一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
CPSY2025-64,DC2025-136(2026-03)

# OSCARベクトルマルチコアの性能・電力評価

野谷　優仁† 　水本　幸希† 　権藤　創太† 　上林　嶺† 　朱　允楷†
北村　俊明† 　笠原　博徳† 　木村　啓二†

† 早稲田大学
E-mail: †notani@moegi.waseda.jp, kouki.mizumoto@fuji.waseda.jp, sota0147@asagi.waseda.jp, rei107@suou.waseda.jp, yunyun3099@moegi.waseda.jp, toshi.kitamura@aoni.waseda.jp, kasahara@waseda.jp, keiji@waseda.jp

**あらまし**　組み込み機器への深層機械学習は広く利用されつつあり，自動運転車やスマートロボットなどを動作させるためには必要不可欠な技術となっている．しかしながら，組み込みデバイス上で動作する深層機械学習の演算は高いコストを要するため，組み込み機器を前提としたシステムでは電力がより厳しい制約となる．ユースケース上想定される電力制限下で最大限の性能を得るためには，アーキテクチャの設計段階でシステムアーキテクチャレベルでの正確な性能・電力モデルが重要である．本稿では，著者らの開発したベクトルマルチコアアーキテクチャであるOSCAR ベクトルマルチコアについて，TSMC 28nm High Performance Compact Plus (28HPC+) を対象に論理合成後のデザインの性能・電力評価を行う．また，パフォーマンスカウンタと電力の分析を通じて，各構成ユニットの積和演算時の電力特性を明らかにする．

**キーワード**　ベクトルアクセラレータ，組み込みシステム，マルチコアアーキテクチャ，ヘテロジニアスシステム，電力評価

# Performance and Power Evaluation of OSCAR Vector Multicore

Masahito NOTANI†, Koki MIZUMOTO†, Sota GONDO†, Ryo KANBAYASHI†, Yunkai ZHU†,
Toshiaki KITAMURA†, Hironori KASAHARA†, and Keiji KIMURA†

† Waseda University
E-mail: †notani@moegi.waseda.jp, kouki.mizumoto@fuji.waseda.jp, sota0147@asagi.waseda.jp, rei107@suou.waseda.jp, yunyun3099@moegi.waseda.jp, toshi.kitamura@aoni.waseda.jp, kasahara@waseda.jp, keiji@waseda.jp

## 1. はじめに

自動運転車やスマートロボットのような組み込み機器においても深層機械学習は広く利用されている．深層機械学習を用いた組み込み機器は活発に研究・開発が行われており，トラックの自動運転，倉庫内の運搬・配送，医療や介護の現場での作業など，適用先は多岐にわたる [1]．人手不足の現場における労働力の代替や人手が必要な複雑な作業の自動化を行う上で，これらの深層機械学習を用いた組み込み機器には大きな期待が寄せられている．

しかしながら，組み込み機器上で動作する深層機械学習の演算はコストが高いため，実行時間や消費電力が問題となる．深層機械学習では，畳み込みニューラルネットワーク (CNN) が広く使用されており，プログラムの実行時間やエネルギー消費の大半を占める．CNN の演算を高速に行うためには，処理能力の高いハードウェアが必要となり，その結果として消費電力が増加する．このため，特に組み込み機器上での実行においては，消費電力が厳しい制約条件となる [2].

著者らは高性能かつ低消費電力なアーキテクチャを実現するために，コンパイラとハードウェアの協調動作が重要であるという観点から，OSCAR 自動並列化コンパイラと OSCAR マルチコアを開発してきた [3], [4]．特に，スマートロボットに対する取り組みとして，OSCAR ベクトルマルチコアと，OSCAR コンパイラを中心としたコンパイルフローの開発を進めてき

た [5]．OSCAR ベクトルマルチコアは，深層機械学習に用いられる CNN や行列積演算を加速するベクトルアクセラレータを各コアに搭載するマルチコアアーキテクチャである．OSCAR コンパイラを中心としたコンパイルフローは，学習済み深層学習モデルから OSCAR ベクトルマルチコアで動作する並列化・ベクトル化された実行コードを生成する．

想定されるユースケースにおいて，所与の電力制限下で最大限の性能を発揮するためには，OSCAR ベクトルマルチコアチップに搭載する演算器数，メモリ容量，コア数等を，それらの消費電力に基づき決定する必要がある．また，コンパイラの最適化やコード生成が消費電力に及ぼす影響の考慮も重要である．そのためには，アーキテクチャレベルでの正確な性能・電力推定モデルと，その実チップでの評価を通した検証が重要となる．

筆者らは，OSCAR ベクトルマルチコアとコンパイルフローの有効性を検証するために TSMC 28nm High Performance Compact Plus (28HPC+) のプロセス [6] を用いたマルチコアチップを開発した．本稿では，開発したマルチコアチップの論理合成後のデザインを対象として性能・電力評価を行い，実チップだけでは計測が困難な部分の評価を行う．また，設計評価ツールとして広く用いられている McPAT [7] は有用な枠組みを提供しているが，本研究では合成後のデザインによる評価を行うことで，設計を反映したより高精度な電力特性を明らかにする．さらに，パフォーマンスカウンタに基づく統計的解析を行い，構成ユニットの入出力に着目した少数のカウンタ情報から，より多くのカウンタを用いた場合と同程度の精度で電力を推定できることを示す．

本稿の構成は以下の通りである．2. 節で本稿で評価を行う OSCAR ベクトルマルチコアについて説明し，3. 節で性能・電力評価を行う．そして，4. 節でパフォーマンスカウンタを使った電力の解析を行う．

## 2. OSCAR ベクトルマルチコア

### 2.1 OSCAR ベクトルマルチコアの構成

OSCAR ベクトルマルチコアアーキテクチャは OSCAR マルチコアアーキテクチャ [4] のベクトルアクセラレータ拡張を行ったマルチコアアーキテクチャであり，従来の GPU に比べて深層機械学習の推論に特化したハードウェアとなっている．OSCAR 自動並列化コンパイラ [3] と協調動作することでマルチコア並列化・ベクトル化に加えてローカルデータメモリ最適化，データ転送スケジューリング最適化，電力制御最適化が可能となる．OSCAR ベクトルマルチコアは，中央集中メモリ (CSM) としてチップ内部に存在する On-chip CSM とチップ外部に存在する Off-chip CSM がある．また，各コアに RISC-V ISA の CPU，ベクトルアクセラレータ (VA)，ローカルデータメモリ (LDM)，データ転送ユニット (DTU) を持っている．LDM は同一コア内の CPU，VA，DTU からアクセスできるだけでなく，他のコアからもアクセスすることができるので，分散共有メモリ (DSM) としても機能する．DTU はプログラム実行機能を持たせることによって，CPU と独立して制御することが可能である [8]．

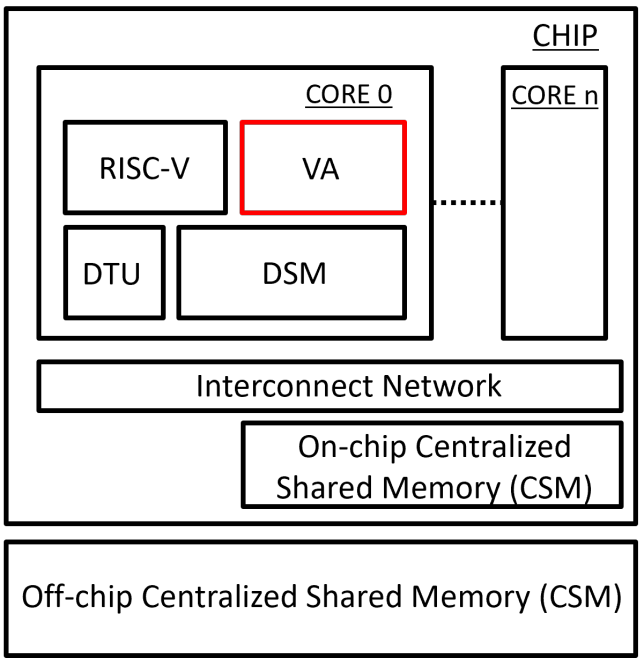


図 1　OSCAR ベクトルマルチコアの構成

### 2.2 ベクトルアクセラレータ

OSCAR ベクトルマルチコアアーキテクチャに搭載される VA はベクトルプロセッサ型のアクセラレータである [9]．本論文で評価した VA はスカラプロセッサ，加算・減算ベクトルパイプライン，乗算・除算ベクトルパイプライン，ベクトルロードストアパイプラインをそれぞれ持つ．レジスタファイルとしては整数レジスタ (SR)，浮動小数点レジスタ (FR)，ベクトルレジスタ (VR)，マスクレジスタ (MR) を持っており，それぞれのサイズは 256B，256B，16KB，2KB である．VA は整数演算と浮動小数点演算に対応している．各演算ベクトルパイプラインおよびベクトルロードストアパイプラインの同時に処理できる要素数は利用可能な半導体資源に応じて拡張可能である．本研究で用いた VA は，各ベクトルパイプラインの同時に処理できるデータ幅は 128bit であり，最大ベクトル長は 4096bit まで指定することができる．

## 3. 性能・電力評価

本節では，TSMC 28HPC+ プロセスを用いたマルチコアチップの論理合成後のデザインに対する性能および電力を示す．本評価で使用したプロセス，電圧，温度はそれぞれ Typical-Typical，0.9V，25°C である．論理合成および電力の評価には Synopsys 社の Design Compiler [10] を用い，ゲートレベルシミュレーション (GLS) とスイッチングアクティビティの取得には Cadence 社の Xcelium Simulator [11] を使用した．電力評価では，取得したスイッチングアクティビティを Design Compiler に読み込むことで特定のプログラムの電力評価を行った．特に断りのない限り，動作周波数はコアクロックを 300 MHz，メモリクロックを 100 MHz として評価を行う．

### 3.1 マルチコアチップの基本性能

マルチコアチップはコアクロックを 300 MHz，メモリクロッ

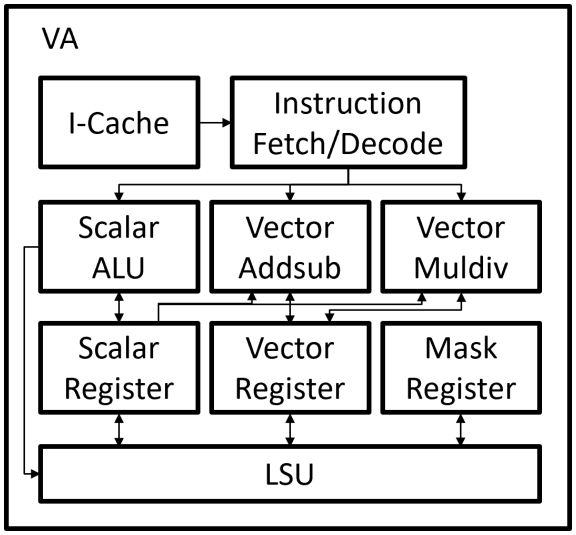


図 2　ベクトルアクセラレータの構成

クを 100 MHz として，4 コアで論理合成を行った．表 1 に開発したマルチコアチップの諸元を示す．

表 1　OSCAR ベクトルマルチコアの仕様

| | |
|---|---|
| コア数 | 4 |
| キャッシュサイズ | 32 KB |
| DSM サイズ | 128 KB |
| On-chip CSM サイズ | 1024 KB |

コアクロックのタイミングについては，クリティカルパスの信号到着時刻が 3.46 ns となるので動作周波数は 289 MHz となる．

面積の評価では，合成後のデザインに対してチップ全体および内部のインスタンスの面積を求め，チップ全体に対する割合を求めた．チップ全体の面積は 7.160 $mm^2$ となった．表 2 にチップ全体に対する面積の割合を示す．pchip がチップ全体のトップのモジュールとなり，その内部にコアと CSM をまとめる cluster_wrapper が存在している．そして，その内部にコアである core0 から core3 と CSM である CSM0 と CSM1 および CSM の arbiter となる CSM_bus_arbiter が存在する．各コアと各 CSM の構成はインスタンスごとに変化しないので，それぞれ同一の面積となっている．

表 2　チップ全体に対する面積の割合

| インスタンス | 割合 (%) |
|---|---|
| pchip | 100.00 |
| cluster_wrapper | 96.44 |
| core0 | 17.23 |
| CSM0 | 13.57 |
| CSM_bus_arbiter | 0.05 |

表 3 にコアに対する主要なインスタンスなどの面積の割合を示す．これらは全てコアのモジュール直下のインスタンスとして生成されたものであり階層構造を含まない．コア内部には図 1 に示したように，RISC-V ISA の CPU，VA，DTU，DSM が存在する．論理合成時に VA，VA 内部のスカラの ALU，命令フェッチユニット，およびその他のレジスタファイル周りのバウンダリが解除されているので，該当部分の面積は推定できる部分のみの総和を結果として載せている．VA の面積としてコア全体から CPU，DTU，DSM を取り除いた値を算出するとコア全体の 44.08% に相当する．pva が先頭についているインスタンスは RTL のデザインでは VA の内部に存在していたものである．pva_addsub，pva_faddsub，pva_muldiv，pva_fmuldiv はベクトル演算パイプラインであり，それぞれ整数の加算減算，浮動小数点の加算減算，整数の乗算除算，浮動小数点の乗算除算を行う．

表 3　コアに対する主要なインスタンスなどの面積の割合

| インスタンス | 割合 (%) |
|---|---|
| CPU | 15.47 |
| DTU | 6.06 |
| DSM | 34.39 |
| pva_instruction_first_cache | 3.83 |
| pva_addsub | 0.61 |
| pva_faddsub | 1.42 |
| pva_muldiv | 3.24 |
| pva_fmuldiv | 2.63 |
| pva_scalar_alu | 0.64 |
| pva_instruction_fetch | 0.15 |
| pva_vr | 22.04 |
| pva_mr | 2.16 |
| pva_temp_vr | 0.69 |

### 3.2　周波数ごとの電力と実チップとの電力の比較

周波数ごとの電力について実チップと論理合成後のデザインに対して比較を行った．本マルチコアチップではチップに 20 MHz のクロックを入力し，そこから内部の PLL により 25，50，100MHz のメモリクロックを生成する．さらに，メモリクロックを基準として，メモリクロックの 1 倍から 5 倍までの任意の整数倍のコアクロックを生成する．

シミュレーションによる電力推定値と実チップでの電力測定値を比較するため，$16 \times 1 \times 16$ の行列ベクトル積を行う 3 種類のプログラムを用いて周波数ごとの電力評価を行った．シミュレーションによる評価では，設計上動作可能な周波数範囲における電力評価を行った．実チップは電圧を 0.90V，メモリクロックを 50 MHz として，コアクロックを 2 倍から 4 倍とした条件での電力の計測を行った．図 3 に電力評価の結果を示す．

電力はコアクロックの周波数にほぼ比例する結果となった．また，50 MHz において，実チップに対して合成後のデザインの電力は最大で 6.14%，最小で 0.71% の誤差となった．このとき，合成後のデザインのリーク電力はチップ全体で 11.6mW，各コアで 2.26mW であった．

### 3.3　ベクトルアクセラレータのアイドル状態の電力

VA のアイドル状態の電力を調べるために，コア 0 を対象に VA をなるべく動かさない場合の電力を計測した．DSM アクセスによる外乱を少なくするために CPU はその場で無限ループをさせる．評価を行う VA の動作は，その場での無限ループ，

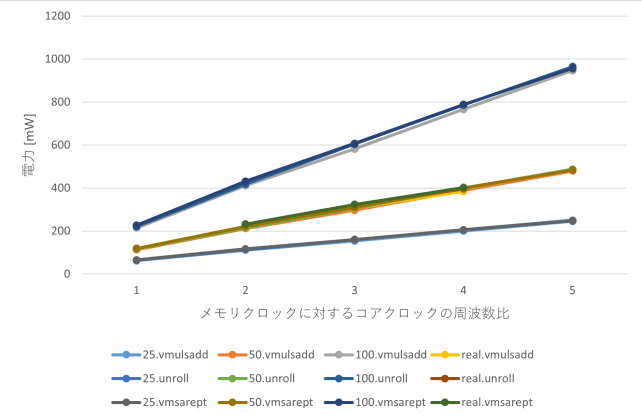


図 3　実チップとシミュレーション上での周波数比ごとの電力

DSM へのフラグ同期，命令キャッシュミス発生時の待機状態，VA が処理を終了して stop 状態で待機している状態の 4 つの評価を行った．命令キャッシュミスを発生時はデータが返ってこないシミュレーショントップを構築し，パイプラインやメモリアクセスのための回路の動作が完全に止まっている状態の電力を計測した．結果を表 4 に示す．電力が最も低くなったのは命令キャッシュミスを発生させて回路を動作させないようにした場合となった．

表 4　CPU をその場で無限ループさせた状態での VA の動作ごとの電力 [mW]

| VA の動作 | 動的電力 | リーク電力 | 総消費電力 |
|---|---|---|---|
| その場での無限ループ | 134.60 | 2.26 | 136.87 |
| DSM へのフラグ同期 | 134.05 | 2.26 | 136.32 |
| 命令キャッシュミス時 | 124.26 | 2.26 | 126.52 |
| VA の stop 状態 | 126.12 | 2.26 | 128.38 |

### 3.4　演算器に入力するデータの性質による電力

演算器に利用するデータにおいて，前後のデータ間のハミング距離によって電力は大きく変化することが知られている [12]．ベクトル加算・減算パイプラインを対象に演算器に入力するデータを変更することで，電力がどの程度変化するか確認した．ベクトル長を 128 とし，FP32 と int32 のベクトル加算命令を実行した時の演算器にデータが流れている状態の電力を計測した．評価に用いたデータとしては，ベクトルレジスタを全て 0 にした場合と全て 1 にした場合と全ビットを 0 と 1 を交互にフリップさせたものおよび乱数を使用した．乱数については，FP32 では $\pm 10^6$ の範囲の一様乱数，int32 では完全ランダムな乱数セットをそれぞれ 10 種類用意して評価を行った．表 5 にデータの種類ごとの電力を示し，表 6 に乱数を与えた時の電力について示す．

表 5 より，全て 0 の場合ではデータ型によって電力が変化することはなく，全て 1 の場合でも電力はほとんど変化しなかったことがわかる．一方で，ビットをフリップさせた場合では，int32 の演算では整数型の加算減算パイプラインの電力が約 4mW，FP32 の演算では浮動小数点型の加算減算パイプラインの電力は約 9mW 上昇した．このとき，演算を行っていない演算器の電力に着目すると，演算を行っていないにも関わらず，多くの電力を消費していることがわかる．これは，本パイプラインの現在の実装では，演算を行っていない時に演算器に流れるデータを止めていないため，電力を消費してしまうからである．したがって，今後のベクトル演算器における電力の削減方針として，使用していない演算パイプラインのデータ部分を 0 にすることで電力削減が可能であることがわかる．

表 5　ベクトル長が 128 におけるデータの種類ごとのベクトル加算を行った時の動的電力 [mW]

| データの種類 | addsub | faddsub | VR | コア |
|---|---|---|---|---|
| int32 全ビット 0 | 0.33 | 2.43 | 46.85 | 154.56 |
| int32 フリップ | 4.37 | 6.79 | 47.70 | 165.07 |
| FP32 全ビット 0 | 0.33 | 2.43 | 46.85 | 154.56 |
| FP32 全ビット 1 | 0.33 | 2.43 | 46.86 | 154.56 |
| FP32 フリップ | 4.35 | 9.02 | 47.36 | 166.54 |

表 6　乱数を与えてベクトル加算を行った時の動的電力 [mW]

| データ型 | 統計量 | addsub | faddsub | VR | コア |
|---|---|---|---|---|---|
| int32 | 平均 | 2.89 | 7.07 | 47.28 | 162.80 |
| | 標準偏差 | 0.03 | 0.07 | 0.00 | 0.10 |
| FP32 | 平均 | 2.57 | 6.42 | 47.22 | 161.66 |
| | 標準偏差 | 0.02 | 0.03 | 0.00 | 0.06 |

また，乱数を用いた場合の電力について着目すると VR における電力の標準偏差は加算減算パイプラインよりも少ないことがわかる．乱数におけるハミング距離は基本的に 32 ビットに対して半分の 16 程度となるため，電力についてもフリップさせた場合と全て 0 の場合の平均を取った値が乱数を演算器に流した時の平均の値に近いことがわかる．

### 3.5　ベクトル繰り返し命令

本チップにおけるベクトル繰り返し命令による電力の削減効果を検証するために電力評価を行った．ベクトル繰り返し命令は本チップに実装された命令であり，繰り返し数を指定することで 1 命令で多数のベクトルスカラ乗算とベクトル加算を行う命令である [13]．本評価では，DSM に載る最大サイズである $128 \times 128 \times 128$ の行列積における積和演算の命令を対象に電力の内訳を解析し，命令フェッチ回数の減少による電力削減効果を検証した．

$128 \times 128 \times 128$ の行列積はベクトル繰り返し命令を使用しない場合かつアンローリングの最適化を行うことを考慮すると本アーキテクチャでは，累積回数を 30 とすると最も演算効率が良くなる．端数処理は累積回数を 8 回とする．ベクトル繰り返し命令を使用した場合としない場合の演算パイプラインの動作を合わせるために，ベクトル繰り返し命令についても繰り返し数を 30 回と 8 回の 2 つの場合の電力を評価した．

演算に用いたデータは-1 から 1 までの一様乱数とし，計測対象のベクトル繰り返し命令の前後にベクトル繰り返し命令を 1 つずつ配置して電力の計測を行った．アンローリングのプログラムについても計測対象とする行列演算の前後にベクトル加算命令とベクトル乗算をそれぞれ 1 つずつ配置した．計測結果は図 4 のとおりである．vmsarept はベクトル繰り返し命令，

unroll はアンローリングのプログラムであり，添え字は累積回数となっている．

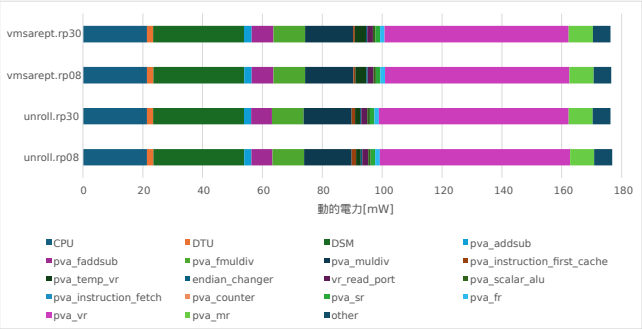


図 4　128 × 128 × 128 の積和演算命令を動作させた時の動的電力

評価結果からは，temp_vr やその他のベクトルレジスタ周りによる電力の増加により，全体の電力がベクトル繰り返し命令を使用した場合の方が高くなる場合があることがわかる．これらの問題は，ベクトルスカラ乗算とベクトル加算が 1 つの演算器で行われるような実装や，パイプラインとベクトルレジスタの挙動が完全に一致させた場合などの VR の余分な読み出しを抑えた実装とすれば回避できる．

128 × 128 × 128 と同等の演算となるように重み付けを行いエネルギーとして算出し比較したところ，命令キャッシュにおける電力削減率は FP16 で 50.23%，FP32 で 34.56% となった．また，命令キャッシュの電力はコアに対して FP16 で 0.35%，FP32 で 0.18%，VA に対して FP16 で 0.50%，FP32 で 0.27%，減少した．FP16 の方が電力削減率が大きいのは，演算サイクル数が少なく，時間あたりの命令フェッチ回数の減少率が高いためである．

## 4.　パフォーマンスカウンタを用いた電力解析

VA を構成するユニットごとの電力コストの要因を明らかにすることを目的として，コア内部を主要な構成ユニットに分割し，パフォーマンスカウンタと電力計測値を用いた線形回帰分析を行った．評価には 3 種類のベンチマークとなるプログラム群を用い，コア 0 の VA を対象に評価を行った．各構成ユニットに対してその入出力に着目した少数のパフォーマンスカウンタのみを用いて消費電力を推定し，それらを合計して得られた VA の消費電力が，評価に用いた 12 個すべてのカウンタを使用した場合と同程度の精度で電力推定が可能であることを示す．

### 4.1　ベンチマークプログラムの構成

ベンチマークプログラムとして，行列積，行列ベクトル積，畳み込み演算の 3 種類のプログラム群を用意した．行列積は，$16\times4\times16$，$32\times4\times32$，$64\times4\times64$，$128\times4\times64$，$256\times4\times64$ の 5 種類のプログラムを用意した．行列ベクトル積は，ベクトル長を $vl$，累積回数を $rp$ とした $vl\times1\times rp$ の演算を行うプログラムとし，$vl = 4, 8, 16, 32, 64, 128, 256$，$rp = 1, 4, 9, 16, 30, 31$ のプログラムを用意した．畳み込み演算のプログラムはフィルタサイズを 3 × 3，最大性能の出るブロックサイズとし，ベクトル長は行列ベクトル積と同じ長さを使用した．また，畳み込み演算においてはアンローリングファクターとして，フィルタ部分，フィルタ部分と $W$ 方向，フィルタ部分と $W$ 方向と $H$ 方向の 3 種類のアンローリングを行ったプログラムを評価した．

プログラムを動作させる上でシミュレーション時間が膨大となることを避けるために，行列積はループの先頭部分で分割を行いパフォーマンスカウンタの値が完全に一致した場合はその部分の電力は計測しないものとした．行列ベクトル積については $vl$ が 16，128，256 または $rp$ が 9，30 のいずれかの条件に当てはまるプログラムをデータとして用いた．また，行列ベクトル積と畳み込み演算は実際に演算が行われる部分を抽出して計測を行った．プログラムに用いたデータ型は FP16 と FP32 とし，アンローリングによる最適化を行ったプログラムとベクトル繰り返し命令を使用したプログラムで計測を行った．

### 4.2　構成ユニットの電力パラメータの解析

評価を行う動的電力を目的変数，カウンタの値を実行サイクル数で割った値であるデューティ比を説明変数として線形回帰を行った．評価に用いたカウンタは，ユニットごとに直接作用するものをそれぞれ選択した．本ベンチマークでは乗算回数と加算回数が必ず同一となるので，両方のカウンタが作用するユニットに対しては，代表して 1 つのカウンタを選択している．ベクトル加算・減算パイプラインは動作状態のカウンタを使用し，ベクトルスカラ乗算を行うベクトル乗算・除算パイプラインに関しては，動作状態と命令の発行回数のカウンタを使用する．レジスタファイルや命令フェッチユニットなどの読み出しと書き込みポートが複数個またはそれぞれ独立して動作するユニットに関しては，必要に応じて 2 つまたは 3 つのカウンタを用いる．ローカルセルや識別不可能なモジュールについてはランダムロジックとしてまとめ，他で使用したカウンタを全て用いて線形回帰を行う．

線形回帰の例として図 5 に浮動小数点型のベクトル加算・減算器，図 6 に浮動小数点型のベクトル乗算・除算器の結果をそれぞれ示す．GEMM は行列積，GEMV は行列ベクトル積，Conv は畳み込み演算のデータである．

浮動小数点型のベクトル加算・減算器では GEMM，GEMV，Conv における相関係数はそれぞれ 0.99，0.99，0.98 となり，残差標準偏差はそれぞれ 0.14，0.12，0.29 となった．一方で，浮動小数点型のベクトル乗算・除算器では，GEMM，GEMV，Conv における自由度補正ありの残差標準偏差はそれぞれ 0.62，0.49，0.65 となった．また，その他の構成ユニットについても同等の処理を行った結果，全てのベンチマークプログラム群において，整数型の加算減算パイプライン，命令キャッシュ，temp_vr の相関係数は 0.99 以上となった．

最後に，ユニットごとに推定した電力の整合性を評価するために，使用した 12 個全てのカウンタを用いて VA 全体の電力を 1 つの目的変数として線形回帰を行った．そして，VA 全体の電力とユニットごとに推定した電力の総和に対して残差標準偏差を算出した．表 7 に自由度補正ありの残差標準偏差と実測値の平均電力を示す．

ユニットごとに推定した電力の総和の残差標準偏差が最も大きい行列積に着目して相対残差標準偏差を計算すると VA 全体

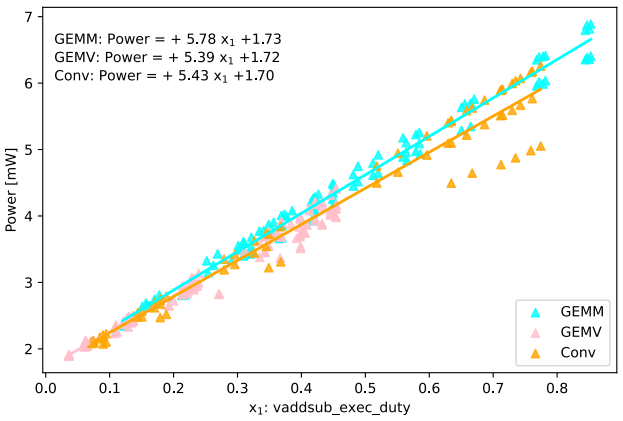


図 5　浮動小数点型のベクトル加算・減算器の電力

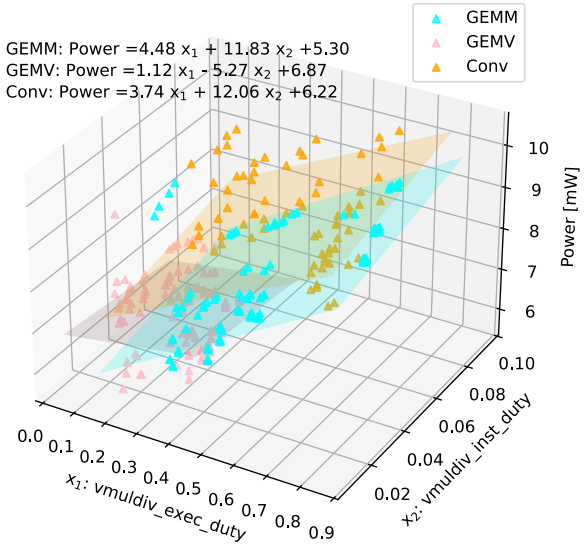


図 6　浮動小数点型のベクトル乗算・除算器の電力

表 7　VA のベンチマークごとの残差標準偏差と平均電力

| ベンチマーク | 残差標準偏差 | | 平均電力 |
|---|---|---|---|
| | VA 全体 | ユニットごと電力の総和 | [mW] |
| GEMM | 1.95 | 2.75 | 106.47 |
| GEMV | 1.79 | 2.05 | 97.46 |
| Conv | 2.03 | 2.34 | 100.73 |

は 1.83%，ユニットごとは 2.58% となり 0.75% の増加となった．行列ベクトル積と畳み込み演算も同様にして計算すると，行列ベクトル積で VA 全体が 1.84%，ユニットごとが 2.10%，畳み込み演算で VA 全体が 2.02%，ユニットごとが 2.32% となった．これらの結果から，ユニットごとに適切なパフォーマンスカウンタを用いることで，VA 全体に対する各ユニットの電力寄与が正確に推定可能であることが確認できた．

## 5. ま と め

本稿では，開発したマルチコアチップの論理合成後のデザインを対象として性能および電力評価を行った．実チップとの比較を通じて，論理合成後のデザインを用いた電力評価の有効性を示すとともに，アイドル状態，演算器に入力するデータの性質，およびベクトル命令繰り返し命令の評価により，電力特性を詳細に分析した．さらに，パフォーマンスカウンタに基づく統計的解析を行い，構成ユニットの入出力に着目した少数のカウンタから，より多くのカウンタを用いた場合と同程度の精度で電力を推定できることを示した．

**謝辞**　本研究の成果の一部は JST【ムーンショット型研究開発事業】【JPMJMS2031】の支援を受けたものです．

## 文　献

[1] S. Govinda, B. Brik, and S. Harous, “A Survey on Deep Reinforcement Learning Applications in Autonomous Systems: Applications, Open Challenges, and Future Directions,” IEEE Transactions on Intelligent Transportation Systems, vol.26, no.7, pp.11088–11113, 2025.

[2] A. Boumendil, W. Bechkit, and K. Benatchba, “On-device deep learning: Survey on techniques improving energy efficiency of dnns,” IEEE Transactions on Neural Networks and Learning Systems, vol.36, no.5, pp.7806–7821, 2025.

[3] 岡本雅巳，合田憲人，宮沢稔，本多弘樹，笠原博徳他，“OSCAR マルチグレインコンパイラにおける階層型マクロデータフロー処理手法，” 情報処理学会論文誌，vol.35，no.4，pp.513–521，1994.

[4] K. Kimura, Y. Wada, H. Nakano, T. Kodaka, J. Shirako, K. Ishizaka, and H. Kasahara, “Multigrain parallel processing on compiler cooperative chip multiprocessor,” 9th Annual Workshop on Interaction between Compilers and Computer Architectures (INTERACT’05), pp.11–20, 2005.

[5] F. Onishi, R. Otaka, K. Fujita, T. Suetsugu, T. Kawasumi, T. Kitamura, H. Kasahara, and K. Kimura, “Automatic Deep Learning Parallelization for Vector Multicore Chips with the OSCAR Parallelizing and the TVM Open-Source Deep Learning Compiler,” Proceedings of The 36th International Workshop on Languages and Compilers for Parallel Computing (LCPC 2023), pp.96–110, Lexington, Kentucky, USA, Oct. 2023.

[6] Taiwan Semiconductor Manufacturing Company, “Logic Technology,” `https://www.tsmc.com/english/dedicatedFoundry/technology/logic/`. Accessed: 2026-01.

[7] 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,” 2009 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), pp.469–480, 2009.

[8] 末次智貴，野谷優仁，水本幸希，大西文彬，権藤創太，朱允楷，川角冬馬，北村俊明，笠原博徳，木村啓二，“プログラム実行機構を持つ OSCAR ベクトルマルチコア用データ転送ユニット，” 研究報告システム・アーキテクチャ（ARC），vol.252，no.33，pp.1–8，Mar. 2025.

[9] M. Awaga and H. Takahashi, “The μ VP 64-Bit Vector Coprocessor: A New Implementation of High-Performance Numerical Computation,” IEEE Micro, vol.13, no.5, p.24–36, Sept. 1993. https://doi.org/10.1109/40.237999

[10] Synopsys, Inc., “Design Compiler,” `https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/design-compiler.html`. Accessed: 2026-01.

[11] Cadence Design Systems, Inc., “Xcelium Logic Simulator,” `https://www.cadence.com/en_US/home/tools/system-design-and-verification/simulation-and-testbench-verification/xcelium-simulator.html`. Accessed: 2026-01.

[12] J. Lucas and B. Juurlink, “ALUPower: Data Dependent Power Consumption in GPUs,” 2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), pp.95–104, 2016.

[13] 野谷優仁，末次智貴，水本幸希，大西文彬，朱允楷，権藤創太，川角冬馬，北村俊明，笠原博徳，木村啓二，“OSCAR ベクトルマルチコアへのベクトル繰り返し命令拡張，” 研究報告システム・アーキテクチャ（ARC），vol.252，no.36，pp.1–7，Mar. 2025.