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

2008－SLDM－133（24）
2008 / 1 / 17

プロセッサモードを組み込んだALUベース動的再構成デバイス

神山 真一† 廣本 正之† 越智 裕之† 中村 行宏††

† 京都大学大学院 情報学研究科 通信情報システム専攻
〒 601-8501 京都市左京区吉田本町
†† 立命館大学 総合理工学研究機構
〒 525-8577 滋賀県草津市野路東 1-1-1
E-mail: †reconf@easter.kuee.kyoto-u.ac.jp

あらまし ALUを演算要素とする粗粒度再構成デバイスはワード単位の処理を効率よく実行できる一方で，複雑な制御を伴う逐次処理などにおいては性能を発揮しがたい．本研究では，プロセッサモードまたはALUアレイモードのいずれかを選択して使用できるタイルと呼ばれる基本要素がアレイ状に配列した構造をもつ粗粒度再構成デバイスを提案する．このデバイスはアプリケーションに応じてプロセッサとハードウェアエンジンの比率を変えて処理効率を最適化できる．また，タイル中のALUを始めとする多くの回路資源はプロセッサモードとALUアレイモードの両方で有効に活用されており，面積効率に優れている．本稿では，提案デバイスのアーキテクチャと設計結果について報告する．

キーワード 再構成デバイス，粗粒度，ALUアレイ，ソフトコアプロセッサ

A Tile Based Dynamically Reconfigurable Architecture with Dual ALU-array/RISC Processor Operating Mode Capability

Shin'ichi KOUYAMA†, Masayuki HIROMORO†, Hiroyuki OCHI†, and Yukihiro NAKAMURA††

† Dept. of Communications and Computer Eng., Graduate School of Informatics, Kyoto Univ.
†† Research Organization of Science and Engineering, Ritsumeikan Univ.
E-mail: †reconf@easter.kuee.kyoto-u.ac.jp

Abstract ALU-based coarse-grained reconfigurable devices are suitable for parallel word-wise processing. However, they are not efficiently used for sequential processing with complicated controls. In this study, we propose a tile based dynamically reconfigurable architecture with dual ALU-array/RISC processor operating mode capability, in order to achieve performance improvement by changing operating mode of the device according to characteristics of target applications. The proposed device consists of an array of the basic element "tile." Each "tile" operates either as a 16-bit integer RISC processor or as ALU-based hardware accelerator. This paper shows the architecture of the device and a result of its implementation.

Key words reconfigurable device, coarse-grained, ALU-array, softcore processor

1. はじめに

近年，再構成デバイスが，プロセッサ類より低消費電力，専用回路よりも低コストといった長所もあり，広く利用されている．再構成デバイスとして，(a)FPGA等の一括再構成あるいは部分的再構成可能デバイス，(b)PCA [1], [2], Cell Matrix [3], [4] などの動的自己再構成型デバイス，(c) DAP/DNA [5], DRP [6] などのコンテキストスイッチ型動的再構成可能デバイスなどが提案，開発されてきた．(a) や (b) は LUT (Look-Up Table) のような細粒度の再構成可能な回路資源が敷き詰められた構造を持ち，柔軟性が高い反面，構成情報量，回路規模が非常に大きく，再構成に要するオーバヘッドが大きい．一方，(c) はALUのような粗粒度の演算資源をベースにしており，中央集権的な制御機構により比較的少数の演算器を高速に再構成することで高い性能を発揮することができる．しかし，中央集権的な制御機構に依存する動的再構成デバイスでは，(b) のように単一チップ上で複数のタスクを並列動作させるような応用は望めず，プロセス技術が進歩してより多くの演算資源が単一チップ上に

実装できるようになった場合，中央の制御機構が性能上のボトルネックになると危惧される．我々は，プロセス技術が進歩し，ALU のような粗粒度の演算器が 1 チップ上に多数実装できるようになったとき，外部からの制御によらないデバイス内での自己再構成が可能な粗粒度デバイスが有効であると考えている．

しかし，粗粒度の再構成デバイスは，ワード単位の並列化可能な処理を空間的に展開して高速に実行できるという長所を有する一方，複雑な制御を伴う逐次的な処理を少ない資源で効率よく実行することが難しいという短所がある．本研究が狙いとする自己再構成の制御も粗粒度再構成デバイスでの実現が非効率な種類の処理であり，逐次処理を効率よく実現するための機構が必要と考えられる．プロセッサ類は逐次処理を効率よく処理できるため，作り込みのプロセッサを持たせた再構成デバイス [7] も提案・利用されている．しかし，対象とするアプリケーションの逐次処理・並列処理の割合によっては未使用の回路資源が大量に発生してしまい，面積効率が悪くなる可能性がある上，分散制御型の自己再構成という点からも，プロセッサの位置が固定されるしまうより自由に配置できることが望まれる．

そこで本研究では，4 個の ALU などからなるタイルと呼ばれる基本要素がアレイ状に配列した構造をもつ粗粒度再構成デバイスを提案する．各タイルはプロセッサモードまたは ALU アレイモードのいずれかを選択して使用できるため，アプリケーションに応じてタイルアレイ中のプロセッサとハードウェアエンジンの比率を変えることが可能である．また，タイル中の ALU を始めとする多くの回路資源はプロセッサモードと ALU アレイモードの両方で有効に活用されており，面積効率に優れている．

以下，本稿では，2 章で提案アーキテクチャに必要な要件を議論し，3 章でアーキテクチャの概要を説明する．4 章で設計を行い，デバイスの利用例として 2 次元 DCT をマッピングする．5 章でまとめと今後の課題とする．

2. アーキテクチャの構成検討

2.1 全体構成

提案デバイスは，先述したように (1) 並列処理・逐次処理の割合が異なる様々なアプリケーション群を実現するときに，作り込みでプロセッサを持つ再構成デバイスより面積効率よく実現し，(2) デバイスが自己再構成を行う時に，任意の位置にプロセッサや ALU アレイを配置するために，タイルと呼ばれる基本単位からなる均質なアレイアーキテクチャとする．

タイルは内部に，演算要素となる ALU 群，配線要素となるセレクタ群，インストラクションメモリ，データメモリなどとして利用されるメモリ・レジスタ群と制御用回路を持つものとする．タイルはこれらの回路要素を組み替えることで，プロセッサにも ALU アレイにもなるとする．ALU アレイとして利用されている複数のタイルは互いに接続され，大きな 1 つの ALU アレイとしても利用可能なものとする．

2.2 ALU アレイのアーキテクチャ

本デバイスにおける ALU アレイで主に対象とする処理は，ワード単位の並列化可能処理である．例としてフィルタ処理，行列演算，展開されたループ処理などが挙げられる．これらの処理，特にフィルタ系の処理や行列演算には，積和演算は必須の演算であり，少なくとも，ALU アレイのデータパス上に積和演算が無駄なくマッピングできる必要がある．

上述以外のアプリケーションについても，より ALU の利用効率が高くマッピングできることが望ましい．しかし，ALU の利用効率を上げるためには，その分，配線の自由度を高くする必要がある．配線の自由度は高ければ高いほど，回路規模や動作速度にオーバヘッドとして現れる．オーバヘッドを減らすためにも，ALU アレイのデータパスは，プロセッサ時のデータパスと共通化しつつ，必要最小限のセレクタの追加で ALU アレイとして必要な配線の自由度を持たせたい．ここで，細かいループ，分岐を含む処理やフィードバック制御はプロセッサが実行するということを考えると，ALU アレイにおけるデータの流れは一方通行であることが多いと考えられる．そこで本デバイスの ALU アレイのデータパス部分は，ALU のカスケード接続を基本とし，隣接するタイルとデータを入れ替えつつ，パイプライン的に処理が進む構成とする．この構成は，パイプライン化されたプロセッサのデータパス部分と共通化できるとも期待できる．

2.3 プロセッサのアーキテクチャ

本デバイス上に実現されるプロセッサで主に対象とする処理は，複雑な制御処理を伴う逐次処理である．そのため，プロセッサのアーキテクチャとしては，ループや分岐を得意とし，ハードウェアでは扱いづらいサブルーチン呼び出しもできる命令セットを持つことが好ましい．

また，前項での検討を踏まえて，プロセッサはパイプライン化されているものを考える．本デバイスでは，DLX アーキテクチャ [8] にならい，IF, ID, EX, MEM, WB の 5 段構成とする．

2.4 プロセッサと ALU アレイの一体化

タイルはその内部に持つ ALU 群を組み替えることでプロセッサとしてのデータパス，ALU アレイとしてのデータパスを構成する．このタイル中の ALU 群について検討する．粗粒度動的再構成デバイスの多くは，乗算等の可能な ALU と，加減算のみが可能な ALU を 1:2〜1:3 程度の比率で持っている [9]．提案するデバイスではこの比率を 1:3 と仮決めし，乗算等の可能な ALU 1 個とその他の ALU 3 個を 1 タイルに持たせることとする．プロセッサモードの時も，これら 4 個の ALU が有効に活用されるようアーキテクチャを設計する．

タイルの制御部分では，プロセッサモード時は ID〜WB ステージに対応する 4 つのインストラクションレジスタ（IR）が必要であり，ALU アレイモード時も，セレクタや ALU 等を制御するための構成情報メモリが必要となる．そこで ALU アレイモード時には，4 つの IR を構成情報メモリに流用して回路資源の有効活用を図る．

3. 提案アーキテクチャ

3.1 概　　要

本デバイスは，タイルと呼ばれる要素の 2 次元アレイ構造を持っている．各タイルは，4 つの 16bit 整数 ALU，24bit 語長のインストラクションメモリ (IM) および 16bit 語長のデータメモリ (DM) を内蔵する．各タイルは，プロセッサモードでは 16bit 整数プロセッサとして，ALU アレイモードでは ALU ベースのハードウェアエンジンとして使用することが可能で

ある．

なお，今回はALUアレイとプロセッサのデータパスの共通化の実現可能性を明らかにする目的で設計を行った．このため，自己再構成で必要となるタイル同士が相互に再構成やモード切り替えを制御し合う機能は盛り込んでおらず，全タイルがデバイス外部から直接制御を受ける仕様となっている．この機能の実現は，今後の研究で取り組む．

3.2 タイル

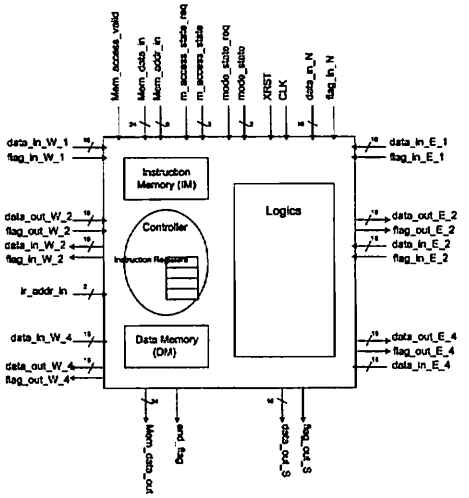

図1 タイルブロック図

図1にタイルのブロック図を示す．隣接するタイルと接続されるデータ線，外部から各種モードを制御したりメモリを読み書きするための信号線，タイルが演算終了を外部に知らせるフラグなどの入出力を持つ．表1にタイルが持つ動作モード，表2にタイルがもつメモリアクセスモードを示す．動作モードとメモリアクセスモードは基本的に独立して設定される．これらのモード選択はタイル外部からの制御信号によって行われる．タイルの動作中にも外部からIMとDMの読み書きが可能であり，IM，DMをダブルバッファのように利用できる．

あるタイルをプロセッサとして動作させたい場合，まずIM_writeおよびDM_writeモードを用いて当該タイルのIM，DMに予め必要なプログラム，データを書き込み，次にタイルの動作モードをプロセッサモードに遷移させれば，IMの0番地から順にプログラムが実行される．ALUアレイとして動作させたい場合は，まずIM_writeモードを用いて予め構成情報4語をIMに書き込み，次にIM_copyモードを用いてこれをIRへコピーし，最後にタイルの動作モードをALUアレイモードに遷移させればよい．

3.3 プロセッサモード

プロセッサモード時は16bitの5段パイプラインのRISCプロセッサとして動作する．プロセッサモード時のブロック図を図2に示す．本プロセッサでは，プログラムカウンタ用の加算器としてのALUとEXステージにおけるデータ処理用のALUとは別に，データメモリのアドレス生成用のALUを用意している．さらに，データ処理用のALUも，乗算器付きのALUとバレルシフタ付きのALUの2段カスケードとすることで，先述した4つのALUを有効活用している．また，本プロセッサでは，分岐のオーバヘッドを少しでも削減するため，全ての分岐系命令はIDステージで実行される．またデータハザード対策としてフォワーディングも実装されている．

表1 タイルの動作モード

プロセッサモード	IMに予め格納された命令列に従ってプロセッサとして動作
ALUアレイモード	IRに予め格納された構成情報に従ってALUアレイとして動作
ホールドモード	レジスタの値が維持され，演算が一時停止する
停止モード	パイプラインレジスタの値がリセットされる．タイルの初期モードでもある

表2 タイルのメモリアクセスモード

DM_write	外部からDMにデータを書き込むためのモード
DM_read	外部からDMのデータを読み出すためのモード
IM_write	IMに命令列あるいは構成情報を書き込むためのモード
IM_read	IMの内容を読み出すためのモード
IM_copy	IMからIRへ構成情報をコピーするためのモード

プログラマが利用することになるメモリ，レジスタを表3に示す．プログラムカウンタ(PC)とアドレスレジスタ(AR)はALUアレイモード時にALUカスケードのパイプラインレジスタとしても動作するため16bit分が用意されている．プロセッサモードでは下位9bitがそれぞれIM，DMへのアドレスとして使用される．

表4に本プロセッサの命令セット，図3に命令語の形式を示す．表4において，Rdは第3フィールド，Rsは第4フィールドにあたるレジスタ，immediateが第5フィールドの即値である．R7は汎用レジスタ7番であり，サブルーチン呼び出し命令であるCALL命令においてPCの保存先として指定されている．命令長は21bitであり，IMの下位21bitが割り当てられる．IMが24bitとなっているのは使用しているSRAMマクロの制約のためである．第1フィールドが「0」の場合，その命令語はプロセッサのインストラクションとして扱われ，「1」の場合はNOP命令として扱われる．

表3 内部メモリ/レジスタ

メモリ/レジスタ名	ワード幅
インストラクションメモリ	24bit × 512words
データメモリ	16bit × 512words
プログラムカウンタ	16bit
アドレスレジスタ	16bit
汎用レジスタ	16bit × 8words

3.4 ALUアレイモード

ALUアレイモード時のブロック図を図4に，ALUアレイモード同士の隣接タイルの接続を図5に示す．図2と図4にお

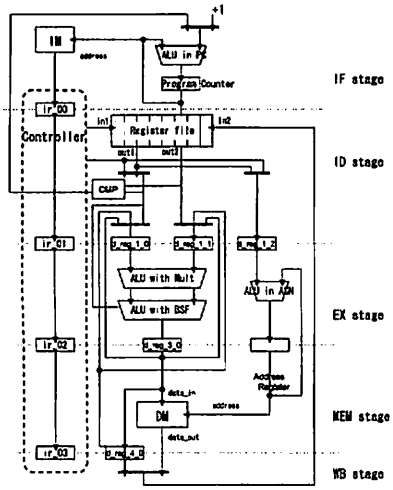

図 2　CPU ブロック図

第1フィールド:プロセッサ時の命令(0) or ALUアレイ時の構成情報(1)

第2フィールド:オペコード

第3フィールド:ディスティネーションレジスタ 兼 ソースレジスタ1

第4フィールド:ソースレジスタ2

第5フィールド:即値

図 3　命令語の形式

いて，共用化されている要素回路やレジスタには同じ名前が付いている．データパスは基本的に上から下への一方通行となっており，1 段目と 2 段目の ALU の前に両サイドのタイルとのデータの入れ替えを行うためのセレクタ群が設けられている．1 段目の ALU を簡単な前処理用，2，3 段目の ALU をメインの乗算などの処理用，4 段目の ALU を簡単な後処理かアキュムレーション用に利用することを想定しており，積和演算が無駄なくマッピングできるような構成としている．

DM へのデータ入力は当該タイルと両サイドの ALU カスケードの出力を受け付けることができる．データメモリへのアドレス生成のために簡単なアドレス生成器 (図 4 における Address Generator) が設けられている．データメモリへのアドレスはアドレス生成器からだけではなく，当該タイルの ALU カスケードの出力も受け付けられ，アドレス生成器では生成できない複雑なアドレスも扱うことができる．

表 5 に各 ALU が持つ命令を示す．算術演算とは加減算，論理演算とは AND，OR，XOR，NOT である．

3.5　2 つのモード間の回路資源の共用

プロセッサモードと ALU アレイモードでは，多くの回路資源を共用している．

3.5.1　ALU

演算資源としての ALU 群は，プロセッサモード，ALU アレイモード双方で利用されている．ALU アレイモードにおける

表 4　命令セット

命令語	機能
LD	AR = AR + immediate ; Rd = DM(AR)
ST	AR = AR + immediate ; DM(AR) = Rd
SETH	Rd[16:8] = immediate
SETL	Rd[7:0] = immediate
ARSET	AR = Rd
ARSTI	AR[7:0] = immediate
ARADD	AR = AR + Rd
JUMP	PC = {00000000, immediate}
JUMPR	PC = Rs
BEQZ	if (Rd==0) PC = Rs
BP	if (Rd >0) PC = Rs
BM	if (Rd <0) PC = Rs
CALL	R7 = PC; PC = {00000000, immediate}
CALLR	R7 = PC; PC = Rs
RET	PC = R7
HOLD	終了フラグを立てホールドモードになる
ADD	Rd = Rd + Rs
ADDI	Rd = Rd + immediate
ADDST	AR = AR + immediate ; Rd = Rd + Rs ; DM(AR) = Rd + Rs
ADIST	AR = AR + Rs ; Rd = Rd + immediate; DM(AR) = Rd + immediate
SUB	Rd = Rs − Rd
SUBI	Rd = immediate − Rd
SUBST	AR = AR + immediate ; Rd = Rs − Rd ; DM(AR) = Rs − Rd
SBIST	AR = AR + Rs ; Rd = immediate − Rd ; DM(AR) = immediate − Rd
MUL	Rd = Rd * Rs >>immediate[4:0]
SHTR	Rd = Rd >>immediate[3:0]
SHTRR	Rd = Rd >>Rs[3:0]
SHTL	Rd = Rd <<immediate[3:0]
SHTLR	Rd = Rd <<Rs[3:0]
AND	Rd = Rd & Rs
ANDST	AR = AR + immediate ; Rd = Rd & Rs ; DM(AR) = Rd & Rs
OR	Rd = Rd |Rs
ORST	AR = AR + immediate ; Rd = Rd |Rs ; DM(AR) = Rd |Rs
XOR	Rd = Rd ^Rs
XORST	AR = AR + immediate ; Rd = Rd ^Rs ; DM(AR) = Rd ^Rs
NOT	Rd = ~Rs
NOTST	AR = AR + immediate ; Rd = ~Rs ; DM(AR) = ~Rs

1 段目の ALU はプロセッサモードにおける PC 用の ALU，2，3 段目の ALU はデータ処理用の ALU，4 段目の ALU はアドレス生成用の ALU となっている．

3.5.2　レジスタ・メモリ

プロセッサモード時の 4 個のインストラクションレジスタ (IR) は，先の検討の通り，ALU アレイモード時には構成情報レジスタとして利用している．ALU アレイモード時の 1 タイルあたりの構成情報が 78bit となったため，図 3 のようにプロ

表 5　各 ALU が持つ演算

ALU 名	算術演算, NOP	論理演算	大小比較	ビットシフト	セレクタ
ALU_in_PC	○	○	○	○	○
ALU_with_MUL	乗算命令あり	○	×	×	×
ALU_with_BSF	○	○	○	バレルシフト命令あり	○
ALU_in_AGN	○	○	○	○	○

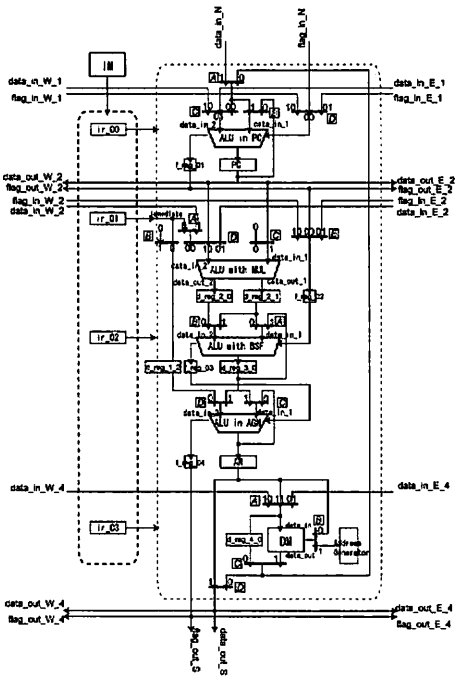

図 4　ALU アレイ内部ブロック図

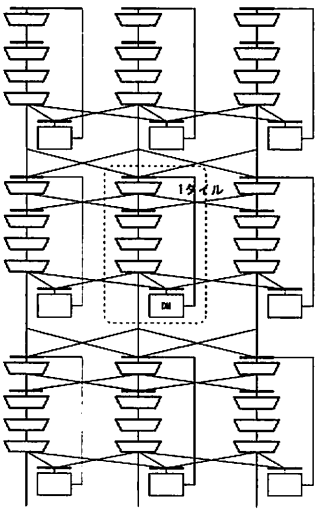

図 5　ALU アレイ接続関係

セッサモード時の命令語長（最上位の判定ビットをのぞく）を 20bit として命令セットアーキテクチャを設計した．これにより，いずれの動作モードでも IR のほぼ全ビットが有効活用されている．

各 ALU をつなぐデータパス部分に含まれるパイプラインレジスタについても，両モードでの共有を行っている．PC や AR などがそれにあたる．

DM，IM も共有化されており，ALU アレイモード時には，DM はデータの読み出し，または保存先メモリとして，IM はコンテキストメモリとして利用される．

3.5.3　配線資源

各 ALU をつなぐデータパス部分ではセレクタ群についても，レジスタと同様，共有化が行われている．しかし，ALU アレイモード時に隣接タイルとデータの交換を行うための配線リソースなどで，両モードで共有できていないセレクタが存在しており，両モード一体化時と単モードと比較した場合のオーバヘッドとなっている．

4.　設計とマッピング例

4.1　設　　計

前章で設計したアーキテクチャを Verilog-HDL で記述し，論理合成を行った．使用プロセスは Rohm 0.18μm，論理合成ツールは Synopsys Design Compiler である．表 6 に 1 タイルあたりの論理合成時の回路規模を示す．動作周波数は 125MHz となっている．この周波数は使用している SRAM マクロの動作速度に制限されている．プロセッサモードのみのタイル，ALU アレイモードのみのタイルとは，それぞれ mode_state 信号をプロセッサモード，ALU アレイモードで固定した状態で合成を行った場合の結果である．

回路資源の共用化をすることで，作り込みのプロセッサと ALU アレイで同等の機能と持たせたタイルと比べ，プロセッサと ALU アレイが一体化されたタイルを，ロジックが 58%，レジスタが 54%で実現できた．

4.2　マッピング例

本デバイスの利用例として，画像処理でよく利用される 8×8 点の 2 次元 DCT を ALU アレイモード上にマッピングした．本例では全てのデータ幅を 16bit と仮定している．

マッピングの概要を図 6 に示す．本例では，1 次元 DCT としてチェンの高速 DCT アルゴリズム [10] を用いている．8 点 DCT の場合，最大 8 並列で積和演算が実現できる．本デバイスでは隣接タイル間でしかデータのやり取りができないという配線リソースの制約があるため，今回は 2 並列での実装とした．

図 7 に動作のタイミングチャートを示す．この回路は図に示されるようにパイプライン動作する．まず，1 のデータ入力で，8×8=64 ワードの 1 ブロックのデータが 2 つの DM へそれぞれ 32 ワードずつ入力される．2 の 1 次元 DCT では入力されたデータが 1 ワードづつ読み出され，DCT 係数との積和演算が行われる．DCT1 ワードあたり 4 回の乗算が必要であるた

表 6 回路規模

	ロジック部分	レジスタ部分	SRAM セル部分
両モード一体化したタイル	7,900 gates (92,000μm^2)	1,500 gates (18,000μm^2)	- gate (427,000μm^2)
プロセッサモードのみのタイル (a)	6,900 gates (81,000μm^2)	1,400 gates (16,000μm^2)	- gate (427,000μm^2)
ALU アレイモードのみのタイル (b)	6,900 gates (81,000μm^2)	1,400 gates (16,000μm^2)	- gate (427,000μm^2)
(a) + (b)	13700 gates (162,000μm^2)	2,800 gates (32,000μm^2)	- gate (854,000μm^2)

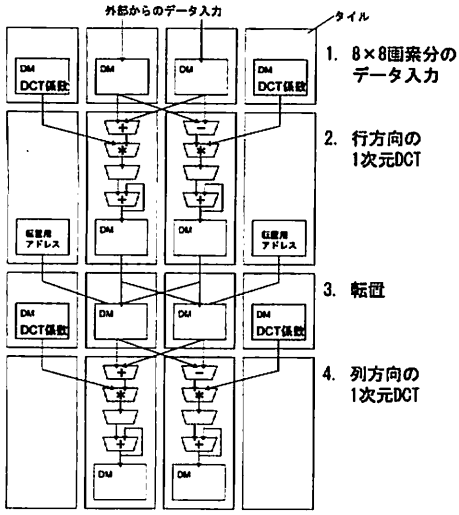

図 6 ALU アレイモードへの 2 次元 DCT マッピング例

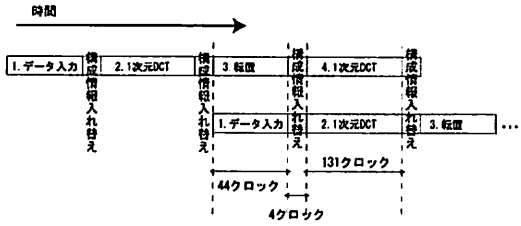

図 7 2 次元 DCT の動作のタイミングチャート

め，2 では 4×32 クロックに，タイル内のパイプラインを詰めるためのクロック 3 クロック分が加算された 131 クロックが必要となる．次に 3 の転置でデータを入れ替える．データの入出力に 32 クロック必要なだけでなく，一度タイルの再構成を挟むので，構成情報の入れ替えに 4 クロック，タイルのレイテンシが 4 クロックが 2 回分となるため，必要となるクロック数を 44 と見積っている．4 の 1 次元 DCT は，2 の 1 次元 DCT と同じである．動作周波数は 125MHz であるから，レイテンシは 366 クロック=3.3μs となり，スループットは 680k ブロック/s となる．

5. まとめと今後の課題

本稿では，プロセッサと ALU アレイの 2 つの動作モードをもつタイルのアレイからなる粗粒度再構成デバイスの提案し，アーキテクチャの検討，設計を行った．また，デバイスの利用例として，2 次元 DCT のマッピング例を示した．設計では，回路資源の共有をすることで作り込みのプロセッサと ALU アレイで実現した場合と比べ，ロジックが 58%，レジスタが 54%で機能が一体化できた．

今後の課題として，(1) 提案デバイスの性能見積もりと，(2) 自己再構成のための拡張が挙げられる．(1) については，プロセッサと ALU アレイを一定割合で混合した再構成デバイス，商用 DSP 類などを対象に性能比較を行い，提案デバイスの優位性を示す．本稿では，アプリケーション例として 2 次元 DCT を利用したが，最終的には DCT のような並列化可能処理と逐次処理が混在している画像コーデックのような処理を対象とする．(2) については，本設計では全てのタイルがデバイス外部から直接制御されているという前提で設計を行ったが，いずれは 1 チップ上に無数のタイルが実装され互いに制御しあうことを考えている．今後，この機構の検討と設計を行っていく．

謝辞 本研究は一部，日本学術振興会科学研究費補助金基盤研究 (B)17300016 による．また本研究の論理合成およびチップ設計は，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，シノプシス株式会社ならびにローム株式会社の協力で行われたものである．

文 献

[1] 小栗, 伊藤, 塩澤, 永見, 小西, 中村："布線論理による汎用計算機構", 第 11 回回路とシステム (軽井沢) ワークショップ, pp. 193–198 (1998).

[2] K. Nagami, K. Oguri, T. Shiozawa, H. Ito and R. Konishi: "Plastic Cell Architecture: A Scalable Device Architecture for General-Purpose Reconfigurable Computing", IEICE Trans. on Electronics, **E81-C**, 9, pp. 1431–1437 (1998).

[3] N. J. Macias: "The PIG Paradigm: The Design and Use of a Massively Parallel Fine Grained Self-Reconfigurable Infinitely Scalable Architecture", Proc. of the First NASA/DoD Workshop on Evolvable Hardware, pp. 175–180 (1999).

[4] L. J. K. Durbeck and N. J. Macias: "The Cell Matrix: An Architecture for Nanocomputing", 8th Foresight Conference on Molecular Nanotechnology, pp. 217–230 (2000).

[5] 佐藤："アイピーフレックスのリコンフィギュラブ・プロセッサ・デバイス - DAP/DNA ダイナミック・リコンフィギュラブル・プロセッサの概要", 第 1 回リコンフィギャラブルシステム研究会, pp. 165–172 (2003).

[6] T. Awashima: "Dynamically Reconfigurable Processor: DRP and its C-level Design Tool", Proc. of the International Symposium on Advanced Reconfigurable Systems 2005, pp. 159–172 (2005).

[7] XILINX: "Virtex-II Pro Platform FPGAs : Advance Product Specification" (2003).

[8] J. L. HENNESSY: Computer Architecture A Quantitative Approach (1996).

[9] 津野田, 高田, 秋田, 田中, 佐藤, 伊藤："ディジタルメディア向け再構成型プロセッサ FE-GA の概要", 電子情報通信学会技術研究報告, No. RECONF2005-65, pp. 37–41 (2005).

[10] W. H. Chen, C. H. Smith and S. C. Fralick: "A Fast Computational Algorithm for the Discrete Cosign Transform", IEEE Tran. on Commun, **COM-25**, 9, pp. 1004–1009 (1977).