計算科学研究センターにおける スーパーコンピュータの2つの取り組み: 演算加速器系と汎用プロセッサ系

朴　泰祐

筑波大学計算科学研究センター

HPC研究部門主任／副センター長

http://www.hpcs.cs.tsukuba.ac.jp/~taisuke/

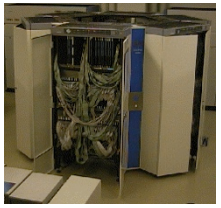
超並列計算機PAX(PACS)の開発の歴史

- 1977年に研究開始（星野・川合）
- 1978年に第一号機が完成
- 1996年のCP-PACSはTOP500第一位

1978
第1号機PACS-9

1980
第2号機PAXS-32

1989
第5号機QCDPAX



1996
世界最高速を達成した
第6号機CP-PACS

2006
バンド幅重視クラスタ
PACS-CS

2012~2013
GPU演算加速クラスタ
HA-PACS

完成年	名称	性能
1978年	PACS-9	7 KFLOPS
1980年	PACS-32	500 KFLOPS
1983年	PAX-128	4 MFLOPS
1984年	PAX-32J	3 MFLOPS
1989年	QCDPAX	14 GFLOPS
1996年	CP-PACS	614 GFLOPS
2006年	PACS-CS	14.3 TFLOPS
2012~13年	HA-PACS	1.166 PFLOPS
2014年	COMA (PACS-IX)	1.001 PFLOPS

- 計算科学者＋計算機工学者の共同開発による「実行性能重視スパコン」
- Application-drivenな開発
- 持続的な開発による経験の蓄積

この他：科研費特別推進研究によるハイブリッドクラスタ FIRST

CCSにおいて現在運用中の2系列のスパコン

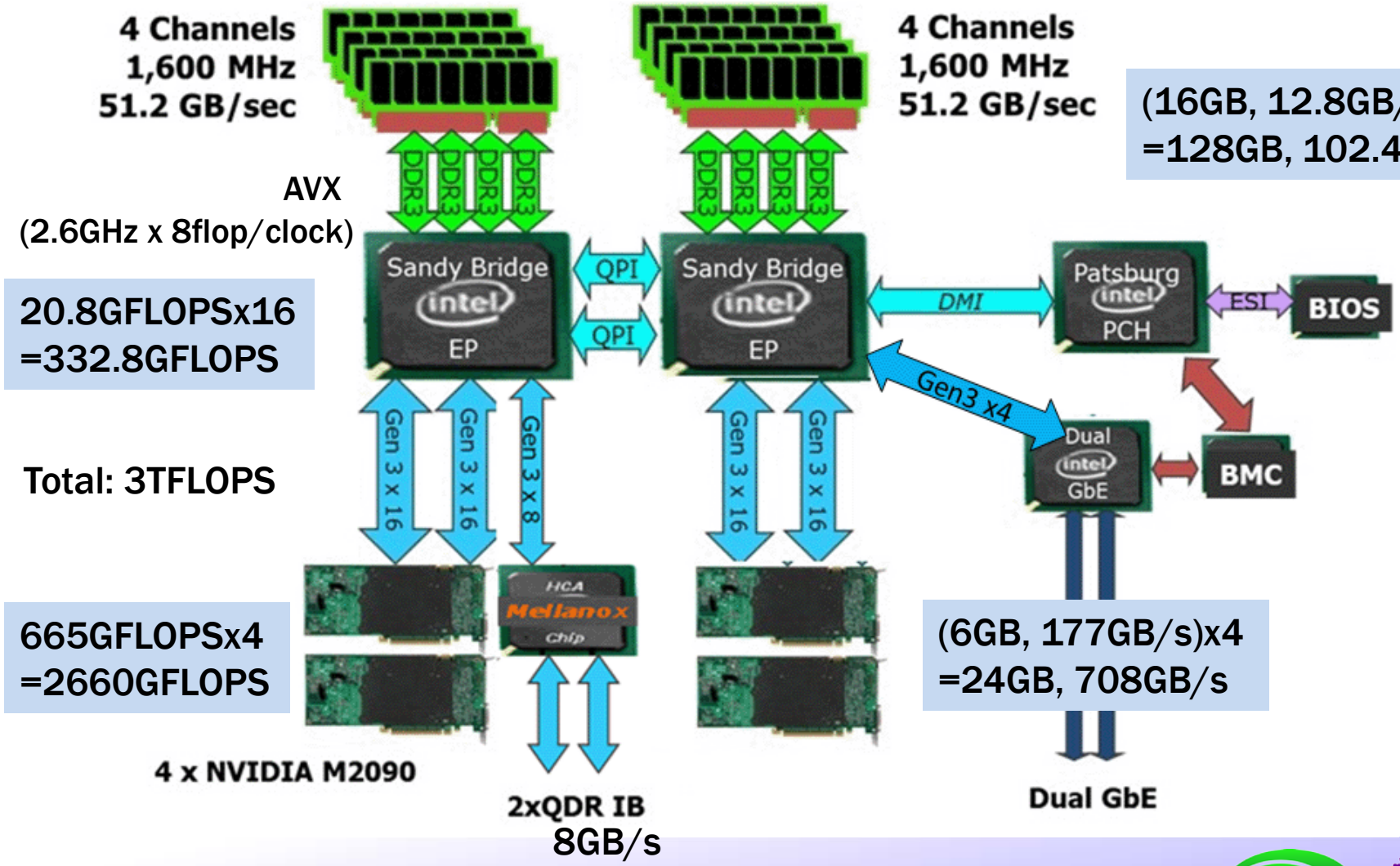
- HA-PACS (PACS-VIII)
  - GPU cluster、一部に独自開発のGPU間直接通信ネットワーク(TCA)を実装
  - PFLOPSクラスのGPUクラスタにより accelerated computing によるコード開発とプロダクトランを実施
  - novice user よりも professional user をターゲットとした先進的マシン

- COMA (PACS-IX)
  - Many Core cluster
  - PFLOPSクラスの many core architecture クラスタにより many core processor の性能特性を理解しつつ次世代の many core 向けコード開発とプロダクトランを実施
  - 筑波大・東大で共同設置した「最先端共同HPC基盤施設 (JCAHPC)」において2015年度に導入予定の many core システムの先行研究と各種テスト実施
  - 汎用並列処理向け、比較的広いユーザ層を対象
    ➟ 現在、MICは一種の演算加速器だがその本質は汎用プロセッサ

HA-PACS Base Cluster

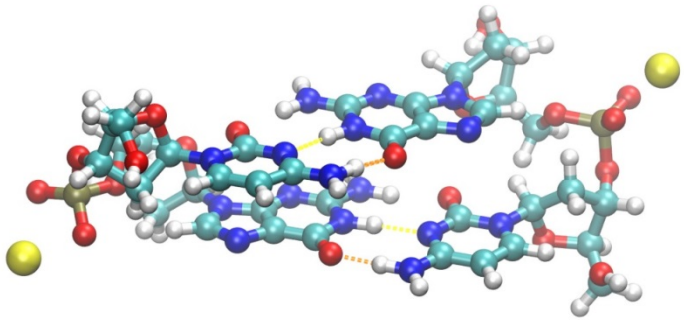
- Appro International 社
- CCSとして初めてGPUを本格採用したクラスタ
- 268ノード（2 Xeon E5-2670 +
  4 Fermi M2090)
- Mellanox IniniBand QDR x 2rail, Fat Tree
- File Server: DDN
  500 TB (RAID6+Lustre)
- 802 TFLOPS
  (HPL: 421 TFLOPS)
- TOP500 #41
  （国内 #5）

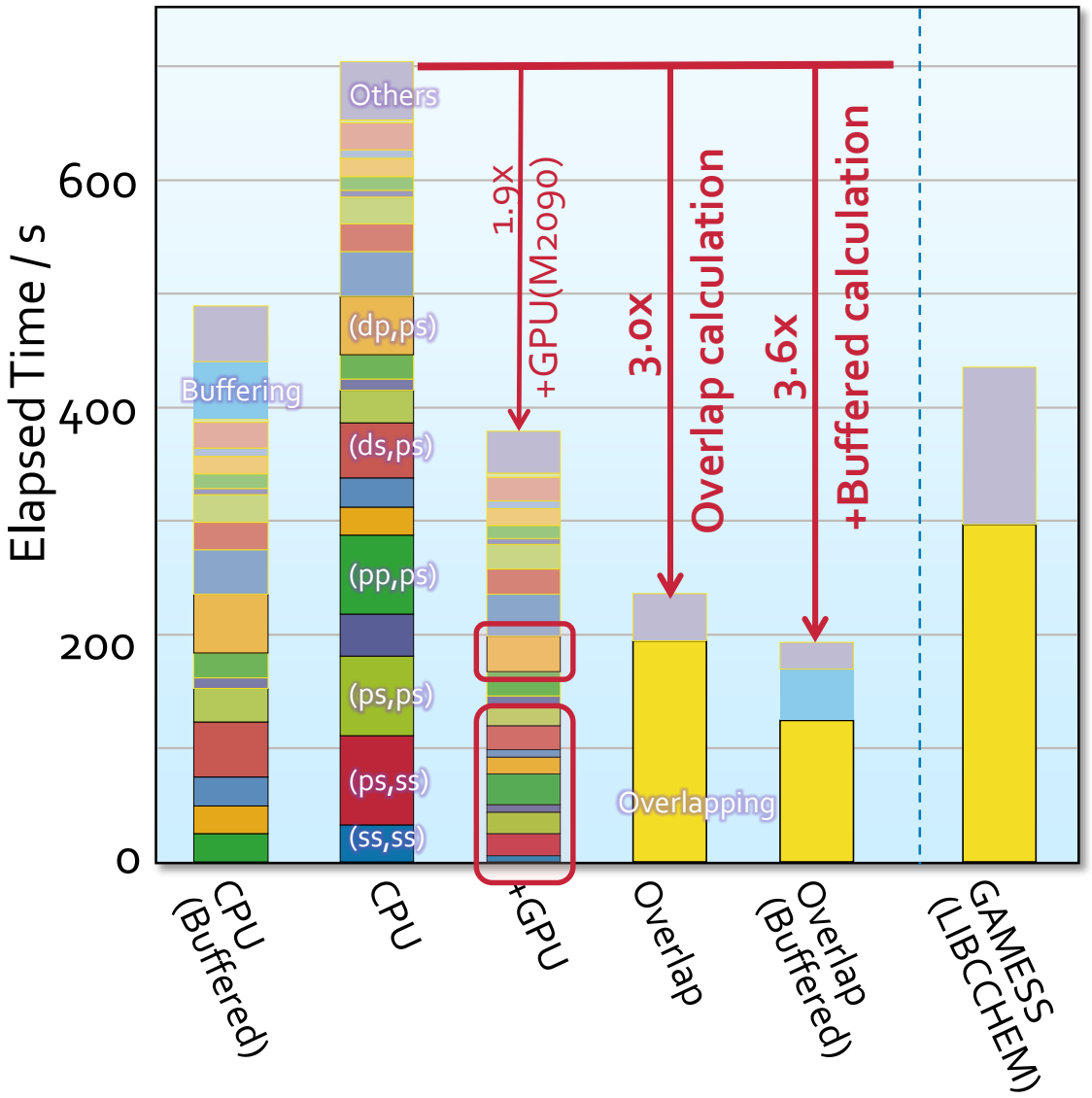
Computation node of Base Cluster



HA-PACSの成果例：HF計算の性能評価

- Model DNA (CG)2
  - HF/6-31G(d)
  - 126 atom, 1,208 AO
  - 14 SCF iterations
- HA-PACS 1node
  - 16 CPU cores
    - Intel SandyBridge-E5, 2.6GHz
  - 4 GPU(NVIDIA M2090)
- Software
  - OpenFMO
  - GAMESS
    - Version: 1 MAY 2013 (R1)
    - GPU support (LIBCCHEM)



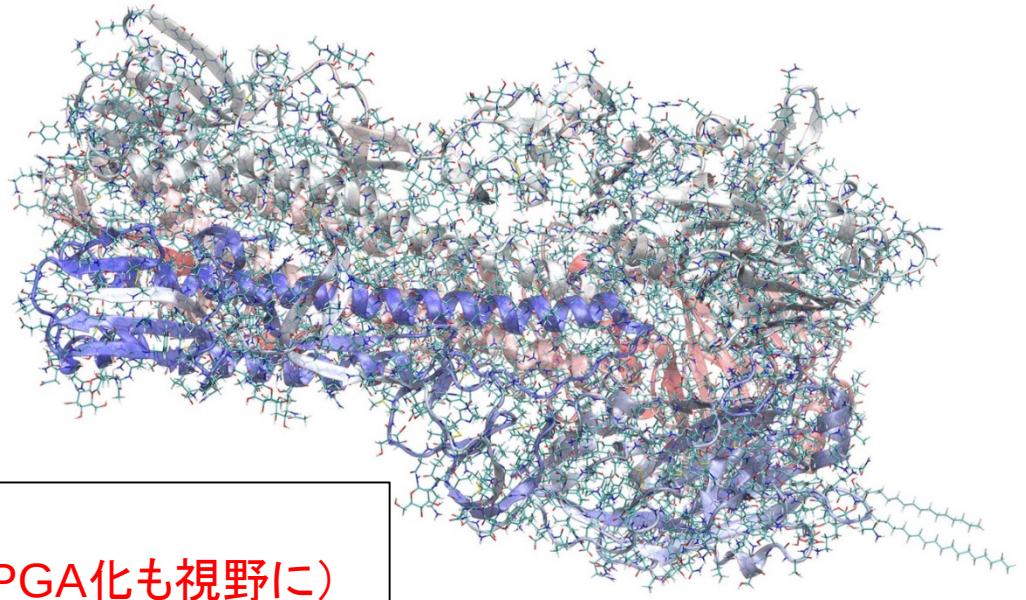


フラグメント分子軌道法のGPU化

- インフルエンザHA3タンパク質 (23,460原子, 721フラグメント)
- FMO-HF/6-31G(d)
- HA-PACS ベースクラスタ 64ノード
  - 1024 CPU core + 256 GPU
- 84ワーカグループ
  - 3 MPI ランク/ワーカグループ

- 京コンピュータ24,576 ノード (3.1PFLOPS)で11分
- 今回: HA-PACSベースクラスタ64ノード(386TFLOPS)で120分

	HA3
#nodes (#GPU)	64 (256)
SCC [hr]	0.52
Dimer SCF [hr]	0.90
ES Dimer [hr]	0.45
Total [hr]	1.97

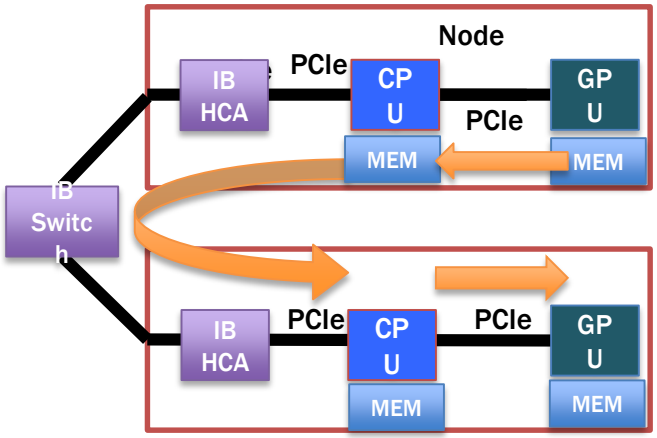


今後の課題

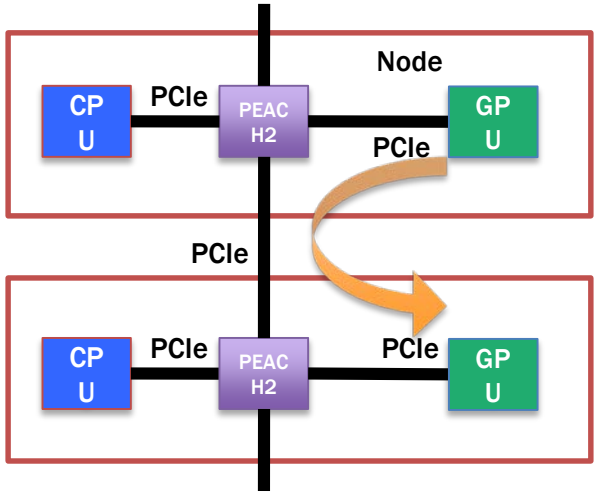
電子相関手法のGPU化(+TCA, FPGA化も視野に)

HA-PACSにおけるシステム研究

- TCA (Tightly Coupled Accelerators)
  - PCIe によるAD間直接通信(ノード内・ノード間)を実現
  - PEACH2チップ(FPGAによるプロトタイプ)によるインテリジェントなPCIeスイッチ＋コントローラ
  - ホストCPU・メモリ・結合網に依存しないAD間直接通信



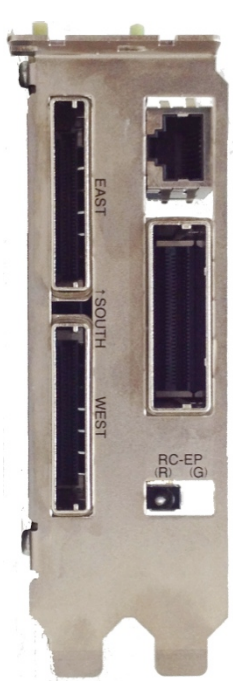
通常のノード間GPU間通信



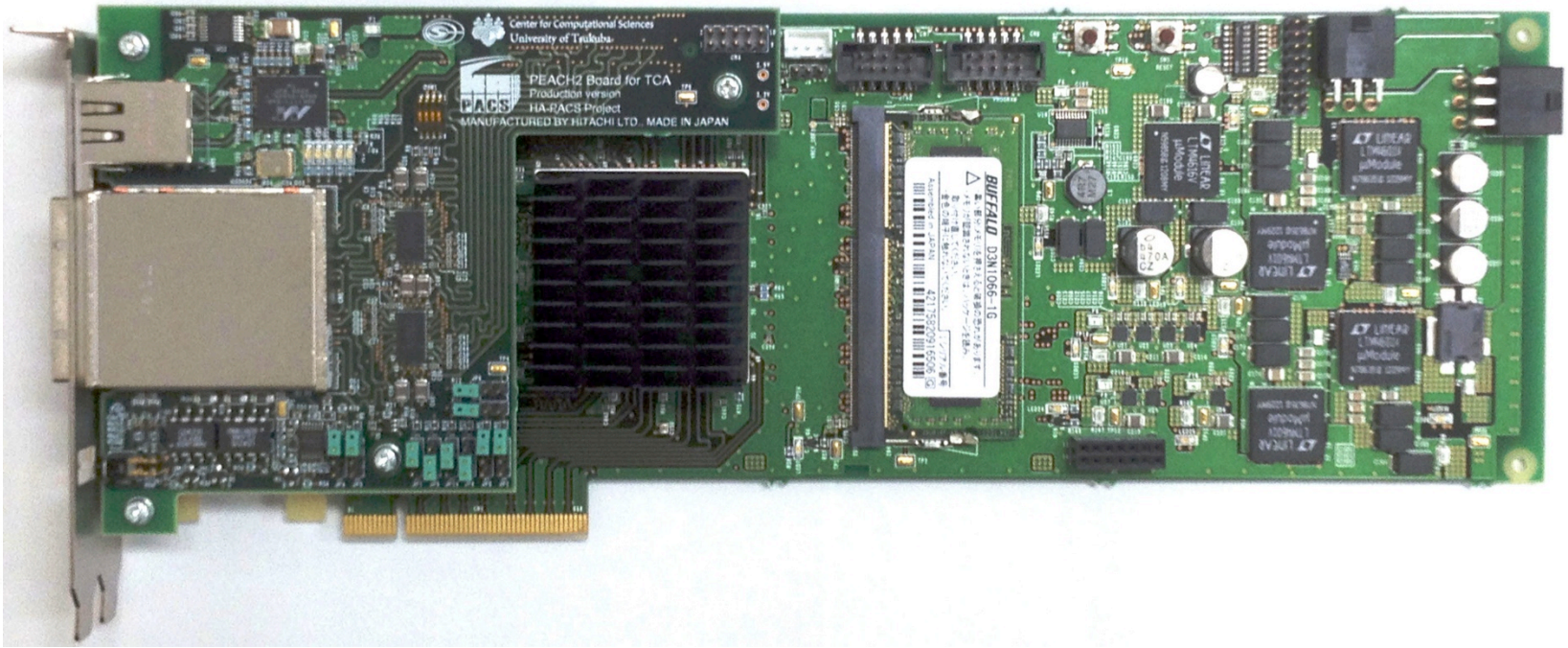
TCAによるノード間GPU間通信

PEACH2 board

- PCI Express Gen2 x8 peripheral board
  - Compatible with PCIe Spec.

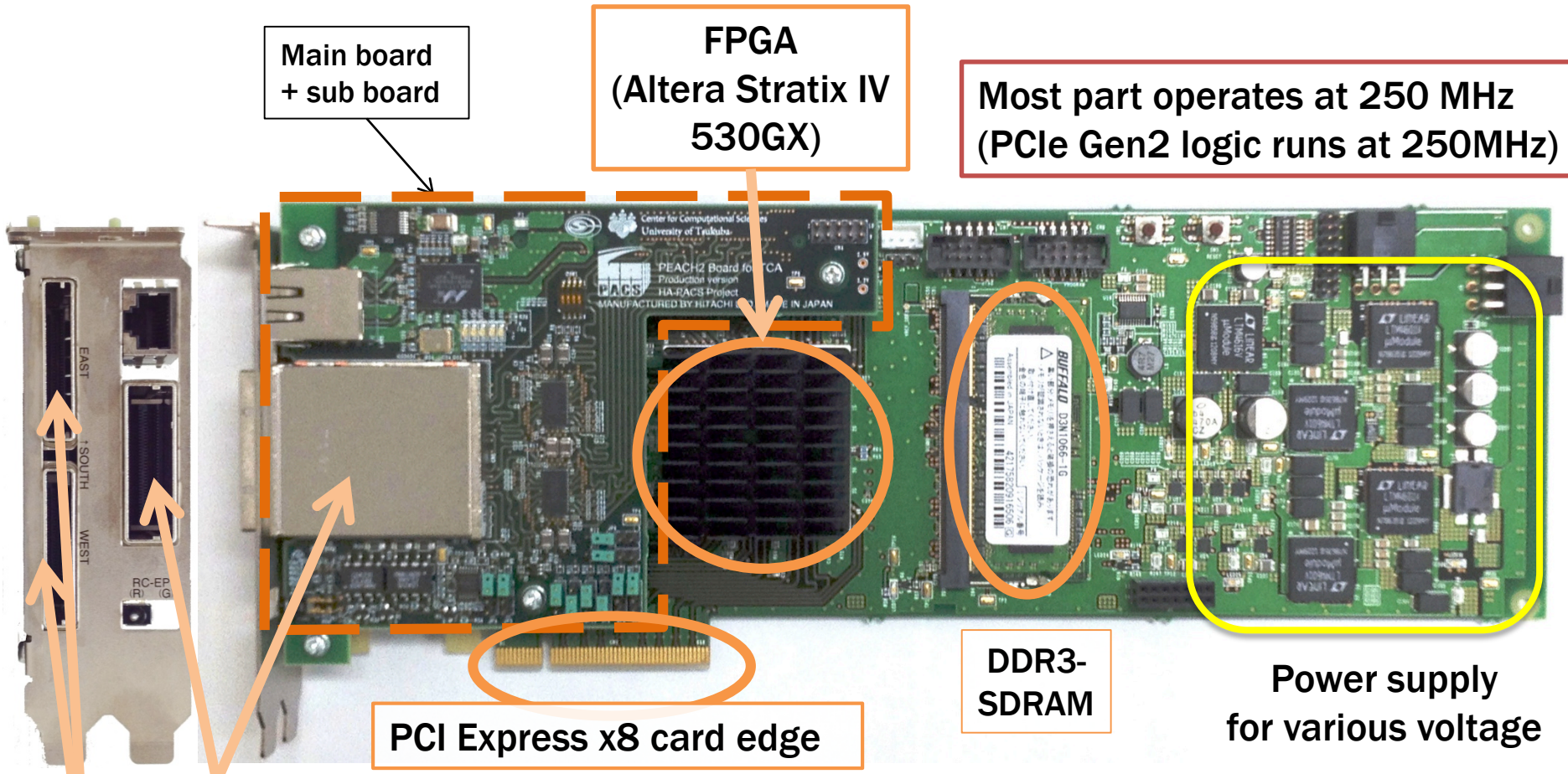


Side View



Top View

PEACH2 board

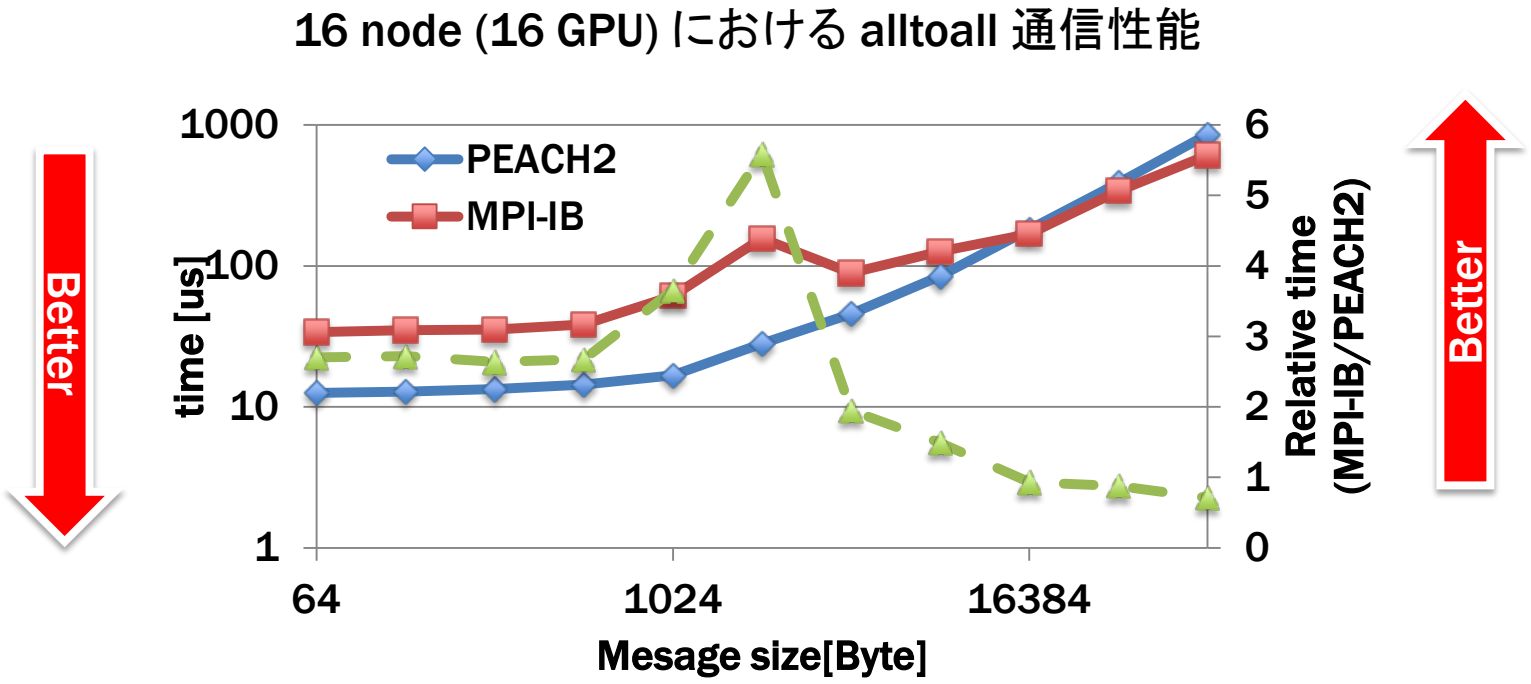


HA-PACS Base Cluster + TCA
(2013/11稼働開始)

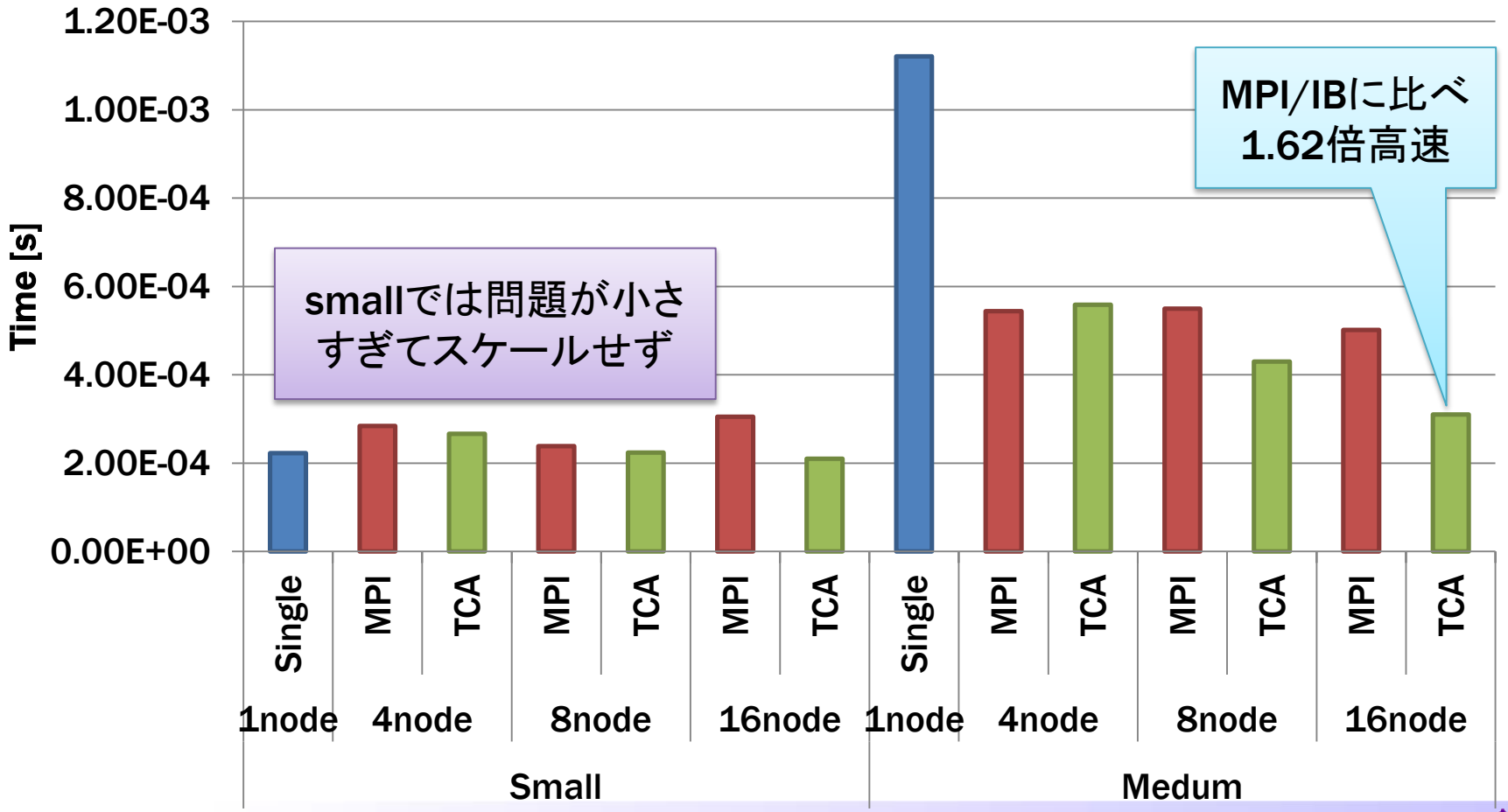
- HA-PACS Base Cluster = 2.99 TFlops x 268 node = 802 TFlops
- HA-PACS/TCA = 5.69 TFlops x 64 node = 364 TFlops (Green500 #3)
- TOTAL: 1.166 PFlops

FFTEベンチマーク：alltoall通信

- 高橋@筑波大のFFTE (6 step FFT)のPCクラスタ版をTCAに対応させたベンチマーク
- TCAによるalltoall通信をDMA chainingと同一GPU内のデータコピーにもcudaMemcpyAsync()ではなくPEACH2を使うことにより、GPU内データ移動をより高速化



FFTEの性能 (Small=2^{14}, Medium=2^{16})



今後の演算加速器系システム

- MICは過渡的に演算加速器であるが基本的にはx86互換の汎用プロセッサ
  - メニーコアであるが各コアは比較的非力であり、浮動小数点SIMD命令の強化でピーク性能を上げている
  - 単純なOpenMPプログラミンでは高速化は難しい（後述）
- 依然として演算加速器の本命はGPU
  - NVIDIAを中心としてGPUの高性能化が続いている
  - 電力は徐々に上がっているがFLOPS/Wは下がり続けている
  - これだけで十分か？

今後の演算加速器系システム(続き)

- 「頭でっかち」になりがちなGPUを今後どのように有効利用していくか
  - GPU単体のsustained performance向上
  - ノード間通信性能をより向上させる必要がある
  - バンド幅だけでなくレイテンシの削減が重要
    ➟ strong scalability
- TCA (Tightly Coupled Accelerators)コンセプトはこの問題に対する一つの解
  - 今後、演算加速器はそれ自体の持つ超並列通信網で結合されるのが望ましい
    ➟ ホストCPUからの脱却
  - とはいえ、ホストの助けなしにGPUは自立できない

FPGAを積極的に利用した演算加速器システム

- GPUと並列処理ネットワークの統合
  - TCA/PEACH2のFPGA実装
  - 当初はASIC開発を避け、実装の柔軟性のためFPGAを採用
  - より積極的なFPGA利用へ➟ FPGA offloading
- ノード内計算において…
  - 演算的には単純だがGPU処理に向かない（ポインタがある、分岐が多い等）
  - 多様な演算精度：倍、単、半、四半
  - ノード間通信に直接関連するデータを on the fly で処理したい

➟ 従来のGPU+ネットワークでは速度向上、strong scalingが不十分！

アプリケーションFPGAオフローディング（宇宙物理部門＋慶應義塾大学との共同研究）

PEACH2によるオフローディング：宇宙物理学の重力計算において、LET(Locally Essential Tree)を on-the-flyで作成

- 幅優先探索によるTree code on GPU を実装
  - 扇谷コード，中里コードよりも高速
  - まだ高速化の余地がある
- 現在実装中の内容
  - GPU 版 tree make
  - 近傍粒子探索＋PH-key jump の動的決定
- 並列版はもう少し調整が必要
  - 計算時間の比を用いて各GPUに割り当てる粒子数を設定しているが，MPIの実行時間だけをうまく省かないといけない

LET (Locally Essential Tree) 作成

mask[id]配列を用意

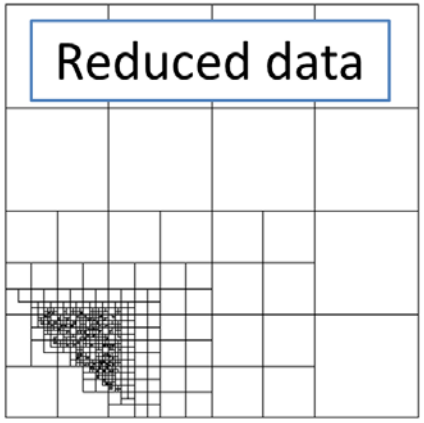
mask[id] == 0　間引く

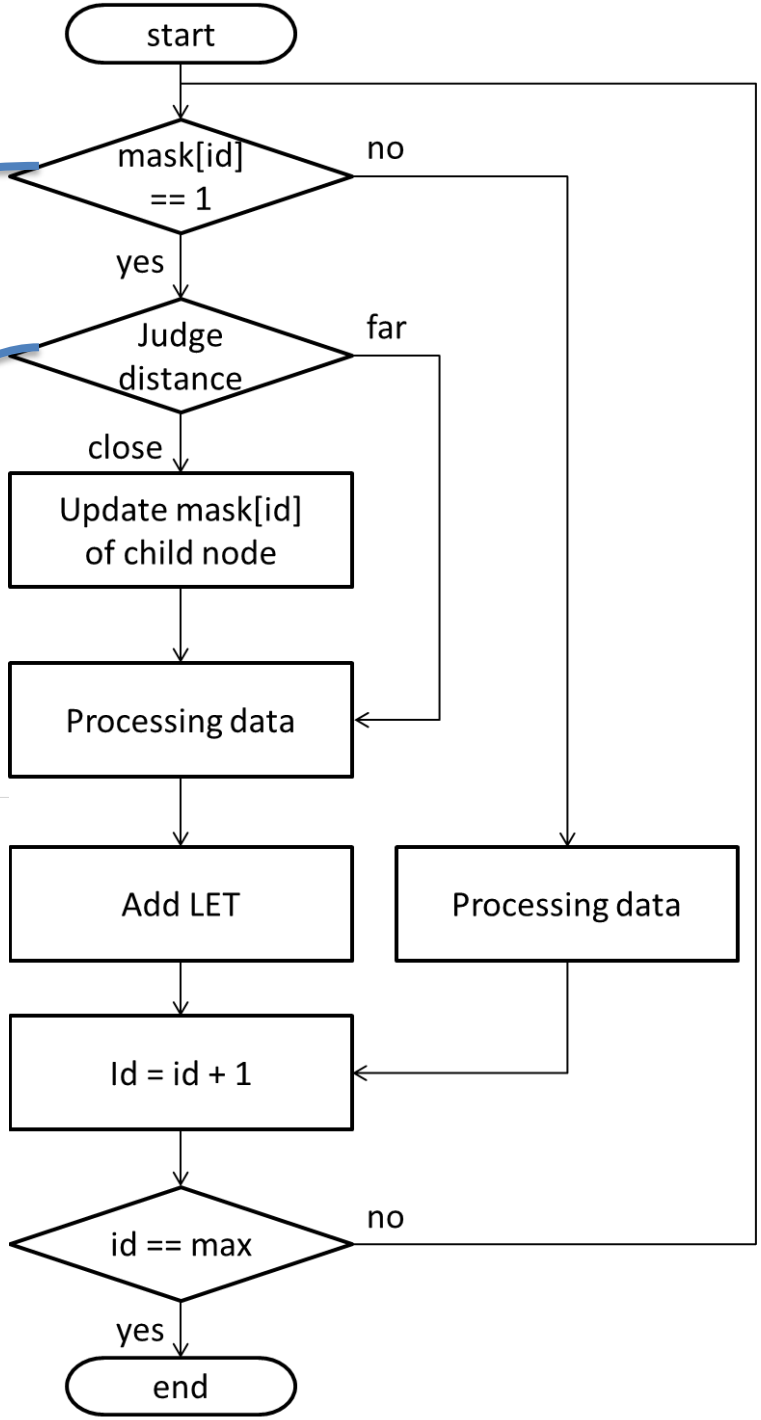
mask[id] == 1　LETに追加

距離判定

受信側の領域データの一部と

送信側の各セルにて距離判定



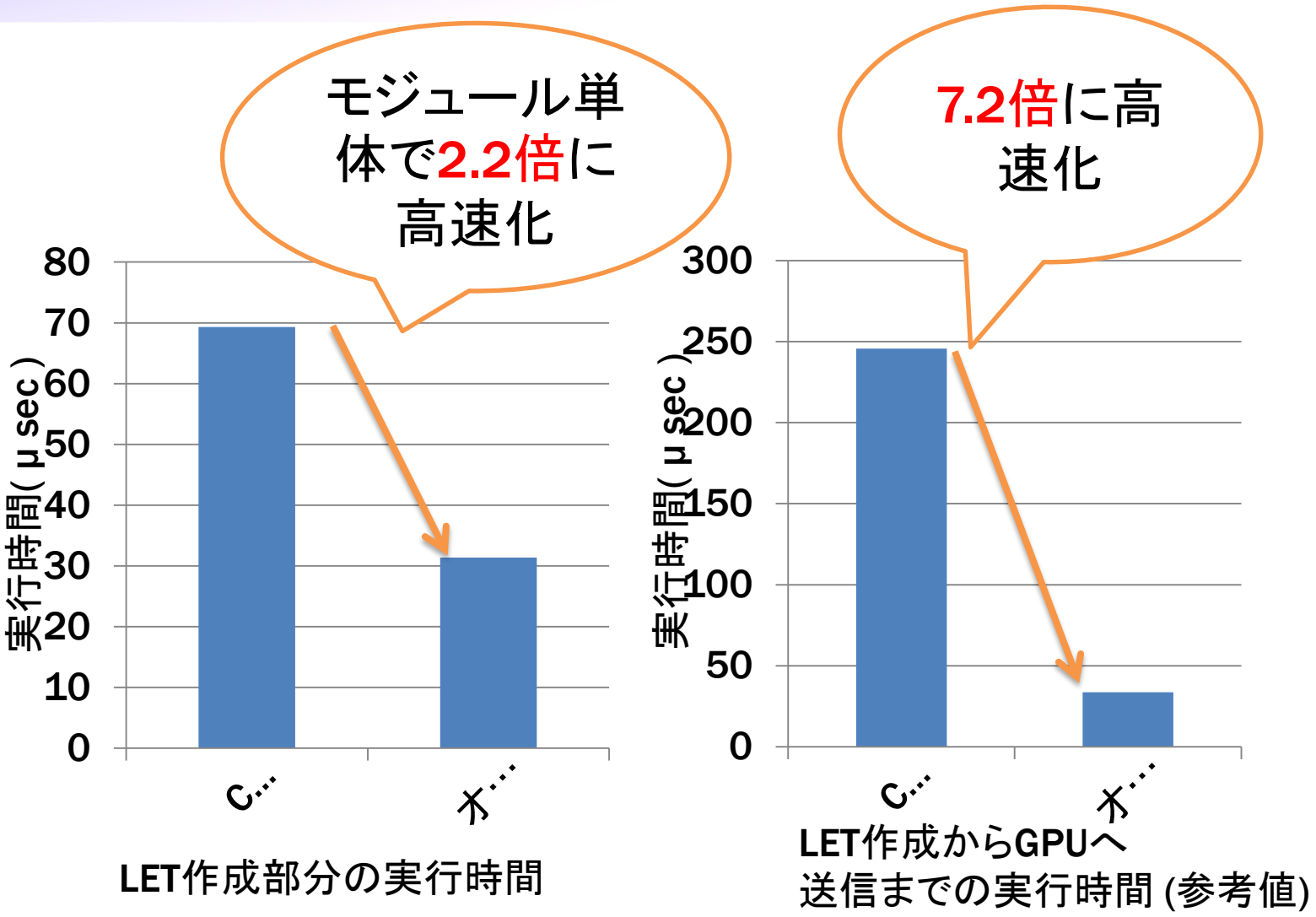


FPGA Gate Usage

	PEACH2 Used (%)	LET generator & PEACH2 Used (%)
Logic utilization	46 %	67 %
Combinational ALUTs	65665 (28 %)	74561 / 232960 (32 %)
Dedicated logic registers	83690 (36 %)	122714 / 232960 (53 %)
Total block memory bits	2964560 (21 %)	2744448 / 13934592 (20 %)
DSP block elements	4 (<1 %)	36 / 832 (4 %)

※ PEACH2 has enough logic for LET generator.

性能評価


モジュール単体で2.2倍に高速化
7.2倍に高速化
実行時間(μ sec)
80
70
60
50
40
30
20
10
0
300
250
200
150
100
50
0
C…
不…
LET作成部分の実行時間
LET作成からGPUへ送信までの実行時間 (参考値)

FPGA offloading with communication

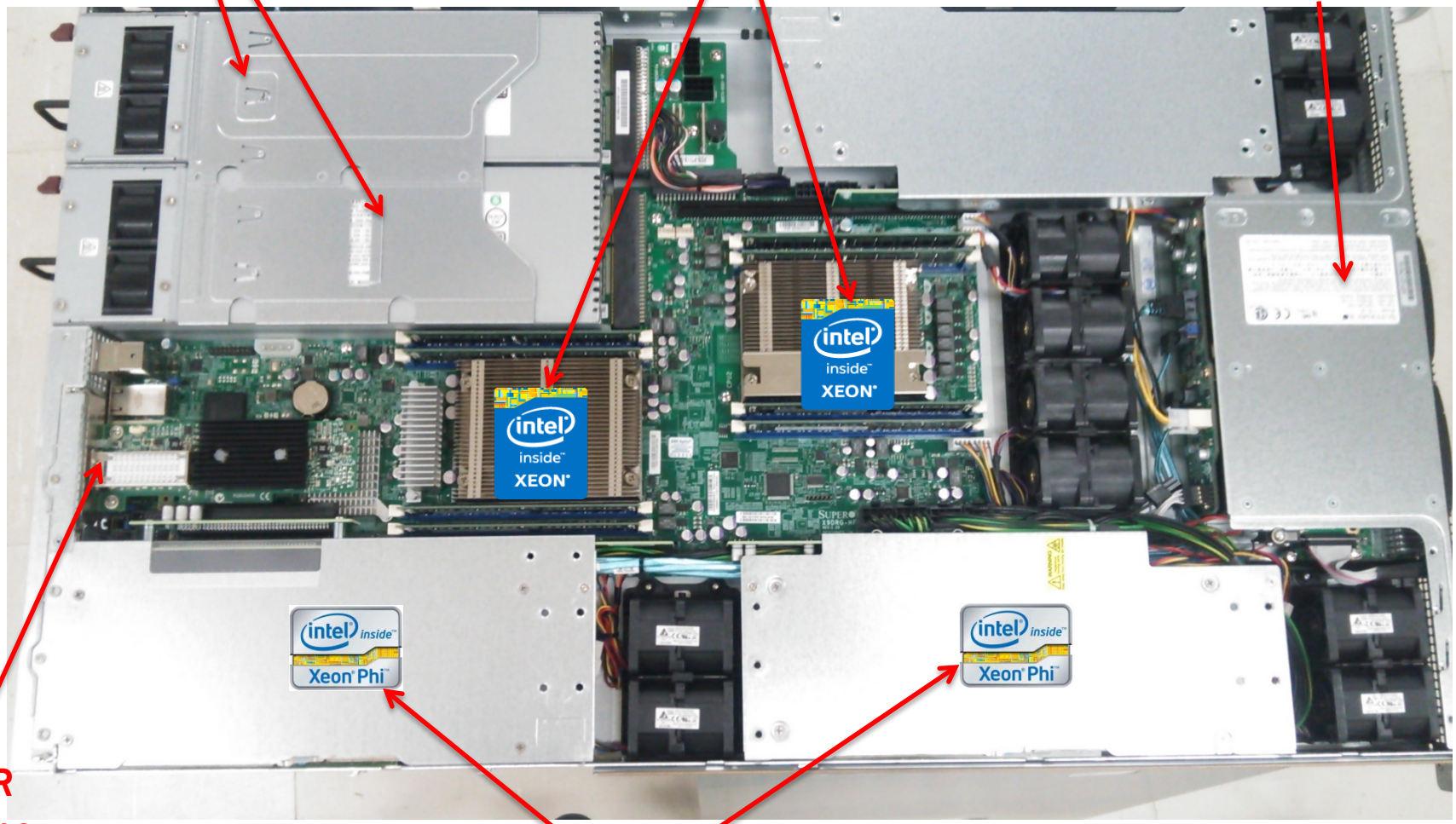
- 新しい演算加速器系システム
  - GPU + FPGA + short latency network
  - 現在 JST/CREST においてPEACH2実装用FPGA上で研究を展開中
  - PEACH2 (PEACH3) ネットワーク機能をIPとして残し、これに演算部分オフローディングによる加速モジュールを接続、部分再構成技術で実装
- このアイデアをCCSにおける新システム計画 PACS-X で実現することを目指す

  ➟ Cosmo Simulator

COMA (PACS-IX)

- Cray社 CS300 ベース
- Intel Xeon Phi (KNC: Knights Corner)を全面採用
- 393ノード（2 Xeon E5-2670v2 +
  2 Xeon Phi 7110P)
- Mellanox IniniBand FDR, Fat Tree
- 2015/10時点でXeon Phi搭載クラスタとして日本最大
- File Server: DDN 1.5PB (RAID6+Lustre)
- 1.001 PFLOPS
  (HPL: 746 TFLOPS)
  June '14 TOP500 #51
- HPL効率 74.7%

COMA (PACS-IX) 計算ノード (Cray 1U 1027GR)



COMA (PACS-IX) overview

- T2K-Tsukubaの後継システムとして導入
  - H26年4月稼働開始
- システム構成
  - 計算ノード：汎用CPU＋メニーコアプロセッサ
  - ノード構成
    - CPU x 2: Intel Xeon E5-2670v2
    - MIC x 2: Intel Xeon Phi 7110P
    - Memory: CPU=64GB MIC=16GB
    - Network: IB FDR Full-bisection b/w Fat Tree
  - ノード数： 393
  - ピーク性能： CPU=157.2 TFlops　MIC=843.8 TFlops
    TOTAL: 1001 TFlops = **1.001 PFLOPS**
- システムベンダー: Cray Inc.

What is COMA ?

- Cluster of Many-core Architecture processor
- COMA＝「かみのけ座」
  - 代表的な銀河団の一つ
  - 銀河＝星の集まり（＝Many Core）
  - 銀河団＝銀河の集まり（＝Cluster）
- 同時にPACSシリーズ第9世代のマシンとなるため、”PACS-IX”のコード名を併用

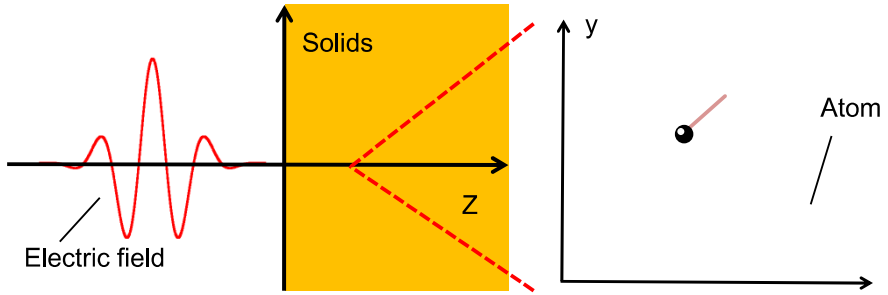
MIC (KNC) の実効性能は？

- 「そのまま（OpenMPのみ）」では性能アップは厳しい
- KNCを知った上での様々なプログラミング／チューニング
  - SIMDを意識、特にintrinsicで明示的プログラミングを行うと大幅にに性能向上する場合が多い
  - data localityへの配慮
    ➟ メモリバンド幅を有効利用するのが難しい
  - スレッド数調整
    ➟ 最大240スレッド（4スレッド×60コア）だがそれがベストではない
- アプリケーションがMPI化されていて、かつ負荷分散調整の余地がある場合
  - Symetric mode（CPUとMICを同列にx86コアの集合として利用）が適用可能
- 実アプリケーションのCOMA上における大規模実行
  - QCDアプリケーションの Native Mode 実行
  - 物性電子動力学コードの Symmetric Mode 実行

ARTED: 電子動力学コード(量子物性＋HPC)

ポスター
発表有り！

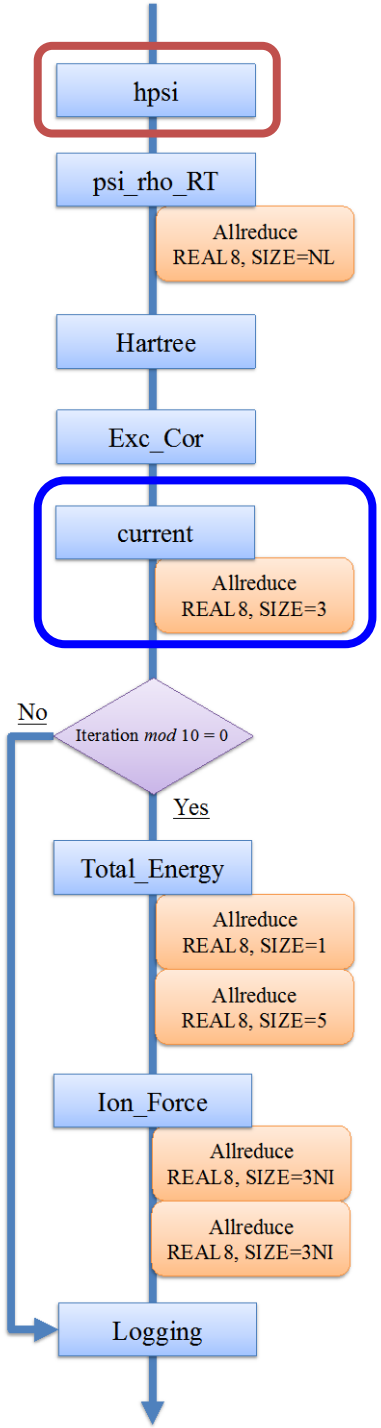
- ARTED – Ab-initio Real-Time Electron Dynamics simulator
- Multi-scale simulator based on RTRSDFT (Real-Time Real-Space Density Functional Theory) developed in CCS, U. Tsukuba to be used for Electron Dynamics Simulation
  - RSDFT : basic status of electron (no movement of electron)
  - RTRSDFT : electron 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

Computation domain and amount

- Parameters for wave function expression
  - k-points (NK), band-number (NB), 3-D lattice points (NL)
  - valuables are in double precision complex with matrix of (NK, NB, NL)
  - for stencil computation, size NL of calculation is performed NKxNB times
- Parameters used in this research (two models)
  - SiO_2 : (4^3, 48, 36000 = (20, 36, 50)) -> not enough large
  - Si : (24^3, 32, 4096 = (16, 16, 16)) -> larger parallelism on thread
- NK is parallelized by MPI, then NKxNB is parallelized in OpenMP
  - domain of each process: (NK/NP, NB, NL)
    (NP = number of processes)
  - space domain is not decomposed to minimize MPI communication



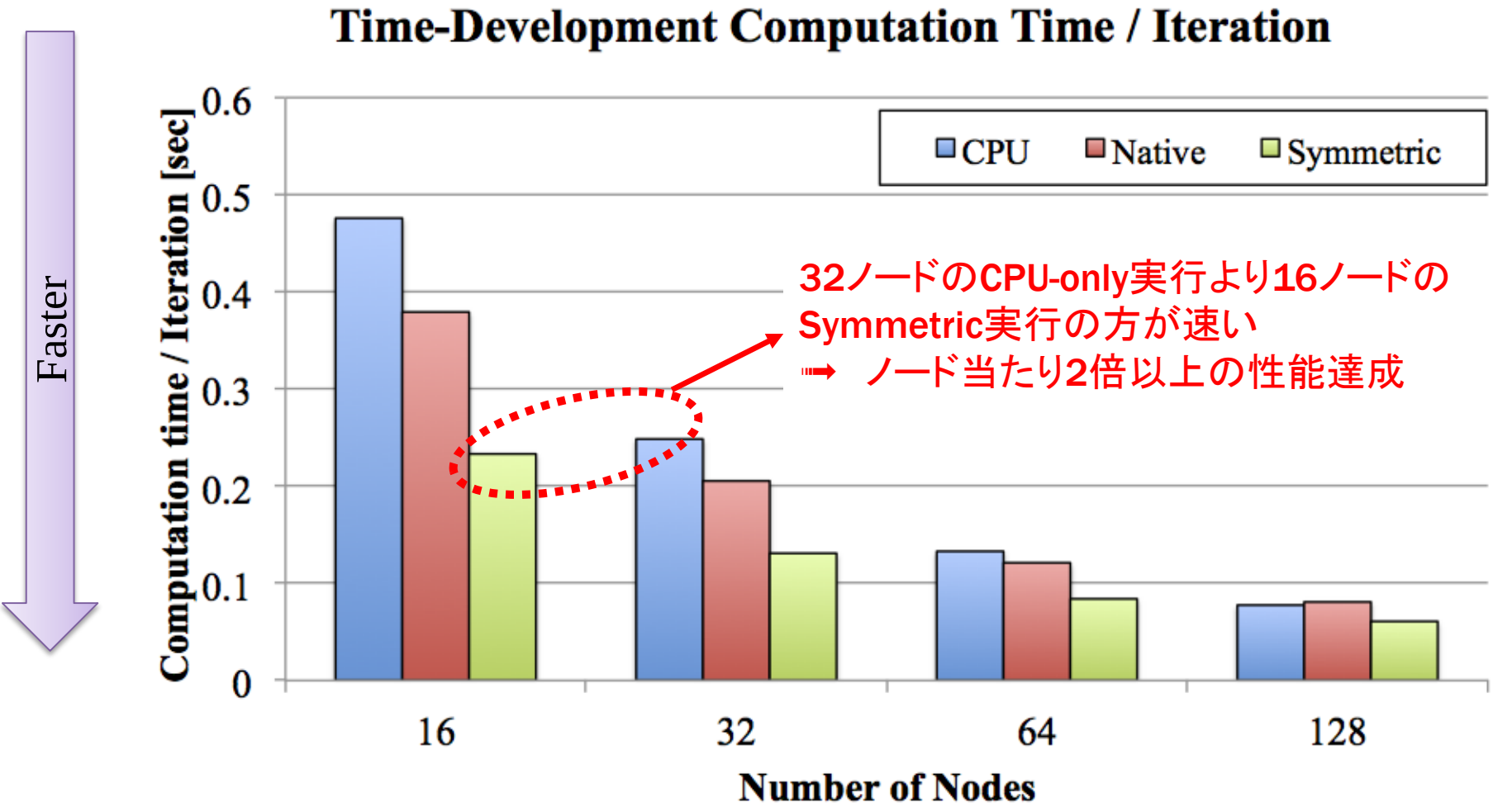
Time development part of ARTED

- in *hpsi*, size NL of 25-point stencil (3-D with depth=4), 4 times / iteration
  - parallelization: (NK / NP) x NB
  - 20Byte/Flop on each dimension
    → highly memory intensive
  - cyclic boundary condition
- *current* computes stencil calc. in each iteration
- *Allreduce* is invoked 6 times at maximum
  - double precision value of NL size for Allreduce
  - communication time is negligible
    → no meaning to use Offload model

How to improve the performance on MIC

- Compilation options
  - Common on CPU and MIC
    -ipo -fp-model fast=2 -complex-limited-range
    -no-vec-guard-write -qopt-ra-region-strategy=block
  - on MIC
    -qopt-threads-per-core=4 -qopt-gather-scatter-unroll=4
    -qopt-assume-safe-padding -opt-streaming-cache-evict=0
    -qopt-streaming-stores always
- Array expression modified for good vectorization
- Ref: Data reference pattern changed
- SoA: AoS -> SoA
- Ex: Loop unrolling on a specific dimension

ARTEDコード全体のスケーラビリティ



CCSにおける次の汎用スーパーコンピュータ計画

- 最先端共同HPC基盤施設
  （JCAHPC: Joint Center for Advanced HPC）
  - 筑波大学と東京大学がT2K終了後のスパコン共同調達・共同運用を行う施設
  - 東京大学柏キャンパス内に設置
  - 2016年10月（目標）に国内最大規模のスパコンを調達、運用
  - many core architecture に基づくプロセッサを想定
- COMAはこの先鞭として many-core base application の開発ベースとしての役割を果たす

JCAHPCシステム概要(暫定)

- 計算ノード
  - many-core architecture processor を演算加速器としてではなく main CPU として利用
  - single socket / node
  - memory: fast=16GB　slow=96GB　以上
- ネットワーク
  - Full Bisection B/W の fat-tree 構成
  - 100Gbps テクノロジ(InfiniBand EDR 相当)
- 全体システム
  - 総ピーク性能25PFLOPS以上
  - 20PB以上の共有ファイル・システム(フラット構造)
  - 単一スケジューラの下で様々な運用

JCAHPCシステムの調達・運用

- 調達スケジュール（予定）
  - 現在、意見招請中
  - 2015/12頃：最終仕様公開予定
  - 2016/2頃：入札
  - 2016/3頃：開札・契約
  - 2016/10：運用開始予定（試験運用）
    HPCI等は2017年度から
- 運用方針
  - 筑波大・東大の両大学の個別運用ポリシーを継続
    - 両大学の有償・無償利用プログラム
  - リソース配分は調達予算に比例
    - ノードや領域で分割するのではなく、ノード時間積の総和で調整
  - 大規模共用
    - HPCI
    - 特別な超大規模運用機会（GBAチャレンジ等）

まとめ

- 筑波大CCSではスペース性能・電力性能の観点から、accelerated computing を今後の重要な研究プラットフォームと捉え、HA-PACS, COMAの導入を行い、大規模並列演算加速計算を推進中
- HA-PACSのGPU accelerated code はかなり成熟している
  ➟ routine 運用に近い
- COMAではmany-core applicationの開発を推進中だが、かなり頑張ってチューニングしないと性能が出ない
  ➟ KNCでチューニングを進める
  ➟ 今後の many-core の性能・機能向上に期待
- CCSのシステム方針
  - 独自システム(筑波大内)としては演算加速器系を継続、FPGAの導入を含めた技術革新とユーザ努力も含めた超高性能化を目指す
  - JCAHPCシステム(柏設置)としては従来のOpenMP+MPIアプリの継続とユーザに使い易い汎用システムを目指す