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

Kyushu Univ.:
- PC Cluster (55Tflops, 18.8TB)
- SR16000 L2 (25.3Tflops, 5.5TB)
- PC Cluster (0.5Tflops, 0.64TB)

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

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

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

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

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

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

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

Tokyo Institute of Technology:
- Tsubame 2 (2.4 Pflops, 100TB)

AICS, RIKEN:
- K computer (10 Pflops, 4PB)
- Available in 2012

Oakforest-PACS
- 25 PF KNL-based System
- It will be the fastest system in Nov. 2016 in Japan
JCAHPC and Oakforest-PACS (a.k.a Post-T2K project)

- 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

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total peak performance</strong></td>
<td>25 PFLOPS</td>
</tr>
<tr>
<td><strong>Total number of compute nodes</strong></td>
<td>8,208</td>
</tr>
<tr>
<td><strong>Compute node</strong></td>
<td><strong>Product</strong></td>
</tr>
<tr>
<td><strong>Processor</strong></td>
<td>Intel® Xeon Phi™ (Knights Landing) Xeon Phi 7250 (1.4GHz TDP) with 68 cores</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>High BW</td>
</tr>
<tr>
<td><strong>Inter-connect</strong></td>
<td>Low BW</td>
</tr>
<tr>
<td><strong>Login node</strong></td>
<td><strong>Product</strong></td>
</tr>
<tr>
<td><strong># of servers</strong></td>
<td>20</td>
</tr>
<tr>
<td><strong>Processor</strong></td>
<td>Intel Xeon E5-2690v4 (2.6 GHz 14 core x 2 socket)</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>256 GB, 153 GB/sec (DDR4-2400 x 4ch x 2 socket)</td>
</tr>
</tbody>
</table>
Full bisection bandwidth Fat-tree by Intel® Omni-Path Architecture

12 of 768 port Director Switch (Source by Intel)

362 of 48 port Edge Switch

1. Uplink: 24
2. Downlink: 24

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.

<table>
<thead>
<tr>
<th>Compute Nodes</th>
<th>8208</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login Nodes</td>
<td>20</td>
</tr>
<tr>
<td>Parallel FS</td>
<td>64</td>
</tr>
<tr>
<td>IME</td>
<td>300</td>
</tr>
<tr>
<td>Mgmt, etc.</td>
<td>8</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>8600</strong></td>
</tr>
</tbody>
</table>
(pre) Photo of computation node

Computation node (Fujitsu next generation PRIMERGY) with single chip Intel Xeon Phi (Knights Landing, 3+TFLOPS) and Intel Omni-Path Architecture card (100Gbps)
## Specification of Oakforest-PACS (I/O)

<table>
<thead>
<tr>
<th>Parallel File System</th>
<th>Type</th>
<th>Lustre File System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Capacity</td>
<td></td>
<td>26.2 PB</td>
</tr>
<tr>
<td><strong>Meta data</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product</td>
<td>DataDirect Networks MDS server + SFA7700X</td>
<td></td>
</tr>
<tr>
<td># of MDS</td>
<td>4 servers x 3 set</td>
<td></td>
</tr>
<tr>
<td>MDT</td>
<td>7.7 TB (SAS SSD) x 3 set</td>
<td></td>
</tr>
<tr>
<td><strong>Object storage</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product</td>
<td>DataDirect Networks SFA14KE</td>
<td></td>
</tr>
<tr>
<td># of OSS (Nodes)</td>
<td>10 (20)</td>
<td></td>
</tr>
<tr>
<td>Aggregate BW</td>
<td>~ 500 GB/sec</td>
<td></td>
</tr>
</tbody>
</table>

| 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

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

- **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 1\textsuperscript{st} step system operation (~410 nodes)
- 2016/12/1 2\textsuperscript{nd} 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)

- **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

---

![Diagram of Macroscopic and Microscopic grids](image.png)

**RSDFT**: Real-Space Density Functional Theory

**RTRSDFT**: Real-Time RSDFT
ARTED preliminary result (Bin3 KNL)

All data is on MCDRAM in KNC and KNL

* data is preliminary and not published yet

Si case

<table>
<thead>
<tr>
<th>Performance [GFLOPS]</th>
<th>Hamiltonian GFLOPS</th>
<th>Stencil GFLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sparc 64 VIIIfxe (K Computer)</td>
<td>296 GFLOPS</td>
<td>KNC = 660 GFLOPS</td>
</tr>
<tr>
<td>Tesla K40 (Kepler)</td>
<td>296 GFLOPS</td>
<td></td>
</tr>
<tr>
<td>Xeon Phi 7210 (KNL)</td>
<td>296 GFLOPS</td>
<td></td>
</tr>
<tr>
<td>Xeon Phi 7110P x2 (KNC)</td>
<td>296 GFLOPS</td>
<td></td>
</tr>
<tr>
<td>Xeon E5-2670v2 x2 (IVB)</td>
<td>296 GFLOPS</td>
<td></td>
</tr>
</tbody>
</table>

Stencil part: single KNC = 296GFLOPS ⇒ KNL = 660GFLOPS
Towards the Next Flagship Machine

<table>
<thead>
<tr>
<th>PostT2K</th>
<th>PostK</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Arch:</strong> Upscale Commodity Cluster Machine</td>
<td><strong>Flagship Machine</strong></td>
</tr>
<tr>
<td><strong>Soft:</strong> Technology Path-Forward Machine</td>
<td><strong>Arch:</strong> co-design by RIKEN and Vendor</td>
</tr>
<tr>
<td><strong>Manycore architecture</strong></td>
<td><strong>Manycore architecture</strong></td>
</tr>
<tr>
<td><strong>O(10K) nodes</strong></td>
<td><strong>O(100K-1M) nodes</strong></td>
</tr>
</tbody>
</table>

- **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**

The diagram shows the progression from T2K to K Computer, then to Oakforest-PACS, and finally to the post K Computer system, with key milestones and components highlighted.
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”

- 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

<table>
<thead>
<tr>
<th>Program</th>
<th>Brief description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GENESIS</td>
<td>MD for proteins</td>
</tr>
<tr>
<td>Genomon</td>
<td>Genome processing (Genome alignment)</td>
</tr>
<tr>
<td>GAMERA</td>
<td>Earthquake simulator (FEM in unstructured &amp; structured grid)</td>
</tr>
<tr>
<td>NICAM+LETK</td>
<td>Weather prediction system using Big data (structured grid stencil &amp; ensemble Kalman filter)</td>
</tr>
<tr>
<td>NTChem</td>
<td>molecular electronic (structure calculation)</td>
</tr>
<tr>
<td>FFB</td>
<td>Large Eddy Simulation (unstructured grid)</td>
</tr>
<tr>
<td>RSDFT</td>
<td>an ab-initio program (density functional theory)</td>
</tr>
<tr>
<td>Adventure</td>
<td>Computational Mechanics System for Large Scale Analysis and Design (unstructured grid)</td>
</tr>
<tr>
<td>CCS-QCD</td>
<td>Lattice QCD simulation (structured grid Monte Carlo)</td>
</tr>
</tbody>
</table>
## Target Applications’ Characteristics

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

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

Target Applications
- 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

Communities
- HPCI Consortium
- PC Cluster Consortium
- OpenHPC
- ...

Domestic Collaboration
- Univ. of Tsukuba
- Univ. of Tokyo
- Univ. of Kyoto

International Collaboration
- DOE-MEXT
- JLESC
- ...

Basic science
- DOE
- MEXT
- JLESC

Society with health and longevity
- Innovative Drug Discovery
  - RIKEN Quant. Biology Center
- Personalized and Preventive Medicine
  - Inst. Medical Sciences, U. Tokyo

Disaster prevention and global climate
- Hazard and Disaster induced by Earthquake and Tsunami
  - Earthquake Res. Inst., U. Tokyo
- Environmental Predictions with Observational Big Data
  - Center for Earth Info., JAMSTEC

Energy issues
- Innovative Clean Energy Systems
  - Grad. Sch. Engineering, U. Tokyo
- High-Efficiency Energy Creation, Conversion/Storage and Use
  - Inst. Molecular Science, NINS

Industrial competitiveness
- Innovative Design and Production Processes for the Manufacturing Industry in the Near Future
  - Inst. of Industrial Science, U. Tokyo
- Fundamental Laws and Evolution of the Universe
  - Cent. for Comp. Science, U. Tsukuba
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 programming language and libraries
What we have done

- **Hardware**
  - Instruction set architecture

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

Continue to design
- Node architecture
- System configuration
- Storage system
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-supported barrier
  - Sector cache
  - Hardware prefetch assist

---

![Post-K: Fujitsu HPC CPU to Support ARM v8](image_url)

Post-K fully utilizes Fujitsu’s proven supercomputer microarchitecture.
Fujitsu, as a “lead partner” of ARM HPC extension development, is working to realize an ARM Powered® supercomputer w/ high application performance.

ARM v8 brings out the real strength of Fujitsu’s microarchitecture.

<table>
<thead>
<tr>
<th>HPC apps acceleration feature</th>
<th>Post-K</th>
<th>FX100</th>
<th>FX10</th>
<th>K computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>FMA: Floating Multiply and Add</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Math. acceleration primitives*</td>
<td>Enhanced</td>
<td>✔️ Enhanced</td>
<td>✔️ Enhanced</td>
<td>✔️ Enhanced</td>
</tr>
<tr>
<td>Inter core barrier</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Sector cache</td>
<td>Enhanced</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Hardware prefetch assist</td>
<td>Enhanced</td>
<td>✔️ Integrated</td>
<td>✔️ Integrated</td>
<td>✔️</td>
</tr>
<tr>
<td>Tofu interconnect</td>
<td>✔️ Integrated</td>
<td>✔️ Integrated</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

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

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
● 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 programming 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 to new/future application demands
    - Big-Data, in-situ applications
      - Support data flow from Internet devices to compute nodes
      - Optimize data movement

- **Our Approach:**
  - Linux with Light-Weight Kernel

<table>
<thead>
<tr>
<th>Approach</th>
<th>Pros.</th>
<th>Cons.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-Weight Kernel (FWK) e.g. Linux</td>
<td>Disabling, removing, tuning, reimplemention, and adding new features</td>
<td>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</td>
</tr>
<tr>
<td>Light-Weight Kernel (LWK)</td>
<td>Implementation from scratch and adding new features</td>
<td>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</td>
</tr>
</tbody>
</table>
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

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
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.
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!

- netCDF/MPI: 1.8 sec
  Simulator and DA have 26.4 sec execution time
**XcalableMP (XMP)**  [http://www.xcalablemp.org](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**

- **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**

```c
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];
}
```

- **Possibility of Performance tuning**
  - Programming cost
  - Automatic parallelization
  - Possibility of Performance tuning

- **Possible code**

```c
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];
}
```
**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

```c
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
```

![Diagram showing task distribution and communication between nodes](image)
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