U.S. NAVAL RESEARCH LABORATORY

Evaluation of Cavium ThunderX2 for DoD Applications

ARM Users Group

November 2018

Keith Obenschain, Gopal Patnaik

U.S. NAVAL RESEARCH LABORATORY

Why is HPCMP looking at ARM Architecture?

High Performance Computing Modernization Program (HPCMP) needs viable options

- Large number of codes run, source code not available for all
  - ARM potentially a path to achieve higher performance with some refactoring
- ARM, Marvel and others are placing considerable resources into supporting HPC market
- Usual Suspects building HPC systems (HPE, Cray, Penguin Computing)

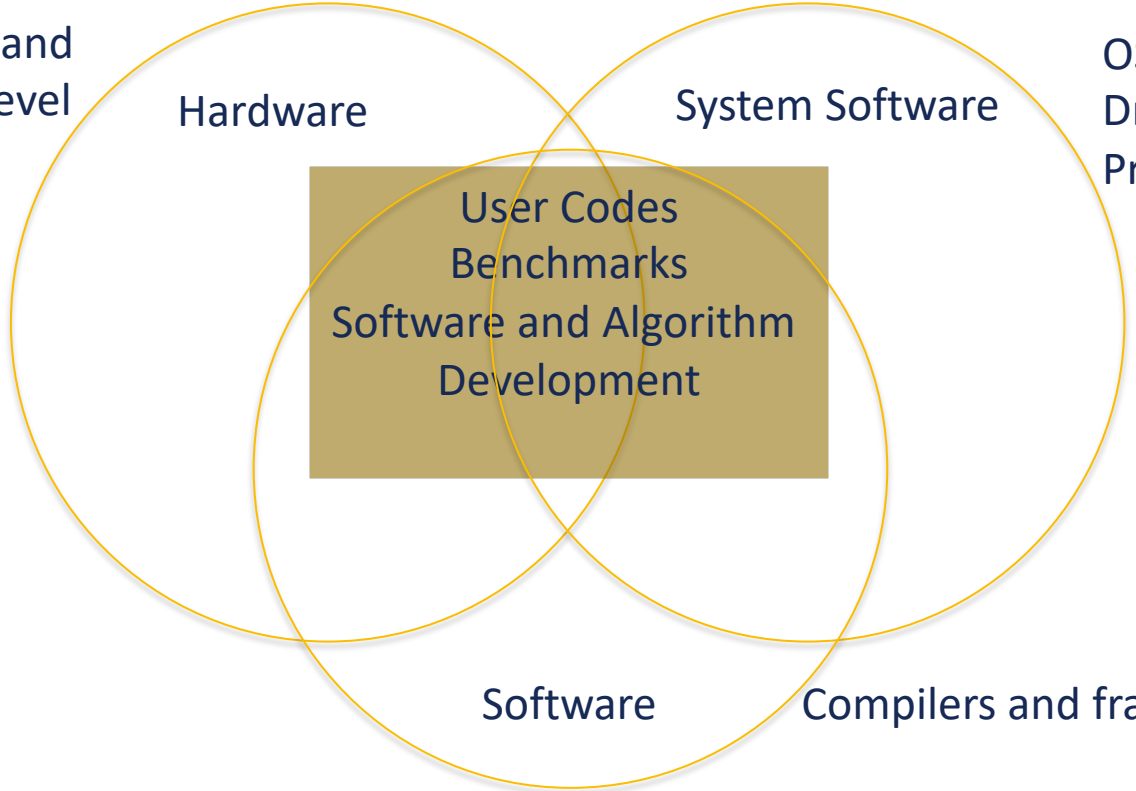
U.S. NAVAL RESEARCH LABORATORY

ARM Evaluation at NRL

Goal: Evaluate ARM architecture for feasibility for the next DoD HPCMP procurement cycle in 2019

- Overall system
  - System and CPU Architectures
  - Development Toolchains (Compilers, Profilers, etc)
  - Operating System, User Environment
- Synthetic and Mini-App Benchmarks
- Real-life production codes

ARM Hardware and Software Environment



U.S. NAVAL RESEARCH LABORATORY

NRL ARM Evaluation Status

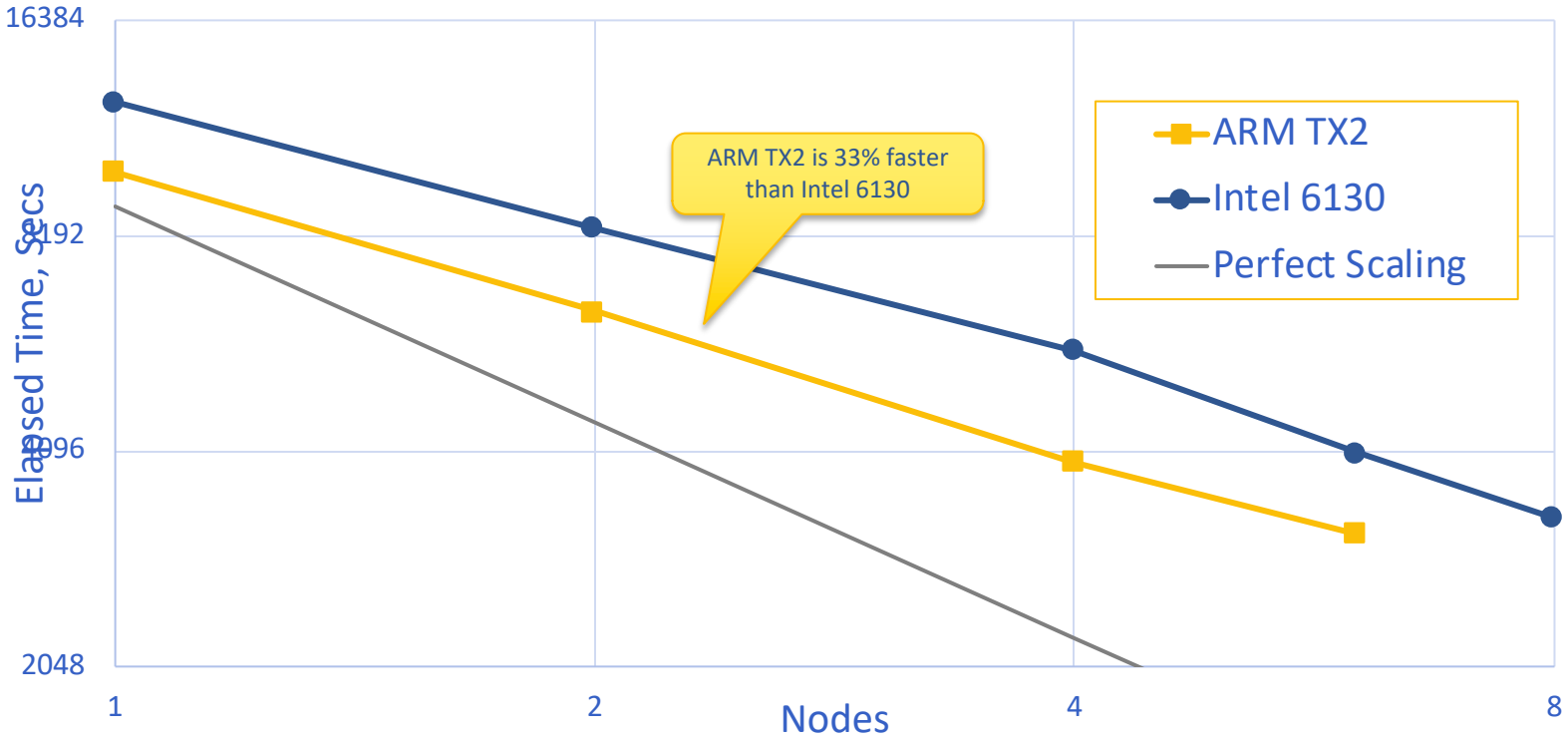
- Evaluation of NRL Codes and subset of HPCMP benchmarking codes, community benchmarks (e.g. HPCG, STREAMS)
- Testing IB on Cray system
  - Cavium hardware firmware is maturing
- Collaborating with ARM on their ARM LLVM Implementation

U.S. NAVAL RESEARCH LABORATORY

ARM Test Cluster

- Cray Early-Access ARM System
  - "White Box" system with partial Cray Environment
  - Cray selected to gain access to Cray Compiler Environment CCE
- Mellanox Infiniband at 100Gbps
- Cavium ThunderX 2 Processors
  - Eight Nodes with B0/B1 Stepping
  - For most benchmarks running with two SMT threads enabled at hardware level
- OS: Suse 12 SP3, Centos 7
- Cray, GCC and ARM Compilers

CTH Scaling Study



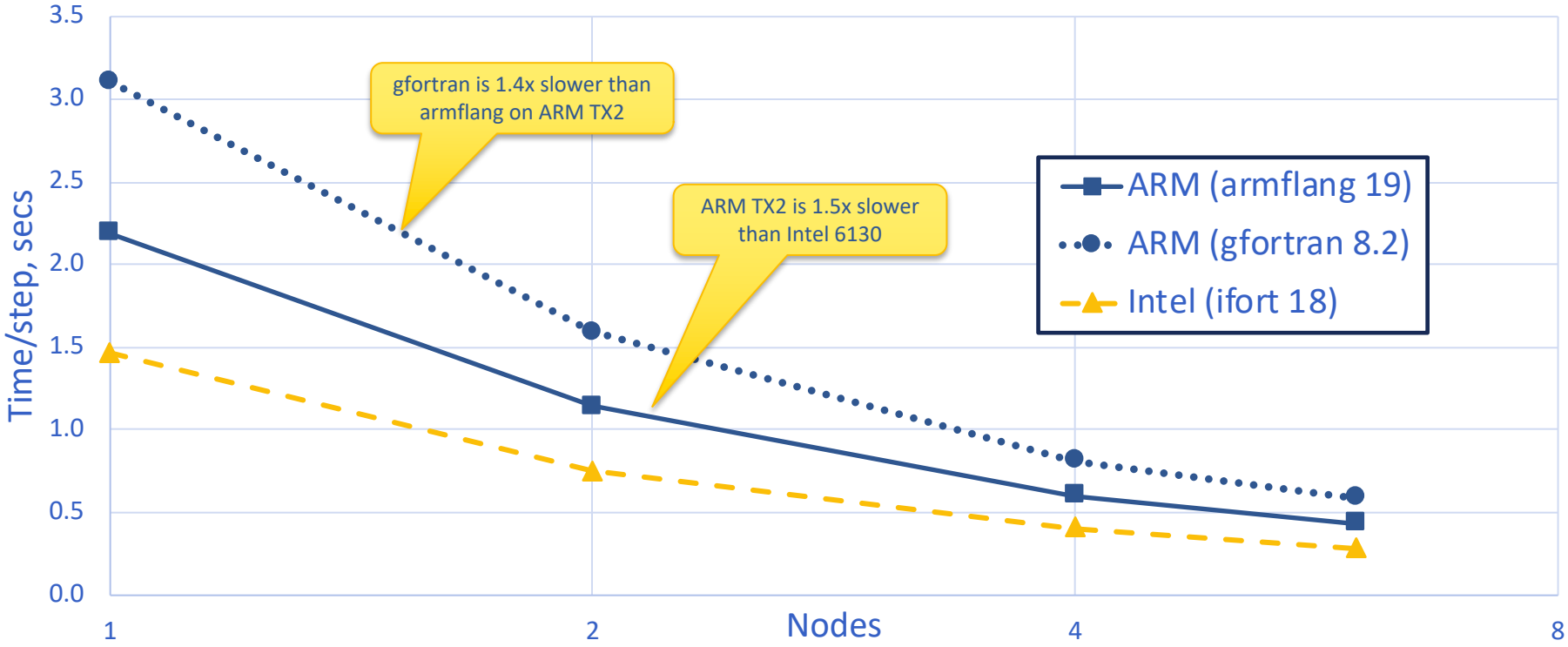
U.S. NAVAL RESEARCH LABORATORY

HYCOM Timings / "Small" HPCMP Benchmark

Processor	Compiler / MPI	Number of Nodes	Number of MPI Processes	MPI Processes/Socket	Timing (sec)
Cavium ThunderX2	GNU/OpenMPI	1	16	8	357
		2	32	16	209
		2	64	16	109
	LLVM/OpenMPI	1	16	8	326
		1	32	16	194
		2	64	16	102
Intel Xeon Gold 6130	GNU/OpenMPI	1	16	8	204
		1	32	8	121
		2	64	8	66
	LLVM/OpenMPI	1	16	8	224
		1	32	16	177
		2	64	16	80
	Intel/IntelMPI	1	16	16	151
		1	32	16	112
		2	64	16	60

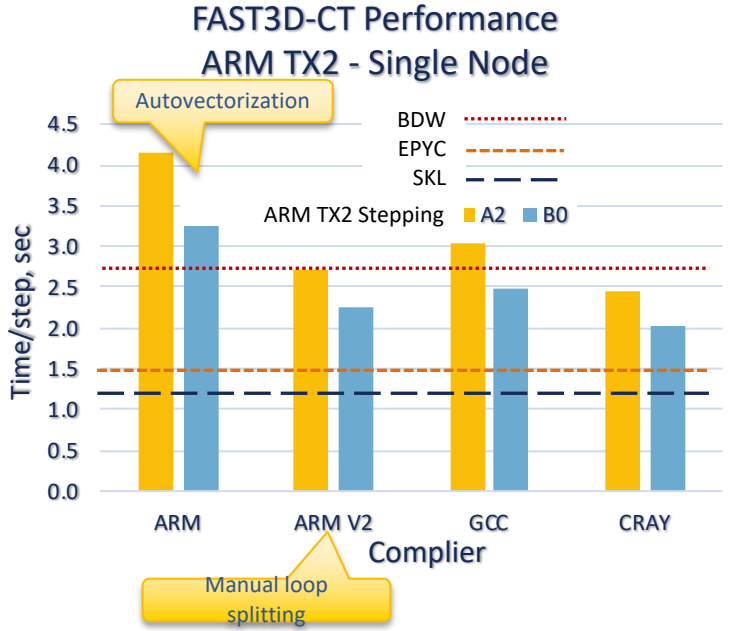
U.S. NAVAL RESEARCH LABORATORY

FAST3D-CT Scaling Study



Vectorization with Flang Fortran Compiler

- Flang compiler unable to autovectorize certain loops
  - Observed with ***all*** flang compilers
  - Loops vectorize with ifort, cray or gcc
- Read-after-write dependency
  - Use loop splitting to isolate offending statements
  - Done automatically since XMP days
  - Manual loop splitting recovers vector performance
- ARM working on Flang vectorization

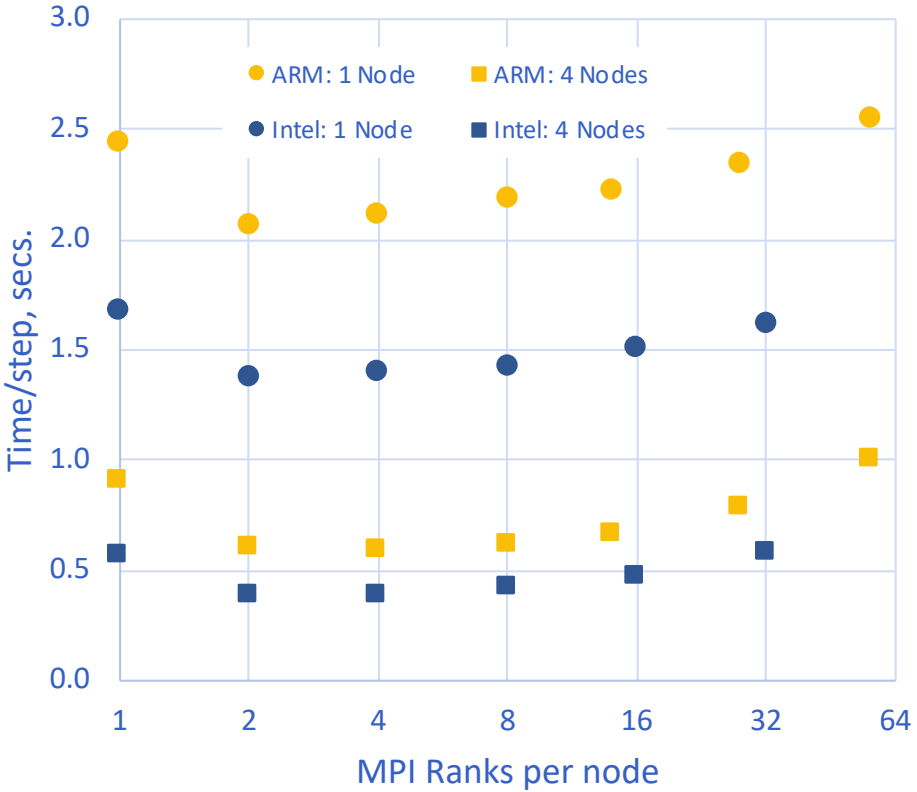


U.S. NAVAL RESEARCH LABORATORY

ARM LLVM Compiler

- NRL is designated “stress tester” for ARM LLVM Fortran
  - Found issues with OpenMP and vector implementation
  - ARM sorted issue and propagated changes to community, collateral fixes for other arch LLVM implementations
- Performance issues, doesn’t vectorize loops that other compilers can (e.g. Cray, Intel, GCC)

OpenMP / MPI Tuning



Only requires one rank per socket, less NUMA effects

Sweet spot corresponds to 1 rank per socket; no MPI communications needed within socket

U.S. NAVAL RESEARCH LABORATORY

ARM Interconnect Evolution

- Working with Cavium, ARM and Mellanox
- Combination of A2 Processors and older Firmware is a problem (~200-400MB/sec)
- B0 and new CPU firmware mitigates this issue

CPU	Firmware	Hosted NUMA Domain	Remote NUMA Domain
A2	4.x	400 MB/sec	200 MB/sec
B0	5.1	11.4 GB/sec	8.2 GB/sec
B0	6.3	11.5 GB/sec	9.6 GB/sec

U.S. NAVAL RESEARCH LABORATORY

ARM HPC Pros

Pros:

- ***Million+ line codes (e.g. CTH) compiled and run***
  - Software environment works, but needs to be optimized for ARM
- Very good memory bandwidth
  - Majority of DoD Codes are memory bandwidth bound
- Very good performance interconnect between cores in socket for hybrid OpenMP/MPI codes (one rank per socket)
  - Fewer NUMA effects makes runtime optimization easier
- Motivated hardware and software teams at companies
  - Responsive to issues found
- Additional competition in HPC space

U.S. NAVAL RESEARCH LABORATORY

ARM HPC Cons

Cons:

- Double-precision performance needs to improve
  - Future chips will have significant performance upgrades (e.g. ARM SVE ISA)
- Only one viable U.S. based HPC ARM vendor in the near-term
- Libraries and compilers need to be optimized
  - ARM, Cray, Cavium and others working on optimizing software

Current Work: ASTER on Astra

- **Direct drive inertial confinement fusion code**
  - Developed at University of Rochester
  - Spherical, structured grid
  - Two temperature explicit CFD
  - Tabular equation of state
  - Implicit species heat conduction
  - Laser ray tracing
  - Fusion reactions
  - Multi-group radiation diffusion
  - Operator split time stepping
- **Goal: Scientifically useful run on ASTRA**

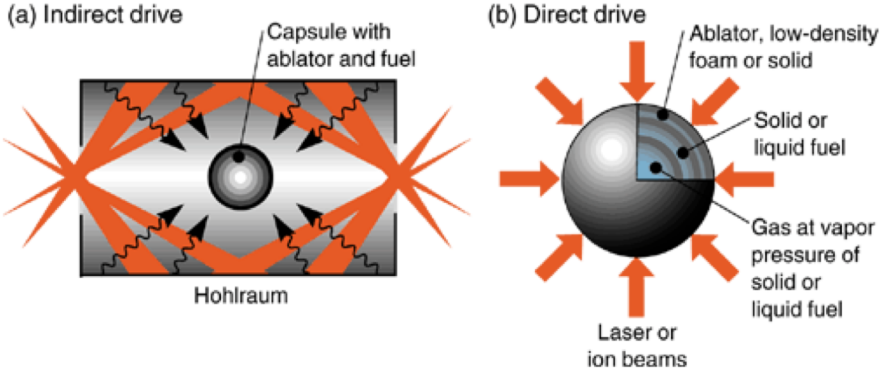


Figure 1. Ignition experiments on NIF will use indirect drive to heat the inside of a cylinder (hohlraum). The incident laser light will enter the cylinder through holes at its end caps and will be converted to x rays that will converge on a 2-millimeter-diameter capsule. NIF also has an option to conduct direct-drive experiments, with the laser light directly incident on a capsule.

[1]I.V. Igumenshchev, et al. **"Three-Dimensional Modeling of Direct-Drive Cryogenic Implosions on OMEGA."** *Physics of Plasmas* 23, 052702 (2016).
[2]Image: https://str.llnl.gov/str/Haan.html

U.S. NAVAL RESEARCH LABORATORY

Summary

- ARM is an important architecture for upcoming HPCMP procurements
  - Tremendous progress with floating-point, memory bandwidth and software
- Issues with ARM for HPC exist, but the vendors and community are working through them

U.S. NAVAL RESEARCH LABORATORY

Backups

U.S. NAVAL RESEARCH LABORATORY

DoD/DOE ITAR Issues

Problem: How can we collaborate with international community and vendors?

- **Important DoD/DOE Codes fall under ITAR restrictions**
- ARM HPC Community is international
  - Significant HPC efforts in Europe, Asia
  - ARM compiler teams in UK
  - ARM IP owned by Japan based SoftBank
- DOE uses purpose-built mini-apps, not feasible for DoD

Potential Solution: Automatic generation of Mini-Apps

Automatic generation of ASTER mini-app

- **Extract the multigrid solver to make *miniaster***
- ***Smaller codebase***
- ***Does not fall under ITAR***

!$kgen decoration

```
!$kgen begin_callsite multigrid
!jgw! %fine is not written to kgen files so this association is added
sg => top_grid
do while(associated(sg%crse))
   sg%crse%fine => sg
   sg           => sg%crse
end do

! the real kernel
! remove "freq_groups_loop" label as kgen can't handle it.
!freq_groups_loop: do ifg=1,nfg
do ifg=1,nfg
   sg => top_grid
   call multigrid_solver(ifg,errmax,sg)
end do !freq_groups_loop
!$kgen end_callsite multigrid
```

KGen invocation via Makefile

```
File Edit Options Buffers Tools Makefile Help
KGEN_HOME := $(HOME)/devel/packages/KGen
KGEN := ${KGEN_HOME}/bin/kgen

SRC_DIR := ${PWD}/../src
SRC := ${SRC_DIR}/radtrans/rad_diffusion.f90

test:
        ${KGEN} \
        --cmd-build "cd ${SRC_DIR}; make -j8" \
        --cmd-clean "cd ${SRC_DIR}; make clean" \
        --cmd-run   "cd ${SRC_DIR}; make run" \
        --exclude-ini exclude.ini \
        --invocation 0-383:0:0 \
        --mpi enable \
        --check tolerance=1.0D-10 \
        --verbose 3 \
        ${SRC}
```

Python miniapp extractor from UCAR

https://github.com/NCAR/KGen

Fortran only

- Requires modules, no free subroutines
- Does not handle object oriented Fortran well

U.S. NAVAL RESEARCH LABORATORY

ARM Current Status

- Testing CentOS ARM Edition
- Working with Cavium, ARM and Mellanox on IB performance issues
- Upgrade to ARM B0
- ARM Compiler collaborations