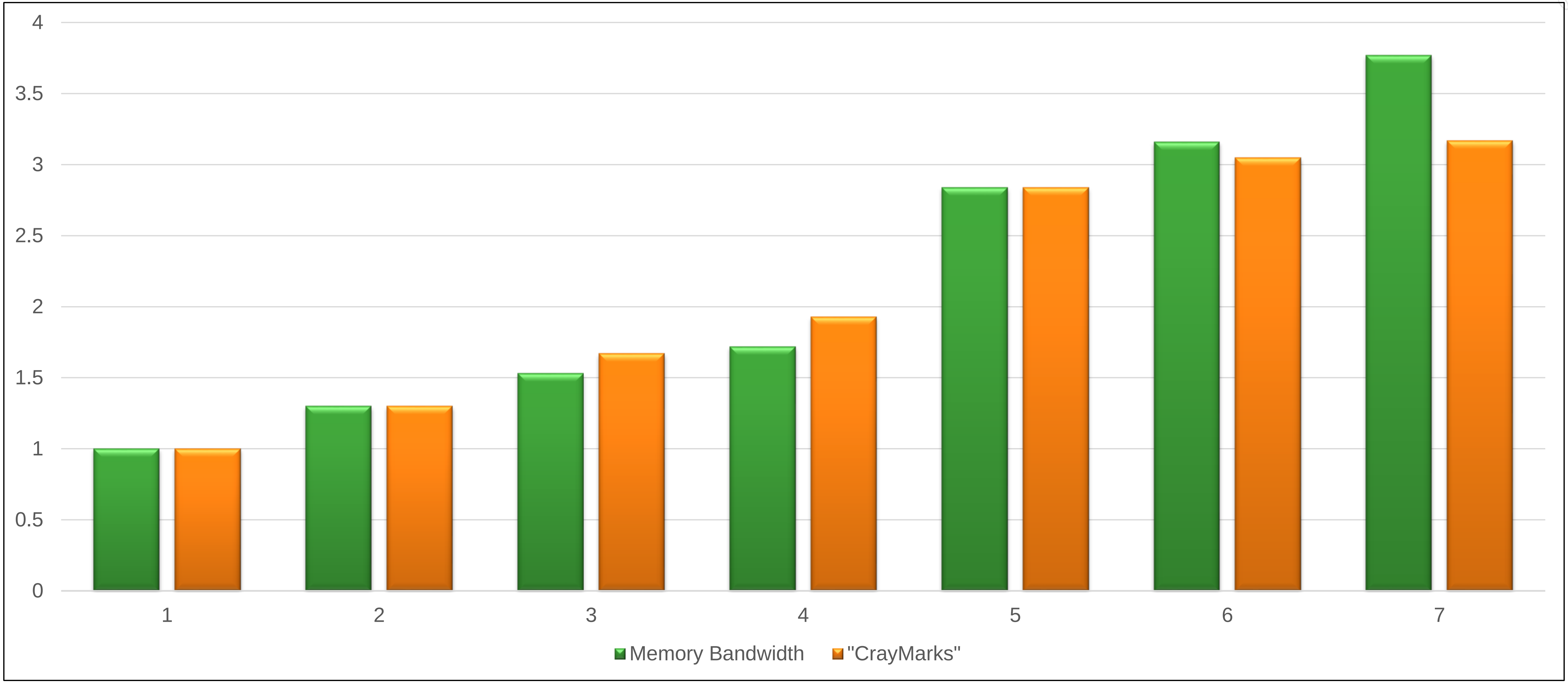
The Ecosystem Way

1. Decide you want to build hardware to address a certain market

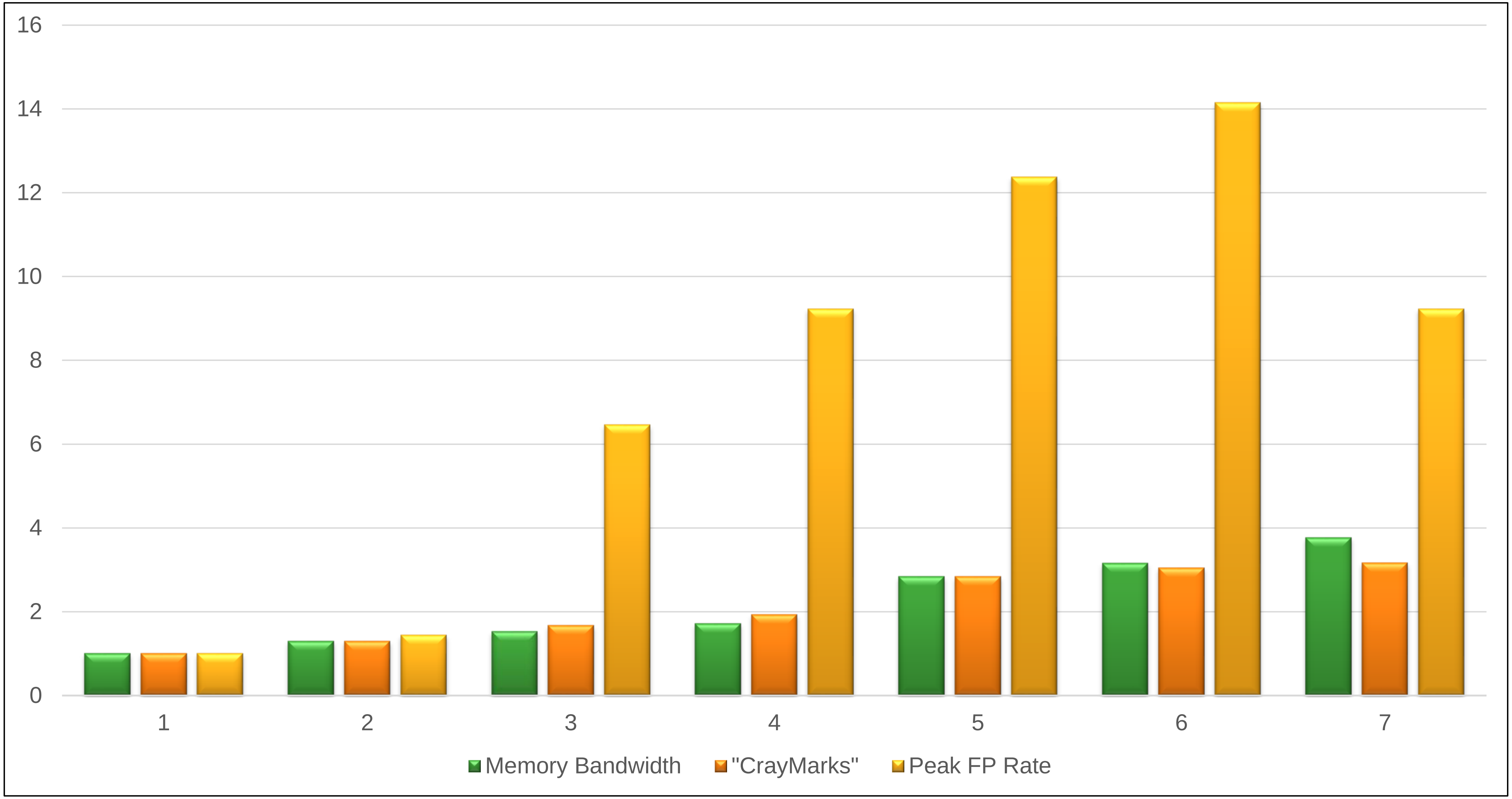
Ecosystem

7. Go to Market!
8. Go back to step 1

Application Performance Correlates with Bandwidth

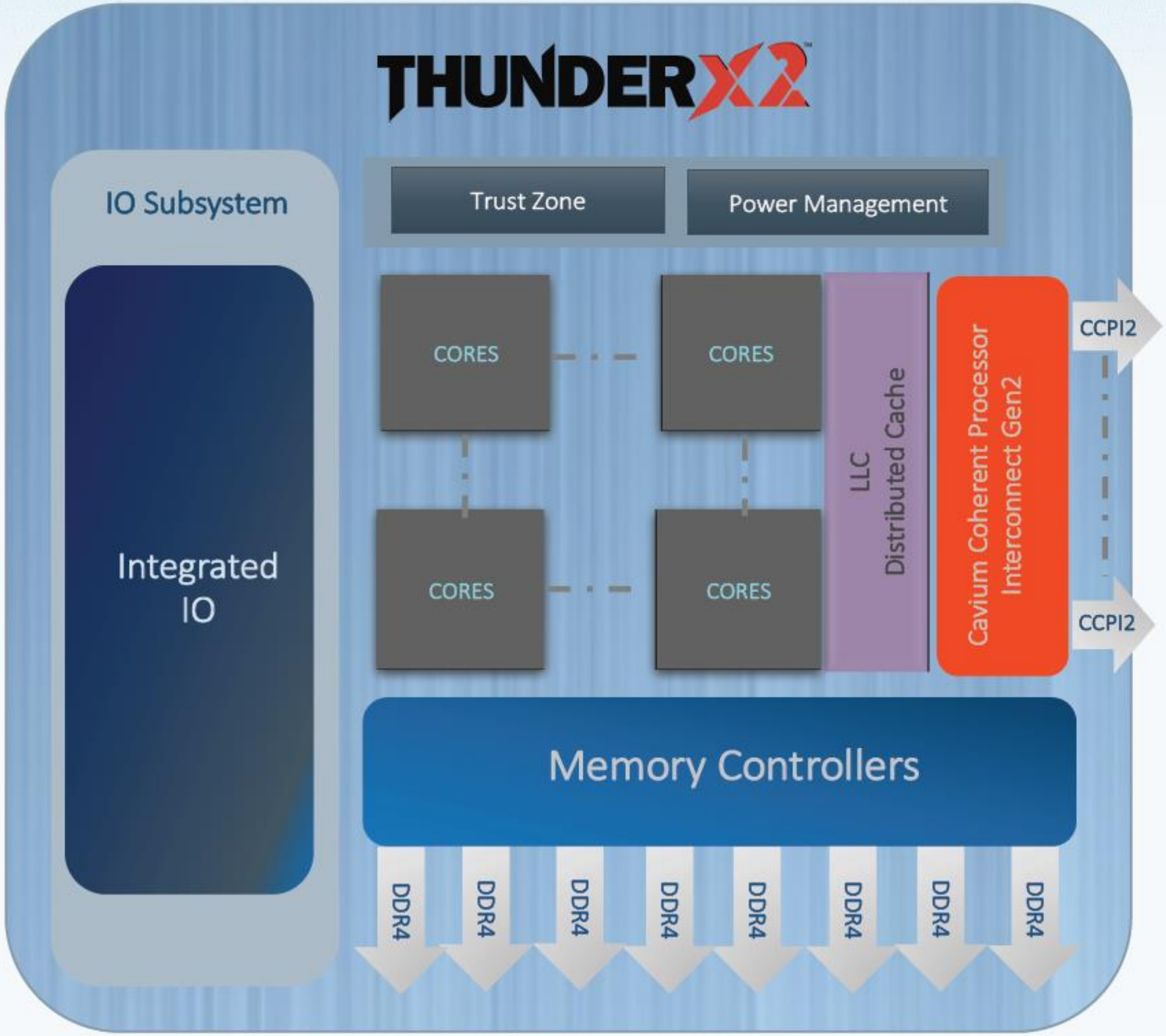


…Not FLOPS



16
14
12
10
8
6
4
2
0
1
2
3
4
5
6
7
🟩 Memory Bandwidth
🟧 "CrayMarks"
🟨 Peak FP Rate

Cavium CN99XX - 1st member of THUNDERX2™ Family

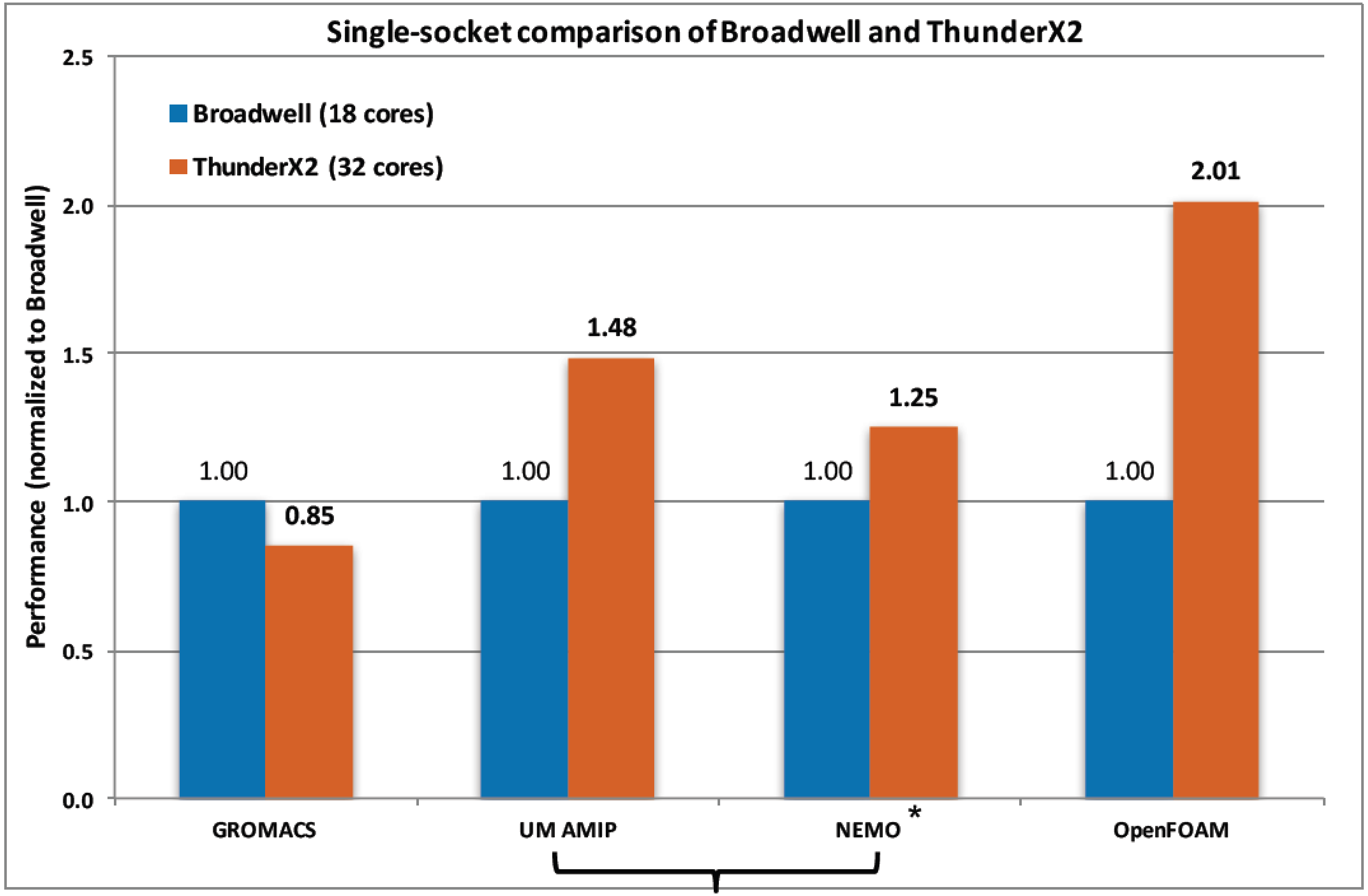


- 24/28/32 Custom Armv8 cores
- Fully Out-Of-Order (OOO) Execution
- 1S and 2S Configuration
- Up to 8 DDR4 Memory Controllers
- Up to 16 DIMMs per Socket
- Server Class RAS features
- Server class virtualization
- Integrated IOs
- Extensive Power Management

2nd gen Arm server SoC

Delivers 2-3X higher performance

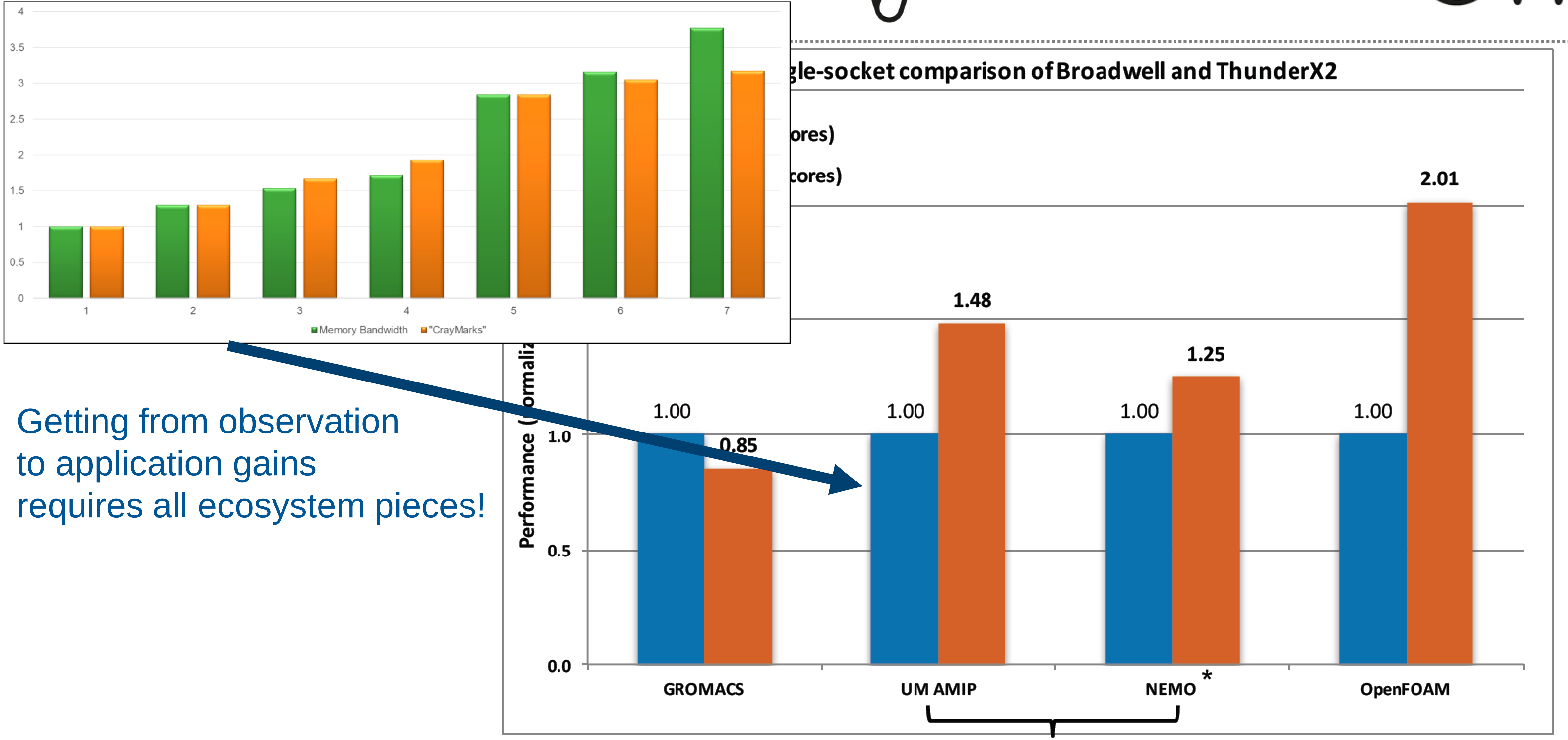
University of BRISTOL Isambard GW4



Benchmarked by the UK's Met Office

* = NEMO runs from a 28 core, 2.0GHz TX2

Application Performance Correlates with Bandwidth



Getting from observation to application gains requires all ecosystem pieces!

Benchmarked by the UK's Met Office

* = NEMO runs from a 28 core, 2.0GHz TX2

Hardware Ecosystem

- **Multiple CPU vendors have product**
  - Cavium and Qualcomm are the biggest
  - More CPU vendors are active
- **OEM/ODMs have developed infrastructure to support these**
- **Core IPs are reaching maturity with NEON and have good ILP**
- **Open server standards for various memory/IO types are available**
  - DDR4, HBM, PCIe, CCIX, OpenCAPI, Gen-Z
- **Future SVE ISA is available and developments are ongoing**

Software Ecosystem

Areas of Interest:

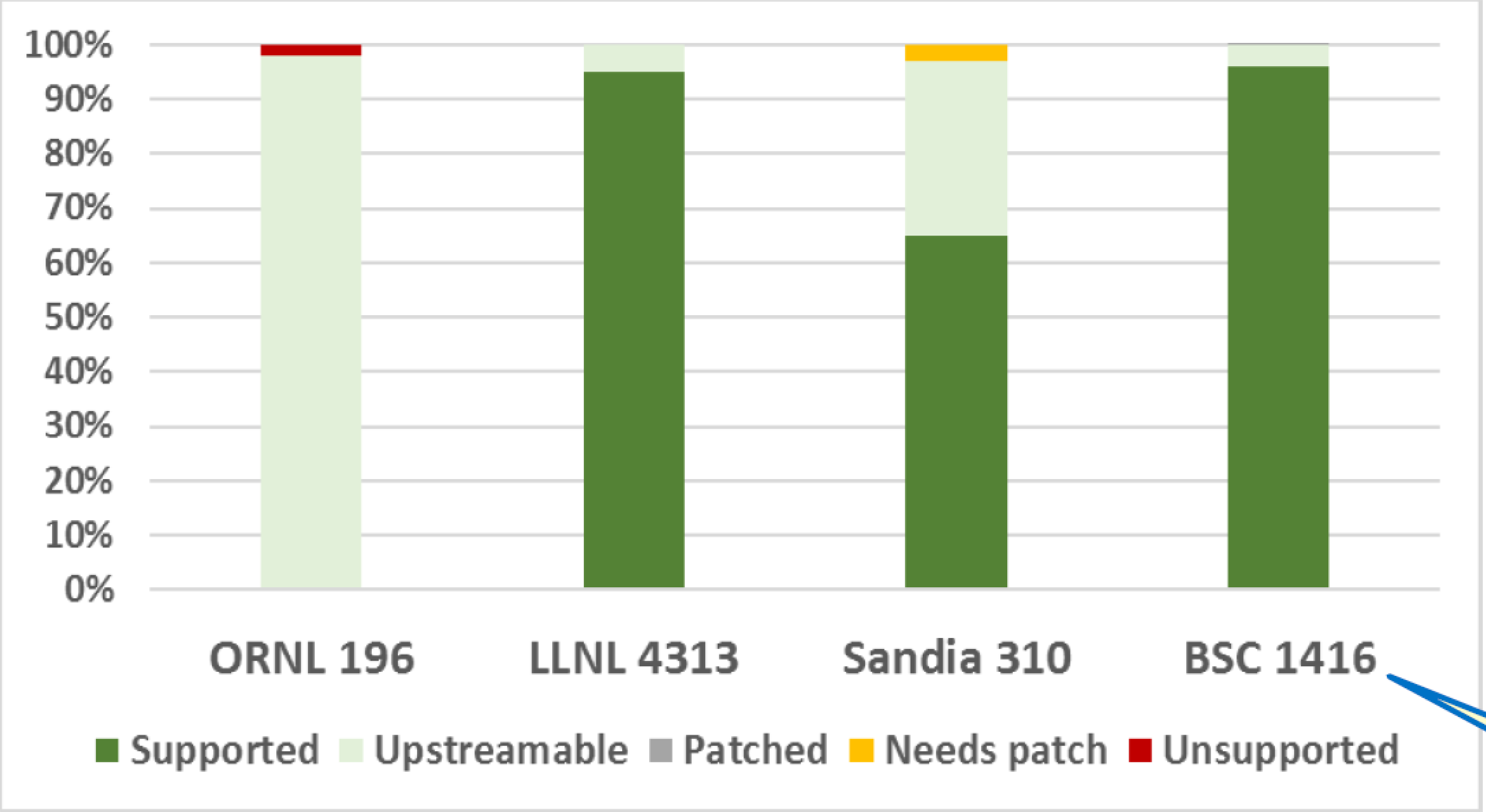
- **OS and related system infrastructure**
  - OS largely a solved problem - fully supported by RHEL, SLES, etc.
  - Support from KVM and other virtualization/container technologies

- **Management features and interfaces**
  - BIOS, system firmware, schedulers and resource management, power management, etc.
  - Some open source (OpenStack, etc.) and vendor-based solutions

- **User-facing software**

From SC17 Panel: “The Arm Software Ecosystem: Are We There Yet?”
Data Collected by CJ Newburn, Nvidia

PACKAGE READINESS



Supported - commercial package officially supports 64-bit ARMv8, e.g. from RHEL, EPEL

Upstreamable - open-source project builds from upstreamed source on Arm with no modifications (apart from tweaking compiler options for performance);

Patched - detailed modification steps for the package are available

Needs Patch - patches are required, but are not documented in detail

Unsupported - haven't made it work yet

pkgs of interest

HPC-Specific Ecosystem

- OpenHPC stack contains large set of widely used open-source HPC software
- As of 1.3.3, official builds for Arm on both SLES and CentOS
  - Arm hardware in the testbed path, so regression is tested
- Maintains source-level compatibility for elements of joint stack
  - Minimizes porting pains

Functional Areas	Components include
Base OS	CentOS 7.4, SLES 12 SP3
Administrative Tools	Conman, Ganglia, Lmod, LosF, Nagios, pdsh, pdsh-mod-slurm, prun, EasyBuild, ClusterShell, mrsh, Genders, Shine, test-suite
Provisioning	Warewulf
Resource Mgmt.	SLURM, Munge
I/O Services	Lustre client (community version)
Numerical/Scientific Libraries	Boost, GSL, FFTW, Metis, PETSc, Trilinos, Hypre, SuperLU, SuperLU_Dist,Mumps, OpenBLAS, Scalapack, SLEPc, PLASMA, ptScotch
I/O Libraries	HDF5 (pHDF5), NetCDF (including C++ and Fortran interfaces), Adios
Compiler Families	GNU (gcc, g++, gfortran), LLVM
MPI Families	OpenMPI, MPICH
Development Tools	Autotools (autoconf, automake, libtool), Cmake, Valgrind,R, SciPy/NumPy, hwloc
Performance Tools	PAPI, IMB, pdtoolkit, TAU, Scalasca, Score-P, SIONLib

ARM® Supercomputing

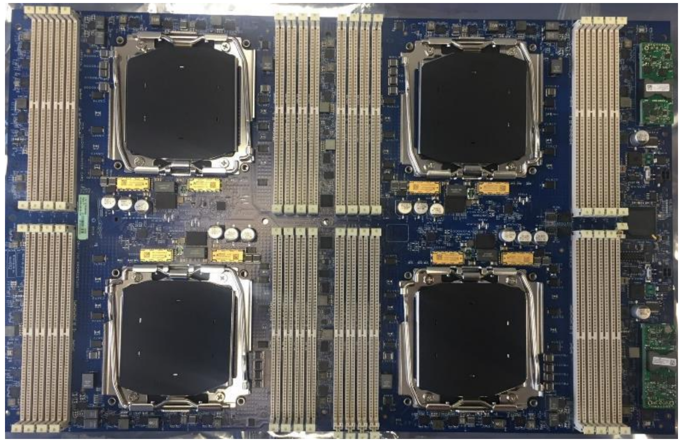
…Made Possible by Cray

CRAY CATAPULTS ARM-BASED PROCESSORS INTO SUPERCOMPUTING

Cray Adds Arm Processors with Complete Software Stack to the Cray XC50 Supercomputer

Seattle, WA – November 13, 2017 – Global supercomputer leader Cray Inc. (Nasdaq: CRAY) today announced the Company is creating the world's first production-ready, Arm®-based supercomputer with the addition of Cavium (Nasdaq: CAVM) ThunderX2™ processors, based…

CAVIUM



CRAY

Programming Environment

Cray Programming Environment for XC Systems

Programming Languages

- 🟪 Fortran
- 🟪 C
- 🟪 C++
- 🟦 Chapel
- 🟪 Python
- 🟪 R

Programming Models

Distributed Memory

- 🟦 Cray MPI
- 🟦 SHMEM

Shared Memory / GPU

- 🟦 OpenMP OpenACC
- 🟪 Global Arrays

PGAS & Global View

- 🟦 UPC
Fortran coarrays
Coarray C++
Chapel

Programming Environments

PrgEnv-

- 🟦 Cray Compiling Environment PrgEnv-cray
- 🟩 GNU PrgEnv-gnu
- ▨🟩 3rd Party compilers (Intel, PGI) PrgEnv-intel, etc.

Libraries

Scientific Libraries

- 🟦 LAPACK
- 🟦 ScaLAPACK
- 🟦 BLAS
- 🟦 Iterative Refinement Toolkit
- 🟪 FFTW

I/O Libraries

- 🟩 NetCDF
- 🟩 HDF5

Development Tools

Environment setup

- 🟪 Modules

Debuggers

- ▨ TotalView
- ▨ DDT
- 🟦 gdb4hpc

Debugging Support

- 🟦 Abnormal Termination Processing (ATP)
- 🟩 STAT

Development Tools

Performance Analysis

- 🟦 CrayPAT
- 🟦 Cray Apprentice^2

Porting

- 🟦 Reveal
- 🟦 CCDB

ML Frameworks

- 🟦 Cray PE ML Plugin for TensorFlow

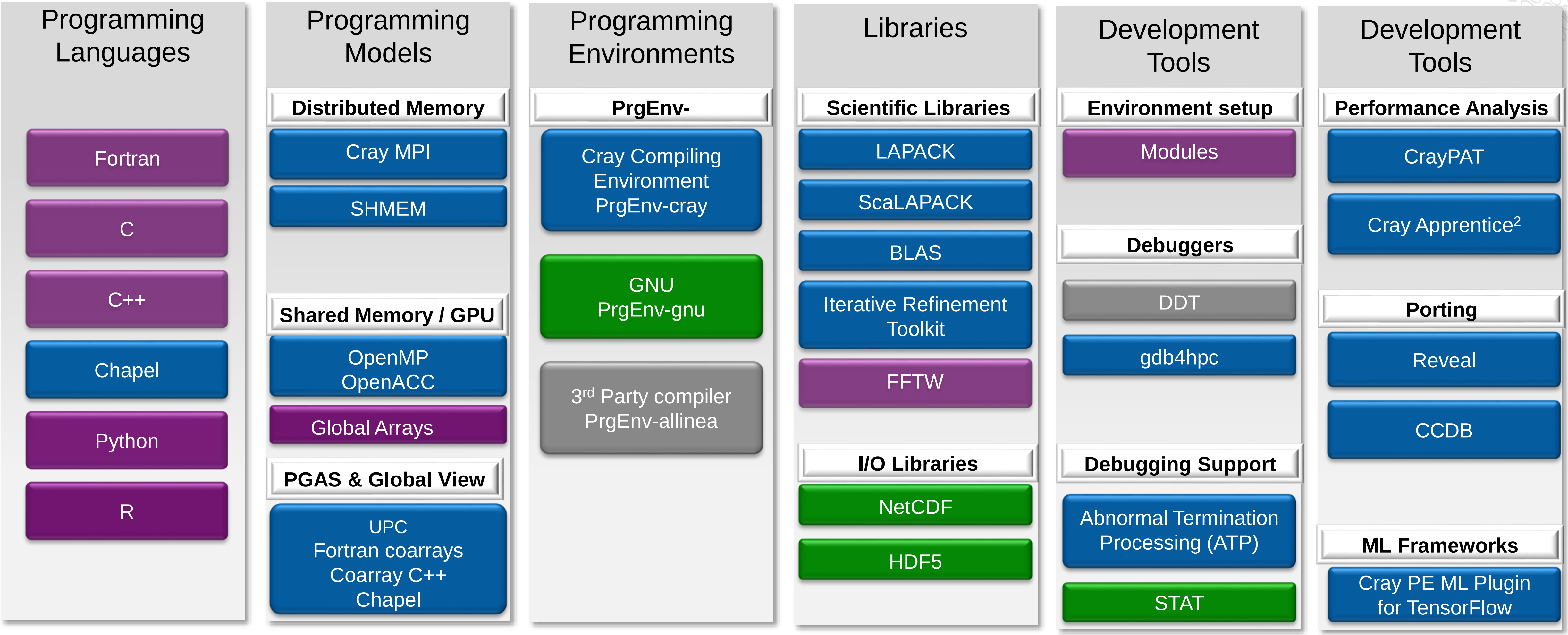
🟦 Cray Developed

🟪 Cray added value to 3rd party

🟩 3rd party packaging

▨ Licensed ISV SW

Cray Programming Environment for XC50 with ARM



Programming Languages

- Fortran
- C
- C++
- Chapel
- Python
- R

Programming Models

Distributed Memory

- Cray MPI
- SHMEM

Shared Memory / GPU

- OpenMP
OpenACC
- Global Arrays

PGAS & Global View

- UPC
Fortran coarrays
Coarray C++
Chapel

Programming Environments

PrgEnv-

- Cray Compiling Environment
PrgEnv-cray
- GNU
PrgEnv-gnu
- 3rd Party compiler
PrgEnv-allinea

Libraries

Scientific Libraries

- LAPACK
- ScaLAPACK
- BLAS
- Iterative Refinement Toolkit
- FFTW

I/O Libraries

- NetCDF
- HDF5

Development Tools

Environment setup

- Modules

Debuggers

- DDT
- gdb4hpc

Debugging Support

- Abnormal Termination Processing (ATP)
- STAT

Development Tools

Performance Analysis

- CrayPAT
- Cray Apprentice2

Porting

- Reveal
- CCDB

ML Frameworks

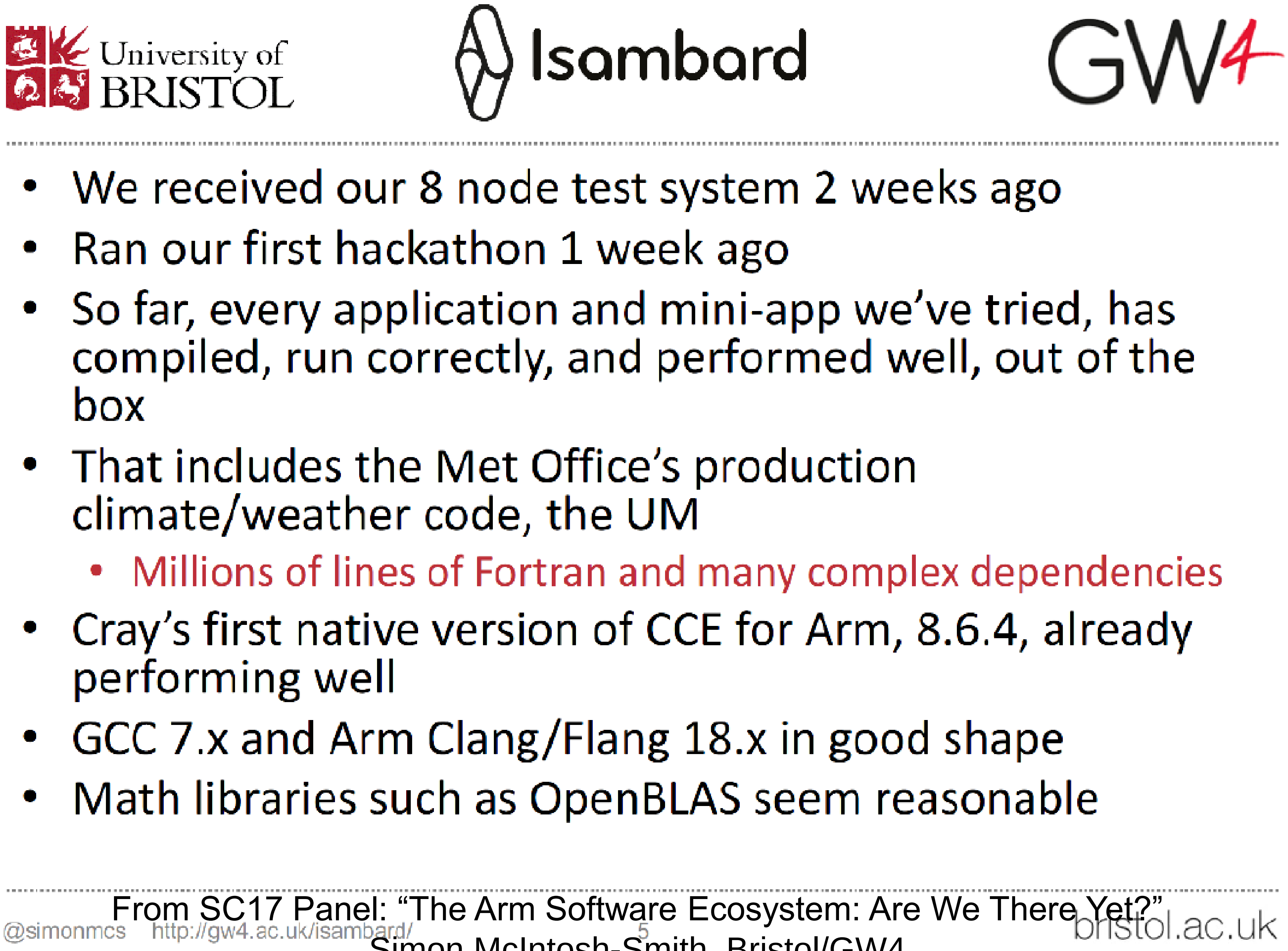
- Cray PE ML Plugin for TensorFlow

🟦 Cray Developed

🟪 Cray added value to 3rd party

🟩 3rd party packaging

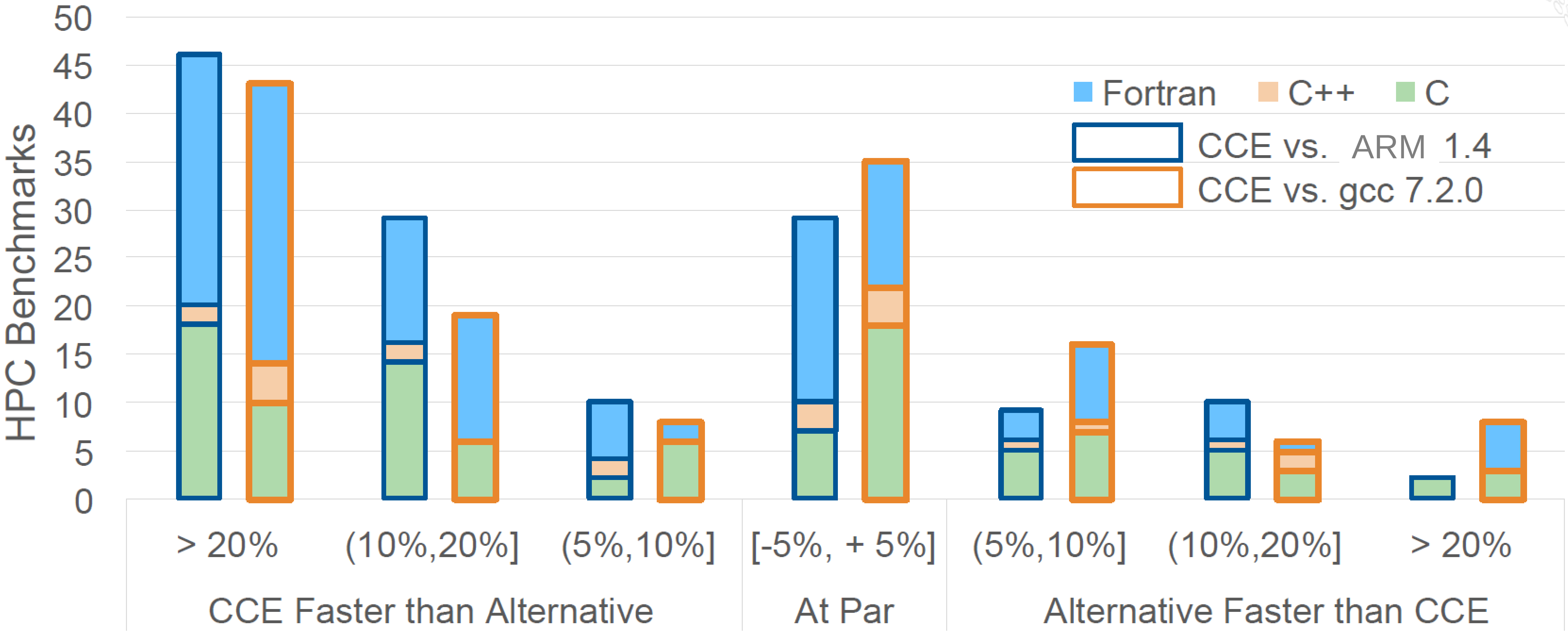
▨ Licensed ISV SW



- We received our 8 node test system 2 weeks ago
- Ran our first hackathon 1 week ago
- So far, every application and mini-app we've tried, has compiled, run correctly, and performed well, out of the box
- That includes the Met Office's production climate/weather code, the UM
  - Millions of lines of Fortran and many complex dependencies
- Cray's first native version of CCE for Arm, 8.6.4, already performing well
- GCC 7.x and Arm Clang/Flang 18.x in good shape
- Math libraries such as OpenBLAS seem reasonable

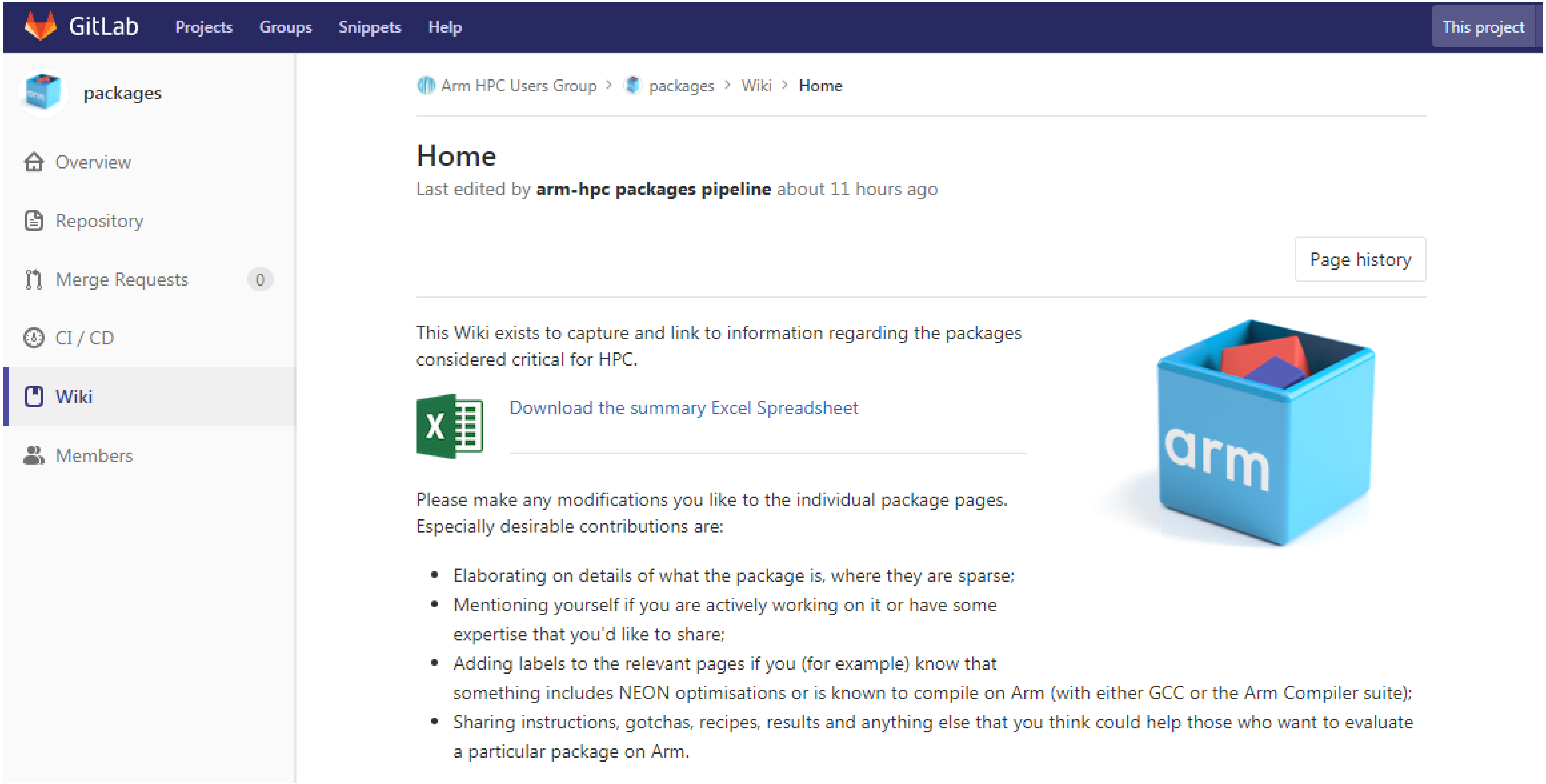
From SC17 Panel: “The Arm Software Ecosystem: Are We There Yet?”
Simon McIntosh-Smith, Bristol/GW4

Benchmarks: CCE vs Alternative Arm Compilers



Notable Gaps

- **Some IO bits**
  - Lustre is functional, but tuning is in process
  - GPFS client?
- **Nvidia Tesla stack**
  - Basics are there, but
  - Significant gaps where porting has not happened
- **ISV codes, generally**
  - Always the last to move to new things
  - Need to be motivated by customer deployments or market indications



Home

Last edited by **arm-hpc packages pipeline** about 11 hours ago

This Wiki exists to capture and link to information regarding the packages considered critical for HPC.

Download the summary Excel Spreadsheet

Please make any modifications you like to the individual package pages. Especially desirable contributions are:

- Elaborating on details of what the package is, where they are sparse;
- Mentioning yourself if you are actively working on it or have some expertise that you'd like to share;
- Adding labels to the relevant pages if you (for example) know that something includes NEON optimisations or is known to compile on Arm (with either GCC or the Arm Compiler suite);
- Sharing instructions, gotchas, recipes, results and anything else that you think could help those who want to evaluate a particular package on Arm.

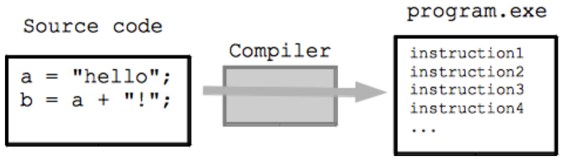
https://gitlab.com/arm-hpc/packages/wikis/home

Ecosystem for Future Technologies

Partnership on Future Architectures

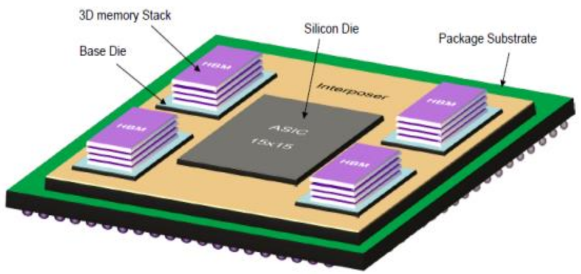
- Cray and Arm have collaborated since 2013 on future HPC architectures:
- Scalable Vector Extension (SVE), which leverages ISA elements pioneered by Cray systems and compilers
- Developing the ecosystem of HPC-relevant technologies, including open interfaces, memories, and software infrastructure
- Partnered with DOE through FastForward 2 and PathForward contracts to understand impact on end-user applications

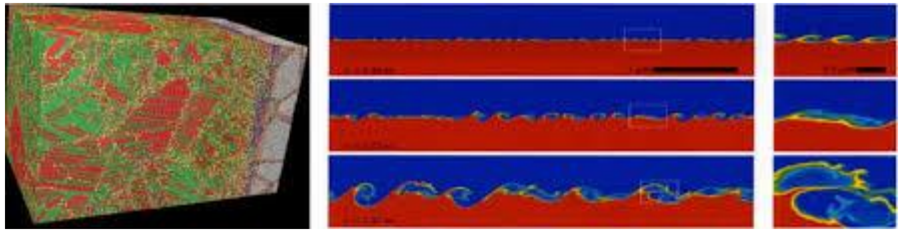
Scalable
Vector
Extension



```
a = "hello";
b = a + "!";
```

```
instruction1
instruction2
instruction3
instruction4
...
```





Ecosystem for Future Technologies

- Tools development has started across the industry for SVE

- Arm Research has early tools for building/emulation of SVE codes
  - Currently decoupled from performance models

- Work underway from various vendors (Arm, Fujitsu, Cray, etc.) to make more toolchains available
  - ABIs and base libraries
  - QEMU
  - LLVM support

- Point of interaction: what components/tools would the applications community find most useful in exploring these technologies?

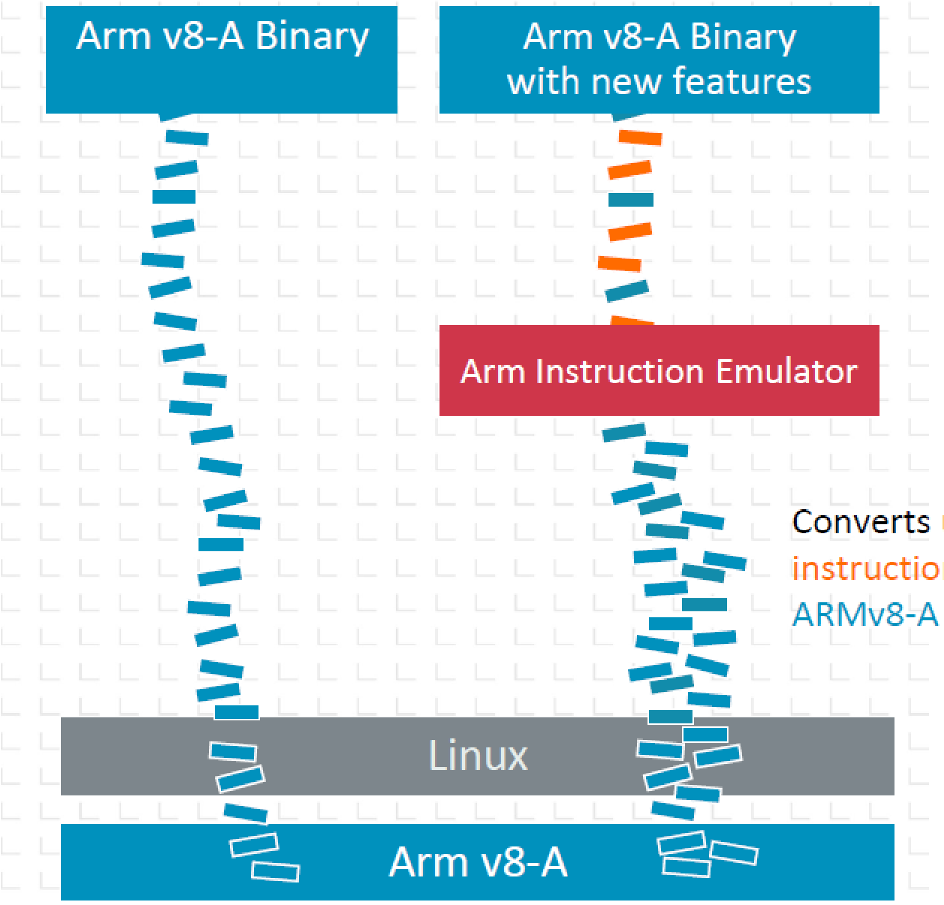
Arm Instruction Emulator

Develop your user-space applications for future hardware today

Run Linux user-space code that uses new hardware features (SVE) on current Arm hardware

Simple “black box” command line tool

```
$ armclang hello.c --march=armv8+sve
$ ./a.out
Illegal instruction
$ armie -a=armv8+sve ./a.out
Hello
```



Arm v8-A Binary

Arm v8-A Binary with new features

Arm Instruction Emulator

Linux

Arm v8-A

Converts unsupported instructions to native ARMv8-A instructions

Source: Eric Van Hensbergen, Arm Research

Open Call: What Would Help?

What needs to be ported?
What resources are necessary?