Scientific Computing on ARM
Part 1

ARM

Chris Adeniyi-Jones
Principal Engineer, Software and Large-Scale Systems
ARM Research

Linaro Connect - Budapest
Thursday 9th March 2017

Agenda

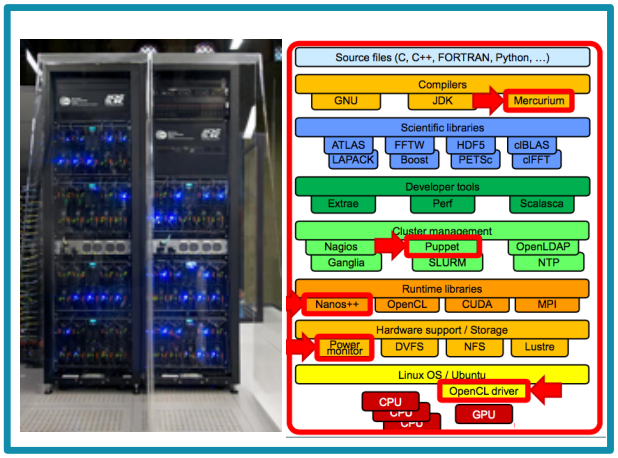
- How this got started …
- Ecosystem for HPC
  - Compilers
  - Libraries
  - Debuggers
  - Profilers
- Scalable Vector Extension
- Final Thoughts
- Hands-on Sessions

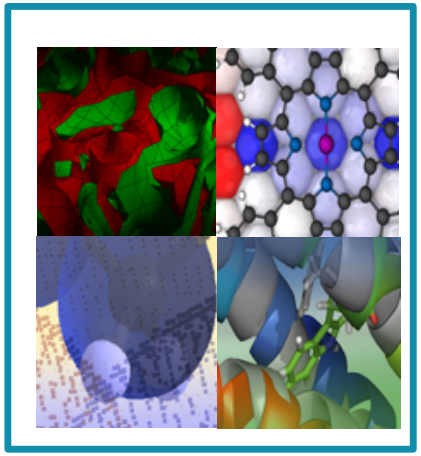
Small beginnings: Mont-Blanc Project - 2011

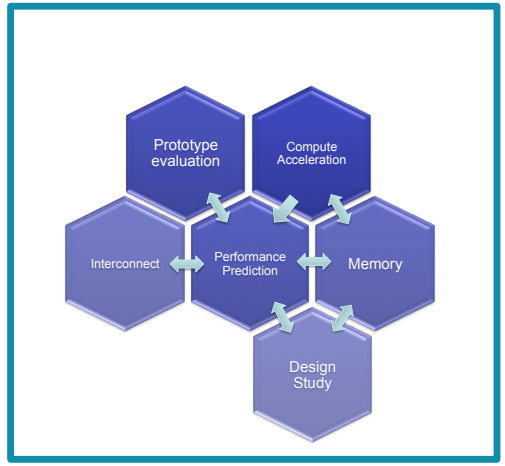
Develop an energy-efficient HPC prototype using low-power commodity embedded technology.

Port and optimize HPC applications on the prototype

Research into technologies required for the next-generation HPC system

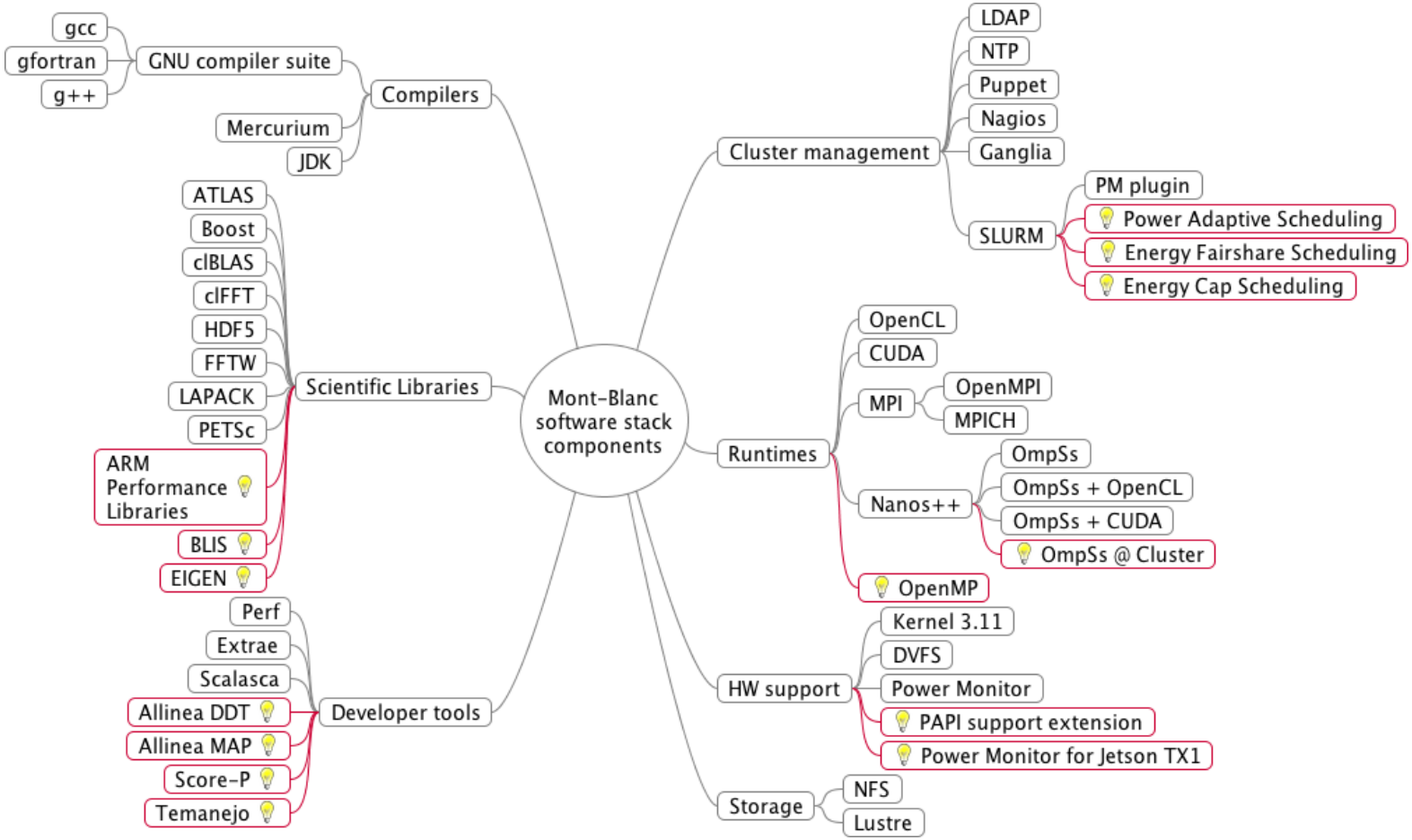






ARM

Mont-Blanc HPC Software Ecosystem



ARM

Why "Commodity"?

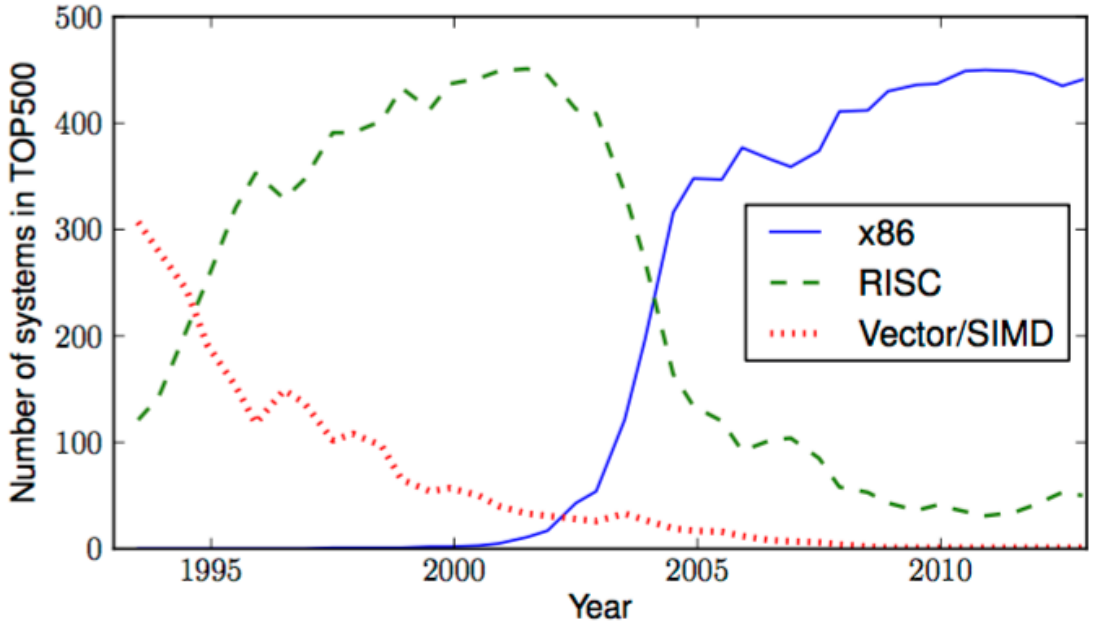


Figure 1: TOP500: Special-purpose HPC replaced by RISC microprocessors, in turn displaced by x86

Cost
Availability
Performance

Energy efficiency?

Supercomputing with Commodity CPUs: Are Mobile SoCs Ready for HPC?
Nikola Rajovic, Paul M. Carpenter, Isaac Gelado, Nikola Puzovic, Alex Ramirez, Mateo Valero

Serious ARM HPC deployments starting in 2017

- Two big announcements in 2017 about ARM in HPC in Europe

Bull Atos to Build for HPC Prototype for Mont-Blanc Project using Cavium ThunderX2 Processor

January 16, 2017 by staff Leave a Comment

Today the Mont-Blanc European project announced it has selected Cavium's ThunderX2 ARM server processor to power its new HPC prototype.

The new Mont-Blanc prototype will be built by Atos, the coordinator of phase 3 of Mont-Blanc, using its Bull expertise and products. The platform will leverage the infrastructure of the Bull sequana pre-exascale supercomputer range for network, management, cooling, and power. Atos and Cavium signed an agreement to collaborate to develop this new platform, thus making Mont-Blanc an Alpha-site for ThunderX2.

Bull atos technologies

inside HPC

"As the race to Exascale intensifies, we are pleased to be the vendor of choice partner with Atos to deliver Mont-Blanc platform" said Rishi Chugh, Director of Marketing, Data Center Processor Group at Cavium. "ThunderX2 builds on established architecture and ecosystem of ThunderX delivering performance competitive with next generation of incumbent processors".

University of BRISTOL GW4 January 17th 2017

Announcing the **GW4 Tier 2 HPC service, 'Isambard'**: named after Isambard Kingdom Brunel

System specs:

- Cray CS-400 system
- **10,000+** ARMv8 cores
- HPC optimised software stack
- Technology comparison:
  - x86, KNL, Pascal
- To be installed March-Dec 2017
- £4.7m total project cost over 3 years

I.K.Brunel 1804-1859

Simon McIntosh Smith, simonm@cs.bris.ac.uk, @simonmcs 5 bristol.ac.uk

Fujitsu HPC CPU

Post-K: Fujitsu HPC CPU to Support ARM v8

ARM FUJITSU

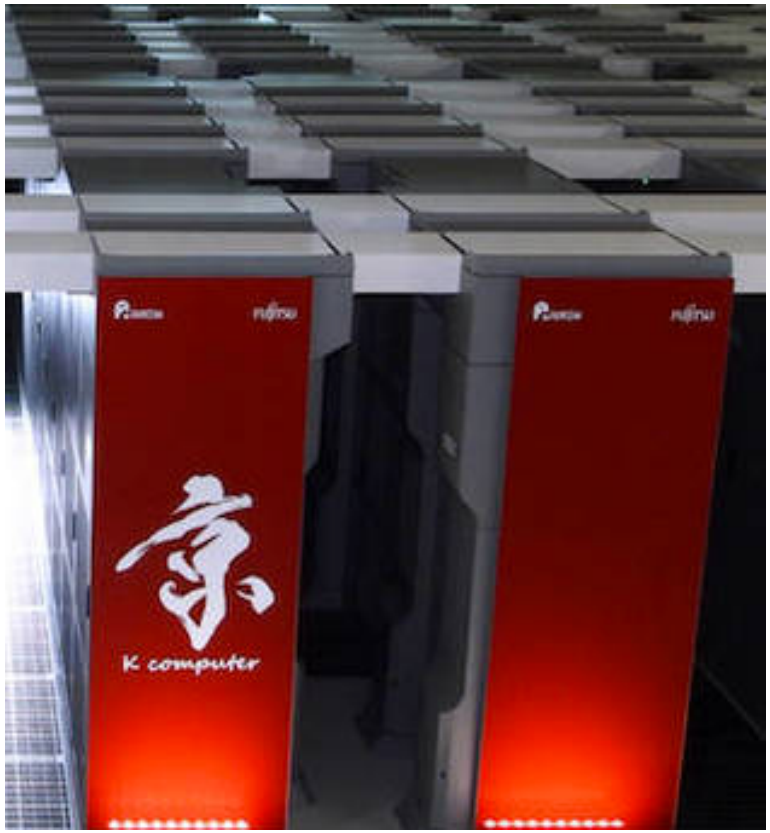
Post-K fully utilizes Fujitsu proven supercomputer microarchitecture

Fujitsu, as a lead partner of ARM HPC extension development, is working to realize ARM Powered® supercomputer w/ high application performance

ARM v8 brings out the real strength of Fujitsu's microarchitecture

HPC apps acceleration feature	Post-K	FX100	FX10	K computer
FMA: Floating Multiply and Add	✔	✔	✔	✔
Math. acceleration primitives*	✔Enhanced	✔	✔	✔
Inter core barrier	✔	✔	✔	✔
Sector cache	✔Enhanced	✔	✔	✔
Hardware prefetch assist	✔Enhanced	✔	✔	✔
Tofu interconnect	✔Integrated	✔Integrated	✔	✔

* Mathematical acceleration primitives include trigonometric functions, sine & cosines, and exponential...



slides from Fujitsu at ISC'16

Ecosystem for HPC

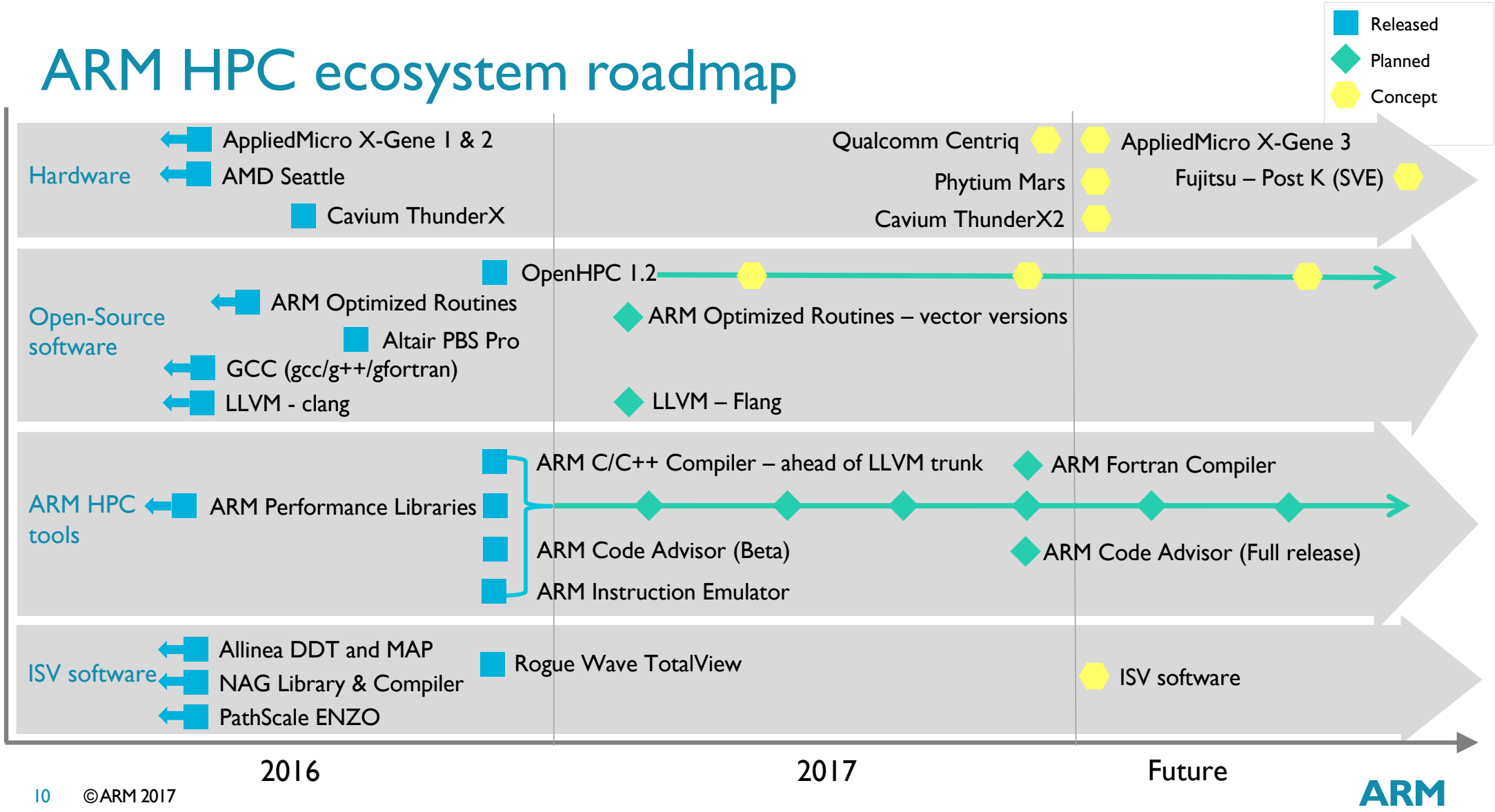
Ecosystem for HPC

List of software components needed:

- Linux OS availability
- Compilers
- Libraries
- Debuggers
- Profilers
- Job schedulers

Mix of open source and commercial products and applications…

ARM HPC ecosystem roadmap



ARM

Open source in the ARM HPC ecosystem

- Over the past 12 months many more packages and applications have been successfully ported to ARM HPC

Linux / FreeBSD w/ AARCH64 support

Debian 8 adds AARCH64 – April 2015

12.04LTS & 14.04LTS ← Also 14.10 & 15.04 releases released

Fedora 22 released – May 2015
Fedora 23 released – Nov 2015

Red Hat Enterprise Linux Server for ARM
7.2 BETA – Sept, 2015

CentOS Linux 7 for AArch64
GA – August 2015

OpenSUSE 13.2 – Nov 2014

SUSE Launches Partner Program to Bring SUSE Linux Enterprise 12 to 64-bit ARM
July 2015 @ ISC

← Engaged with FreeBSD foundation / Semi-half & Cavium to get FreeBSD on ARMv8
FreeBSD Beta version demo'd by Semihalf – Nov. 2015

HPC filesystems

Software	Status
LUSTRE	Ported
HDFS	Ported
CEPH	Ported
BeeGFS	Ported

Workload and cluster managers

Software	Status
IBM LSF	Ported
HP CMU	Ported
SLURM	Ported
Adaptive Computing (Moab)	Future
Altair PBS Works	Ported
OpenLava (LSF port)	Ported

Compilers

Open source and commercial compilers

- GCC
  - C, C++, Fortran
  - OpenMP 4.0

- PathScale
  - C, C++ Fortran
  - OpenACC
  - OpenMP 4.0

- LLVM
  - C, C++
  - OpenMP 3.1, (4.0 coming soon)
  - Fortran coming Q1 2017

- NAG
  - Fortran
  - OpenMP 3.1

- ARM C/C++ Compiler
  - LLVM based
  - Includes SVE

ARM C/C++ Compiler

Commercially supported C/C++ compiler for Linux user-space HPC applications

LLVM-based

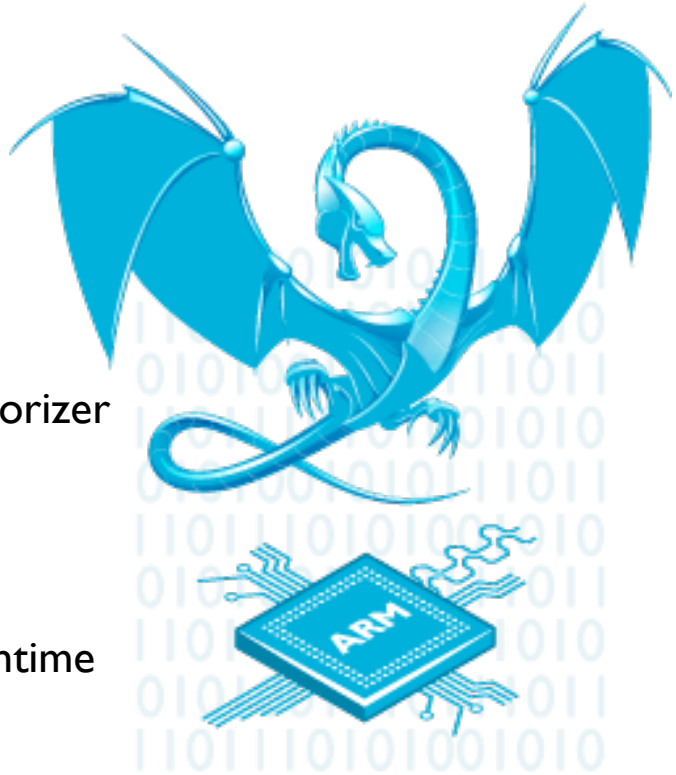
- ARM-on-ARM compiler
- For application development (not bare-metal/embedded)

Regularly pulls from upstream LLVM, adding:

- SVE support in the assembler, disassembler, intrinsics and autovectorizer
- Compiler Insights to support ARM Code Advisor

OpenMP

- Uses latest open source (now ARM-optimized) LLVM OpenMP runtime
- Changes pushed back to the community



ARM C/C++ Compiler – usage

- To compile C code:

```
% armclang -O3 file.c -o file
```

- To compile C++ code:

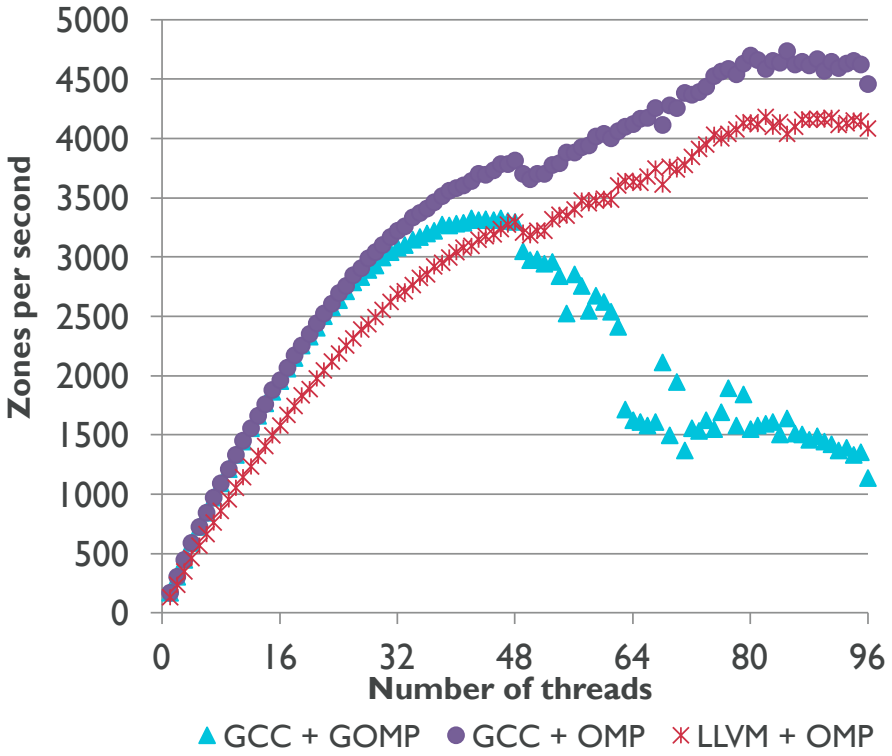
```
% armclang++ -O3 file.cpp -o file
```

Common `armclang` options

Flag	Description
`--help`	Describes the most common options supported by ARM C/C++ Compiler
`--vsn` `--version`	Displays version information and license details
`-O<level>`	Specifies the level of optimization to use when compiling source files. The default is `-O0`
`-c`	Performs the compilation step, but does not perform the link step. Produces an ELF object `.o` file. Run **`armclang`** again, passing in the object files to link
`-o <file>`	Specifies the name of the output file
`-fopenmp`	Use OpenMP
`-S`	Outputs assembly code, rather than object code. Produces a text `.s` file containing annotated assembly code

LLVM OpenMP development

- We have been contributing ARM-related upstream patches to LLVM's 'libomp'
- Example shown here is the Lulesh benchmark at size=80 running on 1-96 cores
- GCC has slightly better serial performance
- libomp demonstrates superior scaling
  - Even when used with GCC!



ARM

Parallelism to enable optimal HPC performance

- OpenMP
  - We are adding enhancements to the LLVM OpenMP implementation to get better AArch64 performance
  - ARM is active member of the OpenMP Standards Committee
- OpenACC
  - PathScale and PGI are strong supporters of OpenACC
    - supported for ARM within ENZO
- Auto-vectorization
  - ARM actively works on vectorization in GCC and LLVM, and encourages work with vectorization support in the compiler community.
  - PathScale's compiler has vectorization support built in
- MPI parallelism "just works"
  - Better Infiniband driver support is coming from Mellanox

Libraries

openHPC – now on ARM

OpenHPC is a community effort to provide a common, verified set of open source packages for HPC deployments

ARM's participation:

- Silver member of OpenHPC
- ARM is on the OpenHPC Technical Steering Committee in order to drive ARM build support

Status: 1.2.0 release out now

- All packages built on ARMv8 for CentOS and SUSE
- ARM-based machines are being used for building and also in the OpenHPC build infrastructure

Functional Areas	Components include
Base OS	RHEL/CentOS 7.1, SLES 12
Administrative Tools	Conman, Ganglia, Lmod, LosF, ORCM, Nagios, pdsh, prun
Provisioning	Warewulf
Resource Mgmt.	SLURM, Munge. Altair PBS Pro*
I/O Services	Lustre client (community version)
Numerical/Scientific Libraries	Boost, GSL, FFTW, Metis, PETSc, Trilinos, Hypre, SuperLU, Mumps
I/O Libraries	HDF5 (pHDF5), NetCDF (including C++ and Fortran interfaces), Adios
Compiler Families	GNU (gcc, g++, gfortran)
MPI Families	OpenMPI, MVAPICH2
Development Tools	Autotools (autoconf, automake, libtool), Valgrind,R, SciPy/NumPy
Performance Tools	PAPI, Intel IMB, mpiP, pdtoolkit TAU

Open source library AArch64 inbuilt tuning work

ARM actively working with the community for increased support and performance

- OpenBLAS
  - ARMv8 kernels included
- BLIS
  - BLIS developers have close relationship with ARM
  - BLIS supports various ARM processors by default (e.g. ARM Cortex-A53, Cortex-A57 CPUs)
  - Also currently conducting ARM big.LITTLE development
- ATLAS
  - Work ongoing with ARM Research team
  - Cortex-A57/A53 patches went into ATLAS
- FFTW
  - Just works. NEON options built into v3.3.5

Commercial library support

Product availability for 64-bit ARMv8-A

- ARM Performance Libraries
  - See following slides
- NAG Library
  - Largest commercially available collection of numerical and statistical algorithms
    - > 1800 functions
  - *Scales well, takes advantage of ARM Performance Libraries*
  - Tested on Juno (ARM) and ThunderX (Cavium)
- PathScale BLAS implementation
  - Comes with their ENZO compiler

ARM Performance Libraries

Commercial 64-bit ARMv8 math libraries

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

Best-in-class performance with commercial support

- Tuned by ARM for Cortex-A72, Cortex-A57 and Cortex-A53
- Maintained and Supported by ARM for a wide range of ARM-based SoCs
- Regular benchmarking against open source alternatives

Silicon partners can provide tuned micro-kernels for their SoCs

- Partners can collaborate directly working with our source-code and test suite
- Alternatively they can contribute through open source route

Performance on par
with best-in-class math libraries

Commercially Supported
by ARM

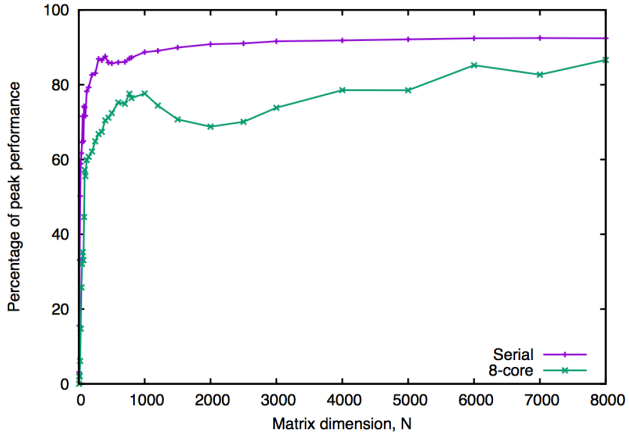
Validated with
NAG test suite

ARM Performance Libraries

Version 2.0: Improving performance and interoperability

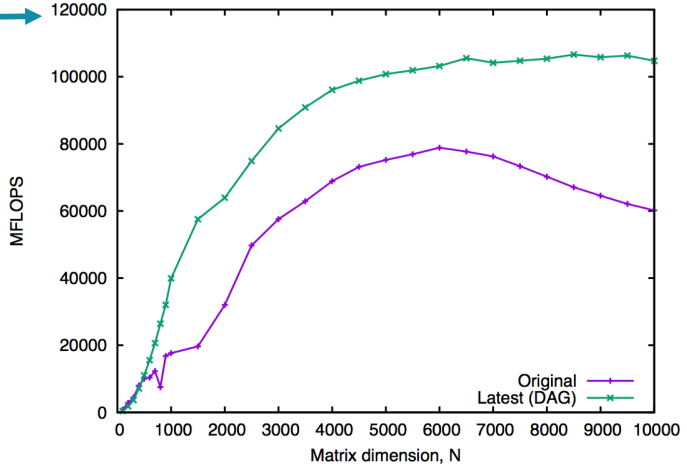
BLAS

- Enhanced parallelism for BLAS level 1
- Hand-tuned kernels for BLAS level 3



LAPACK (3.6.1)

- Added PLASMA-style Directed Acyclic Graphs for better parallelism



FFTW Interface

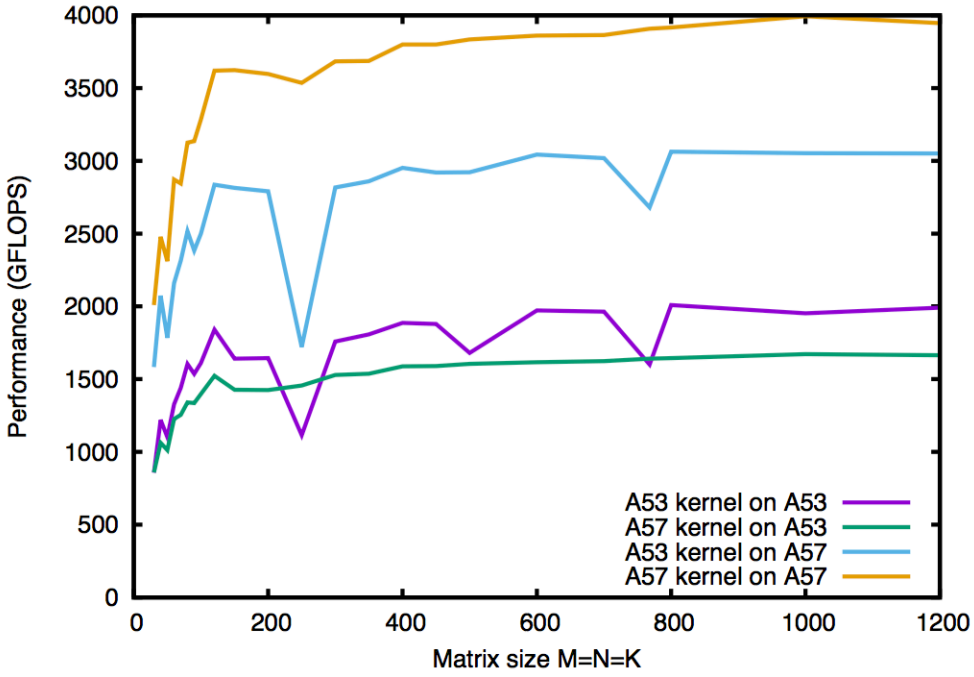
- Support for FFTW-compatible basic and advanced DFT interfaces

Scalable Vector Extensions

- Libraries built with SVE-capable compilers
- Hand-written DGEMM and SGEMM kernels

ARM Performance Libraries – micro-architecture

- ARM cores have a variety of designs, created by both ARM and our partners
- ARM Performance Libraries are creating tailored versions of routines to target these different *micro-architectures*
- It is important to ensure that the correct version is installed on your system
- For example consider the different performance in DGEMM running the Cortex-A53 and Cortex-A57 kernels on the right and wrong cores…



ARM Performance Libraries – linking

- In order to compile applications using BLAS, LAPACK and FFT routines from the ARM Performance libraries, four options are provided:
  - Serial and OpenMP builds
  - 32-bit and 64-bit integers
- These translate into four binaries in `/opt/arm/armpl-*/lib/`
  - `libarmpl.a`
  - `libarmpl_mp.a`
  - `libarmpl_int64.a`
  - `libarmpl_int64_mp.a`
- Shared libraries (`libarmpl*.so`) are also provided
- Compile and link using, for example

```
armclang -O3 file.c -fopenmp -c file.o -I${ARMPL_DIR}/include
armclang -O3 file.o -fopenmp -o file -L${ARMPL_DIR}/lib -larmpl_mp
```

Scientific Computing on ARM Part 2

ARM

Chris Adeniyi-Jones
Principal Engineer, Software and Large-Scale Systems
ARM Research

Linaro Connect - Budapest
Thursday 9th March 2017

Debuggers

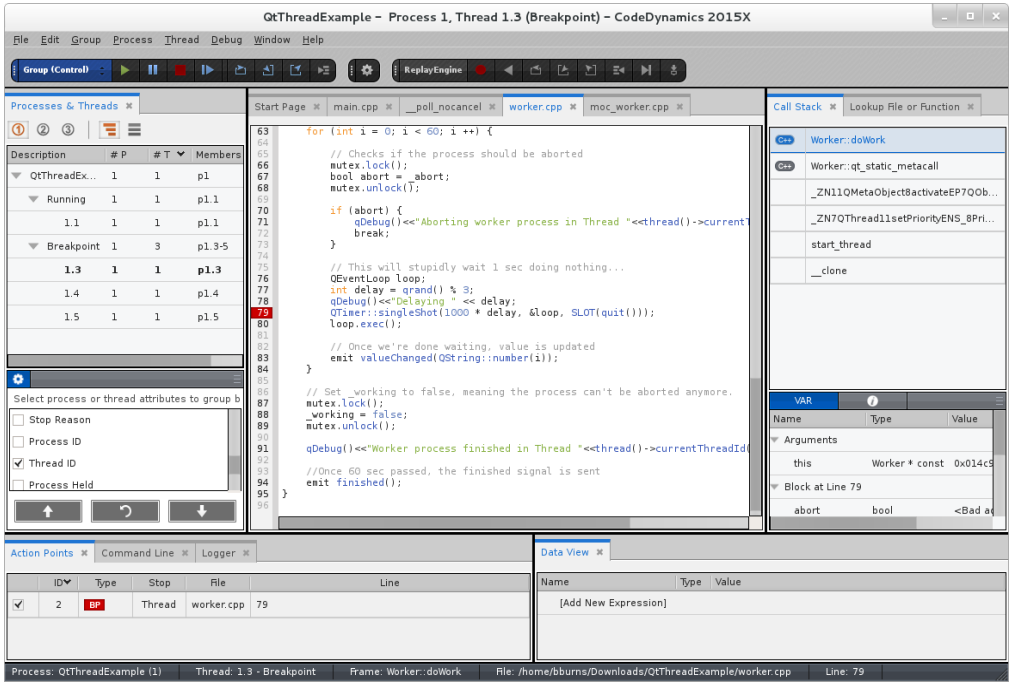
Open source debugging tools

- All the usual open source tools you would use for debugging work as expected on ARMv8
  - printf()
  - gdb
  - valgrind (and its associated tools)

- It is worth ensuring that the versions on your system are recent
  - Check Linaro webpages for up to date source

TotalView for HPC

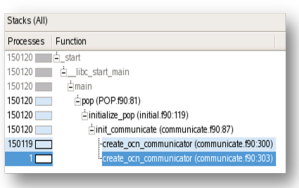
- Common HPC debug environment
  - Active development for 30+ years
  - Thread specific breakpoints
  - Control individual thread execution
  - View thread specific stack and data
  - View complex data types easily
- Track memory leaks in running applications
- Supports C/C++ on Linux
- Allowing the business to have
  - Predictable development schedules
  - Less time spent debugging
- ARM support
  - Beta just concluded
  - Early access available November 2016 – version 2016.07
  - Full release planned for Feb 2017 – version 2017.1



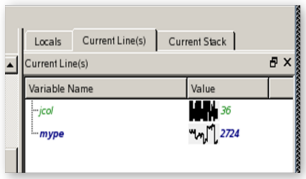
Allinea DDT

The debugger for C, C++ and Fortran threaded and parallel code

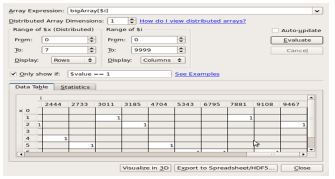
- ***Who* had a rogue behaviour ?**
  - Merges stacks from processes and threads
- ***Where* did it happen?**
  - Allinea DDT leaps to source automatically
- ***How* did it happen?**
  - Detailed error message given to the user
  - Some faults evident instantly from source
- ***Why* did it happen?**
  - Unique "Smart Highlighting"
  - Sparklines comparing data across processes



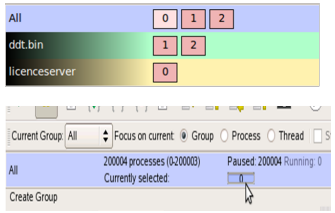
Parallel stack view



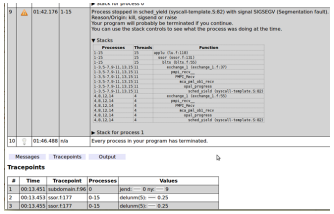
Automated data comparison: sparklines



Parallel array searching



Step, play, and breakpoints



Offline debugging

ARM debugging gotcha

Weakly ordered memory model

- Weakly ordered memory access means that changes to memory can be applied in any order as long as *single-core* execution sees the data needed for program correctness
- Most parallel HPC codes we encountered used GCC's libgomp (-fopenmp)
  - These behaved correctly on AArch64 using GCC 5.2
- Some HPC codes we have come across have their own bespoke parallelization
  - Usually based directly on top of pthreads
  - Written to have more control over the threads of execution and how they synchronize
  - Problems are almost always down to a lock-free thread interaction implementation

Thread 0

```
A=1;
set=2;
```

Thread 1

```
if (set==2) C = A;
```

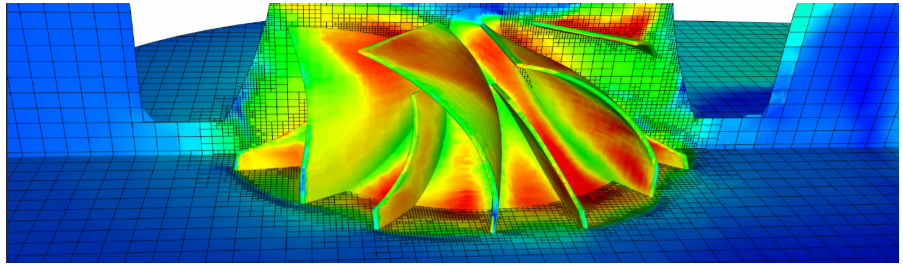
Profilers

Performance hints

- Use latest compiler version, not OS default
- Use OMP_PROC_BIND=TRUE
  - Make sure you use OMP_PLACES if doing more than one run

Profiling – always important!

- **Challenges**
  - One (non-ARM) customer came to Allinea with help getting their code working better
- **Huge speed up on CONVERGE: from 2h down to 4 sec**
  - Now possible to run jobs efficiently on hundreds of cores
  - Now possible to scale up from 2 million to 20 million nodes



Full case study:

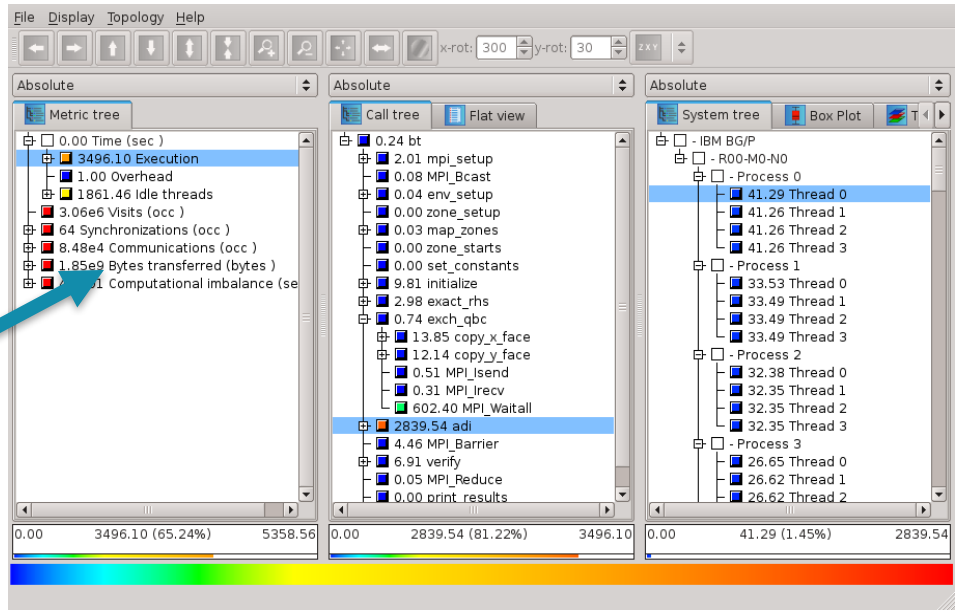
http://www.allinea.com/news/201509/convergent-science-ignites-combustion-simulation-performance-allinea-forge

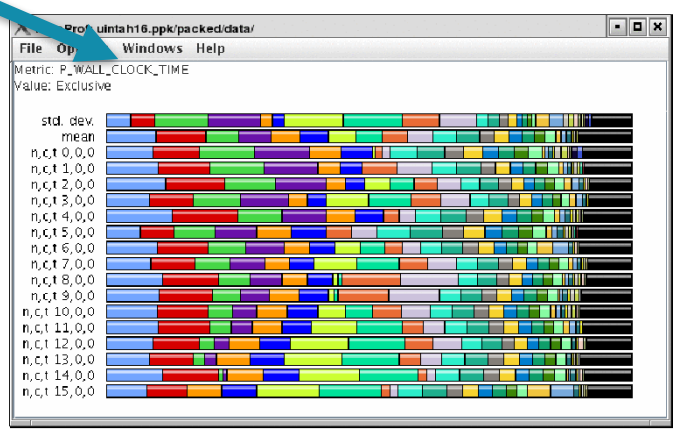
Profiling/debugging tools

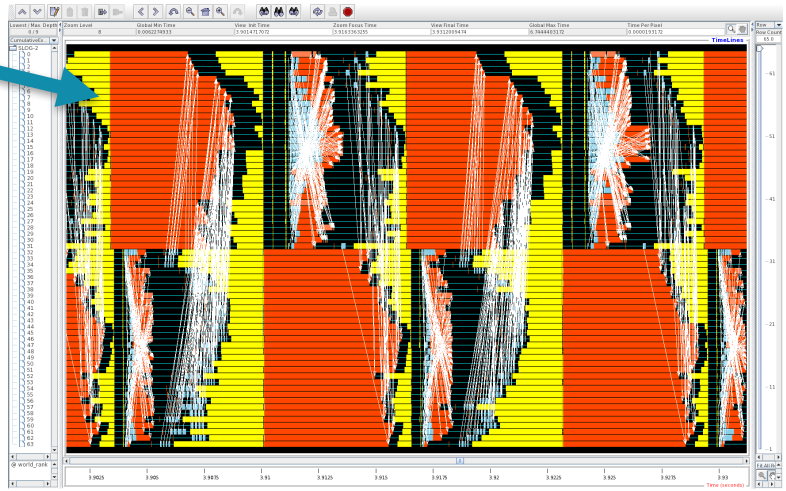
Open source tools ported to ARM HPC and "just work"

Tested applications include:

- PAPI and Perf counters
- Score-P, Cube, Scalasca
- MPE and Jumpshot
- TAU





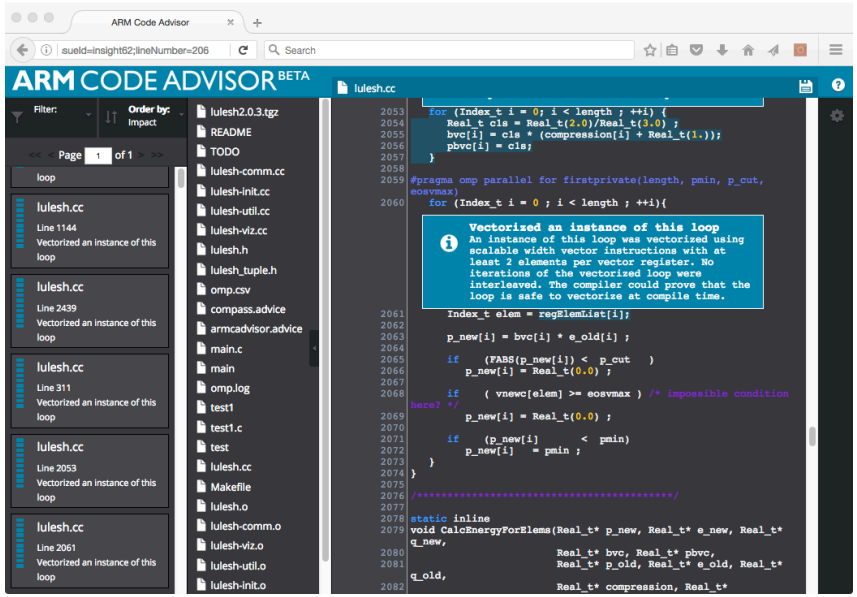


ARM Code Advisor (Beta)

Combines static and dynamic information to produce actionable insights

Performance Advice

- Compiler vectorization hints
- Compilation flags advice
- Fortran subarray warnings
- OpenMP instrumentation



Insights from compilation and runtime

- Compiler Insights are embedded into the application binary by the ARM Compilers
- OMPT interface used to instrument OpenMP runtime

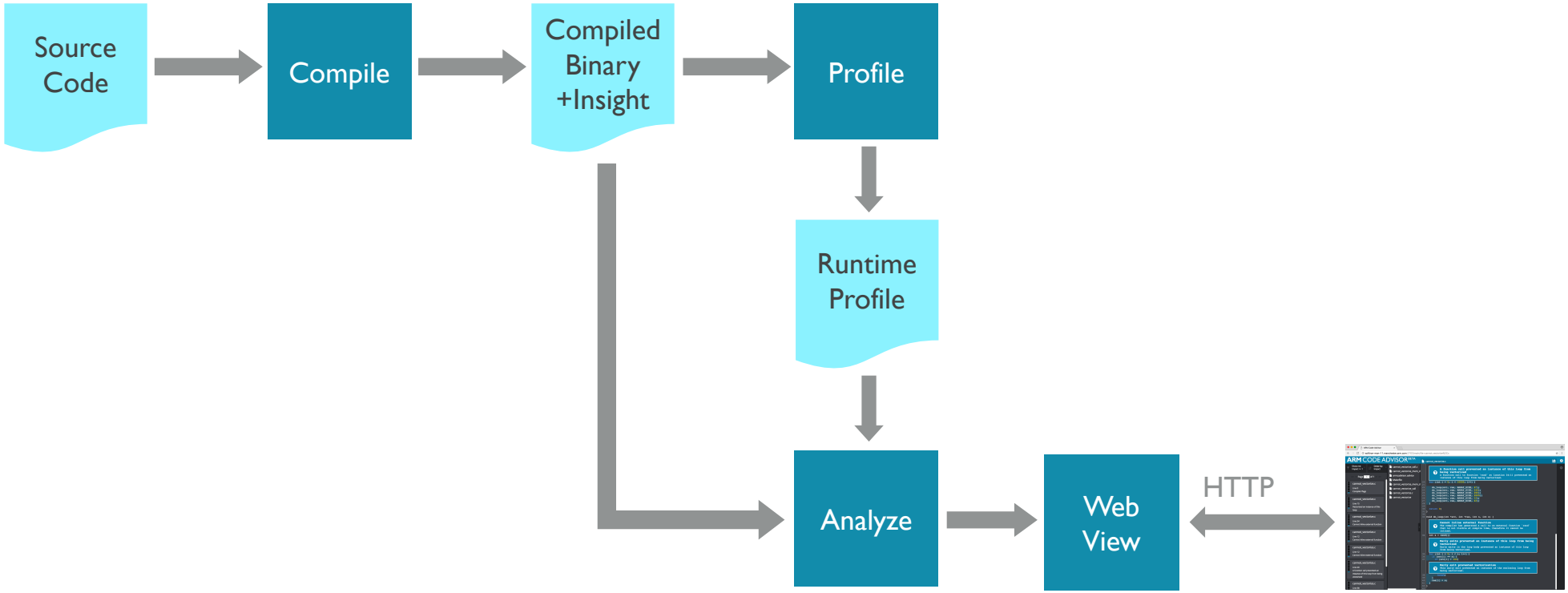
Extensible Architecture

- Users can write plugins to add their own analysis information
- Data accessible via web-browser, command-line, and REST API to support new user interfaces

ARM

ARM Code Advisor (Beta)

Typical workflow



ARM Code Advisor – usage

- Compile application with Insight functionality:

```
% armclang++ -O3 -insight file.cpp -o file
```

- Run code:

```
% armcadvisor collect ./example
Starting collection of program [./example] to
profile temp in /home/user1/armcadvisor-profiles
```

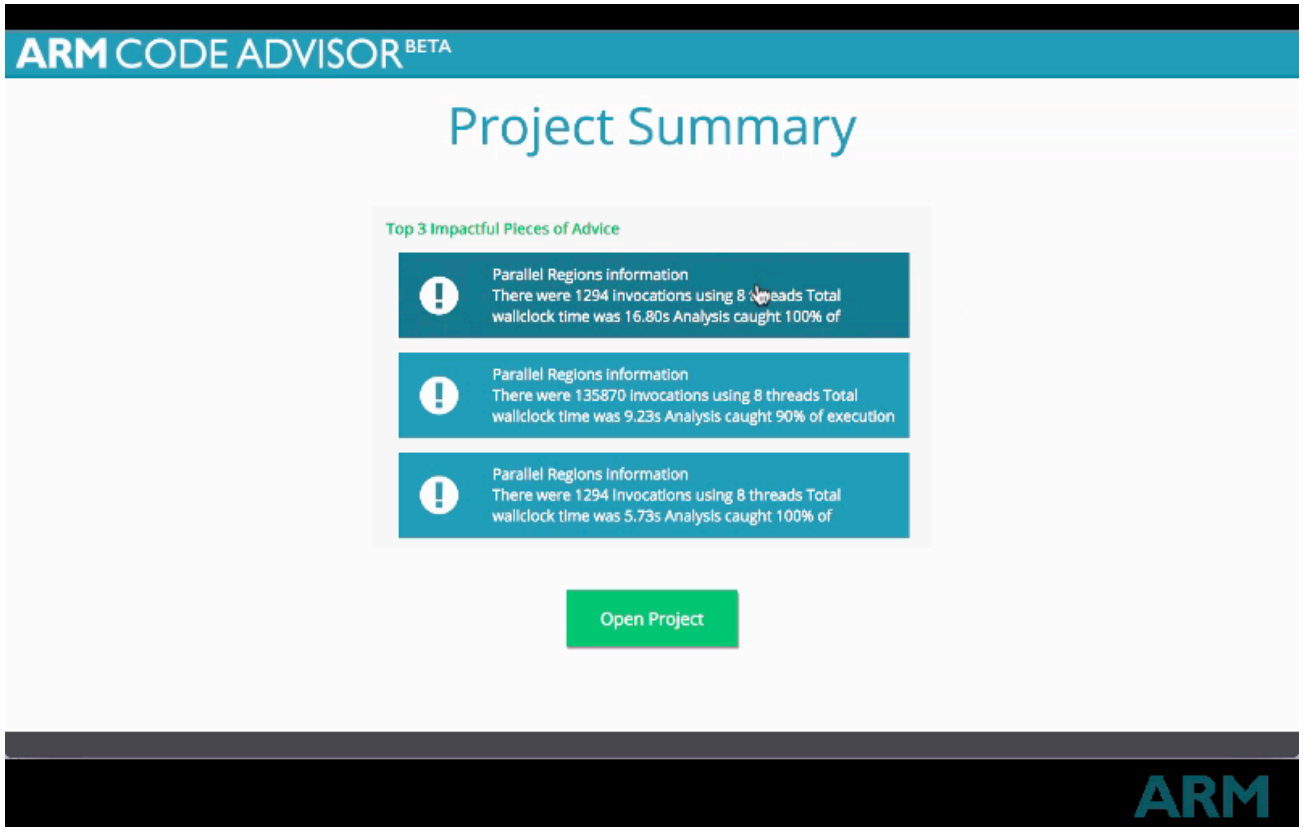
- Analyze collected data:

```
% armcadvisor analyze example
Generating analysis file, armcadvisor.advice
```

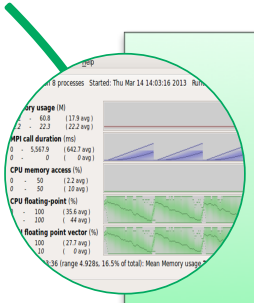
- View analysis:

```
% armcadvisor web --network --everyone -p 2010
Open your browser to one of:
   http://server.arm.com:8080
   http://127.0.0.1:8080
```

ARM Code Advisor – video

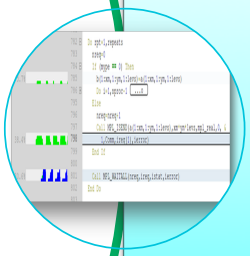


ARM Allinea MAP



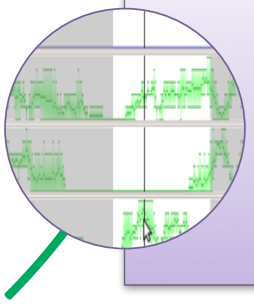
Low overhead measurement

- Accurate, non-intrusive application performance profiling
- Seamless – no recompilation or relinking required



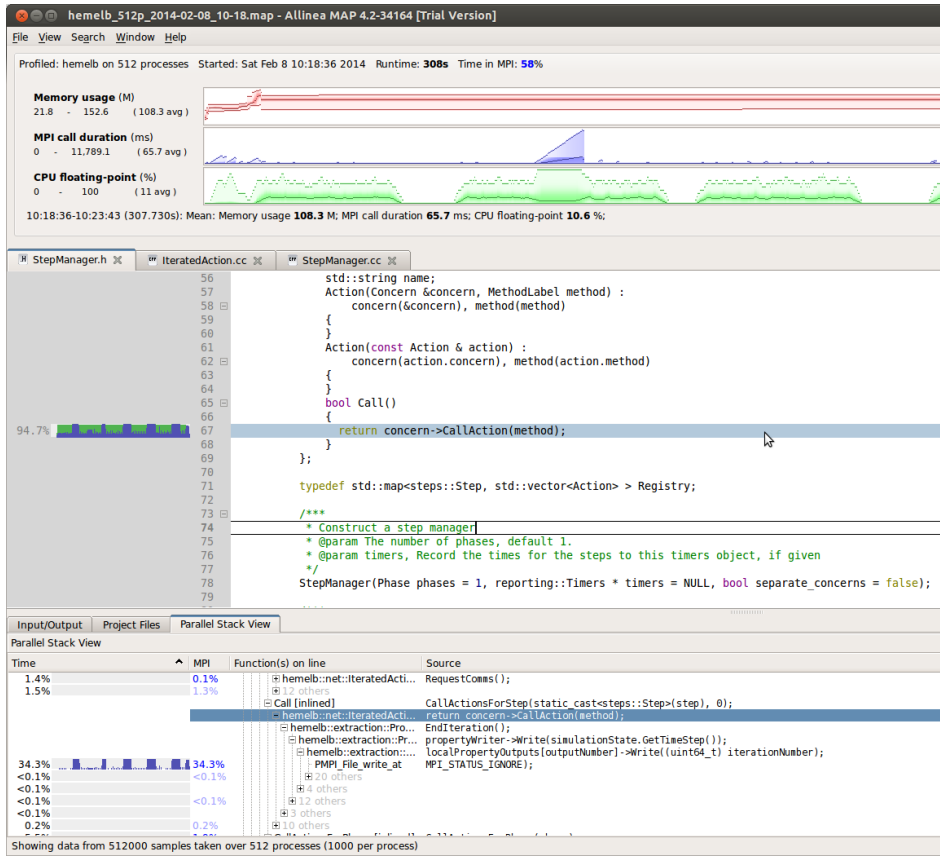
Easy to use

- Source code viewer pinpoints bottleneck locations
- Zoom in to explore iterations, functions and loops

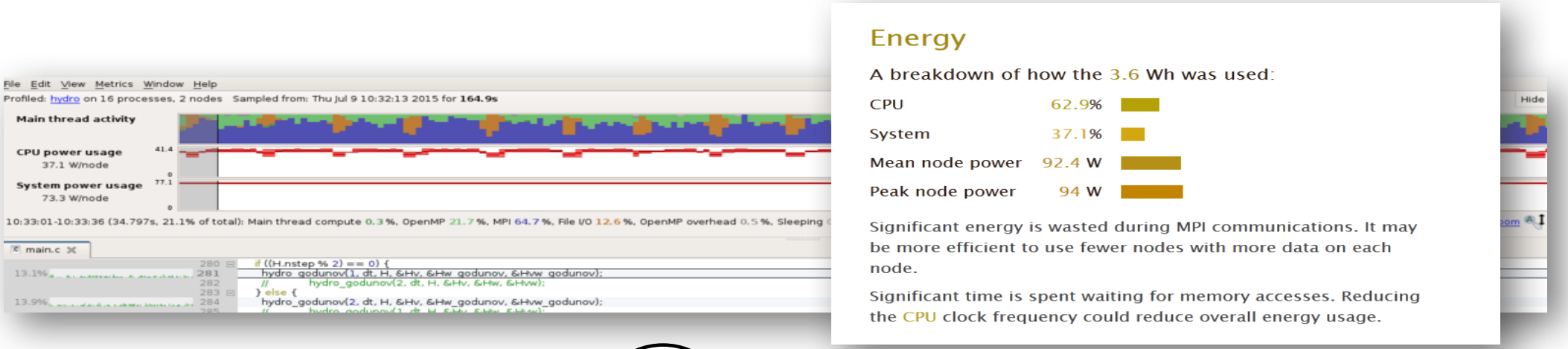


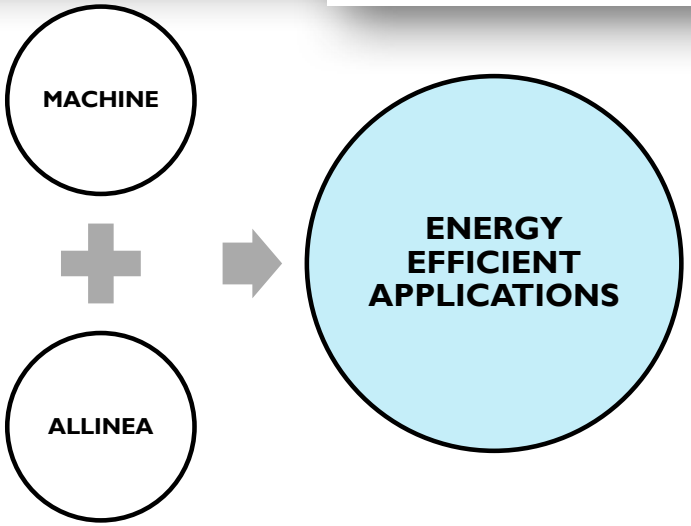
Deep

- Measures CPU, communication, I/O and memory to identify problem causes
- Identifies vectorization and cache performance



Energy efficiency with ARM's Allinea tools

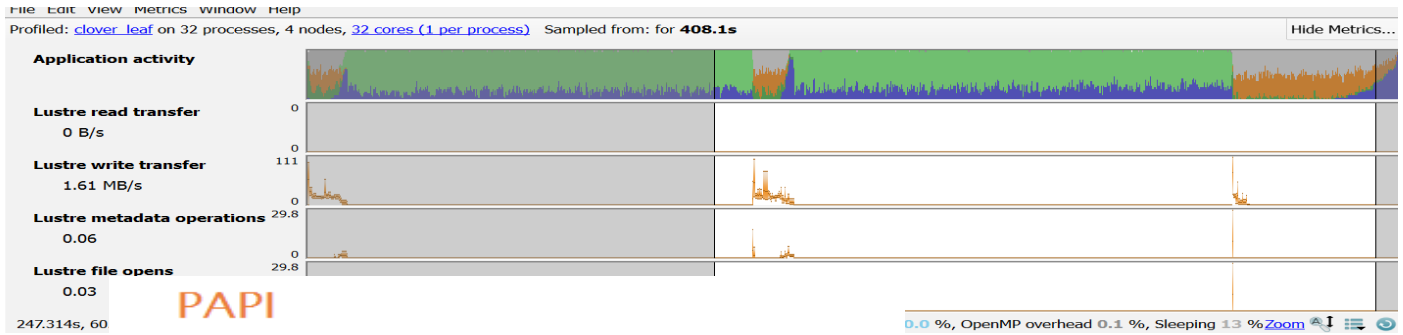




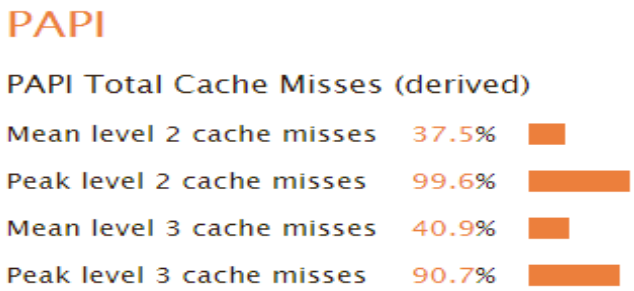
ARM

Allinea Forge: what's new in 7.0 – examples

Custom metrics in MAP

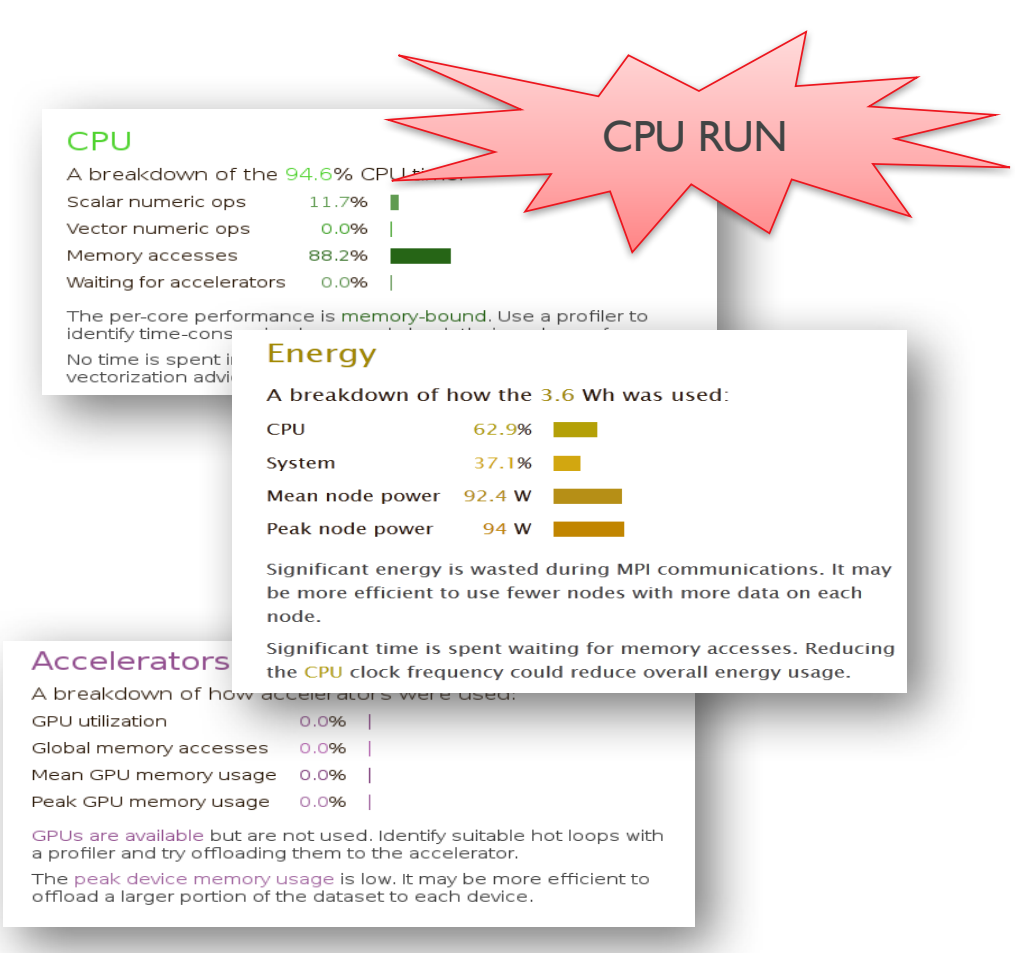


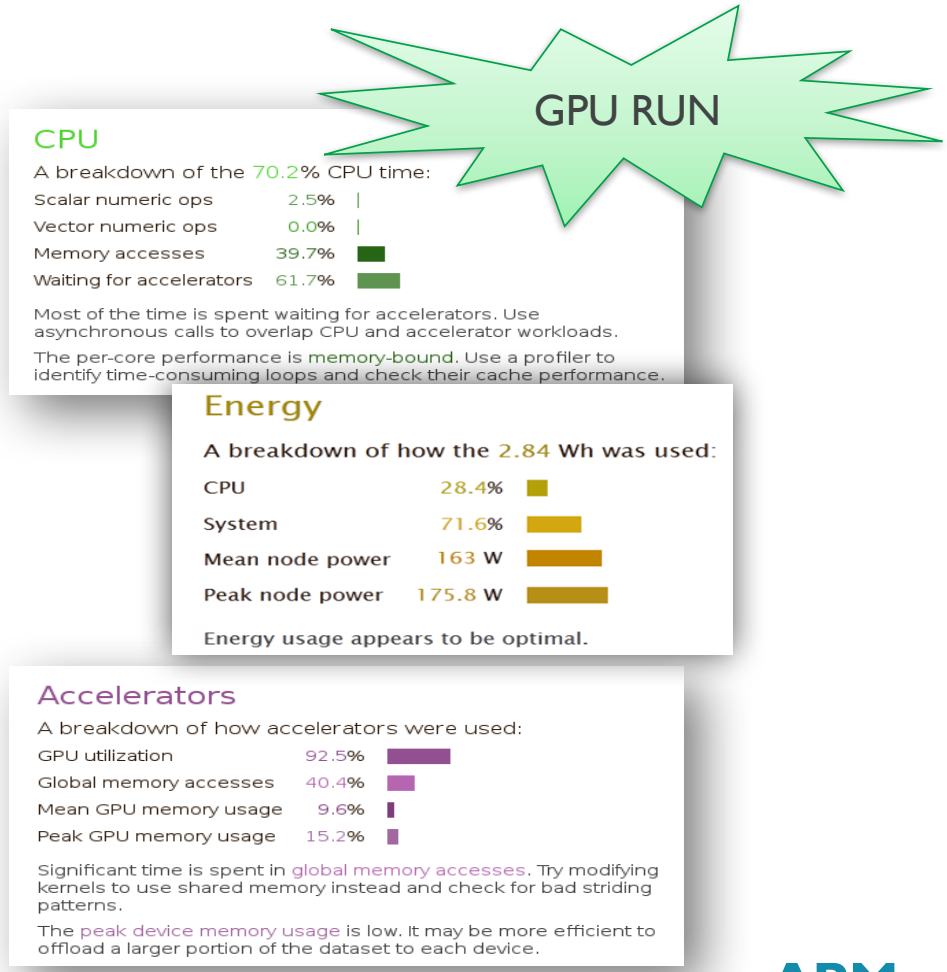
Lustre metrics in MAP



PAPI metrics in PR

Quantify gains immediately





ARMv8-A
Scalable Vector Extension

Introducing the Scalable Vector Extension (SVE)

A vector extension to the ARMv8-A architecture; its major new features:

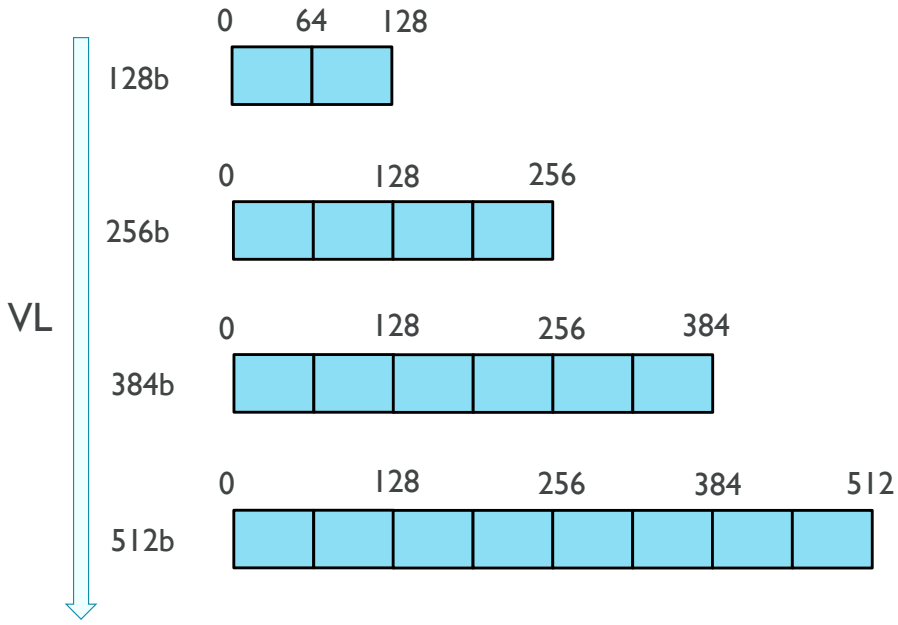
- Gather-load and scatter-store
- Per-lane predication
- Predicate-driven loop control and management
- Vector partitioning and SW managed speculation
- Extended integer and floating-point horizontal reductions

SVE is **not** an extension of Advanced SIMD

- A separate architectural extension with a new set of A64 instruction encodings
- Focus is HPC scientific workloads, not media/image processing

What's the vector length?

- There is **no** preferred vector length
  - Vector Length (VL) is hardware choice, from 128 to 2048 bits, in increments of 128
  - Does not need to be a power-of-2
  - *Vector Length Agnostic* programming adjusts dynamically to the available VL
  - No need to recompile, or to rewrite hand-coded SVE assembler or C intrinsics
  - Has extensive implications for loop optimizations



SVE Compiler status

- ARM C/C++ Compiler has SVE capabilities built in
- We have a public snapshot of our LLVM changes
  - https://github.com/ARM-Software/LLVM-SVE
  - These are being incrementally upstreamed into the main LLVM codebase
- We have started upstreaming our GCC changes
- Changes to GNU *binutils* are already upstream

- For more details of SVE usage see our blogs and whitepapers
  - https://www.community.arm.com/processors/b/blog/posts/technology-update-the-scalable-vector-extension-sve-for-the-armv8-a-architecture
  - https://cms.developer.arm.com/-/media/developer/developers/hpc/white-papers/a-sneak-peek-into-sve-and-vla-programming.pdf

Compiling with SVE

- To compile C code:

```
% armclang -O3 -march=armv8-a+sve  file.c -o file
```

- To compile C++ code:

```
% armclang++ -O3 -march=armv8-a+sve file.cpp -o file
```

ARM Instruction Emulator

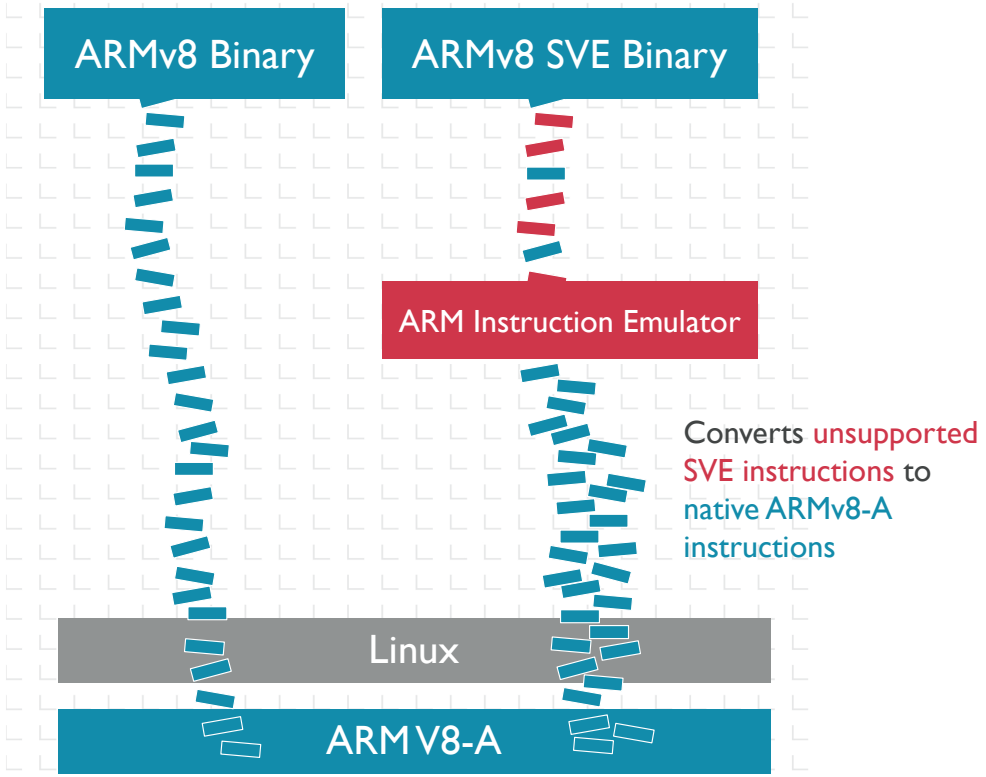
Run SVE binaries at near native speed on existing ARMv8-A hardware

Trap-and-emulate of illegal userspace instructions

- Natively supported instructions run at full speed
- Unsupported instructions are faithfully emulated in software

Full integration with ARM Code Advisor

- Plugin allows ARM Instruction Emulator to provide hotspot information and other metrics
- Command-line integration allows ARM Code Advisor workflows to seamlessly integrate with ARM Instruction Emulator



Running with ARM Instruction Emulator (1)

- To run SVE compiled code:

```
% armie --vector-length 256 ./example
```

- To list valid vector lengths:

```
% armie --list-vectors
128 256 384 512 640 768 896 1024 1152 1280 1408
1536 1664 1792 1920 2048
```

ARM Code Advisor with SVE

- Compile code with Insight as before

```
% armclang -O3 -march=armv8-a+sve -insight file.c –o file
```

- Run the executable using **both** ARM Code Advisor and ARM Instruction Emulator

```
% armcadvisor collect --armie --vector-length 256 \
      "./lulesh2.0 -s 15"
```

- Analysis and visualization stages as before

Final thoughts

ARM HPC in 2017

- The software ecosystem has matured significantly in the past two years
- Commercial compilers, libraries, debuggers and profilers are all now available to complement open source projects
- Users will notice very few hurdles to migrating codes as programming environments are very familiar
- ARM HPC hardware continues to appear from many partners with small proof of concept systems turning into bigger systems
  - SVE machines will only enhance performance further

Developer website : arm.com/hpc

ARM launched an HPC-specific microsite – home to our HPC ecosystem offering:

- Technical reference material
- How-to guides
- Latest news and updates from partners
- Links to downloads of HPC libraries
- Third-party software recommendations
- Web forum for community discussion and help

Participate and help drive the community

ARM HPC Products from arm.com/hpc

Supported, Integrated and Performant, available now

ARM Compiler for HPC

- ARM C/C++ Compiler
- ARM Performance Libraries
- GCC 6.0

ARM SVE Compiler for HPC

- ARM C/C++ SVE Compiler
- ARM SVE Performance Libraries
- GCC 6.0
- ARM Instruction Emulator
- GCC for SVE with LLVM OpenMP

ARM Code Advisor Beta

- ARM Code Advisor
- ARM C/C++ Compiler
- gfortran with LLVM OpenMP for ARM Code Advisor

Summary of today

- Overview of the state of the ARM HPC Ecosystem
- Hands-on experience running codes on a selection of ARM-based hardware
- Introduced to commercial HPC tools developed by ARM and ecosystem partners
- Visit arm.com/hpc for more information and tool access

ARM

http://www.arm.com/hpc

Contact: hpc@arm.com

Practical

Session 1

Using Linaro Developer Cloud instances

Username	Password	Work-Node
user00	a5ociKTy	node-0

```
ssh  user00@64.28.99.111          # Login to the login server
ssh node-0                        # then login to a worker node

Files describing the hands-on are in your home directory:
   session_1.txt    session_1.pdf

Run this on your local machine to copy the session info file:
  scp user00@64.28.99.111:/home/user00/session_1.pdf session_1.pdf
```

Session 2

Using Linaro Developer Cloud instances

```
Example Username   Password   Work-Node
          user00     a5ociKTy     node-0

  ssh  user00@64.28.99.111          # Login to the login server
  ssh node-0                        # then login to a worker node

  Files describing the hands-on are in you home directory:
     session_2.txt     session_2.pdf

  Run this on your local machine to copy the session info file:
    scp user00@64.28.99.111:/home/user00/session_2.pdf session_2.pdf
```