Arm in High Performance Computing:

Fortran on AArch64

Nathan Sircombe

Arm Manchester

nathan.sircombe@arm.com

© 2018 Arm Limited

70%

of the world's population
uses Arm technology

© 2018 Arm Limited

Total computing experience

Arm defines the pervasive intelligence shaping today's connected world, transforming solutions everywhere compute happens.

As the foundation of a global ecosystem of technology innovators, we empower the world's most successful business and consumer brands with computing everywhere.

Total Computing experience.

Consumer

Automotive

Smart Everything

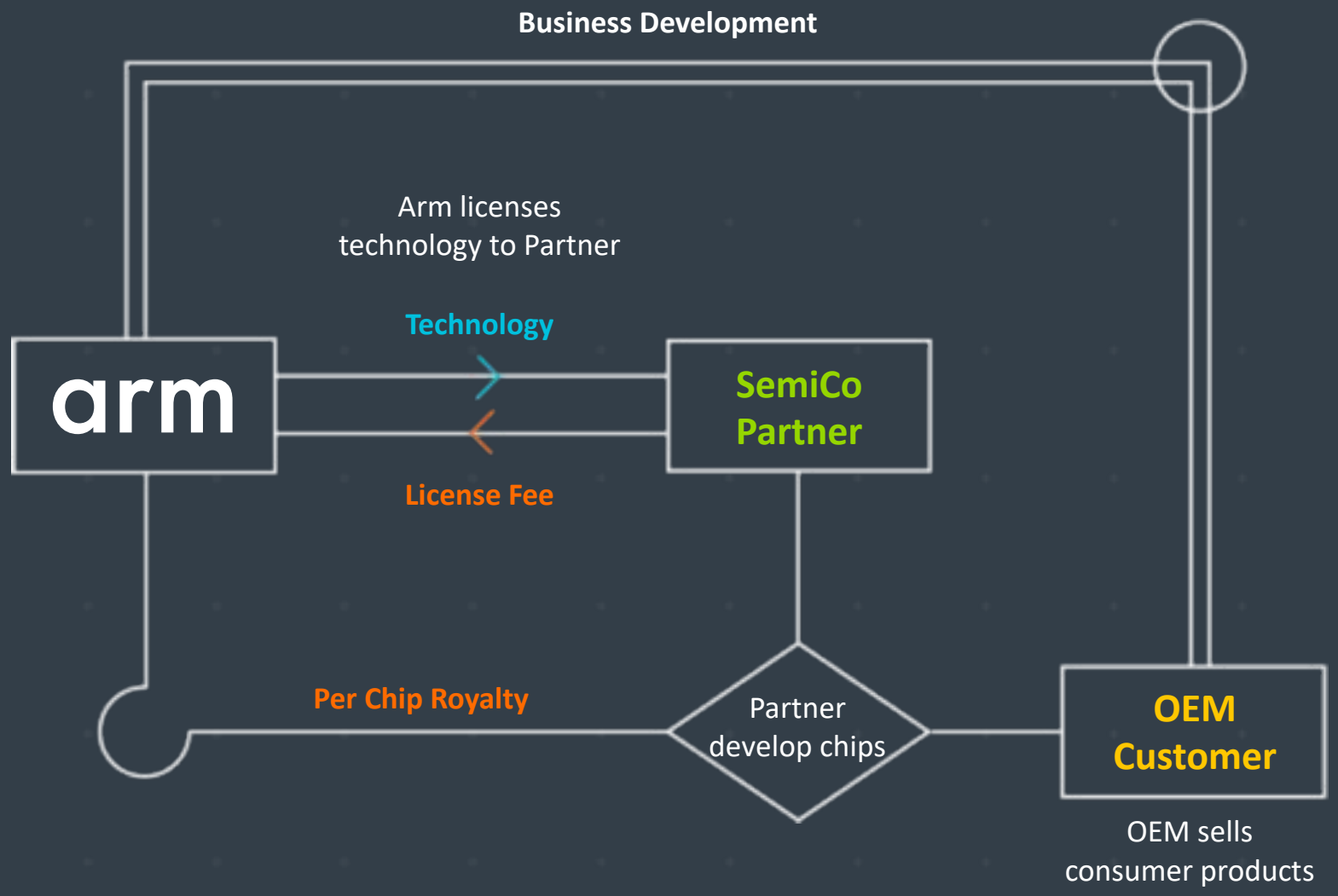
Enterprise

© 2018 Arm Limited

A continuous partnership model

Arm develops technology that is licensed to semiconductor companies.

Arm receives an upfront license fee and a royalty on every chip that contains its technology.

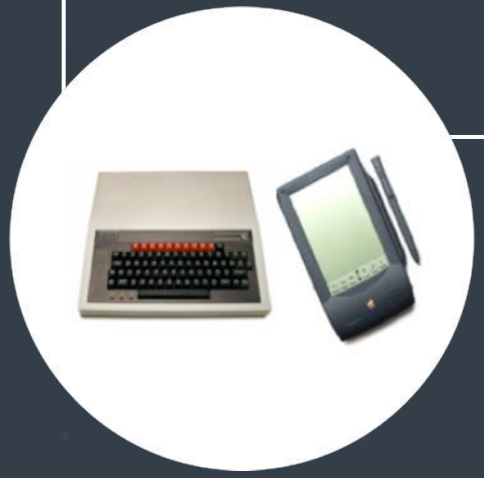


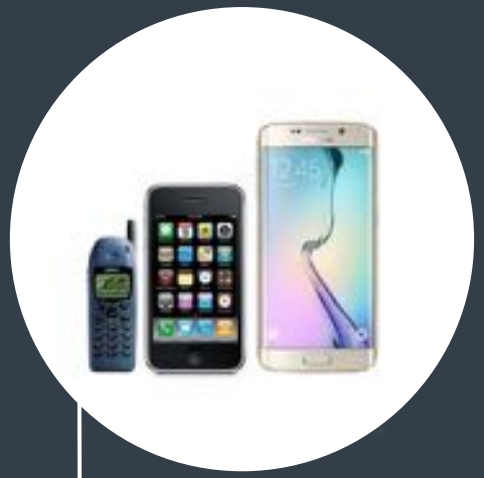
© 2018 Arm Limited

From inception to now

1990

Joint venture between Acorn Computers and Apple.





Designed into first mobile phones and then smartphones.

1993 onwards

Today

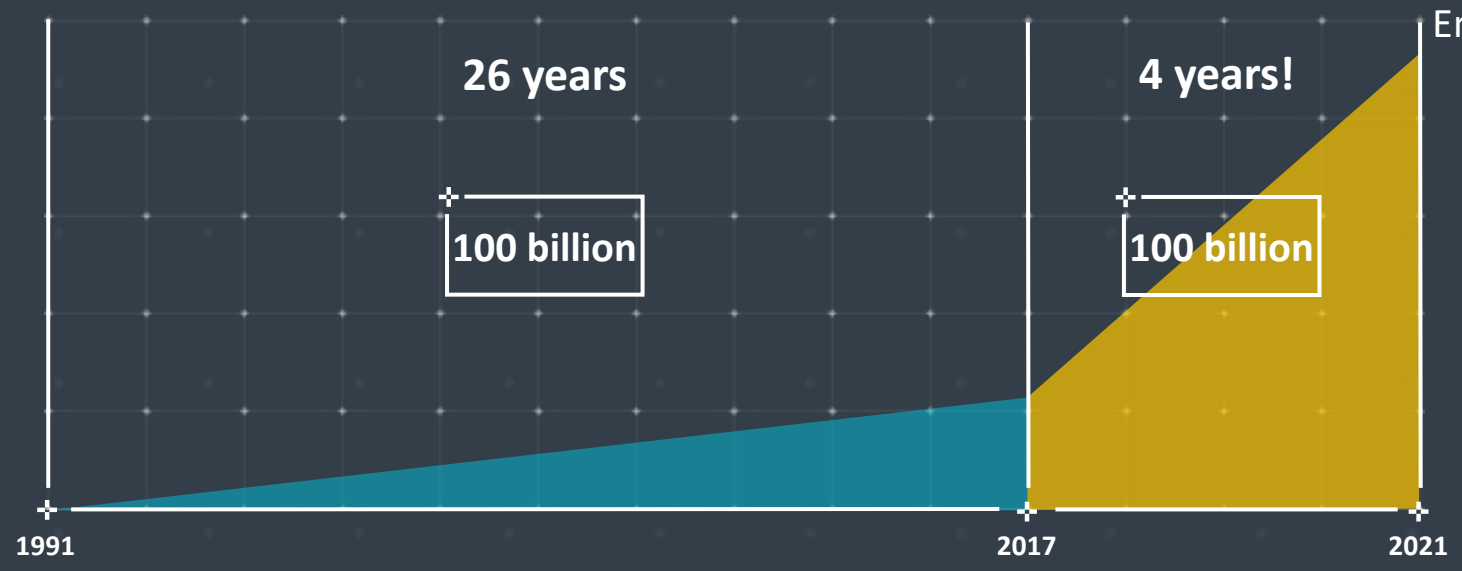
Now all electronic devices can use intelligent Arm technology.

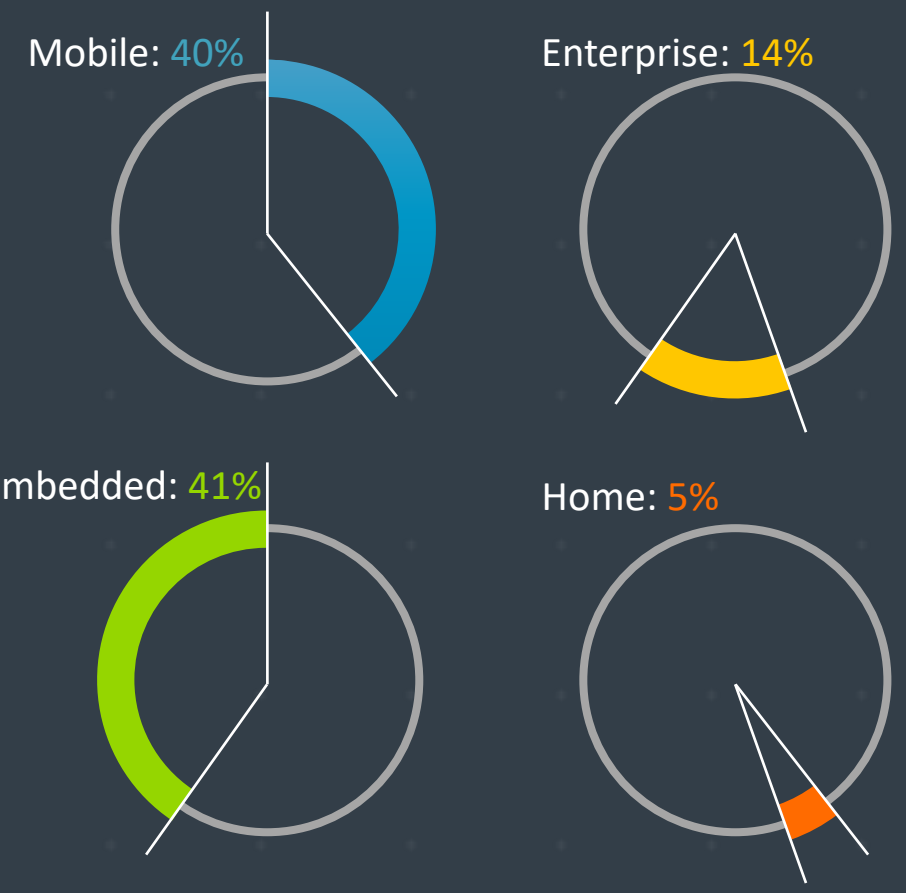
© 2018 Arm Limited

The next 100 billion in 4 years

Another 100 billion chips are projected to ship from 2017 – 2021.

14% in enterprise systems





© 2018 Arm Limited

Arm in High Performance Computing

From AArch64, to Arm Allinea Studio, major platform deployments and beyond...

2012: Arm launches its 64-bit instruction set – AArch64

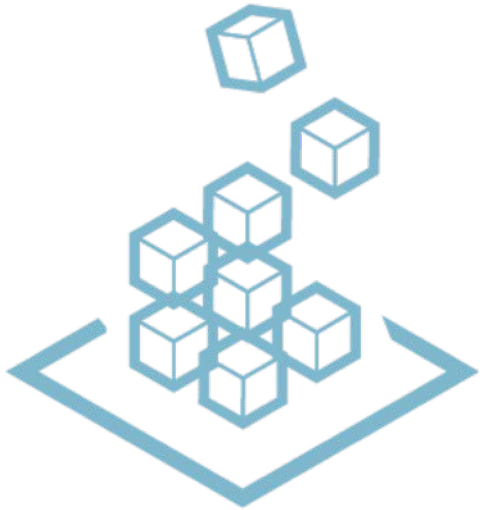
opening up the worlds of **enterprise** servers, **data centres** and **High Performance Computing** (HPC)

2017: Release of Arm's Fortran compiler

compilers and libraries alongside market-leading debug and optimization tools
Complement the quality open source solutions available on Arm today

The Arm server ecosystem now has access to an end-to-end commercial suite, in addition to proven open-source tools, for building and porting HPC applications

2018: large main-stream deployments of our 64-bit server platforms in HPC



© 2018 Arm Limited

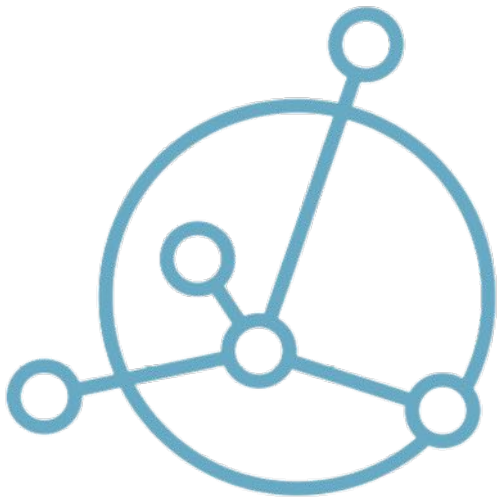
Arm in High Performance Computing

From AArch64, to Arm Allinea Studio, major platform deployments and beyond...

Major Linux distributions now support the Armv8-A architecture
with Red Hat and SUSE announcing enterprise-level Arm server support

Arm's ecosystem is built on partnership and choice
we work with many organizations to drive hardware design and deliver better software

Building the software ecosystem and tools is an important part of this story
We enhance open source software as well as developing commercially supported options



© 2018 Arm Limited

Cross-platform debug and profile tools

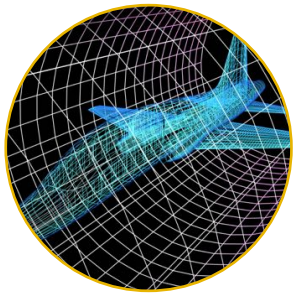
Arm-only Compiler and Libraries

arm ALLINEA STUDIO

Forge (DDT and MAP) and **Performance Reports** with support for **Arm**

Arm Fortran, C & C++ Compilers, *interoperable with Forge and Performance Reports*
Arm Performance Libraries

arm ALLINEA STUDIO

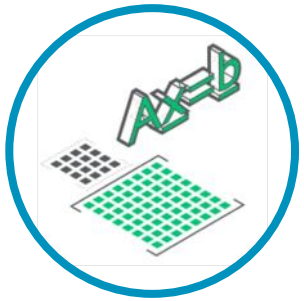


Fortran Compiler

- Fortran 2003 support
- Partial Fortran 2008 support
- OpenMP 3.1
- Directives to support explicit vectorization control
- SVE ready

C/C++ Compiler

- C++ 14 support
- OpenMP 4.5 without offloading
- SLEEF vector maths
- SVE ready



Performance Libraries

- Optimized math libraries
- BLAS, LAPACK and FFT
- Threaded parallelism with OpenMP
- Optimized maths intrinsics

Forge

- Profile, Tune and Debug
- Scalable debugging with DDT
- Parallel Profiling with MAP

Performance Reports

- Analyze your application
- Memory, MPI, Threads, I/O, CPU metrics

Tuned by Arm for server-class Arm-based platforms

© 2018 Arm Limited

arm COMPILER

Commercial Fortran/C/C++ compiler with best-in-class performance

Compilers tuned for Scientific Computing and HPC

Latest features and performance optimizations

Commercially supported by Arm

Tuned for Scientific Computing, HPC and Enterprise workloads

- Processor-specific optimizations for various server-class Arm-based platforms
- Optimal shared-memory parallelism using latest Arm-optimized OpenMP runtime

Linux user-space compiler with latest features

- C++ 14 and Fortran 2003 language support with OpenMP 4.5*
- Support for Armv8-A and SVE architecture extension
- Based on LLVM and Flang, leading open-source compiler projects

Commercially supported by Arm

- Available for a wide range of Arm-based platforms running leading Linux distributions – RedHat, SUSE and Ubuntu

© 2018 Arm Limited

arm PERFORMANCE LIBRARIES

Commercial 64-bit Armv8-A math libraries

- Commonly used low-level math routines - BLAS, LAPACK and FFT
- Optimised maths intrinsics
- Validated with NAG's test suite, a de-facto standard

Best-in-class performance with commercial support

- Tuned by Arm for specific cores like the Thunder X2 and Cortex-A72
- Maintained and supported by Arm for a wide range of Arm-based SoCs

Silicon partners can provide tuned micro-kernels for their SoCs

- Partners can contribute directly through open source route
- Parallel tuning within our library increases overall application performance

Best-in-class performance

Commercially Supported by Arm

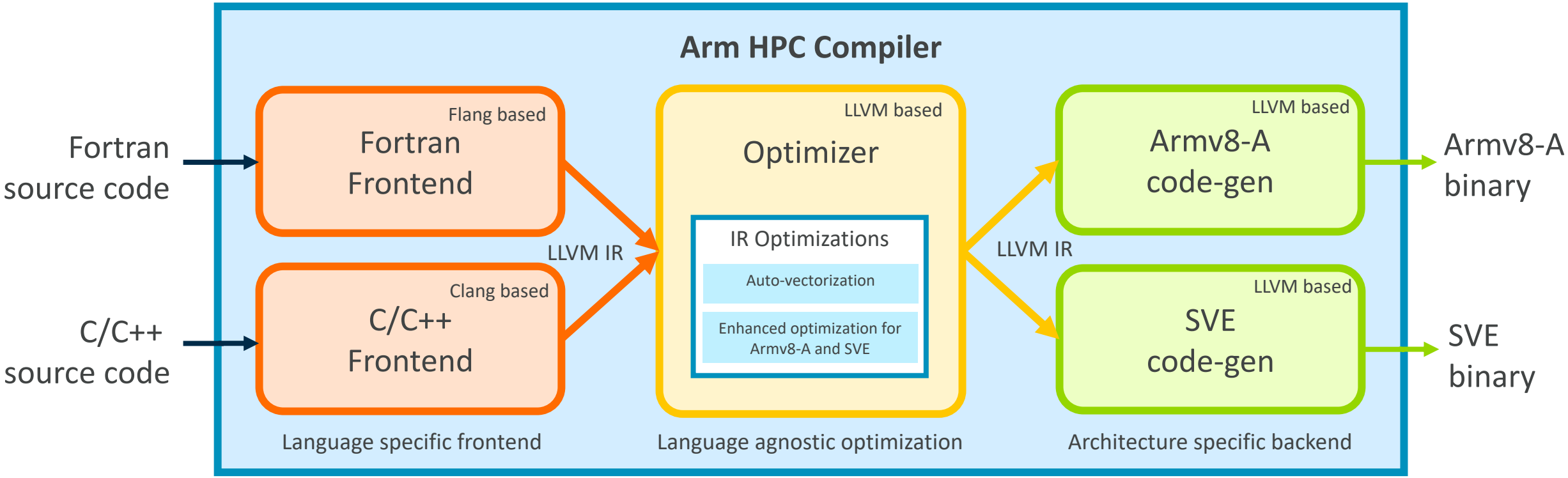
Validated with NAG test suite

© 2018 Arm Limited

Arm Compiler

Building on LLVM, Clang and Flang

Combines broad-based, community, cross-platform, Open-Source development and dedicated Arm-specific engineering to deliver an AArch64 optimised HPC compiler suite.



 © 2018 Arm Limited

F18 – a new Flang

Flang originates from an NNSA funded project to develop an OS Fortran compiler, based on PGI's commercial Fortran compiler

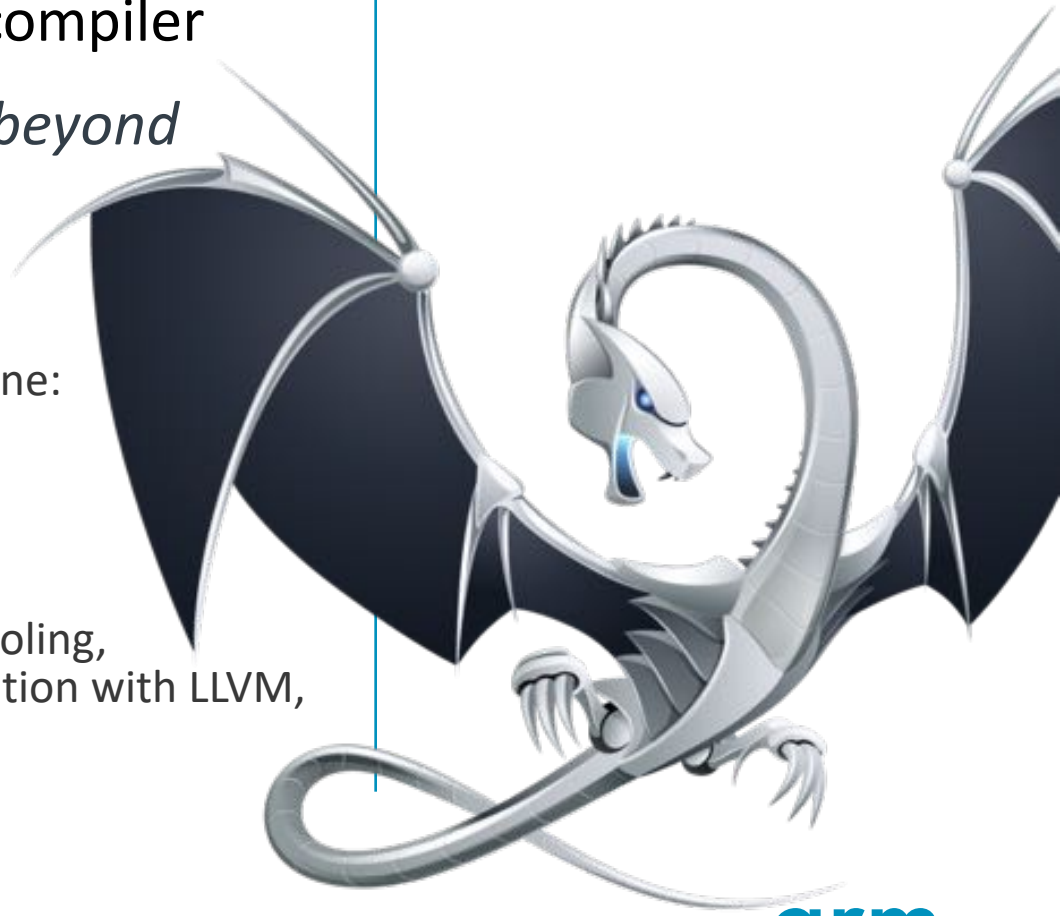
F18 is a Fortran frontend targeting **Fortran 2018**, *and beyond*

- Apache licence
- A 'clang quality' front end for Fortran
- Currently under development; and Open-Source project from day one:

https://github.com/flang-compiler/f18

Flang can be adopted into **LLVM**

- language, structural issues, support for development of compiler tooling, and presence of 'legacy' code currently complicates a closer integration with LLVM, preventing it from being part of the upstream LLVM repository



© 2018 Arm Limited

Working with the Arm Fortran Compiler

© 2018 Arm Limited

arm

Support for common compiler flags

And AArch64 specific flags

The **Arm Fortran Compiler** accepts **GCC's gfortran** flags wherever possible

Handy **orientation guides** for ifort, pgfortran and gfortran developers available at:

https://developer.arm.com/products/software-development-tools/hpc/arm-allinea-studio/compiler-orientation-guides

The **Arm Fortran Compiler's** '`-mcpu`' flag can be set to build for a specific AArch64 target

```
-mcpu=ThunderX2T99
```

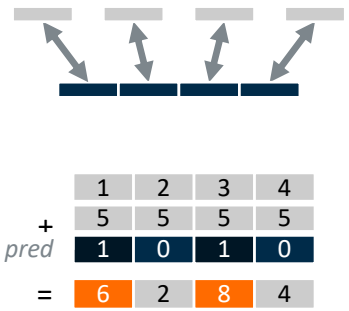
or set to 'native' to automatically detect the host implementation.

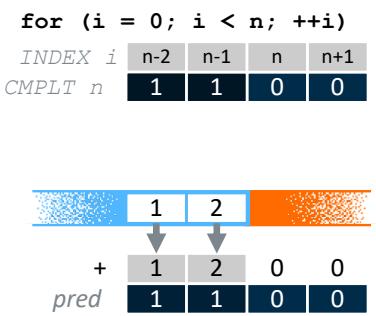
Architecture revision can be manually specified using '`-march`'

 © 2018 Arm Limited

Scalable Vector Extension (SVE)

A vector extension to the Armv8-A architecture with some major new features:





Gather-load and scatter-store

Loads a single register from several non-contiguous memory locations.

Per-lane predication

Operations work on individual lanes under control of a predicate register.

Predicate-driven loop control and management

Eliminate scalar loop heads and tails by processing partial vectors.

Vector partitioning and software-managed speculation

First Faulting Load instructions allow memory accesses to cross into invalid pages.

No preferred vector width

The above features allow the production of compiled binaries that are agnostic to hardware vector width (which can be between 128-2048 bit at 128 bit increments).

© 2018 Arm Limited

Arm Instruction Emulator

Based on DynamoRIO

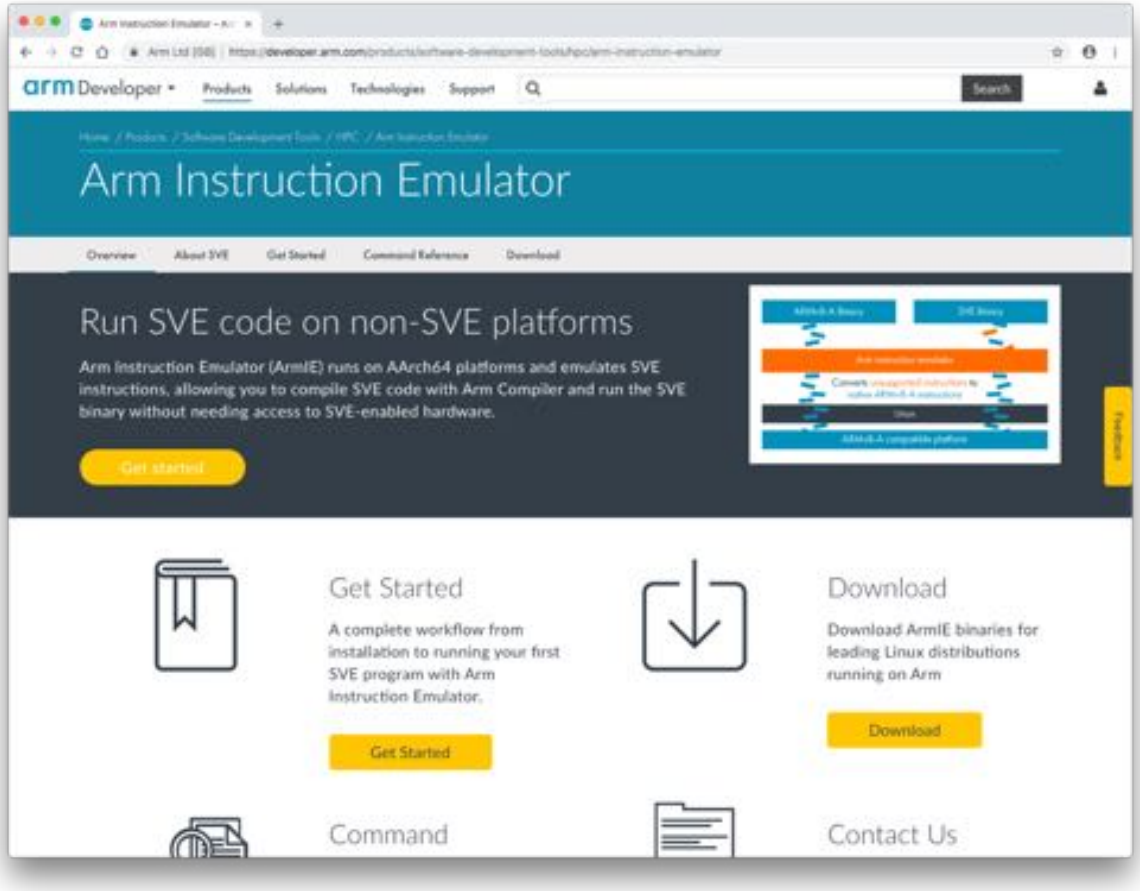
Build Fortran, C and C++ binaries generating SVE instructions by setting:

```
-march=armv8a+sve
```

...then run them on **Arm** hardware **now** using **Arm Instruction Emulator**

Set vector length at runtime:

```
armie -msve-vector-bits=512 ./hello_world
```



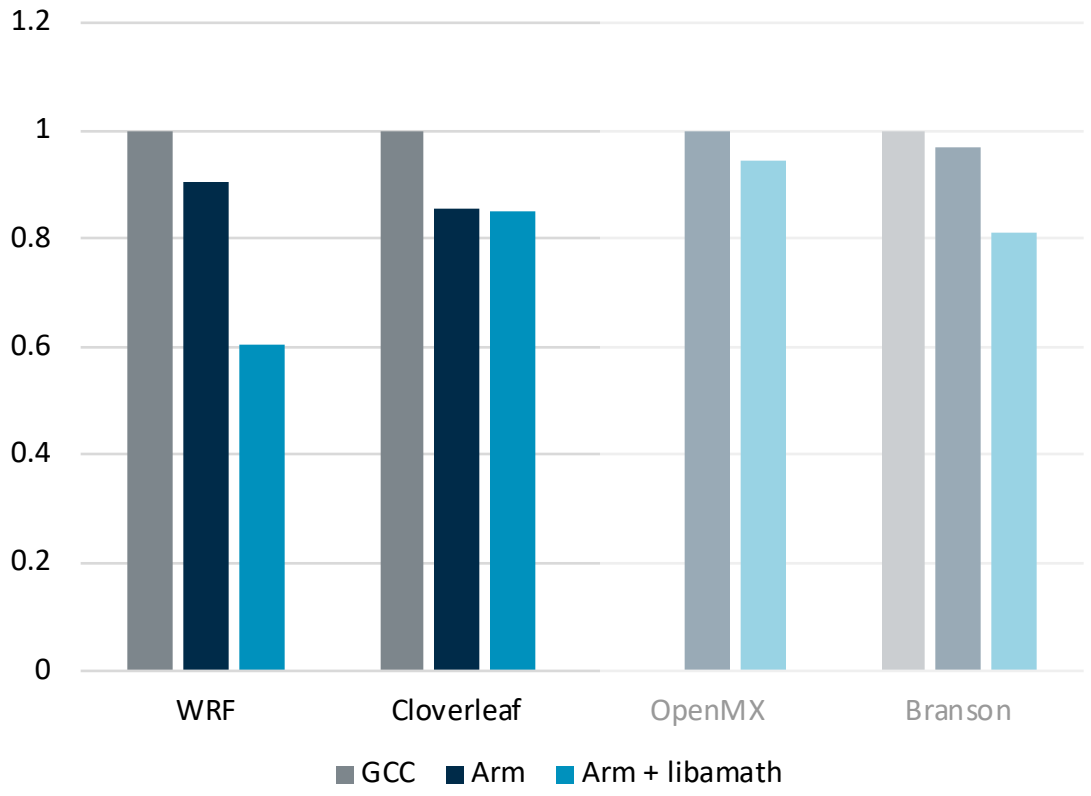
https://developer.arm.com/products/software-development-tools/hpc/arm-instruction-emulator/

© 2018 Arm Limited

Arm-optimized maths intrinsics

libamath

Normalised runtime



Arm PL provides libamath

- With Arm PL module loaded, include `-lamath` in the link line.
- Algorithmically better performance than standard library calls
- No loss of accuracy
- single and double precision implementations of: `exp()`, `pow()`, and `log()`
- single precision implementations of: `sin()`, `cos()`

...more to come.

© 2018 Arm Limited

Compiler remarks

Help with vectorization, loop unrolling etc.

Optimization Remarks provide information about the choices made by the compiler on **in-lining**, **vectorization** and more.

Enabled by passing `-Rpass` command line options.

- Information about successful vectorization and inline optimization:
  `-Rpass=\(loop-vectorize\|inline\)`
- Information about what has been analyzed:
  `-Rpass-analysis=\(loop-vectorize\|inline\)`
- Information about where attempted in-lining and vectorisation has failed:
  `-Rpass-missed=\(loop-vectorize\|inline\)`
- *Note: Optimization remarks requires that an appropriate debug flag is set, such as* `-g`.

https://developer.arm.com/products/software-development-tools/hpc/documentation/using-optimization-remarks-with-arm-compiler

© 2018 Arm Limited

Common build problems

Autotools config:

- Out-of-date Autotools supplied with an application may get the triple wrong for Arm systems
- Obtaining up-to-date versions to fix:

```
wget 'http://git.savannah.gnu.org/gitweb/?p=config.git;a=blob_plain;f=config.guess;hb=HEAD' -O config.guess
wget 'http://git.savannah.gnu.org/gitweb/?p=config.git;a=blob_plain;f=config.sub;hb=HEAD' -O config.sub
```

Libtool:

- Libtool does not recognise ‘armflang’ or ‘flang’, and does not set the required linker flags
- Fixed in the libtool script distributed with many common packages, e.g. OpenMPI
- Otherwise, can be manually fixed post-configure:

```
sed -i -e 's#wl=""#wl="-Wl,"#g' libtool
sed -i -e 's#pic_flag=""#pic_flag=" -fPIC -DPIC"#g' libtool
```


© 2018 Arm Limited

Flang specific issues

things to watch out for...

Line lengths

- Line lengths of 264 characters, with up to 254 continuation lines, are accepted for free-form source
- In excess of the 132 in the standard, but less than accepted by ifort and gfortran
- Can be a particular problem for source using complicated macros
- *This issue should be fixed in the next release, permitting 1000 character lines*

Fortran 2003 array semantics

- Defaults is F95 - *no dynamic (re)allocation* on array assignments. Equivalent to gfortran's `-fno-realloc-lhs`
- Setting `-Mallocatable=03` will enable F2003 array semantics
- *This issue should be fixed in the next release, defaulting to F2003 semantics*

Guarded PGI-specific bug-fixes

- Some bugs to which flang is susceptible may have already been fixed and the fixes guarded with `#ifdef __PGI`.
- Manually set `__PGI` macro, or changing guards to `__FLANG`

 © 2018 Arm Limited

Other things to look out for

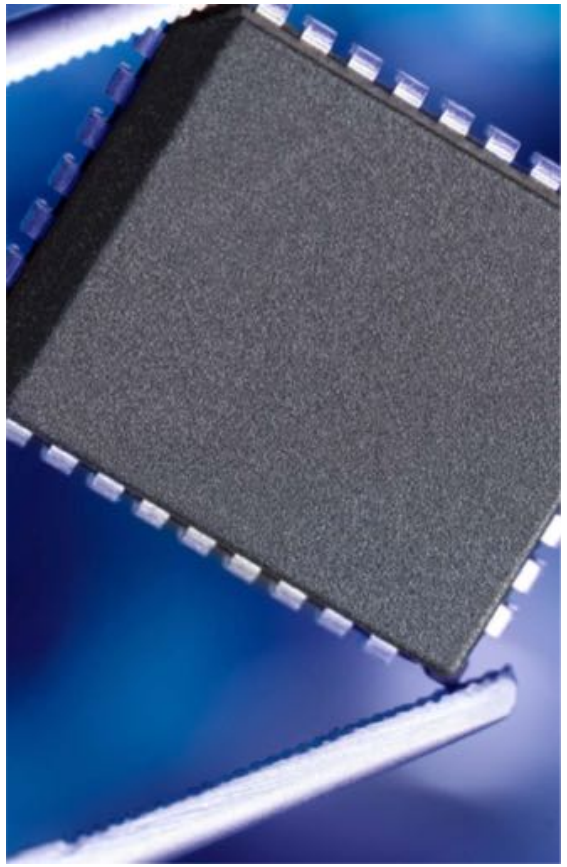
Integer divide-by-zero: In AArch64, integer divide-by-zero returns zero

Lazy evaluation: Beware using impure logical functions in IF statements

Weak memory model: reads and writes can be re-ordered, explicit memory barriers may be needed.
Not likely to be an issue for Fortran codes, but can be for some key libraries

Thread-safe recursion: `-frecursive` flag allocates all local variables on the stack.
Allows thread-safe recursion , applied implicitly for source compiled `-fopenmp`
Advisable for procedures called from within an OpenMP parallel region in source compiled `-fopenmp` flag

Automatic arrays: Automatic arrays are stored on the heap, regardless of the `-frecursive` flag, unless `-fstack-arrays` is specified.



 © 2018 Arm Limited

Arm HPC Ecosystem

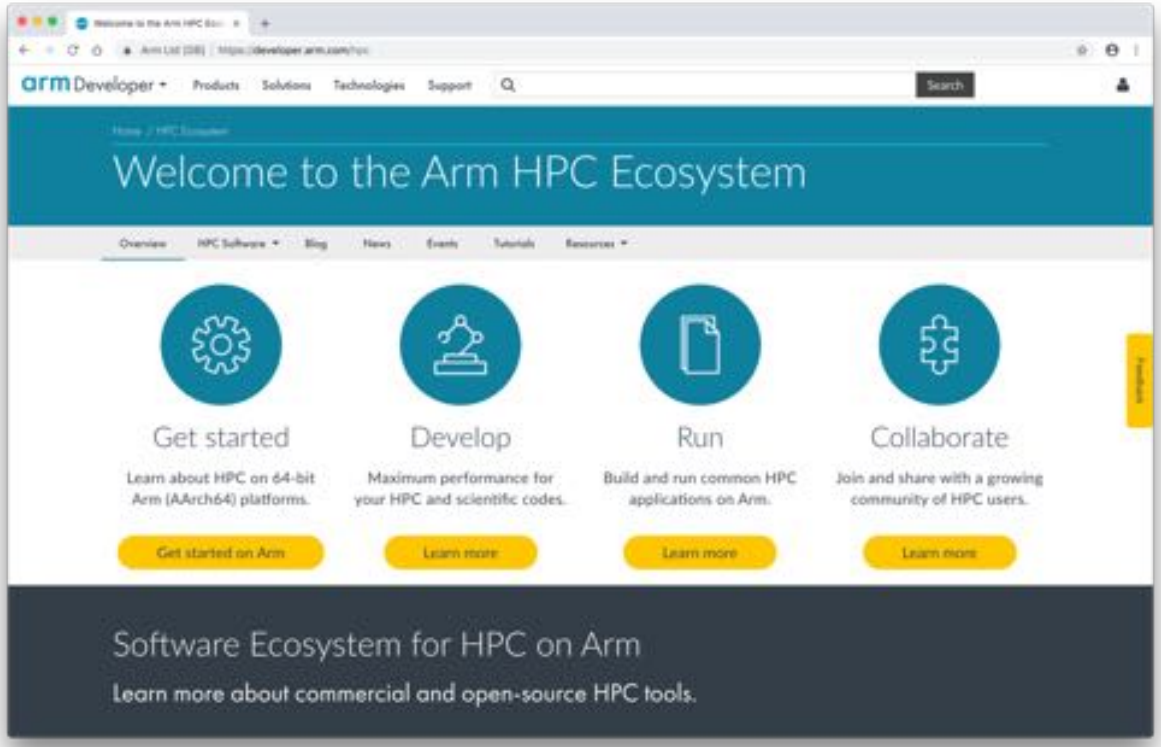
© 2018 Arm Limited

arm

Arm HPC Ecosystem website: **www.arm.com/hpc**

Starting point for developers and end-users of Arm for HPC

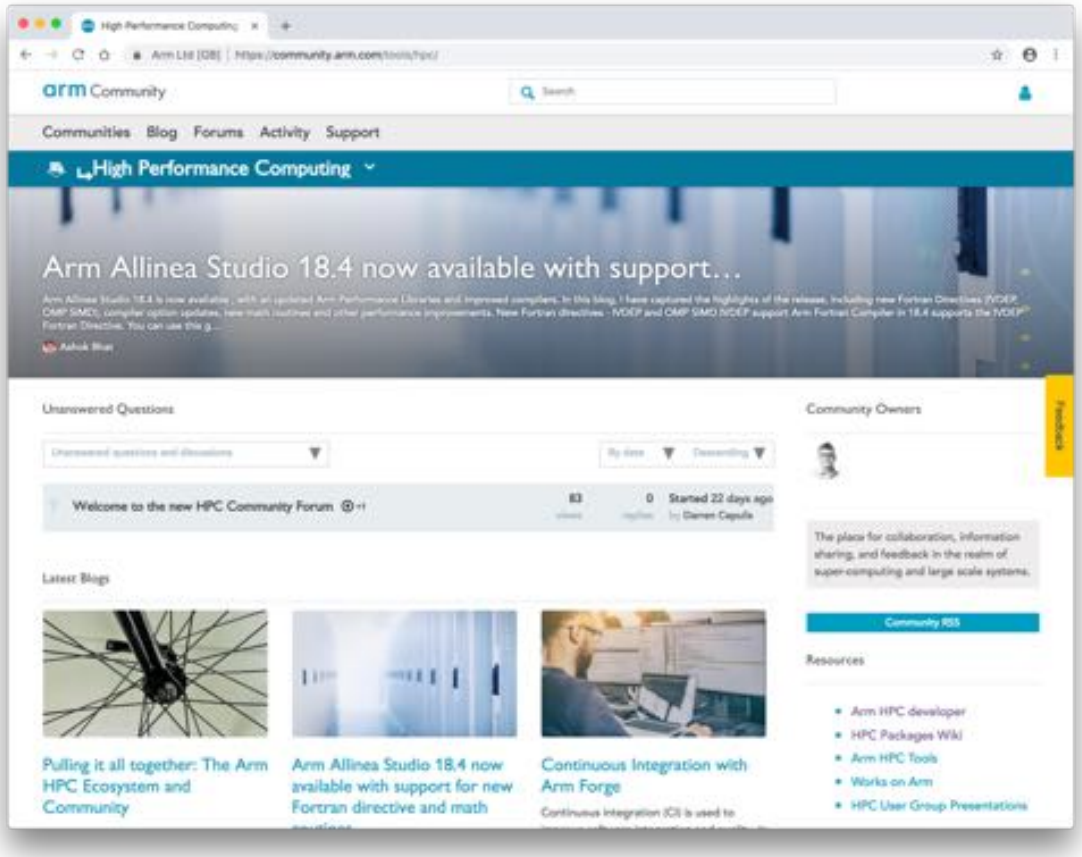
- Latest events, news, blogs, and collateral including whitepapers, webinars, and presentations
- Links to HPC open-source & commercial SW packages
- Guides for porting HPC applications
- Quick-start guides to the Arm Fortran Compiler for **ifort**, **gfortran** and **pgfortran** users
- Links to community collaboration sites
- Curated and moderated by Arm



© 2018 Arm Limited

Arm HPC Community: community.arm.com/tools/hpc/

HPC Community-driven Content



- **Blogs** by Arm and our HPC community
- **Calendar** of upcoming events such as workshops and webinars
- **HPC Forum** with questions & posts curated and moderated by Arm HPC technical specialists

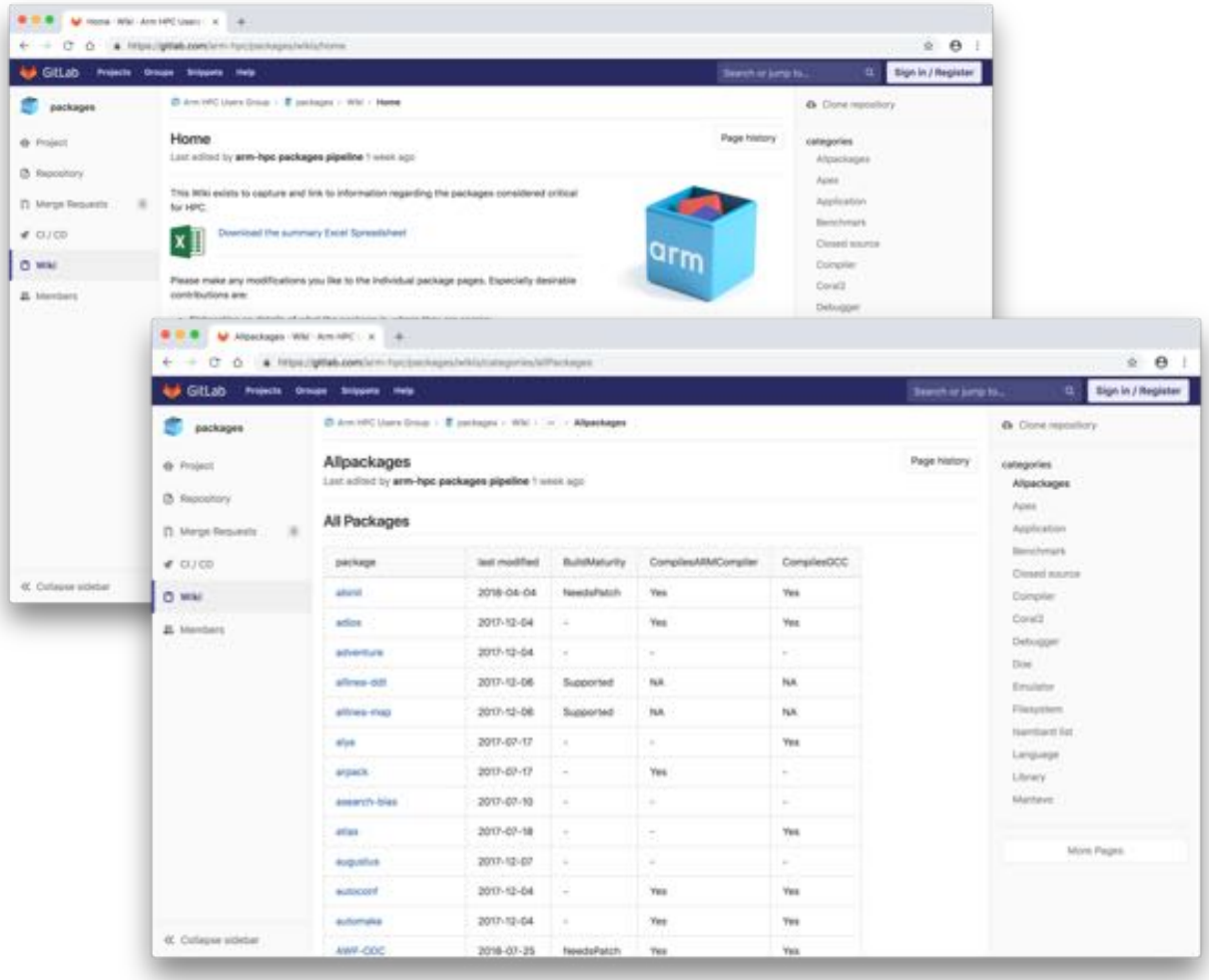
Ask, answer, share progress and expertise

© 2018 Arm Limited

Arm HPC Packages WIKI: gitlab.com/arm-hpc/packages/wikis

An essential element of the **Arm HPC ecosystem**

- Dynamic list of common HPC packages
- Status and porting recipes
- **Community** driven
- **Anyone can join** and contribute
- Provides **focus for porting** progress
- Allows developers to **share** and **learn**



© 2018 Arm Limited

Arm HPC deployments

© 2018 Arm Limited

arm

Deployments: Isambard at GW4

Cray XC50 series system

- Aries Interconnect

10,000+ Armv8.1a cores

- Cavium Thunder X2
- 2 x 32 cores @ > 2.0GHz

Cray Programming Environment

Platform for technology comparison

- x86, GPU, Armv8.1a

Arm components arriving soon

University of BRISTOL

Isambard

GW4

 © 2018 Arm Limited

arm

Deployments: Catalyst UK

HPE, in conjunction with Arm and SUSE, announced in April the "**Catalyst UK**" program: deployments to accelerate the growth of the Arm **HPC** ecosystem into three universities

Each machine will have:

- 64 HPE Apollo 70 systems:
  - Two 32-core Cavium ThunderX2 processors (i.e. 4096 cores per system)
  - 128GB of memory
  - Mellanox InfiniBand interconnect
- SUSE Linux Enterprise Server for HPC

Bristol: VASP, CASTEP, Gromacs, CP2K, Unified Model, NAMD, Oasis, NEMO, OpenIFS, CASINO, LAMMPS

EPCC: WRF, OpenFOAM, Two PhD candidates

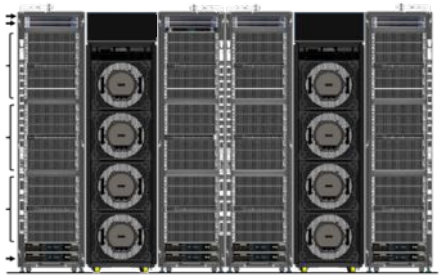
Leicester: Data-intensive apps, genomics, MOAB Torque, DiRAC collab

Department for Business, Energy & Industrial Strategy

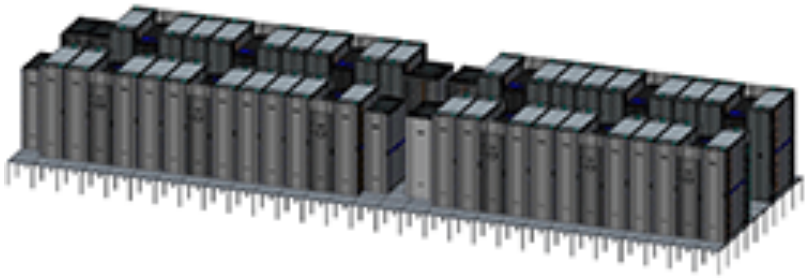
© 2018 Arm Limited

Deployments: Astra at Sandia

Mapping performance to real-world mission applications



- HPE to supply the United States National Nuclear Security Administration (NNSA)
- **2.3 petaflops** peak @1.2MW power (Top100)
- 64-bit Armv8-A Cavium ThunderX2 processors with 28-cores @ 2.0 Ghz
- 2 592 HPE Apollo 70 nodes in 9 SU
  - 145,152 cores in total
- Mellanox EDR InfiniBand



- 8 Memory Channels per socket
  - 332 TB aggregate memory capacity
  - 885 TB per second of aggregate memory bandwidth

© 2018 Arm Limited

Arm in High Performance Computing

AArch64, Arm's 64-bit instruction set, opens up the worlds of enterprise servers, data centres and High Performance Computing (HPC)

The emergence of innovative, infrastructure-ready Arm-based CPUs has heralded the arrival of the first large-scale Arm platforms

Fortran is at the heart of scientific HPC and delivering a credible Arm-based HPC solution requires a healthy developer ecosystem underpinned by both proprietary and opensource Fortran tools.

Arm Allinea Studio provides an end-to-end commercial suite for building and porting HPC applications on AArch64, including a Fortran compiler based on the community-driven projects LLVM and Flang.

© 2018 Arm Limited

arm

The Arm trademarks featured in this presentation are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. All other marks featured may be trademarks of their respective owners.

www.arm.com/company/policies/trademarks

 © 2017 Arm Limited

Useful links

Arm HPC ecosystem

https://www.arm.com/hpc

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

https://developer.arm.com/

Armflang orientation guides

https://developer.arm.com/products/software-development-tools/hpc/arm-allinea-studio/compiler-orientation-guides

Flang

https://github.com/flang-compiler/flang

https://github.com/flang-compiler/f18

© 2018 Arm Limited

Software Ecosystem – HPC Applications Porting

GROMACS	LAMMPS	CESM	MrBayes	Bowtie
NAMD	AMBER	Paraview	SIESTA	VMD
WRF	Quantum ESPRESSO	CP2k	MILC	GEANT4
OpenFOAM	GAMESS	VisIT	DL_POLY	NEMO
BLAST	NWCHEM	Abinit	BWA	QMCPACK

Build recipes online at https://gitlab.com/arm-hpc/packages/wikis/home

Chem/Phys	Weather	CFD	Visualization	Genomics