Development of system software in post K supercomputer

Post-K and post-T2K project

Yutaka Ishikawa, Project Leader

Mitsuhisa Sato Team Leader of Architecture Development Team

FLAGSHIP 2020 project
RIKEN Advance Institute of Computational Science (AICS)

IWOMP2016, 6th October, 2016

Outline of Talk

- **Post T2K project (Installation of Oakforest-PACS)**

  Slide courtesy of Prof. Taisuke Boku, CCS, University of Tsukuba

- **An Overview of FLAGSHIP 2020 project**
- **An Overview of post K system**
- **System Software**
- **Concluding Remarks**

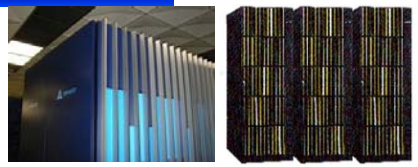
AICS and Supercomputer Centers in Japanese Universities

AICS

AICS, RIK
K comput , 4PB)
Available in 2012

Post K

Hokkaido Univ. :
SR11000/K1(5.4Tflops, 5TB)
PC Cluster (0.5Tflops, 0.64TB)



Kyoto Univ.
T2K Open Supercomputer
(61.2 Tflops, 13 TB)

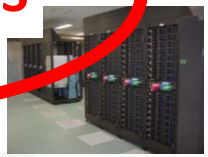
Tohoku Univ. :
NEC SX-9(29.4Tflops, 18TB)
NEC Express5800 (1.74Tflops, 3TB)

Osaka Univ. :
SX-9 (16Tflops, 10TB)
SX-8R (5.3Tflops, 3.3TB)
PCCluster (23.3Tflops, 2.9TB)

Univ. of Tsukuba :
T2K Open Supercomputer
95.4Tflops, 20TB

Oakforest-PACS

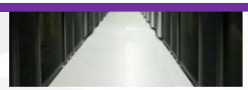
Univ. of Tokyo :
T2K Open Supercomputer
(140 Tflops, 31.25TB)



Kyushu Univ. :
PC Cluster (55Tflops, 18.8TB)
SR16000 L2 (25.3Tflops, 5.5TB)
PC Cluster (18.4Tflops, 3TB)

Nagoya Univ. :
FX1(30.72Tflops, 24TB)
HX600(25.6Tflops, 10TB)
M9000(3.84Tflops, 3TB)

Tokyo Insti
Tsubame 2
(2.4 Pflops



25 PF KNL-based System
It Will be the fastest system in Nov. 2016 in Japan

RIKEN

JCAHPC and Oakforest-PACS (a.k.a Post-T2K project)

JCAHPC

- **Joint Center for Advanced High Performance Computing (http://jcahpc.jp)**
  - Organization for Post T2K project

- **March 2013: U. Tsukuba and U. Tokyo exchanged agreement for "JCAHPC establishment and operation"**
  - Center for Computational Sciences, University of Tsukuba and Information Technology Center, University of Tokyo
- **April 2013: JCAHPC started**
  - 1st period director: Mitsuhisa Sato (Tsukuba), vice director: Yutaka Ishikawa (Tokyo)
  - 2nd period (2016~) director: Hiroshi Nakamura (Tokyo), vice director: Masayuki Umemura (Tsukuba)
- **July 2013: RFI for procurement**
  - at this time, the joint procurement style was not fixed
    -> then a single system procurement was decided
  - to give enough time for very advanced technology for processor, network, memory, etc., more than 1 year of period was taken to fix the specification
- **It is the first trial to introduce a shared single supercomputer system by multiple national universities in Japan !**

Specification of Oakforest-PACS

JCAHPC

Total peak performance			25 PFLOPS
Total number of compute nodes			8,208
Compute node	Product		Fujitsu Next-generation PRIMERGY server for HPC (under development)
	Processor		Intel® Xeon Phi™ （Knights Landing） Xeon Phi 7250 (1.4GHz TDP) with 68 cores
	Memory	High BW	16 GB, > 400 GB/sec (MCDRAM, effective rate)
		Low BW	96 GB, 115.2 GB/sec (DDR4-2400 x 6ch, peak rate)
Inter-connect	Product		Intel® Omni-Path Architecture
	Link speed		100 Gbps
	Topology		Fat-tree with full-bisection bandwidth
Login node	Product		Fujitsu PRIMERGY RX2530 M2 server
	# of servers		20
	Processor		Intel Xeon E5-2690v4 (2.6 GHz 14 core x 2 socket)
	Memory		256 GB, 153 GB/sec (DDR4-2400 x 4ch x 2 socket)

Full bisection bandwidth Fat-tree by Intel® Omni-Path Architecture

JCAHPC

12 of
768 port Director Switch
(Source by Intel)

362 of
48 port Edge Switch

Firstly, to reduce switches&cables, we considered :
- All the nodes into subgroups are connected with FBB Fat-tree
- Subgroups are connected with each other with >20% of FBB

But, HW quantity is not so different from globally FBB, and globally FBB is preferred for flexible job management.

Compute Nodes	8208
Login Nodes	20
Parallel FS	64
IME	300
Mgmt, etc.	8
Total	8600

(pre) Photo of computation node

JCAHPC

Computation node (Fujitsu next generation PRIMERGY)
with single chip Intel Xeon Phi (Knights Landing, 3+TFLOPS)
and Intel Omni-Path Architecture card (100Gbps)

Chassis with 8 nodes, 2U size

Specification of Oakforest-PACS (I/O)

JCAHPC

Parallel File System		Type	Lustre File System
		Total Capacity	26.2 PB
	Meta data	Product	DataDirect Networks MDS server + SFA7700X
		# of MDS	4 servers x 3 set
		MDT	7.7 TB (SAS SSD) x 3 set
	Object storage	Product	DataDirect Networks SFA14KE
		# of OSS (Nodes)	10 (20)
		Aggregate BW	~ 500 GB/sec
Fast File Cache System		Type	Burst Buffer, Infinite Memory Engine (by DDN)
		Total capacity	940 TB (NVMe SSD, including parity data by erasure coding)
		Product	DataDirect Networks IME14K
		# of servers (Nodes)	25 (50)
		Aggregate BW	~1,560 GB/sec

Software of Oakforest-PACS

JCAHPC

	Compute node	Login node
OS	CentOS 7, McKernel	Red Hat Enterprise Linux 7
Compiler	gcc, Intel compiler (C, C++, Fortran)	
MPI	Intel MPI, MVAPICH2	
Library	Intel MKL LAPACK, FFTW, SuperLU, PETSc, METIS, Scotch, ScaLAPACK, GNU Scientific Library, NetCDF, Parallel netCDF, Xabclib, ppOpen-HPC, ppOpen-AT, MassiveThreads	
Application	mpijava, XcalableMP, OpenFOAM, ABINIT-MP, PHASE system, FrontFlow/blue, FrontISTR, REVOCAP, OpenMX, xTAPP, AkaiKKR, MODYLAS, ALPS, feram, GROMACS, BLAST, R packages, Bioconductor, BioPerl, BioRuby	
Distributed FS		Globus Toolkit, Gfarm
Job Scheduler	Fujitsu Technical Computing Suite	
Debugger	Allinea DDT	
Profiler	Intel VTune Amplifier, Trace Analyzer & Collector	

McKernel support

- **McKernel (A light weight kernel for Many-Core architecture)**
  - developed at U. Tokyo and now at AICS, RIKEN (lead by Y. Ishikawa)
  - KNL-ready version is almost completed
  - It can be loaded as a kernel module to Linux
  - Batch scheduler is noticed to use McKernel by user's script, then apply it
  - Detach the McKernel module after job execution

Memory Model (on trial now)

JCAHPC

- **Our trial – dynamic switching of CACHE and FLAT modes**
  - Initial: nodes in the system are configured with a certain ratio of mixture (half and half) of Cache and Flat modes
  - Batch scheduler is noticed about the memory configuration from user's script
  - Batch scheduler tries to find appropriate nodes without reconfiguration
  - If there are not enough number of nodes, some of them are rebooted with another memory configuration
  - Reboot is by warm-reboot, not to take so long time (maybe)
  - Size limitation (max. # of nodes) may be applied

- **NUMA model**
  - ??? (maybe quadrant mode only)
  - (perhaps) we will not dynamically change it ??

Schedule

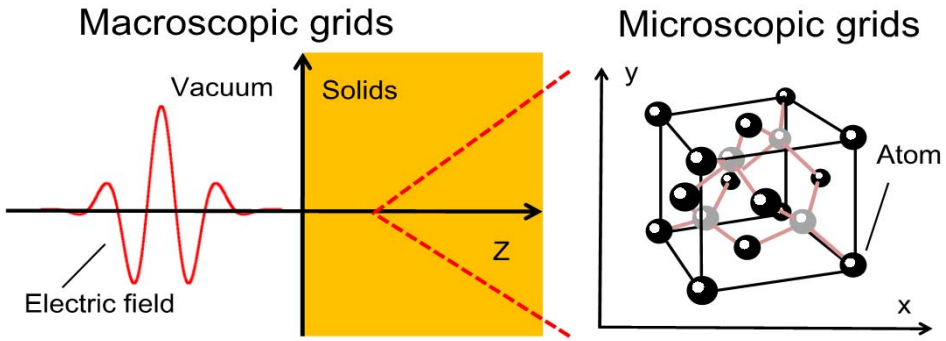
- 2013/7 RFI
- 2015/1 RFC
- 2016/1 RFP
- 2016/3/30 Proposal deadline
- 2016/4/20 Bid opening
- 2016/10/1 1st step system operation (~410 nodes)
- 2016/12/1 2nd step, full system operation
- 2017/4 National open use starts including HPCI
- 2022/3 System shutdown (planned)

Xeon Phi tuning on ARTED (with Yuta Hirokawa under collaboration with Prof. Kazuhiro Yabana, CCS)

JCAHPC

courtesy of Prof. BUni

- **ARTED – Ab-initio Real-Time Electron Dynamics simulator**
- **Multi-scale simulator based on RTRSDFT (Real-Time Real-Space Density Functional Theory) developed in CCS, U. Tsukuba to be used for Electron Dynamics Simulation**
  - RSDFT : basic status of electron (no movement of electron)
  - RTRSDFT : electron state under external force
- **In RTRSDFT, RSDFT is used for ground state**
  - RSDFT : large scale simulation with 1000～10000 atoms (ex. K-Computer)
  - RTRSDFT : calculate a number of unit-cells with 10 ~ 100 atoms



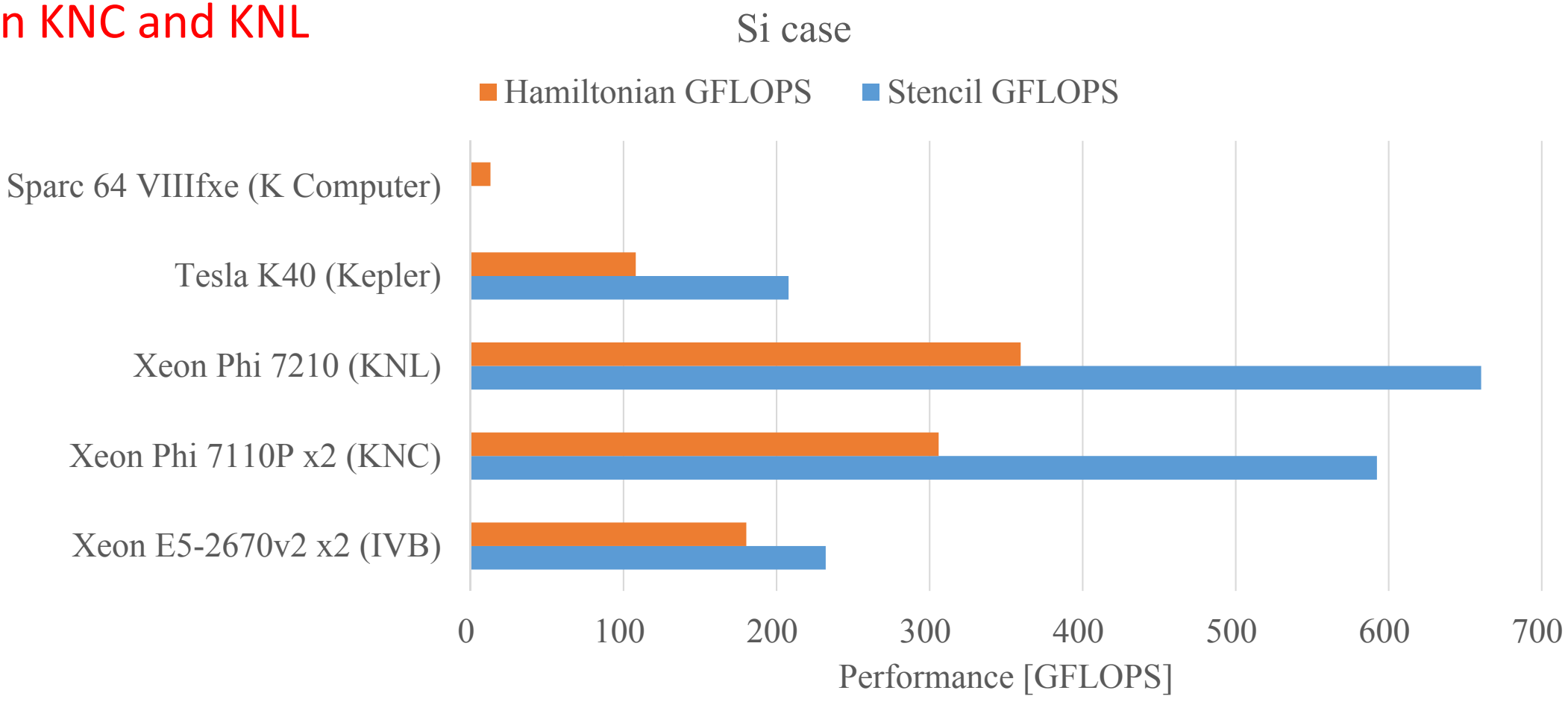
RSDFT : Real-Space Density Functional Theory
RTRSDFT: Real-Time RSDFT

ARTED preliminary result (Bin3 KNL)

JCAHPC

All data is on MCDRAM
in KNC and KNL

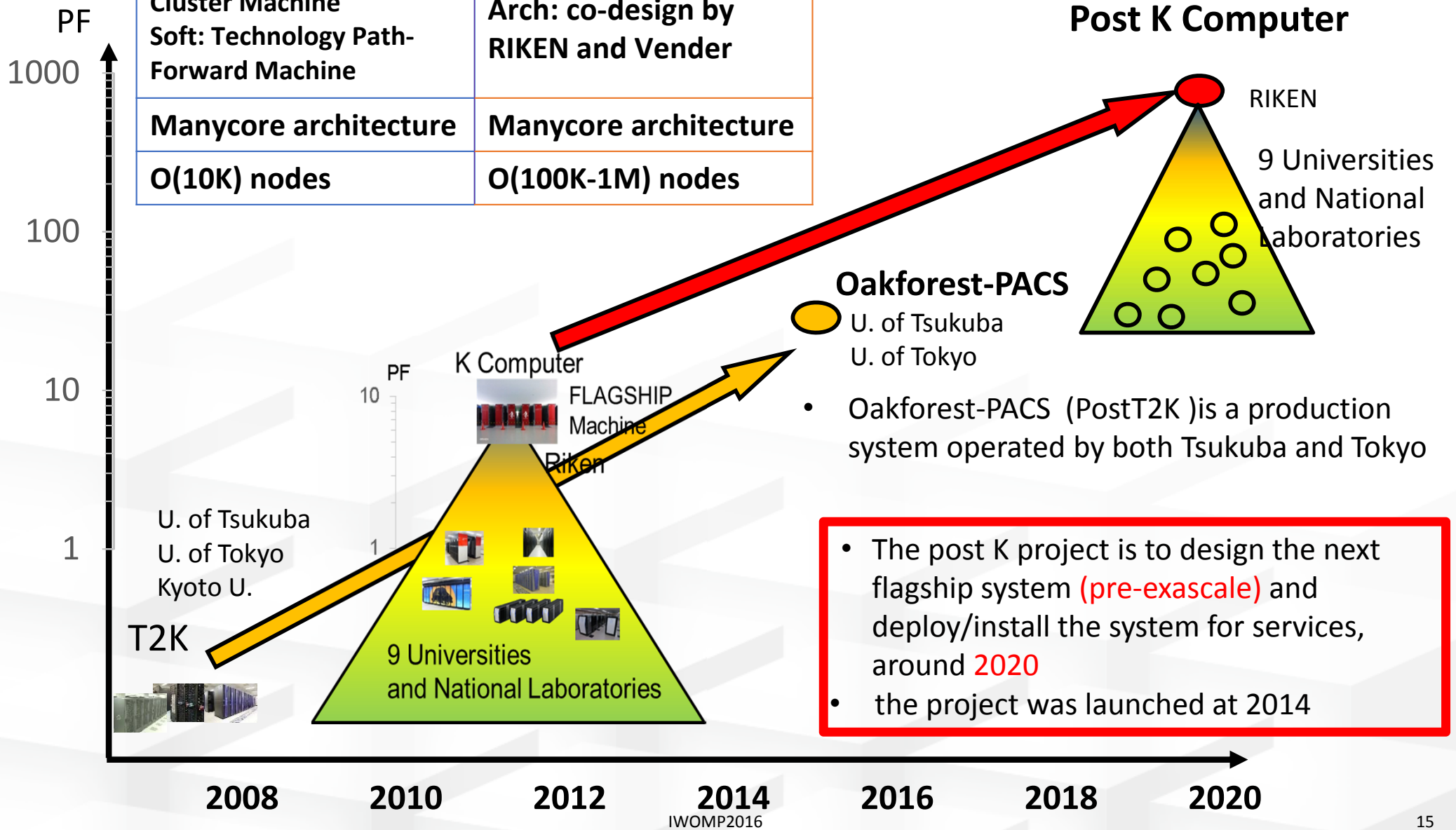
* data is preliminary and not published yet



Stencil part: single KNC = 296GFLOPS ⇒ KNL = 660GFLOPS

Towards the Next Flagship Machine

PostT2K	PostK
Arch: Upscale Commodity Cluster Machine Soft: Technology Path-Forward Machine	Flagship Machine Arch: co-design by RIKEN and Vender
Manycore architecture	Manycore architecture
O(10K) nodes	O(100K-1M) nodes



- Oakforest-PACS (PostT2K)is a production system operated by both Tsukuba and Tokyo

- The post K project is to design the next flagship system (pre-exascale) and deploy/install the system for services, around 2020
- the project was launched at 2014

Outline of Talk

- An Overview of FLAGSHIP 2020
- An Overview of post K system
- System Software
- Concluding Remarks

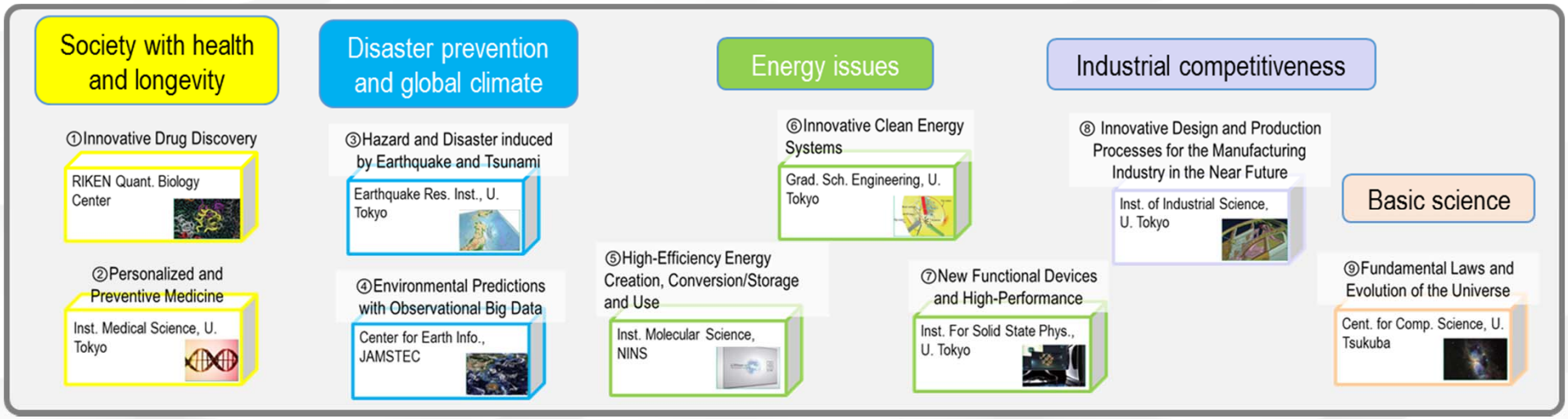
An Overview of Flagship 2020 project

- **Developing the next Japanese flagship computer, so-called "post K"**

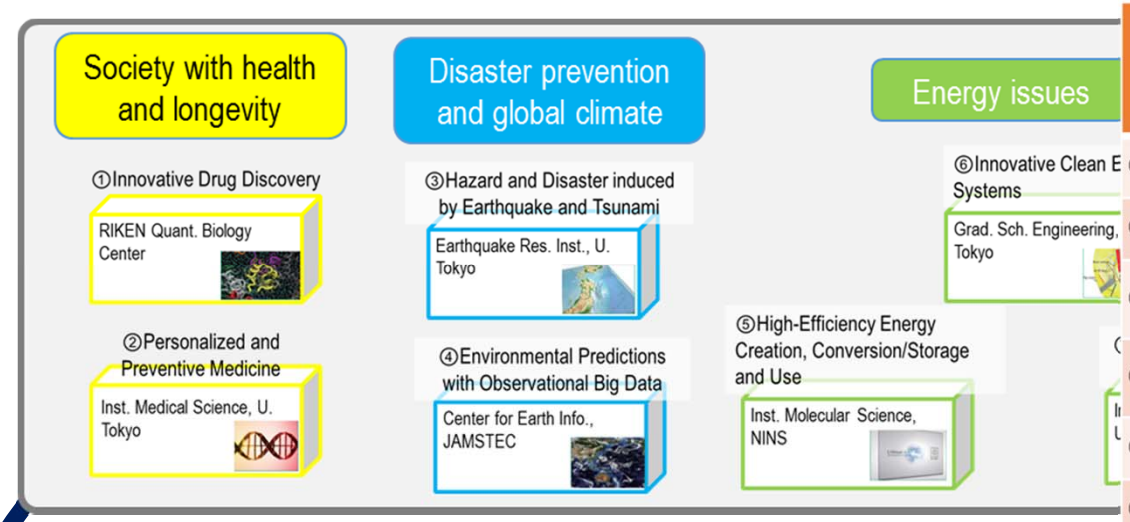
Vendor partner

- **Developing a wide range of application codes, to run on the "post K", to solve major social and science issues**

The Japanese government selected 9 social & scientific priority issues and their R&D organizations.

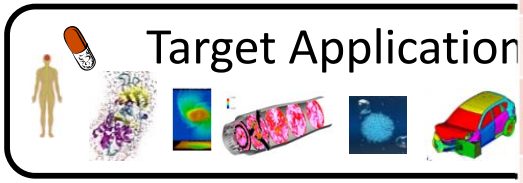


Co-design



Architectural Parameters

- #SIMD, SIMD length, #core, #NUMA node
- cache (size and bandwidth)
- memory technologies
- specialized hardware
- Interconnect
- I/O network

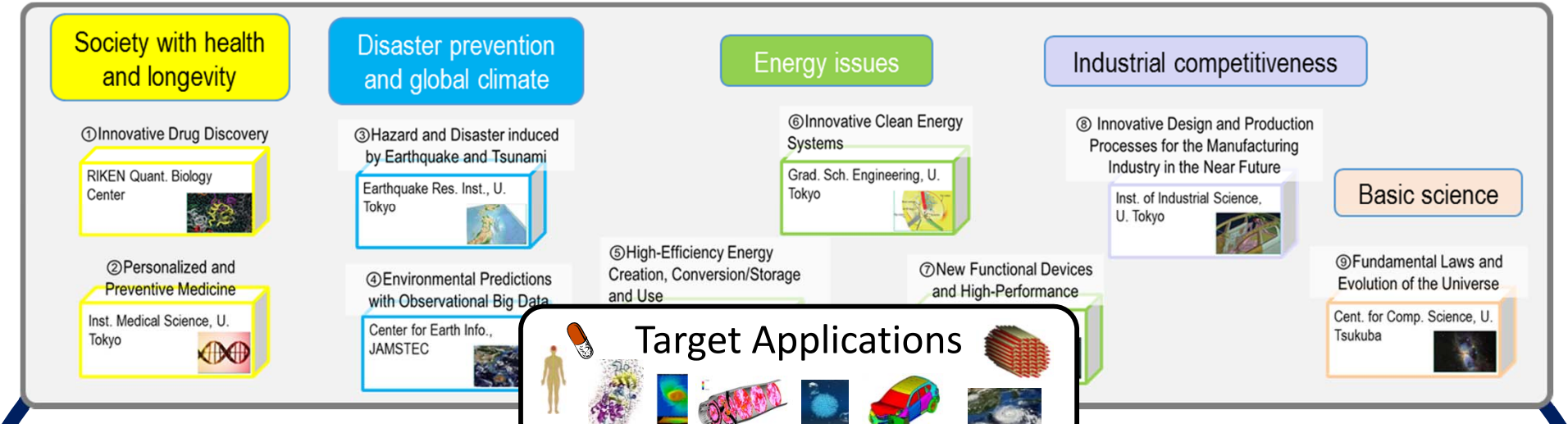


	Target Application	
	Program	**Brief description**
①	GENESIS	MD for proteins
②	Genomon	Genome processing (Genome alignment)
③	GAMERA	Earthquake simulator (FEM in unstructured & structured grid)
④	NICAM+LETK	Weather prediction system using Big data (structured grid stencil & ensemble Kalman filter)
⑤	NTChem	molecular electronic (structure calculation)
⑥	FFB	Large Eddy Simulation (unstructured grid)
⑦	RSDFT	an ab-initio program (density functional theory)
⑧	Adventure	Computational Mechanics System for Large Scale Analysis and Design (unstructured grid)
⑨	CCS-QCD	Lattice QCD simulation (structured grid Monte Carlo)

Target Applications' Characteristics

	Target Application		
	Program	Brief description	Co-design
①	GENESIS	MD for proteins	Collective comm. (all-to-all), Floating point perf (FPP)
②	Genomon	Genome processing (Genome alignment)	File I/O, Integer Perf.
③	GAMERA	Earthquake simulator (FEM in unstructured & structured grid)	Comm., Memory bandwidth
④	NICAM+LETK	Weather prediction system using Big data (structured grid stencil & ensemble Kalman filter)	Comm., Memory bandwidth, File I/O, SIMD
⑤	NTChem	molecular electronic (structure calculation)	Collective comm. (all-to-all, allreduce), FPP, SIMD,
⑥	FFB	Large Eddy Simulation (unstructured grid)	Comm., Memory bandwidth,
⑦	RSDFT	an ab-initio program (density functional theory)	Collective comm. (bcast), FFP
⑧	Adventure	Computational Mechanics System for Large Scale Analysis and Design (unstructured grid)	Comm., Memory bandwidth, SIMD
⑨	CCS-QCD	Lattice QCD simulation (structured grid Monte Carlo)	Comm., Memory bandwidth, Collective comm. (allreduce)

Co-design



Architectural Parameters

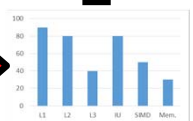
- #SIMD, SIMD length, #core,
- cache (size and bandwidth)
- memory technologies
- specialized hardware
- Interconnect
- I/O network

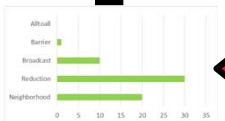
- Mutual understanding both computer architecture/system software and applications
- Looking at performance predictions
- Finding out the best solution with constraints, e.g., power consumption, budget, and space

Prediction of node-level performance

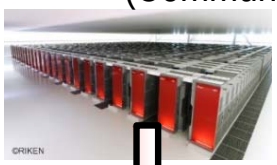
Profiling applications, e.g., cache misses and execution unit usages

Prediction Tool

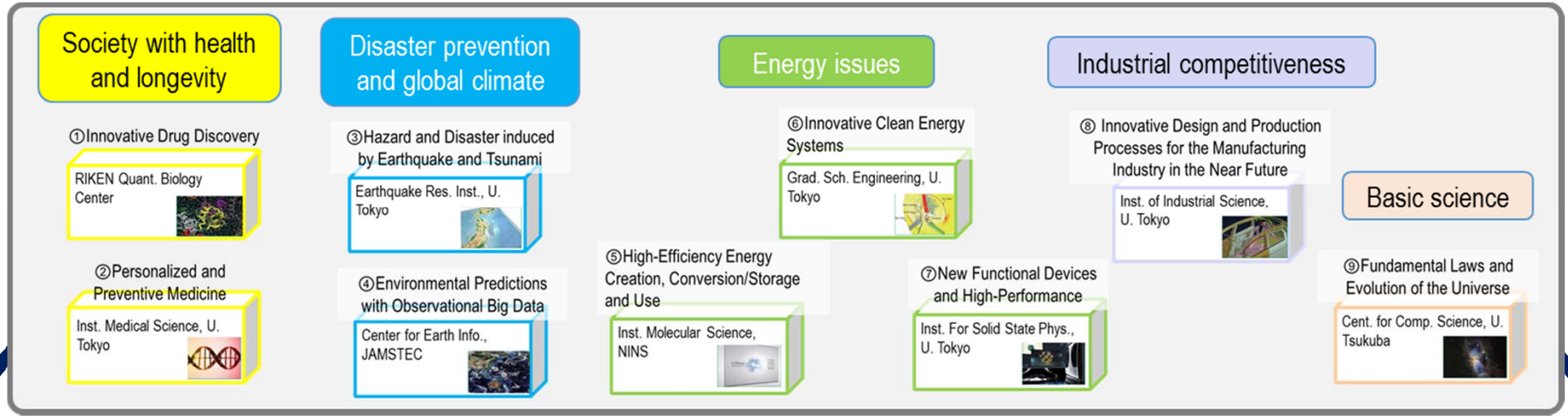




Prediction of scalability (Communication cost)



R&D Organization



FUJITSU

RIKEN AICS

Communities

- HPCI Consortium
- PC Cluster Consortium
- OpenHPC
- …

Domestic Collaboration

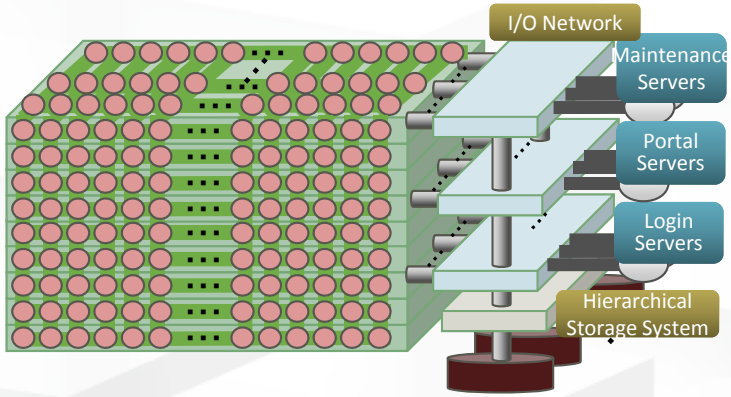
- Univ. of Tsukuba
- Univ. of Tokyo
- Univ. of Kyoto

International Collaboration

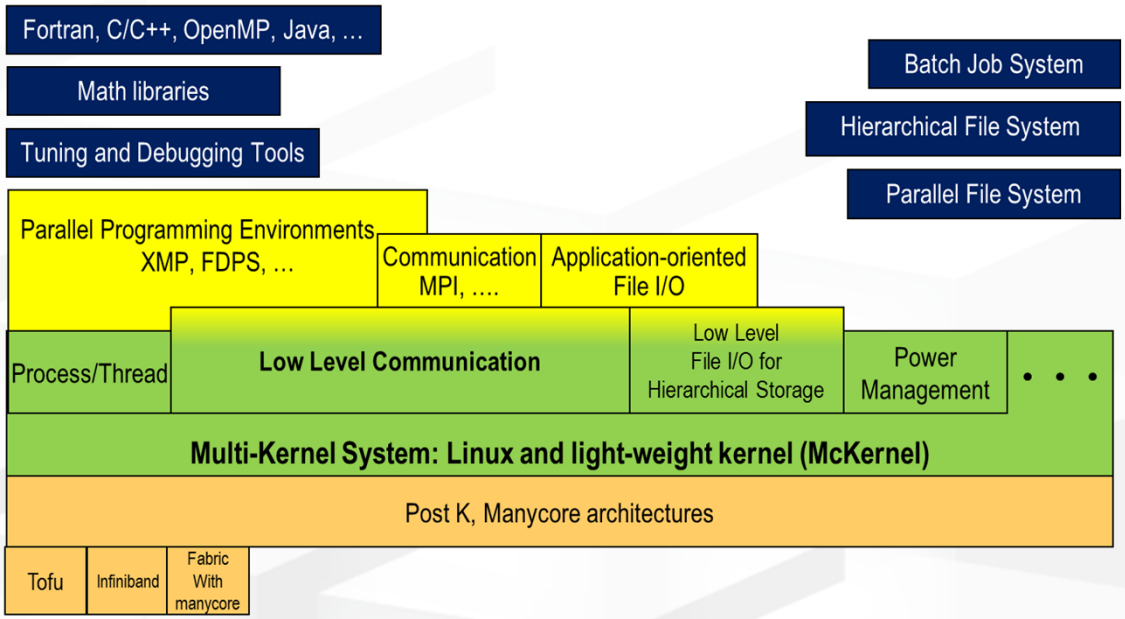
- DOE-MEXT
- JLESC
- …

An Overview of post K

- **Hardware**
  - Manycore architecture
  - 6D mesh/torus Interconnect
  - 3-level hierarchical storage system
    - Silicon Disk
    - Magnetic Disk
    - Storage for archive

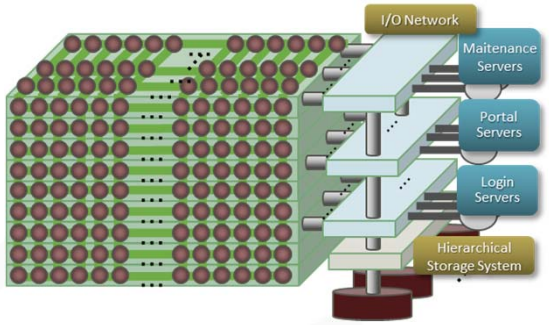


- **System Software**
  - Multi-Kernel: Linux with Light-weight Kernel
  - File I/O middleware for 3-level hierarchical storage system and application
  - Application-oriented file I/O middleware
  - MPI+OpenMP programming environment
  - Highly productive programing language and libraries



What we have done

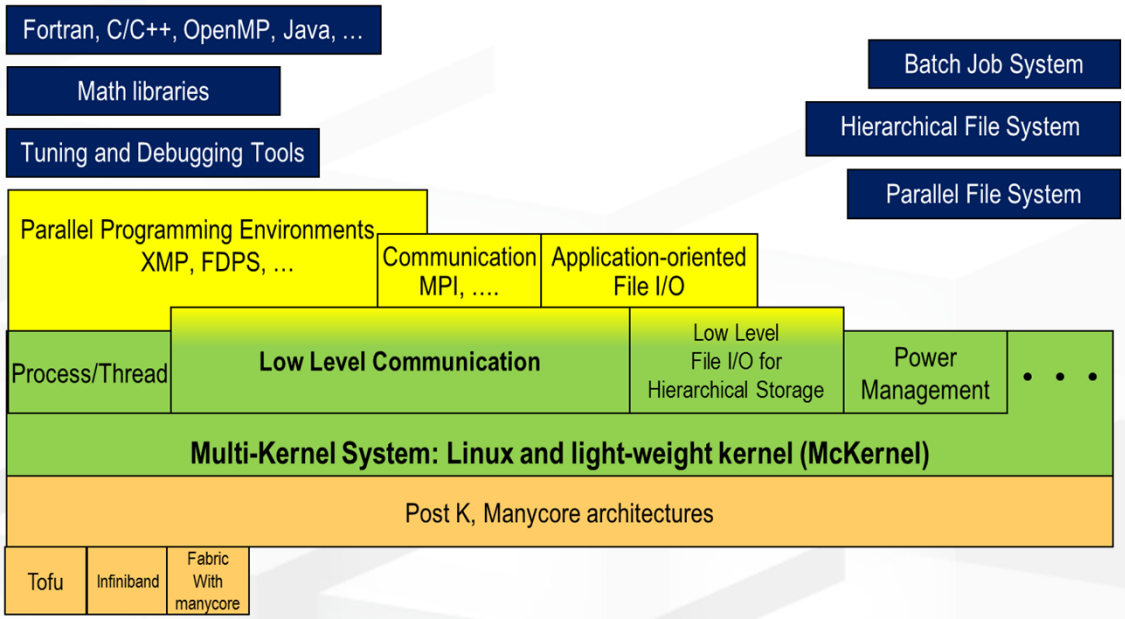
- **Hardware**
  - Instruction set architecture



Continue to design

- Node architecture
- System configuration
- Storage system

- **Software**
  - OS functional design
  - Communication functional design
  - File I/O functional design
  - Programming languages
  - Mathematical libraries



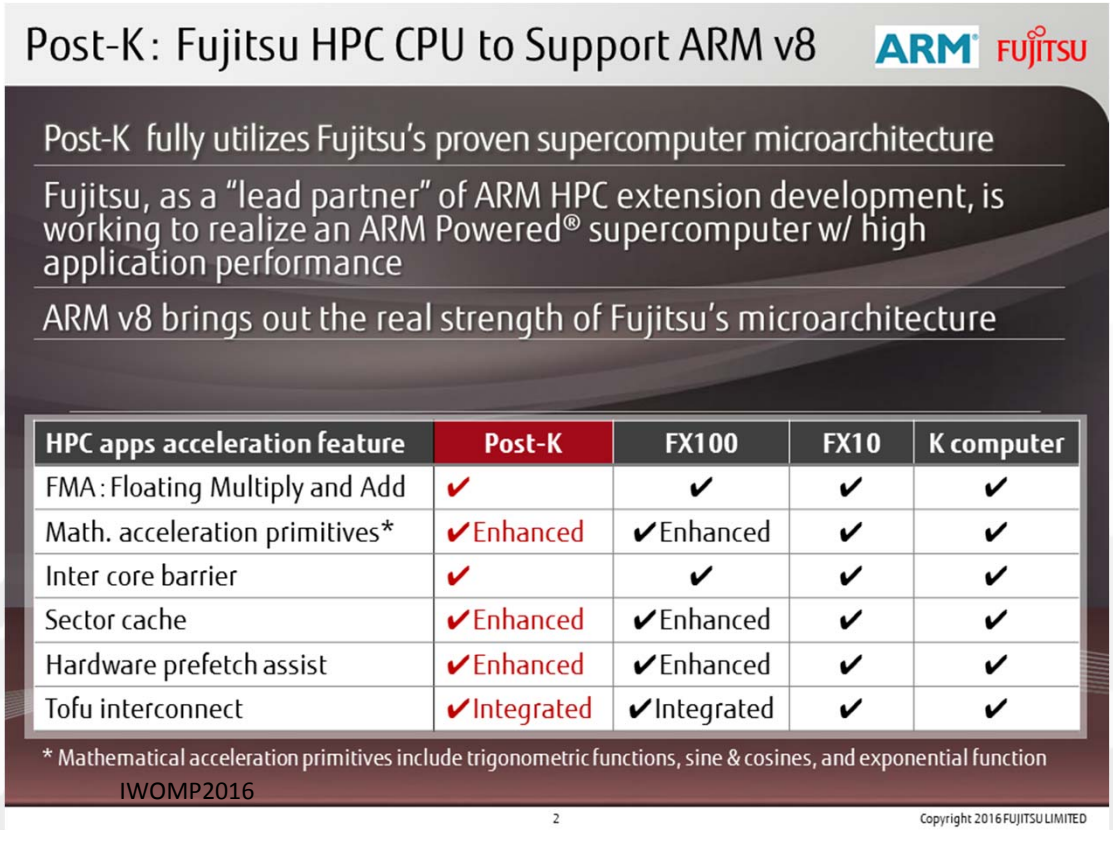
Instruction Set Architecture

- **ARM V8 with HPC Extension SVE**
  - Fujitsu is a lead partner of ARM HPC extension development
  - Detailed features were announced at Hot Chips 28 - 2016

    http://www.hotchips.org/program/
    Mon 8/22 Day1 9:45AM GPUs & HPCs
    "ARMv8-A Next Generation Vector Architecture for HPC"

SVE (Scalable Vector Extension)

- **Fujitsu's additional support**
  - FMA
  - Math acceleration primitives
  - Inter-core hardware-suppoted barrier
  - Sector cache
  - Hardware prefetch assist



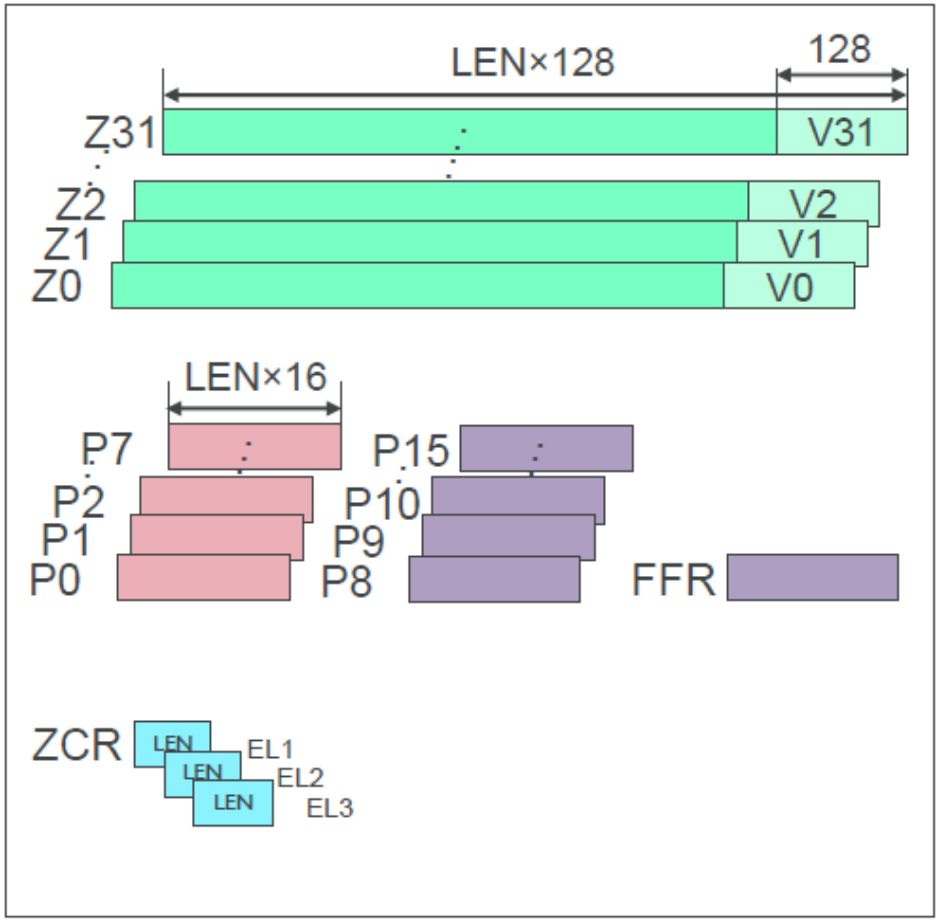
HPC apps acceleration feature	Post-K	FX100	FX10	K computer
FMA: Floating Multiply and Add	✔	✔	✔	✔
Math. acceleration primitives*	✔Enhanced	✔Enhanced	✔	✔
Inter core barrier	✔	✔	✔	✔
Sector cache	✔Enhanced	✔Enhanced	✔	✔
Hardware prefetch assist	✔Enhanced	✔Enhanced	✔	✔
Tofu interconnect	✔Integrated	✔Integrated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

ARM v8 Scalable Vector Extension (SVE)

- **SVE is a complementary extension that does not replace NEON, and was developed specifically for vectorization of HPC scientific workloads.**
- **The new features and the benefits of SVE comparing to NEON**
  - **Scalable vector length (VL)** : Increased parallelism while allowing implementation choice of VL
  - **VL agnostic (VLA) programming**: Supports a programming paradigm of write-once, run-anywhere scalable vector code
  - **Gather-load & Scatter-store**: Enables vectorization of complex data structures with non-linear access patterns
  - **Per-lane predication**: Enables vectorization of complex, nested control code containing side effects and avoidance of loop heads and tails (particularly for VLA)
  - Predicate-driven loop control and management: Reduces vectorization overhead relative to scalar code
  - Vector partitioning and SW managed speculation: Permits vectorization of uncounted loops with data-dependent exits
  - Extended integer and floating-point horizontal reductions: Allows vectorization of more types of reducible loop-carried dependencies
  - Scalarized intra-vector sub-loops: Supports vectorization of loops containing complex loop-carried dependencies

SVE architectural state

- Scalable vector registers
  - Z0-Z31 extending NEON's V0-V31
    - DP & SP floating-point
    - 64, 32, 16 & 8-bit integer
- Scalable predicate registers
  - P0-P7 lane masks for ld/st/arith
  - P8-P15 for predicate manipulation
  - FFR *first fault register*
- Scalable vector control registers
  - ZCR_ELx vector length (LEN=1..16)
  - Exception / privilege level EL1 to EL3



SVE example

DAXPY (scalar)

```
// ----------------------------------------
//       subroutine daxpy(x,y,a,n)
//       real*8 x(n),y(n),a
//       do i = 1,n
//          y(i) = a*x(i) + y(i)
//       enddo
// ----------------------------------------
// x0 = &x[0], x1 = &y[0], x2 = &a, x3 = &n
daxpy_:
    ldrsw   x3, [x3]            // x3=*n
    mov     x4, #0              // x4=i=0
    ldr     d0, [x2]            // d0=*a
    b       .latch
.loop:
    ldr     d1, [x0,x4,lsl 3]   // d1=x[i]
    ldr     d2, [x1,x4,lsl 3]   // d2=y[i]
    fmadd   d2, d1, d0, d2      // d2+=x[i]*a
    str     d2, [x1,x4,lsl 3]   // y[i]=d2
    add     x4, x4, #1          // i+=1
.latch:
    cmp     x4, x3              // i < n
    b.lt    .loop               // more to do?
    ret
```

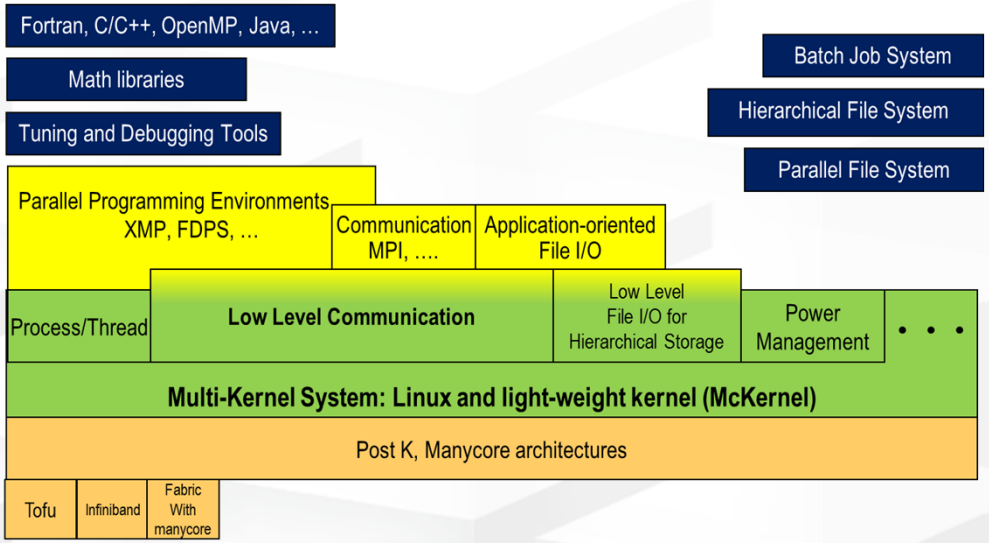
DAXPY (SVE)

```
// ----------------------------------------
//       subroutine daxpy(x,y,a,n)
//       real*8 x(n),y(n),a
//       do i = 1,n
//          y(i) = a*x(i) + y(i)
//       enddo
// ----------------------------------------
// x0 = &x[0], x1 = &y[0], x2 = &a, x3 = &n
daxpy_:
    ldrsw   x3, [x3]            // x3=*n
    mov     x4, #0              // x4=i=0
    whilelt p0.d, x4, x3        // p0=while(i++<n)
    ld1rd   z0.d, p0/z, [x2]    // p0:z0=bcast(*a)
.loop:
    ld1d    z1.d, p0/z, [x0,x4,lsl 3] // p0:z1=x[i]
    ld1d    z2.d, p0/z, [x1,x4,lsl 3] // p0:z2=y[i]
    fmla    z2.d, p0/m, z1.d, z0.d    // p0?z2+=x[i]*a
    st1d    z2.d, p0, [x1,x4,lsl 3]   // p0?y[i]=z2
    incd    x4                        // i+=(VL/64)
.latch:
    whilelt p0.d, x4, x3        // p0=while(i++<n)
    b.first .loop               // more to do?
    ret
```

- **Compact code for SVE as scalar loop**
- **OpenMP SIMD directive is expected to help the SVE programming**

Outline of Talk

- An Overview of FLAGSHIP 2020
- An Overview of post K system
- **System Software**
  - Multi-Kernel: Linux with Light-weight Kernel
  - File I/O middleware for 3-level hierarchical storage system and application
  - Application-oriented file I/O middleware
  - MPI+OpenMP programming environment
  - Highly productive programing language and libraries
- Concluding Remarks



OS Kernel

- **Requirements of OS Kernel targeting high-end HPC**
  - Noiseless execution environment for bulk-synchronous applications
  - Ability to easily adapt to new/future system architectures
    - E.g.: manycore CPUs, heterogenous core architectures, deep memory hierarchy, etc.
      - New process/thread management, memory management, …
  - Ability to adapt t
    - Big-Data, in-s
      - Support data
      - Optimize data movement

Our Approach:
Linux with Light-Weight Kernel

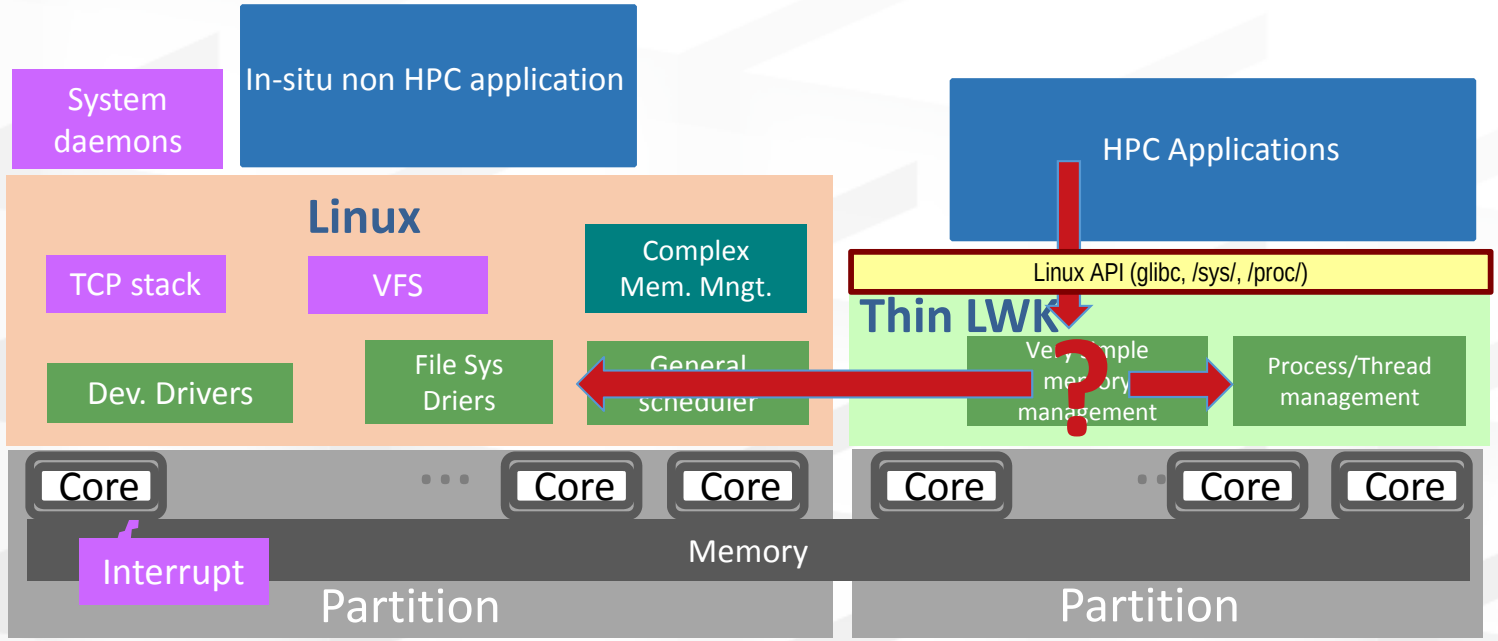
	Approach	Pros.	Cons.
Full-Weight Kernel (FWK) e.g. Linux	Disabling, removing, tuning, reimplementation, and adding new features	Large community support results in rapid new hardware adaptation	• Hard to implement a new feature if the original mechanism is conflicted with the new feature • Hard to follow the latest kernel distribution due to local large modifications
Light-Weight Kernel (LWK)	Implementation from scratch and adding new features	Easy to extend it because of small in terms of logic and code size	• Applications, running on FWK, cannot run always in LWK • Small community maintenance limits rapid growth • Lack of device drivers

McKernel developed at RIKEN

- **Enable partition resources (CPU cores, memory)**
- **Full Linux kernel on some cores**
  - System daemons and in-situ non HPC applications
  - Device drivers
- **Light-weight kernel(LWK), McKernel on other cores**
  - HPC applications

- **McKernel is loadable module of Linux**
- **McKernel supports Linux API**
- **McKernel runs on**
  - Intel Xeon and Xeon phi
  - Fujitsu FX10

McKernel is deployed to the Oakforest-PACS supercomputer, 25 PF in peak, at JCAHPC organized by U. of Tsukuba and U. of Tokyo

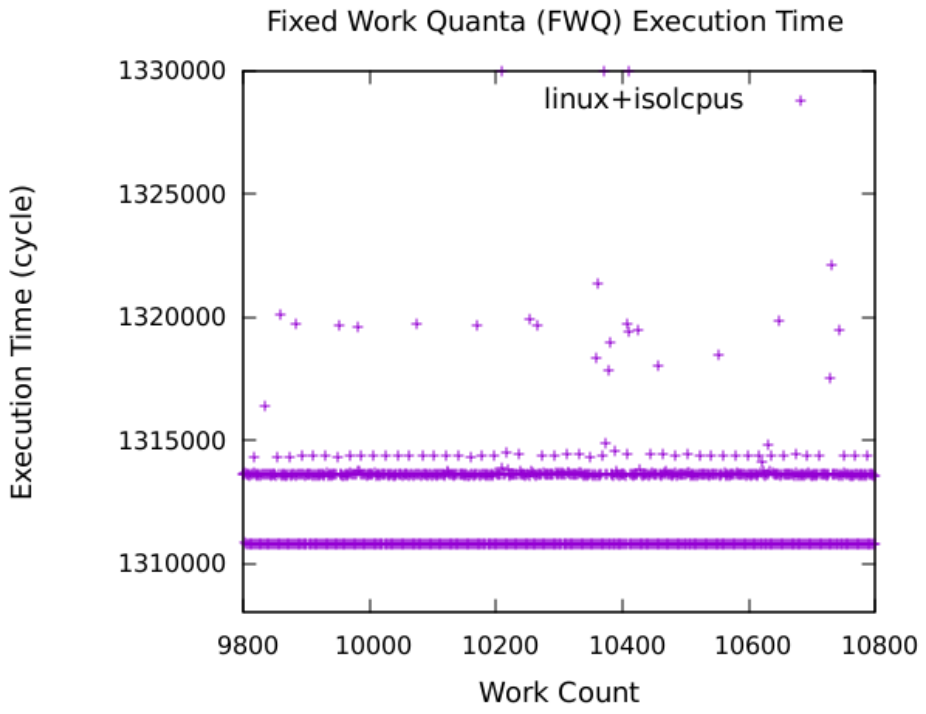


OS: McKernel

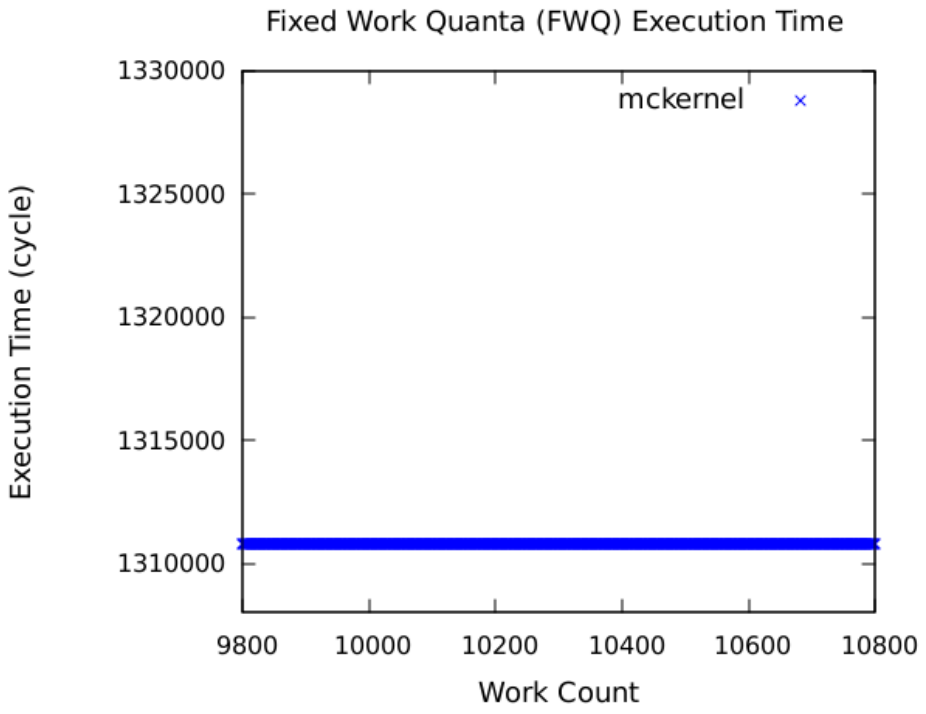
Results of FWQ (Fixed Work Quanta)

https://asc.llnl.gov/sequoia/benchmarks

Linux with isolcpus



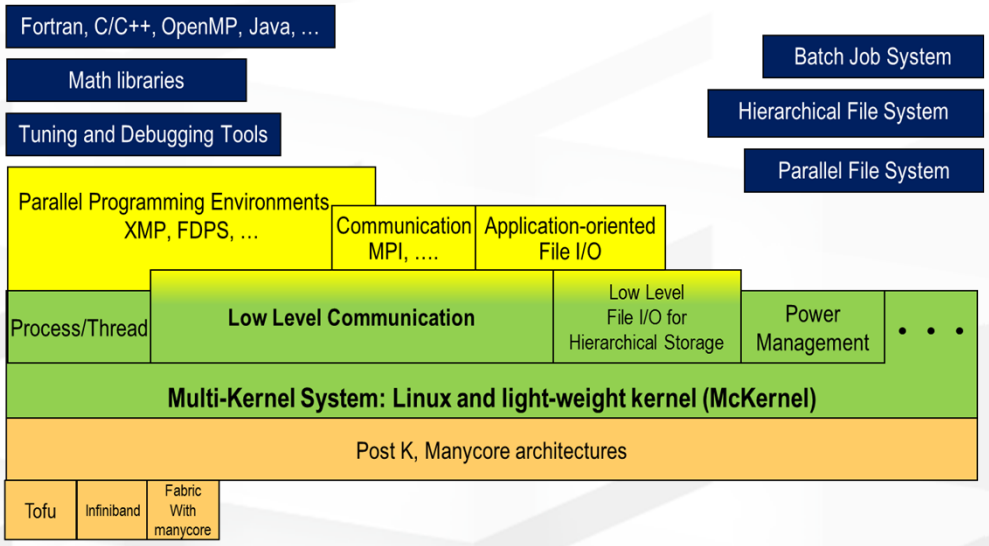
McKernel



isolcpus — Isolate CPUs from the kernel scheduler.

Outline of Talk

- An Overview of FLAGSHIP 2020
- An Overview of post K system
- **System Software**
  - Multi-Kernel: Linux with Light-weight Kernel
  - File I/O middleware for 3-level hierarchical storage system and application
  - Application-oriented file I/O middleware
  - MPI+OpenMP programming environment
  - Highly productive programing language and libraries
- Concluding Remarks



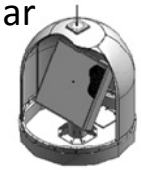
File I/O for Big data

PI: Takemasa Miyoshi, RIKEN AICS
"Innovating Big Data Assimilation technology for revolutionizing very-short-range severe weather prediction"

An innovative 30-second super-rapid update numerical weather prediction system for 30-minute/1-hour severe weather forecasting will be developed, aiding disaster prevention and mitigation, as well as bringing a scientific breakthrough in meteorology.

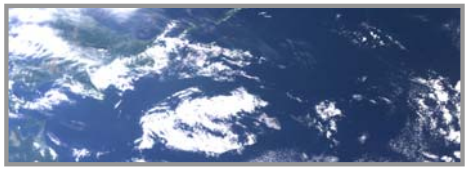
Phased array weather radar

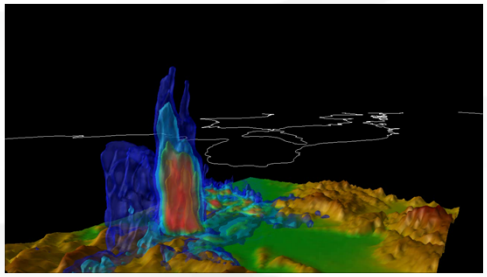
151.2 MB/30sec

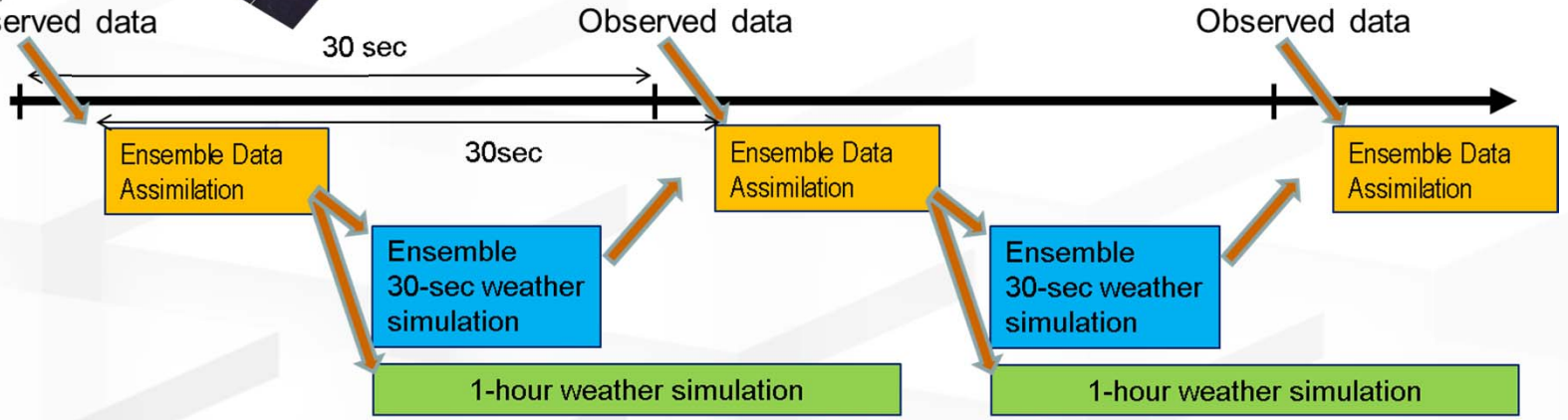


Satellite

500 MB/2.5min





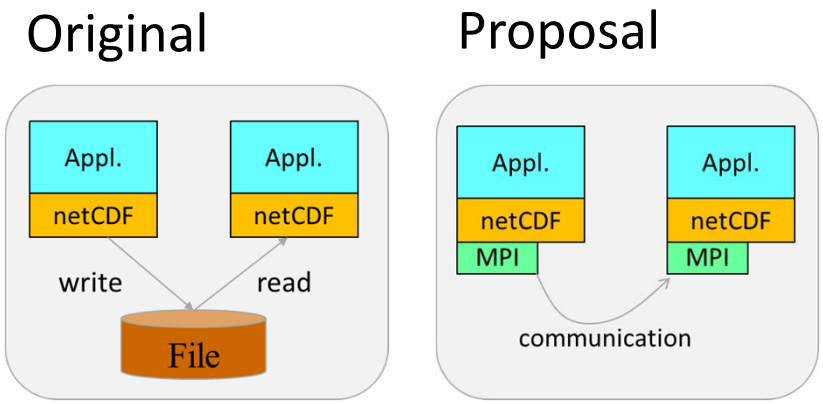


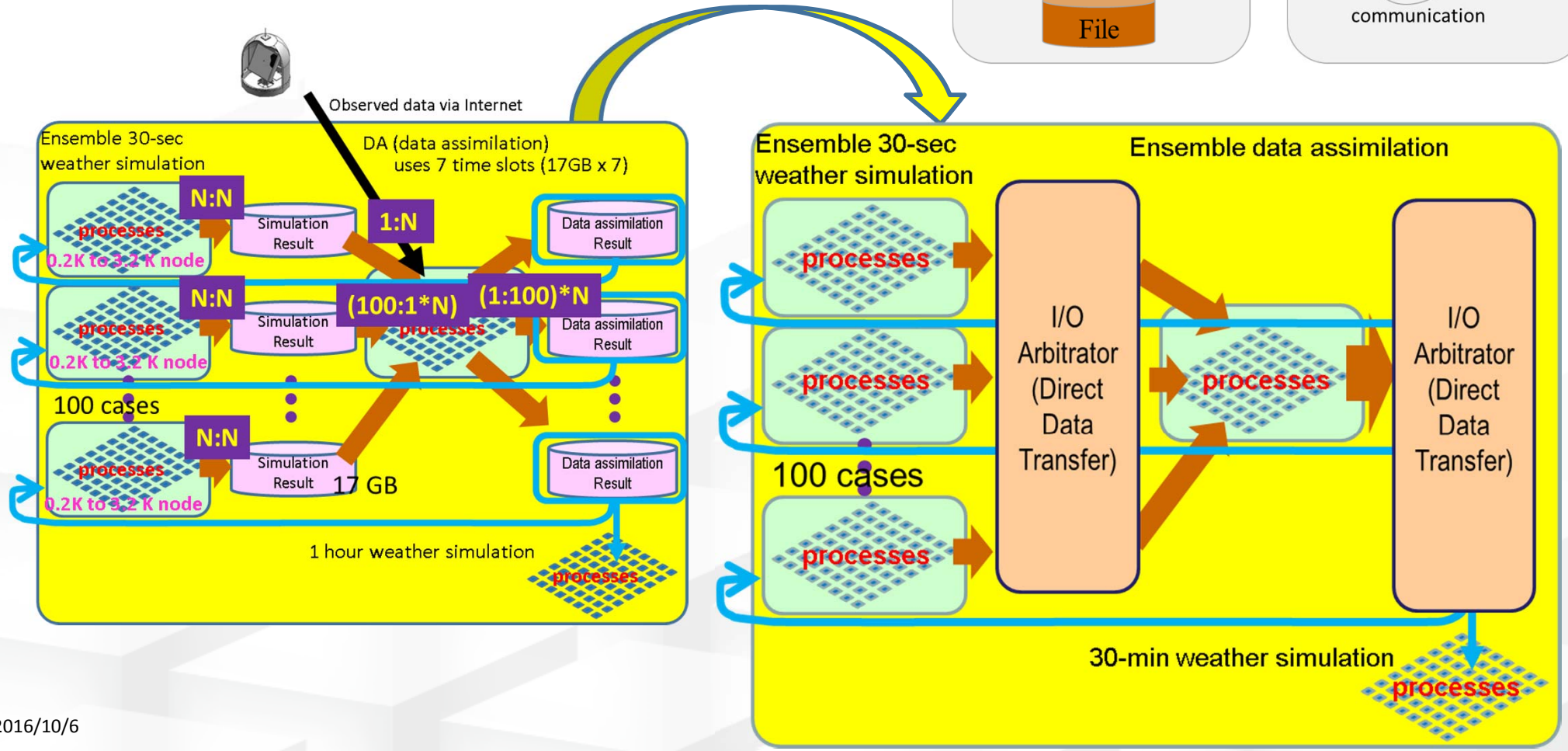
To meet real-timeness, 30 second responsibility, data exchange between data assimilation and weather simulations must be fast as much as possible

Rain particle

Approach: I/O Arbitrator

- **Keeping the netCDF file I/O API**
- **Introducing additional API in order to realize direct data transfer without storing data into storage**
  - E.g., asynchronous I/O



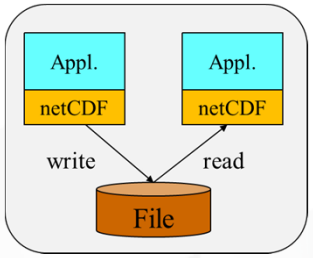


Prototype System Evaluation at RIKEN AICS

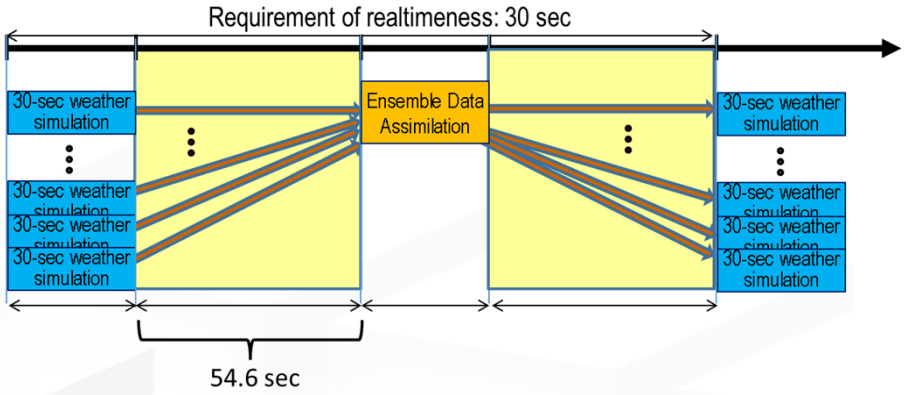
Case Study

- There are totally 11 variables, and each variable has 384 * 288 * 36 grid data (double precision). The size of transfer data between 100-case simulations and data assimilation process is about 533GB. 4,800 nodes are used.

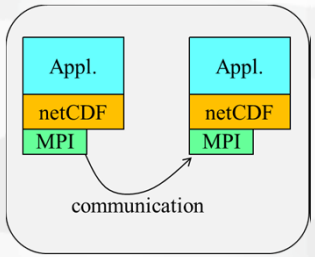
- netCDF/File I/O: 54.6 sec
  Cannot realize 30 second responsibility !

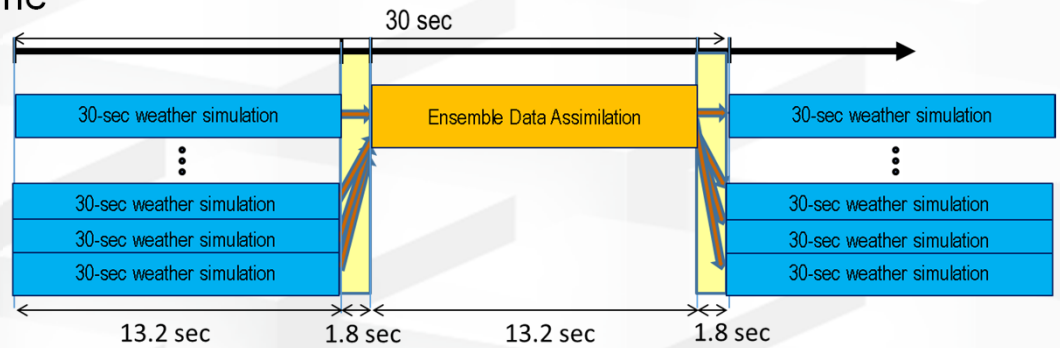


A result on K computer



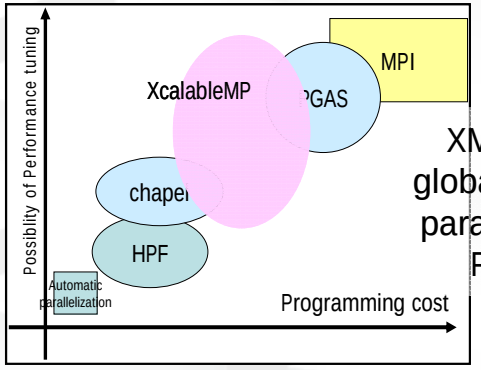
- netCDF/MPI: 1.8 sec
  Simulator and DA have 26.4 sec execution time





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 2016)**
  - XMP Spec **Version 1.2.1** is available at XMP site. new features: mixed OpenMP and OpenACC , libraries for collective communications.
  - Reference implementation by U. Tsukuba and Riken AICS: **Version 1.0 (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**



XMP provides a global view for data parallel program in PGAS model

- 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

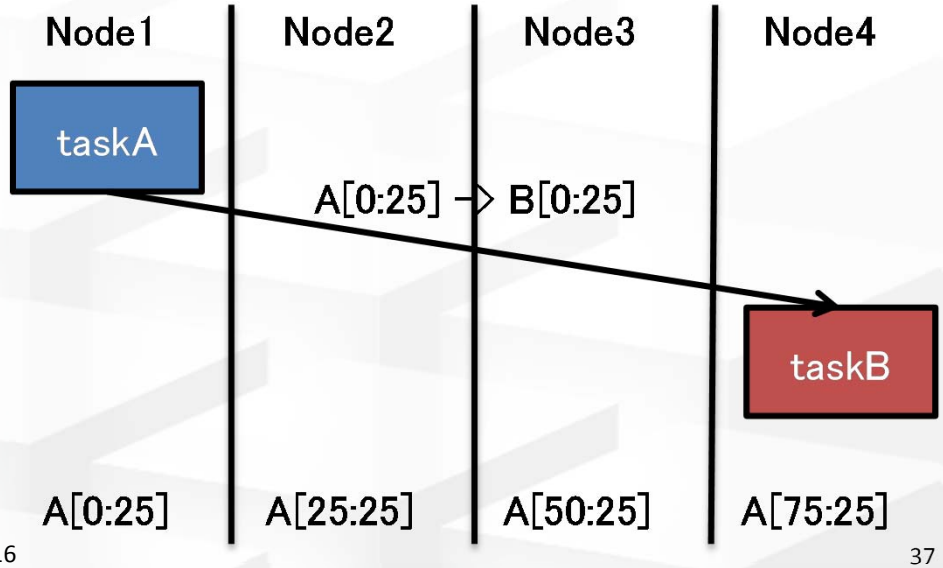
add to the serial code : incremental parallelization

work sharing and data synchronization

XcalableMP 2.0

- **Specification v 1.2:**
  - Support for Multicore: hybrid XMP and OpenMP is defined.
  - Dynamic allocation of distributed array
- **A set of spec in version 1 is now "converged". New functions should be discussed for version 2.**
- **Main topics for XcalableMP 2.0: Support for manycore**
  - Multitasking with integrations of PGAS model
  - Synchronization models for dataflow/multitasking executions
  - Proposal: tasklet directive
    - Similar to OpenMP task directive
    - Including inter-node communication on PGAS

```
int A[100], B[25];
#pragma xmp nodes P()
#pragma xmp template T(0:99)
#pragma xmp distribute T(block) onto P
#pragma xmp align A[i] with T(i)
/ … /
#pragma xmp tasklet out(A[0:25], T(75:99))
  taskA();
#pragma xmp tasklet in(B, T(0:24)) out(A[75:25])
  taskB();
#pragma xmp taskletwait
```



Concluding Remarks

- **The system software stack for Post K is being designed and implemented with the leverage of international collaborations**
  - The software stack developed at RIKEN is Open source
  - It also runs on Intel Xeon and Xeon phi
  - RIKEN will contribute to OpenHPC project