

FLAGSHIP2020プロジェクトと エクサスケールに向けたプログラミングモデルの課題

佐藤 三久

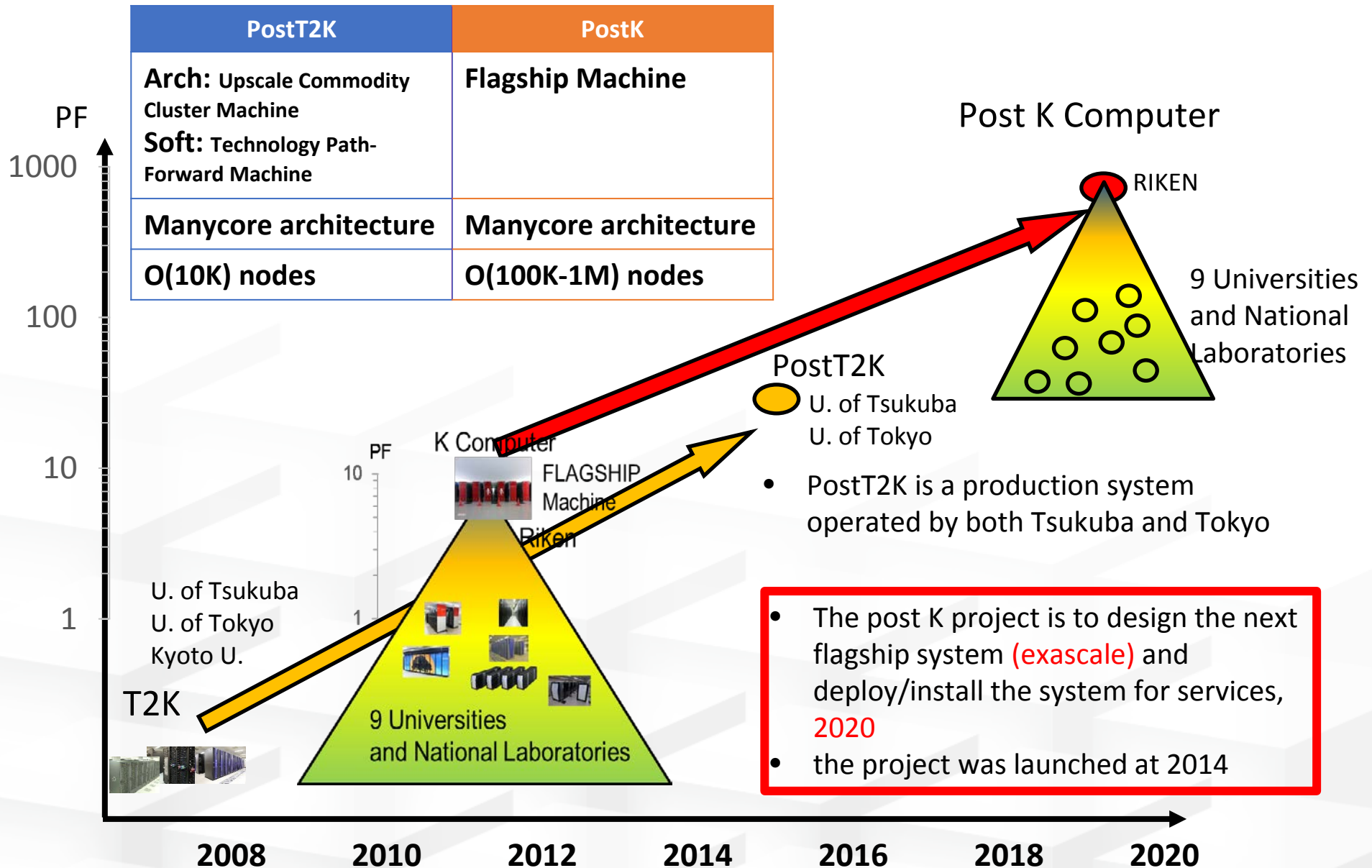
アーキテクチャ開発チーム・チームリーダー

エクサスケールコンピューティング開発プロジェクト
理化学研究所 計算科学研究機構

2015年/10月/28日

- **FLAGSHIP 2020 project**
 - to develop the next Japanese flagship computer system, “post-K”
 - “co-design” effort to design the system
- **Challenges for Parallel Programming Models and Languages for exascale computing**
 - Plan for XMP 2.0

Towards the Next Flagship Machine



FLAGSHIP 2020 Project

● Missions

- Building the Japanese national flagship supercomputer, Post K, and
- Developing wide range of HPC applications, running on Post K, in order to solve social and science issues in our country.

● Planned Budget

- 110 Billion JPY (about 0.91 Billion USD at the rate 120 JPY/\$)
- including research, development (NRE) and acquisition/deploy, and application development

● Post K Computer: System and Software

- RIKEN AICS is in charge of development
- Fujitsu is selected as a vendor partner
- Started from 2014

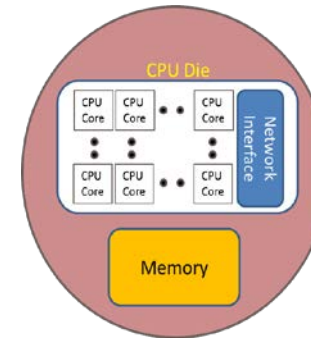
CY	2014				2015				2016				2017				2018				2019				2020			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
	Basic Design				Design and Implementation								Manufacturing, Installation, and Tuning								Operation							

Current status of the post-K project

- **The procurement for the development of the post-K computer system was done.**
 - Fujitsu was selected as the vender partner.
- **In the specification of RFP:**
 - Constraints are:
 - Power capacity (about 30MW)
 - Space for system installation (in Kobe AICS building)
 - Budget (money) for development (NRE) and production.
 - ... some degree of compatibility to the current K computer.
- **We are now finishing the “basic design” of the system with the vender partner.**
- **The system should be designed to maximize the performance of applications in each computational science field.**
 - "Co-design" is a keyword!

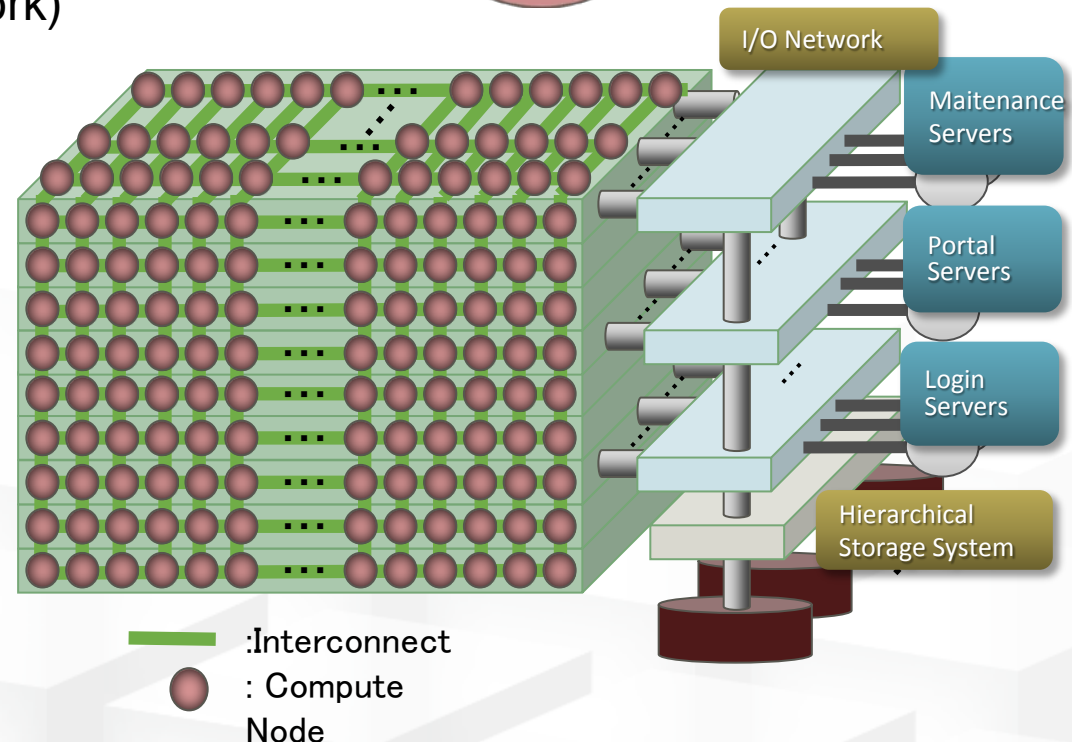
Post K Computer

- ✓ CPU
 - Many-core with Interconnect interface integrated on chip
 - Power Knob feature for saving power
- ✓ Interconnect
 - TOFU (mesh/torus network)



Co-design may include:

- Compute Node Features
 - Core architecture, FP performance
 - Memory hierarchy, control, capacity, and bandwidth
- Network Performance
- I/O Performance



- **なぜ、コデザインが必要か？（特に、“エクサ・スケールシステム”に向けて！）**
 - 電力の制約： 一定の電力の制約の上で、システムの性能を上げる必要がある（postKの仕様書では、約 30 MW）
 - コストの制約： コストも同じように抑える必要がある。

アプリケーションの特性を考慮した設計が必要 ⇒ コデザイン

- **HPCにおけるコデザインは、できるだけ多くのアプリをカバーしつつ、性能を最適化する必要がある。**
 - 組み込みの“コデザイン”とは、異なる。組み込み向けのシステムでは、特定のアプリケーションに“特化”したデザインのことを意味する場合が多い。
 - 一方、HPCシステムは、システムのコストが高くなるため、たくさんアプリケーションを実行できなくてはならない。

- Hardware/architecture

- Node architecture (#core, #SIMD, etc...)
- cache (size and bandwidth)
- network (topologies, latency and bandwidth)
- memory technologies (HBM and HMC, ...)
- specialized hardware
- #nodes
- Storage, file systems
- ... system configurations

- System software

- Operating system for many core architecture
- communication library (low level layer, MPI, PGAS)
- Programming model and languages

- Algorithm and math lib

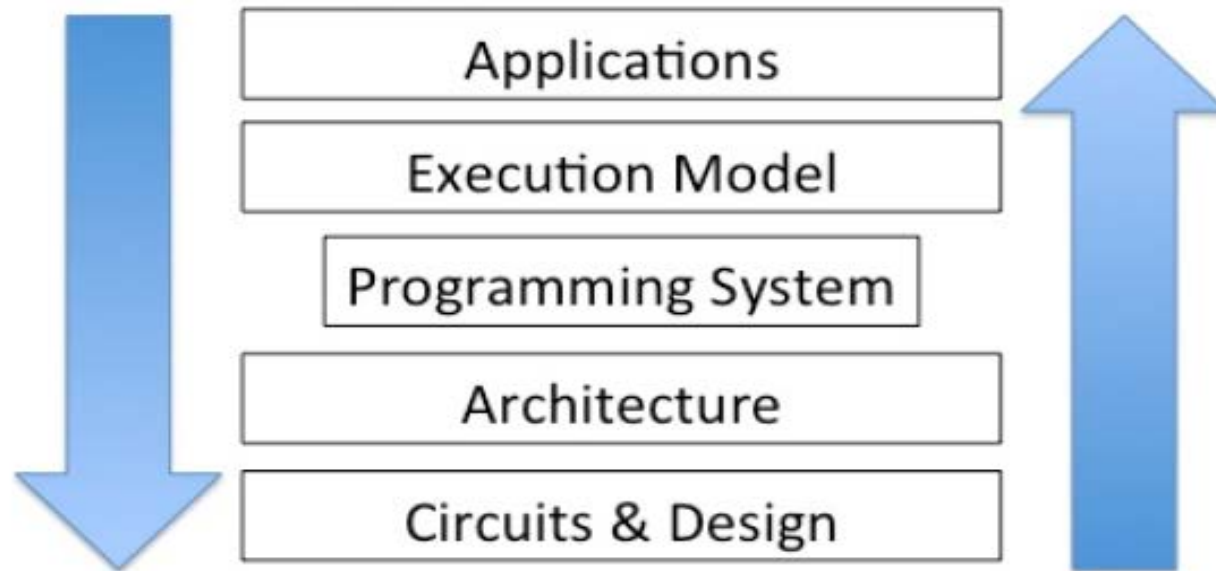
- Dense and Sparse solver
- Eigen solver
- Domain-specific lang & lib and framework

- And, Applications!

HPCにおけるコデザイン （２）

- Richard F. BARRETT, et.al. “On the Role of Co-design in High Performance Computing”, *Transition of HPC Towards Exascale Computing* より

*Analysis of applications to devise
the most efficient solutions*

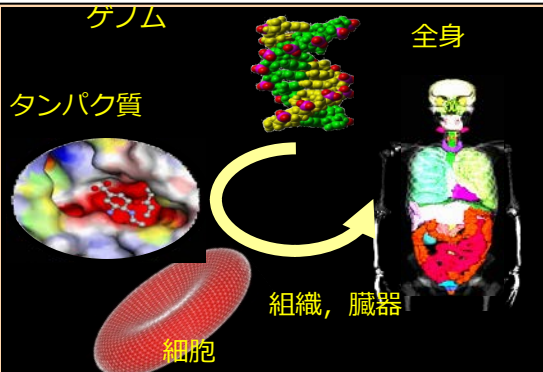


*Issues and opportunities
to exploit*

- 「京」の時には、戦略プログラム SPIRE (Strategic Programs for Innovative Research) を対象とした
 - これは、京が稼働した後。
 - 京の設計・稼働前には、「グランドチャレンジプログラム」があった。
- Post Kに向けては：
 - 昨年度において、委員会が組織され、9つの重点課題が選定され、それぞれの重点課題の研究開発実施機関が選定された。
 - それぞれの重点課題から、ターゲットとなるアプリケーションと実行シナリオが提案された。

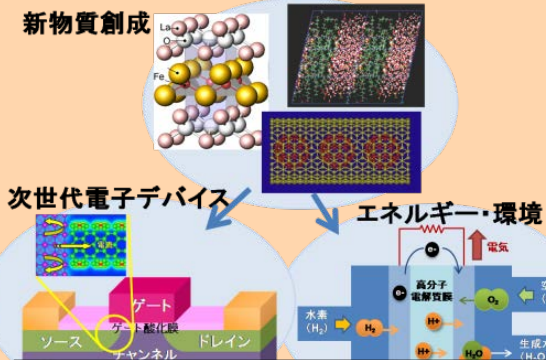
Five strategic areas of SPIRE

Life science/Drug manufacture



Toshio YANAGIDA
(RIKEN)

New material/energy creation



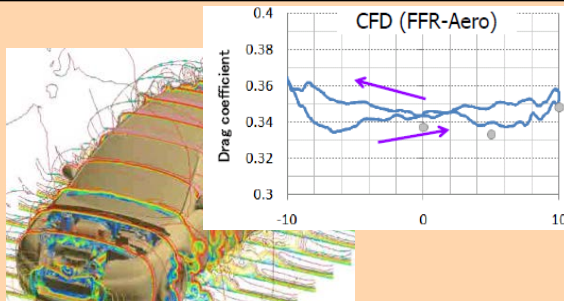
Shinji TSUNEYUKI
(University of Tokyo)

Global change prediction for disaster prevention/mitigation



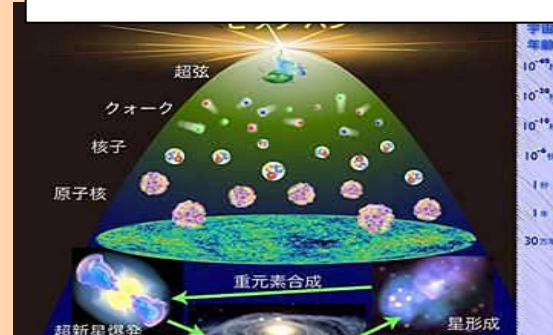
Shiro IMAWAKI
(JAMSTEC)

Monodukuri
(Manufacturing technology)





Chisachi KATO
(University of Tokyo)

The origin of matter and the universe

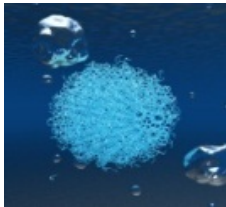
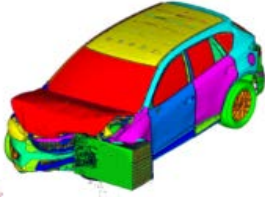
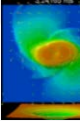


Shinya AOKI
(University of Tsukuba)

- ①社会的・国家的見地から高い意義がある、
- ②世界を先導する成果の創出が期待できる、
- ③ポスト「京」の戦略的活用が期待できる課題を「重点課題」として選定。

カテゴリ	重点課題
健康長寿社会の実現 	① 生体分子システムの機能制御による革新的創薬基盤の構築 超高速分子シミュレーションを実現し、副作用因子を含む多数の生体分子について、機能阻害ばかりでなく、機能制御までも達成することにより、有効性が高く、さらに安全な創薬を実現する。
	② 個別化・予防医療を支援する統合計算生命科学 健康・医療ビッグデータの大規模解析とそれらを用いて得られる最適なモデルによる生体シミュレーション（心臓、脳神経など）により、個々人に適した医療、健康寿命を延ばす予防をめざした医療を支援する。
防災・環境問題 	③ 地震・津波による複合災害の統合的予測システムの構築 内閣府・自治体等の防災システムに実装しうる、大規模計算を使った地震・津波による災害・被害シミュレーションの解析手法を開発し、過去の被害経験からでは予測困難な複合災害のための統合的予測手法を構築する。
	④ 観測ビッグデータを活用した気象と地球環境の予測の高度化 観測ビッグデータを組み入れたモデル計算で、局地的豪雨や竜巻、台風等を高精度に予測し、また、人間活動による環境変化の影響を予測し監視するシステムの基盤を構築する。環境政策や防災、健康対策へ貢献する。

本日この後紹介

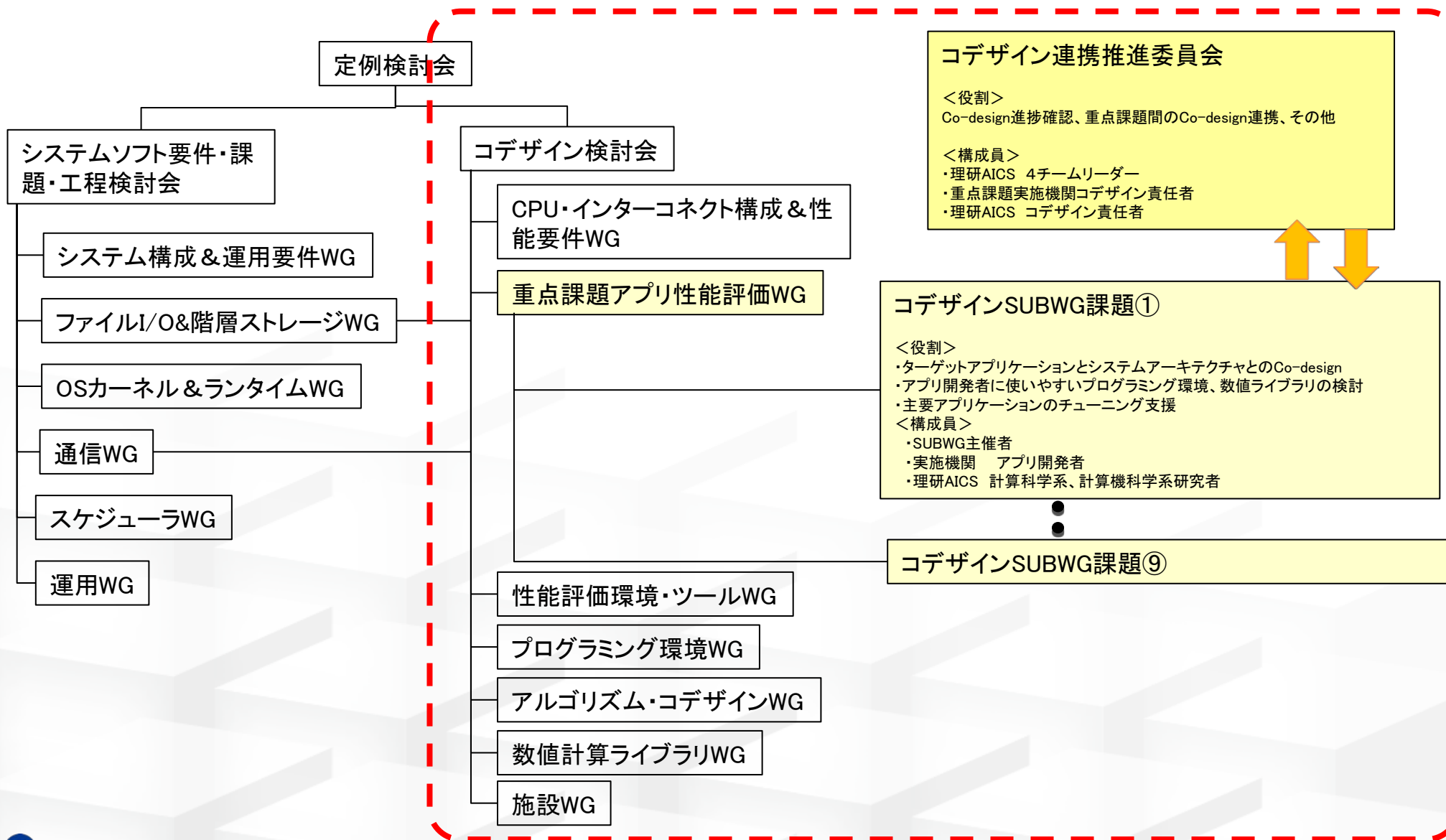
カテゴリ	重点課題
エネルギー問題 	<p>⑤ エネルギーの高効率な創出、変換・貯蔵、利用の新規基盤技術の開発 複雑な現実複合系の分子レベルでの全系シミュレーションを行い、高効率なエネルギーの創出、変換・貯蔵、利用の全過程を実験と連携して解明し、エネルギー問題解決のための新規基盤技術を開発する。</p> <p>⑥ 革新的クリーンエネルギーシステムの実用化 エネルギーシステムの中核をなす複雑な物理現象を第一原理解析により、詳細に予測・解明し、超高効率・低環境負荷な革新的クリーンエネルギーシステムの実用化を大幅に加速する。</p>
産業競争力の強化 	<p>⑦ 次世代の産業を支える新機能デバイス・高性能材料の創成 国際競争力の高いエレクトロニクス技術や構造材料、機能化学品等の開発を、大規模超並列計算と計測・実験からのデータやビッグデータ解析との連携によって加速し、次世代の産業を支えるデバイス・材料を創成する。</p> <p>⑧ 近未来型ものづくりを先導する革新的設計・製造プロセスの開発 製品コンセプトを初期段階で定量評価し最適化する革新的設計手法、コストを最小化する革新的製造プロセス、およびそれらの核となる超高速統合シミュレーションを研究開発し、付加価値の高いものづくりを実現する。</p>
基礎科学の発展 	<p>⑨ 宇宙の基本法則と進化の解明 素粒子から宇宙までの異なるスケールにまたがる現象の超精密計算を実現し、大型実験・観測のデータと組み合わせて、多くの謎が残されている素粒子・原子核・宇宙物理学全体にわたる物質創成史を解明する。</p>

カテゴリ	重点課題名	選定実施機関
健康長寿社会の実現	①生体分子システムの機能制御による革新的創薬基盤の構築	理化学研究所生命システム研究センター (課題責任者：奥野 恭史・客員主管研究員) 他 5 機関
	②個別化・予防医療を支援する統合計算生命科学	東京大学医科学研究所 (課題責任者：宮野 悟・教授) 他 5 機関
防災・環境問題	③地震・津波による複合災害の統合的予測システムの構築	東京大学地震研究所 (課題責任者：堀 宗朗・教授) 他 4 機関
	④観測ビッグデータを活用した気象と地球環境の予測の高度化	海洋研究開発機構地球情報基盤センター (課題責任者：高橋 桂子・センター長) 他 3 機関
エネルギー問題	⑤エネルギーの高効率な創出、変換・貯蔵、利用の新規基盤技術の開発	自然科学研究機構分子科学研究所 (課題責任者：岡崎 進・教授) 他 8 機関
	⑥革新的クリーンエネルギーシステムの実用化	東京大学大学院工学系研究科 (課題責任者：吉村 忍・教授) 他 11 機関
産業競争力の強化	⑦次世代の産業を支える新機能デバイス・高性能材料の創成	東京大学物性研究所 (課題責任者：常行 真司・教授) 他 8 機関
	⑧近未来型ものづくりを先導する革新的設計・製造プロセスの開発	東京大学生産技術研究所 (課題責任者：加藤 千幸・教授) 他 6 機関
基礎科学の発展	⑨宇宙の基本法則と進化の解明	筑波大学計算科学研究センター (課題責任者：青木 慎也・客員教授) 他 7 機関

重点課題からのアプリケーション

	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)

Co-design推進体制



(基本設計における) コデザインの取り組み

- 各アプリをベースに、システムの基本構成・パラメータの決定
- ベンダーが提供するツール
 - ① 性能電力予測ツール： FX-100(もしくはFX-10)のプロファイル情報を入力して、post-Kの性能を予測するツール
 - ② 性能シミュレータ+コンパイラ： post-Kのシミュレーション環境（但し、カーネル評価に限定される）
- 性能評価： 各アプリについて実施
 - (1) 性能概算見積もり - 定式化による性能見積もり（roof-lineモデル等）
 - (2) 詳細性能見積もり - ①のツールを利用した見積もり
 - (3) カーネル性能見積もり - ②のシミュレータを利用。但し、カーネルの切り出しが必要。
- コスト・全体電力を勘案し、プロセッサアーキテクチャ・ネットワークの基本的なパラメータを策定
 - コア数、演算性能、キャッシュ構成、メモリ構成、ネットワーク構成、…

(基本設計における) コデザインの取り組み

- 制約条件としてのコスト・全体電力からのシステム構成の検討
- 各アプリでの電力制御の方式・可能性の検討
 - ネットワークのバンド幅選択やCPUの周波数等のPower-Knob制御
- プログラミング環境（言語コンパイラ等）・性能ツール・数値計算ライブラリ
 - 基本設計を行うとともに、ユーザからヒアリングを行い基本設計に反映
- 粒子系、連続系などの典型的なアプリに対するDSLの設計・プロトタイピング
- システムソフトウェア
 - ファイルシステム、…

●「京」の時からの違い

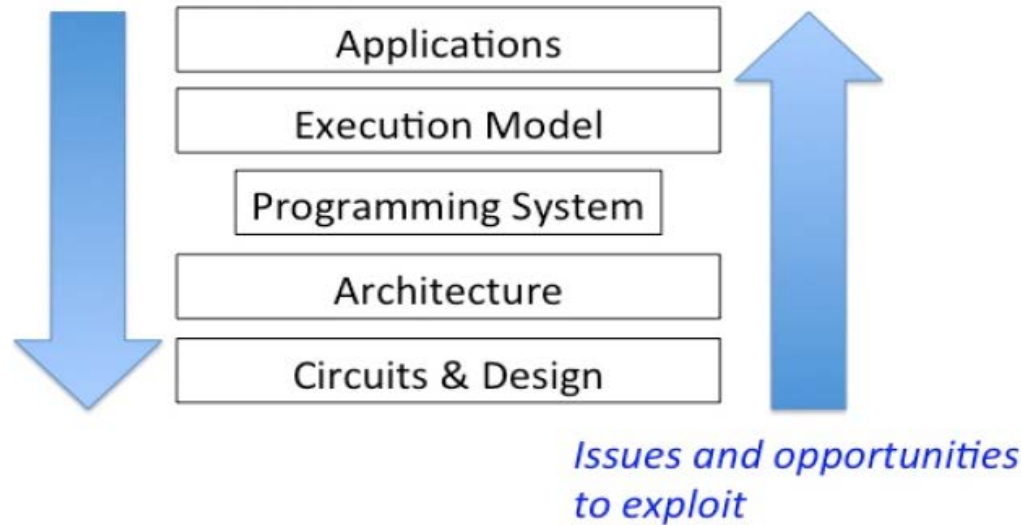
- ツールの高度化
- ターゲットアプリの明確化
- アプリの実行シナリオを考慮（「京」の時は、capability的なシナリオが主だった）
- ベースとなるアーキテクチャ、経験がある。

● スパコンセンター等の調達でのコデザインとの違い

- プロセッサのアーキテクチャまで踏み込んでいる。調達では、コデザインはプロセッサ・ネットワークの「選択」
- 規模が違う（が、最近のスパコンセンターのシステムでも電力・規模はシステム設計の重要な要素）

これからのコデザイン計画

*Analysis of applications to devise
the most efficient solutions*



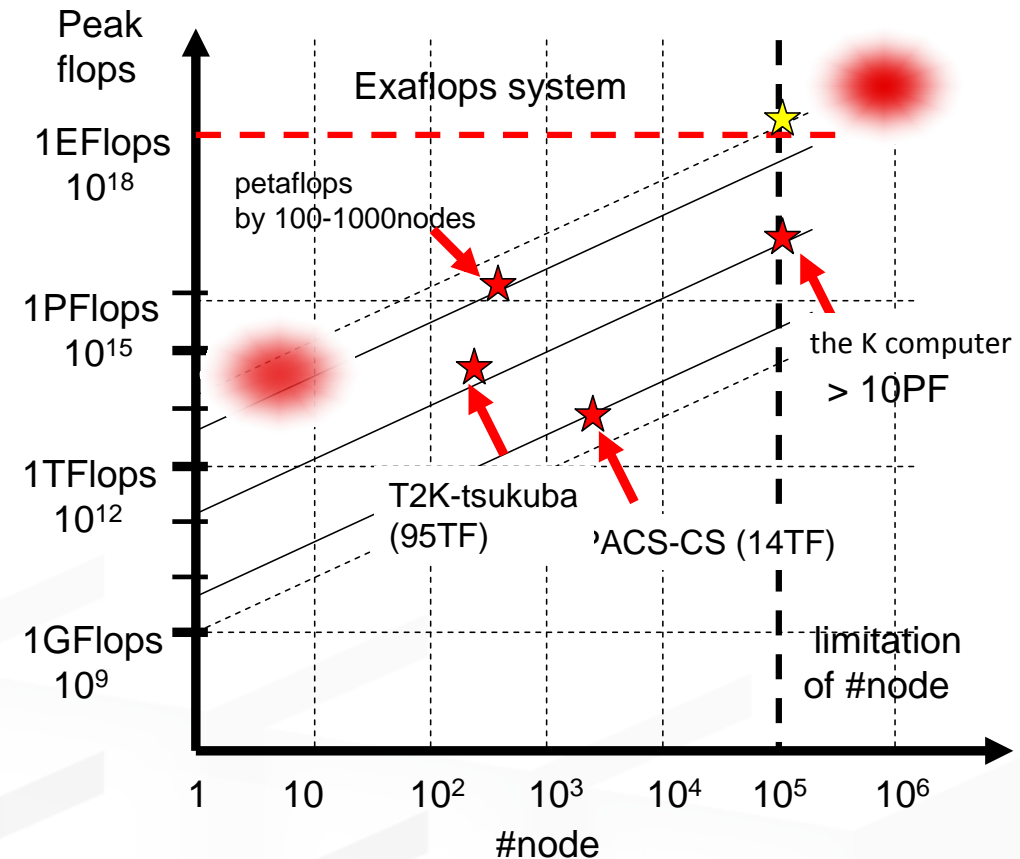
- 問題点・コメント
 - 既存のアプリからの検討で、必ずしも新しい“革新的な”アーキテクチャが生まれるわけではない。
 - 最適化されているアプリは、ハードウェアの選択の幅を狭くする。
 - 多様なプログラムをサポートするのも重要な要素

- 今までは、主に“上から下へ”。
- ターゲットアプリの性能の確保、複数のアプリを支えるUnionのアーキテクチャ
- これからは“下から上へ”も進める必要がある
 - 全体電力・コストの制約はこの一つ
 - アーキテクチャの特徴（メニーコアなど）を生かしたアプリ、プログラミングモデル、アルゴリズムの開発
 - 電力を考慮したアプリ開発、電力制御方式
- さらに新しいアプリ・課題（たとえば、「ゲリラ豪雨予測」）

エクサスケールに向けた プログラミングモデルの課題

Issues for exascale computing

- Important aspects of post-petascale computing
 - Large-scale system
 - $< 10^6$ nodes, for FT
 - Strong-scaling
 - $> 10\text{TFlops/node}$
 - accelerator, many-cores
 - Power limitation
 - $< 20\text{-}30\text{ MW}$



Simple relationship between
#nodes and node performance
to achieve exascale

A projection: Pre-exa, exa, post-exa

	Pre-exa	exascale	Post-exa
System performance (PF)	50~500	500~5,000	1,000~10,000
node performance (TF)	1~10	5~50	10~100
#number of node (K)	5~500	10~1,000	10~1,000
Performance/ power(GF/W)	2~20	20~200?	400?
Memory bandwidth and technology	0.5~1TB/s (HBM) 150GB/s (DDR4)	1~4TB/s (HBM)	???

- Node performance must increase! Because the system scale is limited by space and power.
- Memory performance will be limited. So, the cap between B/F will be getting worse.
- Improvement of performance/power will be difficult and limited.

Challenges of Programming Languages/models for exascale computing

- **Scalability, Locality and scalable Algorithms in system-wide**
- **Strong Scaling in node**
- **Workflow and Fault-Resilience**
- **(Power-aware)**

“MPI+X” for exascale?

- **X is OpenMP!**
- **“MPI+Open” is now a standard programming for high-end systems.**
 - I’d like to celebrate that OpenMP became “standard” in HPC programming
- **Questions:**
 - “MPI+OpenMP” is still a main programming model for exa-scale?

Question

- What happens when executing code using all cores in manycore processors like this ?

```
MPI_recv ...  
#pragma omp parallel for  
for ( ... ; ... ; ... ) {  
    ... computations ...  
}  
MPI_send ...
```

Data comes into “main shared memory”

Cost for “fork” become large

data must be taken from Main memory

Cost for “barrier” become large

MPI must collect data from each core to send

- What are solutions?
 - MPI+OpenMP runs on divided small “NUMA domains” rather than all cores?

Barrier in Xeon Phi

• Omni OpenMP

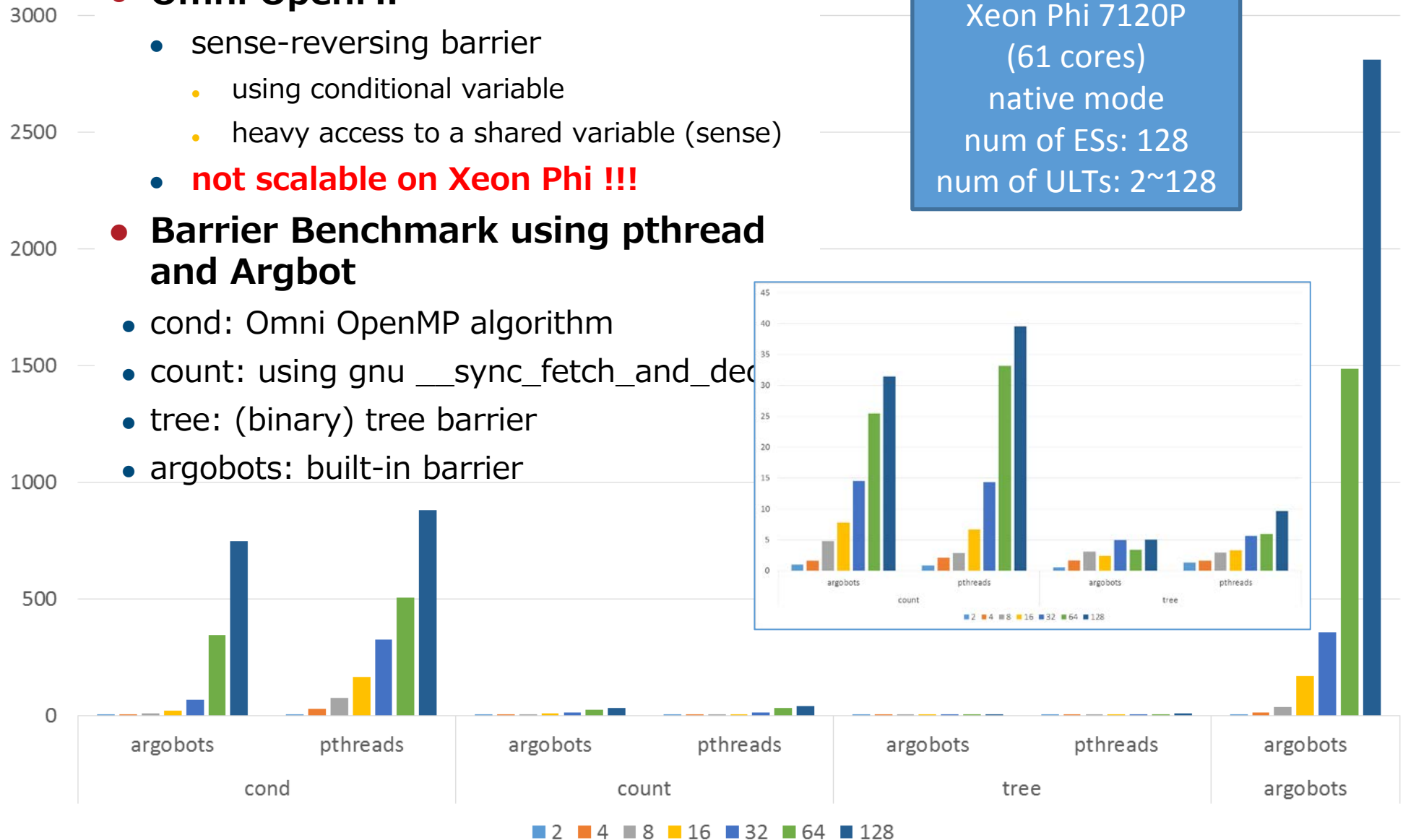
- sense-reversing barrier
 - using conditional variable
 - heavy access to a shared variable (sense)

• **not scalable on Xeon Phi !!!**

• Barrier Benchmark using pthread and Argbot

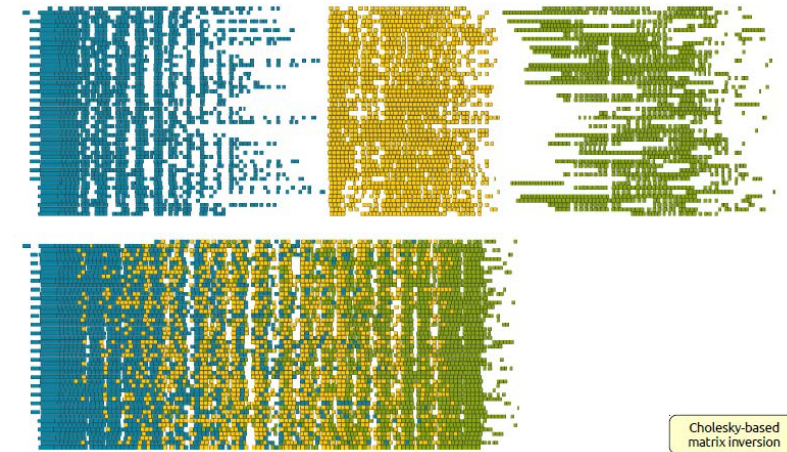
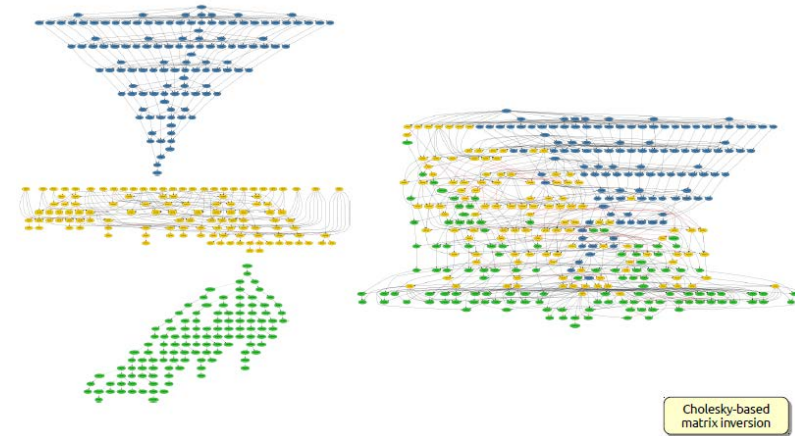
- cond: Omni OpenMP algorithm
- count: using gnu __sync_fetch_and_dec
- tree: (binary) tree barrier
- argobots: built-in barrier

Xeon Phi 7120P
(61 cores)
native mode
num of ESs: 128
num of ULTs: 2~128



Multitasking model

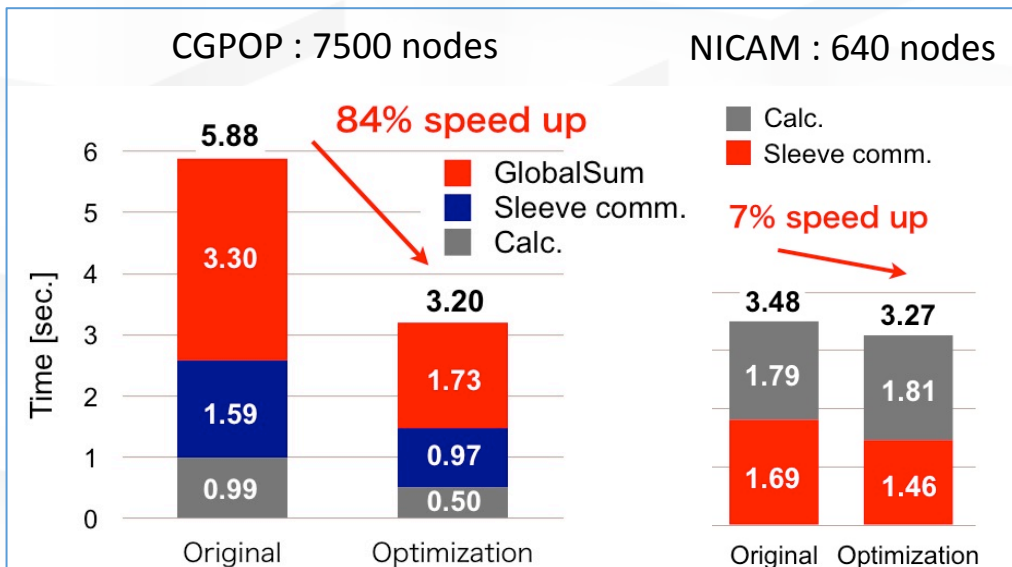
- **Multitasking/Multithreaded execution:** many “tasks” are generated/executed and communicates with each others by data dependency.
 - OpenMP task directive, OmpSS, PLASMA/QUARK, StarPU, ..
 - Thread-to-thread synchronization /communications rather than barrier
- **Advantages**
 - Remove barrier which is costly in large scale manycore system.
 - Overlap of computations and computation is done naturally.
 - New communication fabric such as Intel OPA (OmniPath Architecture) may support core-to-core communication that allows data to come to core directly.
- **New algorithms must be designed to use multitasking**



From PLASMA/QUARK slides by ICL, U. Tennessee

PGAS (Partitioned Global Address Space) models

- Light-weight one-sided communication and low overhead synchronization semantics.
- PAGES concept is adopted in Coarray Fortran, UPC, X10, XMP.
 - XMP adopts notion Coarray not only Fortran but also “C”, as “local view” as well as “global view” of data parallelism.
- Advantages and comments
 - Easy and intuitive to describe, not only one side-comm, but also strided comm.
 - Recent networks such as Cray and Fujitsu Tofu support remote DMA operation which strongly support efficient one-sided communication.
 - Other collective communication library (can be MPI) are required.



Case study of XMP on K computer
CGPOP, NICAM: Climate code

5-7 % speed up is obtained by replacing
MPI with Coarray

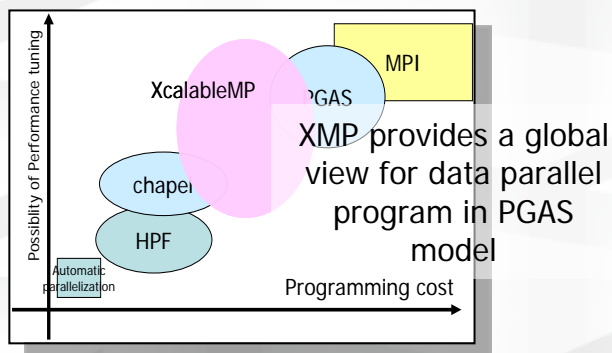
• What's XcalableMP (XMP for short)?

- 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 Nov. 2014)

- XMP Spec **Version 1.2** is available at XMP site. new features: mixed OpenMP and OpenACC , libraries for collective communications.
- Reference implementation by U. Tsukuba and Riken AICS: **Version 0.9 (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

```
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)
```

data distribution

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

add to the serial code : incremental parallelization

```
#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];
  }
}
```

work sharing and data synchronization

XcalableMP as evolutionary approach

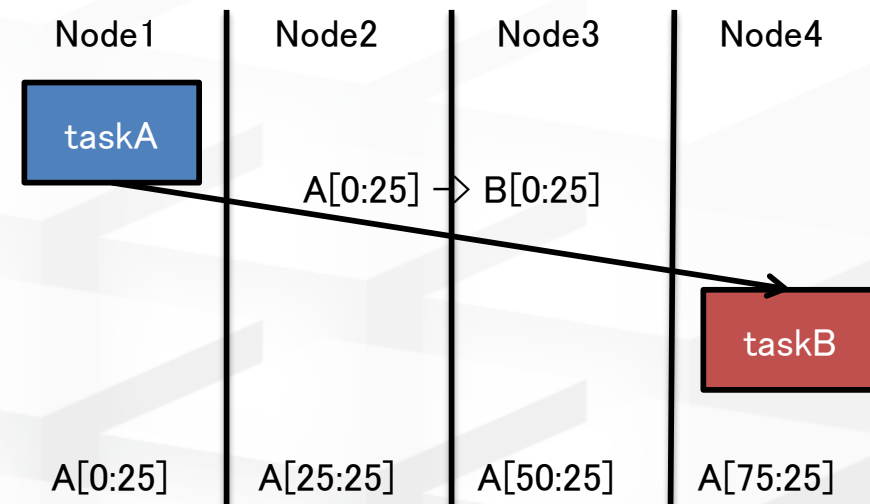
- **We focus on migration from existing codes.**
 - Directive-based approach to enable parallelization by adding directives/pragma.
 - Also, should be from MPI code. Coarray may replace MPI.
- **Learn from the past**
 - Global View for data-parallel apps. Japanese community had experience of HPF for Global-view model.
- **Specification designed by community**
 - Spec WG is organized under the PC Cluster Consortium, Japan
- **Design based on PGAS model and Coarray (From CAF)**
 - PGAS is an emerging programming model for exascale!
- **Used as a research vehicle for programming lang/model research.**
 - XMP 2.0 for multitasking.
 - Extension to accelerator (XACC)

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
  
```



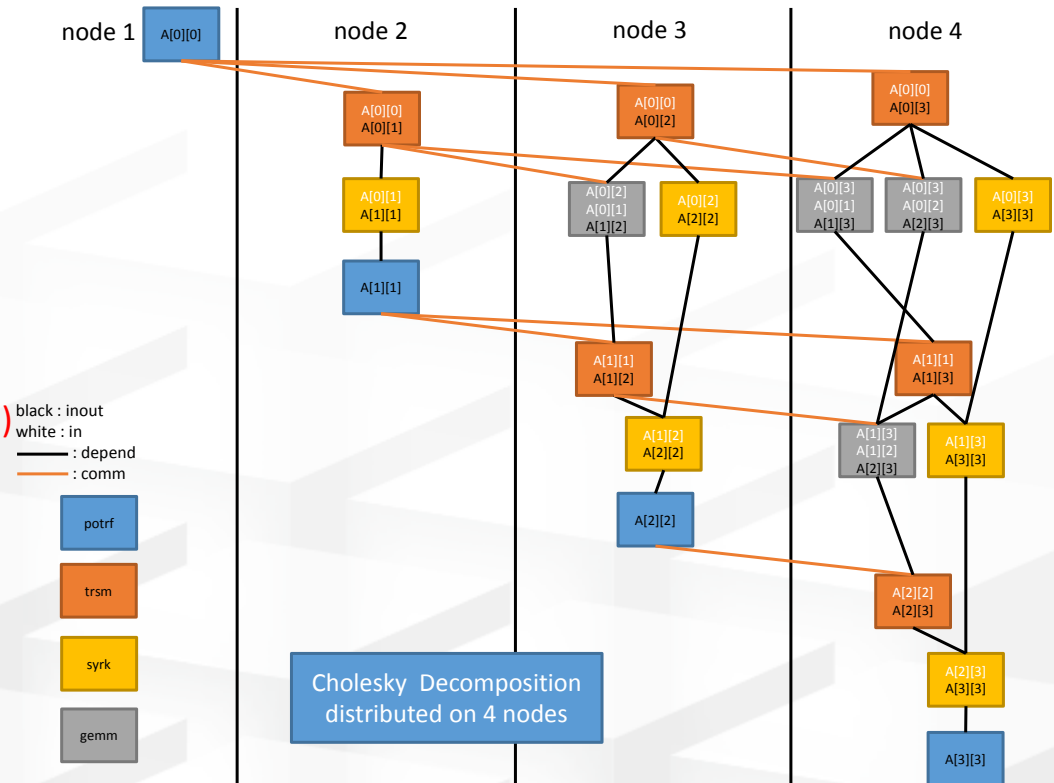
Proposal of Tasklet directive

- The detail spec of the directive is under discussion in spec-WG
- Currently, we are working on prototype implementations and preliminary evaluations
- Example: Cholesky Decomposition

```
double A[nt][nt][ts*ts], B[ts*ts], C[nt][ts*ts];
#pragma xmp node P(*)
#pragma xmp template T(0:nt-1)
#pragma xmp distribute T(cyclic) onto P
#pragma xmp align A[*][i][*] with T(i)
```

```
for (int k = 0; k < nt; k++) {
  #pragma xmp tasklet inout(A[k][k], T(k+1:nt-1))
  omp_potrf (A[k][k], ts, ts);

  for (int i = k + 1; i < nt; i++) {
    #pragma xmp tasklet in(B, T(k)) inout(A[k][i], T(i+1:nt-1))
    omp_trsm (B, A[k][i], ts, ts);
  }
  for (int i = k + 1; i < nt; i++) {
    for (int j = k + 1; j < i; j++) {
      #pragma xmp tasklet in(A[k][i]) in(C[j], T(j)) inout(A[j][i])
      omp_gemm (A[k][i], C[j], A[j][i], ts, ts);
    }
    #pragma xmp tasklet in(A[k][i]) inout(A[i][i])
    omp_syrk (A[k][i], A[i][i], ts, ts);
  }
}
#pragma xmp taskletwait
```



Strong Scaling in node

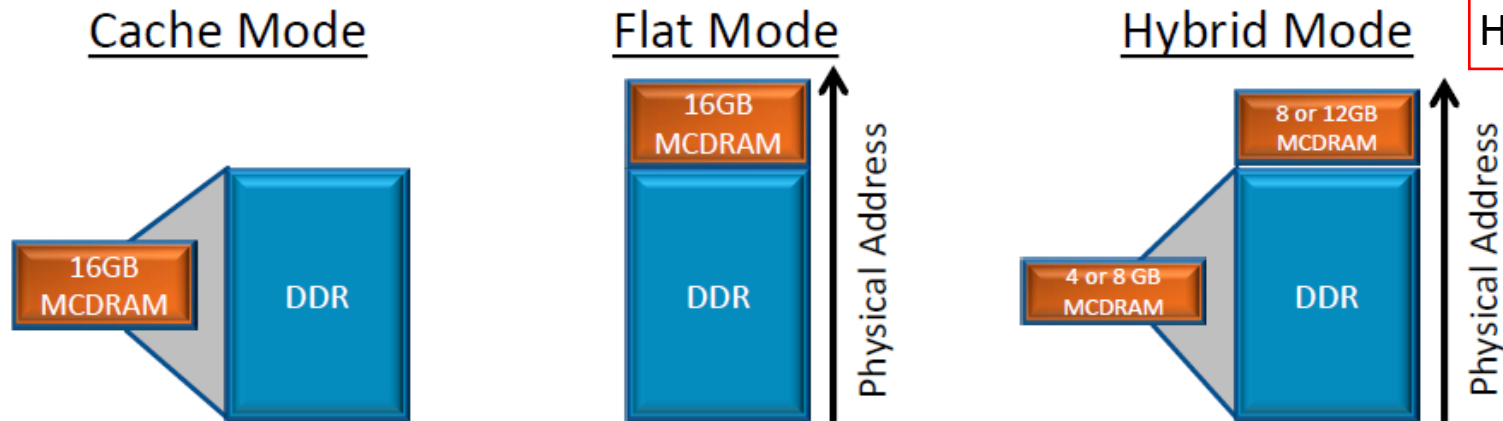
- **Two approaches:**
 - SIMD for core in manycore processors
 - Accelerator such as GPUs
- **Programming for SIMD**
 - Vectorization by directives or automatic compiler technology
 - Limited bandwidth of memory and NoC
 - Complex memory system: Fast-memory (MD-DRAM, HBM, HMC) and DDR , VMRAM
- **Programming for GPUs**
 - Parallelization by OpenACC/OpenMP 4.0. Still immature but getting matured soon
 - Fast memory (HMB) and fast link (NV-Link): similar problem of complex memory system in manycore.
 - Programming model to be shared by manycore and accelerator for high productivity.

How to use MC-DRAM in KNL?

- New Xeon Phi (KNL) has fast memory called MC-DRAM.
 - KNL performance: < 5 TF (Theoretical Peak)
 - DDR4: 100~200 GB/s, MC-DRAM: 0.5 TB/s
 - How to use?

Memory Modes

Three Modes. Selected at boot



From Intel Slide
presented at
HotChips 2015

- SW-Transparent, Mem-side cache
- Direct mapped. 64B lines.
- Tags part of line
- Covers whole DDR range

- MCDRAM as regular memory
- SW-Managed
- Same address space

- Part cache, Part memory
- 25% or 50% cache
- Benefits of both

XcalableACC(ACC) = XcalableMP+OpenACC

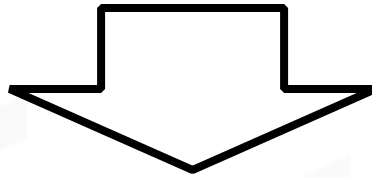
● Extension of XcalableMP for GPU

- A project of U. Tsukuba leaded by Prof. Taiuske Boku
- “vertical” integration of XcalableMP and OpenACC
 - Data distribution for both host and GPU by XcalableMP
 - Offloading computations in a set of nodes by OpenACC
- Proposed as unified parallel programming model for many-core architecture & accelerator
 - GPU, Intel Xeon Phi
 - OpenACC supports many architectures

Source Code Example: NPB CG

```
#pragma xmp nodes p(NUM_COLS, NUM_ROWS)
#pragma xmp template t(0:NA-1,0:NA-1)
#pragma xmp distribute t(block, block) onto p
#pragma xmp align w[i] with t(*,i)
#pragma xmp align q[i] with t(i,*)
double a[NZ];
int rowstr[NA+1], colidx[NZ];
...
#pragma acc data copy(p,q,r,w,rowstr[0:NA+1]¥
                        , a[0:NZ], colidx[0:NZ])
{
    ...
    #pragma xmp loop on t(*,j)
    #pragma acc parallel loop gang
    for(j=0; j < NA; j++){
        double sum = 0.0;
        #pragma acc loop vector reduction(+:sum)
        for (k = rowstr[j]; k < rowstr[j+1]; k++)
            sum = sum + a[k]*p[colidx[k]];
        w[j] = sum;
    }
    #pragma xmp reduction(+:w) on p(:,*) acc
    #pragma xmp gmove acc
    q[:] = w[:];
    ...
} //end acc data
```


- Petascale system was targeting some of “capability” computing.
- In exascale system, it become important to execute huge number of medium-grain jobs for parameter-search type applications.

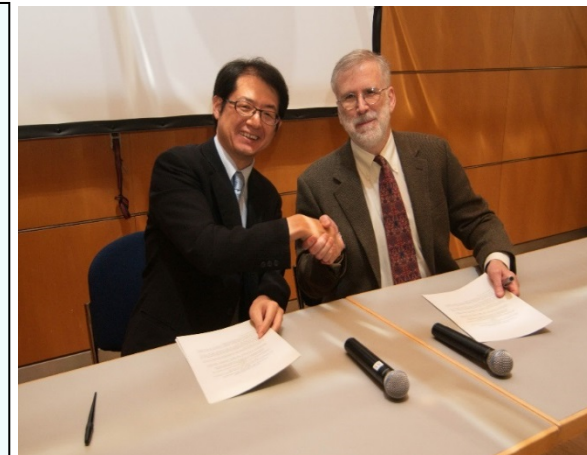


Workflow to control and collect/process data is important, also for “big-data” apps.

International Collaboration between DOE and MEXT

PROJECT ARRANGEMENT
UNDER THE IMPLEMENTING ARRANGEMENT
BETWEEN
THE MINISTRY OF EDUCATION, CULTURE, SPORTS, SCIENCE AND TECHNOLOGY OF JAPAN
AND
THE DEPARTMENT OF ENERGY OF THE UNITED STATES OF AMERICA
CONCERNING COOPERATION IN RESEARCH AND DEVELOPMENT IN ENERGY AND RELATED
FIELDS

CONCERNING COMPUTER SCIENCE AND SOFTWARE RELATED TO CURRENT AND FUTURE
HIGH PERFORMANCE COMPUTING FOR OPEN SCIENTIFIC RESEARCH



Yoshio Kawaguchi (MEXT, Japan)
and William Harrod (DOE, USA)

Purpose: Work together where it is mutually beneficial to expand the HPC ecosystem and improve system capability

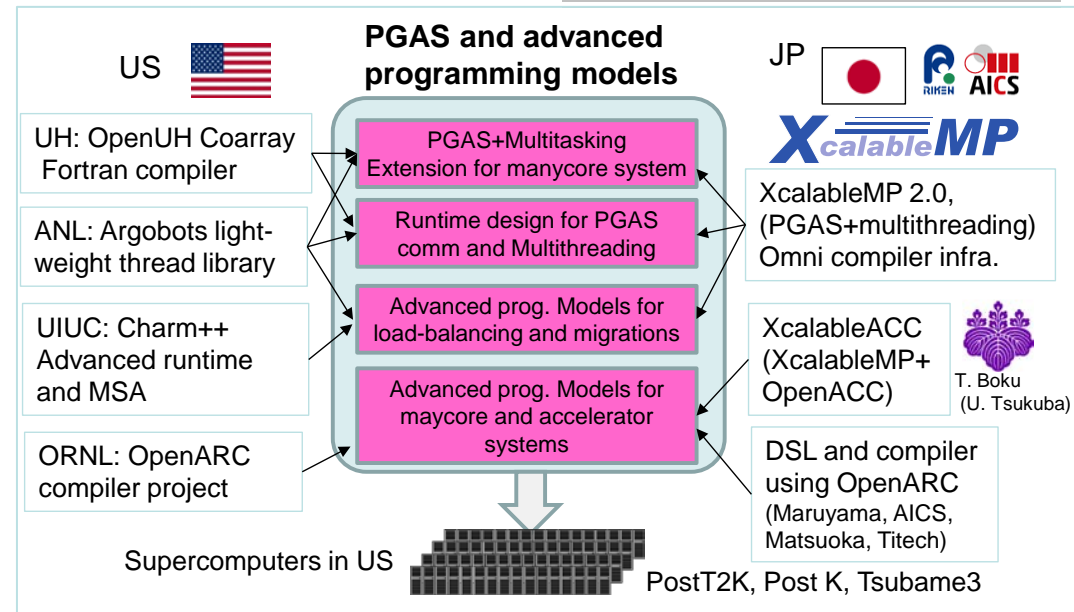
- Each country will develop their own path for next generation platforms
- Countries will collaborate where it is mutually beneficial
- Joint Activities
 - Pre-standardization interface coordination
 - Collection and publication of open data
 - Collaborative development of open source software
 - Evaluation and analysis of benchmarks and architectures
 - Standardization of mature technologies

Technical Areas of Cooperation

- Kernel System Programming Interface
- Low-level Communication Layer
- Task and Thread Management to Support Massive Concurrency
- Power Management and Optimization
- Data Staging and Input/Output (I/O) Bottlenecks
- File System and I/O Management
- Improving System and Application Resilience to Chip Failures and other Faults
- Mini-Applications for Exascale Component-Based Performance Modelling

PGAS and Advanced programming models for exascale systems

- Coordinators
 - US: P. Beckman (ANL), JP: M. Sato (RIKEN)
- Leaders
 - US: L. Kale (UIUC), B Chapman (U Huston), J. Vetter (ORNL), P. Balaji (ANL)
 - JP: M Sato (RIKEN)
- Collaborators
 - S. Seo (ANL), D Bernholdt (ORNL), D. Eachempati(UH)
 - H. Murai (RIKEN), J. Lee (RIKEN), N. Maruyama (RIKEN), T. Boku (U. Tsukuba)
- Collaboration topics
 - Extension of PGAS (Partitioned Global Address Space) model with language constructs of multitasking (multithreading) for manycore-based exascale systems
 - Runtime design for PGAS communication and multitasking
 - Advanced programming models to support both manycore-based and accelerator-based exascale system for high productivity.
 - Advanced programming models for dynamic load-balancing and migration in exascale systems
- How to collaborate
 - Twice meetings per year
 - Student / young researchers exchange, sharing codes
 - Funding:
 - US: ARGO, X-stack(XPRESS), X-stack(Vancouver, ARES)
 - JP: FLAGSHIP 2020, PP-CREST (JP)



- Deliverables
 - Concepts for PGAS and multithreading integration for manycore-based exascale systems.
 - Concepts for advanced programming model to be shared by both manycore and accelerators-based systems.
 - Pre-standardization of Application Programming Interface for multithreading (based on Argobots) and PGAS
- Recent activities and plans
 - AICS teams visited UH, UIUC and ANL for discussions.
 - Start using Argobots for Omni OpenMP compiler and produced preliminary results on intel Xeon Phi.
 - AICS invited Post-doc from UH for collaborations on PGAS
 - ORNL visited AICS to have a meeting for the collaboration
 - JP (AICS, Tsukuba) will send Post-doc and students to ANL and UH, ORNL
 - JP and ORNL will have a meeting in JP or US how to collaborate.