Vol.2021-HPC-178 No.10
2021/3/15

次世代HPCシステムのためのプロセッサアーキテクチャ評価環境と電力性能予測

有間 英志[1,a)]　児玉 祐悦[2]　小田嶋 哲哉[2]　辻 美和子[2]　佐藤 三久[2]

概要：ポストエクサスケールシステムを含む次世代 HPC システムでは、ワークロードの多様化、半導体製造プロセスの微細化限界、電力供給限界等、様々な問題に直面することが予想されており、この様な状況の下、所望の性能向上を達成するためには、ハードウェアアーキテクチャの最適化がより一層重要となる。本稿では、その中でも特に重要なプロセッサのマイクロアーキテクチャに焦点を当て、特に、スーパーコンピュータ富岳に搭載されている A64FX のアーキテクチャ、コンパイラ、コード性能、並びに電力評価等のために用いられてきた、Gem5 や McPAT 等のツールに対して拡張、整備を行い、さらにそれを用いた予備評価を行う。具体的には、A64FX プロセッサを 3nm までスケーリングさせた際の電力、面積削減効果を見積り、その上で、チップ電力、面積等の制約の下、スケーリングによるチップのスループット向上効果について、特に、コア数、SIMD 幅、FP パイプライン幅をスケールさせて検証する。その結果、コア数の増大による性能向上は見込めるものの、SIMD 幅、パイプライン幅の増大については、コアの複雑さ、特に L1 データキャッシュの複雑さの増大により、性能向上が頭打ちとなる事を確認した。これら定量的評価結果を元に、今後の HPC 向け SIMD プロセッサの方向性について議論を行う。

1. はじめに

スーパーコンピュータ「富岳」は、2020 年に理化学研究所計算科学研究センターに導入され、ISC'20/SC'20 にて発表された Top500 を含む複数の性能ランキングにおいて、1 位を獲得するに至った [2], [3], [5]。具体的には、2020 年 11 月現在、ピーク性能において 537PFLOPS、Linpack 実行時でも 442PFLOPS もの性能を達成しており、HPL-AI ベンチマークでは、世界で初めてエクサスケールに到達し (2.0EFLOPS)、その他、実アプリケーションに近い Graph500、HPCG ベンチマークでも、軒並み高い性能を誇っている。この様に幅広い尺度において高い性能を示すに至った一因として、富岳に導入されている A64FX プロセッサが挙げられ、この具体的な特徴は以下の通りである: (1) 命令セットアーキテクチャに SVE 拡張版の Armv8.2-A を採用、(2) 効率の良いアウト・オブ・オーダーコアを多数導入したメニー・コア型アーキテクチャの採用、(3) メインメモリに HBM2 を採用し、1 ノード当たり最大 1TB/s のメモリバンド幅を実現 [30], [34]。

一方、ポストエクサスケールシステムを見据えた次世代 HPC システムの研究開発では、この様な既存システムの利点を踏襲しつつも、ワークロードの多様化による要求性能の変化、半導体製造プロセスの微細化限界、電力供給限界等、様々な問題を勘案した上で、所望の性能向上を達成する必要がある。特に、半導体製造プロセスの微細化に伴って、これまでの HPC システムの指数関数的な高性能化がなされてきたこともあり、今後、これまで通りの性能向上を達成するには、ハードウェアアーキテクチャの最適化がより一層不可欠となる。そのためには、既存の HPC システムの設計を踏襲しつつも、電力、性能、コスト等様々なファクターを考慮した、マイクロアーキテクチャの設計空間探索だけでなく、ドメイン特化型のアクセラレータや新しいハードウェア機能の検討も可能なシミュレーション環境が不可欠となる。

この様な背景のもと、本稿では、将来的な HPC システム向けプロセッサの性能、電力の予測を行うための、プロセッサアーキテクチャ評価環境の整備を行い、さらに実際にこれを用いた電力性能評価を、特に半導体製造プロセスの微細化によるコストの増大、電力供給限界等を勘案しつつ行う。具体的には、現行の FinFET トランジスタによって到達可能と見られている 3nm までを対象とし [6]、A64FX プロセッサをスケーリングさせた際の電力・面積削減効果を見積り、その上で、チップ電力及び、面積制約の下、スケーリングによるチップのスループット向上効果

[1] 東京大学情報基盤センター
[2] 理化学研究所計算科学研究センター
[a)] arima@cc.u-tokyo.ac.jp

を、特に、コア数、SIMD 幅、FP パイプライン幅をスケーリングさせて検証する。さらに、これら定量的評価結果を元に、今後の HPC 向け SIMD プロセッサの方向性について議論を行う。

本稿における具体的な貢献は以下の通りである。

(1) 当該評価・分析を行うための、ワークフローやフレームワークの整備を行なった。特に、理研等における先行研究 [18], [19], [25], [30] にて用いられてきた Gem5[12] 及び McPAT[20] を選定し、特に電力評価部分について、A64FX 実機等からパラメタを推定し、これを用いた環境の調整を行なった。
(2) 上述の評価環境を用いて、将来の HPC プロセッサのための電力性能予測を行なった。ここでは、アクセラレータ等は考慮せず、A64FX プロセッサを対象とし、これを 3nm までスケーリングさせて電力・面積削減効果を見積った。
(3) さらに、電力、面積制約の下、スケーリングによるスループット向上効果を見積り、それを元に今後の HPC 向け SIMD プロセッサの方向性について議論を行った。具体的には、チップの浮動小数点演算性能を左右する、コア数、SIMD 幅、FP パイプライン幅をスケールさせた際の性能向上を定量的に評価し、これを元にプロセッサの進化の方向性について議論を行なった。

本稿の構成は以下の通りである。まず、次章では関連研究及び本研究の位置付けについてまとめる。続く第 3 章では、評価環境と電力性能予測に関して、まず概要を述べたのち、その各構成要素について詳細を述べる。さらに第 4 章では、本評価環境を用いた電力、性能等に関する評価結果を示す。第 5 章では、本評価結果を元に、今後の HPC 向け SIMD プロセッサの未来について述べる。最後に第 6 章では、本稿のまとめと今後への指針について述べる。

2. 関連研究と本研究の位置付け

シミュレーション環境を用いたマイクロアーキテクチャの最適化及び電力性能予測は、これまでにも数多く行われてきた。例えば、キャッシュの制御アルゴリズムの最適化やハードウェア電力削減機構の検討に使われてきた [10], [11]。また、A64FX を含む多種多様なプロセッサ〜アプリケーションのコデザインに用いられている [18], [19], [25], [30]。さらには、シミュレーション環境を用いたアーキテクチャパラメタ空間探索を効率的に行うための、モデルや AI 等を用いた手法の研究も広く行われている [16], [17]。これらと比較した場合、本研究の特色は、A64FX を例として実際の HPC 向けプロセッサから電力等のパラメタを推定し、これを用いたシミュレーションにより、HPC 向け SIMD プロセッサの将来について、特にコア数、SIMD 幅、FP パイプライン幅という浮動小数点演算性能を左右する核となる部分に着目して定量的に評価し、これを元にプロセッサの進むべき進化の方向性について議論した点にある。

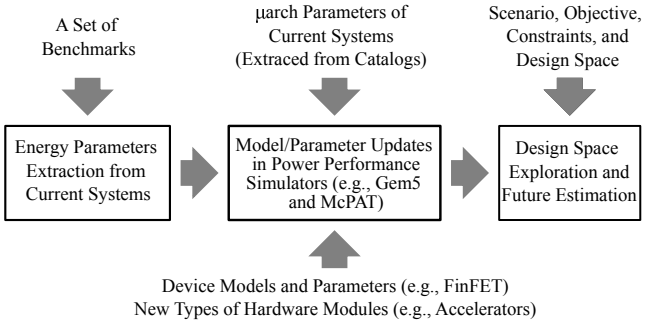
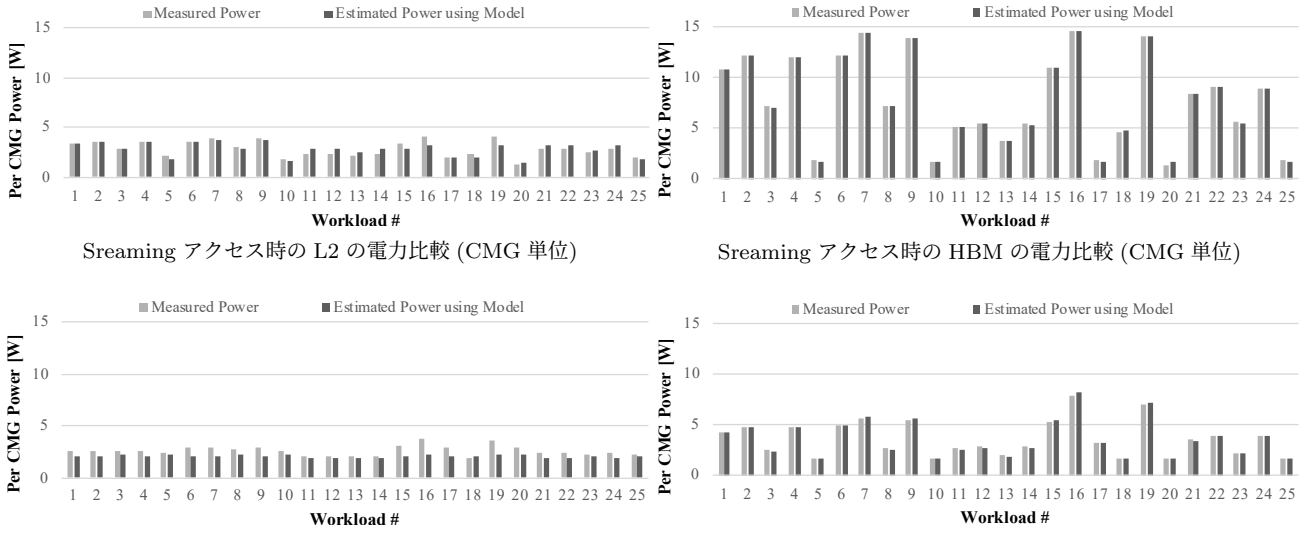


図 1 評価における全体的なワークフロー

Gem5[12]、McPAT[20] 等のシミュレーション環境自体も年々機能拡張を続けており、これにより、よりデバイス〜アクセラレータを含む多角的な視点を含んだプロセッサアーキテクチャの評価も可能となりつつある。J. Power らは Gem5 に対して GPU を含んだシミュレーションが行える様に拡張を行なった [28]。Y. S. Shao らはさらなる拡張を加え、SoC インターフェイスとアクセラレータとのコデザインを行なった [31]。A. Mohammad は分散コンピューティング環境のシミュレーションを行うための拡張を行なった [22]。A. Tang らは、既存の McPAT に対して、FinFET モデルを適用し、さらには製造プロセスばらつきも考慮した、電力評価ができるよう拡張しており [33]、A. Guler らは、モジュール、回路、トランジスタ等異なるレベルでの三次元実装も視野に入れた評価もできるよう拡張している [15]。

3. 評価環境と電力性能予測

図 1 に、本研究における電力性能予測のための、全体的なワークフローを示す。中央部分は、電力性能予測のためのシミュレーション環境である。本研究では、性能評価には Gem5[12]、電力面積評価には McPAT[20] を用いており、前述の通り、これらは幅広い用途で用いられている。本稿では、A64FX プロセッサを元に、HPC 向けプロセッサの将来の進むべき方向性を確認するのが目標であり、これを行うために、上記ツール内のパラメタやコードの修正を行う。具体的には、A64FX のマイクロアーキテクチャパラメタ及び電力パラメタの双方を、これらの評価環境に入力し、性能、電力、面積の見積もりを行った上で、これらのスケーリングに応じた変化について、FinFET デバイスモデルを用いて評価を行う。ここで、マイクロアーキテクチャパラメタについては、理研における既存の研究 [18], [19], [25], [30] を踏襲し、富士通から公開されている A64FX Microarchitecture Manual[1] を用いてパラメータチューニングされた RIKEN Simulator[4] を用い、電力パラメタについては図に示す通り、シンセティックなベンチマークコードを用いて実機からパラメタを推定すること



Sreaming アクセス時の L2 の電力比較 (CMG 単位)

Sreaming アクセス時の HBM の電力比較 (CMG 単位)

ランダムアクセス時の L2 の電力比較 (CMG 単位)

ランダムアクセス時の HBM の電力比較 (CMG 単位)

図 2 計測電力とモデルによる予測値の比較

によって行う。

本章の残りは次の様に構成される。まず 3.1 章では、具体的な実機を使った電力パラメタ推定手法について述べる。次に 3.2 章では、具体的に行なったツールの修正及び、これを用いた評価について、その前提条件を述べる。

3.1 A64FX 実機を用いた電力パラメタ推定

本稿では、特に L2 キャッシュ、メインメモリ (HBM)、及び FPU の電力パラメタを実機から推定し、シミュレーション環境に反映した。ここでは、特に重要なこれらのコンポーネントに限定しているが、ハードウェアパフォーマウンスカウンタの対応状況によっては、ベンチマークを適切に用意することで、その他コンポーネントの電力パラメタの推定も可能である。

3.1.1 L2 キャッシュ・HBM の電力パラメタ推定

一般的に、キャッシュやメインメモリを含むメモリシステムの電力 (P_*) は以下の様な数式で表すことができる (* は l2 や hbm 等対象とするメモリ階層)。

$$P_* = E_*^{dy} F_* + P_*^{st} \quad (1)$$

ただし、E_*^{dy}、F_*、P_*^{st} は対象メモリ階層に対する 1 回当たりのアクセスエネルギー (キャッシュライン単位)、単位時間当たりのアクセス回数 (キャッシュライン単位)、スタティック電力を表す。

一方で A64FX では、ハードウェアパフォーマンスカウンタの値から、L2 及びメインメモリ (HBM) の消費エネルギーの推定値を取得することが可能であり、さらに、これらのコンポーネントに対するアクセス数も取得することが可能である (CMG[*1] 単位での集計となる)。これらを実行時間で割ることにより、前述の P_* 及び F_* を求めることができ、これらペアを複数用いて回帰分析を行うことで、電力パラメタ E_*^{dy} 及び P_*^{st} を推定することができる。ただし、その他の商用プロセッサでも同等の機能を有するのが一般的であるため、本手法は一般的に成り立つものであることに注意されたい。

そこで、実際に A64FX を搭載するシステム (FX700) を用いて当該実験を行なった。具体的には、配列へのストリーミングアクセスを行うコードを用意し (B[]+=A[])、各メモリ階層へのアクセス頻度が変化する様に、データサイズ (16MiB, 64MiB, 256MiB, 1GiB, 4GiB) 及びスレッド数 (4, 8, 12, 16, 48) を変化させて実行し、実行毎に対象区間のカウンタ値を取得し、得られた値を用いて回帰分析を行なった。実機評価環境は、表 1 の通りである。スレッドアフィニティの設定は scatter であり、ストリーミング、ハードウェアパフォーマンスカウンタへのアクセスは PAPI6.0[23] を用いて行なった。その結果得られたパラメタを表 2 に示す。

さらに、この評価結果の当てはまりについて、ストリー

表 1 FX700 システム構成

Name	Remarks
CPU Package	A64FX@1.8GHz x1 socket, 4CMGs, 12 cores/CMG, 8MiB L2/CMG (= total 48 cores, 32MiB L2)
Main Memory	8GiB HBM@256GiB/s per CMG (= total 32GiB, 1TiB/s)
Compiler	Fujitsu C/C++ Compiler 4.2.0, **Options:** -Kfast -Kopenmp_simd -Kopenmp -O3
OS	Red Hat Enterprise Linux release 8.0

表 2 推定されたメモリ電力パラメタ

Component	Parameters
L2 cache	E_{l2}^{dy}=2.44 ± 0.31[nJ],P_{l2}^{st}=1.51 ± 0.18[W]
HBM	E_{hbm}^{dy}=14.8 ± 0.1[nJ],P_{hbm}^{st}=1.62 ± 0.04[W]

[*1] Core Memory Group: 複数コア、L2 キャッシュ、メモリコントローラからなる集合であり、A64FX プロセッサは複数の CMG によって構成される。

```
//copy loop
#pragma omp parallel for simd
for (i = 0; i < N; i ++)
        B[i] = A[i];
```

```
//mla loop
#pragma omp parallel for simd
for (i = 0; i < N; i ++)
        B[i] = A[i] * A[i] + 2.0;
```

```
//add loop
#pragma omp parallel for simd
for (i = 0; i < N; i ++)
        B[i] = A[i] + 2.0;
```

```
//mul loop
#pragma omp parallel for simd
for (i = 0; i < N; i ++)
        B[i] = A[i] * A[i];
```

図 3 浮動小数点演算に要するエネルギーの算出用コード

ミング (上記と同様) 及びランダムアクセスを行うコード (A[I[]]+=x) を用いて検討を行なった。その評価結果を図 2 に示す。ただし、横軸の番号はある実行条件 (スレッド数、データサイズ) に対応しており、全部で 25 通りである。ストリーミングとランダムアクセスを比較した場合、L2、HBM 共にランダムアクセス時の電力が低くなっているが、これはランダムアクセスでは一般的にアクセス頻度が下がるためである。また、L2 のランダムアクセスでは、誤差が若干増大しているが、これは、キャッシュコヒーレンスや CMG[34] を跨ぐトラフィック増加等の影響であると考えられ、それによる影響を F_* に反映する等により、対処することができる。

3.1.2 FPU の電力パラメタ推定

浮動小数点演算 1 回当たりに要する FPU のエネルギーを算出するため、図 3 に示すようなコードを用いた。図中の最上ループはコピーのみを行うものであり、それ以外は、ロード・ストア数不変のまま、1 ループあたり各々積和 (mla)、和 (add)、積 (mul) を 1 命令追加で行うものである。ここで、本評価環境では、対応した sve 命令が追加されることを確認している。前述同様、各々のループの前後で PAPI を用いてハードウェアカウンタにアクセスし、総浮動小数点演算回数 (N_*) 及びコア部分の消費電力 (P_*) を各々のコードで取得できる (* = mla, add, mul, copy)。ただし、$N_{copy} = 0$ となる。

各浮動小数点演算を 1 要素実行するのにかかるエネルギー (fetch から commit までの全パイプラインステージを含む) を E_* とおくと (* = mla, add, mul)、上述のコピーのみを行うループと比較することにより、これは以下の数式で表すことができる。ただし、T_* は総実行時間であり、$T_* \simeq T_{copy}$ (* = mla, add, mul) となることを確認している。

$$E_* = (P_* - P_{copy})T_*/N_* \qquad (* = mla, mul, add) \quad (2)$$

上述のエネルギー E_* のうち、FPU での演算を除いた部分のエネルギーを E_{else} とおき、FPU での 1 要素の演算に要するエネルギーを E_{flop} とおけば、E_* (*=mla, mul, add) は以下の様に表すことができる。

$$E_* = \begin{cases} 2E_{flops} + E_{else}, & (* = mla) \\ E_{flops} + E_{else}, & (* = mul, add) \end{cases} \quad (3)$$

これら定式化を元に、E_{flop} 及び E_{else} を求めた。具体的には、E_* (* = mla, add, mul) を求める際には、総データサイズ 1GiB、2GiB、4GiB、8GiB の 4 通りについて、各ループを 1K 回実行し、平均を取ることでこれを行い (48 スレッドで実行)、E_{else} は $E_{add} + E_{mul} - E_{mla}$ から求め、E_{flops} は $E_{mla} - E_{add}$ と $E_{mla} - E_{mul}$ の平均を取ることにより求めた。この評価結果をまとめたものを表 3 に示す (ただし、倍精度の結果である)。FX700 では、1 コアで 56GFLOPS、チップ全体で 2.7TFLOPS の浮動小数点演算性能を有していることから、表のパラメタを用いることで、FPU の動作消費電力は 1 コア当たり最大約 0.71W 程度、チップ全体で最大約 34W と推定され、E_{else} 部分を含めると、前者は約 0.87W、後者は約 42W と推定される。

3.2 シミュレーション環境と評価内容

まず、A64FX プロセッサにおけるアプリケーション実行時の性能や各コンポーネントの利用率等詳細な統計情報を取得するため、理研等における先行研究 [18], [19], [25], [30] を踏襲し、富士通から公開されている A64FX Microarchitecture Manual[1] を用いてパラメータチューニングされた RIKEN Simulator[4] を用いる。一方、電力や面積評価についてはマイクロアーキテクチャ分野で広く用いられている McPAT、特にその FinFET 拡張版 [15] を用い、7nm における、FPU、L2 キャッシュのエネルギーが前述の評価結果と同等になるように、シミュレータ内で用いられているモデルに係数を掛けることで行なった。その他、メモリについてはメモリコントローラの PHY 部分の面積、電力が HBM 向けのそれとなる様に係数を掛けて調整し [9]、HBM 部分の電力評価は McPAT には存在しないため、この部分についてはスケーリングとは独立に、前述の実測に基づく線形回帰モデルをそのまま用いている。また、スケーリングによる静電容量等のパラメタの変化はツールのものを用いている。

表 3 推定された倍精度浮動小数点演算にかかる各種エネルギー

E_{mla}	E_{add}	E_{mul}	E_{flops}	E_{else}
31.10[pJ]	18.83[pJ]	17.94[pJ]	12.71[pJ]	5.676[pJ]

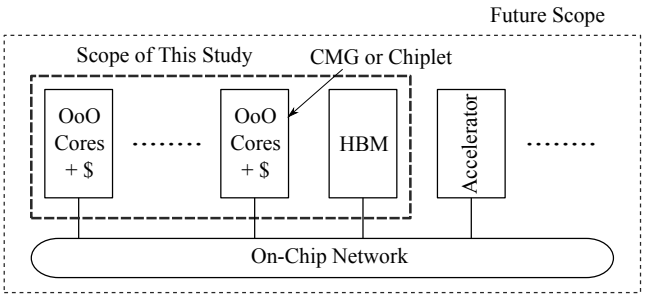


図 4 本稿における評価のスコープ

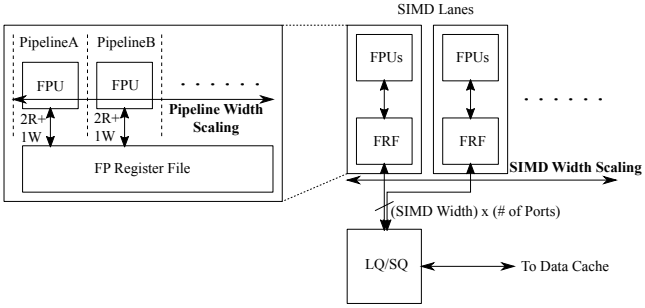


図 5 SIMD 幅及び FP パイプライン数のスケーリング

評価については、まず、7nm における A64FX プロセッサの電力 (本稿では、DGEMM または STREAM コード実行時)、及び面積について実機とシミュレーション結果を比較し、その上でさらに、3nm までスケーリングさせた場合の面積、電力削減効果を見積もる。ここで、DGEMM については、富士通コンパイラの BLAS ライブラリを用いており、STREAM については、自作の配列のコピーのみを行う自作のコードを用いた。

その上で、3nm スケーリング時における、チップの浮動小数点演算スループットの向上効果について、特に、電力、面積制約を考慮しつつ行う。将来的な HPC 向けプロセッサについては、アクセラレータや異なるメモリデバイスを含むヘテロジニアスな構成も有効であるが、本稿でのスコープは図 4 に示すとおり、A64FX の設計を踏襲した、アウト・オブ・オーダーコア部分や HBM 部分のみに限る。このアウト・オブ・オーダーコア部分について、総浮動小数点演算性能のピーク値 (R_{peak}) は、演算器が 1 クロック内に 1 要素に行える最大演算数 (α)、1SIMD 演算のビット数 (W_{simd})、コア内の FP パイプライン数 (W_{pipe})、1 要素のビット数 (N_{bit})、コア数 (N_{cores})、動作周波数 (F) を用いて次の式で表される。

$$R_{peak} = \alpha \times W_{simd}/N_{bit} \times W_{pipe} \times N_{cores} \times F \quad (4)$$

本稿では、まずコア数 N_{cores} のみスケーリングさせた場合に到達できるピーク性能の向上について考え、その上で、コア内の浮動小数点性能、すなわち、SIMD 幅 (W_{simd})、及び FP パイプライン幅 (W_{pipe}) を向上させた場合の更なる面積、電力効率の向上について検証を行う。このスケーリングに関する詳細は、図 5 に示す通りである。まず、FP

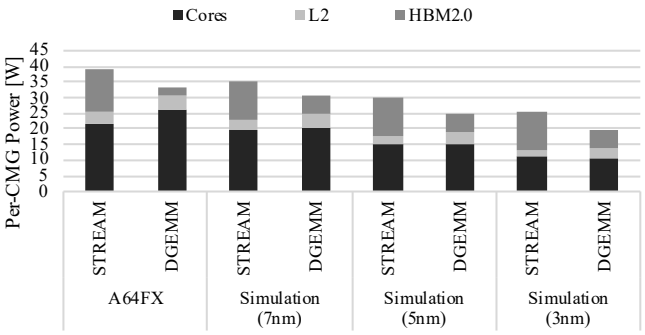


図 6 STREAM/DGEMM 実行時の電力比較及びスケーリングによる電力削減効果

レジスタファイル (FRF) は複数要素 (評価では倍精度で 8 要素) ごとに分割され、各々の FP レジスタファイルは対応する FPU 群に密に接続された状況を考える。SIMD 幅をスケーリングさせる場合、この固定されたサイズの FPU 群及び FP レジスタファイルのペアの数をスケーリングする事に相当する。一方、FP パイプライン数をスケーリングさせた場合には、FP レジスタファイルに接続される FPU 数も増える事になり、その結果として必要なポート数もそれに応じて増やす必要があり、その上、各種リソースの幅やエントリ数も増やす必要がある。これらに加えて、L1 データキャッシュへのアクセスについても、スループットを向上させる必要があり、本評価では、ラインサイズを SIMD 幅及び FP パイプライン幅に応じてスケーリングさせている。さらには、L1 データキャッシュの容量についても、配置できるパイプライン当たりのベクトル本数を一定に保つため、スケーリングさせている (そうでなければ、ベクトル化がより困難となるためである)。これら、リソースの変更についてまとめたものを表 4 に示す。

4. 評価結果

図 6 では、STREAM/DGEMM 実行時の実機で取得した電力、シミュレータが算出した電力を比較しており、さらにスケーリングによる電力削減効果をシミュレーションにより見積もっている。ただし、縦軸の電力は 1CMG 当たりのものである。DGEMM では特にシミュレーションによる数値の方が小さく出力されているが、これは、実機ではセクタキャッシュ等の利用によりコア当たりの性能が 48.4GFLOPS 出ているのに対し、シミュレーションではその様な最適化には対応しておらず、32GFLOPS 程度が限界

表 4 FP リソースのスケーリング評価に用いる設定

W_{simd}xW_{pipe}	Parameter Settings
512x2	A64FX default
1024x2	(linesize)x2, (L1 dcache size)x2, (FRF size)x2
2048x2	(linesize)x4, (L1 dcache size)x4, (FRF size)x4
512x4	(linesize)x2, (L1 dcache size)x2, (FRF size)x2, (FRF # of ports)x2, (commit/issue width)+=2, (# of hardware threads)x2

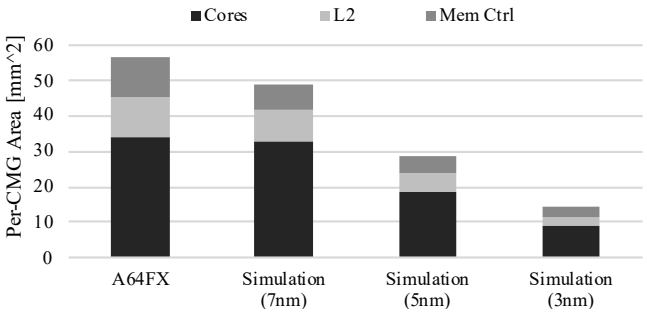


図 7 面積比較及びスケーリングによる面積削減効果

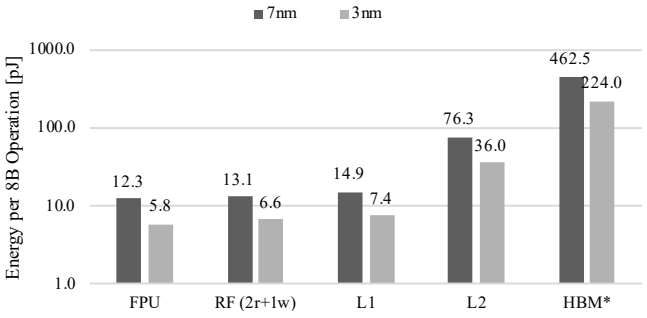


図 8 SIMD1 要素の操作に要するエネルギー (倍精度)

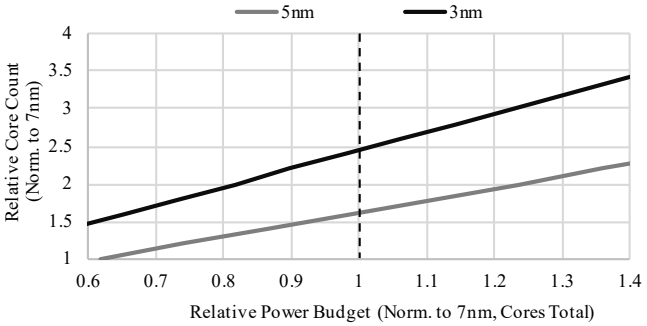


図 9 電力バジェットに対するコア数のスケーリング効果

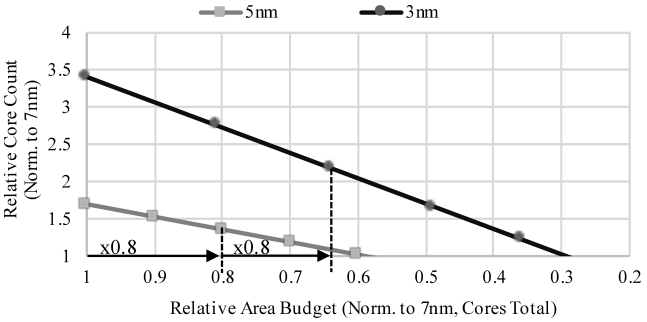


図 10 面積バジェットに対するコア数のスケーリング効果

となるためである。FPU 及び FP レジスタファイルの利用率が 100%となった場合には、シミュレーションにおいても、実機の DGEMM と同等のコア電力を消費することを確認している。また、シミュレーションについて、STREAM 実行時のコアの消費電力は L1 データキャッシュが支配的なのに対し、DGEMM では、L1 データキャッシュの電力はそれと比較すると少なくなっており、FPU や FP レジスタファイルの電力も大部分を占めている。また、HBM 部分については、HBM2.0 で固定としているため、電力はほぼ変化していないが、これを HBM3.0 等に置き換えることで、電力削減効果が見込める。

さらに、実機とシミュレーションの面積比較、及びスケーリングによる面積削減効果のシミュレーション結果を図 7 に示す。ただし、ここでは、チップ内の 1CMG に着目しており、コア部分の総面積、L2 キャッシュ、メモリコントローラ部分について示している。ここで、実機の面積については、A64FX プロセッサのパッケージのサイズが 60mmx60mm であることが分かっており、さらに、パッケージや CPU ダイの画像のピクセル数をカウントすることで、パッケージサイズに対する CPU ダイサイズの比率及び、CPU ダイサイズに対する各コンポーネントのサイズの比率を算出することができるため、これらを掛け合わせることで面積を算出した [21], [32]。図に示す通り、特にコア部分については、実機とほぼ同等の面積をシミュレータが出力しており、さらに、スケーリングによる面積削減効果も見込めることが分かった。

図 8 に FPU 及び各メモリ階層における、1SIMD 要素の操作に必要なエネルギーを、特に 7nm 及び 3nm の場合について示す。ここで、HBM については、7nm では HBM2.0、3nm では HBM3.0 を用いており、後者のエネルギーについては文献 [26] を参考にしている。また、図の縦軸はログスケールであることに注意されたい。図に示す通り、HBM へのアクセスエネルギーは他と比べて 1 桁以上相対的に大きくなっている。各コンポーネントに対する電力バジェットは限りがあるため、それによりスループットが制限を受け、結果として浮動小数点演算スループットに対するメモリバンド幅が足りなくなる。従って、FPU と同等のエネルギー消費に抑えられている、L1 データキャッシュや FP レジスタファイルにできるだけデータを格納し、計算を行うことが重要であることが、電力・エネルギーの観点からも分かる。また、3nm にスケーリングすることで、各階層で半分程度にエネルギーを削減できているが、倍精度ではなく、単精度、半精度の浮動小数点演算を用いることで、同様に半分、1/4 に削減できるため、これらを用いるのは数世代スケーリングを推し進めるのと同等の効果があることが分かる。

次に、電力バジェットに対するコア数のスケーリング効果を特に 5nm 及び 3nm の場合について図 9 に示す。ただし横軸は与えられた電力バジェットを表しており、縦軸はコア数を表している (いずれも 7nm 時と比較した場合の相対値)。また、ここでの電力バジェットは、L2 キャッシュやメモリコントローラ等を除いたコア群全体のみを指している。図に示す通り、電力バジェットをこれまでと同等に設定した場合でも、コア数のスケーリング効果はあり、3nm では、2.5 倍程度にコア数をスケールさせることができる。

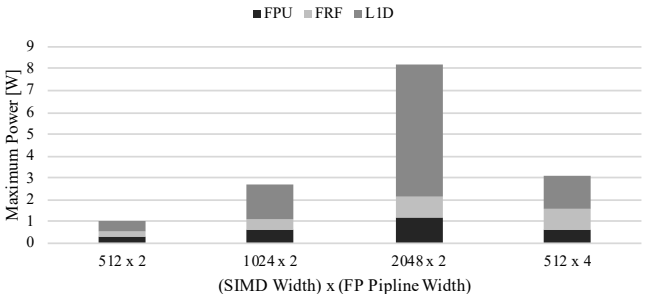


図 11 SIMD 幅、FP パイプライン幅のスケーリングに応じた電力の変化

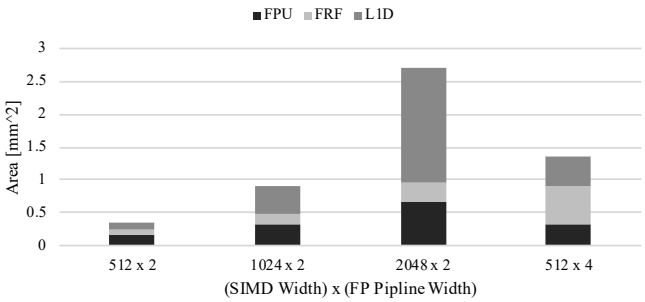


図 12 SIMD 幅、FP パイプライン幅のスケーリングに応じた面積の変化

一方、図 10 では、面積バジェットに対するコア数のスケーリング効果を特に 5nm 及び 3nm の場合について示す。ただし、ここでも、横軸は与えられた面積バジェットを表しており、縦軸はコア数を表しており、いずれも 7nm 時と比較した場合の相対値である。また、ここでの面積バジェットも同様に、L2 キャッシュやメモリコントローラ等を除いたコア群全体のみに限定する。スケーリングによる面積削減効果は期待できるものの、チップの製造コストはスケーリングに応じて増大していることが指摘されており [24]、そのため、コストを固定した場合の面積バジェットは削減される。仮に、スケーリング 1 世代毎に面積バジェットが 0.8 倍になると仮定した場合、図に示す通り、3nm 時点ではコア数は 2 倍強にスケーリングされる。

以上により、電力・面積の両観点から、スケーリングによるコア数の増大効果は 2 倍程度見込めることが分かったが、ここでは、さらに、コア内のリソース量の変化 (特に SIMD 幅及び FP パイプライン幅) についても確認し、更なるチップのスループット向上効果が見込めるかどうかを確認する。この評価の詳細については前述の通りであり、表 4 に具体的な設定を示している。これらをスケーリングさせた際の電力の変化を図 11 に (ただし最大電力である)、そして面積の変化を図 12 に示す。前述の通り、演算律速のワークロードの場合、FPU、FP レジスタファイル (FRF)、及び L1 データキャッシュのバランスが重要であり (図 8 参照)、ここでは特にこれらに着目する。図に示す通り、SIMD 幅、FP パイプライン幅スケーリングのいずれの場合も、L1 データキャッシュのオーバーヘッドが大きく、面積、電力の面で足を引っ張ることが分かる。これは、RAM の面積、エネルギーが、一般的に、容量やポート数やラインサイズ等、バンド幅関連の変数の多項式関数で表され、特に 2 乗の項の影響が無視できないためである [29]。これは、配線の影響と考えられ、特にスケーリングが進んだ際には支配的となる。また、FP パイプライン幅のスケーリングについては、パイプライン幅の増大に応じてポート数を増やす必要があり、結果として、FP レジスタファイルのオーバーヘッドが他と比較して大きくなっている。

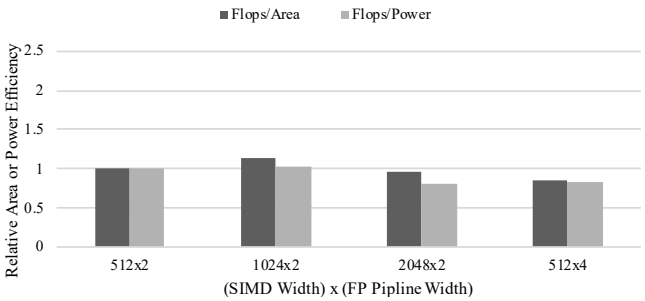


図 13 コア設定毎の面積対性能、電力対性能の比較

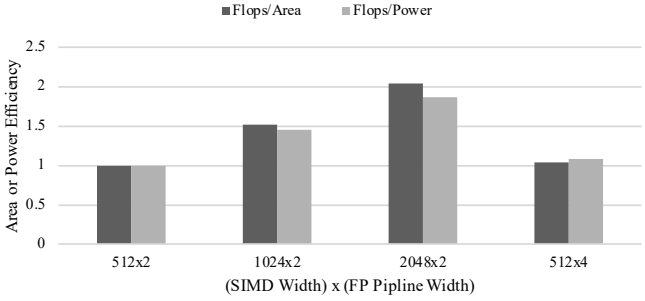


図 14 コア設定毎の面積対性能、電力対性能の比較 (Ideal)

次に、図 13 に、これらのコア設定について、面積当たりの性能及び電力当たりの性能を比較したものを示す。ただし、面積、電力については、前述同様にコア部分のみに限っており、性能はピーク倍精度浮動小数点演算性能を表している。図に示す通り、SIMD 幅や FP パイプライン幅をスケーリングさせても、前述のオーバーヘッドにより、面積対性能、電力対性能ともに向上しないことが分かる。特に、FP パイプライン幅のスケーリングについては、アウト・オブ・オーダー実行リソースの増大 (幅、エントリ数) が必要となり、そのためのオーバーヘッドが追加されるため、得策ではない。一方で、図 14 に、SIMD 幅や FP パイプライン幅をスケーリングさせても、L1 データキャッシュの面積や電力が増えないという理想的な状況の場合の面積対性能、電力対性能も示しておく。図に示す通り、この場合は、SIMD 幅スケーリングについては、面積対性能、電力対性能の向上が見込めることが分かる。

5. HPC 向け SIMD プロセッサの方向性

SIMD 幅のスケーリングについては、アウト・オブ・オーダー実行機構の複雑さを増やすことなく、FP 演算性能を

向上させることができるため、一見すると効果的の様に思われるが、実際には現時点の 512bit で限界に近づいており、この方向性は効果的でない。評価で示した通り、その分のレジスタファイル、L1 データキャッシュ等 RAM に対する負担 (特に配線による) が大きくなり、電力・面積効率の向上は難しい。一方、L1 データキャッシュの設計最適化 (特にマイクロアーキテクチャ的工夫) により、この部分のオーバーヘッドを抑えることができれば、SIMD 幅のスケーリングについてもスループット向上効果が期待できることが、評価の結果分かっている。しかし、その場合でも、L0 バッファの導入等様々な方向性が考えられるが、ラインサイズの増大による gather/scatter アクセスの非効率化などの問題もあり、それなりのモチベーション (HPC アプリケーションに内在する並列性は、短いベクトルでは抽出しきれない等) がなければ、安易に進まない方が良いと言える。また、FP パイプライン数を増やすのが得策ではないことも明らかである。

現在の A64FX の様な HPC 向け SIMD プロセッサコア内の演算器比率を増やすには、FP レジスタファイルや L1 データキャッシュへの面積、バンド幅負担を減らす様に、コア当たりの演算幅を減らしつつ、コア当たりの RAM のバンド幅、サイズを減らすのが方向性として正しい。しかし、その場合は、コア当たりの演算器面積を減らすことに繋がり、その結果、アウト・オブ・オーダー実行機構のオーバーヘッドが顕著になるために、この制御機構をシンプルにしたイン・オーダー CPU や GPU 等のスループットに特化したアクセラレータに近いデザインとなる。しかし、現在の A64FX プロセッサのデザインであっても、コア面積の大部分を FPU や FP レジスタファイルが占めているため、この方向に進むことによって得られる性能向上も実際にはそれ程大きくなく、プログラマへの負担に見合った性能向上が得られない可能性がある。結果として現在のデザインを維持したまま、コア数を増やすのが最善策の一つとなる。

一方、1 要素の操作にかかるエネルギーについて着目すると、FPU が問題という訳ではなく、FPU にデータを送り込むための、メモリ階層全体での電力消費及び RAM 面積の肥大化・非効率化こそが問題であり、これはスケーリングが進むにつれて顕著化してきた問題である。これを解決するためには、データの情報量について着目しつつ、その転送量を SW/HW の両観点から削減することこそが鍵となり、昨今の半精度や単精度の利用は一形態であると言える。一方で、それ以外の方向性についても、例えば、データ・キャッシュ圧縮 [8], [27] 等を利用するのは有力であり、例えば圧縮は HBM 側、及び解凍は演算直前で行う方式等も考えられ、さらには単純な操作は HBM 側で行う Processing-in-memory[7], [13], [14] 等も将来長い目で見れば有力な手段であると言える。特に、圧縮に関しては、レイテンシを多少犠牲にしつつも、バンド幅や容量をメモリ階層全体で稼ぐことができ、演算律速・メモリ律速アプリケーションの両者に有効である。

以上をまとめると、A64FX プロセッサの様な HPC 向け SIMD プロセッサの場合、コア数をスケールさせるのが有力であり、さらに、半精度・単精度の利用や圧縮等によりデータ転送量を削減し、見かけのバンド幅やキャッシュ容量を増やすことによって、メモリシステムの負担を減らすことが重要であり、これを前提とした設計であれば、更なる FP 演算性能の向上にもつながる。

6. まとめ

本稿では、富岳スーパーコンピュータに用いられている A64FX プロセッサを元に、将来的な HPC システム向けプロセッサの電力性能予測、及びその為の評価環境構築を行なった。具体的には、既存の Gem5 や McPAT といったツールを元に、A64FX 実機等から取得したパラメタを反映させ、FinFET 3nm までスケールさせた際の性能、電力、面積について評価し、その上で、コア数、SIMD 幅、FP パイプライン幅をスケールさせた際のチップのスループット向上効果について、定量的に議論を行なった。さらに、その上で、今後の HPC 向け SIMD プロセッサの方向性についても議論を行なった。

謝辞 本研究の一部は、”特定先端大型研究施設運営費等 補助金（次世代超高速電子計算機システムの開発・整備等）”の助成を受けたものである。

参考文献

[1] “A64fx microarchitecture manual,” https://github.com/fujitsu/A64FX (last accessed: Feb 12, 2021).
[2] “Graph500,” https://graph500.org/ (last accessed: Jan 23, 2021).
[3] “HPCG - November 2020,” https://www.top500.org/lists/hpcg/hpcg-november-2020/ (last accessed: Jan 23, 2021).
[4] “Riken simulator,” https://github.com/RIKEN-RCCS/riken_simulator (last accessed: Feb 12, 2021).
[5] “Top 500,” https://www.top500.org/lists/top500/ (last accessed: Jan 23, 2021).
[6] “International Roadmap for Devices and Systems,” 2020.
[7] J. Ahn, S. Yoo, O. Mutlu, and K. Choi, “PIM-Enabled Instructions: A Low-Overhead, Locality-Aware. Processing-in-Memory Architecture,” in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 336–348.
[8] A. R. Alameldeen and D. A. Wood, “Adaptive Cache Compression for High-Performance Processors,” in *31st Annual International Symposium on Computer Architecture (ISCA)*, 2004, pp. 212–223.
[9] G. Allan, “Choosing between DDR4 and HBM in memory-intensive applications,” http://www.techdesign-forum.com/practice/technique/choosing-between-ddr4-and-hbm-in-memory-intensive-applications/ (last accessed: Feb 11, 2021), 2018.

[10] E. Arima, "Classification-Based Unified Cache Replacement via Partitioned Victim Address History," in *2020 23rd Euromicro Conference on Digital System Design (DSD)*, 2020, pp. 101–108.

[11] E. Arima *et al.*, "Immediate Sleep: Reducing Energy Impact of Peripheral Circuits in STT-MRAM Caches," in *International Conference on Computer Design (ICCD)*, 2015, p. 149–156.

[12] N. Binkert *et al.*, "The Gem5 Simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, p. 1–7, 2011.

[13] J. Draper, J. Chame, M. Hall, C. Steele, T. Barrett, J. LaCoss, J. Granacki, J. Shin, C. Chen, C. W. Kang *et al.*, "The Architecture of the DIVA Processing-In-Memory Chip," in *16th International Conference on Supercomputing (ICS)*, 2002, pp. 14–25.

[14] M. Gokhale, B. Holmes, and K. Iobst, "Processing in Memory: The Terasys Massively Parallel PIM Array," *Computer*, vol. 28, no. 4, pp. 23–31, 1995.

[15] A. Guler and N. K. Jha, "McPAT-Monolithic: An Area/Power/Timing Architecture Modeling Framework for 3-D Hybrid Monolithic Multicore Systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 10, pp. 2146–2156, 2020.

[16] E. Ïpek, S. A. McKee, R. Caruana, B. R. de Supinski, and M. Schulz, "Efficiently Exploring Architectural Design Spaces via Predictive Modeling," in *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2006, p. 195–206.

[17] R. G. Kim, J. R. Doppa, and P. P. Pande, "Machine Learning for Design Space Exploration and Optimization of Manycore Systems," in *International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–6.

[18] Y. Kodama, T. Odajima, M. Matsuda, M. Tsuji, J. Lee, and M. Sato, "Preliminary Performance Evaluation of Application Kernels Using ARM SVE with Multiple Vector Lengths," in *International Conference on Cluster Computing (CLUSTER)*, 2017, pp. 677–684.

[19] Y. Kodama, T. Odajima, A. Asato, and M. Sato, "Accuracy Improvement of Memory System Simulation for Modern Shared Memory Processor," in *International Conference on High Performance Computing in Asia-Pacific Region (HPCAsia)*, 2020, p. 142–149.

[20] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," in *International Symposium on Microarchitecture (MICRO)*, 2009, pp. 469–480.

[21] S. Matsuoka, "A64fx and Fugaku - A Game Changing, HPC / AI Optimized Arm CPU to enable Exascale Performance," https://connect.linaro.org/resources/san19/san19-300k1/ (last accessed: Feb 11, 2021), 2019.

[22] A. Mohammad, U. Darbaz, G. Dozsa, S. Diestelhorst, D. Kim, and N. S. Kim, "dist-gem5: Distributed Simulation of Computer Clusters," in *2017 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2017, pp. 153–162.

[23] P. J. Mucci, S. Browne, C. Deane, and G. Ho, "PAPI: A Portable Interface to Hardware Performance Counters," in *Proceedings of the department of defense HPCMP users group conference*, vol. 710, 1999.

[24] S. Naffziger, "Chiplet Meets the Real World: Benefits and Limits of Chiplet Designs," in *2020 Symposia on VLSI Technology and Circuits*, 2020.

[25] T. Odajima, Y. Kodama, and M. Sato, "Performance and power consumption analysis of Arm Scalable Vector Extension," *The Journal of Supercomputing*, pp. 1–22, 2020.

[26] J. T. Pawlowski, "Prospects for Memory," in *Workshop on Memory-Centric High-Performance Computing (MCHPC)*, 2019.

[27] G. Pekhimenko, E. Bolotin, N. Vijaykumar, O. Mutlu, T. C. Mowry, and S. W. Keckler, "A Case for Toggle-Aware Compression for GPU systems," in *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2016, pp. 188–200.

[28] J. Power, J. Hestness, M. S. Orr, M. D. Hill, and D. A. Wood, "gem5-gpu: A Heterogeneous CPU-GPU Simulator," *IEEE Computer Architecture Letters*, vol. 14, no. 1, pp. 34–36, 2015.

[29] P. Raghavan, A. Lambrechts, M. Jayapala, F. Catthoor, and D. Verkest, "EMPIRE: Empirical power/area/timing models for register files," *Microprocessors and Microsystems*, vol. 33, no. 4, pp. 295–300, 2009.

[30] M. Sato *et al.*, "Co-Design for A64FX Manycore Processor and "Fugaku"," in *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2020.

[31] Y. S. Shao, S. L. Xi, V. Srinivasan, G. Wei, and D. Brooks, "Co-Designing Accelerators and SoC Interfaces using gem5-Aladdin," in *International Symposium on Microarchitecture (MICRO)*, 2016, pp. 1–12.

[32] T. Shimizu, "Supercomputer Fugaku: Co-designed with application developers/researchers," in *2020 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, 2020, pp. 1–4.

[33] A. Tang, Y. Yang, C.-Y. Lee, and N. K. Jha, "McPAT-PVT: Delay and power modeling framework for FinFET processor architectures under PVT variations," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 9, pp. 1616–1627, 2014.

[34] T. Yoshida, "Fujitsu high performance CPU for the Post-K Computer," in *Hot Chips*, vol. 30, 2018.