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

信学技報
TECHNICAL REPORT OF IEICE.

動的リコンフィギャラブルプロセッサにおける時分割多重実行の性能と消費電力の解析

長谷川揚平† 天野 英晴† 阿部 昌平† 黒瀧 俊輔† ヴ マン トウアン†

† 慶應義塾大学大学院理工学研究科 〒 223–8522 神奈川県横浜市港北区日吉 3-14-1
E-mail: †drp@am.ics.keio.ac.jp

あらまし 本論文では，動的リコンフィギャラブルプロセッサにおける時分割多重実行による性能と消費電力の関係を議論する．要求性能に対して必要エネルギー量を最適化するコストを明らかにするため，動的再構成機能による時分割多重実行の効果を定量的に評価する．実際に，NEC エレクトロニクス社の Dynamically Reconfigurable Processor (DRP) とその開発環境を用いて，いくつかのアプリケーションを様々な構成を想定して実装し，評価を行った．本論文では，実装した各アプリケーションにおいて，コンテキストサイズに対する性能および消費電力の関係を解析した．
キーワード 動的リコンフィギャラブルプロセッサ，DRP, マルチコンテキスト機能，時分割多重実行，消費電力

Performance and Power Analysis of Time-multiplexed Execution on Dynamically Reconfigurable Processor

Yohei HASEGAWA†, Hideharu AMANO†, Shohei ABE†, Shunsuke KUROTAKI†,
and Vu MANH TUAN†

† Department of Information and Computer Science, Keio University
3-14-1, Hiyoshi, Kohokuku, Yokohama, 223–8522, Japan
E-mail: †drp@am.ics.keio.ac.jp

Abstract Dynamically Reconfigurable Processor (DRP) developed by NEC Electronics is a coarse grain reconfigurable processor that selects a data path from the on-chip repository of sixteen circuit configurations, or contexts at run-time. The time-multiplexed execution based on the multi-context functionality is expected to improves area and power efficiency. To demonstrate the impact of time-multiplexed execution, we have implemented several stream applications on DRP-1 with various context sizes. Throughout the evaluation based on real application desings, we analyzed the impact of the time-multiplexed execution to performance and power consumption quantitatively.
Key words Dynamically Reconfigurable Processor, DRP, Multicontext Functionality, Time-multiplexed Execution, Power Consumption

1. ま え が き

動的リコンフィギャラブルプロセッサは，粗粒度のプロセッシングエレメント (PE) をアレイ上に配置した構成をとり，これらの機能と接続を動作中に変更することで，高い面積効率を実現するプログラマブルデバイスである [1]．各 PE は ALU やレジスタから構成され，アレイ内外の分散メモリ中のデータを同時アクセスして並列処理することにより，特にマルチメディア処理や通信分野において高い処理能力を示す．さらに，LSI の高位合成技術を応用することで，開発期間の短縮が可能であり，設計変更にも容易に対応可能である．これらの利点から，近年では複数のメーカより商用のデバイスが登場している [2]〜[5].

これらの動的リコンフィギャラブルプロセッサの最大の特徴は，動的再構成機能による時分割多重実行である．特にマルチコンテキスト型動的リコンフィギャラブルプロセッサは，複数のコンテキストと呼ばれるコンフィギュレーションデータをチップ内部に保持し，これを 1 クロックで切り替えることが可能である．対象のアプリケーションの一連の処理を複数のコンテキストに分割し，これを動作時に切り替えながら実行する．このマルチコンテキスト機能に基づく時分割多重実行により，Field Programmable Gate Array (FPGA) を始めとする従来のプログラマブルデバイスの課題であった面積効率を改善する．

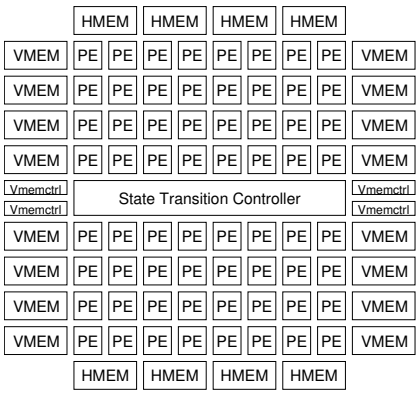


図 1　Tile の構造

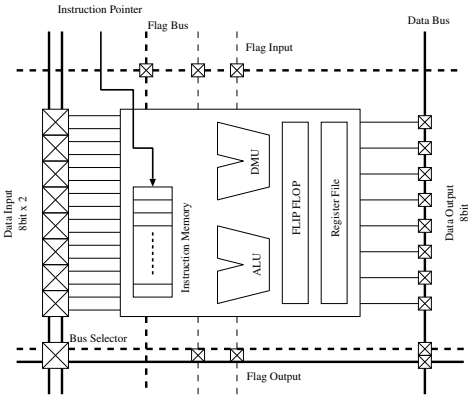


図 2　Processing Element (PE) の構造

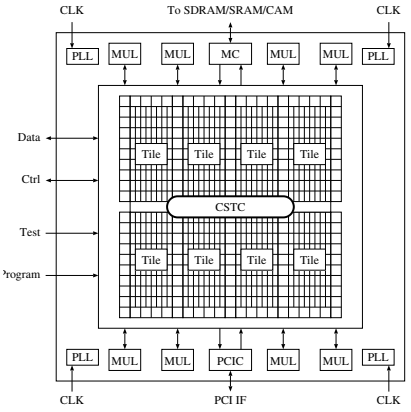


図 3　DRP-1 の構造

一方で，動的リコンフィギャラブルプロセッサは，System-on-a-Chip (SoC) において Intellectual Property (IP) コアとして利用されることが想定されている．近年の半導体産業では，携帯電話に代表されるモバイル端末の普及に伴ない，LSI における消費電力の削減が最重要課題となっている．このため，DRP の場合，対象とする問題の要求性能に応じて，消費電力の面で最も効率のよい PE 数とコンテキスト数を見出すことが重要である．

本論文では，実際の動的リコンフィギャラブルプロセッサである NEC エレクトロニクス社の Dynamically Reconfigurable Processor (DRP) を対象として，時分割多重実行による性能と消費電力の関係を議論する．そこで，DRP のプロトタイプチップ DRP-1 とその開発環境を用いて，いくつかのアプリケーションを，様々な構成を想定して実装し，評価を行った．我々はすでに面積効率に関して評価を行ってきた [6], [7] が，本報告では消費電力に焦点をあてて議論する．

本論文の構成は以下の通りである．節 2. では本論文で対象とする DRP のアーキテクチャについて述べる．続いて，節 3. において，DRP における時分割多重実行モデルを述べる．そして，節 4. で DRP-1 上に実装したアプリケーションに基づいた評価結果を述べ，節 5. で評価結果に関する考察と結論を述べる．

2. DRP アーキテクチャ

DRP は，NEC エレクトロニクス社より 2002 年に発表された動的リコンフィギャラブルプロセッサである [2]．DRP の基本ユニットである Tile のアーキテクチャを図 1 に示す．DRP は，Tile はアレイ状に配置することで，所望の規模の DRP コアを実現する．各 Tile は，8-bit の演算器とレジスタファイルなどから構成される Processing Element (PE) を 8× 8 の 2 次元アレイ状に並べ，その周辺に VMEM，HMEM と呼ばれる分散メモリを縦横に拡散配置し，さらに中央部に State Transition Controller (STC) を配置した構成となっている．Tile ごとに STC と PE のアレイが組み合わされているため，各 Tile が独自のステートマシンの制御下で独立して動作することが可能である．

各 PE の構成を図 2 に示す．各 PE は 8-bit の ALU，シフトやデータ制御，簡単な論理演算を行なう Data Manipulation Unit (DMU)，8-bit の Flip Flop，レジスタファイルから構成される．また，コンフィギュレーション時には命令データが命令メモリに書き込まれ，実行中に STC が発行した命令ポインタを命令メモリが受け，利用する命令データをロードすることでハードウェア構成を変更する．

DRP のプロトタイプチップである DRP-1 の構成を図 3 に示す．DRP-1 はコアの周辺部に 32-bit の乗算器を 8 セット，メモリモジュール，PCI バスインタフェース，SDRAM/SRAM/CAM インタフェースを搭載したシステム LSI である．また，PCI バスインタフェース，SDRAM/SRAM/CAM インタフェースにより単独で PCI バスへの接続や外部メモリの制御が可能である．

DRP-1 では 4 × 2 個の Tile が配置されており，全体では 512 個の PE をもつ．チップ中央には，DRP-1 全体の状態遷移を制御するためのシーケンサである Centeral State Transition Controller (CSTC) が配置されている．また，DRP-1 全体では VMEM を合計 80 セット，HMEM を合計 32 セットもつ．

DRP-1 は内部のコンフィギュレーションメモリに最大 16 コンテキスト分の情報を蓄えることが可能で，クロックサイクル毎にコンテキスト切り替えが可能なである．

DRP の開発環境として，統合開発環境 Musketeer が提供されている．対象のアプリケーションをソフトウェアライクに開発することができるため，短 TAT を実現することができる [8]. DRP コンパイラは，C 言語をハードウェア記述用に拡張した Behavior Design Language (BDL) から，動作合成，テクノロジマッピング，配置配線などの合成を経て，最終的に DRP 上で実行可能なコンフィギュレーションデータを生成する．

DRP-1 はプロトタイプであるため，8 Tile 構成のみで単独動作するが，DRP の本来の利用形態は，組み込み用プロセッサと複数の Tile から構成される DRP コアを接続したシステムである．この場合，アプリケーションに応じて最適な Tile とコンテキストをもった DRP コアを用意する必要がある．本研究の目的は，対象とする問題において，必要な性能を最も効率良く実現する DRP コアの構成を見いだす方法論を確立することである．

3. DRP における時分割多重実行

3.1 問題の並列性と逐次性

DRP の対象とするアプリケーションは，SoC におけるマルチ

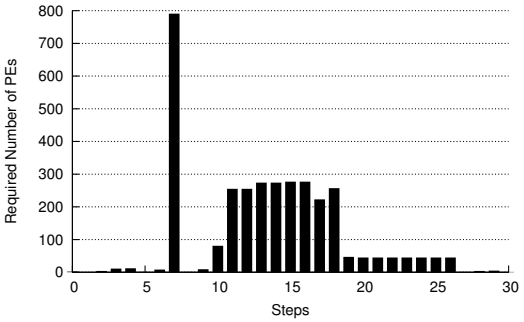


図 4　ステップ毎の必要 PE 数 (DCT)

メディアデータを扱うストリーム処理である．JPEG や MPEG に代表されるストリーム処理では，そのアルゴリズムに潜在的に逐次性と並列性が存在する．動的リコンフィギャラブルプロセッサでストリーム処理を実行する場合，以下の典型的な処理フローが考えられる．

(1)　入力データを分散メモリ（またはレジスタ）に格納する
(2)　分散メモリ（またはレジスタ）の同時アクセスによりデータを読み出し，多数の PE アレイにより並列処理を行う
(3)　演算結果を分散メモリ（またはレジスタ）に書き戻す

これらの処理をコンテキストを切り替えながら反復して実行し，最終的な結果を出力データストリームとしてチップ外部に出力する．本論文では，上記の一連の 3 つの処理を，対象のアルゴリズムにおける一つのステップとみなす．

具体的に，JPEG で用いられる離散コサイン変換 (Discrete Cosine Transform, DCT) を DRP-1 上で実行した場合の，各ステップにおける必要 PE 数を考える．DCT の場合，処理を施すデータストリームは，通常のプログラムでは 8×8 の 2 次元配列に格納され，それぞれの要素を組み合せて加算した結果と，減算した結果について，まず行方向に積和および積差演算処理を行い，続いて列方向に同様の処理を行う．

図 4 は，DCT を DRP-1 上に実装した場合の各ステップにおける必要 PE 数を示す．ここでは，DRP-1 が無限の PE と分散メモリをもつものと仮定している．必要 PE の数は，並列性と同義ではないが，各ステップでの並列性の目安として考えることができる．この実装例の処理の流れは以下の通りである．

- 分散メモリ，レジスタの初期化を行い，入力データを分散メモリに格納する（ステップ 0〜6）
- 行方向への演算を行う（ステップ 7〜10 を 8 回反復する）
- 列方向への演算を行う（ステップ 11〜18 を 4 回反復し，続いてステップ 19〜26 を 1 回実行する）
- 出力データを外部に出力する (ステップ 27〜30)

図 4 に示した実装例では，それぞれ 8 個のデータを 8 個の分散メモリに格納している．この場合，行方向の演算は，同時に 8 個のデータを取り出すことができるため，比較的並列性を損うことなく 8 回の反復で終了することができる．しかし，列方向の演算は，配列の添字を変えながら順に 8 回アクセスする必要があることから，並列性が低下する．ここでは，2 つの分散メモリを同時にアクセスし，読み出して可能になった演算から順に実行する手法と，最終データの格納を演算とオーバラップさせる一種のソフトウェアパイプライニングを用いて，並列性を維持しつつ反復回数を削減している．

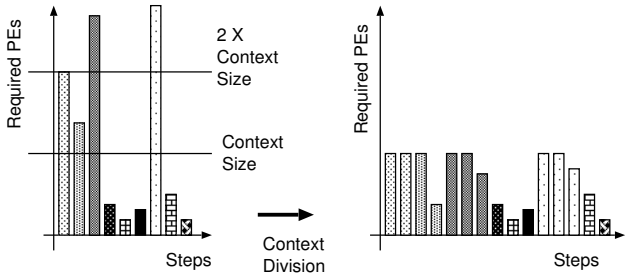


図 5　コンテキスト分割

ここで，あるアプリケーションを実行するための最小ステップ数 S_∞ を，反復を含めたステップの総数と定義する．図 4 に示す DCT のケースでは，$S_\infty = 72$ である（行方向演算 4×8, 列方向演算 $8 \times 4 + 8$）．なお，入出力に要するステップは，ダブルバッファ機構 [9] を採用することで隠蔽可能であることから，本論文では評価の対象としない．

最小ステップ数 S_∞ は，アルゴリズムとデータ構造に制約を受けるため，DCT のケースでは無限個の PE をもつとした場合でも 72 クロックを必要とする．よって，S_∞ は，時分割多重化せずに実行する場合の必要クロック数とみなすことができる．このときの実行時間 T_∞ は $T_\infty = C_\infty S_\infty$ となる．ここで，C_∞ は，1 つのコンテキストのみで実装した場合の最大遅延を示す．

3.2　コンテキスト分割

これまで述べたように，動的リコンフィギャラブルプロセッサ上でアプリケーションを実行する場合には，そのアプリケーションのもつ並列性と逐次性に従って，一連のステップに分割して実行する．しかし，図 4 に示すように，それぞれのステップでの必要 PE 数には，かなりのばらつきが生じる．並列性の高いステップでは多く PE を要求するが，並列性の低いステップではあまり PE を必要としていない．

ここで，実際に DRP-1 上でアプリケーションを実行する際には，1 コンテキストあたりで利用可能な PE 数の範囲内で，各ステップを各コンテキストに割り当てる必要がある．一つのコンテキストで利用可能な PE 数は限られているため，図 5 に示すように，要求する PE 数が一つのコンテキストで利用可能な PE 数よりも多いステップでは，処理をさらに分割する必要がある．本論文では，これをコンテキスト分割と呼ぶ．

リコンフィギャラブルプロセッサにおけるコンテキスト分割は，データフローグラフの流れと垂直にステップを分割する直列分割法が一般的である．ステップの切り口には必ずレジスタを挿入して，コンテキスト間でデータの交信を行う．このため，コンテキスト分割を行うことで，使用するレジスタの数が増加するが，最大遅延が削減される．

コンテキスト分割により，以下の式に示されるように，あるアプリケーションの実行に必要なステップ数 S_{size} は増加する．

$$S_{\text{size}} = \sum_{i=0}^{S_\infty - 1} \frac{PE_i}{PE_{\text{size}}}$$

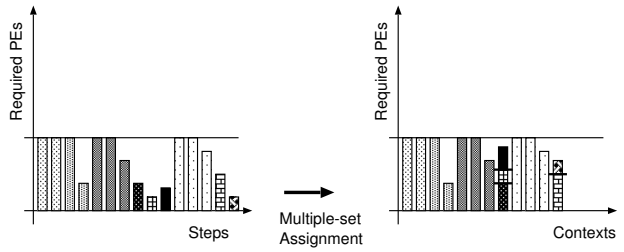


図 6　コンテキスト統合

ここで，PE_i はステップ i における必要 PE 数，PE_{size} は 1 コンテキストあたりの利用可能 PE 数を示す．

ここで，時分割多重実行により，あるコンテキストが半分のサイズとなった場合，最大利用 PE 数は減少するが，完全に半分になるわけではない．これは以下の理由に因る．

- 対象のアプリケーションを均等なサイズに分割することができない．例えば，定数乗算は，DRP コンパイラによりシフトと加算に展開されるが，これ以上細かく分割することが難しい．すなわち，割り当てる処理の中には，これ以上分割することのできないアトミックサイズが存在する．
- コンテキストサイズに適合しない分割が発生する．とりわけ，コンテキストサイズが大きい場合に起こり得る．
- コンテキスト間のデータ交信用のレジスタが増加する．

また，コンテキスト分割を行うことにより，最大遅延 C_{size} は分断される．このため，配線長が削減され動作周波数は向上する傾向にある．コンテキスト分割を行い，時分割多重実行を行う場合の実行時間 T_{size} は，$T_{\text{size}} = C_{\text{size}} S_{\text{size}}$ となる．

3.3 コンテキスト統合

一方，並列性が少なく，利用 PE 数の少ないステップが続く場合，それぞれのステップを一つのコンテキストに割り当てると，PE の利用効率が悪い．そこで，図 6 に示すように，一つのコンテキストに複数のステップを割り当て，各ステップを逐次的に実行することで，PE の利用効率を高めることができる．本論文では，これをコンテキスト統合と呼ぶ．

しかし，コンテキスト統合は以下のオーバヘッドをもたらす．まず，同一のレジスタや分散メモリにアクセスする複数のステップを，同一のコンテキストで実現するためには，データを切り替えるためのマルチプレクサが必要となる．このため，これらのステップを別々のコンテキストで実現する場合に比べて，余分な PE を消費する．

また，一般的に，演算を行うデータパスに含まれる PE は，そのデータが格納されるレジスタや分散メモリの周辺に配置するのが効率的である．ところが，コンテキスト統合により，複数のステップを同一コンテキスト上に実現する場合，データを保持する分散メモリやレジスタから離れた位置にデータパスを形成する PE が配置され，配線遅延の増大を招く可能性がある．

コンテキスト統合は，PE の利用効率を向上することで，必要コンテキスト数を削減する．さらに，コンテキストスイッチの回数を削減することで，電力効率を改善することも期待できる．しかし，必要なステップ数は変わらないため，基本的には実行サイクル数は変化しないことに留意する必要がある．

4. 評　　価

4.1 評価アプリケーションと評価方法

本論文では，前節で示した時分割多重実行による効果を明らかにするため，実際に以下に示す複数のストリームアプリケーションを DRP-1 上に実装し，時分割多重実行における性能と消費電力のトレードオフを解析した．

- JPEG デコーダにおける離散コサイン変換 (Discrete Cosine Transform: DCT)
- MP3 デコーダにおける変形逆離散コサイン変換 (Inverse Modified Discrete Cosine Transform: IMDCT)
- 高速フーリエ変換 (Fast Fourier Transform: FFT)
- アクティブ方向通過型フィルタ (Active Direction Pass Filter: ADPF)
- 誤り訂正符号 Viterbi デコーダ
- 共通鍵ブロック暗号 Advanced Encryption Standard (AES)
- 一方向ハッシュ関数 Secure Hash Algorithm 1 (SHA-1)

これらのアプリケーションを，C レベルの動作記述言語である BDL を用いて記述し，DRP コンパイラおよび統合開発環境 Musketeer を用いてコンパイルした．

ここで，時分割多重実行の性能と消費電力を評価するために，それぞれのアプリケーションに対して，コンテキストサイズを変化させて合成した．コンテキストサイズは，一つのコンテキストで利用可能な PE 数である．DRP の場合には Tile と呼ばれるクラスタ構造をもち，各 Tile は 64 個の PE と，分散メモリである VMEM/HMEM をもつ．本論文では，コンテキストサイズの変更は Tile 単位で行うものとし，実際に利用可能な Tile 数を変化させて合成した．

コンテキスト分割とコンテキスト統合は，手動で行うか，DRP コンパイラのオプションとして自動的に行うことが可能である．今回の実装では，コンテキスト分割は手動で最適化を行い，コンテキスト統合は基本的にはコンパイラに委ね，最適化の必要がある場合のみ手動で行った．

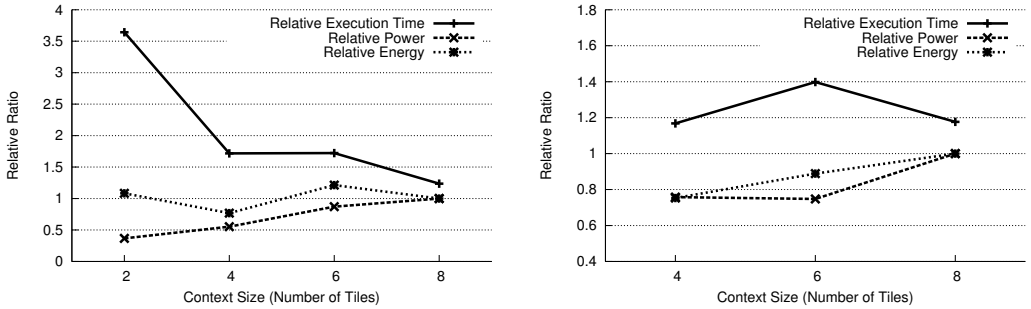
4.2 評 価 項 目

時分割多重実行を行うことによる性能の影響を検討するため，まず時分割多重実行を行う場合と行わない場合の実行時間の比率 R_{size} を評価する．すなわち，

$$R_{\text{size}} = \frac{C_{\text{size}} S_{\text{size}}}{C_{\infty} S_{\infty}}$$

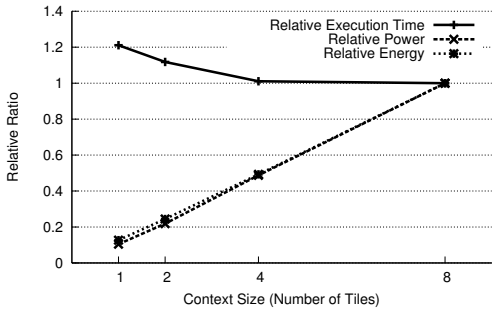
である．ここで，時分割多重実行を行わない場合の必要 PE 数は，しばしば DRP-1 の利用可能 PE 数 (512) を越えるため，実際には DRP-1 上にマッピングすることができないことが多い．この場合は，時分割多重実行を行わない場合の最大遅延 C_{∞} は，実際には計測することができない．このような場合には，最もコンテキストサイズの大きい，Tile 数が 8 のケースの最大遅延 C_{512} を代りに用いるものとする．一般的に，$C_{\infty} \geq C_{512}$ であることから，実際の C_{∞} を用いるよりも，R_{size} の値は小さくなる可能性がある．

次に，実装した各アプリケーションに対して予測される消費電力を計測し，アプリケーションの実行に要する必要エネル

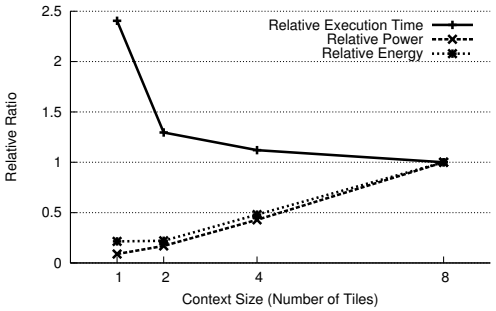


(a) DCT

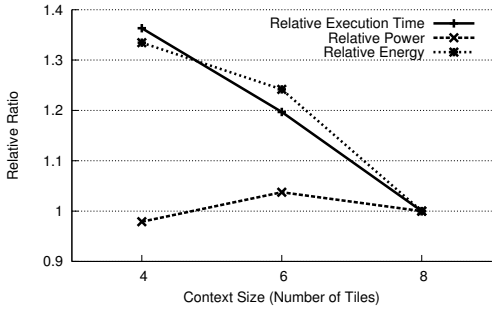
(b) IMDCT



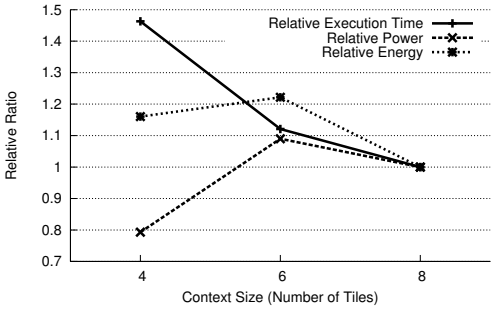
(c) FFT



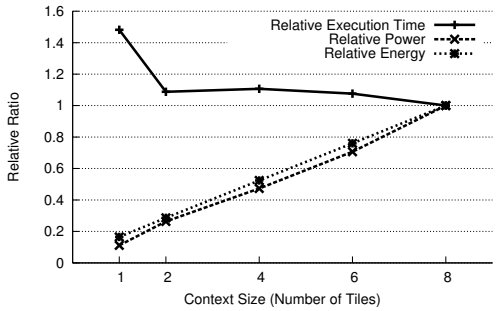
(d) ADPF



(e) Viterbi デコーダ



(f) AES-ECB



(g) SHA-1

図 7　コンテキストサイズと性能および消費電力の関係

ギーを算出する．今回の評価では，消費電力の値は統合開発環境 Musketeer における電力プロファイラの算出した値であり，実測値とは異なる可能性がある．また，消費電力の値は，DRP-1 を対象としたものであり，DRP-1 のコア部のみの電力で，I/O などの 3.3V 系の電力は含まれていない．また，プロファイラの算出する推定値は 8 Tile での消費電力であるため，電力プロファイラの推定値を補正して各コンテキストサイズに対する消費電力を見積った．

また，必要エネルギーは，対象のアプリケーションを実行するために必要な電力量である．すなわち，消費電力を P_{size} とすると，必要エネルギー E_{size} は以下で表すことができる．

$$E_{\text{size}} = P_{\text{size}} C_{\text{size}} S_{\text{size}}$$

4.3 評価結果

図 7 に，コンテキストサイズに対する実行時間，消費電力，必要エネルギーの関係を示す．これらの値は，8 Tile で実現した場合の値に正規化したものである．

各アプリケーションにおいて，コンテキストサイズを増やすことで，実行時間は減少し一定の性能向上を達成可能であることがわかった．特に DCT のように，並列性の高いアプリケーションでは，さらに Tile 数を増やすことで更なる性能向上を期待することが可能である．一方で，並列性が低く必要 PE 数の少ない FFT や ADPF, SHA-1 では，コンテキストサイズを増やしても更に並列実行が可能な部分がないため，必要な実行クロック数を削減することができない．加えて，コンテキスト統合によるマルチプレクサの増大により，最大遅延時間の増大を招き，返って性能を損ねる可能性があることがわかった．

また，コンテキストサイズと消費電力および必要エネルギーの関係において，DRP ではコンテキストサイズが増加すると，命令の割り当てられていない PE も含め稼働する PE の数が増加するため，消費電力も増加する．評価結果より，DCT や IMDCT などの並列性の高いアプリケーションでは，コンテキストサイズが増加するに従い消費電力は増加する．プロファイラの見積りでは，DCT では 2 Tile の場合で約 111[mW], 8 Tile で実装した場合は約 302[mW] の消費電力である．

DRP-1 の場合，消費電力のうちコンテキストスイッチに要する電力の割合は平均 5% 程度と少なく，クロック等の基本となる部分の電力が大半を占めている．故に消費電力は，コンテキストスイッチの頻度よりも動作周波数の影響が大きい．

一方，コンテキストサイズが増加すると，さらなる並列処理により実行クロック数が減少し，動作周波数が低下することで必要エネルギー量は減少する．DCT では，2 Tile から 4 Tile の場合に実行時間が大幅に減少するため，必要エネルギーが最も少ない．Viterbi デコーダや AES-ECB においては，コンテキストサイズの増加に従って順調に実行時間が短縮され，必要エネルギーも減少する．よって，比較的並列性の高いアプリケーションでは，対象の問題を解くための必要エネルギーの観点では，コンテキストサイズを増やした方がやや有利であるといえる．

FFT や ADPF, SHA-1 などの並列性の低いアプリケーションでは，1,2 Tile 以上にコンテキストサイズを増やしても，それ以上の並列性を引き出すことができず必要クロック数は変化しない．このため，コンテキストサイズの増加に伴ない消費電力は増加し，必要エネルギーも増加する．また，コンテキストサイズを増やすと，コンテキスト統合により最大遅延が増大し，必要エネルギーとしては不利になる傾向がある．よって，このようなアプリケーションでは，コンテキストサイズは小さい方が，必要エネルギーの観点では有利である．

これまでの評価結果より，コスト対性能比が最適なコンテキストサイズを選ぶことで，多くの場合において必要エネルギーも最適となることがわかった．

5. むすび

本論文では，動的リコンフィギャラブルプロセッサにおける，時分割多重実行の性能と消費電力の関係を議論した．実際にいくつかのストリームアプリケーションを NEC エレクトロニクス社の DRP-1 上に実装し，評価を行った．

まず，コンテキストサイズに対するアプリケーションの実行時間を評価した．DCT や IMDCT のように，並列性の高いアプリケーションでは，コンテキストサイズを増やすことで一定の性能向上を達成できることがわかった．一方で，並列性の低いアプリケーションでは，コンテキストサイズを増やしても更に並列実行が可能な部分がないため，必要な実行クロック数を削減することができないことがわかった．

また，コンテキストサイズと消費電力，必要エネルギーの評価結果より，DCT や IMDCT などの並列性の高いアプリケーションでは，コンテキストサイズの増加に従い消費電力は増加するが，並列性が高まることにより実行クロック数が減少し必要エネルギー量は減少する．一方，FFT や ADPF, SHA-1 などでは，コンテキストサイズの増加に伴ない消費電力は増加し，必要エネルギーも増加してしまうことがわかった．

全体的に，要求性能に対して，面積効率および電力効率を最適化するためには，事前に対象のアプリケーションの並列性と逐次性を調査しておく必要がある．その上で，IP コアとして登載する DRP コアの規模を決定することが有効であると考えられる．DRP の場合，これらの操作は一般的な C プログラミングの技術をもっていれば，統合開発環境を用いることで比較的容易に実現することが可能である．

今後は，コンテキストの切り替えメカニズムと消費電力の関係などに注目し，時分割多重実行の消費電力への効果をより詳細に渡って評価する予定である．その上で，更に低消費電力，高電力効率を実現する動的リコンフィギャラブルプロセッサのアーキテクチャの検討を行う予定である．

文　献

[1] 末吉, 天野（編）. リコンフィギャラブルシステム. オーム社, 2005.

[2] M. Motomura. A Dynamically Reconfigurable Processor Architecture. *Microprocessor Forum*, October 2002.

[3] T.Sugawara, K.Ide, and T. Sato. Dynamically Reconfigurable Processor Implemented with IPFlex's DAPDNA Technology. *IEICE Transaction on Information & System*, Vol. E87-D, No. 8, pp. 1997–2003, May 2004.

[4] Elixent. http://www.elixent.com/.

[5] PACT. http://www.pactcorp.com/.

[6] 天野, 阿部, 出口. 動的リコンフィギャラブルプロセッサの基本的トレードオフの解析. 第 4 回リコンフィギャラブルシステム研究会 論文集, pp. 25–32, September 2004.

[7] H. Amano, S. Abe, Y. Hasegawa, K. Deguchi, and M. Suzuki. Performance and Cost Analysis of Time-multiplexed Execution on the Dynamically Reconfigurable Processor. In *Proceedings of IEEE Symposium on Field-programmable Custom Computing Machines (FCCM2005)*, April 2005. poster session.

[8] 粟島, 戸井, 中村, 紙, 加藤, 若林, 宮澤, 李. 動的再構成可能チップ DRP の C コンパイラ. 電子情報通信学会技術研究報告 VLD2003-118, Vol. 103, No. 578, pp. 23–28, January 2004.

[9] H. Amano, S. Abe, K. Deguchi, and Y. Hasegawa. An I/O mechanism on a Dynamically Reconfigurable Processor -Which should be moved: Data or Configuration?-. In *Proceedings of International Conference on Field Programmable Logic and Application (FPL2005)*, August 2005. to apear.