Early Evaluation of Fugaku A64FX Architecture Using Climate Workloads

Sarat Sreepathi
Oak Ridge National Laboratory

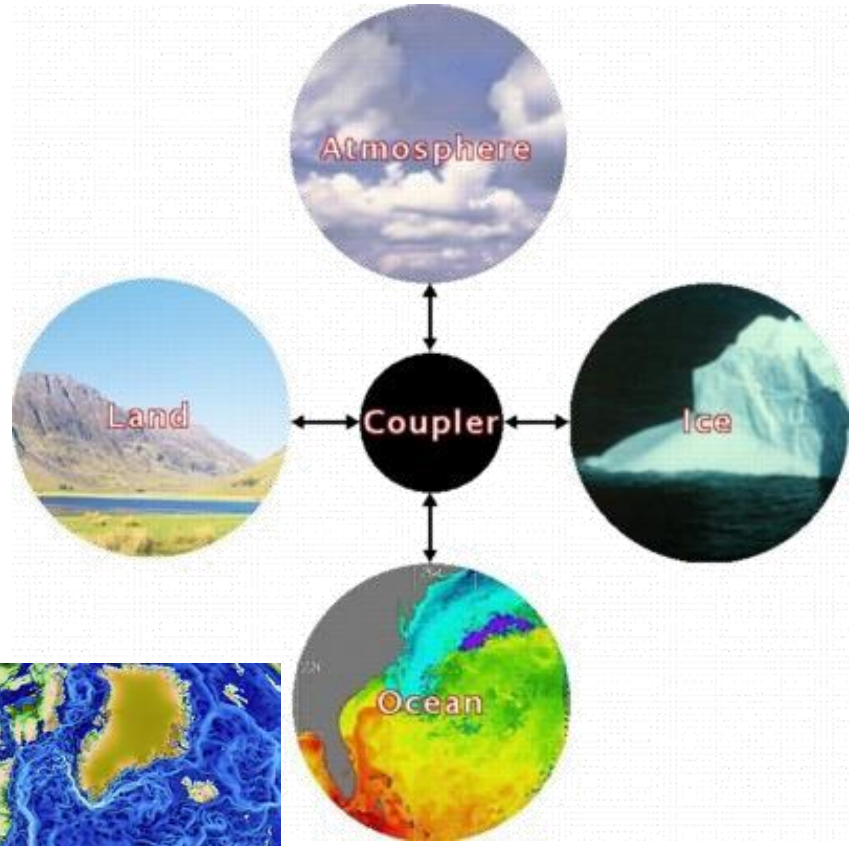
Mark Taylor
Sandia National Laboratories

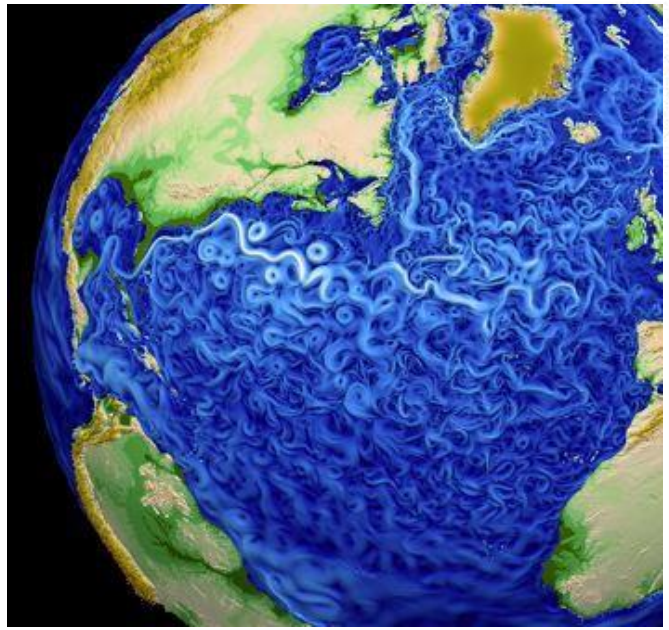
EAHPC Workshop
IEEE Cluster 2021
September 7, 2021

National Nuclear Security Administration
U.S. DEPARTMENT OF ENERGY | Office of Science

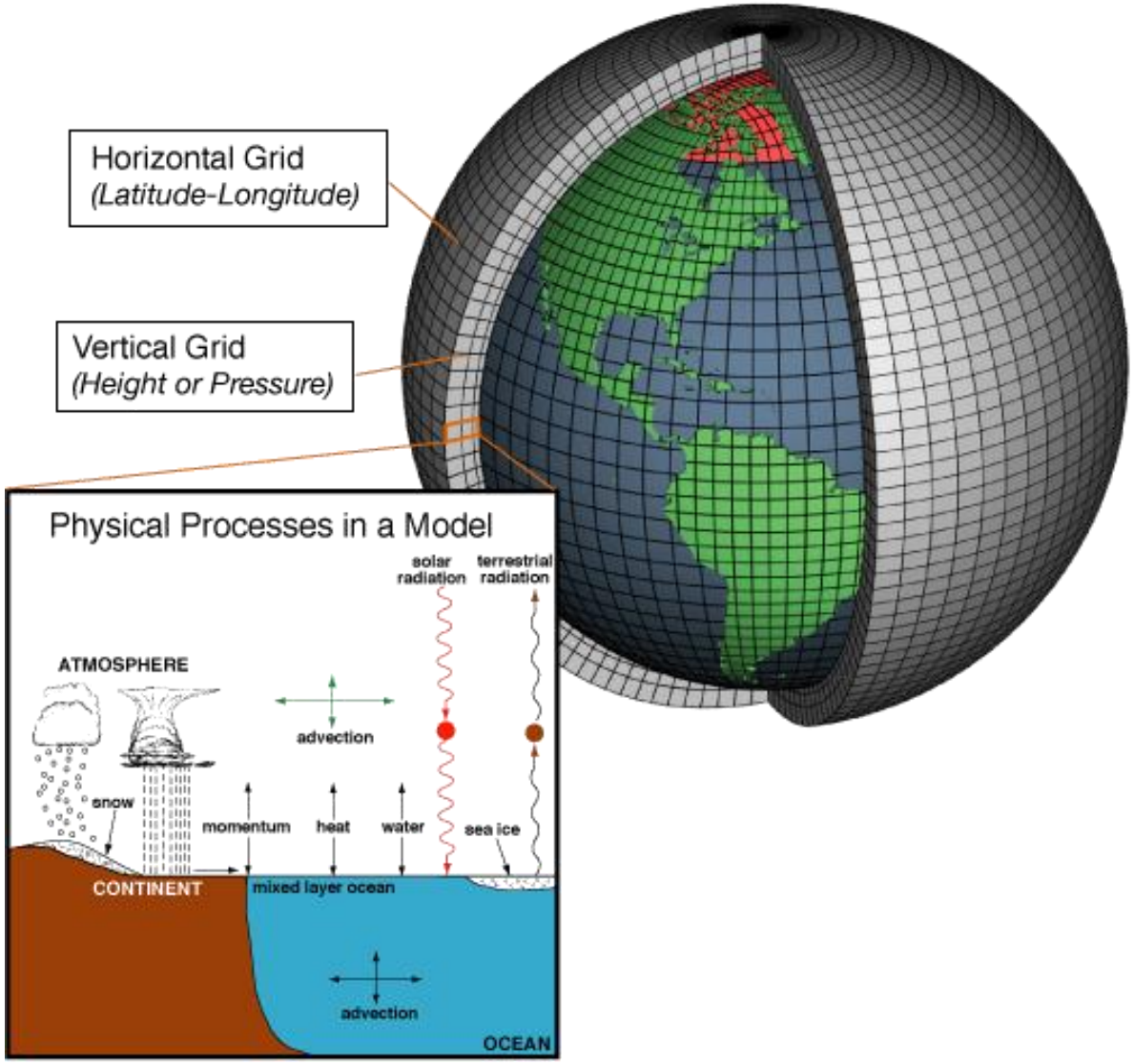
E3SM

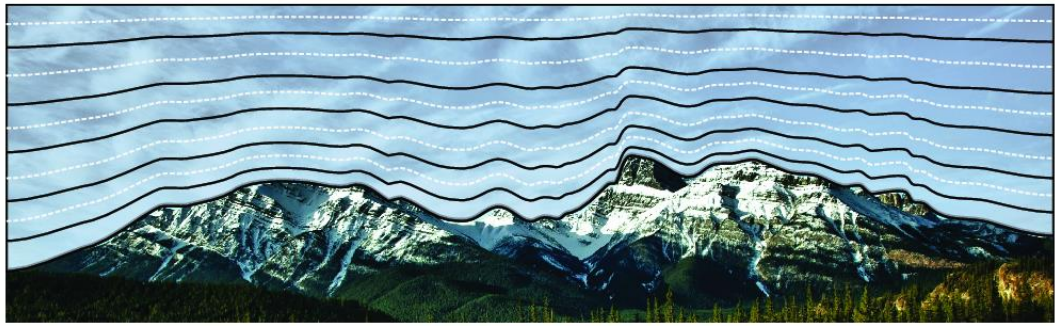
- Global Earth System Model
- Atmosphere, Land, Ocean and Ice component models
- 8 U.S. Dept. of Energy (DOE) labs, 12 university subcontracts, 87 individuals
- Development driven by DOE mission interests: Energy/water issues looking out 40 years
- Key computational goal: Ensure E3SM will run well on upcoming DOE pre-exascale and exascale computers
- E3SM is open source / open development
  - Website: www.e3sm.org
  - Github: https://github.com/E3SM-Project
  - DOE Science youtube channel: https://www.youtube.com/channel/UC_rhpi0lBeD1U-6nD2zvlBA





Atmosphere Component





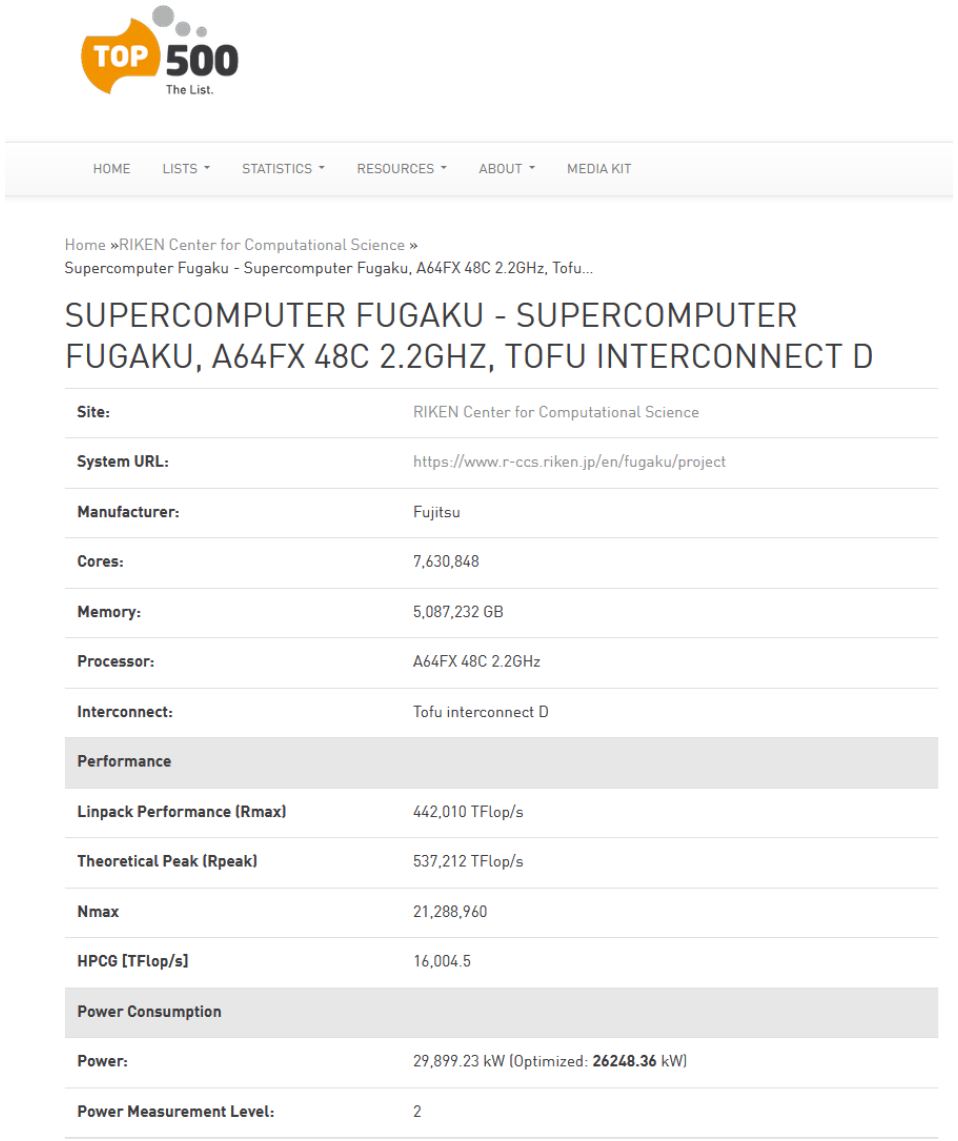
hydrostatic-pressure terrain-following coordinates

- Dynamical Core
  - Solves the Atmospheric Primitive Equations
  - Linear transport of 40 atmospheric species
  - 72 vertical levels – 0.8 km avg. spacing
  - Benchmark (two versions): Fortran (preqx) and C++ (preqx_kokkos)

Terrain following figure: D. Hall, CU Boulder
Source: http://celebrating200years.noaa.gov/breakthroughs/climate_model/welcome.html

Fugaku

- #1 on Top500
- RIKEN Center for Computational Science
- Key Characteristics of A64FX*
  - Arm 64-bit with 512-bit SVE (Scalable Vector Extensions)
  - High Bandwidth Memory
  - Low Power



*https://www.fujitsu.com/downloads/SUPER/a64fx/a64fx_datasheet_en.pdf

https://www.top500.org/system/179807/

Architecture Comparison: Metrics

- Single node workload for understanding h/w trends (ca 2012+)
- Performance Efficiencv metric: number of element remap timesteps **per second**

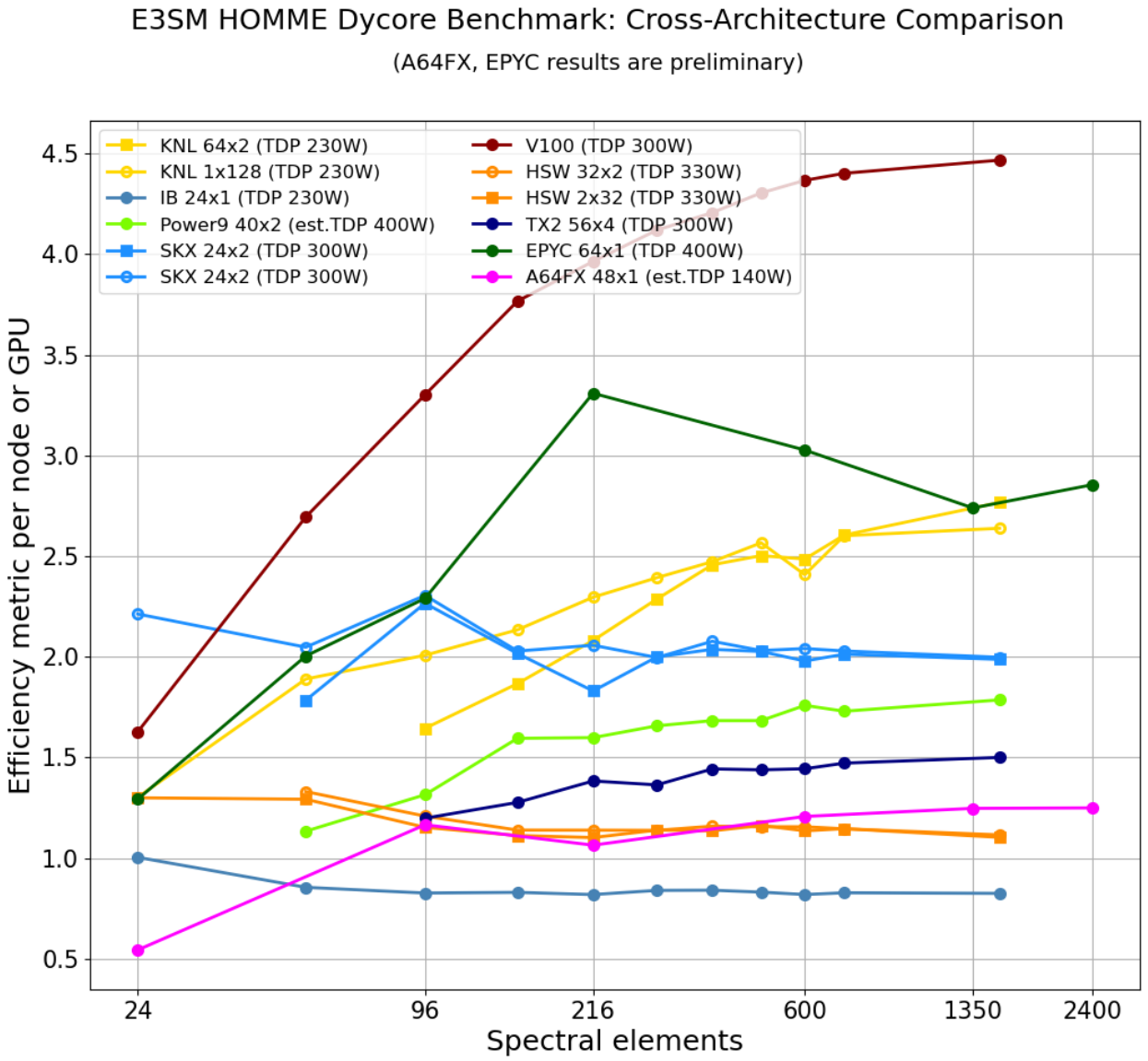
$$E_{perf} = \frac{N_e * N_t}{(prim\_main\_loop * num\_devices)}$$

  - N_e is the number of spectral elements
  - N_t is the number of remap timesteps (34 for the Fugaku experiments)
  - *prim_main_loop* is the main computation loop timer
  - *num_devices* is 1 for CPU nodes or the number of GPUs per node for GPU systems
- Power Efficiency metric: number of element remap timesteps **per Watt**

$$E_{power\_tdp} = \frac{E_{perf}}{\mathrm{TDP}}$$

  - Thermal Design Power (TDP)

Architecture Comparison: Performance Efficiency



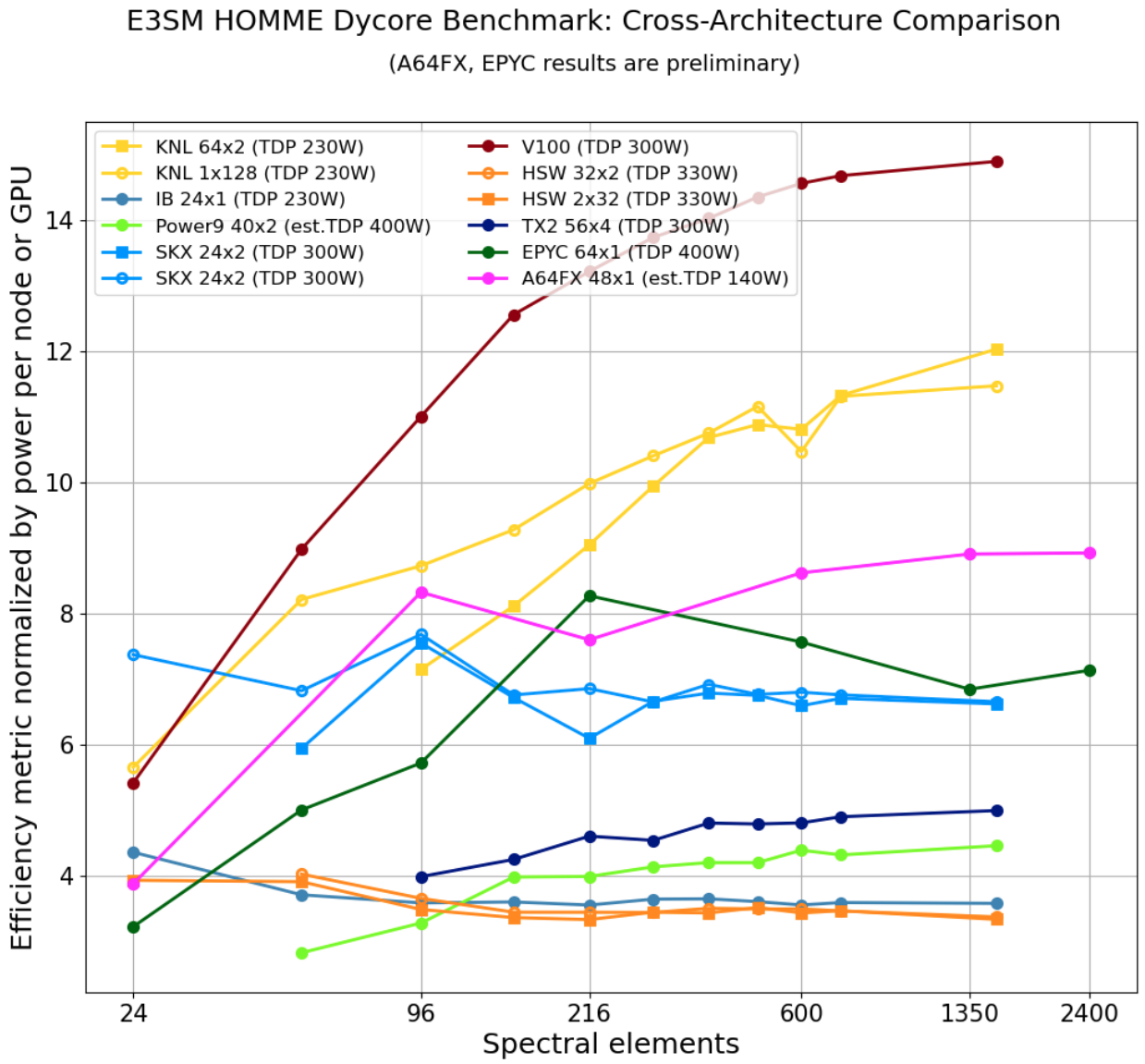
Inform configurations where GPU systems can outperform CPU systems

Fugaku Node: Single A64FX socket
GNU Fortran + MPI (48 ranks)

Note: Top Red (Volta V100), Pink (A64FX), Orange (Dual-socket Haswell)
Higher is better

* Plot of the efficiency metric normalized by power consumption on various hardware architectures. The legend includes a short descriptor for each architecture along with the number of parallel processes times (x) the number of threads and includes TDP in parenthesis. Specifically, the labels map as follows: KNL (Intel® Knights Landing), IB (Intel®Ivy Bridge), SKX (Intel® Skylake), V100 (NVIDIA® Volta), HSW (Intel® Haswell), A64FX (Fujitsu® A64FX), Power9 (IBM® POWER9), TX2 (Marvell®ThunderX2), EPYC (AMD® EPYC).

Architecture Comparison: Power Efficiency



A64FX: Promising performance/watt

Fugaku Node: Single A64FX socket
GNU Fortran + MPI (48 ranks)

Note: Top Red (Volta V100), Pink (A64FX), Yellow (KNL)
Higher is better

* Plot of the efficiency metric normalized by power consumption on various hardware architectures. The legend includes a short descriptor for each architecture along with the number of parallel processes times (x) the number of threads and includes TDP in parenthesis. Specifically, the labels map as follows: KNL (Intel® Knights Landing), IB (Intel®Ivy Bridge), SKX (Intel® Skylake), V100 (NVIDIA® Volta), HSW (Intel® Haswell), A64FX (Fujitsu® A64FX), Power9 (IBM® POWER9), TX2 (Marvell®ThunderX2), EPYC (AMD® EPYC).

Compiler Evaluation

- Fujitsu compilers slower than GNU (unexpected)
  - Fortran ~ 50% slower
  - C++ runtime issues
- Ofast/Kfast option helps!
- Workload: Fortran version faster than C++ version (GNU)
- Needs further investigation: Hybrid mode (MPI+OpenMP) not performant yet

Time (sec)
Lower is better

Spectral Elements (Ne)	GFortran O3	GFortran Ofast	G++ O3	G++ Ofast	FjFortran Kfast
96	3.51	2.85	4.37	3.39	5.62
216	8.52	6.86	10.36	7.96	14.50
600	21.29	16.91	26.02	19.97	37.20
1350	46.58	36.84	56.51	43.31	83.21
2400	82.09	65.30	99.43	76.33	145.23

Compiler versions: GNU 10.2.0 and Fujitsu 4.5.0 (tcsds-1.2.31).
Column labels: Gfortran, FJFortran refer to Fortran benchmark built with GNU and Fujitsu respectively. G++ refers to C++(preqx_kokkos) version. Boost mode (2.2 GHz) enabled on Fugaku compute nodes.

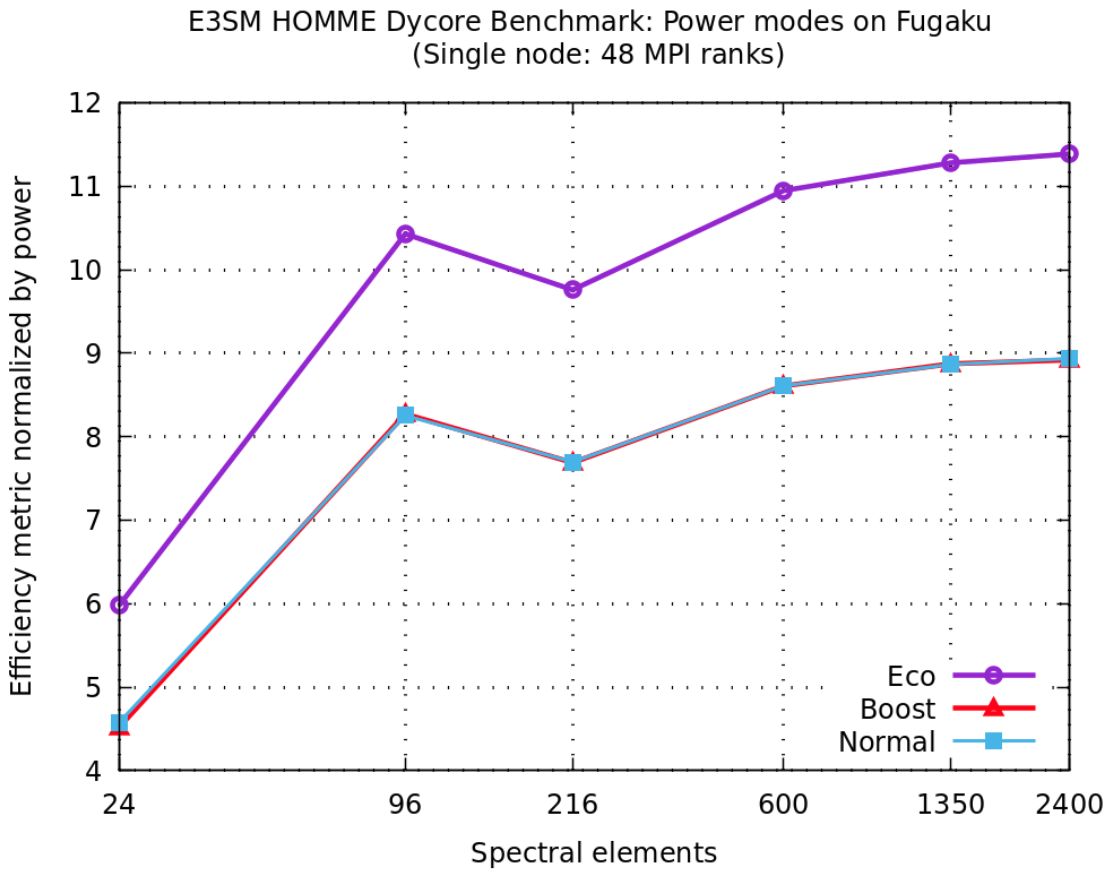
Power and Performance tradeoffs

- Power Efficiency metric normalized by the measured power on the compute node

$$E_{power\_measured} = \frac{E_{perf}}{measured\_power}$$

- PowerAPI
- Three modes
  - Normal (2 GHz)
  - Boost (2.2 GHz)
  - Eco (2 GHz/eco_state=2)
- Fortran version with GNU

Higher is better



Power and Performance tradeoffs

TABLE II
MAIN LOOP TIMER (SEC) WITH DIFFERENT COMPILERS AND POWER MODES ON FUGAKU

Spectral Elements (N_e)	Boost GFortran	Normal GFortran	Eco GFortran	Boost FJFortran	Normal FJFortran
96	2.849	3.19	3.273	5.619	6.198
216	6.863	7.667	7.825	14.495	15.905
600	16.91	18.901	19.199	37.2	40.617
1350	36.844	41.206	41.802	83.211	91.512
2400	65.299	72.819	73.817	145.229	160.263

TABLE III
MEASURED POWER (W) ON FUGAKU COMPUTE NODE

N_e	Boost	Normal	Eco
96	138.51	123.98	95.63
216	139.18	124.47	96.18
600	140.16	125.38	97.08
1350	140.41	125.64	97.35
2400	140.11	125.4	97.09

Performance Characterization

Performance Characterization

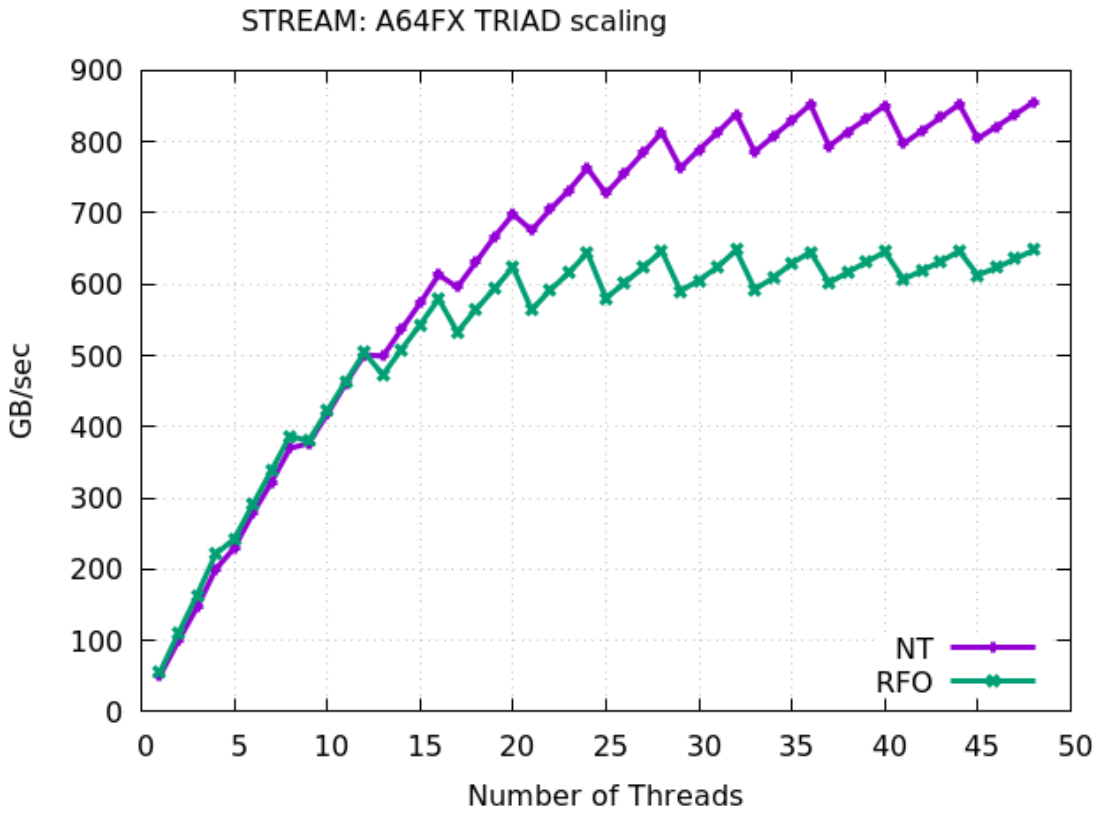
- Fujitsu Profiling Tools
- Work with Fujitsu compiler executables only
- Recall: GNU Fortran twice as fast as Fujitsu
- Motivation: Potential for improved memory bandwidth?
- Identify relevant optimizations

TABLE IV
PERFORMANCE PROFILE OF ATMOSPHERIC BENCHMARK (2400 SPECTRAL ELEMENTS)

Metric	Value
GFLOPS	42.95
Floating-point peak ratio(%)	1.27
Memory throughput (GB/s)	57.12
Memory throughput peak ratio(%)	5.57
SIMD instruction rate (%)	7.15
SVE operation rate (%)	75.10
Instructions per cycle (IPC)	0.92

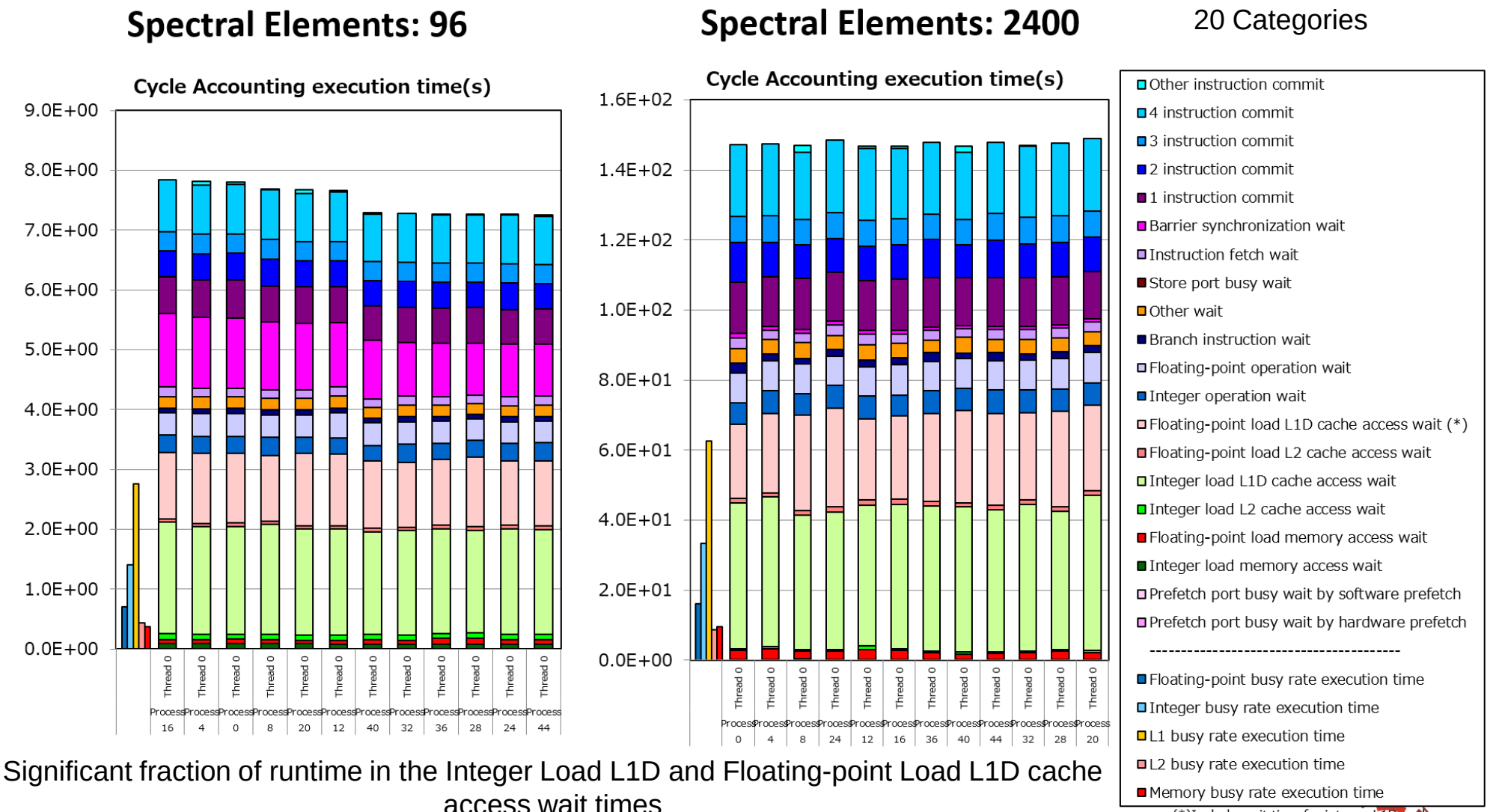
Memory Bandwidth Detour: STREAM

- Understand memory bandwidth characteristics on A64FX
  - TRIAD kernel using Fujitsu compiler on Fugaku
  - Best rate of 856 GB/s using 48 threads
  - Needs extra compiler options for optimal performance
- High sensitivity to NUMA effects
- Benefit from demand paging when straddling multiple Core Memory Groups (CMGs)



NT: Non-temporal stores - requires streaming store instructions (Fujitsu compiler generated these, GNU didn't)
RFO: Read for ownership - extra read into cache line before writing

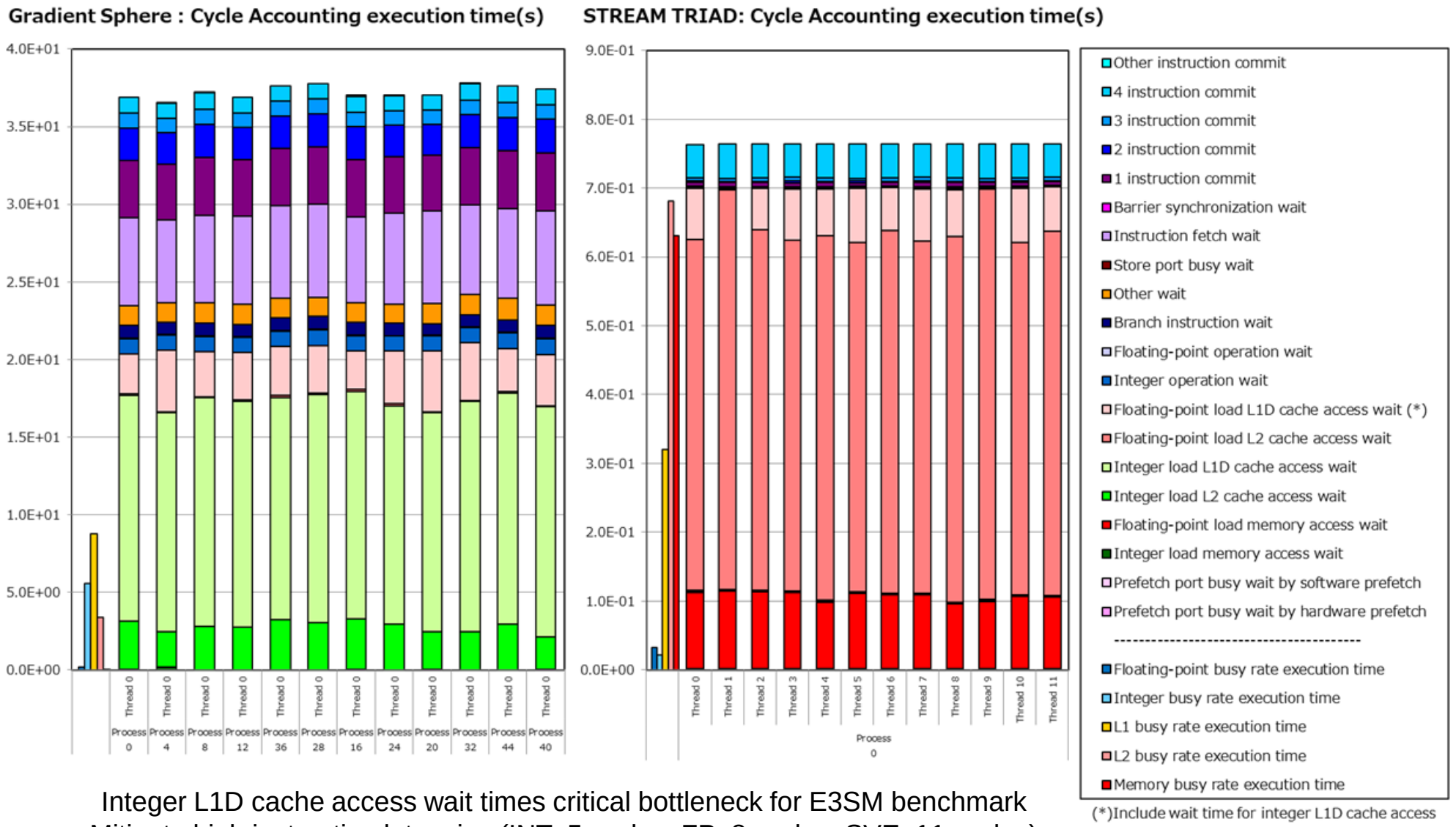
Performance Characterization: Instruction mix



Significant fraction of runtime in the Integer Load L1D and Floating-point Load L1D cache access wait times

Left: pink section is Barrier synchronization wait

Instruction mix Comparison: Gradient sphere kernel vs. STREAM TRIAD



Integer L1D cache access wait times critical bottleneck for E3SM benchmark

Mitigate high instruction latencies (INT: 5 cycles, FP: 8 cycles, SVE: 11 cycles)

Biharmonic operator kernel

```
function laplace_sphere_wk(s,deriv,elem) result(laplace)
  real(kind=real_kind), intent(in   ) :: s(np,np)
  type (derivative_t) , intent(in   ) :: deriv
  type (element_t)    , intent(in   ) :: elem
  real(kind=real_kind) :: laplace(np,np)
  real(kind=real_kind) :: grads(np,np,2), oldgrads(np,np,2)
  integer :: i,j
  grads = gradient_sphere(s,deriv,elem%Dinv)
  oldgrads = grads
  do j = 1 , np
    do i = 1 , np
      grads(i,j,1) = oldgrads(i,j,1)*elem%tensorVisc(i,j,1,1) + &
                     oldgrads(i,j,2)*elem%tensorVisc(i,j,1,2)
      grads(i,j,2) = oldgrads(i,j,1)*elem%tensorVisc(i,j,2,1) + &
                     oldgrads(i,j,2)*elem%tensorVisc(i,j,2,2)
    enddo
  enddo
  laplace = divergence_sphere_wk(grads,deriv,elem)
end function laplace_sphere_wk



subroutine biharmonic_wk_scalar(elem,qtens,deriv,nets,nete)
  type (element_t)     , intent(in   ) :: elem(:)
  real (kind=real_kind), intent(inout) :: qtens(np,np,nlev,qsize,nets:nete)
  type (derivative_t)  , intent(in   ) :: deriv
  integer              , intent(in   ) :: nets
  integer              , intent(in   ) :: nete
  integer :: k,i,j,ie,q
  do ie = nets , nete
    do q = 1 , qsize
      do k = 1 , nlev
        qtens(:,:,k,q,ie) = laplace_sphere_wk(qtens(:,:,k,q,ie),deriv,elem(ie))
      enddo
    enddo
  enddo
end subroutine biharmonic_wk_scalar
```

TABLE V
PERFORMANCE OF BIHARMONIC KERNEL

Compiler	Option	Time (sec)
Fujitsu Fortran	NOSVE	0.078
Fujitsu Fortran	SVE	0.092
GFortran	NOSVE	0.015
GFortran	SVE	0.013

- GNU clear winner
- Optimization reports shed light
- Fujitsu Fortran compiler
  - Issue: Inline expansion of functions
  - Despite explicit flags
- GNU does it by default

https://github.com/E3SM-Project/codesign-kernels/blob/master/atmosphere/biharmonic_wk_kernel.F90

Conclusions & Future Work

- A64FX delivered decent power efficiency for climate workloads
- Compilers: GNU Fortran was the best performer
  - Fujitsu compiler yielded about half the performance
- Fugaku: Power modes offer performance vs. power trade-offs
  - Eco mode was optimal for our workload

Future Work

- Potential for improving memory bandwidth utilization
- Investigate optimization strategies using kernels
- Debug hybrid (MPI + OpenMP) performance
- Scaling studies to understand interconnect performance

17

Community interactions

- CMake did not recognize Fujitsu compilers and math libraries
  - Worked with Kitware, ARM and Fujitsu folks to test patches
  - Kitware added initial support
  - Superseded by official patch from Fujitsu
  - This benefited lot of other applications and libraries
- Sandia and ORNL A64FX testbeds were invaluable in evaluating various compilers (ARM, GNU and Fujitsu)
- Fugaku Helpdesk: Productive interactions on multiple issues
  - Support for GNU compiler with Fujitsu MPI

Acknowledgments

EXASCALE COMPUTING PROJECT

Energy Exascale Earth System Model

powered by FUGAKU

©RIKEN
画像の無断使用・無断転載を禁じます

OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY

DOE-MEXT Collaboration

Questions?
sarat@ornl.gov