パネルディスカッション
「次世代HPC計算基盤構築に向けて夢を語ろう」
ポジショントーク

牧野淳一郎 神戸大学

PCCC22 パネルディスカッション 2022/12/6

まとめ

- 半導体技術が深く「ポストムーア」の時代にはいり、なかなかスパコンの性能向上は期待できなくなっている
- では夢も希望もないか?というと、そうでもなくて、そう見えるのは我々が問題を十分理解していないからである
- 問題は２つで、一つは、 HPL のようなもっとも理論ピークに近い性能がでるベンチマークを考えても、電力やトランジスタバジェットのうち本当に演算に使われているのはほんのわずかであること、もう一つは、多くの実アプリケーションでは HPL より１桁以上実行効率が低いことである
- つまり、半導体技術があんまり進歩しなくても、特にアプリケーション性能はまだ 2-3 桁向上の余地がある
- それにはどうすればいいか？というのが我々の FS の課題である。とはいえ、基本的な方向は MN-Core で既に実現している (つもりである)

話の構成

- 半導体技術の方向 — ポストムーアの技術的制約はなにか？
- 「富岳」にみるポストムーア制約
- もうちょっと原理的なところから考えてみる
- まとめ

半導体技術の方向 — ポストムーアの技術的制約はなにか？

(今更ですが、、、)
ムーアの法則の半導体技術側のベース: CMOS スケーリング則 (デナード則):
トランジスタのサイズ (1 次元的) を半分にした時に

- 電圧を半分にする。寄生容量は半分になる。
- 速度は 2 倍、スイッチングの消費電力は 1/8、
- (トランジスタの値段は 1/4)

つまり: 並列性を完全に利用できるとすれば、同じ面積のチップで、トランジスタサイズが半分になると消費電力が同じで性能が 8 倍になる。
1985-2015 の間、概ね 4.5 年 (3 年はちょっと嘘) でロジック LSI のプロセスルールは半分になったので、「並列性を有効に使えて」「デナード則が成り立つなら」計算の性能は同じ消費電力で 10 年で 100 倍。

実際には

CMOS スケーリングは 3 つの意味で終焉

- **低電圧化の限界—2006 年くらい、大体 1V (今は 0.4V という話もあるが低速)**
- **寄生容量低減の限界— 2014 年、 FinFET への移行と同時。**
- **コスト低減の限界—これも 2014 年、 FinFET への移行と同時。**

あともうひとつ: 配線遅延・配線消費電力の問題
微細化しても、「単位長さ辺りのキャパシタンス」はかわらない、ところが、「単位長さあたりの抵抗」は配線幅の二乗に反比例して増える:

- **配線長がスケールしても配線遅延は「デザインルールに反比例して」増える**
- **配線長一定だと遅延は二乗で増える。電力は減らない**

どういう問題が起こるか？

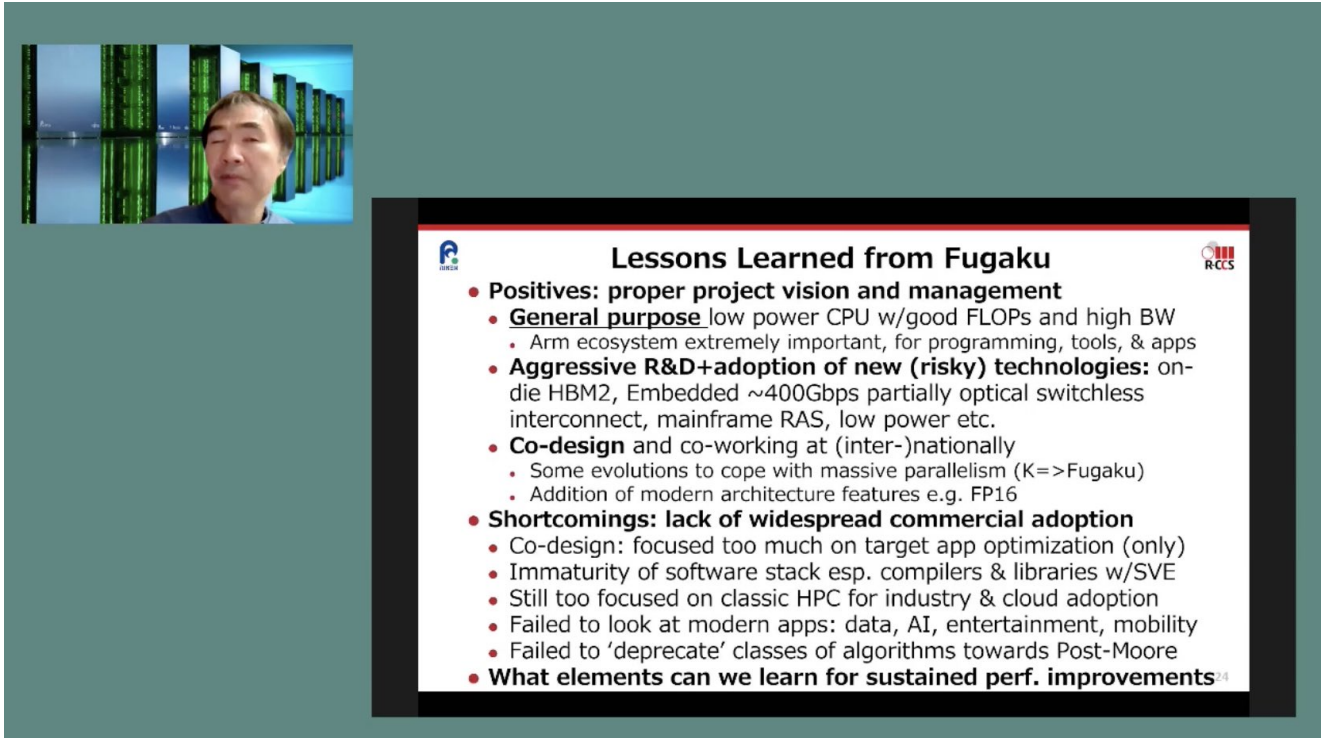
- プロセッサコア
  - 同じロジックでシュリンクしても性能があまり上がらない — SIMD 幅増やしてきた理由
  - SIMD 幅あまり増やすとコアの物理寸法が大きくなる – 遅延、消費電力が増える
  - 最近はトランジスタの価格が上がる – 高クロック化が必要になってきている
- チップ全体
  - 「ちょっと」長い配線 (共有 L2$ とか LLC までとか) の遅延が微細化が進むと大きくなり、電力が下がらない = LLC の遅延が巨大になり、バンド幅をあげられない –チップレット化の理由の 1 つ

「富岳」にみるポストムーア制約

- プロセッサコア
  - SIMD 幅は 512bit 2way。この辺が限界。
  - 演算、L1 レイテンシは 9 サイクルと大きい (x86 は 4 程度)
  - おそらく電力制約から OoO 資源は最小限 (x86 の半分程度)
- チップ全体
  - 共有 L2 アーキテクチャのため、L2 までの配線が長い (4 コアグループなのでチップサイズの半分にはなる): 非常に大きな遅延 (50 サイクルくらい)、
  - バンド幅も大きいとはいいがたい。コア当り 43/21 bytes/cycle。 B/F として 1 程度。メモリは 0.36。(load/store)

ピーク電力性能は CPU としては素晴らしく高いが、アプリケーション実効性能は、、、

これを使うほうから見ると、、、



SIAMPP22

「昔のアプリケーションに最適化されすぎてモダンなアプリケーションに対応できてない」

とはいえ要するに:「実用アプリケーションで性能でてないじゃないか」
(お前がいうなという声が聞こえますが)

ポスト富岳の方向性

- (例えば) A64fx のコアの延長はありそうにないようにみえる
- では代替は？ GPU なのか？
- そもそも「これが駄目だからあっち」という考え方でいいのか？

本質的な問題: では、どういうプロセッサが「一番良い」のか。そもそも「一番良い」とはどういうことか?

つまり問題は

- 実は我々は「このプロセッサはどれくらい良いか」を定量的に判断する規準をもっていない。
- ヘネパタ本にある「実用的なプログラム」での「コスト、性能、電力」は局所的。大域的に有効ではない。
- 「実用的なプログラム」は「既存の (だいぶ古い)」プロセッサを前提にしているから。

「高性能」という言葉は何を意味するべきか？

よくわかんないので他の業界を考える。

- エンジン: 燃料の熱エネルギーをなるべく沢山機械的運動エネルギーに変えられるのがよいエンジン。熱力学第二法則による限界がある。
- 飛行機: 主翼に対する摩擦抵抗以外は原理的には減らせるはず。誘導抗力、圧力抗力。
- (アルゴリズム: 演算数で比較はできる。)
- 「エネルギー最小」はどうか？

HPC における「エネルギー最小」の定義の問題

- エンジン: 「効率 100%」が定義可能。入力熱エネルギーから第二法則の限界まで機械的エネルギーに。飛行機も同様: 圧力抗力、誘導抗力ゼロ。
- 計算機: 半導体技術、動作電圧、ターゲット周波数とかで同じ論理回路でも電力は全く変わる。そもそも、アプリケーションだって色々ある。基準にできるものはあるか？
- でも実はもう半導体技術はこの先大して進歩しないので、半導体技術は given でいい。それなら、「演算に必要な電力と総電力の割合」でどうか?
- そうすると、結局、効率を、「全消費電力と、その半導体技術で構成した理想的な演算回路の組合せ論理部分の消費電力の比」と定義できたことになる。(これは実は半導体技術に依存しない定義になっている)

アプリケーションはどれくらい色々あるか?

日、米、欧のエクサスケールプロジェクトでの対象アプリケーションを見ると:

分類	数
構造格子	14
非構造格子	13
粒子	2
ランダムグラフ	2
密行列	3
その他	3

「ランダムグラフ」は神経回路シミュレータ。NEST とかは実装は構造格子に近い。
「その他」の 1 つはゲノミクス。　日本の GENOMON(中身は BWA-MEM、、、)
今はもちろん深層学習が重要。これは結局 (低精度) 密行列。

つまり

- 構造格子・非構造格子が多い
- 残りの多くが深層学習と量子化学・物性計算 (密行列)
- あと粒子法とゲノム (探索+動的計画法)
- これくらいできれば実際上「汎用スパコン」
- 現状、深層学習が一番マーケットが大きいなら、深層学習プロセッサがおまけで他のこともできるのがよい?(「現実的な HPC のエコシステム」)
- (とはいえ 深層学習「専用」プロセッサは上手くいっているかというと、、、)

現実的汎用性からみた過去と現在のプロセッサ

- 現実のプロセッサについて、「アプリケーションでの演算部論理の消費電力の割合」をだすのはそんなに簡単ではない。
- 雑だが相関はありそうな指標: コアロジックの全トランジスタ数に対する、演算ロジックに使われていそうなトランジスタ数の割合。
- 過大評価: 演算器以外はフルに動いてるとは限らない。
- 過小評価: 効率 100%で演算器が動いてるわけではない
- ちょっと別の話: 特に 7nm 以降、「トランジスタの価格は上昇」
- それなりに意味があるかも？

データ

	Single Core	MIMD	In core SIMD	Global SIMD	Coherent Cache	Non-Coherent Cache	Local memory	Transistor efficiency
CM-2	-	-	-	✓	-	-	✓	30%
Illiac IV	-	-	-	✓	-	-	✓	7%
Cray-1	✓	-	-	-	-	-	-	6.5%
Intel i860	✓	-	-	-	-	✓	-	6.5%
PEZY-SC	-	✓	-	-	-	✓	✓	2.4%
NVIDIA A100	-	✓	-	✓	-	✓	✓	1.7%
Fujitsu A64fx	-	✓	✓	-	✓	-	-	1.3%
Intel Skylake	-	✓	✓	-	✓	-	-	0.8%
Intel P4	✓	-	✓	-	-	✓	-	0.3%
Intel Core2	-	✓	✓	-	✓	-	-	0.08%
SW26010	-	✓	✓	-	-	-	✓	-

表見ると:

- MIMD+In core SIMD+コヒーレントキャッシュでは 1% を超えるのは難しそう。
- A64fx は 1% 超えていて数字は確かに良い。
- A100 と PEZY-SC は 2%前後で、さらに良い。
- 昔のベクトルプロセッサや SIMD 超並列マシンは非常によい。

要するに: マルチコア+コヒーレント階層キャッシュで高い効率のプロセッサを設計するのは「無理」
ある意味当たり前: キャッシュが大量のトランジスタ (と電力) を制御しにくいやり方で消費する。

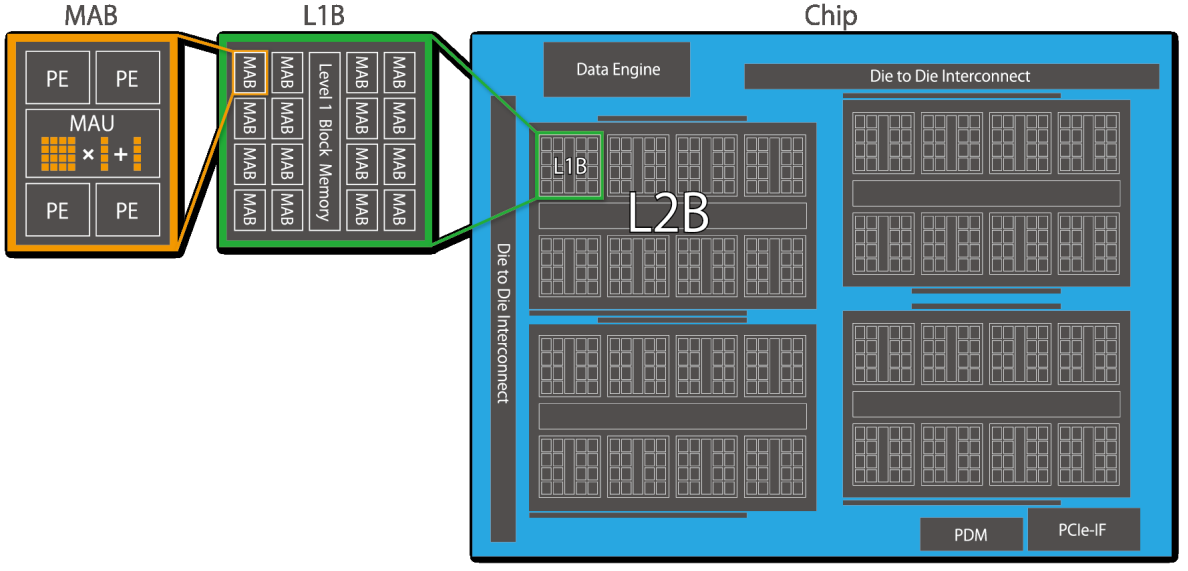
アプリケーション実行効率は、、、SC22 ゴードンベル賞の AMR PIC だと A100 で 1.2TF/device。19.5TF がピークなので、、、

必要な方向

ここは神戸大学 FS の方向。全然違う方向もありえる。

- near-memory computing. DRAM と演算器を (物理的に) 近くに、配線も短く。(3D 実装。TSV は高価なのでできれば 2 枚貼り合わせ)
- 階層キャッシュではなくて演算器に近いメモリを一番大容量にしてデータ移動を減らす。DRAM もそこからつなげる
- キャッシュではなく on-chip SRAM と DRAM の間のデータ移動を明示的に制御するアーキテクチャ
- MN-Core は既に大体そういう方向
- そんなののプログラミングはやってられないのでそこをなんとかする DSL やコンパイラ

MN-Core アーキテクチャ



PE のローカルメモリが一番大きい。L1B, L2B にもメモリはあるがそれらは転送のバッファリングのための最小限。

ソフトウェアの方向

- 深層学習なら TensorFlow, PyTorch, JAX とか (MN-Core で実用)。このレベルだとわりと本当にアーキテクチャに依存しない記述が可能 (OpenACC の pragma みたいなのはマシン依存、、、)
- あとはアプリケーションのタイプ毎に。構造格子なら JAX で、とか JAX を生成するものとか
- 例えば気象コードならそれ+OpenACC 的な、格子点での物理過程を Fortran90 で書けるものは必要

まとめ

- 半導体技術が深く「ポストムーア」の時代にはいり、なかなかスパコンの性能向上は期待できなくなっている
- では夢も希望もないか?というと、そうでもなくて、そう見えるのは我々が問題を十分理解していないからである
- 問題は２つで、一つは、 HPL のようなもっとも理論ピークに近い性能がでるベンチマークを考えても、電力やトランジスタバジェットのうち本当に演算に使われているのはほんのわずかであること、もう一つは、多くの実アプリケーションでは HPL より１桁以上実行効率が低いことである
- つまり、半導体技術があんまり進歩しなくても、特にアプリケーション性能はまだ 2-3 桁向上の余地がある
- それにはどうすればいいか？というのが我々の FS の課題である。とはいえ、基本的な方向は MN-Core で既に実現している (つもりである)