GRAPE-DR プロジェクトの概要

牧野淳一郎、玉造潤史、丹羽純平 (東大理)、平木敬、稲葉真理 (東大情報理工)、観山正見、小久保英一郎 (国立天文台)、中村誠 (東大基盤センタ)、五十嵐喜良、平原正樹 (通総研)、村上満雄、福田健平 (NTT コミュニケーションズ)、福重俊幸、船渡陽子 (東大総合文化)、赤坂泰孝、名村　健 (日本 IBM)、戎崎俊一、古石貴裕、高橋徹 (理研)、渋谷哲郎、荒木通啓 (東大医科研)、有田正規 (東大新領域)

講演概要

- 動機
- アーキテクチャ
- ターゲットアプリケーション
- 予算・開発目標
- まとめ

科学技術用高速計算機の現状

- 「汎用性」の低下
  アーキテクチャの複雑化
  - メモリ階層
  - スカラー → ベクトル → ベクトル並列 (分散メモリ) → スカラー並列 (PC クラスタ)
    「あらゆる応用プログラムで万能に速い」計算機はなくなった？
- 半導体利用効率の低下
  - チップあたりトランジスタ数は 10 年で 100 倍
  - チップあたりの演算器数は過去 15 年間で 2 倍程度

専用計算機

アプリケーションに専用化したプロセッサ:多数の演算器を使える

商用で成功しているもの

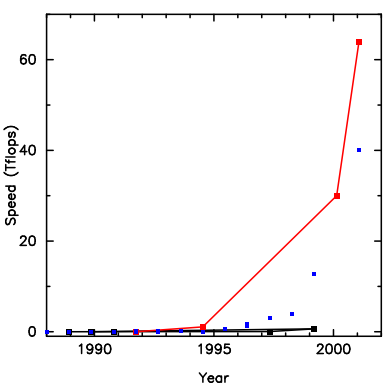
- PC 用グラフィックカード
- Sony PS2 などのゲーム機

演算器は 10-100 個

科学技術計算用専用計算機

- 多体問題専用計算機
  - 粒子間相互作用計算に特化した演算パイプライン
  - 400 個程度の演算器を 1 チップに集積 (GRAPE-6, MDM)

GRAPE の進化



赤: GRAPE

青: 汎用ベクトル/並列

速度はほぼ同等。
開発費は 2 桁違う。

専用計算機の「限界」

開発費の高騰

1990	1μm	1500 万円
1997	0.25μm	1 億円
2004	90nm	3 億円以上？

ある程度広い応用を持つものでないと難しい

FPGA、「リコンフィギャラブルデバイス」は？

トランジスタ利用効率で大きく劣る
↓
データ語長が短い応用でないと高性能は難しい
(GRAPE-7:次の講演)

別のアプローチ？

- 多数の演算器を 1 チップに集積、並列動作させて高い性能を得た専用計算機の特徴を生かす
- 「重力だけ、天文だけ」と言われないようにする

↓

GRAPE-DR アーキテクチャ

GRAPE における並列性

する計算:

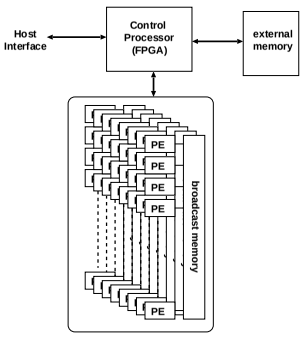
$$a_i = \sum_j f(r_i, r_j, m_j)$$

i についても j についても並列 (j は総和が必要)

1. パイプライン演算 (j 並列)
2. 物理/仮想マルチパイプライン (i並列)
3. 複数チップ (i並列/j並列)

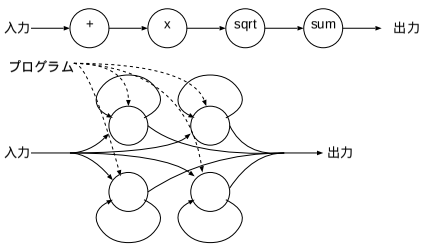
「パイプラインであること」はあまり本質的ではない？

GRAPE-DR の アーキテクチャ



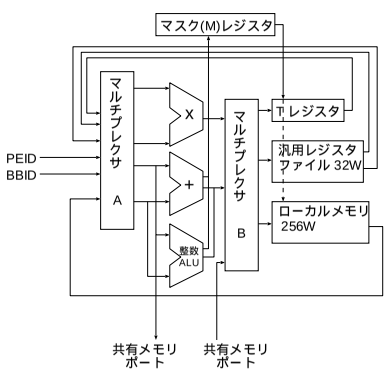
- 非常に多数のプロセッサエレメント (PE) を 1 チップに集積
- PE = 演算器 + レジスタファイル (メモリをもたない)
- PE はプログラムによって並列動作する
- チップ内に小規模な共有メモリ (PE にデータをブロードキャスト)。これを共有する PE をブロードキャストユニット (BU) と呼ぶ。
- 制御プロセッサ、外部メモリへのインターフェースを持つ

パイプライン処理と SIMD 並列処理



パイプラインでできることは SIMD 並列でもできる (縮約演算は追加ハードウェア必要)
専用パイプラインはいろいろメリットある
再構成可能だと、、、

PE の構造



- 浮動小数点演算器
- 整数演算器
- レジスタ
- メモリ (256 語),K とか M ではない。

「GRAPE として」使う

最も単純には

- 全ての PE が、自分の粒子への、同じ粒子からの力を計算
- 力を及ぼすほうの粒子データは外部メモリから供給

現実問題としては

- 違うブロードキャストユニットの同じ位置の PE には同じ粒子データを書く
- 各ブロードキャスト
- 力を及ぼすほうの粒子データはブロードキャストユニット毎に違うものにする。

つまり、 ブロードキャストユニット内で i 並列、ブロードキャストユニット間で j 並列とする。

他の応用

- 分子動力学・SPH 等の粒子法計算
- 密行列計算 (Linpack, LU 分解、固有値計算)
- 境界要素法: ポアソン方程式、ヘルムホルツ問題、、、
- ルジャンドル展開による極座標流体計算
- 分子軌道法での２電子積分

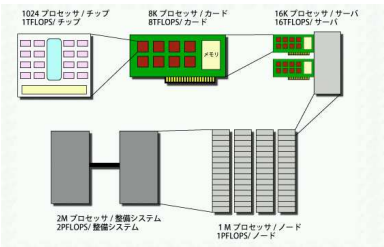
予算

平成１６年度科学技術振興調整費新規課題
「分散共有型研究データ利用基盤の整備」(代表:平木敬)

総額 3億 $(-\alpha)\times$ 5 年

最終システムのイメージ

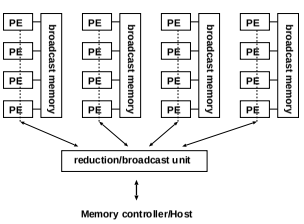
2007年度に基本的には完成 (2008年度に増強)
ピーク性能 2Pflops (単精度)
プロセッサチップ 2048 個。
プロセッサボード当り 4 チップ (PCI-X/Express)
ホストは 512 ノードの PC クラスタ



まとめ

- 次世代 GRAPE は予算がついた。
- 作るものは基本的には SIMD 動作する１チップ超並列プロセッサ。これで従来の GRAPE、MD-GRAPE の他、密度行列計算も可能にする。
- GRAPE 的なメモリ階層をもたせることで、 SIMD 計算機の難点を解消して高い価格性能比を実現する。

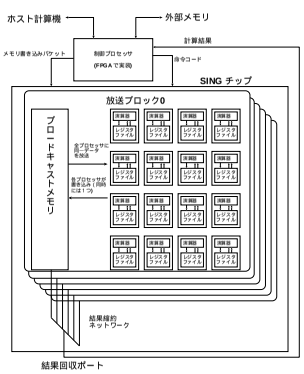
チップ内のネットワーク



- ブロードキャストユニットにメモリポートからデータを放送とランダム書き込みの両方
- ブロードキャストユニットからのデータをリダクション (合計とか) しながらホストに返すネットワーク。

の２つが必要。

ブロードキャストユニットの構成



- 命令コード、ブロードキャストメモリのアドレス等は外から供給される。
- 各 PE は同じデータを受け取る。

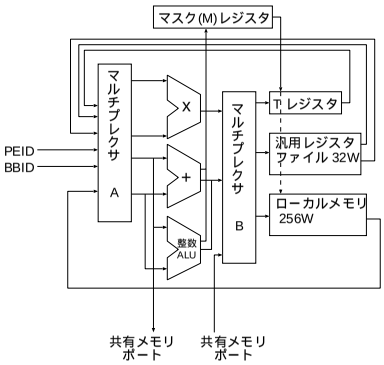
PE の詳細

演算:(最低限) 以下をサポート

- 倍精度浮動小数点加減算
- 倍精度固定 (ブロック) 小数点加減算
- 単精度浮動小数点乗算。倍精度はマルチサイクルで
- 区分多項式近似・スケーリングに必要なビット操作
- 条件付き実行

メモリオペレーション
ローカル共有メモリ: ブロードキャスト、シリアルな読出し、書き込み
汎用ポート: リダクション演算、放送

PE の構造



メモリコントローラ、シーケンサ

なるべくチップは設計を簡単にしたい

シーケンサは設計ミスがもっとも起きやすい場所
メモリコントローラはチップに作り込みたくない

面倒なものは全部 FPGA に任せる。
問題:命令ストリームを十分高速に渡せるか？
手抜きな対応: ベクトル命令なら問題なし。

プロセッサチップのイメージ

- 1024 PE
- 32 放送ブロック (各32PE)
- 500 MHz 動作
- 1 Tflops ピーク (内積演算の時に)

開発スケジュール

(現実的には)

- 夏までに PE の詳細決定、シミュレータでの評価。
- それから1年くらいでテープアウトしたい。
- それ以上は先のことは今考えても、、、

GRAPE に比べるとどれくらい損か？

GRAPE-6: 200万ゲート、 400 演算 = 5Kゲート/演算
PE の大きさ (推定):

ユニット	サイズ
fadd	7K
fmul	8K
register file	10K
合計	25K

場合によっては乗算・加算同時実行可能だとすると、合計では 12.5Kゲート/演算。純粋な GRAPE に比べると 2.5倍損
但し、レジスタファイルが 10Kゲート程度で十分かどうかは微妙。
共有メモリやインターフェース回路はサイズとしては無視できる。

類似のアプローチとの比較 (1)

- SIMD 超並列計算機 (Goodyear MPP, CM-1/2)
  - 外部メモリへのバンド幅、通信ネットワークでプロセッサ数が制限
  - プロセッサがメモリを共有することでプロセッサ数の制限を回避。応用範囲には制限。
- FPGA
  - トランジスタ利用率が低い
  - ソフトウェア開発が困難
  - どちらも回避

類似のアプローチとの比較 (2)

再構成可能計算機

- DSP ブロック入り FPGA
  - ゲート効率 FPGA よりはちょっといい？
- DAP/DNA, DRP
  - ゲート効率悪い
  - 動作速度遅い

他のアプローチとの基本的な違い:

チップ内にスイッチングネットワークをもたない

- (多少) できないことも発生
- デザインの単純化 → 高集積、高速動作

ソフトウェア

PE のプログラムは、基本的にシーケンシャル
(SIMD アプローチの利点)

- アセンブリ言語
- 単純なコンパイラ

上位のソフトウェア

- 複数ホスト+超並列ボード上の並列化
- ホスト計算機/超並列ボードのタスク割り当て

将来展望

- 多対多の相互作用向け SIMD 超並列計算機という新しい概念
  - 専用計算機並の高いコストパフォーマンス
  - 特定の物理系に縛られない広い適用範囲
  - 将来の半導体技術の発展を有効に利用可能
- ベクトル並列、スカラー並列と相補的
- compute-intensive な問題にブレークスルーをもたらす

COTS アプローチ

Intel Pentium 4 等の “off the shelf” プロセッサ

- 大量生産 → 低コスト
- 大量生産 → 莫大な開発費 → 高い性能

単一スカラープロセッサで COTS の性能を超えるのは非常に難しい
性能が memory limited — プロセッサを別のものにしても速くならない

COTS の限界: backward compatibility.
VLIW すら難しい (Itanium の失敗)。