社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2009-SLDM-138 (1)
2009 / 1 / 29

動的リコンフィギャラブルプロセッサMuCCRA-3の実装と再構成オーバヘッドの削減

佐野 徹† 天野 英晴†

† 慶應義塾大学理工学部
〒 223–8522 神奈川県横浜市港北区日吉 3-14-1
E-mail: †muccra@am.ics.keio.ac.jp

あらまし これまで我々は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のアーキテクチャを解析，提案することを目的としてMuCCRA-1,2を実装・評価を行ってきた．その評価結果をフィードバックし，より低消費電力で高い性能を得ることを目的としてMuCCRA-3を実装した．その結果，MuCCRA-1の2倍以上の性能が得られることが分かった．さらに，MuCCRA-3をベースに，動的リコンフィギャラブルプロセッサにおける性能上のボトルネックの1つである，コンフィギュレーションデータ転送を削減することを目的として，2つの手法を提案する．1つ目の手法は，コンフィギュレーションデータを演算用のデータパスを利用して転送するデータパスコンフィギュレーション，2つ目の手法は，レジスタファイルやスイッチといった単位で再構成を行う細粒度部分再構成である．これらの手法を実装したMuCCRA-3は，実装しないMuCCRA-3に比べ転送時間を1/3に削減することができた．
キーワード 動的リコンフィギャラブルプロセッサ，再構成，オーバヘッド

Implementation of Dynamically Reconfigurable Processor MuCCRA-3 and Methods for Reconfiguration Overhead Reduction

Toru SANO† and Hideharu AMANO†

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

Abstract We have developed and evaluated MuCCRA-1 and 2 in order to analyze architectural trade-off in dynamically reconfigurable prosessors. Here, we propose MuCCRA-3 which has higher performance, and works with low-power consumption based on the implementation and evaluation of MuCCRA-1,2. As a result of the evaluation, it appears that MuCCRA-3 achieves almost double performance as MuCCRA-1. Furthermore, we propose two methods to reduce the configuration data transfer cycles which may form a bottleneck of performance. The first method is called "Data Bus Configuration" that enables to transfer the configuration data through the data bus. And the other is "Fine-grained Partial Reconfiguration", which enables to reconfigure unit by unit. These methods reduce the configuration transfer cycle about 30%-50%.
Key words Dynamically Reconfigurable Processor,Reconfiguration,Overhead

1. はじめに

近年，モバイル機器の多機能化・高性能化に伴い，低消費電力で高い性能を得ることができる動的リコンフィギャラブルプロセッサに関する研究が盛んに行われ，様々なアーキテクチャが提案されている．また，既にいくつかの商業用プロセッサが発表，実用化もされている．

一般的な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，単純な演算コアであるProcessing Element(PE)を複数並べた構造を持つ．PEの動作やPE間の接続を決定するコンフィギュレーションデータ(構成情報)を複数セット保持し，これを実行時に切り替えることによって，実行するアプリケーションに適したデータパスの構成と高い面積効率を実現している．

このマルチコンテキスト型動的リコンフィギャラブルデバイスについて，多数の小さなアレイによるマルチコア型アーキテ

表 1　MuCCRA アーキテクチャの比較

アーキテクチャ	半導体プロセス	データ幅	コンテキストメモリ深さ
MuCCRA-1	Rohm 180nm	24	64
MuCCRA-2	Aspla 90nm	16	16
MuCCRA-3	Fujitsu 65nm	16	32

クチャの検討，設計時のパラメータ変更によって，アプリケーションに最適なモジュール構成を可能とすること，また，90nm 以降の微細プロセスを視野に入れた低消費電力化などを目的として，デバイスレベルから検討を行うために，我々は MuCCRA (Multi-Core Configurable Reconfigurable Architecture) プロジェクトを立ち上げた．

これまで，Rohm 社の 180nm プロセスを用いた MuCCRA プロジェクトのプロトタイプチップである MuCCRA-1 [1]，Aspla 社の 90nm プロセスを用いた MuCCRA-2 [2] の実装，評価を行ってきた．その結果，DSP や FPGA に比べ低い消費電力で高い性能が得られる一方，マイクロプロセッサにはない，チップ実装面積オーバヘッド，コンフィギュレーションデータ転送時間のオーバヘッドを生じることが分かった．例えば，MuCCRA-2 では，構成情報転送によるストールが全実行時間の 30%から 50%を占める．また，構成情報を保持するメモリの読み書きによる消費電力が，全体の半分を上回ることが分かった．したがって，これらのオーバヘッドの削減が，マルチコンテキスト型動的リコンフィギャラブルデバイスの更なる低消費電力化，高性能化のために重要である．

そこで本論文では，まず，提案手法のベースとする MuCCRA-3 の実装，予備評価について述べる．MuCCRA-3 アーキテクチャでは，これまでの MuCCRA-1，MuCCRA-2 での評価からのフィードバックを行い，実装上の改善を行った．さらに，構成情報の転送時間を削減することを目的として，データパスを利用した構成情報の転送と，細粒度部分再構成の 2 つの提案手法について述べ，これらの実装方法と評価結果を報告する．

2. MuCCRA-3 アーキテクチャ

MuCCRA-3 は，チップ試作時のダイ面積に余裕を持たせるため，扱うデータ幅を 16bit とした．表 1 に，これまでの MuCCRA アーキテクチャの，利用した半導体プロセス，データ幅，コンテキストメモリの深さの比較をまとめる．MuCCRA-3 に限らず，データ幅や構成情報を保持するコンテキストメモリのワード数は，チップ試作時のダイ面積，利用できる SRAM マクロに依存している．

2.1 PE アレイ

MuCCRA-3 は，これまでの MuCCRA-1,2 同様，4x4 の PE によるアレイ構造を持つ．図 1 に PE アレイの概観を示す．演算データを保持する MEM は，これまでアレイ下部に 4 つ配置していたのものを，アレイ上部にも配置し合計 8 つとした．これは，MuCCRA でのアプリケーションマッピング時，メモリの読み書きがボトルネックの 1 つとなっていたためである．ただし，MEM の深さは MuCCRA-1 の 512 の半分である 256 ワードとした．また，MEM はチップ外とのデータ交換に利用されるが，これまで同様ダブルバンク構造であり，IO 時間の削減，隠蔽が可能である．ただし，チップ外からは常に両バンクの読み書きを許可し，実機におけるデバッグを容易にしている．さらに，MuCCRA-1,2 では MEM の読み書きアドレスは PE

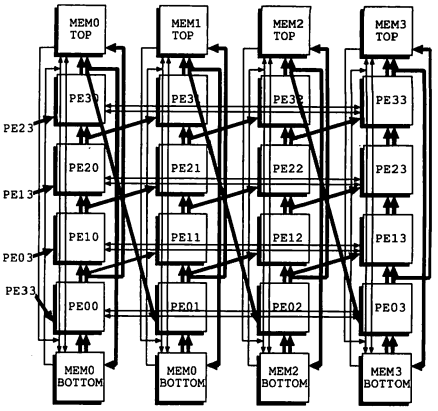

図 1　MuCCRA-3 の PE アレイ

から与えられるのみであったが，これは演算に利用できる PE を減らしてしまうという問題があった．そこで，MuCCRA-3 では，MEM 内にカウンタを実装し，簡単なアドレス計算であれば MEM 単体でできるようにした．カウンタのリセット，カウントダウン，カウントアップ等の動作は，MEM の構成情報により変更可能である．

PE 間を接続する結合網は，これまで MuCCRA-1,2 で採用してきた Switching Element(SE) によるアイランドスタイルに加え，近接の PE と接続する直結接続網との両方により構成される．図 1 中の，太線が直結接続網，細線が SE による接続網を表している．これはアイランドスタイルを採用した MuCCRA-1 を直結接続網にて実装した MuCCRA-D の評価 [3] から，直結接続網が動作周波数を向上させる一方で，データ移動の柔軟性を制限していたことが分かったためである．これにより，アプリケーションマッピング時に，近接の PE へデータを移動する場合には SE を再構成する必要のない直結接続網を利用し，遠い PE へデータを渡す場合には，SE を利用した接続といった使い分けが可能となり，データパスの形成がより容易である．SE によるチャネル網は，A,B の 2 チャネルが利用可能で，SE にて A から B，B から A へ乗り換えを可能とした．SE は配置配線のし易さから，PE 内のユニットとして持つようにした．

2.2 PE

MuCCRA-1,2 では，各 PE は，演算を行う Arithmetic and Logic Unit(ALU)，シフト演算，定数供給等を行う Shift and Mask Unit(SMU)，レジスタファイル (RF) から構成されていた．これに対して MuCCRA-3 では，2 項の四則演算，乗算，ビットシフト，比較演算を行う ALU，PE 内外からのデータ選択または定数生成を行い，1 項のシフト演算，ビット反転をする 2 つの ALU_SEL，これまで同様 8 エントリの 2 リードポート，1 ライトポートの RF によって PE は構成される．RF は，深さ 8 の FIFO として動作する FIFO モードを持つ．図 2 に MuCCRA-3 の PE の内部構造を示す．

ALU_SEL_A,ALU_SEL_B, および RF の *A_IN* には，*RF* の出力，直結接続網，SE による接続網が接続されており，構成情報により選択が可能である．MuCCRA-1,2 の ALU では，例えば同じ加算でも，そのまま演算をする命令，入力をビット反転する命令など多彩な演算が可能であったために，ALU には 32 命令存在したが，MuCCRA-3 では，ALU は単純な 13 命令

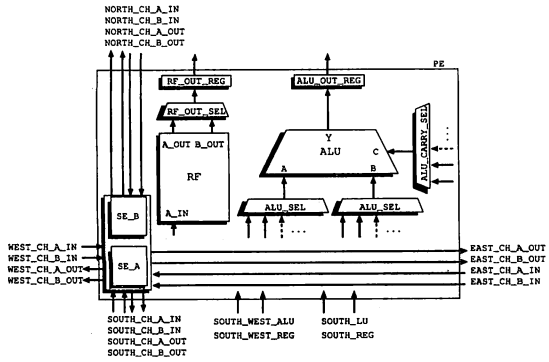

図 2　MuCCRA-3 の PE

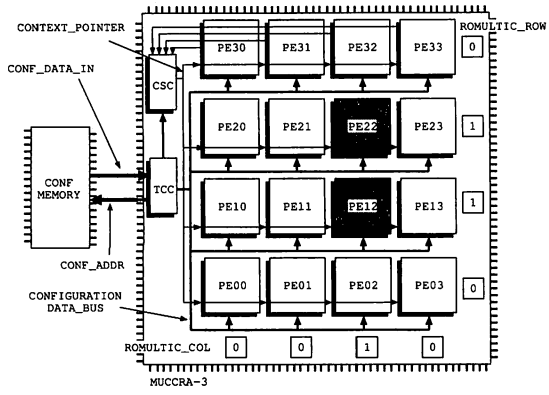

図 3　MuCCRA-3 の制御機構

のみとし，ビット反転などは ALU_SEL が行う．これにより各ユニットの機能分離が明確になり，アプリケーションマッピングを容易にする．

また，MuCCRA-3 では PE の出力は常にラッチされる．MuCCRA-1,2 では PE の出力は直接他の PE へ出力されており，これは任意のデータパスを形成するのに役立つ一方，物理的なクリティカルパスを非常に長くするという問題があった．このため，アプリケーションにより動作周波数が大きく変化し，実装時のタイミング解析も困難であった．MuCCRA-3 では，よりパイプライン動作志向で，より高い周波数での動作をターゲットとしているため，直結接続網の追加とともに，このような出力ラッチという構造をとった．

2.3　再構成機構・動作制御

MuCCRA-3 は，コンテキストとよばれる，複数のコンフィギュレーションデータのセットを切り替えて再構成を行う，マルチコンテキスト型の動的リコンフィギャラブルプロセッサである．各 PE，MEM は自身が持つ CONTEXT MEMORY から，構成情報を読み出して再構成する．したがって実行に先だって，CONTEXT MEMORY へ構成情報を転送しておく必要がある．この転送は Task Configuration Controller(TCC) が行う．図 3 に MuCCRA-3 の制御機構を示す．

まず，TCC は全コンフィギュレーションデータを保持しているチップ外の CONF MEMORY に読み出しアドレス (CONF_ADDR) を与え，構成情報 (CONF_DATA_IN) を得る．TCC と PE,MEM とは共有バスである CONFIGURATION_DATA_BUS で接続されており，TCC は，構成情報と共にユニットの種類も一緒に送り，PE 側は受け取るべきデータであるか判断をして CONTEXT_MEMORY に書き込む．また，MuCCRA-1,2 同様，転送に RoMultiC [4] と呼ばれるマルチキャスト転送を行う．RoMultiC は，行と列のマルチキャストビット (ROMULTIC_ROW,ROMULTIC_COL) を用いる転送手法である．コンフィギュレーションデータは，行と列の該当ビットが共に 1 である場合 (図 3 の例では PE12 と PE22)，CONTEXT_MEMORY に書き込まれる．これにより複数のモジュールへ同一の構成情報を転送する場合に，転送時間を短縮することができる．

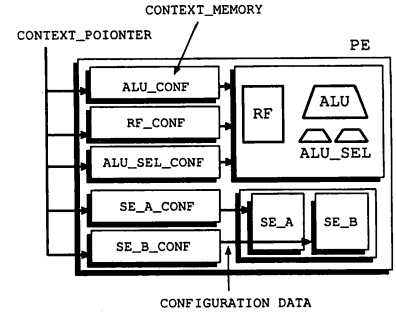

図 4　MuCCRA-3 のコンテキスト読み出し機構

全てのコンフィギュレーションデータを転送すると，Context Switch Controller(CSC) は CONTEXT_MEMORY の読み出しアドレスである CONTEXT_POINTER を与えて実行を開始する．CONTEXT_POINTER は全 PE，MEM，CSC で共通であり，図 4 のように CONTEXT_MEMORY へ接続されている．MuCCRA-1,2 では，PE のコンフィギュレーションデータの幅は 100 ビット近くあり，1 つの CONTEXT_MEMORY からデータを読みだしていたが，MuCCRA-3 では PE 内に ALU,ALU_SEL_A,ALU_SEL_B,RF,SE_A,SE_B ごとに細分化された CONTEXT_MEMORY から読みだす．1 ユニットあたりのコンフィギュレーションデータの幅は共通で 16 ビットである．したがって，PE だけで比較するとコンフィギュレーションデータのビット数は，100 ビットから 64 ビットへ大幅に削減できている．また，細分化により構成情報の数が増加し，マルチキャストビットの割合も増えるが，RoMultiC が効率的に機能し転送サイクルが減少する [5]．CSC 自身も CONTEXT_MEMORY を持ち，再構成をしながら動作制御を行う．この CONTEXT_POINETR を変更することによって MuCCRA-3 は，1 サイクルで再構成が可能である．通常は CONTEXT_POINTER をインクリメントさせるが，PE30,31,32,33 の ALU 出力をもとに条件分岐，または無条件分岐ができる．

2.4　MuCCRA-3 の予備評価

MuCCRA-3 の性能，消費電力の評価を行った．Verilog-HDL にて記述，富士通 65nm プロセスを用いた．論理合成には Synopsys 社 Design Compiler 2007.12-SP3，電力評価には Synopsys 社 Prime-Time 2007.12-SP3 を利用した．評価アプリケーションは 2 次元離散コサイン変換 (2D-DCT) である．動作検証には，Cadence 社 NC-Verilog 8.1 を用いた．

比較対象として，Texas Instruments 社の DSP である TMS32C6201 の評価も行った．ただし，TMS32C6201 が 32bit

表 2　MuCCRA-3 評価結果

アーキテクチャ	実行時間	動作周波数	平均消費電力	消費エネルギー
TMS32C6201	12.3μsec	167MHz	684mW	8412.2nJ
MuCCRA-1	7.8μsec	25MHz	85.1mW	664.8nJ
MuCCRA-3	3.8μsec	41MHz	11.0mW	41.8nJ

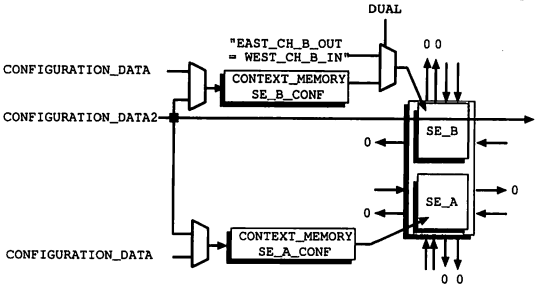

図 5　データバスによるコンフィギュレーションデータバスの再構成

演算を行うの対して，MuCCRA-1 は 24bit，MuCCRA-3 は 16bit であることに注意されたい．MuCCRA-2 の結果がないのは，MuCCRA-2 用に 2D-DCT を実装しなかったためである．結果をみると，DSP と比較して演算に要した消費エネルギーは，約 1/200 となっている．MuCCRA-1 と比較しても，1/20 程度と非常に小さいことが分かる．これは，全消費電力中大きな割合を占めるコンテキストメモリのワード幅を削減したこと，PE が出力をラッチすることにより値の伝搬が減少，結果，トランジスタのスイッチングが減少したためと考えられる．

3. 提案手法

MuCCRA-3 では，2D-DCT を 3.36μsec で実行できる一方，構成情報の転送に 10μsec を要する．そこで，本節では，この構成情報転送による時間上のオーバヘッドを削減するための手法 2 つについて述べる．

3.1 データバスコンフィギュレーション

2.3 節で述べたように，構成情報は専用の CONFIGURATION_DATA_BUS によって転送される．このバスを 2 つにすれば 2 倍の構成情報を転送が可能である．しかし，これは配線面積を増加させる [6]．そこで，演算がストールしているときに転送する場合には，CONFIGURATION_DATA_BUS に加えて，演算用のデータバスに構成情報をのせて転送をすることによって，物理的な配線を追加することなく，2 つ目の構成情報を転送することができる．

図 5 にこの手法の機構を示す．まず，演算がストール，かつ TCC が構成情報を転送する場合，TCC は全 PE に対して DUAL 信号を送る．PE は，DUAL 信号を受け取ると，SE_B 用の CONTEXT_MEMORY の出力セレクタで，CONTEXT_MEMORY のデータではなく，固定の西から東へデータを流す構成情報 ("EAST_CH_B_OUT=WEST_CH_B_IN") に切り替え，図 1 に示した SE による接続網のチャネルの 1 つを，コンフィギュレーションデータバスとして機能するように再構成をさせる．

外部の CONF_MEMORY は，TCC からの CONF_ADDR をもとに連続した 2 つのデータを読み出し，CONF_DATA_IN, CONF_DATA_IN2 として MuCCRA-3 に送る (図 6)．TCC は，CONF_DATA_IN2 の構成情報を SE による接続網 (CONFIGURATION_DATA_BUS2) にのせ，転送をする．PE は，専用バスである CONFIGURATION_DATA_BUS と CONFIGURATION_DATA_BUS2 から，CONTEXT_MEMORY に書くべきデータを選択し，書き込む．コンフィギュレーションデータのフォーマットから，両方のデータを同時に CONTEXT_MEMORY へ書き込むべき状態にはならない．

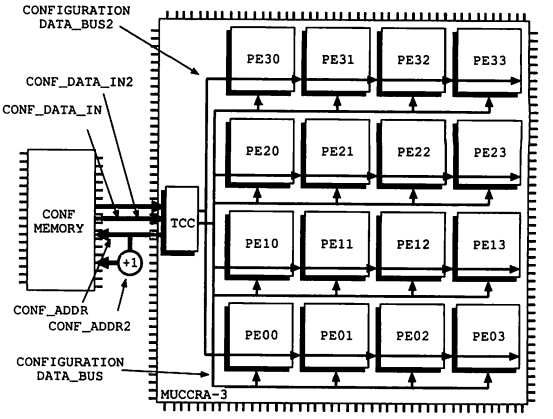

図 6　データバスコンフィギュレーション

このように，この手法は，チップ外部の CONF_MEMORY がデュアルポートメモリであること，またはインタリーブすることが前提であり，また CONF_DATA_IN2 のために MuCCRA-3 の IO ピンは増加するが，配線面積を節約しながらも，2 つの構成情報を同時に転送することが可能となる．

3.2 細粒度部分再構成

MuCCRA-3 では，再構成時に，すなわち CONTEXT_POINTER が変更された時，データパスが変更されない部分があっても全ユニットが再構成する必要がある．例えば，図 7 の例では，CONTEXT3 から CONTEXT4 への変更は ALU_SEL_A ユニットのみであるが，ALU_SEL_A のみならず，全ての PE 内の CONTEXT_MEMORY から構成情報が読み出され再構成される．これは CONTEXT_MEMORY に共通の CONTEXT_POINTER が接続されており，変更を行う必要がないユニットの CONTEXT_MEMORY の読み出しアドレスも変化してしまうためである．さらに，図 7 中の，SE_B は使用されていないが，SE_B が予期しない動作をしないように，また，トランジスタスイッチングによる電力を抑えるため，SE_B も安全な動作をするように再構成を行う．これは動作状態の PE が，使用されない PE に変更される場合においても同様で，全ユニットが安全な動作をするように再構成をする必要がある．このような本来不要な再構成は，CONTEXT_MEMORY の電力消費と構成情報の転送時間を増大させる．

そこで，SE_A，ALU_SEL_A といったユニット単位で部分再構成ができるようにすることで，このオーバヘッド削減が期待できる．これまでリコンフィギャラブルプロセッサの部分再構成は，PE や PE アレイ単位で行うものであったが，本研究で提案する部分再構成は，PE 内のユニット単位で行う細粒度部分再構成である．

これを実装するために，ALU のコンフィギュレーションデータの空きビットに，図 8 のような，ユニットが再構成を行う

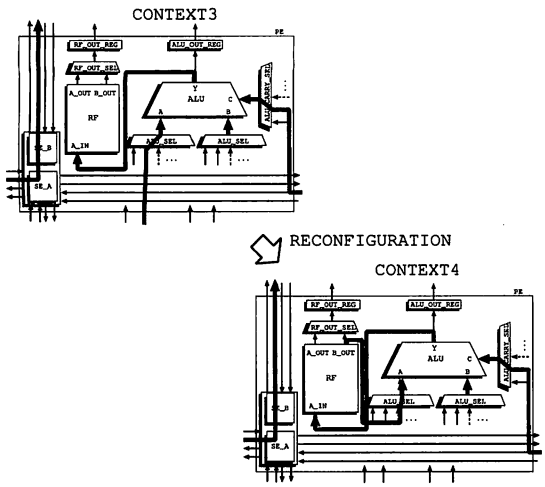

図 7　部分的な再構成

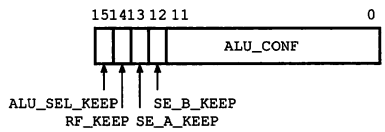

図 8　KEEP_FLAG の追加

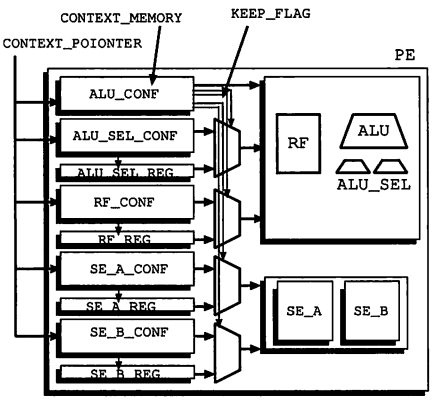

図 9　細粒度部分再構成

かを指示するフラグである KEEP_FLAG を追加した．この KEEP_FLAG が立っている場合には，そのユニットは再構成を行わず，直前の構成を保持する．構成の保持には，単純にレジスタを利用して保持する方法をとった．

図 9 に示すとおり，CONTEXT_MEMORY 毎にワード幅がコンフィギュレーション長，深さ 1 のレジスタを接続し，KEEP_FLAG を先読みし，フラグの立っているユニットは，現在の構成情報をレジスタに書き込んでおく．なお，KEEP_FLAG をもつ ALU_CONF だけは，常に CONTEXT_MEMORY から構成情報を読み出す．再構成時，KEEP_FLAG によって，CONTEXT_MEMORY とレジスタのどちらから読み出すかを選択する．この際，フラグが立っている場合は，CONTEXT_MEMORY を DISABLE にしてメモリの電力消費を抑える．

レジスタを追加する分，面積コストの増加が懸念されるが，Instruction Buffer [7] との共有が可能である．Instrcution Buffer とは，動的リコンフィギャラブルプロセッサの PE を簡易 CPU として利用するための命令バッファである．MuCCRA の対象とするアプリケーションは主にストリーム処理であるが，並列度の高い処理の間には，動的リコンフィギャラブルプロセッサでの動作に向かない低並列度の処理が存在する場合が多い．そこで，このような処理を MuCCRA 上で簡便に処理するため，PE を CPU とみなして処理をする．この際，CONTEXT_MEMORY 内のデータを破壊しないように命令を転送するため Instruction Buffer を用いる．このバッファとの共有により面積コストを増加させることなく細粒度部分再構成の実装が可能である．

表 3　面積コスト

	ゲート数	面積コスト
MuCCRA-3	981,410	-
+データパスコンフィギュレーション	966,539	-1.5%
+細粒度部分再構成	993,303	+1.2%

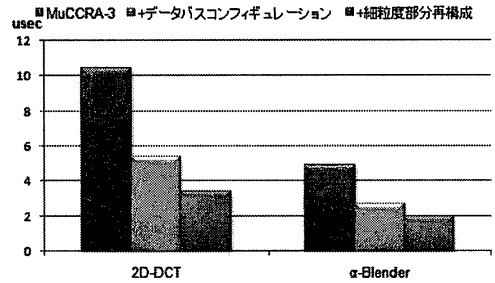

図 10　転送時間

4. 評　　価

MuCCRA-3 にデータパスコンフィギュレーション機構を追加，それに対してさらに細粒度部分再構成を追加して，MuCCRA-3 との比較評価を行った．

4.1 実装面積コスト

2 つの提案手法とも，面積コストを増大させることなく，構成情報転送による再構成オーバヘッドを削減することを目的としている．そこで，まず両手法の面積コストを評価した．いずれもクリティカルパス 26nsec(≒ 38.5MHz) を制約として合成を行い，NAND ゲート数換算で評価した．表 3 に結果を示す．

データパスコンフィギュレーションでは，ゲート数が MuCCRA-3 に対して減少しているが，誤差範囲といえる．細粒度部分再構成では 1%程度の面積コスト増加がみられるが，これは主に CONTEXT_MEMORY とレジスタを切り替えるマルチプレクサが原因であると考えられる．いずれの手法でも，小さい面積コストで実装できていることが分かる．

4.2 実行時間

対象アプリケーションとして，2D-DCT と画像処理フィルタであるアルファブレンダ (α-Blender) を実行，論理合成後のネットリストを用い，コンフィギュレーションデータの転送時間と，アプリケーション実行時間の評価を行った．なお，細粒度部分再構成に関しては，KEEP_FLAG を手動で追加，最適化を行った．いずれも 38.5MHz で実行し，手法に関わらず実行時間は 2D-DCT で 3.8μsec，α-Blender で 2.3μsec であった．変化があったのはコンフィギュレーションデータ転送時間である．図 10 に転送時間の評価結果を示す．

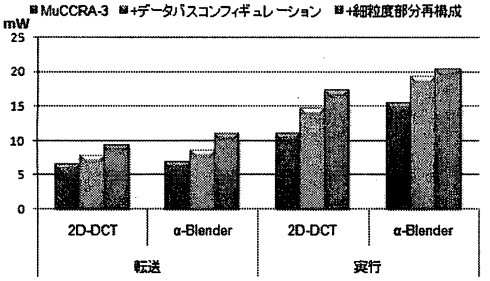

図 11　平均消費電力

表 4　消費エネルギー

	2D-DCT[nJ]	α-Blender[nJ]
MuCCRA-3	109.1	69.1
+データパスコンフィギュレーション	97.3	67.1
+細粒度部分再構成	97.0	69.0

評価結果から，データパスコンフィギュレーションは MuCCRA-3 の 2 倍のコンフィギュレーションデータ転送能力により，転送時間は半減しており，これはアプリケーションに依存しない．細粒度部分再構成では，さらに 1 割から 2 割の転送時間削減を実現している．これは，不要なコンフィギュレーションデータが最適化により削減されたためで，アプリケーションに依存する．主な最適化のポイントは次の 3 点である．

- 初期化処理
- レジスタの書き込み停止のみの変化
- SE 接続網 A と B のうち，未使用の B を再構成させない

これらは，細粒度部分再構成のための最適化が容易かつ高い効がある．さらにデータパスの変化が少なくなるようにアプリケーションをマッピングすることで，更なる最適化が可能であると考えられる．

4.3 消費電力・消費エネルギー

各アプリケーションの実行中，構成情報転送中の平均消費電力を評価した結果を図 11 に示す．転送，実行ともに手法の追加ごとに 1 割から 2 割消費電力が増加してしまっている．より詳細に電力評価を行ってみると，組み合わせ回路による電力消費の割合が増加していることから，データバス上の構成情報によるスイッチングと，コンフィギュレーション選択のマルチプレクサ増加によるものと考えられる．特に，細粒度部分再構成の場合，メモリの電力の割合は減少し，レジスタと組み合わせ回路の電力割合が増加した．したがって，CONTEXT_MEMORY の読み出し回数を減らしたことによる電力削減効果よりも，レジスタ追加による電力増加の方が大きかったため，全体の消費電力が増加してしまったと考えらえる．

次に，転送時間，実行時間とそれぞれの平均消費電力から，アプリケーションの実行に要した全消費エネルギーを求めた．表 4 に結果を示す．各手法では消費電力が増えた一方，転送時間が大幅に削減されたことで消費エネルギーは，減少，もしくは変わらなかったということが分かる．

5. おわりに

本研究では，これまでの MuCCRA の実装を踏まえて改善を行った MuCCRA-3 を実装，DSP や MuCCRA-1,2 に比べ 2 倍以上の高い性能，1/20 以下の低消費電力を実現した．また，主に構成情報の転送時間を削減することを目的として，データパスコンフィギュレーションと細粒度部分再構成を提案した．その結果，少ない面積コストで転送時間を 5 割から 6 割削減できることを示した．ただし，消費電力は 1 割から 2 割増加した．この消費電力は，主に追加したレジスタやマルチプレクサによるもので，今回レジスタの追加という単純な実装方法をとったためである．これは CONTEXT_MEMORY をクロックゲーティングするといった実装方法により改善できると考えられる．今後は，この実装手法を用いた場合の評価，さらに配置配線を行いより精確な評価を行っていく予定である．

謝　辞

本研究は，科学技術振興機構「JST」の戦略的創造研究推進事業「CREST」における研究領域「情報システムの超低消費電力化を目指した技術革新と統合化技術」の研究課題「革新的電源制御による次世代超低電力高性能システム LSI の研究」による．

本研究は東京大学大規模集積システム設計教育研究センターを通し，株式会社半導体理工学研究センター・(株) イー・シャトルおよび富士通株式会社・株式会社先端 SoC 基盤技術開発 (ASPLA)・ローム (株)・凸版印刷 (株)・シノプシス株式会社・日本ケイデンス株式会社・メンター株式会社の協力で行なわれたものである．

文　献

[1] 中村 拓郎, 長谷川 揚平, 堤 聡, 松谷 宏紀, Vasutan Tunbunheng, Adepu Parimala, 西村 隆, 加東 勝, 斉藤 正太郎, 佐野 徹, 関 臣, 平井 啓一郎, 毛 凱毅, 天野英晴：“動的リコンフィギャラブルプロセッサ MuCCRA の実装”, 信学報 RECONF, pp. 43–48 (2007 年 1 月).

[2] H.Amano, Y.Hasegawa, S.Tsutsumi, T.Nakamura, T.Nishimura, V.Tanbunheng, A.Parimala, T.Sano, M.Kato: “MuCCRA chips: Configurable dynamically-reconfigurable processors”, *Proc. of the ASSCC '07. IEEE Asian* (Nov. 2007).

[3] M.Kato, Y.Hasegawa, H.Amano: “Evaluation of MuCCRA-D: A Dynamically Reconfigurable Processor with Directly Interconnected PEs”, *Proc. of The 2008 International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA'08)* (Aug. 2008).

[4] V.Tunbunheng, M.Suzuki, H.Amano: “RoMultiC: Fast and Simple Configuration Data Multicasting Scheme for Coarse Grain Reconfigurable Devices”, *Proc. of IEEE FPT*, Springer, Berlin, pp. 129–136 (2005).

[5] 中村 拓郎, 長谷川 揚平, 堤 聡, Vasutan Tunbunheng, 天野英晴：“動的リコンフィギャラブルデバイスにおける構成情報配送のためのマルチキャスト手法の提案”, 先進的計算基盤システムシンポジウム (SACSIS2008) 論文集, pp. 385–392 (2008).

[6] 佐野 徹, 中村 拓郎, 堤 聡, 長谷川 揚平, 天野英晴：“動的リコンフィギャラブルプロセッサ MuCCRA におけるコンフィギュレーションデータ転送時間の削減”, 信学報 RECONF, pp. 55–60 (2007 年 5 月).

[7] T.Sano, M.Kato, S.Tsutsumi, Y.Hasegawa, H.Amano: “Instruction Buffer Mode for Multi-Context Dynamically Reconfigurable Processors”, *Proc. of The 18th IEEE International Conference on Field-Programmable Logic and Applications (FPL2008)* (Sep. 2008).