The Supercomputer “Fugaku” and Software, programming models and tools

Mitsuhisa Sato Team Leader of Architecture Development Team

Deputy project leader, FLAGSHIP 2020 project
Deputy Director, RIKEN Center for Computational Science (R-CCS)

Professor (Cooperative Graduate School Program), University of Tsukuba

FLAGSHIP2020 Project “Fugaku”

Missions

- Building the Japanese national flagship supercomputer “Fugaku “(a.k.a post K), and
- Developing wide range of HPC applications, running on Fugaku, in order to solve social and science issues in Japan (application development projects was over at the end of march, 2020)

Overview of Fugaku architecture

Node: Manycore architecture

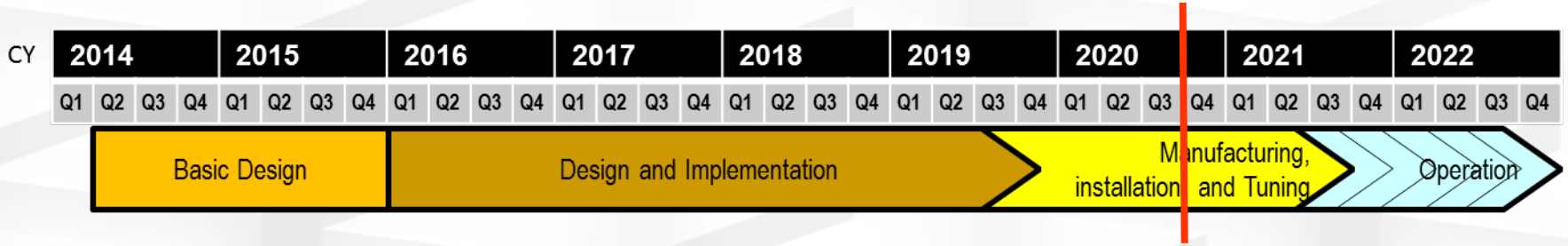
- Armv8-A + SVE (Scalable Vector Extension)
- SIMD Length: 512 bits
- # of Cores: 48 + (2/4 for OS) (> 3.0 TF / 48 core)
- Co-design with application developers and high memory bandwidth utilizing on-package stacked memory (HBM2) 1 TB/s B/W
- Low power : 15GF/W (dgemm)

Network: TofuD

- Chip-Integrated NIC, 6D mesh/torus Interconnect

Status and Update

- March 2019: The Name of the system was decided as “Fugaku”
- Aug. 2019: The K computer decommissioned, stopped the services and shutdown (removed from the computer room)
- Oct 2019: access to the test chips was started.
- Nov. 2019: Fujitsu announce FX1000 and FX700, and business with Cray.
- Nov 2019: Fugaku clock frequency will be 2.0GHz and boost to 2.2 GHz.
- Nov 2019: Green 500 1st position!
- Oct-Nov 2019: MEXT announced the Fugaku “early access program” to begin around Q2/CY2020
- Dec 2019: Delivery and Installation of “Fugaku” was started.
- May 2020: Delivery completed
- June 2020: 1st in Top500, HPCG, Graph 500, HPL-AI at ISC2020



Fugaku won 1st position in 4 benchmarks!

Bencmark	1st	Score	Unit	2nd	Score	1st/ 2nd
TOP500 (LINPACK)	Fugaku	415.5	PFLOPS	Summit（US）	148.6	2.80
HPCG	Fugaku	13.4	PFLOPS	Summit（US）	2.93	4.57
HPL-AI	Fugaku	1.42	EFLOPS	Summit（US）	0.55	2.58
Graph500	Fugaku	70,980	GTEPS	太湖之光 TaihuLight (China)	23,756	2.99

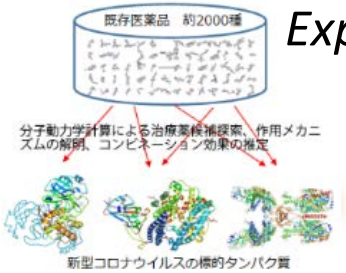
2 to 4 times faster in every benchmark!

MEXT Fugaku Program: Fight Against COVID19

Fugaku resources made available a year ahead of general production (more research topics under international solicitation)

A partner of international COVID-19 HPC Consortium

Medical-Pharma



Exploring new drug candidates for COVID-19

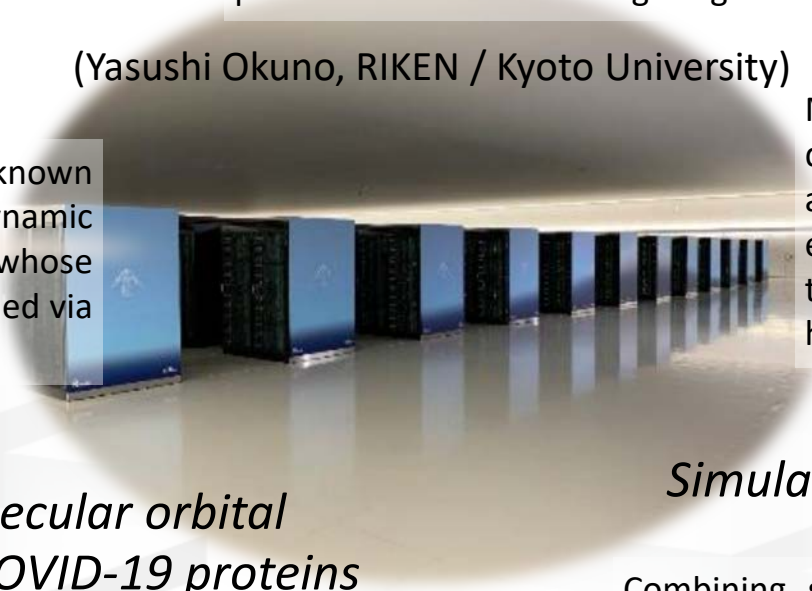
Large-scale MD to search & identify therapeutic drug candidates showing high affinity for COVID-19 target proteins from 2000 existing drugs

(Yasushi Okuno, RIKEN / Kyoto University)

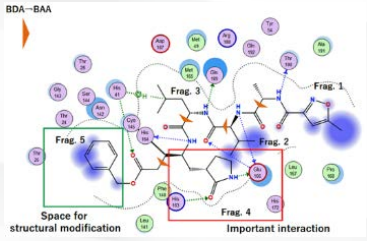
Prediction of conformation… dynamics of proteins on the surface of SARS-Cov-2

GENESIS MD to interpolate unknown experimentally undetectable dynamic behavior of spike proteins, whose static behavior has been identified via Cryo-EM

(Yuji Sugita, RIKEN)



Fragment molecular orbital calculations for COVID-19 proteins



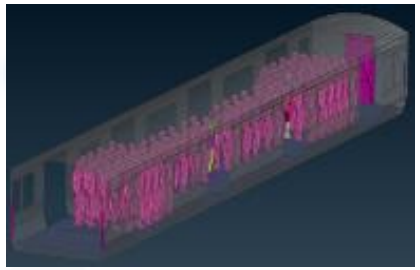
Large-scale, detailed interaction analysis of COVID-19 using Fragment Molecular Orbital (FMO) calculations using ABINIT-MP

(Yuji Mochizuki, Rikkyo University)

Societal-Epidemiology

Prediction and Countermeasure for Virus Droplet Infection under the Indoor Environment

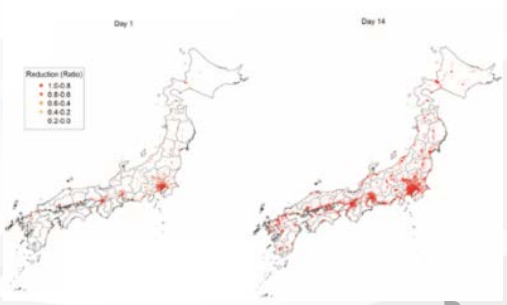
Massive parallel simulation of droplet scattering with airflow and hat transfer under indoor environment such as commuter trains, offices, classrooms, and hospital rooms



(Makoto Tsubokura, RIKEN / Kobe University)

Simulation analysis of pandemic phenomena

Combining simulations & analytics of disease propagation w/contact tracing apps, economic effects of lockdown, and reflections social media, for effective mitigation policies



(Nobuyasu Ito, RIKEN)

KPIs on Fugaku development in FLAGSHIP 2020 project

3 KPIs (key performance indicator) were defined for Fugaku development

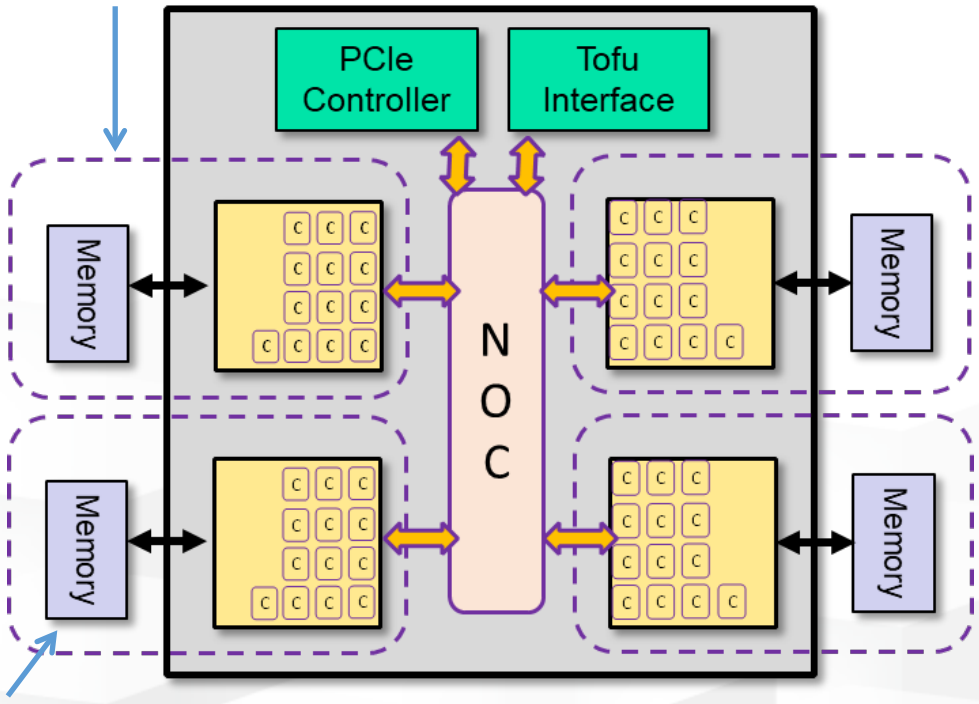
- **1. Extreme Power-Efficient System**
  - Maximum performance under Power consumption of 30 - 40MW (for system)
  - Approx. 15 GF/W (dgemm) confirmed by the prototype CPU => 1st in Green 500 !!!
- **2. Effective performance of target applications**
  - It is expected to exceed 100 times higher than the K computer's performance in some applications
  - 125 times faster in GENESIS (MD application), 120 times faster in NICAM+LETKF (climate simulation and data assimilation) were estimated
- **3. Ease-of-use system for wide-range of users**
  - Co-design with application developers
  - Shared memory system with high-bandwidth on-package memory must make existing OpenMP-MPI program ported easily.
  - No programming effort for accelerators such as GPUs is required.

CPU Architecture: A64FX

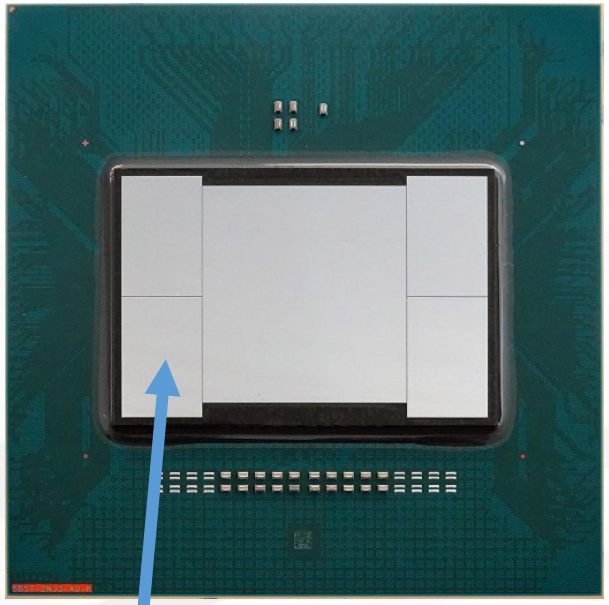
- **Armv8.2-A (AArch64 only) + SVE (Scalable Vector Extension)**
  - FP64/FP32/FP16 (https://developer.arm.com/products/architecture/a-profile/docs)
- **SVE 512-bit wide SIMD**
- **# of Cores: 48 + (2/4 for OS)**
- Co-design with application developers and high memory bandwidth utilizing on-package stacked memory: **HBM2(32GiB)**
- Leading-edge Si-technology (7nm FinFET), low power logic design **(approx. 15 GF/W (dgemm))**, and power-controlling knobs
- Clock frequency: 2.0 GHz(normal), 2.2 GHz (boost)
- Peak performance
  - 3.0 TFLOPS@2GHz (>90% @ dgemm)
  - Memory B/W 1024GB/s (>80% stream)
  - Byte per Flops: 0.33

- "Common" programing model will be to run each MPI process on a NUMA node (CMG) with OpenMP-MPI hybrid programming.
- 48 threads OpenMP is also supported.

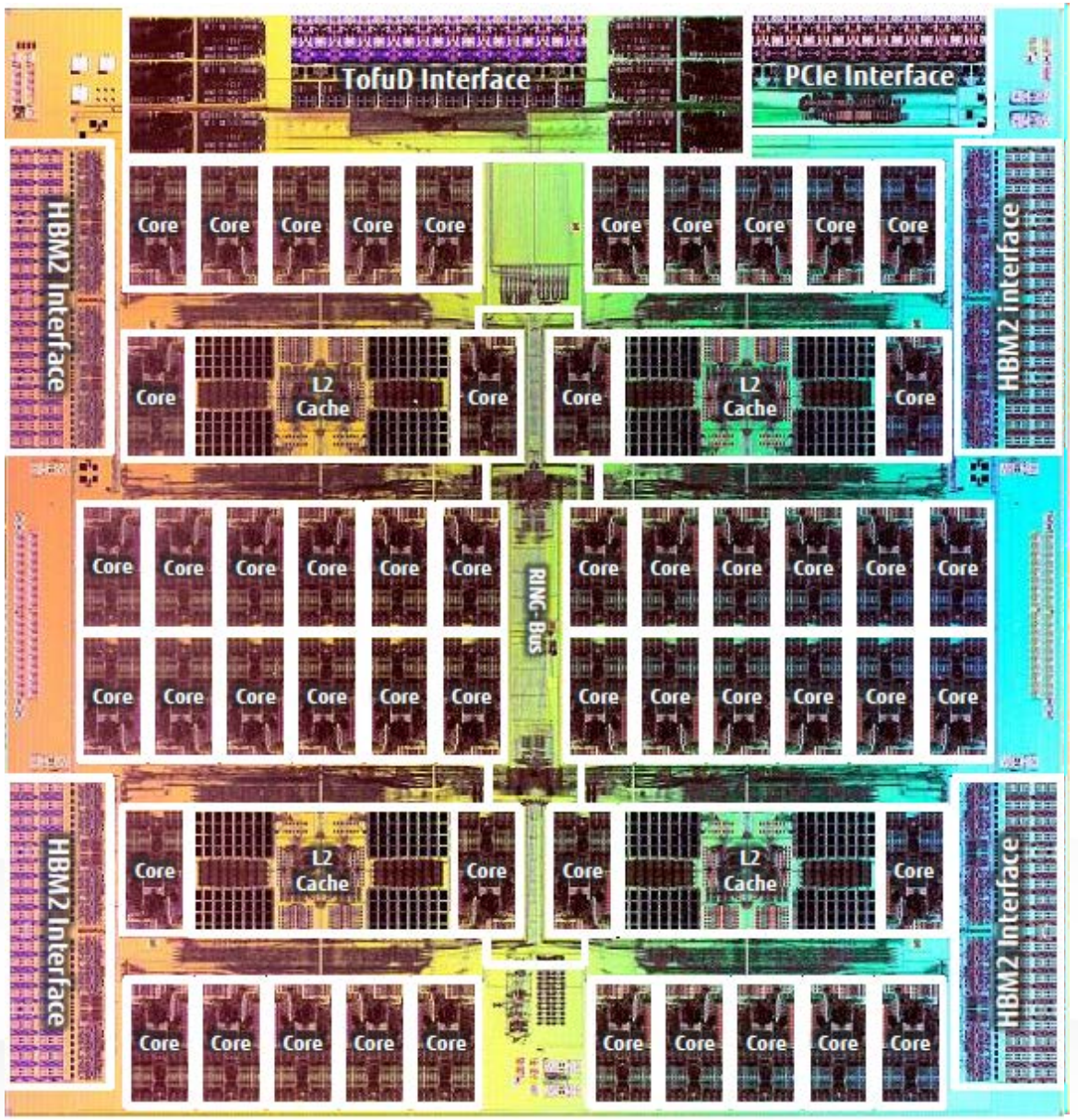
CMG(Core-Memory-Group): NUMA node
12+1 core



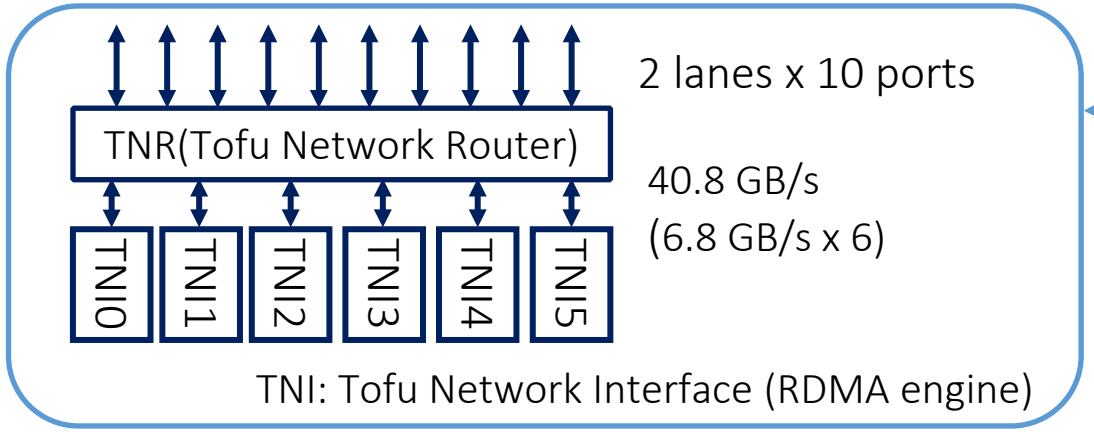
- **TSMC 7nm FinFET**
- **CoWoS technologies for HBM2**

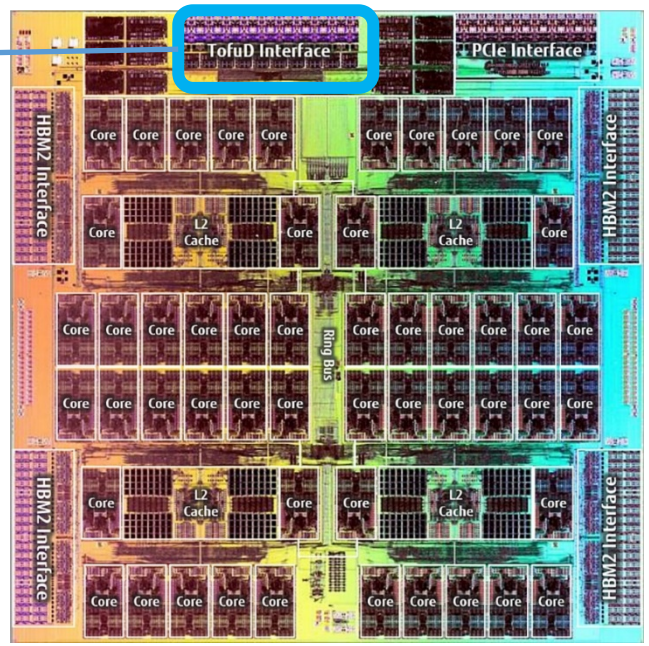


HBM2



TofuD Interconnect





- 6 RDMA Engines
- Hardware barrier support
- Network operation offloading capability

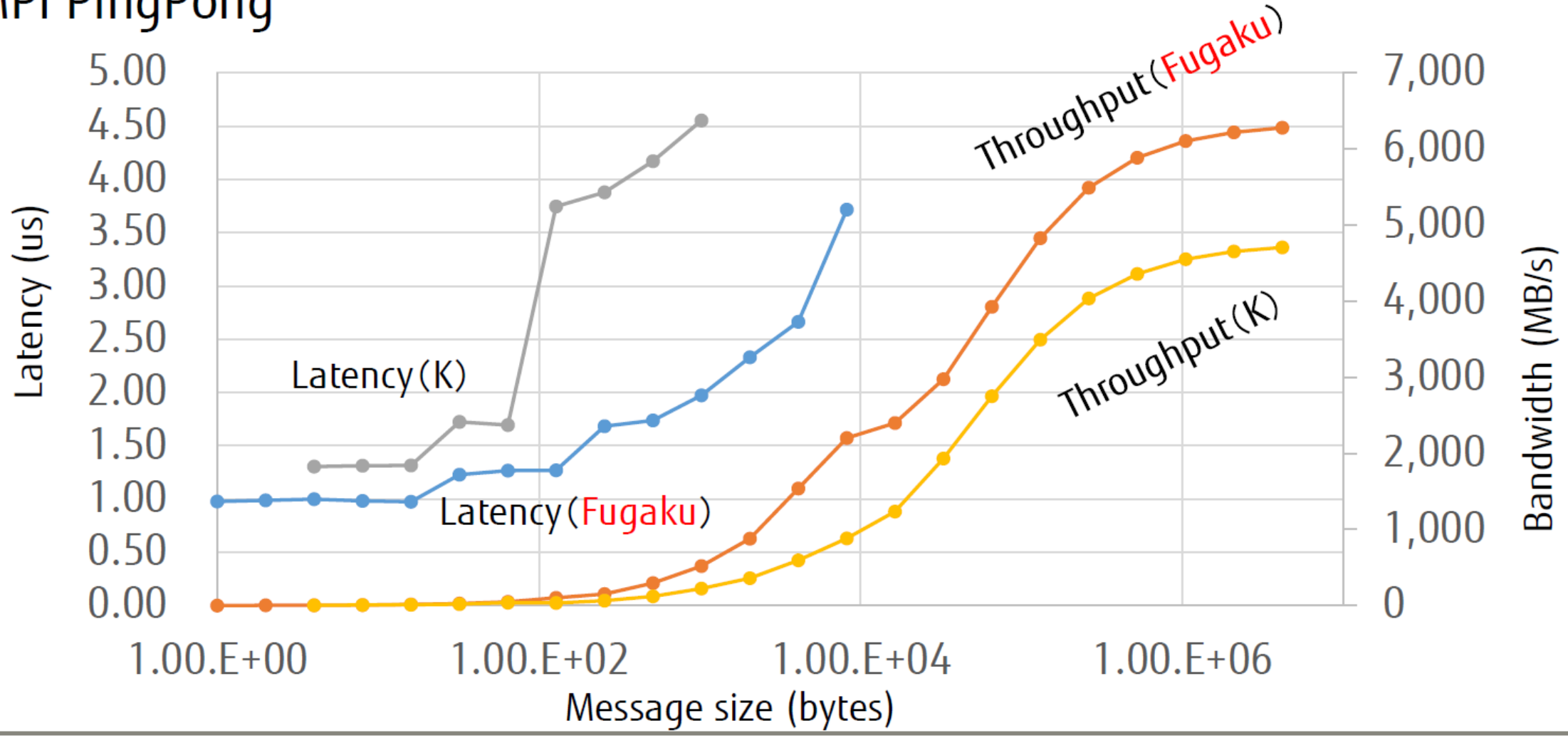
8B Put latency	0.49 – 0.54 usec
1MiB Put throughput	6.35 GB/s

rf. Yuichiro Ajima, et al. , “The Tofu Interconnect D,” IEEE Cluster 2018, 2018.

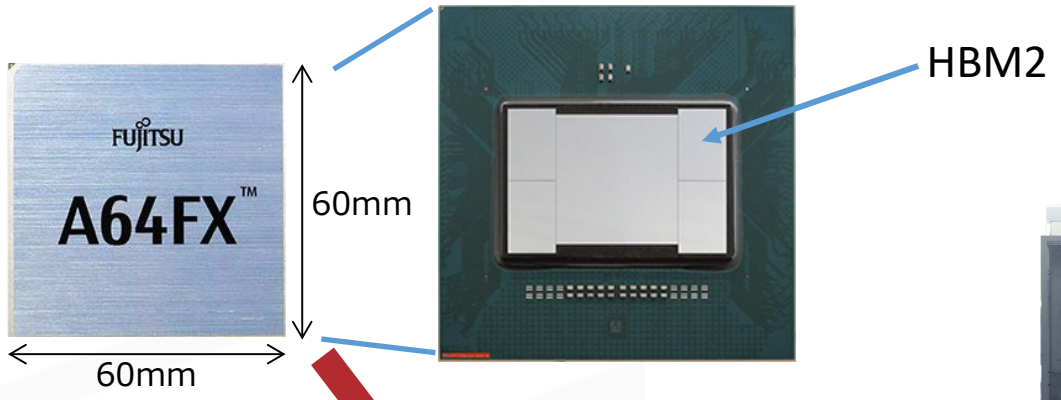
TofuD: MPI_Send/Receive Latency and BW

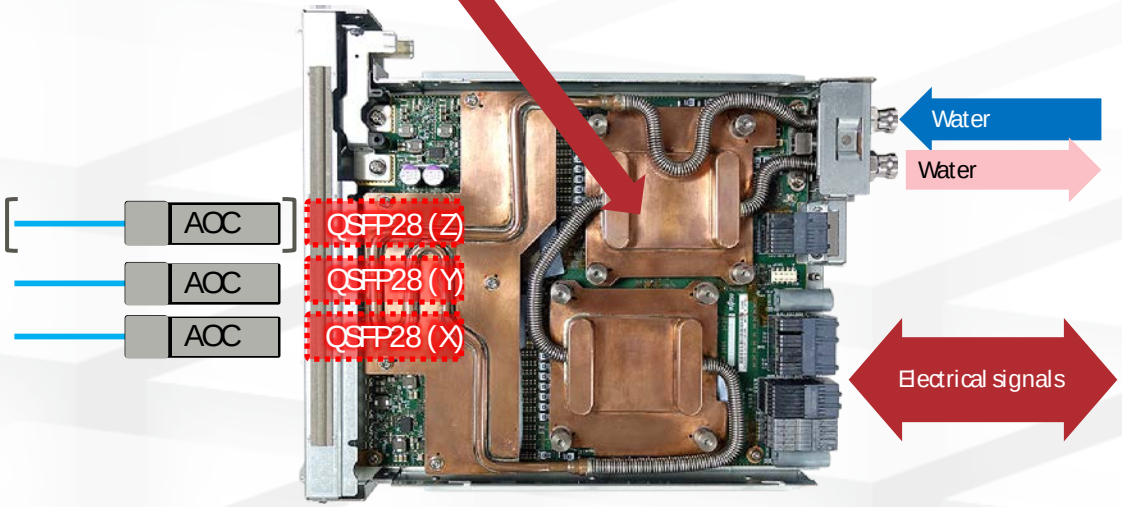
FUJITSU

MPI PingPong

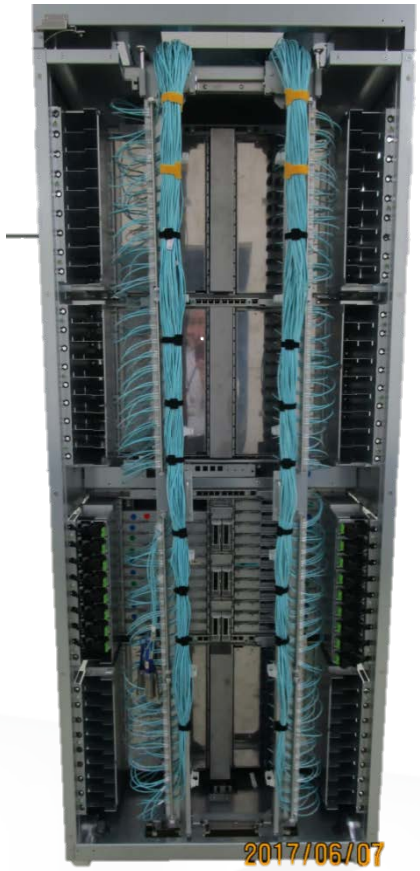


Fugaku prototype board and rack





2 CPU / CMU



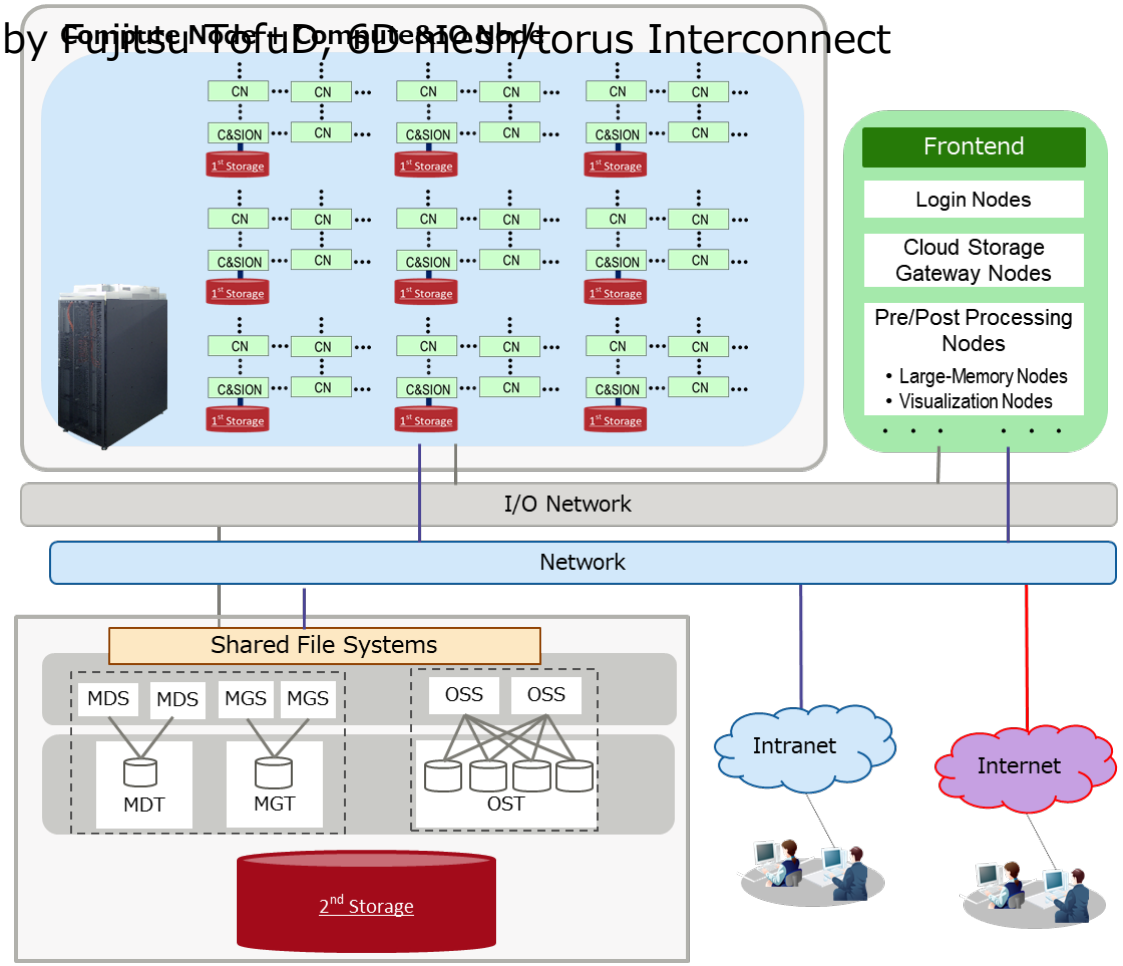
Shelf: 48 CPUs (24 CMU)
Rack: 8 shelves = 384 CPUs (8x48)

Fugaku System Configuration

- 158,976 node

Boost mode: 3.3792TF x 150k+ = 500+ PF

- **Two types of nodes**
- **3-level hierarchical storage system**
  - Compute Node and Compute & I/O Node connected by Fujitsu Tofu D, 6D mesh/torus Interconnect
  - 1st Layer
    - One of 16 compute nodes, called Compute & Storage I/O Node, has SSD about 1.6 TB
    - Services
      - Cache for global file system
      - Temporary file systems
        - Local file system for compute node
        - Shared file system for a job
  - 2nd Layer
    - Fujitsu FEFS: Lustre-based global file system
  - 3rd Layer
    - Cloud storage services



Advances from the K computer

	K computer	Fugaku	ratio
# core	8	48	
Si tech. (nm)	45	7	
Core perf. (GFLOPS)	16	64(70)	4(4.4)
Chip(node) perf. (TFLOPS)	0.128	3.072 (3.379)	24 (26.4)
Memory BW (GB/s)	64	1024	
B/F (Bytes/FLOP)	0.5	0.33	
#node / rack	96	384	4
#node/system	82,944	158,976	
System perf.(DP PFLOPS)	10.6	488 (537) 977(1070)	42.3(52.2) 84.6(104.4)

Si Tech

SVE

CMG&Si Tech

HBM

More than **7.6 M General-purpose cores**!

- SVE increases core performance
- Silicon tech. and scalable architecture (CMG) to increase node performance
- HBM enables high bandwidth

Value in blankets
Indicate the number
At boost mode (2.2GHz)

Benchmark Results on test chip A64FX

- **CloverLeaf (UK Mini-App Consortium), Fortran/C**
  - A hydrodynamics mini-app to solve the compressible Euler equations in 2D, using an explicit, second-order method
  - Stencil calculation

- **TeaLeaf (UK Mini-App Consortium), Fortran**
  - A mini-application to enable design-space explorations for iterative sparse linear solvers
  - https://github.com/UK-MAC/TeaLeaf_ref.git
  - Problem size: Benchmarks/tea_bm_5.in, end_step=10 -> 3

- **LULESH (LLNL), C**
  - Mini-app representative of simplified 3D Lagrangian hydrodynamics on an unstructured mesh, indirect memory access

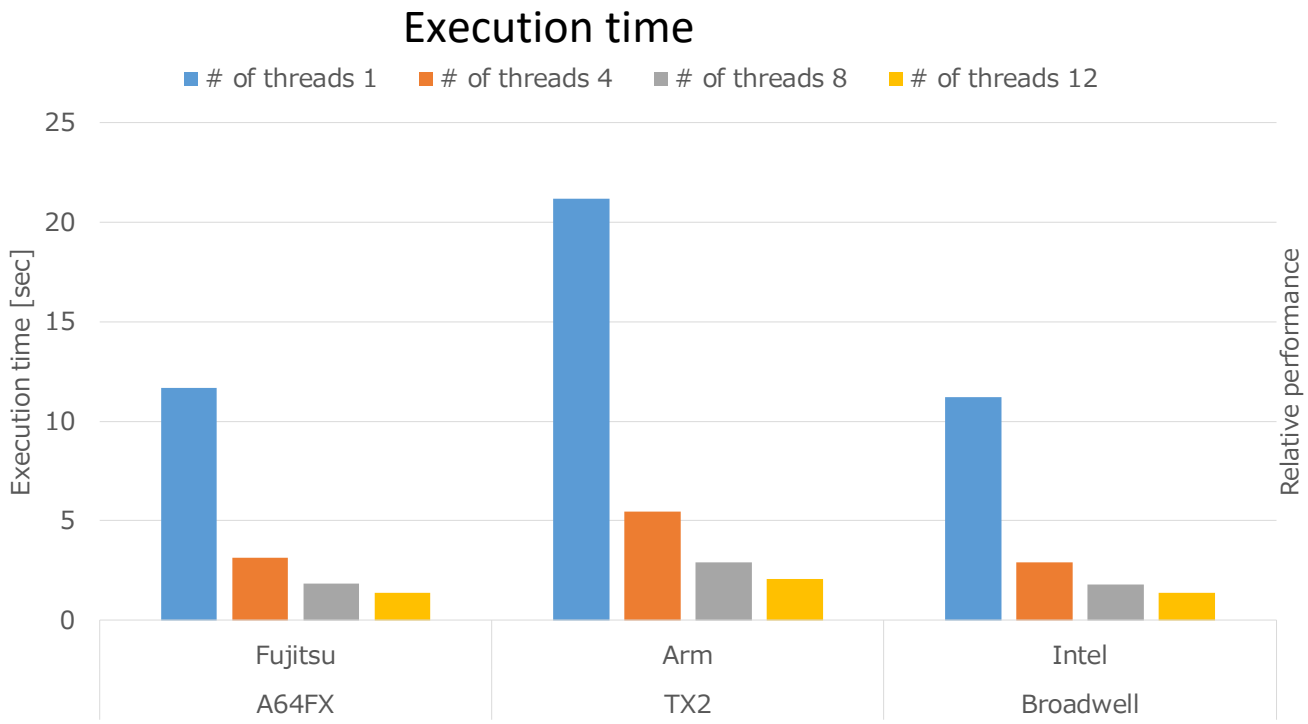
Benchmark Results on test chip A64FX

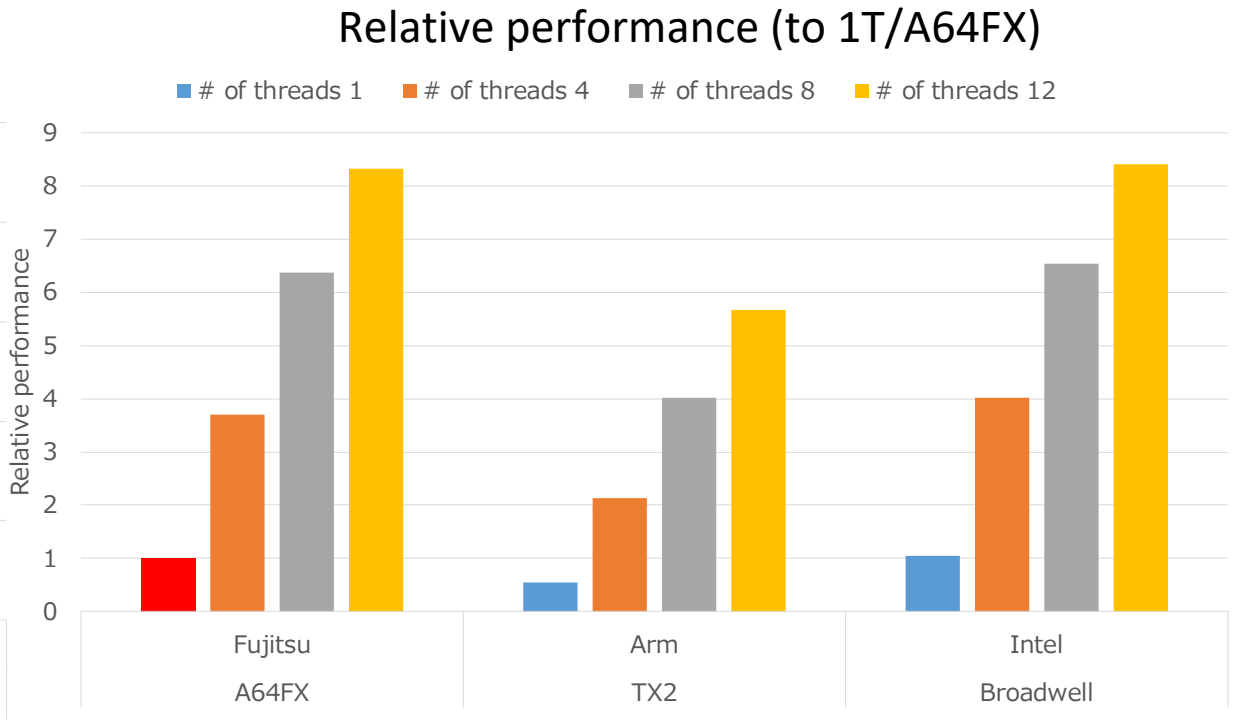
Disclaimer:
The software used for the evaluation, such as the compiler, is still under development and its performance may be different when the supercomputer Fugaku starts its operation.

- **Platform**
  - A64FX test chip (2.0 GHz)
  - ThunderX2 @ Apollo70
    - 28C/2S @ 2.0GHz
    - Arm HPC compiler 19.1
  - Broadwell (Xeon E5-2680 v4)
    - 14C/2S @ 2.4GHz
    - Intel compiler 2019.0.045
  - Skylake (Xeon Gold 6126) @ Cygnus, Univ. of Tsukuba
    - 12C/2S @ 2.6GHz
    - Intel compiler 19.0.3.199

- **Compiler Options**
  - Fujitsu compiler
    - -Kfast,openmp
  - Arm HPC compiler
    - -Ofast -march=armv8-a(+sve)
  - Intel compiler
    - -O3 -qopenmp -march=native

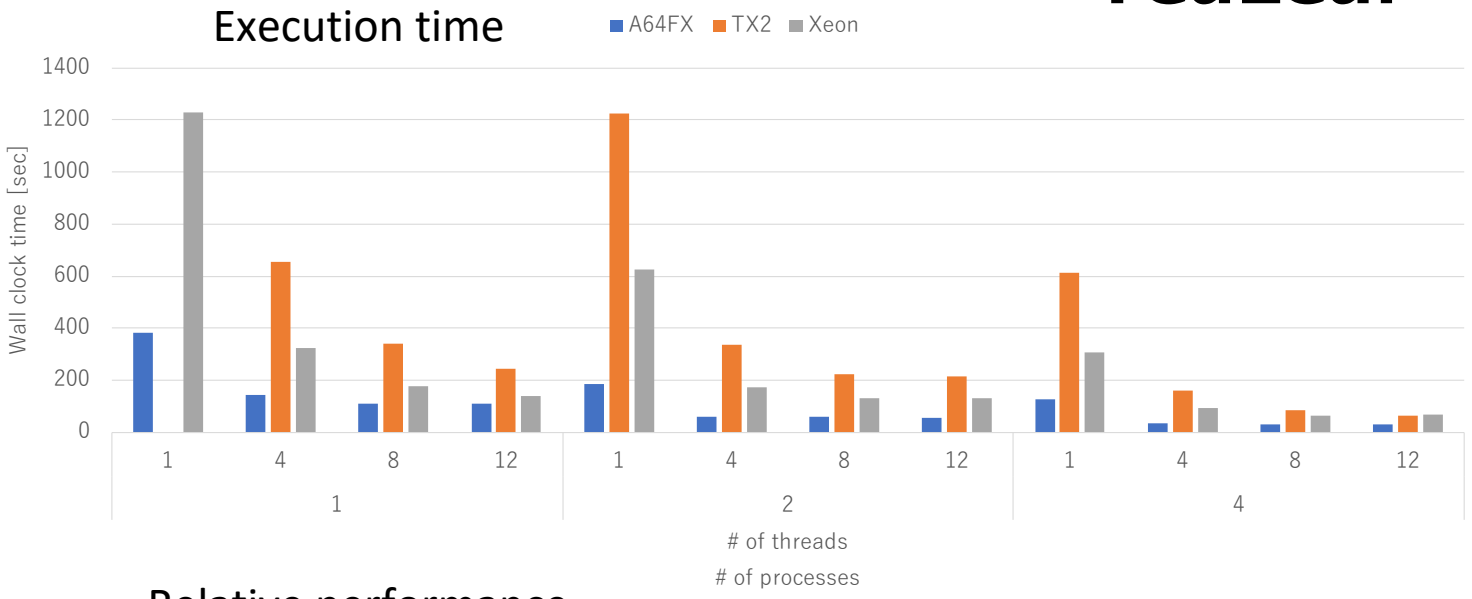
CloverLeaf

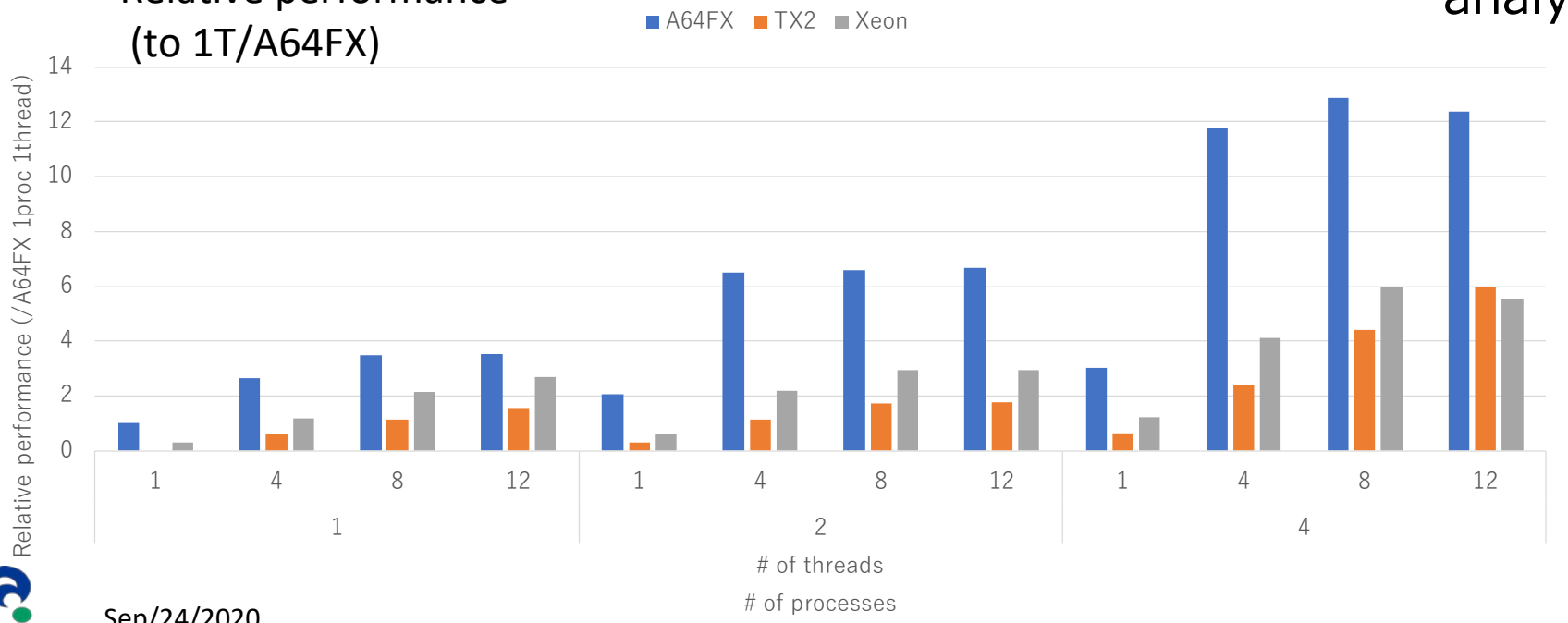




- Evaluation using one CMG(NUMA node) without MPI
- Good scalability by increasing the number of threads within CMG.
- One GMG performance is comparable to Intel one. (Chip contains 4 CMG!)

TeaLeaf

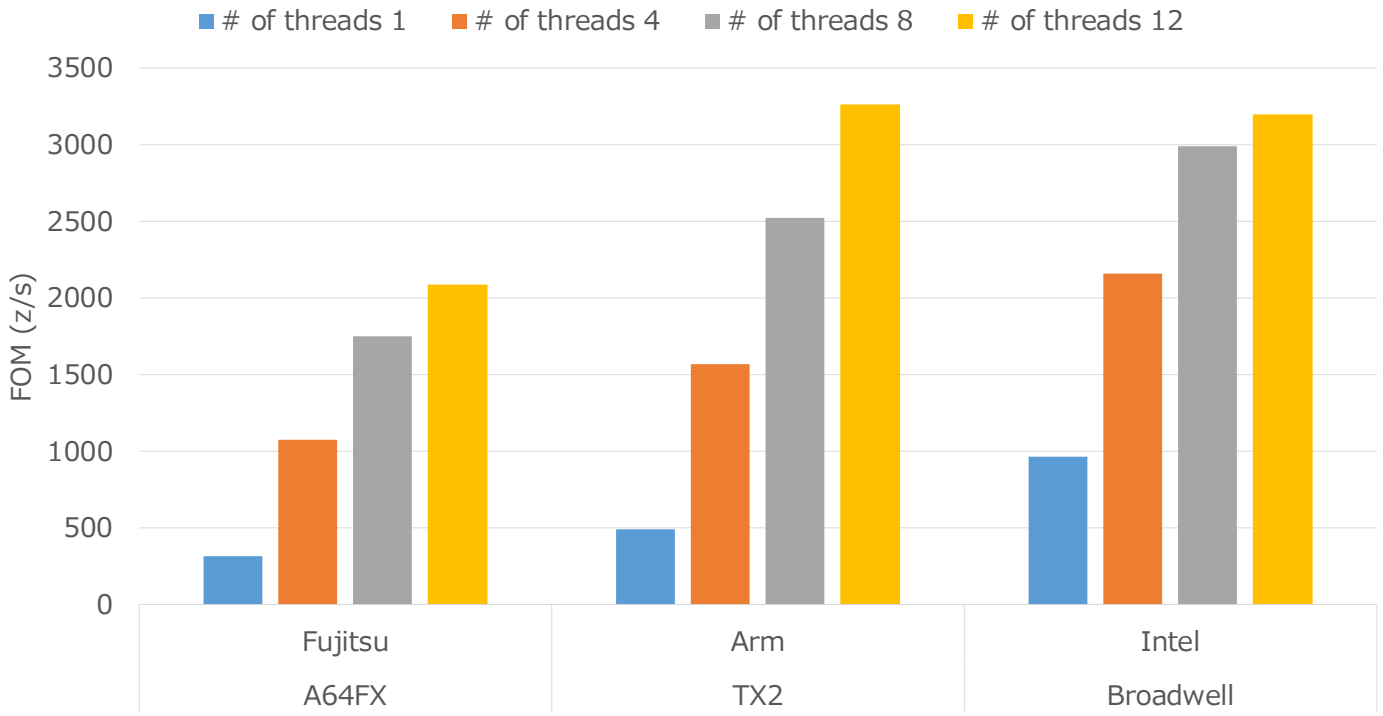




- Evaluation of MPI program within one chip (upto 4 MPI process)
- Changing #threads within CMG
- The speedup is limited for more than 4 threads due to the memory bandwidth (?)
- We need more performance analysis.

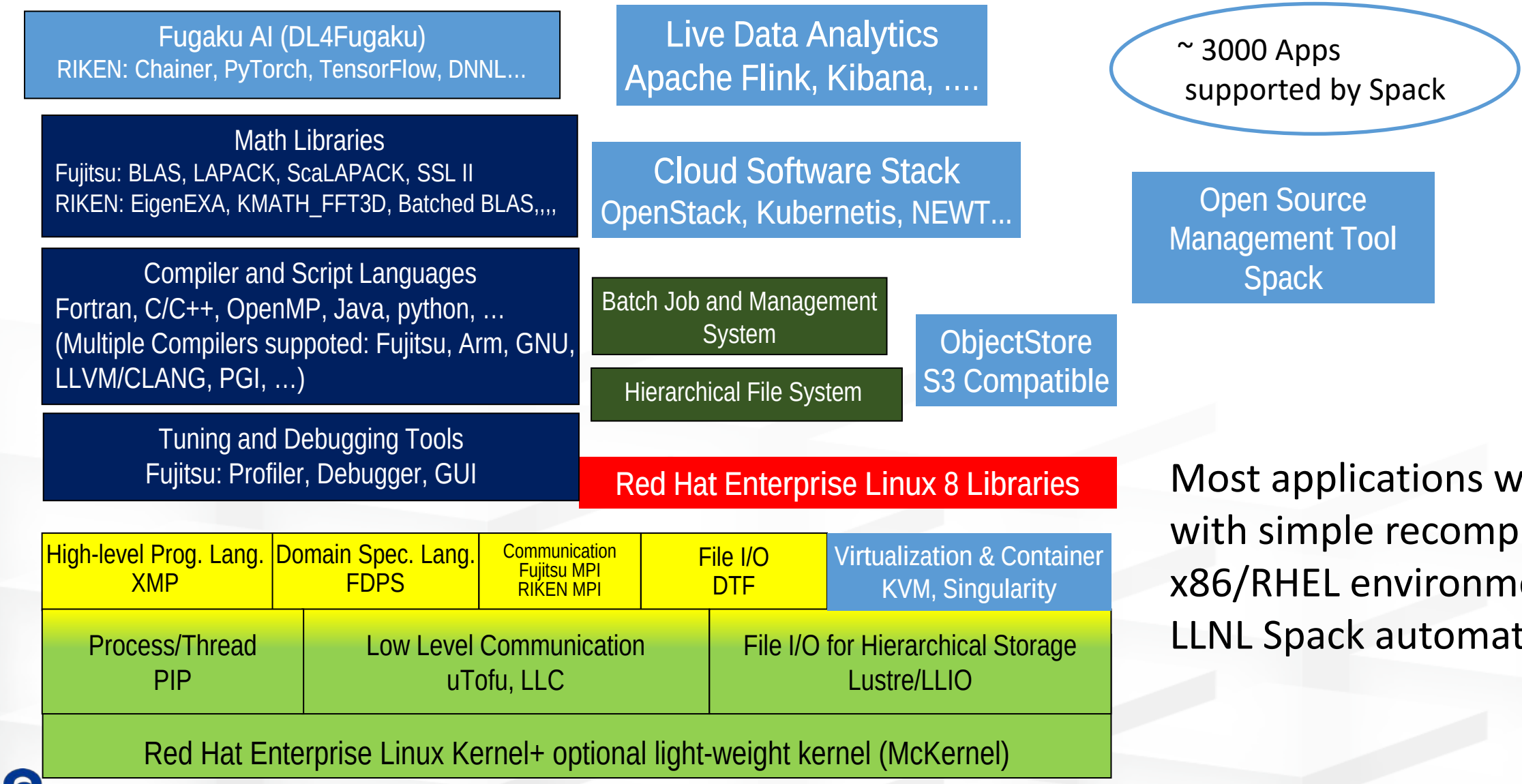
Xeon @ Cygnus, Univ. of Tsukuba
Intel Xeon Gold 6126
2.6GHz; 12 core x 2 socket

LULESH



- Evaluation using one CMG(NUMA node) without MPI
- One CMG performance is less than Thx2 and Intel one
- We found low vectorization (SIMD (SVE) instructions ratio is a few %)
- We need more code tuning for more vectorization using SIMD

Fugaku / Fujitsu FX1000 System Software Stack



Most applications will work with simple recompile from x86/RHEL environment.
LLNL Spack automates this.

OSS Application Porting @ Arm HPC Users Group

FUJITSU

(http://arm-hpc.gitlab.io/)

Application	Lang.	GCC	LLVM	Arm	Fujitsu
LAMMPS	C++	Modified	Modified	Modified	Modified
GROMACS	C	Modified	Modified	Modified	Modified
GAMESS*	Fortran	Modified	Modified	Modified	Modified
OpenFOAM	C++	Modified	Modified	Modified	Modified
NAMD	C++	Modified	Modified	Modified	Modified
WRF	Fortran	Modified	Modified	Modified	Modified
Quantum ESPRESSO	Fortran	Ok in as is	Ok in as is	Ok in as is	Modified
NWChem	Fortran	Ok in as is	Modified	Modified	Modified
ABINIT	Fortran	Modified	Modified	Modified	Modified
CP2K	Fortran	Ok in as is	Issues found	Issues found	Modified
NEST*	C++	Ok in as is	Modified	Modified	Modified
BLAST*	C++	Ok in as is	Modified	Modified	Modified

Low-power Design & Power Management

- **7nm FinFET (TSMC) with low-power logic design**
- **A64FX provides power management function called "Power Knob"**
  - FL pipeline usage: FLA only, EX pipeline usage : EXA only, Frequency reduction …
  - User program can change "Power Knob" for power optimization
  - "Energy monitor" facility enables chip-level power monitoring and detailed power analysis of applications
- **"Eco-mode" : FLA only with lower "stand-by" power for ALUs**
  - Reduce the power-consumption for memory intensive apps.
  - 4 apps out of 9 target applications select "eco-mode" for the max performance under the limitation of our power capacity (Even using HBM2!)
- **Retention mode: power state for de-activation of CPU with keeping network alive**
  - Large reduction of system power-consumption at idle time
- **"Power Knobs" can be controlled by Sandia PowerAPIs and setting running modes.**
  - We are now designing the accounting system to give incentive to make use of power-knobs
  - "Power budget" as well as node-hour budget.

System software and Programming models & languages for “Fugaku”

- Standard programming model is OpenMP (for NUMA node(CMG)) + MPI
  - Both OpenMPI (by Fujitsu) and MPICH (by Riken) are supported.
  - OpenMP 4.x is supported by Fujitsu compiler. LLVM-based compiler and gcc available.
  - uTofu low-level comm. Layer for Tofu-D interconnect.
- Container and Virtual machine (KVM, Singularity, …)
- DL4Fugaku: AI framework for Fugaku, used in Chainer, PyTorch, TensorFlow
- Many Open-source software will be ported using Spack

- System software and Programming tools, Math-Libs developed by RIKEN
  - McKernel: Light-weight Kernel enabling jitter-less environment for large-scale parallel program execution.
  - XcalableMP directive-based PGAS Language
  - FDPS: DLS for Framework for Developing Particle Simulators.
  - EigenExa: Eigen-value math library for large-scale parallel systems.

XcalableMP(XMP) http://www.xcalablemp.org

- **What's XcalableMP (XMP for short)?**
  - A PGAS programming model and language for distributed memory , proposed by **XMP Spec WG**
  - XMP Spec WG is a special interest group to design and draft the specification of XcalableMP language. It is now organized under **PC Cluster Consortium**, Japan. Mainly active in Japan, but open for everybody.

- **Project status (as of June 2019)**
  - XMP Spec **Version 1.4** is available at XMP site. new features: mixed OpenMP and OpenACC , libraries for collective communications.
  - Reference implementation by U. Tsukuba and Riken AICS: **Version 1.3.1 (C and Fortran90)** is available for PC clusters, Cray XT and K computer. Source-to- Source compiler to code with the runtime on top of MPI and GasNet.
- **HPCC class 2 Winner 2013. 2014**

The spec of XcalableMP 1.x is now converged.
We are now moving to XcalableMP 2.0 with global task-based parallel programming and PGAS

- **Language Features**
  - Directive-based language extensions for Fortran and C for PGAS model
  - Global view programming with global-view distributed data structures for data parallelism
    - SPMD execution model as MPI
    - pragmas for data distribution of global array.
    - Work mapping constructs to map works and iteration with affinity to data explicitly.
    - Rich communication and sync directives such as "gmove" and "shadow".
    - Many concepts are inherited from HPF
  - Co-array feature of CAF is adopted as a part of the language spec for local view programming (also defined in C).

Code example

```
int array[YMAX][XMAX];

#pragma xmp nodes p(4)
#pragma xmp template t(YMAX)
#pragma xmp distribute t(block) on p
#pragma xmp align array[i][*] to t(i)

main(){
  int i, j, res;
  res = 0;

#pragma xmp loop on t(i)  reduction(+:res)
  for(i = 0; i < 10; i++)
    for(j = 0; j < 10; j++){
      array[i][j] = func(i, j);
      res += array[i][j];
    }
}
```

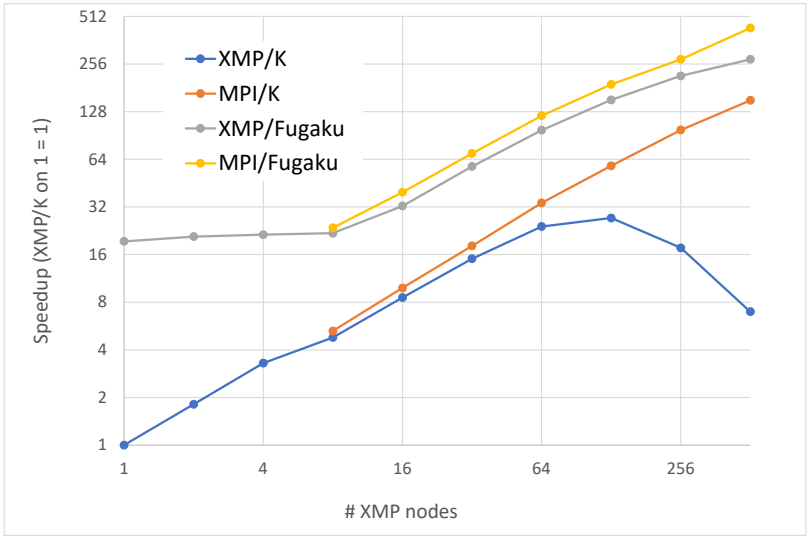
data distribution

add to the serial code : incremental parallelization

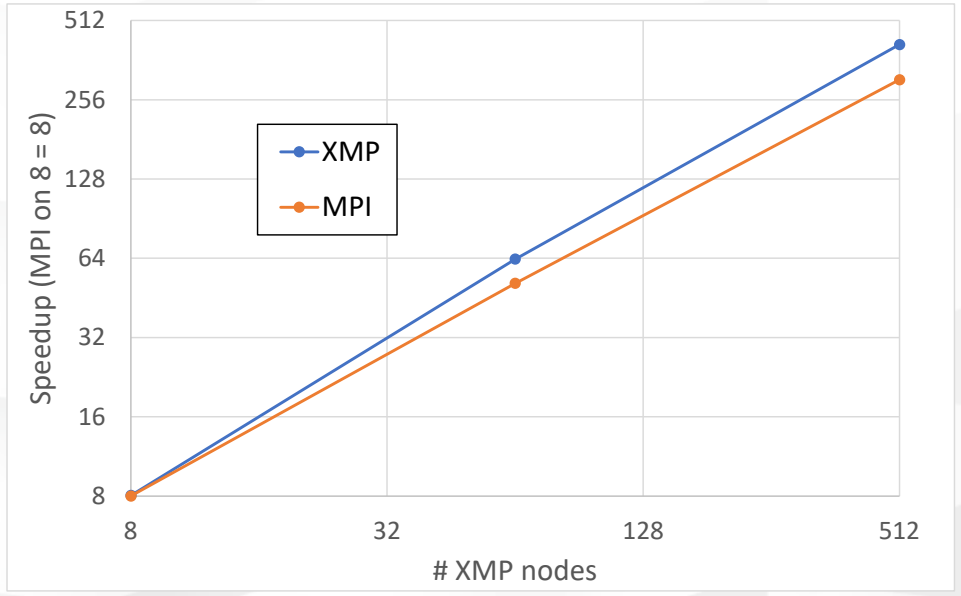
work sharing and data synchronization

Performance of XcalableMP on Fugaku

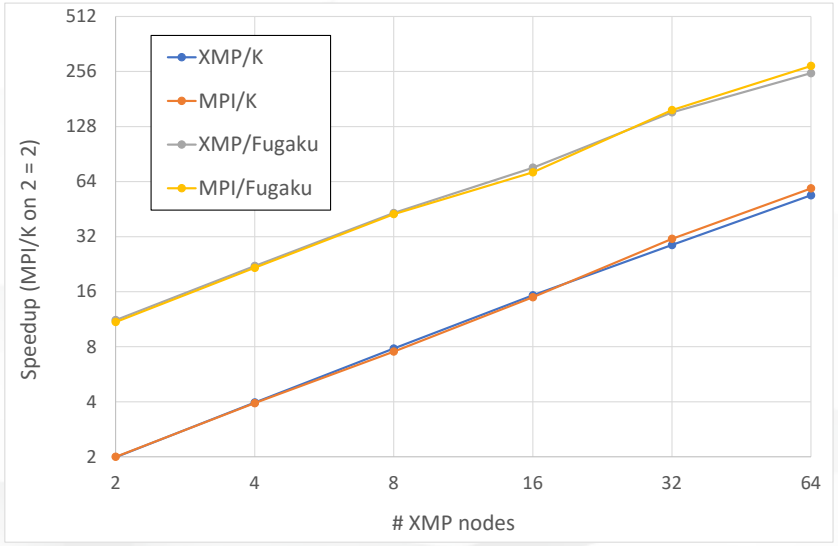
- XcalableMP was taken as a parallel programming language project for improving the productivity and performance of parallel programing.
- XcalableMP is now available on Fugaku and the performance is enhanced by the Fugaku interconnect, Tofu-D.



QCD (Local view programming, Coarray)



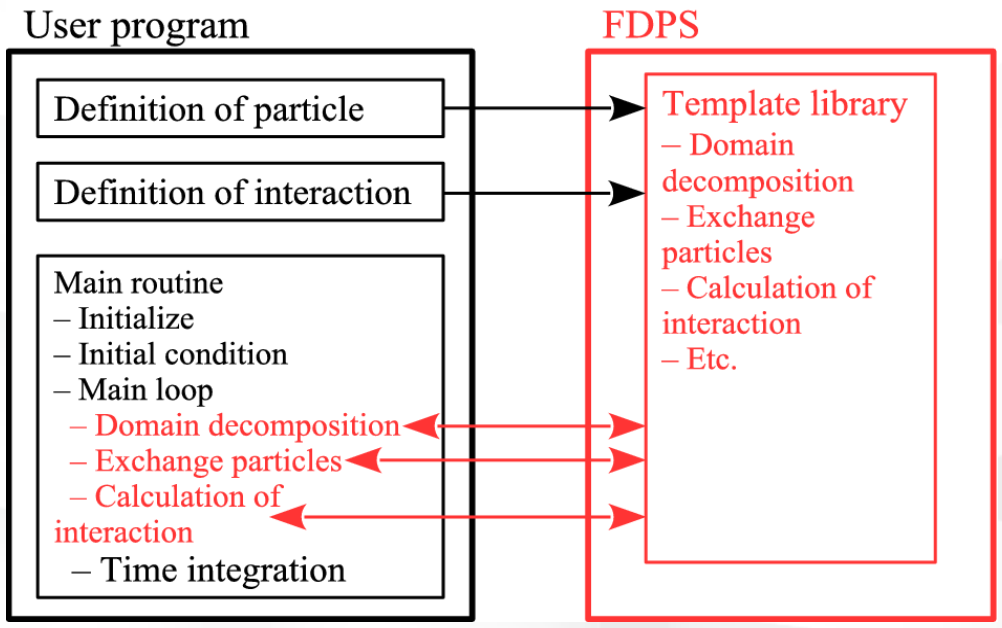
Impact-3D (global view, stencil apps)
Fusion simulation code



NT-Chem (local view programming, Coarray)

FDPS: a framework for developing parallel particle simulation codes

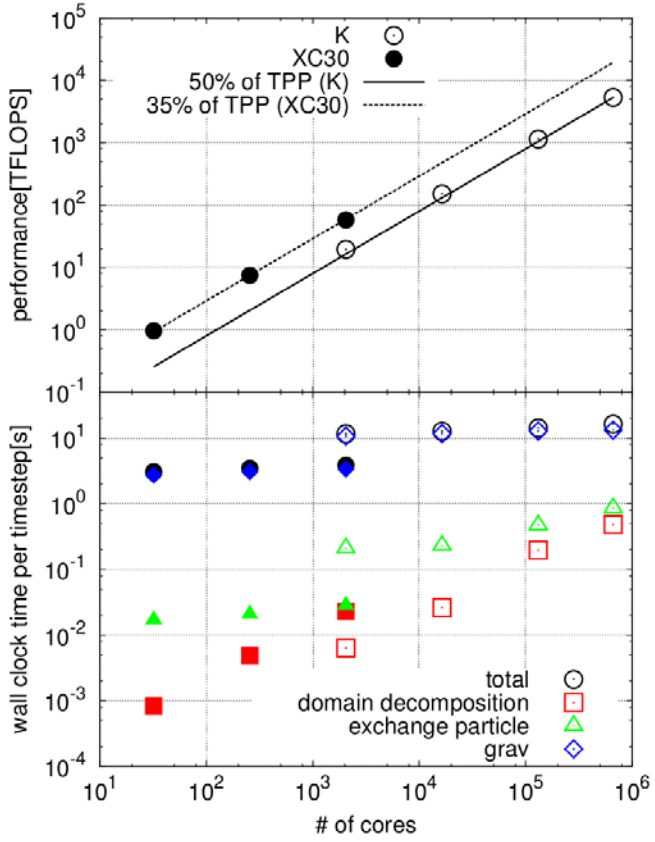
- Developed by Prof. Makino's group, R-CCS
- Basic idea: "abstract" code for
  - domain decomposition
  - particle exchange
  - parallel O(N log N) interaction calculation



Basic concept of FDPS. The user program gives the definitions of particle and interaction to FDPS, and calls ...

- Implemented as a template class library in C++.
- A single program can run on a notebook, a cluster of Intel servers, and the entire K computer, without change (Well, interaction function needs some optimization)
- Works also on GPGPUs

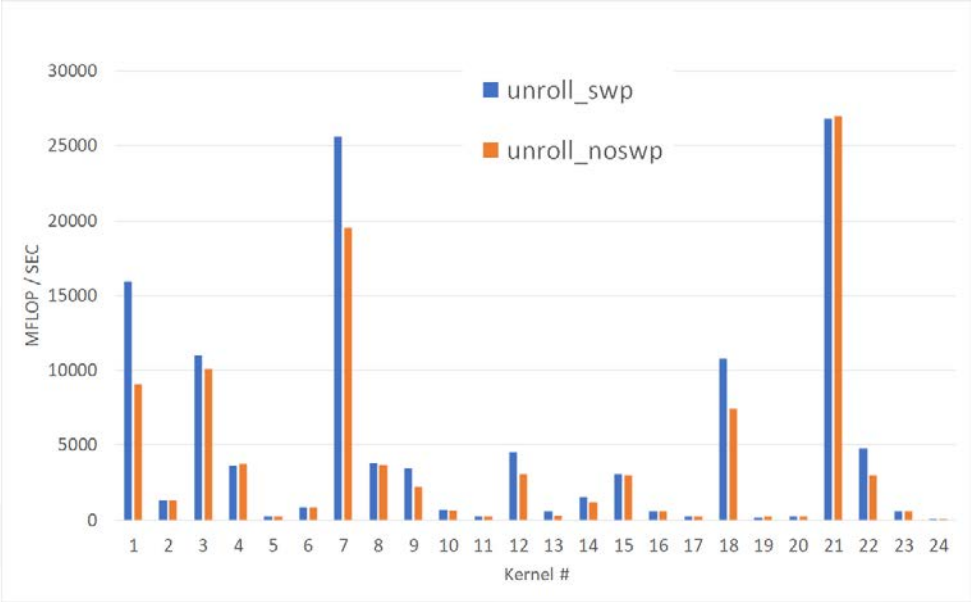
Gravitational N-body (270k/process) Weak scaling performance pretty good for up to all nodes of K computer



Challenges of programming for Fugaku

- **Task-based programming models for "Fugaku"**
  - to exploit parallelism of SIMD and manycore for A64FX, and enables overlapping comp. and comm.
  - OpenMP 4.0 task + MPI (Multithread-aware MPI)
  - XcalableMP 2.0 is being designed for task-based programming on global address space (PGAS)

- **How to exploit SIMD**
  - SIMD is a key for performance on A64FX
  - OpenMP SIMD directives
  - Compiler optimization (Fujitsu compiler)
    - SWP: software pipelining, loop fission, …
  - OpenCL for SVE (Arm SIMD)
- **Comm Optimization by Low-level layer, uTofu.**



Performance improvement by SWP in Livermore Kernels by Fujitsu compiler

Programming models for beyond “Fugaku”

- **Programming support for accelerator-based heterogenous parallel system (incl. FPGA Clusters)**
  - (Fugaku has no accelerators.)
  - XcalableACC: integration of XcalableMP and OpenACC
  - Task-based offloading to accelerators

Concluding Remarks

- **We are interested in fostering eco-system about HPC Arm**
  - A64FX is only a processor supporting SVE!
- **International Collaborations are welcome**
  - DOE-MEXT collaborations
    - Arm Arch collaboration (SNL/NNSA, U. Bristol)
    - Spack for Fugaku (LLNL, going-on)
    - ECP software porting & evaluation (planned)
  - CEA@France, A*STAR@Singapore, U Oregon, …

- **I am interested in an accounting system to give incentives to make use of "Power-knobs".**
  - "Power budget" as well as node-hour budget, to promote power-saving.