PACS-CSについて

朴泰祐
筑波大学 計算科学研究センター/システム情報工学研究科
taisuke@cs.tsukuba.ac.jp
http://www.ccs.tsukuba.ac.jp/PACS-CS/

PACS-CS導入の背景

現在のCCSのスーパーコンピュータ

PACS-CS
2560 nodes
2560 cores
14.4TFLOPS
(2006～)
Hitachi & Fujitsu

FIRST
256 nodes
512 cores
+ BladeGRAPE
3.5TFLOPS
+ 35TFLOPS
(2005～)
HP & Hamamatsu Electronics

T2K-Tsukuba
648 nodes
10368 cores
95 TFLOPS
(2008～)
Appro Int. &
Cray Japan Inc.

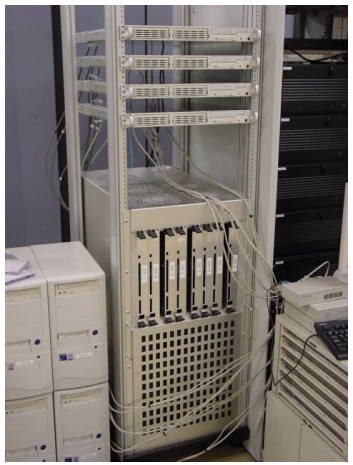
PACS-CS導入前のCCSの計算機資源

＜計算科学全般汎用＞

CP-PACS (1996-2005)
614 GFLOPS
1996.11 #1/TOP500

PACS-CS (2006-)
14.4 TFLOPS
2006.6 #34/TOP500

＜計算宇宙物理専用＞



HMCS (2003-)
汎用：430 GFLOPS
専用：8 TFLOPS

FIRST (2005-)
汎用：3.5 TFLOPS
専用：35 TFLOPS

筑波大学におけるPACS シリーズの歴史

- PACS-9 (1978, 7Kflops), PACS-32 (1980, 500Kflops)
  PACS = Parallel Array Continuum Simulation
- PAX-128 (1983, 4Mflops)
  PAX = Processor Array eXperiment
- PAX-32J (1984, 3Mflops) ⇒ 三井造船により商用化
- QCDPAX (1989, 14Gflops) ⇒ アンリツにより商用化
  Quantum Chromo-Dynamics by PAX
- CP-PACS (1996, 614Gflops) ⇒ 日立製作所より商用化（SR2201）
  Computational Physics
  by Parallel Array Computer System
- PACS-CS
  Parallel Array Computer System
  for Computational Sciences

PACS-CS導入の背景

- CCS主力リソースとしてCP-PACSの後継機を模索
- PFLOPSオーダーを目指す長期的計画とは別に短期的（開発期間1～2年，利用期間5年程度）システムが必要
- しばらくはコモディティ技術が有利（CPU, network）
- センターのニーズと利用形態に即した独自のシステム

クラスタによる10～20TFLOPSクラスのシステム

PACS-CS

(Parallel Array Computer System for Computational Sciences)

PACS-CSの設計・開発コンセプト

CCSで必要なHPCリソースの考え方

- 計算科学研究センターの特徴
  - 対象とする分野の重要課題の認識
  - ある程度アプリケーション（あるいは手法）を絞り込める
    - full QCD （小規模倍精度複素数行列計算＋近接通信）
    - ナノ物性 （CG法）
    - 地球環境（大量のrun-timeファイル出力）

- MPPからの移行
  - 格差の大きい [**演算:メモリ:通信**] 性能比に対する最適な構成・技術を検討する
    ⇒実効性能向上のための性能バランスを重視
  - 主要アプリケーションではいずれもメモリ＆通信のバンド幅が要求される

一般的なHPC向けクラスタ

PACS-CS

- プロセッサはIntel互換（Xeon, Opteron, Itanium2）
- Dual CPU 構成が一般的
  - 全体ピーク性能を保ちつつネットワークインタフェースの数と設定スペースを減らす
  - メモリバンド幅不足（memory wall problemの助長）(Linpackは速い)
  - Network boundなアプリケーションは不得意
- ネットワークは SAN (System Area Network)
  - Myrinet: MyrinetXP ⇒ Myrinet10G
  - Infiniband: SDR ⇒ DDR (2.5GB/s?)
  - ネットワーク性能が必要ない分野ではGbEthernetが中心

PCクラスタに対する我々のスタンス

- 従来のベクトル機・MPPの代替として、高いコストパフォーマンスを提供
- コモディティとして確実に高性能化が進むプロセッサに対し、ハイエンドクラスタではネットワークコストが増大（特に数千台規模になると）
  - Fat-Tree 又は Clos網を基本にした構成
  - NIC及びスイッチの価格（変動が大きく、熟成前は特に高価）
- 「見えない」「不特定な」ユーザ＆アプリケーションに対する対応は難しい
  - 利用形態? CPU intensive or I/O (Network) intensive ?
    How many CPUs / job ?
- 「見える」「分野が限られた」ユーザ＆アプリケーションに対してはもっと効率的な解があるのでは？

PACS-CSのコンセプト

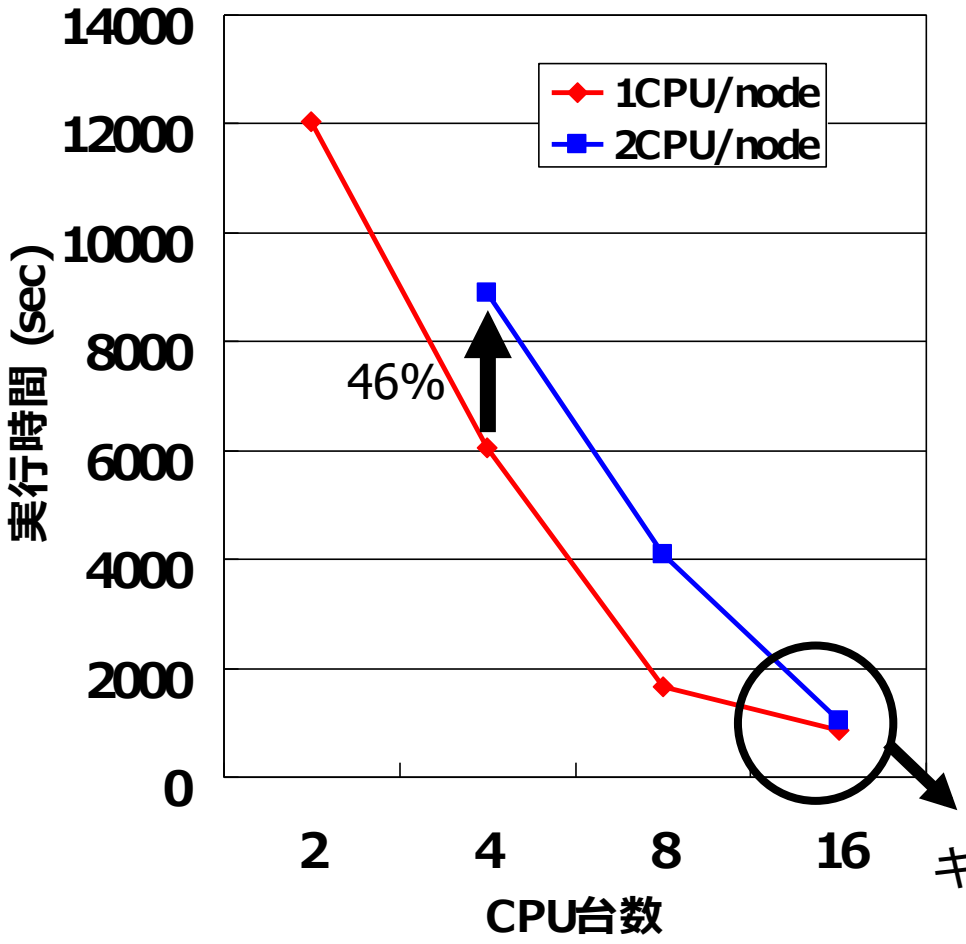
- MPPからコモディティへの移行
  - CP-PACS的な開発（メーカーとの産学協同研究）は難しい
  - 単純に買うのか？⇒今後につながる開発をしたい
- 一種のクラスタ、ただし**CPU性能：メモリ性能：ネットワーク性能**の性能バランスを極力保つ
- 対象問題・CCSの運用の特性を生かす

- コモディティ技術によるMPP（ライクなマシン）を作る
  - チップ等の開発はしない
  - コモディティCPU、コモディティネットワーク
  - ボード・レベルでの開発は行う
  - ソフトウェアもコモディティ（必要最低限のものは開発）

PACS-CSの基本アーキテクチャ

- 分散メモリ方式による超並列クラスタ
- ノード構成
  - コモディティプロセッサ（Intel Xeon）と汎用SDRAM
  - single CPU / node ⇒ 有効メモリバンド幅の追求
  - multiport NIC による Gigabit Ethernet の多用
- ネットワーク構成
  - コモディティNICとスイッチを最大限に活用
  - 多数の Gigabit Ethernet を trunk，さらに多次元展開して高バンド幅を実現
  - 3次元ハイパクロスバ網（3D-HXB）⇒「実空間モデル」への対応
- これらを実現するプラットフォームとしてマザーボードを新規開発

なぜ single CPU / node か？

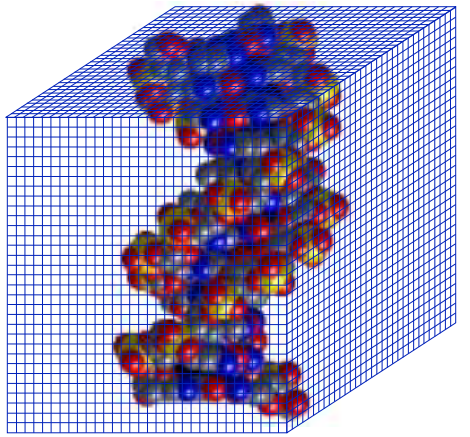


- 物性計算：実空間密度汎関数法によるSi512の計算
- 測定環境
  - CPU: Xeon 2.8GHz dual CPU SMP
  - Memory: PC2100 (4.2GB/s)
  - Network: Myrinet2000
  - 環境： SCore 5.1 PM/Myrinet
- 同じ問題をノード当りのCPU数を変えて実行
- メモリバンド幅を1CPUで占有するか2CPUで共有するかの比較

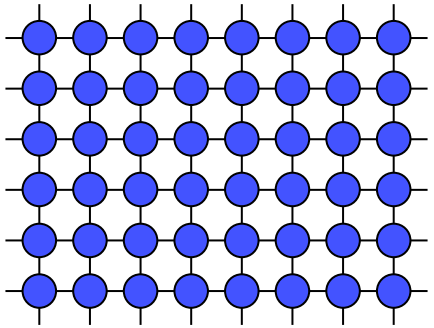
キャッシュに当たり始めたので差がない

計算科学における実空間アプローチ

- 実空間モデルによるアプローチ
  - 実空間離散化コンセプトに基づく超並列向けコード
    ⇒隣接通信＋ブロードキャスト／リダクションを基本とする大規模並列処理
  - 「総演算量削減」ではなく「総演算時間削減」を
    - **従来手法：FFT、スペクトル法、ルジャンドル展開**
      ⇒空間分解能を上げず畳み込みによる間接計算
    - **実空間手法：**
      ⇒実空間上の実モデルを直接シミュレート
  - 将来の拡張に向けてアプリケーションのアルゴリズム・コードの本質的見直しが必要
    ⇒ **We are ready for it !!**



実空間離散化からプロセッサ空間へ直接マップ



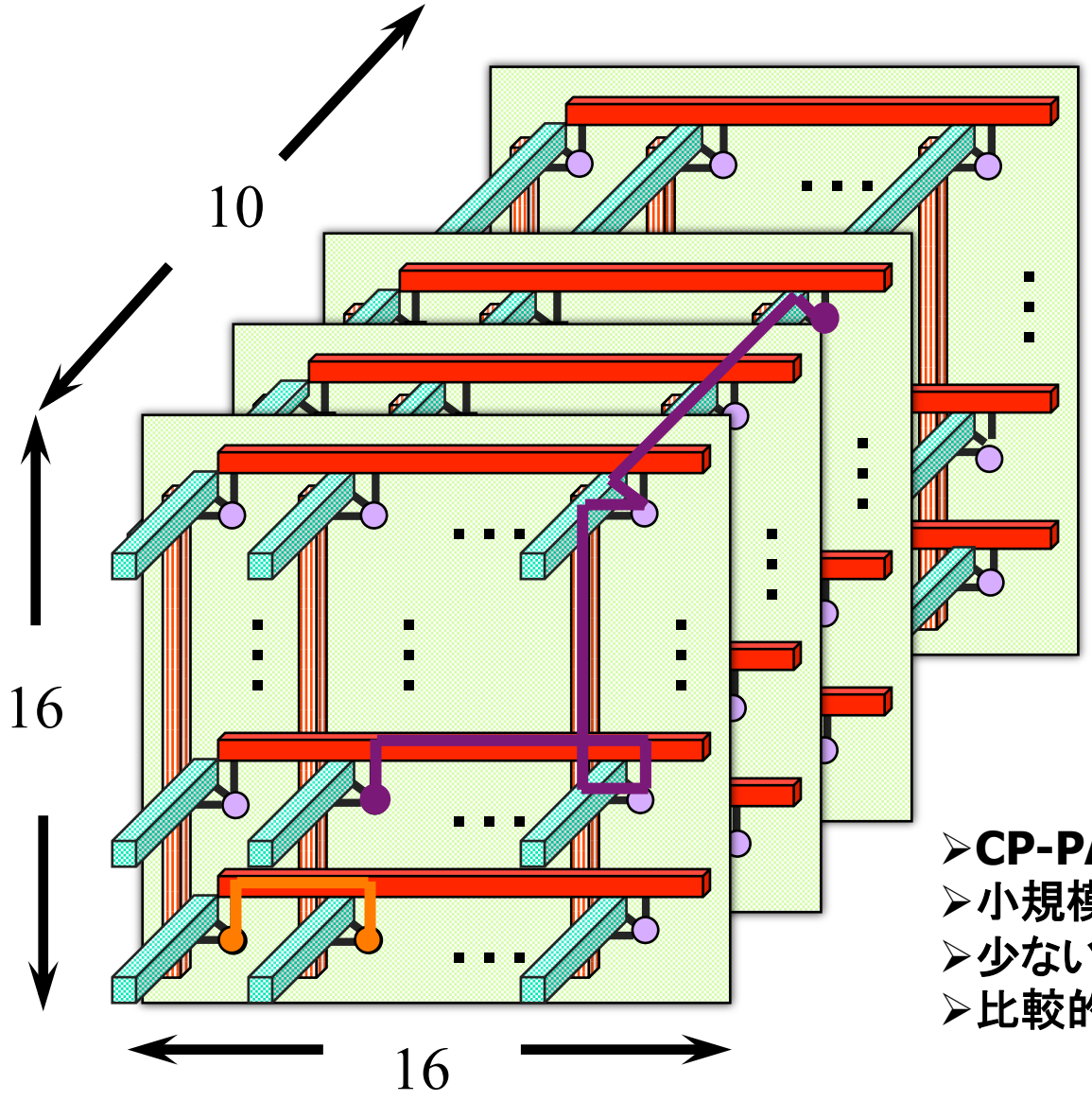
- 実空間アプローチが可能な応用例
  - 生命計算、物質計算
  - 流体計算（計算工学）
  - 連成計算（**multi-physics, multi-scale**）

PACS-CSシステムの概要

PACS-CSシステム諸元

ノード台数	2560 (16 x 16 x 10)
理論ピーク性能	14.3 Tflops
ノード構成	単一CPU / ノード
CPU	Intel LV Xeon EM64T, 2.8GHz, 1MB L2 cache
メモリ容量	2GB/ノード　(5.12 TB/システム)
並列処理ネットワーク	3次元ハイパクロスバ網
リンクバンド幅	単方向 250MB/s/次元 単方向 750MB/s （3次元同時転送時）
ローカルHDD	160 GB/ノード（RAID-1）
総システムサイズ	59ラック
総消費電力（ピーク）	545 kW
ソフトウェア	Linux Fedora Core 3, SCore 5, Intel compiler, YAMPII, MPICH

3次元ハイパクロスバ(3D-HXB)網 (16x16x10=2560 node)



X次元スイッチ

Y次元スイッチ

Z次元スイッチ

計算ノード(CPU単位)

1つのスイッチで直接結合されているCPU間の通信

複数の次元のスイッチを経由して行われるCPU間の通信(中継CPUが必要)

- **CP-PACSで使用された動的網**
- **小規模クロスバスイッチ**
- **少ない次元数(degree)**
- **比較的高いランダム転送能力**

GbEトランキングに基づくハイパクロスバ網

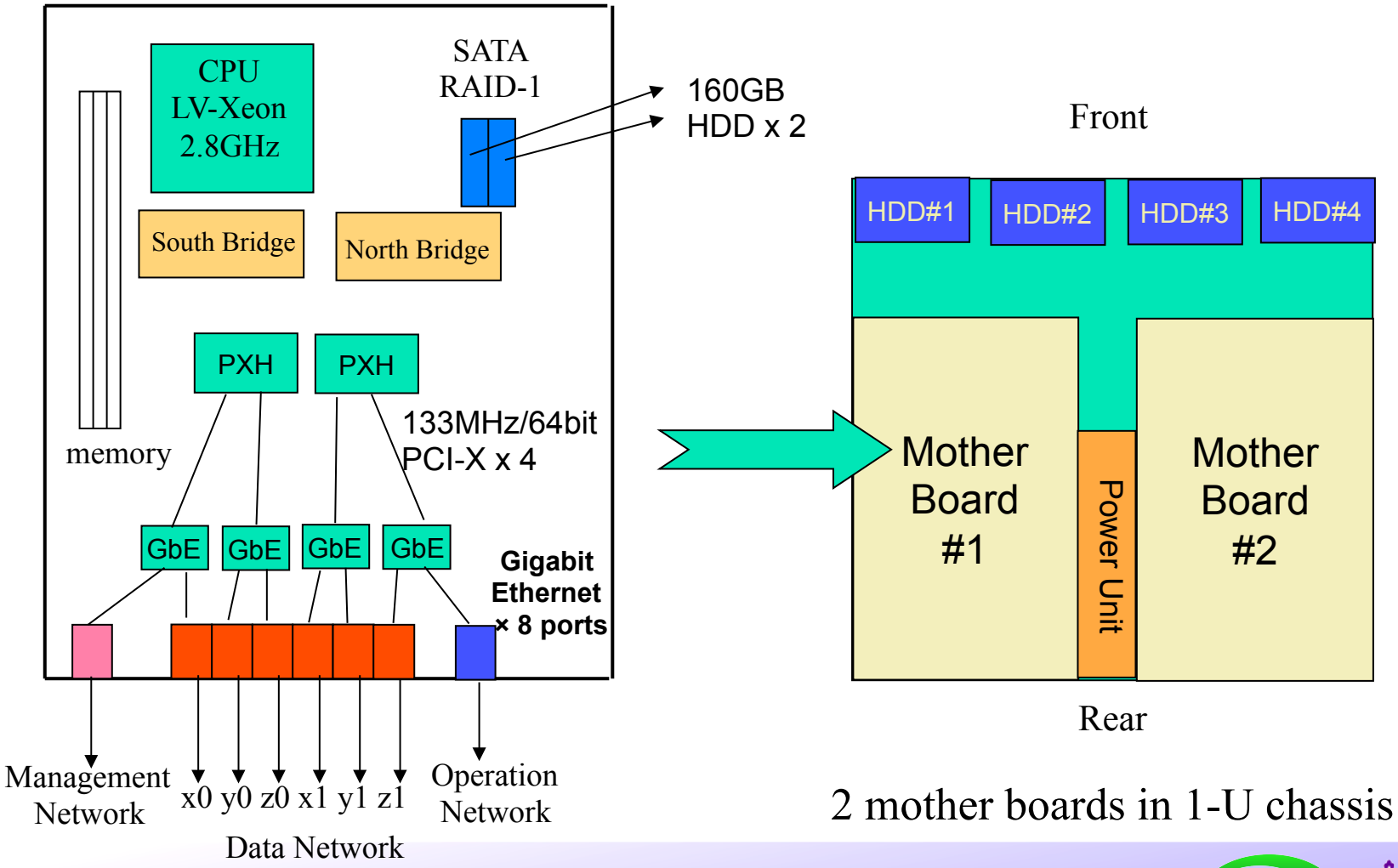
- 対価格性能比の高いコモディティハードウェアの有効利用
  - NIC chip
  - スイッチ（L2, 10～20ポート）
- ネットワークバンド幅
  - GbE単体では十分ではないが少数本のトランキングで稼ぐ
  - 本数が大量でなければソフトウェアトランキングで十分対応可能
  - 多次元方向の同時通信の実現
    ⇒ピークバンド幅＝（リンクバンド幅）×トランク数×次元数
  - single I/O point のネットワークで発生するトラフィックを multi-I/O point で拡散
    ⇒バスを圧迫しない
- 多次元化のためのルーティングが必要
  - コモディティ部品（通常のGbE NIC chip）のためソフトウェアルーティングが必要
  - 1次元当たりのノード数が16程度であれば3次元で十分

⇒効率を上げればソフトウェア処理でも十分対応可能

マザーボードの実装

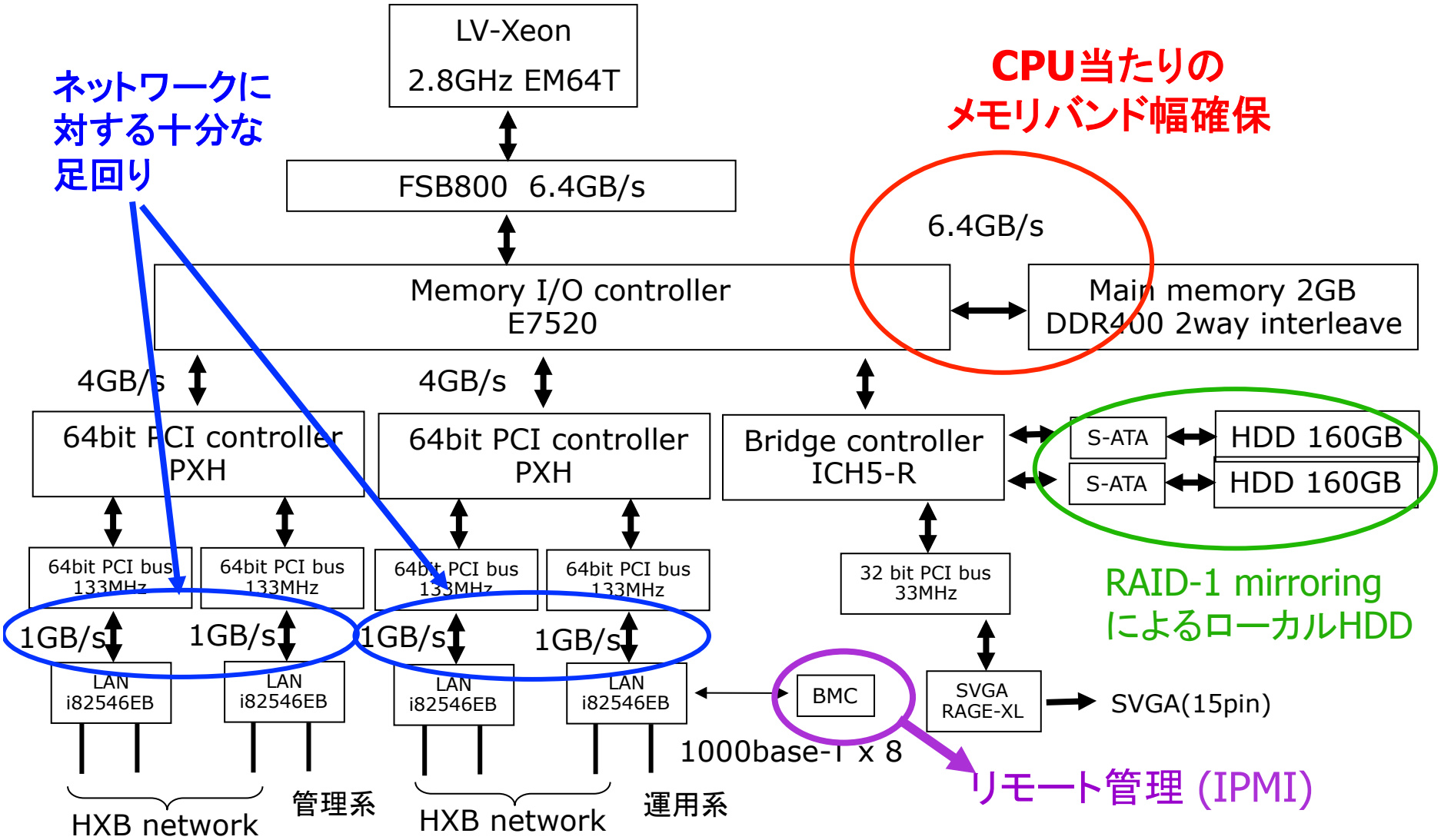
- 新規開発マザーボード
  - 基本的には通常の Xeon single socket ボード
  - dual socket for PC3200 SDRAM (6.4 GB/s total)
  - 3D-HXBのための multiport GbE NIC を支えるfull spec PCI-X (64bit/133MHz) x 4
  - 8 port の GbE NIC
    - 3D-HXB のために6本 (2 trunk x 3次元)
    - 運用ネットワーク(「生活線」)のために1本
    - 制御ネットワーク(serial over LAN によるリモートコンソール)に1本
  - 2つのserial ATA HDDによるRAID-1 mirroring local drive
  - 日立製作所と共同開発

ボードの構成(single CPU/node)

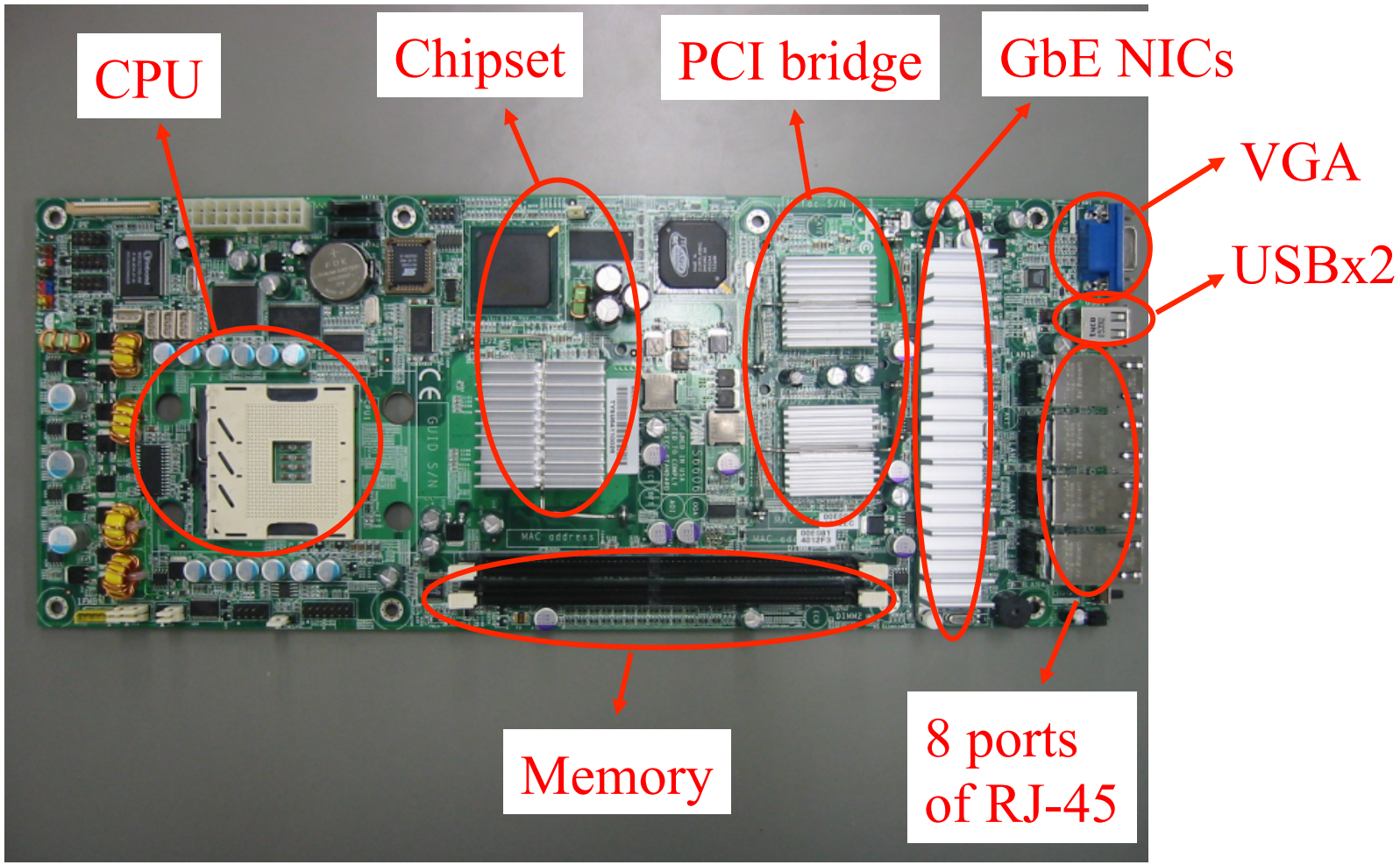


2 mother boards in 1-U chassis

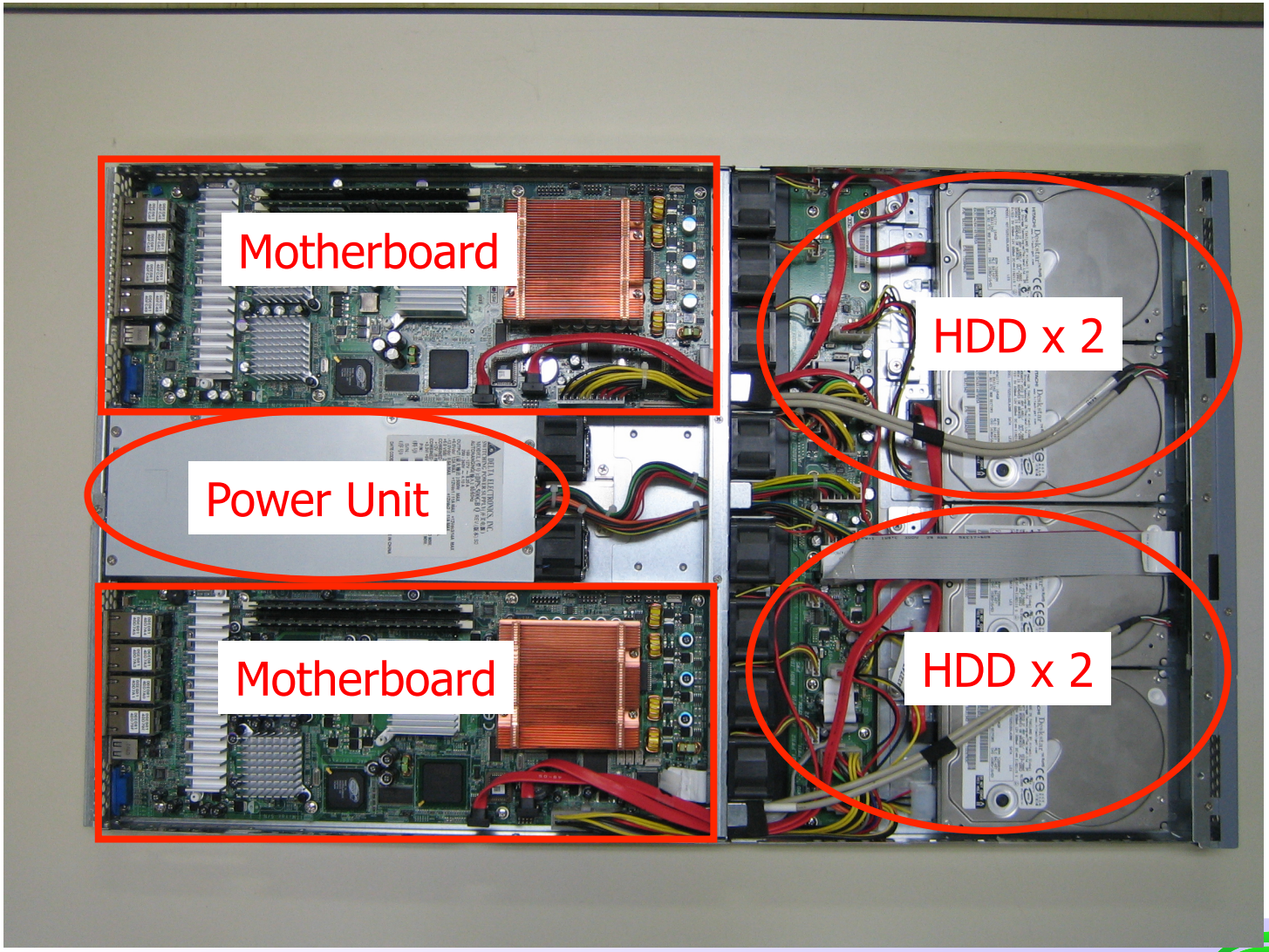
ノード内ブロックダイアグラム



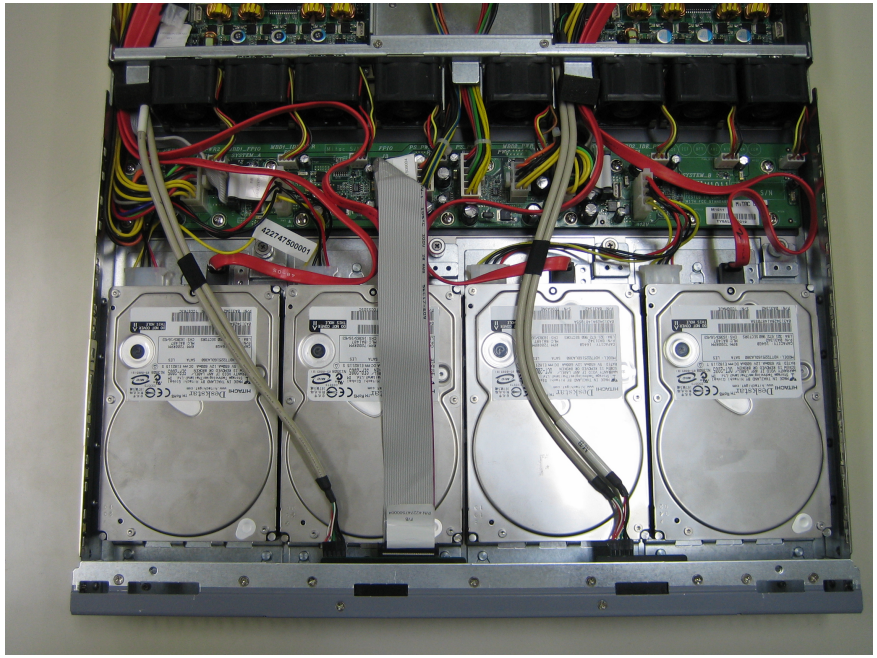
マザーボード写真

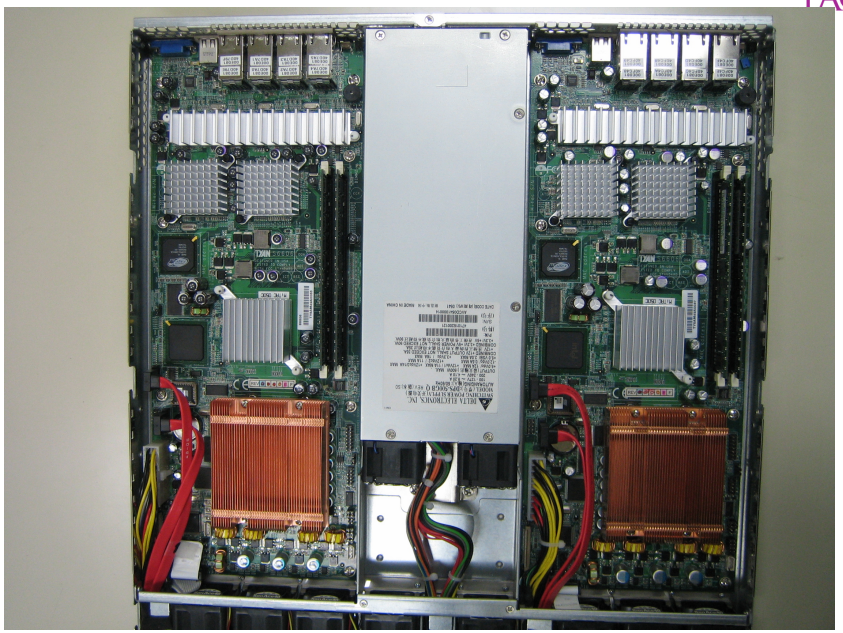

CPU
Chipset
PCI bridge
GbE NICs
VGA
USBx2
Memory
8 ports
of RJ-45

ユニット(19inch x 1U)



電源コントローラとGbEポート群





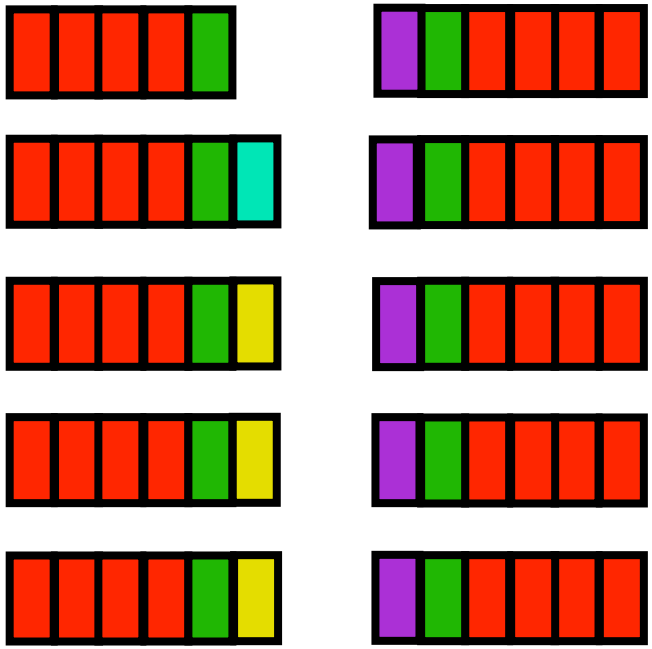
ソフトウェア

- Linux + SCore
  - PM/Ethernet-HXBドライバ
  - パーティショニング、モニタリング
- バッチ／キュー（PBS）、ただしそれほど複雑ではない
  512 node程度を単位とするキューを用意し、固有名指定または全体名指定（node数のみ指定）でジョブ投入
  ⇒local HDDを「つなぎファイル」に使用する場合を想定
- MPIによる並列プログラミング
  - MPICH and YAMPII
  - 特にcollective通信に関し、3D-HXBを有効活用する効率的なライブラリを開発する必要あり
- 言語：Fortran90, C, C++
- 数値計算ライブラリ

3D-HXBネットワークドライバ

- 合計6 portのGbE NICをソフトウェア制御
  - 双方向通信で合計1.5GB/sものバンド幅
  - 低レイテンシ通信（ただし比較的大きなメッセージ）
  - TCP/IPでは高性能は見込めない
- 3D-HXBの要請
  - n 本(n=2)の GbE link をトランクしてバンド幅増強
  - 最大3次元のルーティング機能
- PM/Ethernet技術の発展
  - SCoreクラスタミドルウェアで開発された GbE trunk技術
  - 3D-HXBのために新たに PM/Ethernet-HXB を開発
  - 富士通と共同開発
- SCore contributionとして公開
- その後、性能をより向上させる低レベルPM/Vxを開発

ラック配置とフロアプラン



Node rack

X & Y dim. switch rack

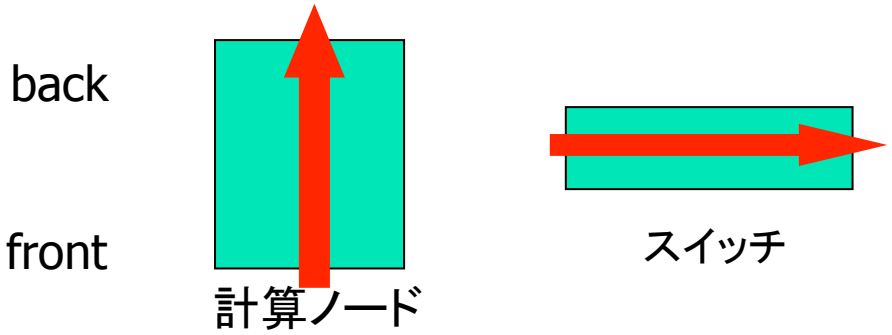
Z dim. switch rack

Operation switch rack

File server & RAID

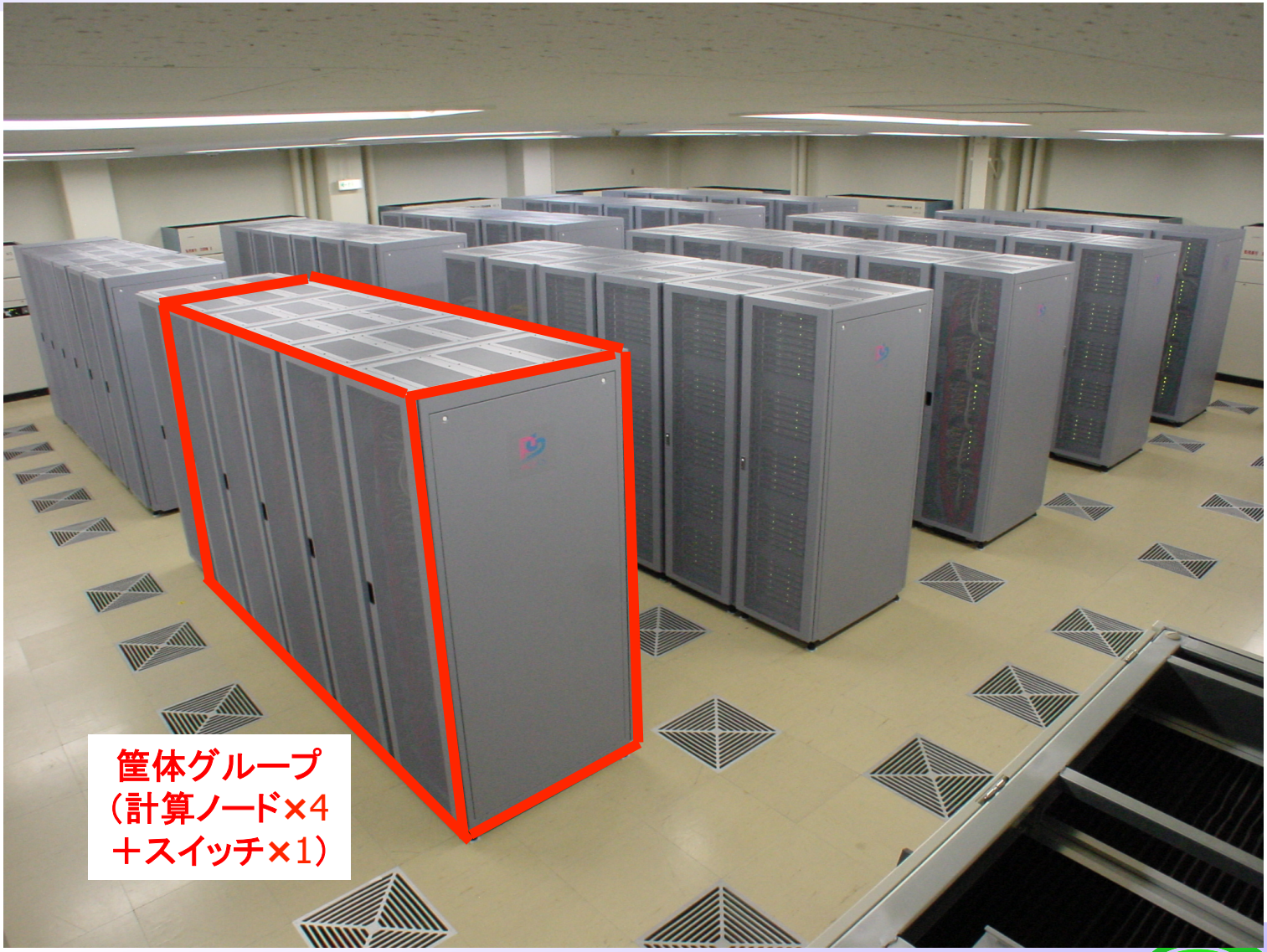
ラック構成

- ノード及びスイッチは全て 1U rack mountable
- 計算ノードラックとスイッチラックを個別に設置
  - X次元だけでもラックに閉じ込めるように混在ラックを検討
    ⇒エアフローの問題により困難

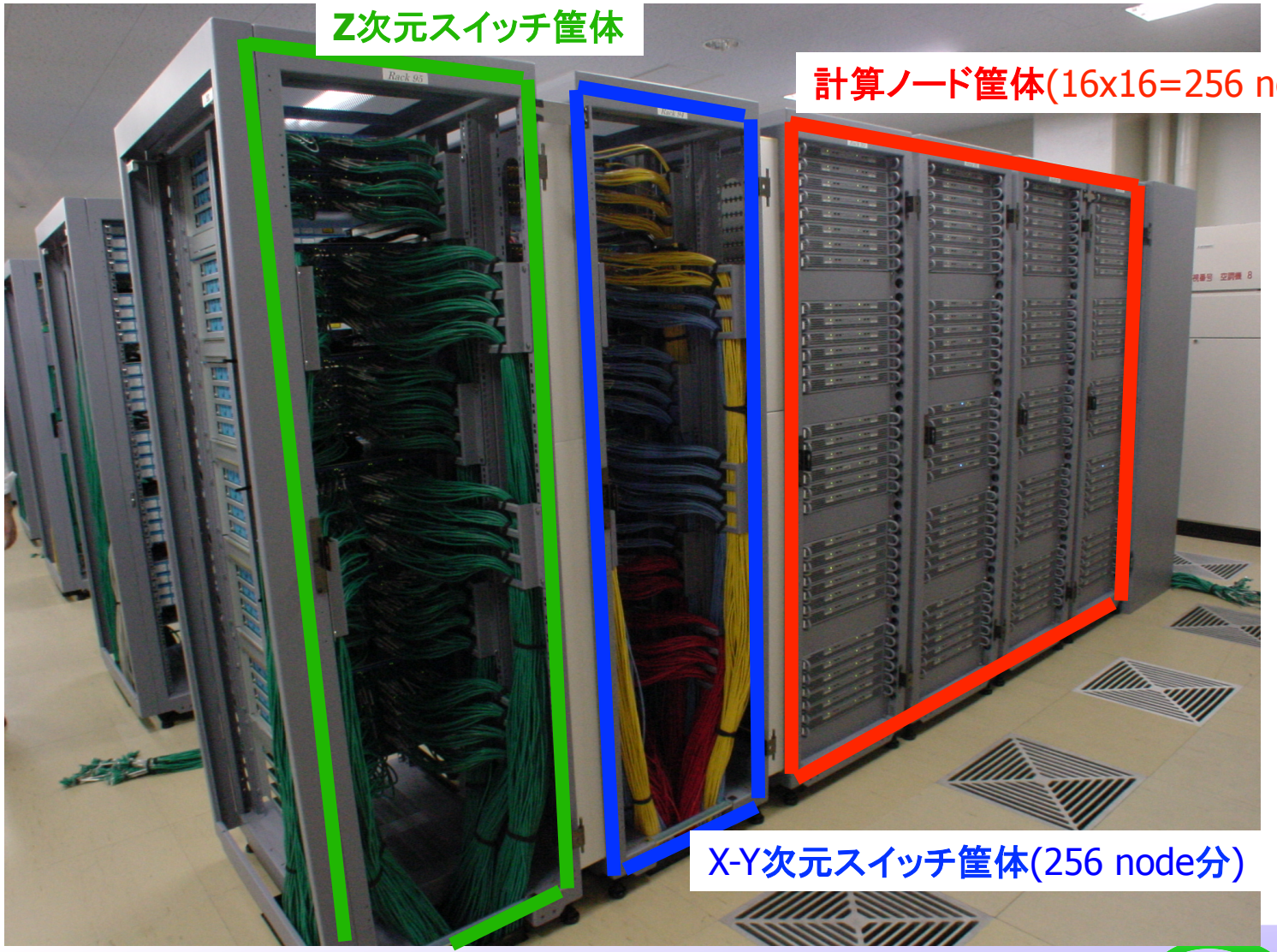


- ノードラック4台（256 node=16x16分）とこれに対応するX, Y次元スイッチラック1台を単位とし，これを10セット配置
- Z次元スイッチラック5台を中央部に分散配置
- SCoreサーバノード（3台），ファイルサーバ（10TB），コンソールノード等を1ラックに集約

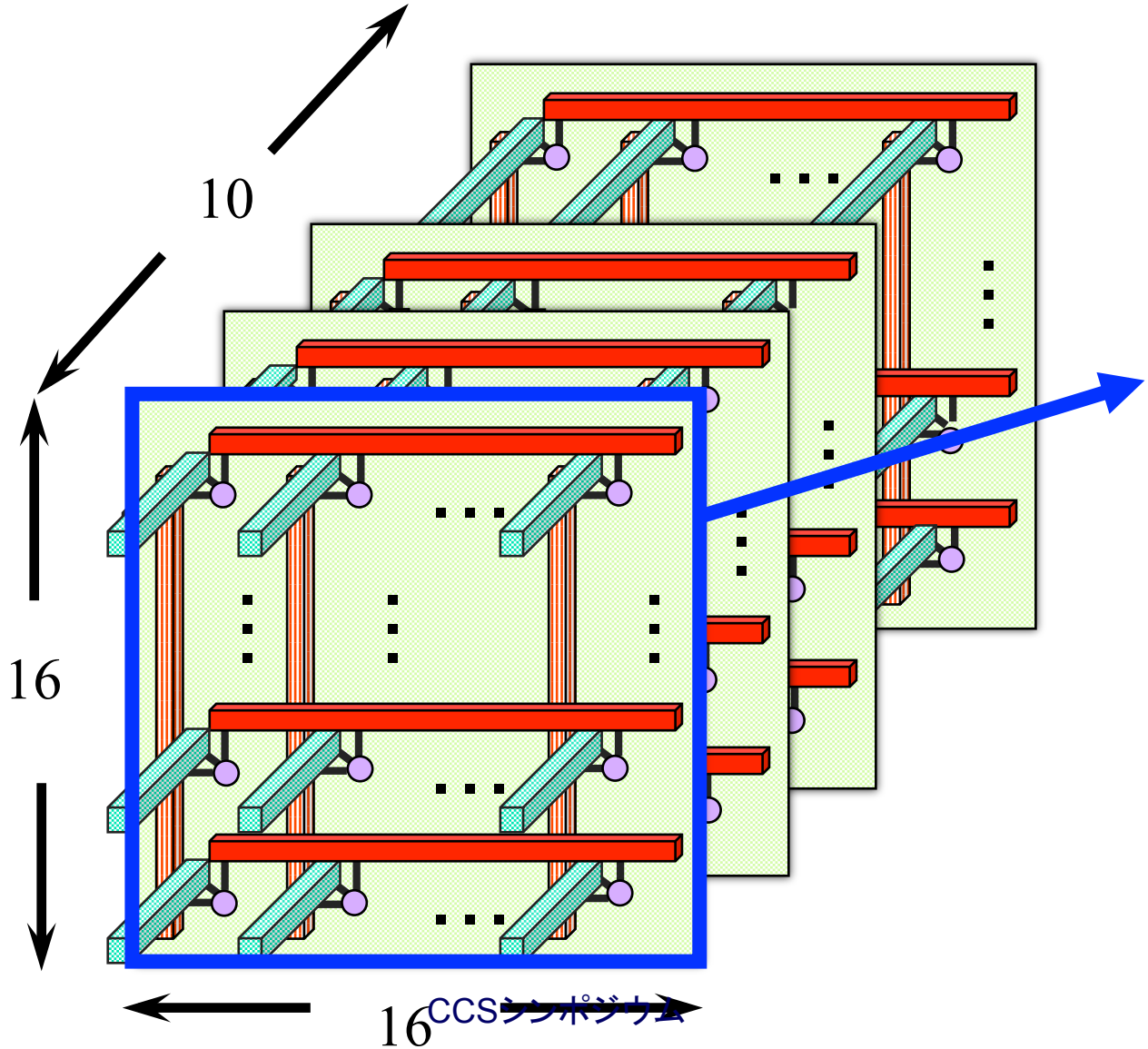
PACS-CS (2560 node)



1筐体グループの構成



物理構成とネットワークの対応



16x16のXY平面=256
これが1つの筐体グループ

X-XB と Y-XB を構成する
16-port switch x 16 x 2
を1つのスイッチラックに

ネットワーク及びスイッチの実装

- 16 port L2 GbE switch が大量(3D-HXB用で256台)に必要
  ⇒実装効率を上げるため48 port L2 switchをVLAN により3分割して利用
- trunkされる2本のリンクは別スイッチに配置
  - 特定次元の集中通信時に負荷を分散
  - 耐故障性(現在計画中)
- 3D-HXBは直交網であるため，各ノードからのケーブルは全て異なるスイッチに向かう
  - 実装はかなり大変
    ⇔ ツリー網では上流に向かってどんどん束ねればよい
  - 実際にノードラックの下でケーブルを直角に曲げ，同じスイッチに向かうグループを改めて作って束ねる
  - 経験：物理空間で直交空間でバンド幅を得ようとするとそれなりに大変

ラックのケーブリングの様子

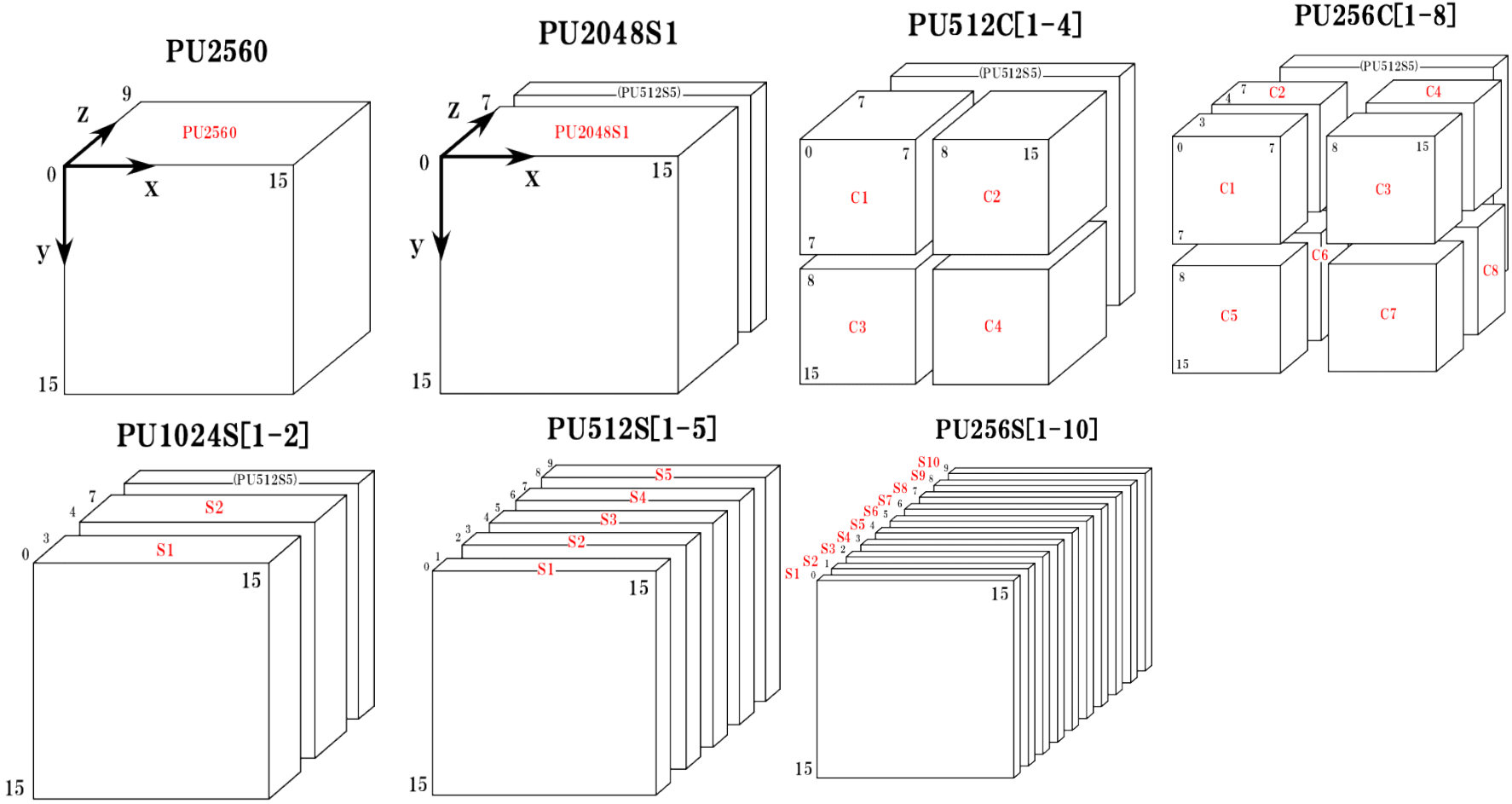
X,Yスイッチラック及びZスイッチラック

ノードラック

PACS-CSロゴ

PACS-CSの運用

パーティション構成



これらを適宜混在させ，ユーザニーズに応じてスケジュール

ファイルサーバとローカルHDD

- ジョブ実行は3フェーズのステージングで構成
  - Input phase
    - ファイルサーバからローカルHDDに転送
    - 64ノードを1グループとし，代表ノードはファイルサーバをNFSマウント
    - 複数のrcpを並列実行して各ノードのローカルHDDにデータコピー
  - Execution phase
    - scount + scrunによってSCore管理下でジョブを実行
    - 計算結果は各ノードのローカルHDDに保存
  - Output phase
    - ローカルHDDからファイルサーバに転送（方法はInput phaseの逆）

3フェーズ制御用スクリプト

```
* : type = mkdir -p %s
/work/user1/app1
* : type = put
/hfs-work/user1/app1/data${rankid}   /work/user1/app1/data
0 : type = put
/hfs-work/user1/app1/paramfile   /work/user1/app1/paramfile

job_step_1: type = job
  class = q512sS3
  job_name = sample.jcl

* : type = get
/work/user1/app1/result   /work/user1/app1/result${rankid}
```

全ノード実行

Rank番号参照

ノード0のみ実行

ファイルサーバ⇒ローカルHDD

パーティション＋クラス
ジョブ実行詳細ファイル

ローカルHDD⇒ファイルサーバ

利用開始当時の状況

- ピーク性能14.3TFLOPSで2006年7月より稼動を開始
- 極めて安定した稼動状況
  - 納入前のM/Bレベルのチェック（MBメーカー）と工場内でのチェック（日立）によるダブルチェック
  - 納入前のディスク全数調査
  - 毎月の定期メンテナンスによる徹底的な予防保守
- Linpackベンチマークで10.35TFLOPSを達成（TOP500中34位，国産では2位）
- SCoreクラスタとして世界最大（現在まで）

プロジェクトによる利用

- CCS内の初期メインターゲットジョブ
  - QCD (Quantum Chromo-Dynamics) with DDHMC (Domain Decomposed Hybrid Monte-Carlo)
    素粒子物理学におけるクォークの基本性質の計算
  - RS-DFT (Real Space - Density Function Theory)
    物性第一原理計算による半導体分子のシミュレーション
- 運用開始3年目から「学際共同利用」に100%供出
  - 計算科学・計算工学の諸問題に対する公募・審査によるプロジェクト選定⇒リソースの無償提供
  - 応用とシステムの協業を強く推奨
  - 毎年約30件程度のプロジェクト採択

PACS-CS安定運用の理由

- 初期不良の徹底的な排除
  - ボードレベルでの予備運転＋エージング
  - HDD等の機械部品について厳しいチェック（PACS-CSの場合，合計5120台のローカルHDDを全数検査して組み込んだ）
  - 品質管理された完成度の高いマザーボード実装
- ダイナミズムをできるだけ減らす
  - パーティション運用
  - ノードのスケジューリング
- ファイルサーバへの接続形態を考慮
  - 大規模システムに対してスケーラブルなファイルサーバを用意するのは基本的に無理
  - スケーラブルな接続形態・運用形態が必要

運用開始後の改良

- 通信性能
  - 当初、隣接転送を中心としたQCD計算においてEthernet trunkingの性能が思ったより上がらなかった
  - 一斉隣接転送は非同期に起こり、じわじわと遅れが伝搬していく
    ⇒当時、OS jitter の問題はそれほど強く認識されていなかった
    ⇒ある程度のアプリケーションレベルでの同期を行い、問題を回避
  - さらにPM-Ethernet/Vxを開発し、基本通信性能を向上
- 省電力対応
  - 256ノードを基本パーティションとするため、スケジューリングが容易
  - 一定時間のアイドルを検出し、パーティション単位でシャットダウン
    ⇒ジョブ投入による再起動
  - アイドル時間をほぼそのまま節電に

PACS-CSの技術と発展

- “Bandwidth-aware PC Cluster” というコンセプト
  - 従来のPCクラスタの抱える「CPU性能とメモリ性能・ネットワーク性能のギャップ」に対する一つの回答
  - その後のT2K-Tsukubaに代表されるfat-node型システムに比べ、実効性能の高さが示された
    - 例：QCD計算：ピーク性能比で約2倍の性能
  - Ethernet trunking技術の育成⇒その後の同種技術へ
- 様々なアプリケーション開発
  - 定常的に数百ノードが利用可能なリソースは当時としては貴重
  - 途中からT2K-Tsukubaと平行して大規模アプリケーション開発、学際共同利用を推進
    ⇒「京」につながる成果（RS-DFT, QCD等）

運用の終了に向けて

- PACS-CSプロジェクトはH18~H23年度、H23年6月末でPACS-CS本体の5年間の運用を終了
  ⇒3ヶ月の延長を行い、H23年9月末で運用終了
- 5年3ヶ月の運用中、常に安定した運用を継続できた
- 平均稼働運用率は85%程度

これらの知見・経験を生かし
さらなる学際計算科学に向け
PACSプロジェクトはHA-PACSへ

謝辞

- 文部科学省学術機関課
- 日立製作所
- 富士通・富士通研究所
- **PC Clusterコンソーシアム**
- アプリケーション開発・プロジェクト利用者各位

ありがとうございました！