暗号処理向け粗粒度再構成可能アーキテクチャの評価

小島　拓也*, 伊藤　向子†

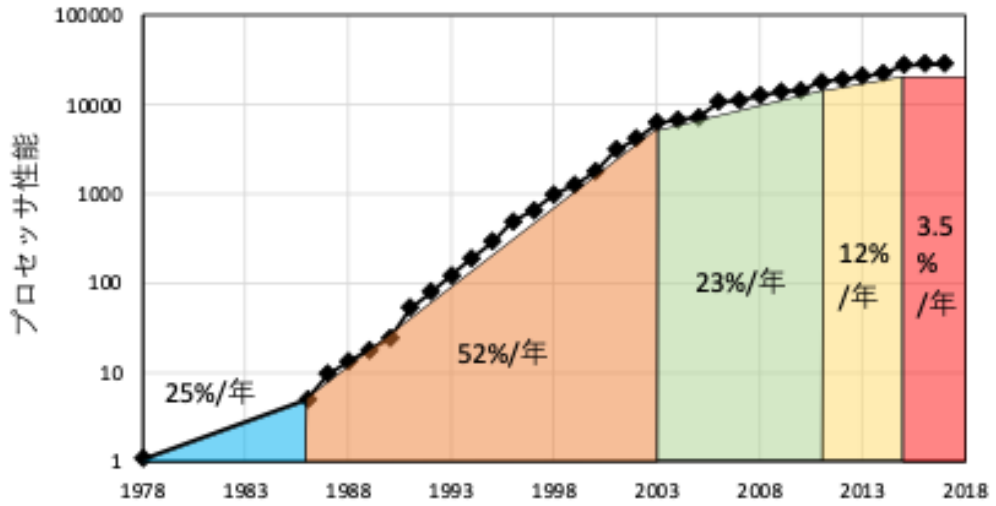
*筑波大学, †東京大学

01 研究背景

領域特化アーキテクチャと
粗粒度再構成可能アーキテクチャ

計算機システムにおける新たな潮流

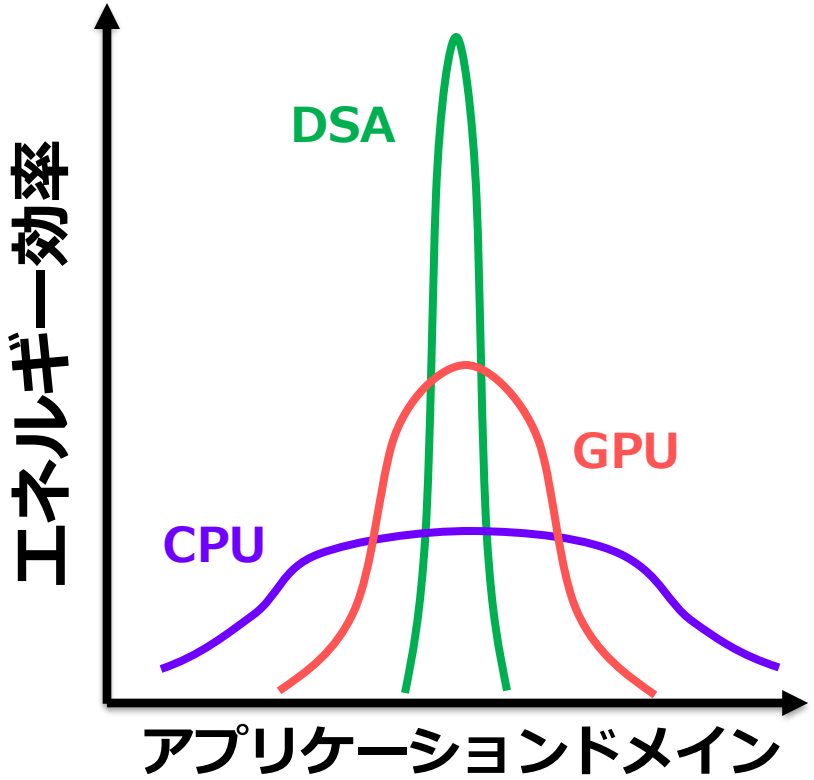
- エネルギー効率の改善が急務
  - データセンタの消費電力は10年で6倍に増加
  - 生成AIに要する電力は全世界で原発2基分の年間発電量を要するとの試算†
- **領域特化アーキテクチャ**の必要性
  (Domain-Specific Architecture, DSA)



今後プロセッサの性能向上では劇的な改善は見込めず‡

DSAの導入

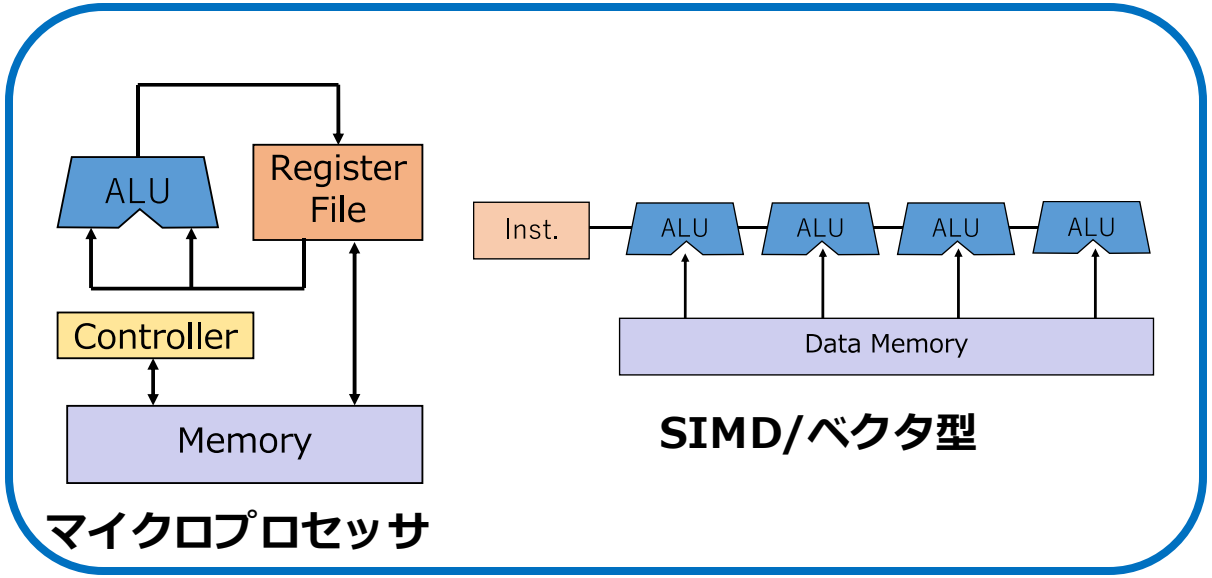
処理可能なアプリケーションドメインを狭くする見返りに電力効率を改善



†MATTHEW S. SMITH. The Hidden Behemoth Behind Every AI Answer, IEEE Spectrum, Oct 2025.
‡Hennessy, John L., and David A. Patterson. *Communications of the ACM* 62.2 (2019): 48-60.

領域特化アーキテクチャの例

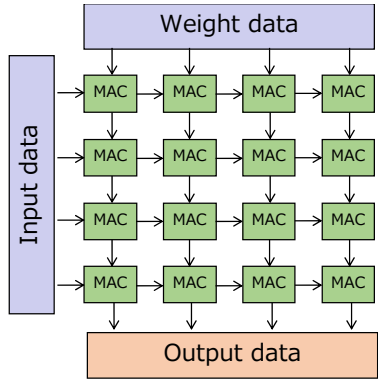
- 汎用のCPUに代わり特定のアプリケーション領域(e.g., AI,マルチメディア)の計算を効率的に処理できる計算機



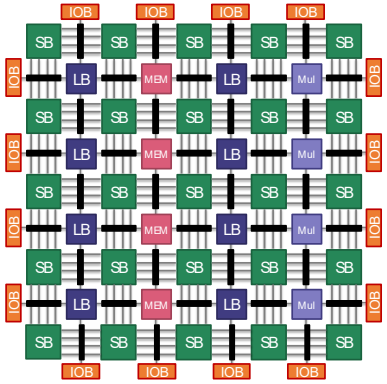
従来のノイマン型計算機

☹ メモリアクセスが性能のボトルネック

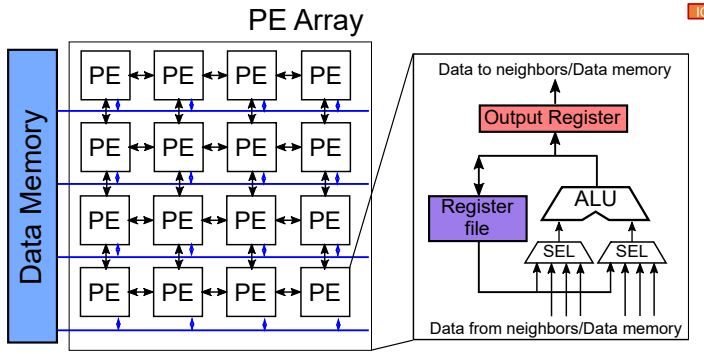
領域・用途に特化



シストリックアレイによる行列積演算



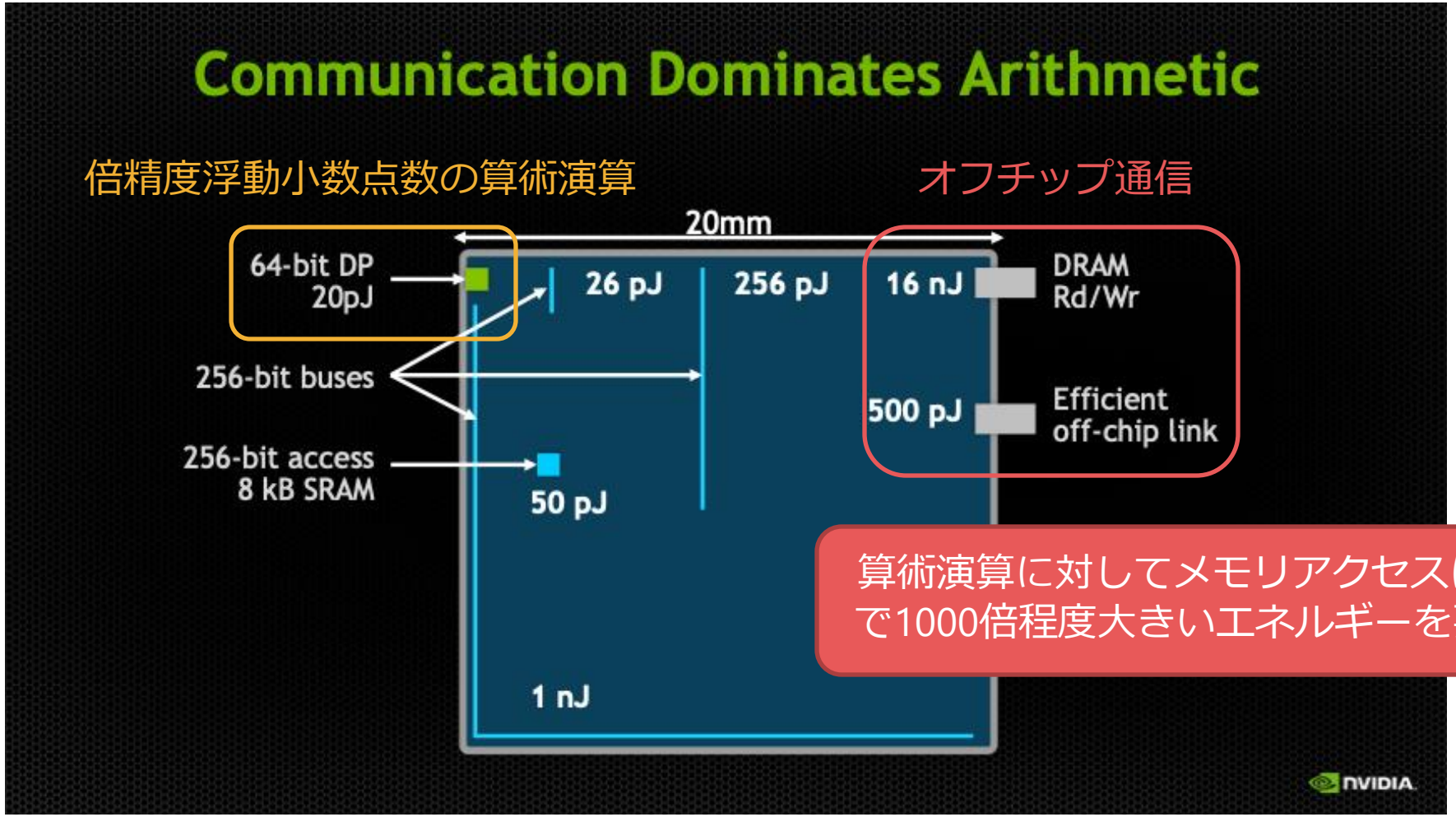
FPGAによる専用HW化



CGRAによるデータフロー計算

☺ 処理性能、エネルギー効率を改善

肝はデータ移動の最小化・効率化



算術演算に対してメモリアクセスは最大で1000倍程度大きいエネルギーを要する

Source: Bill Dally, Challenges for Future Computing Systems, HiPEAC 2015.

CGRA: Coarse-grained reconfigurable architectures

粗粒度再構成可能アーキテクチャ

- FPGAと比較して再構成の粒度が大きい (e.g., 32-bit)

特徴、比較

CGRA

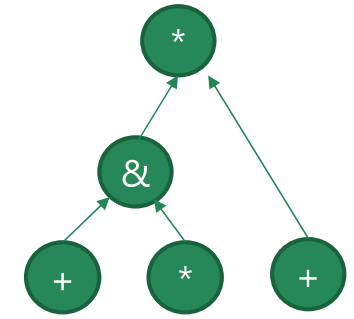
- ワード単位の再構成
- 省面積、省電力、低遅延
- ns-usオーダーで切り替え

電力効率
77倍改善*

FPGA

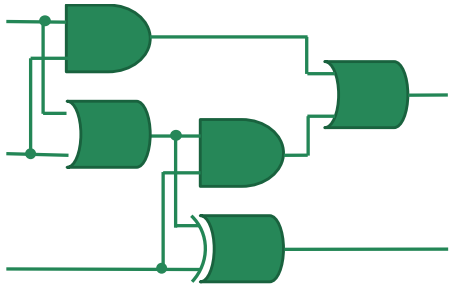
- ビット単位の再構成
- 高いプログラマビリティ
- ms-sオーダーで切り替え

アプリケーション表現



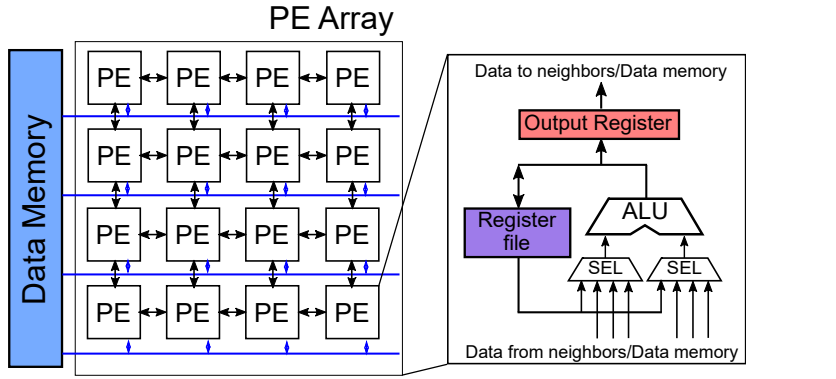
データフローグラフ

マッピング

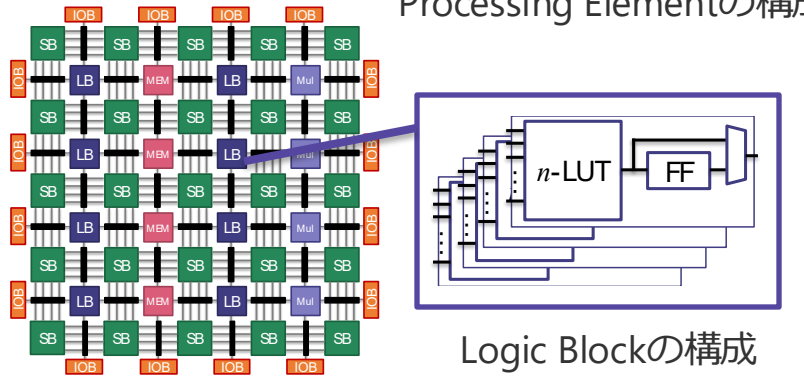


ゲートレベルネットリスト

ハードウェア構成



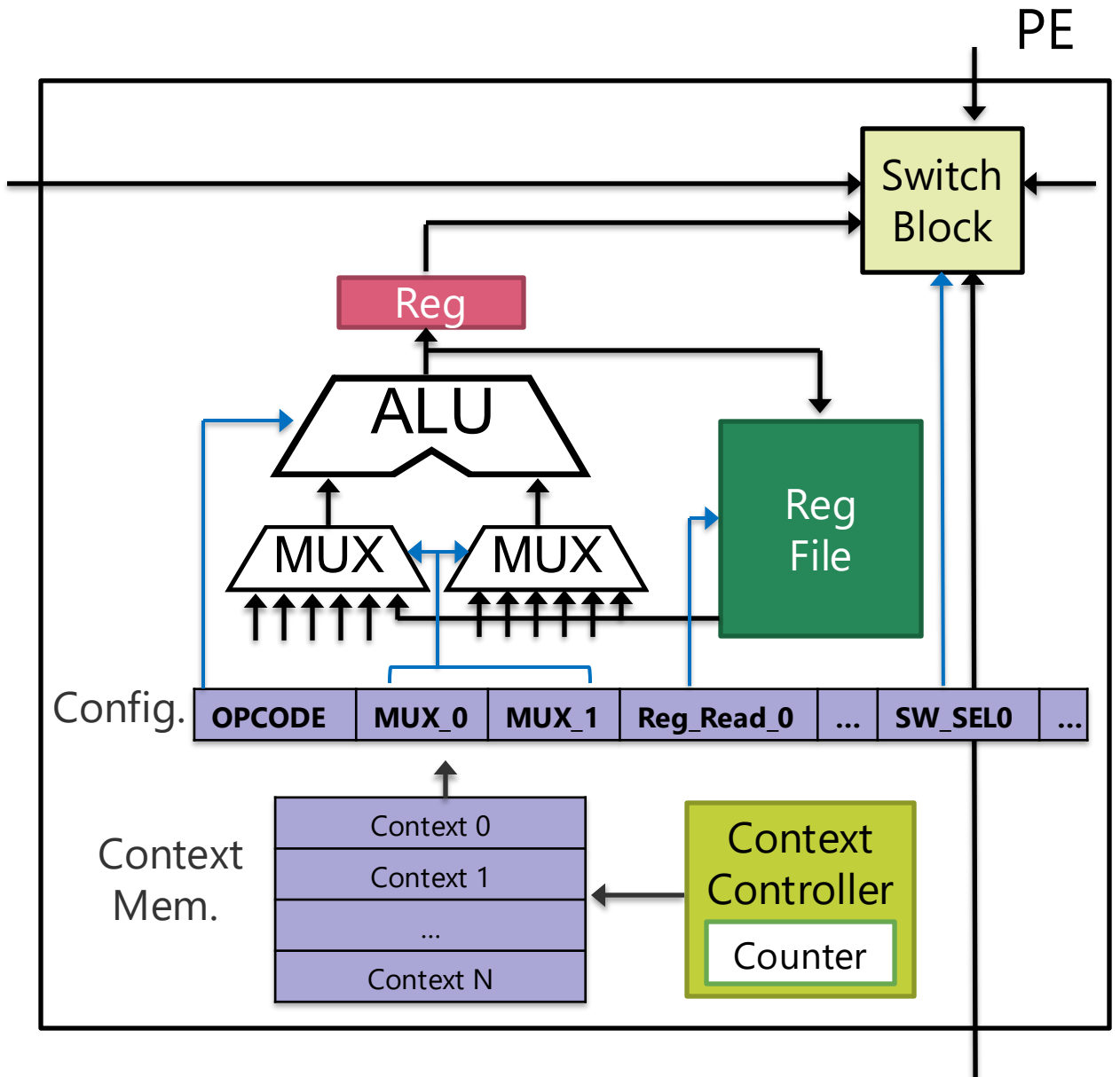
Processing Elementの構成



Logic Blockの構成

*Prabhakar, Raghu, et al. ISCA 2017

一般的なPEの構成とプログラマビリティ



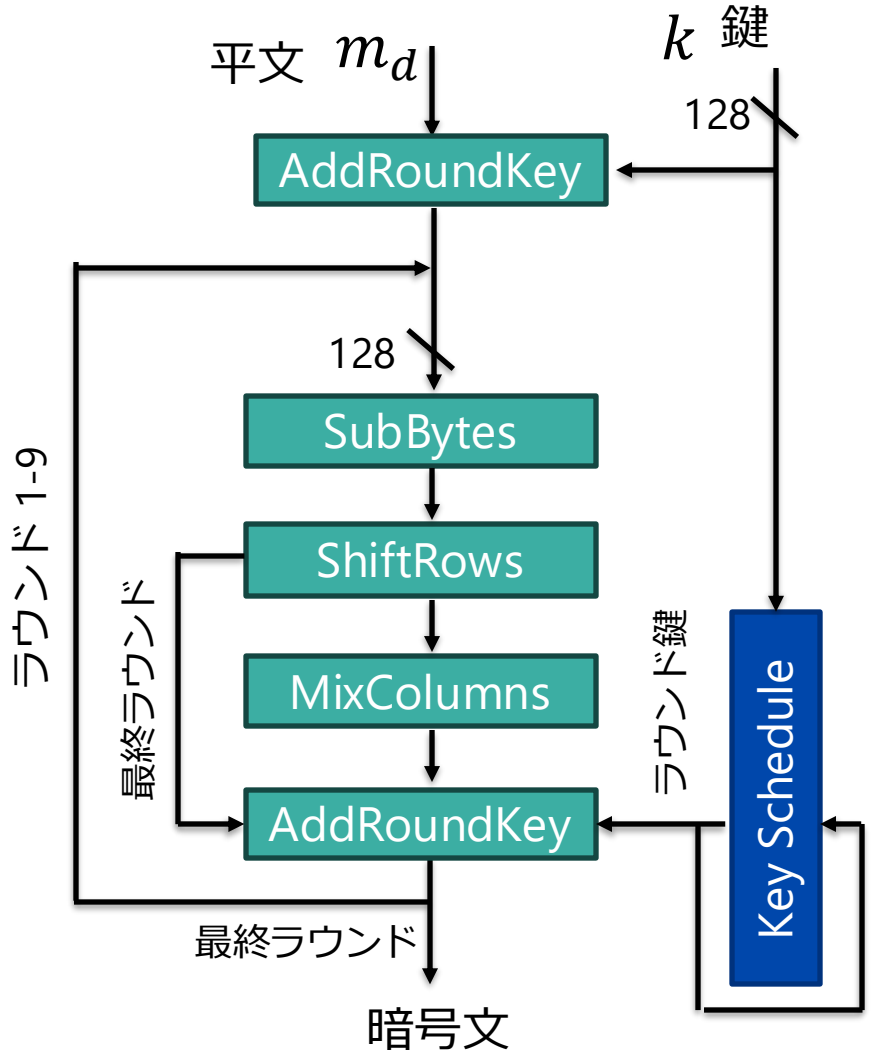
- コンフィギュレーションデータ
  - プロセッサの機械語に似たいくつかのフィールドで構成
  - 各構成モジュールの振る舞いを指定
- 主な再構成可能モジュール
  - ALU
  - マルチプレクサ
  - レジスタファイル or FIFOバッファ
  - 接続網用スイッチ
  - etc.

02 AESの概略と課題

ケーススタディとしてのAES

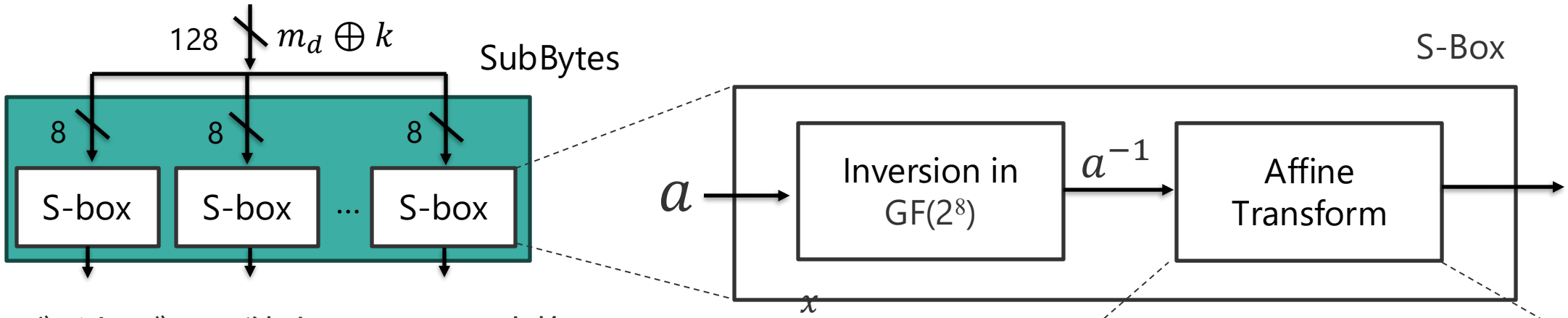
CGRAが直面する課題

AESにおける暗号化の流れ



- Advanced Encryption Standard (AES)
  - 共通鍵暗号
  - ブロック暗号
    - 鍵長: 128bit, 192bit, 256bit
- 共通の処理(ラウンド)を複数回繰り返す
  - 128bit長の場合は10回
- 各ラウンドは4つの処理で構成
  - **SubBytes: S-boxによる非線形変換**
  - ShiftRows : 行シフト
  - MixColumns: 行列変換
  - AddRoundKey: ラウンド鍵とのXOR

非線形変換S-Box



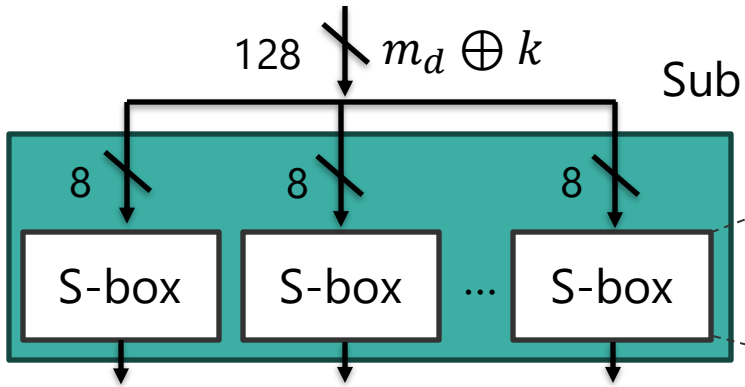
各バイドごとに独立にS-Boxで変換

- ガロア体(GF(2^8))上での演算で定義
  - 既約多項式 $x^8 + x^4 + x^3 + x + 1$ (0x11B)

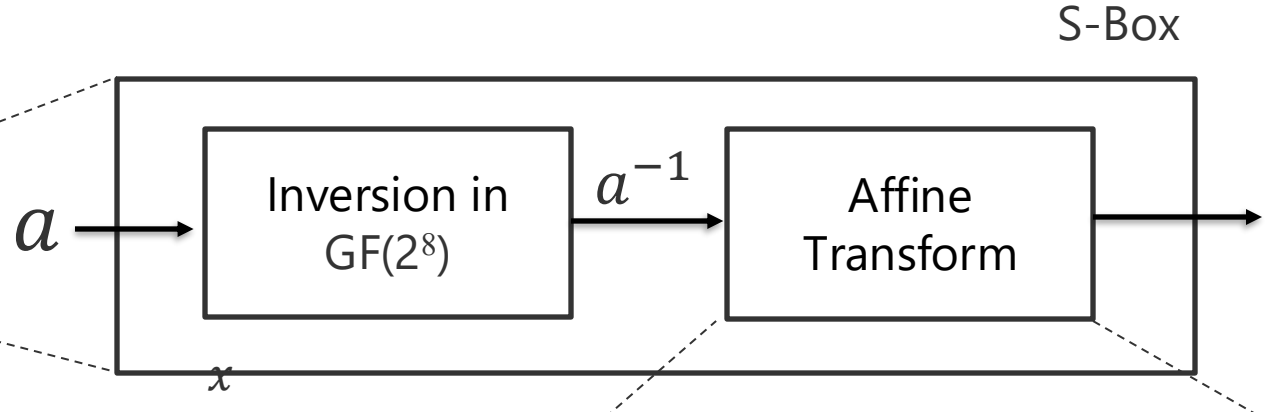
$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

- Step1: 逆元計算
- Step2: アフィン変換
  - GF(2)上の定数行列計算 (ビット方向の加算あり)
  - Rotate shiftによる別表現あり

非線形変換S-Box



各バイドごとに独立にS-Boxで変換



- ガロア体(GF(2^8))上での演算で定義
  - 既約多項式 $x^8 + x^4 + x^3 + x + 1$ (0x11B)

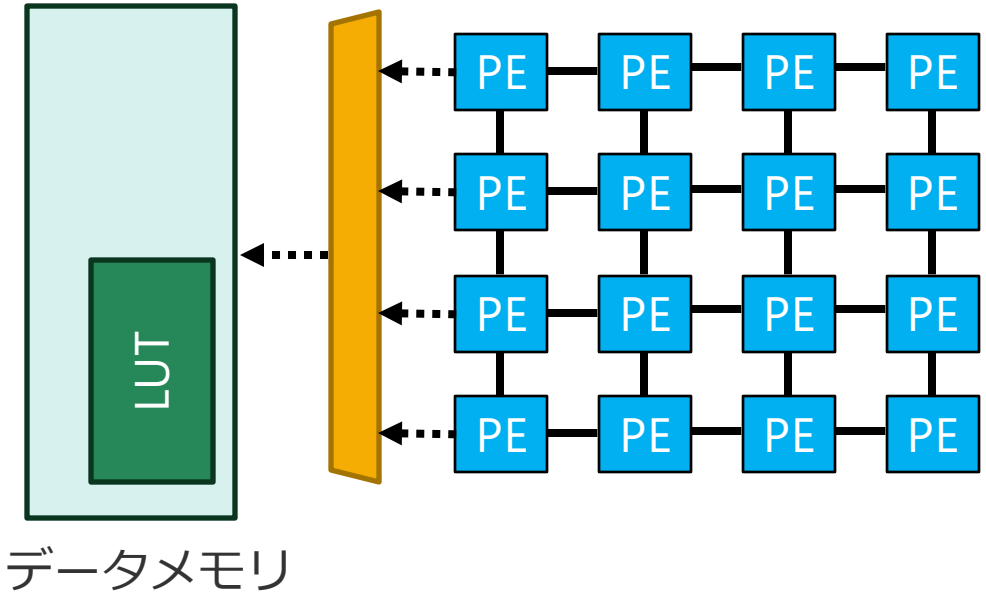
$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ \vdots \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ \vdots & & & & & & & \end{pmatrix} \times \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

- Step1: 逆元計算
- Step2: アフィン変換
  - GF(2)上の定数行列計算 (ビット方向の加算あり)
  - Rotate shiftによる別表現あり

通常の整数演算ALUでは計算が困難

CGRAによるAESの既存実装

- S-Boxを実現する手段としてLook-Up-Tableを使用
  [C.Wang, et. al., IEICE 2017]



データメモリ

一般に一部のPEのみ(e.g., 左端)が
データメモリにアクセス可能

☺ 数クロックサイクルで複雑な演算を実行

☺ 追加のロジックを必要としない

☹ メモリアクセス可能なPEの制約により
並列度に限界

☹ メモリ読み出し時の高いエネルギー

**メモリアクセスの最小化を目指す
データフロー計算と逆行**

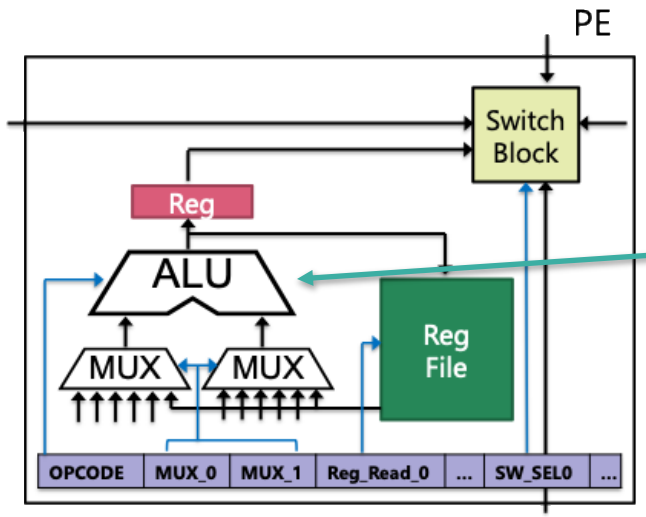
03 ドメイン特化CGRAの提案

暗号処理向けに機能拡張したCGRAを提案

ドメイン特化CGRAによる解決

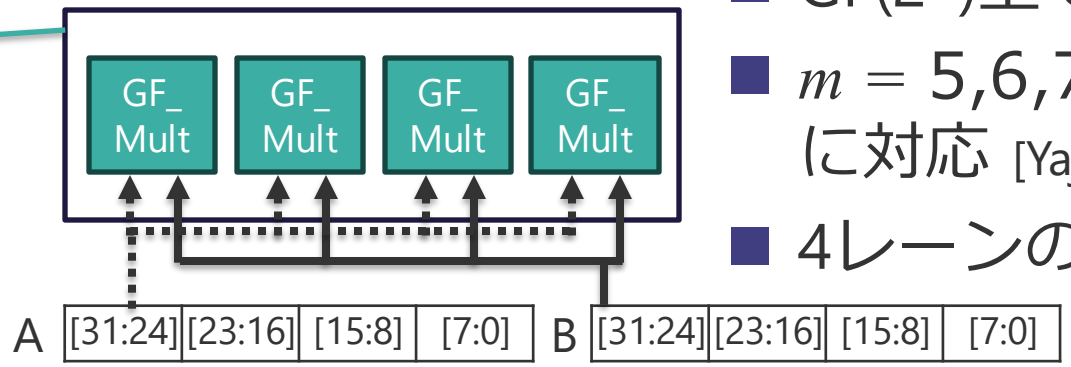
- 一般的なCGRAの演算器(ALU)は算術演算，シフト演算，論理演算などをサポート
- ドメインに特化した演算を追加して高効率化を図るアプローチ
  - ML向け活性化関数 [Yixuan Luo, et al, DAC 2023], 生体信号処理向け近似演算器 [Zahra Ebrahimi, et al, ISCAS 2021]

本研究ではガロア体上の乗算，平方を計算する演算器を追加し，その効果を検証



一般的なPEの構成

機能追加

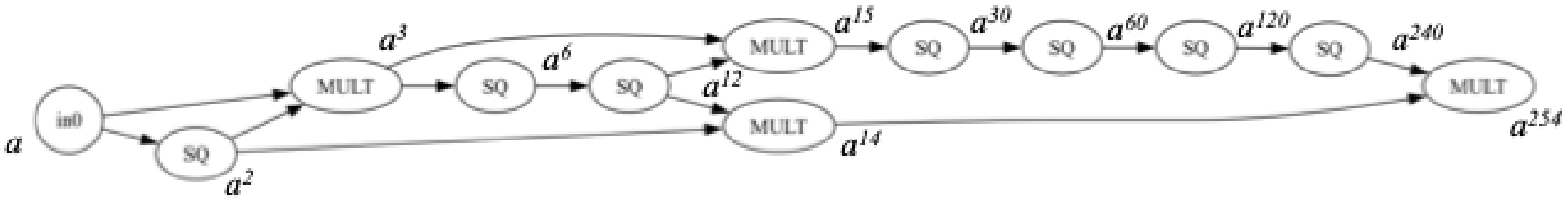


4レーンSIMDのGF乗算器

- GF(2^m)上での乗算可能な演算器
- m = 5,6,7,8と任意の既約多項式に対応 [Yajing Chen, ISCA 2017]
- 4レーンのSIMD

議論: 高機能PE vs 単純化PE

- 最適化された逆元の計算では乗算4回，平方7回が必要
  - 逆元a^{-1} は a^{254}と等価



Ito-Tsujiiアルゴリズムによる逆元計算のデータフロー

単純化PEのケース

- 1つのPEで1回の乗算が可能

☺ 機能追加による面積コストの増加は軽微

☹ 少なくとも10サイクルのレイテンシを要する
PEアレイ上への配置配線が複雑化

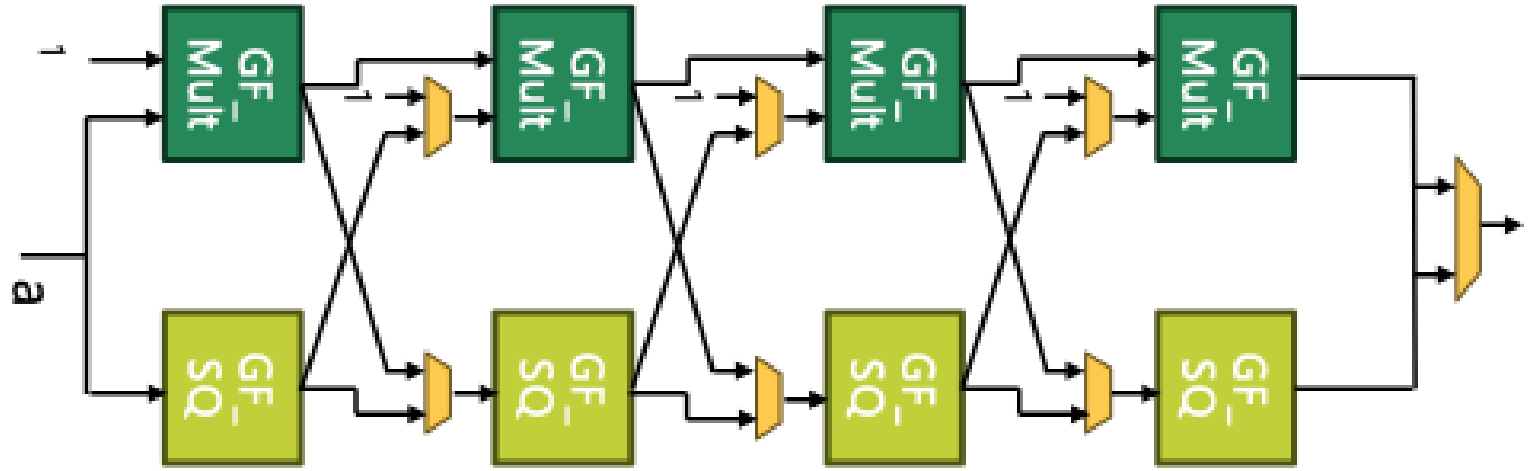
高機化PEのケース

- 1つのPEで1度に逆元を計算可能

☺ データフローが簡略化

☹ 動作周波数の低下, 演算器の利用効率が低下

提案: 4ステージ構成のべき乗演算器



提案するGF-Power演算器

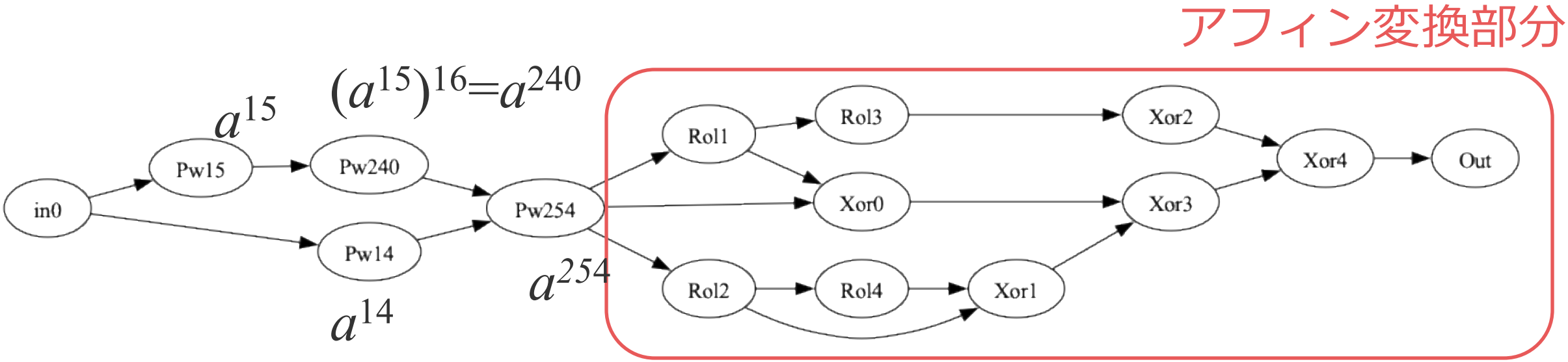
- 遅延時間の予備評価を実施

  (32ビット整数乗算器の遅延) ≒ (GF乗算器の遅延 x 4)

→ステージ数を4段に決定

- 2乗から16乗を計算可能
- これをSIMDの1レーンとして4レーンを1つのPEで計算

S-Box計算のためのデータフロー



- 1ノードは1PEに割り付けされる単位
- 提案演算器を用いると逆元計算は4ノードまで削減
- アフィン変換はRotate ShiftとXORで計算
  - この部分の縮小，効率化は今後検討

04 評価

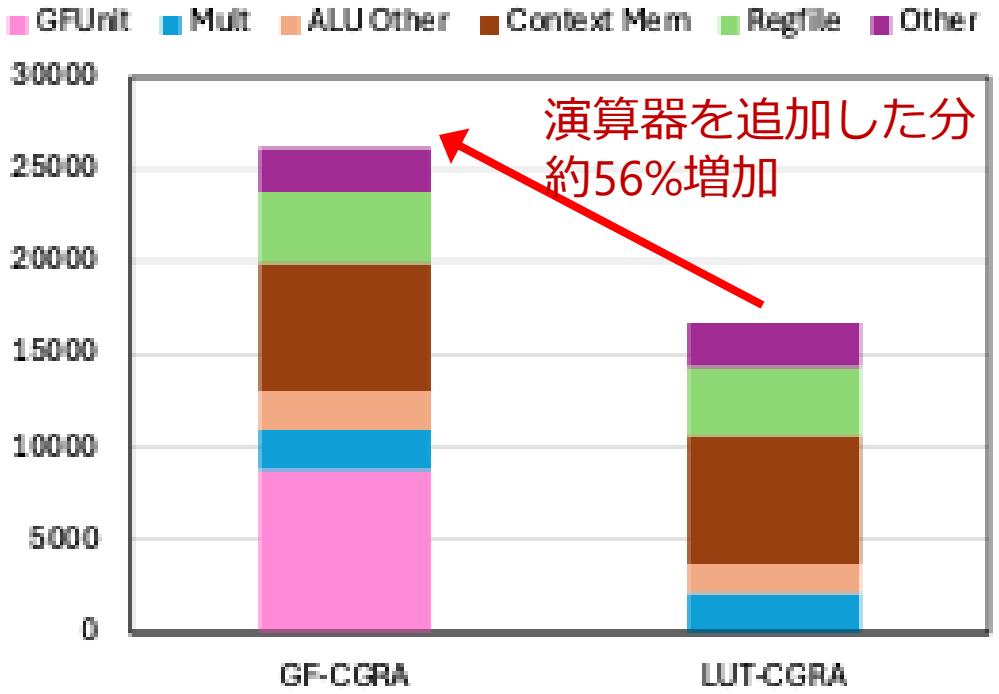
ハードウェアコストと性能改善を議論

評価条件と評価目的

- 目的: 演算器追加による面積増加に対して，性能改善の利得が上回るのかどうかを検証
- 評価条件
  - PEアレイサイズ: 4x4で固定
  - メモリアクセス: 各行で左端のPEのみ可能
  - スタセルライブラリ: NANGATE45
  - データメモリ
    - PE行あたり1KBのSRAMバンク
    - SRAMマクロはOpenRAMで生成 (デュアルポート構成)
  - 論理合成: Synopsys Design Compiler 2021.06
- 比較対象
  - GF演算器を持たないCGRAでLUT(メモリアクセス)によってS-Boxを計算する方式

HWオーバーヘッドの評価

面積比較



PEの面積内訳

- データメモリも含めたCGRAコア全体で見ると33.7%の増加

遅延時間比較

合成可能な周波数 (MHz)

	GF-CGRA	LUT-CGRA
PE単位	240	250
CGRA単位	230	230

- PE単位で見ると若干遅延が大きくなるが，CGRAコア全体では変化なし

性能改善の評価

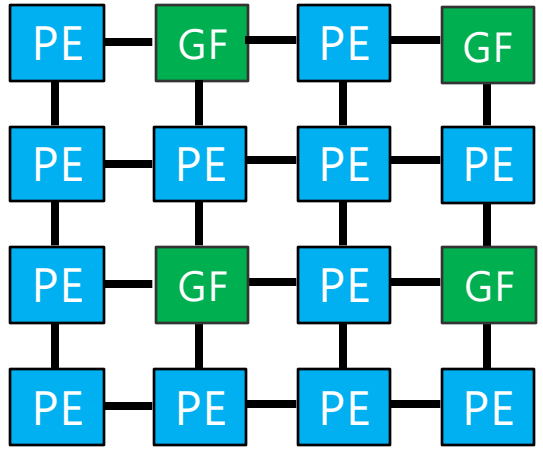
- CGRAは一般に，ループをソフトウェアパイプラインで処理
  →性能指標としてInitiation Interval (II)が重要
- MII: リソースなど構造上の制約から決まる下限値
- 現状のマッピングアルゴリズムでスループットが**1.66倍向上**

データフローグラフの特性とマッピング結果

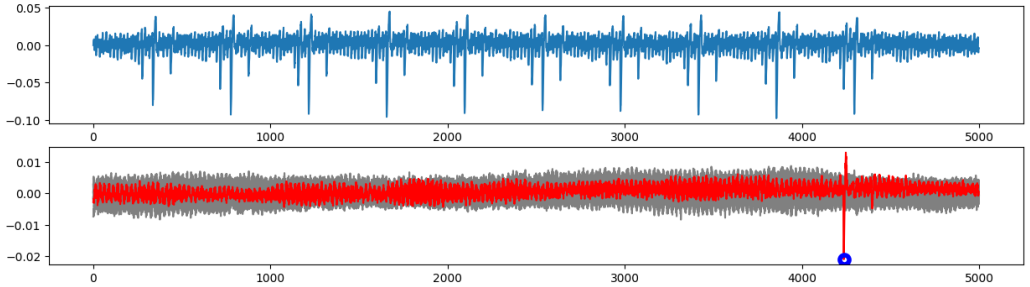
	GF-CGRA	LUT-CGRA
総ノード数	15	15
メモリアクセス数	2	6
MII	2	3
実際に得られたII	6	10

今後の方針

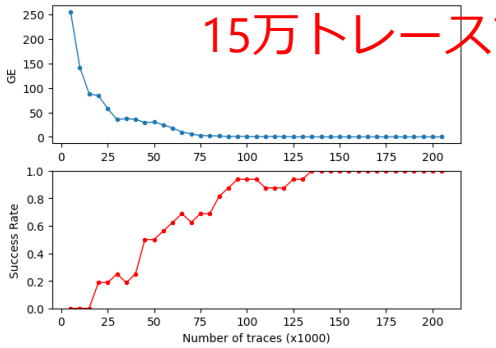
- 今回評価した設計は全PEに等しくGF演算器を追加した極端な設計
  - ヘテロジニアス構成による面積増加を最小化
- マッピングアルゴリズムの改良によってII改善の余地がある
- 電力評価による消費エネルギーを議論
- プログラマビリティを生かしたサイドチャネル攻撃耐性のあるデータフロー&マッピング



ヘテロジアス構成の例



取得波形と相関係数(赤が正解鍵の相関係数) (CW305より取得)



Guessing Entropy(正解鍵の予想ランク, 0が正しい推定)とSuccess Rate (正しく推定された鍵の割合, 全16バイト)