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.

リコンフィギャラブルプロセッサ上でのブロック暗号 RC6 の実装

長谷川揚平† 山田 裕† 出口 勝昭† 安生健一朗†† 粟島 亨†††
天野 英晴†

† 慶應義塾大学大学院理工学研究科 〒 223–8522 横浜市港北区日吉 3–14–1
†† NEC エレクトロニクス株式会社 〒 211-8668 川崎市中原区下沼部 1753
††† NEC マルチメディア研究所 〒 211-8668 川崎市中原区下沼部 1753
E-mail: †drp@am.ics.keio.ac.jp, ††k.anjo@necel.com, †††awash@ccm.cl.nec.co.jp

あらまし NEC が 2002 年に発表した Dynamically Reconfigurable Processor(DRP) は, Processing Element(PE) のアレイからなる Tile の集合でデータパスを構成する. DRP は最大 16 コンテキスト分の構成情報を内部に保持し, これを一括して, または Tile 単位で次々と切替える事ができる. 本稿では, この DRP を用いて, 共通鍵ブロック暗号 RC6 を処理する RC6 プロセッサを設計, 実装し評価を行った. RC6 は, データサイズ, ラウンド数, 鍵サイズがともに可変なスケーラブルなブロック暗号である. 今回実装した RC6 プロセッサは外部から読み込んだ 8 つのブロックを non-feedback モードで同時に並列処理することができる. 組み込み CPU および DSP 上のソフトウェアと比較した結果, それに勝るスループットを達成する事ができた.

キーワード リコンフィギャラブルプロセッサ, マルチコンテキスト機能, DRP, ブロック暗号 RC6

The Implementation of The Block Cipher RC6 on The Reconfigurable Processor

Yohei HASEGAWA†, Yutaka YAMADA†, Katsuaki DEGUCHI†, Kenichiro ANJO††, Toru AWASHIMA†††, and Hideharu AMANO†

† Graduate School of Science and Technology, Keio University
3–14–1 Hiyoshi, Yokohama, Kanagawa, 223-8522 Japan
†† NEC Electronics Corporation 1753 Shimo-Numabe, Nakahara, Kawasaki, 211-8668, Japan
††† NEC Multimedia Research Laboratories 1753 Shimo-Numabe, Nakahara, Kawasaki, 211-8668, Japan
E-mail: †drp@am.ics.keio.ac.jp, ††k.anjo@necel.com, †††awash@ccm.cl.nec.co.jp

Abstract NEC's Dynamically Reconfigurable Processor(DRP) is a multi-context reconfigurable device consisting of eight individually reconfigurable units called Tile. Each Tile consists of 64 Processing Elements(PEs) and the set of Tiles can form a particular datapath. Datapath configuration mapped to each tile can be selected from on-chip repository of sixteen circuit configurations, or contexts. The context switching can be done with a clock cycle. In this work, the block cipher RC6, which is scalable in its block size, number of round and key size, was implemented on DRP. This RC6 Processor on DRP can encrypt and decrypt eight blocks simultaneously in non-feedback mode. Evaluation results show that the throughput of the encryption and decryption of the RC6 implemented on DRP outperform that with the software on MIPS64(500MHz) and DSP TMS320C6713(225MHz).

Key words reconfigurable processor, multi-context functionality, DRP, block cipher RC6

1 はじめに

近年, コンテキストとよばれる複数の構成情報を内部に保持し, 高速な動的再構成が可能なプロセッサのアレイ構造をもつリコンフィギャラブルプロセッサが本格的に市場に登場してきている. リコンフィギャラブルプロセッサは, 8bit-32bit の演算器, シフタ, レジスタ, 分散メモリからなる処理ユニットを多数搭載し, これらの処理ユニットの構成および接続を短時間で

切替える事により, 専用ハードウェアにせまる高速性とソフトウェアにせまる柔軟性を優れた面積効率で実現する.

これらのマルチコンテキストリコンフィギャラブルデバイスでは, ハードウェア資源の不足する携帯端末上で動作させながら状況に応じてハードウェア構成を切替えること [4] や, 通信制御用の機器の機能を動的に変更すること [5] が可能となる. こうして, MPEG や JPEG などのコード圧縮および復号, DES, AES などの暗号化および復号, Viterbi などのエラー修正用のコードなどを含むメディア処理, 通信制御などの応用分野が有望と考えられる.

特に近年では, 暗号化方式は国際的にも国内的にも標準化の動きが進んでいる. 標準化されたものは ASIC 化で高速な処理を達成する事ができるが, 非標準的なものや, これから標準化されるものでは, ASIC 化することでスケールメリットを出しにくいので, プログラマブルでかつ高速に処理する事は重要である.

NEC が 2002 年に発表した DRP のプロトタイプである DRP-1 [1] は, チップ内に最大 16 のコンテキストを保持し, それらを 1 クロックで次々と切替えることができるマルチコンテキストリコンフィギャラブルデバイスである. 本稿ではこの DRP 上にブロック暗号 RC6 [9] を処理する RC6 プロセッサを設計, 実装した. RC6 は, DES に代わる次世代暗号化標準規格 (AES) の最終候補にあがったものであり, ブロック長, ラウンド数, 鍵サイズがともに可変なスケーラブルなブロック暗号である.

まず, 2 節では DRP の詳細を述べる. 3 節では, ブロック暗号 RC6 の詳細を紹介し, 4 節では DRP 上への RC6 の実装を報告する. 最後に 5 節で評価結果を述べる.

2 Dynamically Reconfigurable Processor(DRP)

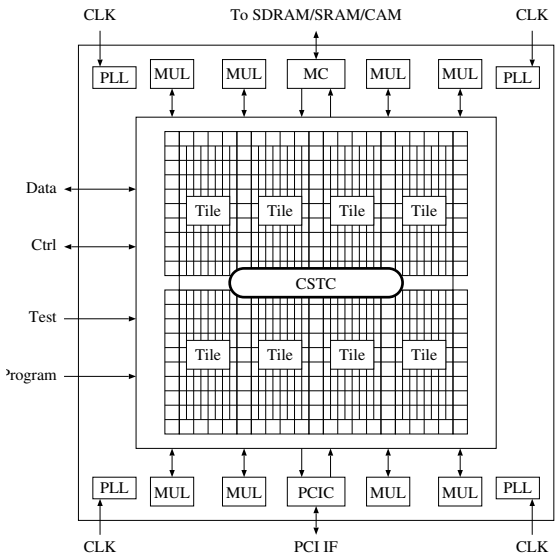

図 1　DRP-1 の構造

DRP [1] は, 粗粒度のマルチコンテキストデバイスであり, プロトタイプチップである DRP-1 の全体の構造は図 1 に示す構造となっている.

DRP-1 は Core の周辺部に 32 ビットの乗算器を 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 の Core 部分には Tile と呼ばれるリコンフィギャラブルユニットが 4×2 個配置されており, 中央にチップ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.

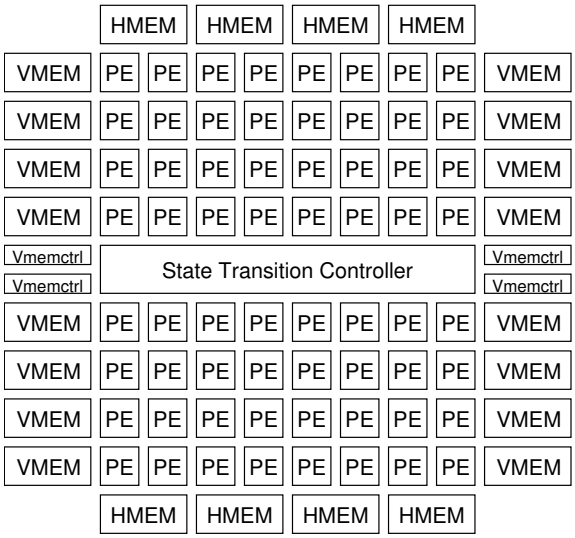

図 2　Tile の構造

各 Tile は図 2 に示す通り, 8×8 の Processing Element (PE) アレイ, 状態制御を行なうシーケンサである State Transition Controller (STC) から構成される. また, 8bit×256 エントリのメモリ 8 セットとこれらを制御するメモリコントローラ 2 セットを両側に持ち, 8bit×8192 エントリのメモリ 4 セットとメモリコントローラを Tile の上部もしくは下部に持つ. DRP 全体では 8bit×256 エントリのメモリを合計 80 セット, 8bit×8192 エントリのメモリを合計 32 セット持つ.

各 PE は図 3 に示す通り, 8bit の ALU, シフトやデータ制御, 簡単な論理演算を行なう Data Management Unit (DMU), 8bit の Flip Flop, レジスタファイルから構成される. また, コンフィギュレーデ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 は, 1 クロックでコンテキスト切り替えが可能なマルチコンテキストデバイスで, Tile 単位の部分再構成が可能である. また DRP は 8bit の PE を構成要素としたリコンフィギャブルなプロセッサである. その特徴をまとめると以下のようになる.

マルチコンテキスト機能: DRP は内部のメモリに最大 16 コンテキスト分の情報を蓄えることができ, 最大で 5 コンテキストの中から次のコンテキストを選択し, 動的に再構成をするこ

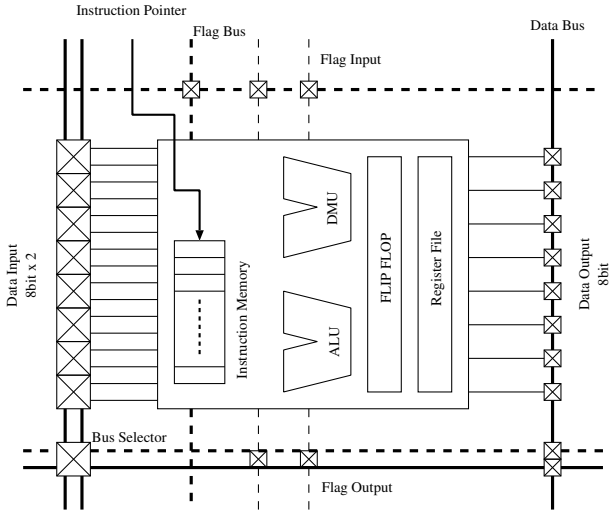

図 3　PE の構造

とができる. 次のコンテキストの選択は STC に信号を送ることで行なう. さらに, 動作中に, 切り替えの対象外のコンテキストに対してコンフィギュレーション ロードが可能である. この特徴により, 仮想ハードウェアや動的適応ハードウェアの効率的な実現が可能となる.

プロセッサアレイ構成: DRP は FPGA 等の LUT によりロジックを実現する細粒度構成とは異なり, データパスを再構成するデバイスである. このため, 8bit を基本単位とした効率的なデータフローの制御および多桁演算の高速実装が可能である. LUT を書き換えるのに比べて高速な再構成が可能であり, 全く無駄なクロックなしに, 切り替えと同時に動作を行なうことができる. また, PE がチップ全体に配置されているため, 並列処理を効率的に行うことが可能である.

パーシャルリコンフィギャラビリティ: DRP はコンテキスト切り替えは Tile 単位で行うことができ, コンフィギュレーション データのロードは PE 単位で行なうことができる. これにより, Tile ごとに異なるコンテキスト構成を取ることができる.

メモリ拡散配置型のアーキテクチャ: DRP は各 PE が Flip Flop およびレジスタファイルを独立に持ち, Tile の両側に 8bit×256 エントリの小規模なメモリを 8 セット, Tile の上側もしくは下側に 8bit×8192 エントリのメモリを 4 セットずつ持つ. これらの記憶要素の分散により, パイプライン構成, データ駆動型制御, フィードバック構成が簡単かつ高速に実現することができる.

3　ブロック暗号 RC6

RC6 [9] は, Ronald Rivest らによって 1998 年に開発された共通鍵暗号方式である. 彼らによって以前に開発された RC5 [10] の後継にあたる暗号化方式であり, RC5 において指摘された問題点が補強されている. RC6 は米国商務省標準技術局 (National Institute of Standards and Technology: NIST) によって選定される次世代暗号標準規格 (Advanced Encryption Standard: AES) の最終候補の一つになったが, AES に採用されたのは Rijndael [13] で RC6 は結局採用されなかった.

AES には Rijndael が採用されたが, RC6 は高いスケーラビリティと安全性をもった暗号化方式であり, さらに最近の暗号標準化動向に伴い, ISO/IEC JTC 1/SC27 [11] の暗号化標準規格である NP 18033 や, 日本の情報処理振興事業協会による Cryptrec [12] の候補にもあがっている.

RC6 のバージョンは, RC6-$w/r/b$ で特定される. ここで, パラメータ w, r, b はそれぞれ, ワードサイズ (bits), ラウンド数, 鍵サイズ (bytes) である. 実際のターゲットでは, 128 ビットデータを 1 ブロックとして扱い, ラウンド数は 20 で 32 ビットワード 4 つを処理し暗号化および復号を行う. 鍵スケジュールでは, ユーザが供給する b バイトの鍵から $2r+4$ 個の w ビットワードが生成される. これらの値はラウンド鍵とよばれる拡張鍵で, アレイ $S[0, \cdots, 2r+3]$ に格納され, 暗号化, 復号の両方で使用される.

RC6-$w/r/b$ の処理では, 以下の 6 つの演算が含まれる;

- $a+b$　整数加算 (modulo 2^w),
- $a-b$　整数減算 (modulo 2^w),
- $a \oplus b$　2 つの w-bit ワードのビットワイズ排他的論理和,
- $a \times b$　整数乗算 (modulo 2^w)
- $a \lll b$　w-bit ワード a の左巡回シフト,
- $a \ggg b$　w-bit ワード a の右巡回シフト.

ただし, $a \lll b$ および $a \ggg b$ のシフト量は b の最下位 $\lg w$ bit で与えられる. 以下, 鍵スケジュール, 暗号化, 復号の各アルゴリズムの詳細を述べる.

3 1　鍵スケジュール

ユーザが与える鍵は b バイトである. RC6 では, ワード単位で処理を行うため, この b バイトの秘密鍵 $K[0], \cdots K[b-1]$ は, c 個の w ビットワードのアレイ $L[0], \cdots L[c-1]$ にコピーされる. ここで, $c = \lceil b/u \rceil, u = w/8$ である. これらの鍵はリトルエンディアンで L にロードされる. すなわち, 鍵の最初のバイトは, $L[0]$ の下位バイトに格納される事となる. ここで, L はあらかじめ 0 に初期化されているものとする.

次のステップは, 秘密鍵から $2r+4$ 個の w ビットワードのラウンド鍵を生成することである. このラウンド鍵は $2r+4$ 個の w ビットワードのアレイ $S[0], \cdots, S[2r+3]$ に格納される. アレイ S の初期化では, 以下の 2 つのワードサイズの定数 P_w, Q_w を用いる. これらの定数は Magic Constant と呼ばれ, それぞれ自然対数の底 ($e = 2.718281828549...$) と黄金比 ($\phi = 1.618033988749...$) から求められ, 以下で定義される.

$$P_w = \mathrm{Odd}((e-2)2^w),$$
$$Q_w = \mathrm{Odd}((\phi-1)2^w).$$

ここで, $\mathrm{Odd}(x)$ は x に最も近い奇数であり, $w = 32$ のケースでは,

$$P_{32} = b7e15163,$$
$$Q_{32} = 9e3779b9$$

である (ともに 16 進数).

具体的な鍵スケジュールの手続きは, S, L を用いてラウンド鍵を生成する事である. その手順は以下の通りである.

$S[0] = P_w$
for $i = 1$ **to** $2r + 3$ **do**
$S[i] = S[i-1] + Q_w$
$A = B = i = j = 0$
$v = 3 \times \max\{c, 2r + 4\}$
for $s = 1$ **to** v **do** {
$A = S[i] = (S[i] + A + B) \lll 3$
$B = L[j] = (L[j] + A + B) \lll (A + B)$
$i = (i + 1) \bmod (2r + 4)$
$j = (j + 1) \bmod c$ }

3.2 暗号化と復号

RC6 では, 初期状態では平文を, 暗号化終了の段階で暗号文をもつ 4 つの w ビットレジスタ A, B, C, D で作業を行う. 平文または暗号文の最初のバイトは A の最下位バイトに置かれ, 最後のバイトは D の最上位におかれる. 暗号化のアルゴリズムは以下の通りである.

$B = B + S[0]$
$D = D + S[1]$
for $i = 1$ **to** r **do** {
$t = (B \times (2B + 1)) \lll \lg w$
$u = (D \times (2D + 1)) \lll \lg w$
$A = ((A \oplus t) \lll u) + S[2i]$
$C = ((C \oplus u) \lll t) + S[2i + 1]$
$(A, B, C, D) = (B, C, D, A)$ }
$A = A + S[2r + 2]$
$C = C + S[2r + 3]$

一方, 復号のアルゴリズムは以下の通りである. 復号でも暗号化と同様に, 4 つの w ビットレジスタ A, B, C, D で作業を行い, それらのレジスタには初期状態では暗号文, 復号終了の段階では平文が格納される.

$C = C - S[2r + 3]$
$A = A - S[2r + 2]$
for $i = r$ **downto** 1 **do** {
$(A, B, C, D) = (D, A, B, C)$
$u = (D \times (2D + 1)) \lll \lg w$
$t = (B \times (2B + 1)) \lll \lg w$
$C = ((C - S[2i + 1]) \ggg t) \oplus u$
$A = ((A - S[2i]) \ggg u) \oplus t$ }
$D = D - S[1]$
$B = B - S[0]$

4 RC6 の DRP 上への実装

本稿では, 前節で示したブロック暗号 RC6 を処理する RC6 プロセッサを設計し, DRP 上に実装し評価を行った.

4.1 仕　　様

RC6-$w/r/b$ の各パラメータの制約は以下の通りである.

- データサイズは固定で $w = 32$[bits],
- ラウンド数は可変で $0 < r \leqq 255$,
- 鍵サイズは可変で $0 < b \leqq 32$[bytes].

以上のパラメータのもとで, 128 ビットのブロックを処理する.

今回実装した暗号化の操作モードは, non-feedback モードである Electronic Codebook(ECB) モードである. ECB モードは, 平文/暗号文を 128 ビットのブロックに分割して, それぞれのブロックを独立に暗号化/復号を行う方式である. ECB モードは, 秘密鍵が 1 つ以上のメッセージに使われる場合, 同一の暗号文は同一の平文となり, セキュリティの面で大きな欠点があるが, 複数のブロックを並列に処理ができるので, 高速に暗号化/復号を行うことができるという利点がある. 一方, CBC モードなどの feedback モードは, 平文を暗号化するときに直前の暗号文との排他的論理和をとってから暗号化する等の処理を行い, ブロックの内容が同じであっても, 出力結果が同じにならないようにしている. しかしながら, 直前の暗号文が必要となるので, 並列に処理を行う事ができない.

今回の実装では, DRP の並列性を利用するため, ECB モードを採用し 8 つのブロックを同時に並列処理する.

4.2 RC6 プロセッサの基本構成

今回実装した RC6 プロセッサの基本構成を図 4 に示す. RC6 プロセッサは, 鍵スケジューラ, 4 つの暗号化器, 4 つの復号器, 入力インタフェース, 出力インタフェースから構成される.

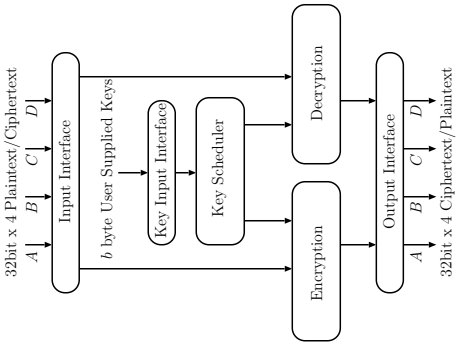

図 4　RC6 プロセッサの基本構成

暗号化の処理の流れは以下の通りである. 外部から入力されたメッセージは 128 ビットのブロックに分割され入力 FIFO に蓄えられる. 各暗号化器は, 入力 FIFO から平文ブロックを取り出し, r 回のラウンド処理を行う. このとき, 各ラウンド間にはデータ依存があるので, 1 つのブロックのラウンド処理が完全に終り結果がでるまでそのブロックは次のラウンドに進めない. さらに, 現在の DRP には乗算器が 8 個付属しているが, 各ラウンドのサイクルで必要な乗算は 2 つで, 1 サイクルで 1 ラウンドを処理するとすると, 4 並列までが限界となってしまう. そこでさらに並列性を高めるため, ラウンド処理を 2 つのステージに分割し, 2 つのステージ間で 2 つのブロックをパイプライン処理を行うことにした (図 5).

Pipeline 0 は積和演算と固定長巡回シフトを処理し, Pipeline 1 は可変長巡回シフトを処理する. この暗号化処理における Pipeline 0, 1 の回路構成を図 6,7 に示す.

図 5 に示すように Pipeline0 と Pipeline1 で, 2 つのブロッ

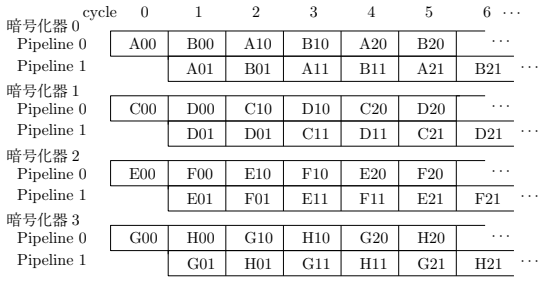

図 5　暗号化器, 復号器のパイプライン. A01 はブロック A の 0 ラウンド目のステージ 1 の処理を示す. 横軸は実行サイクルを表す.

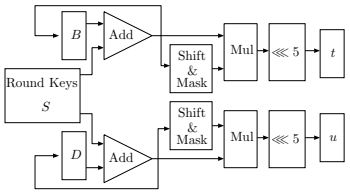

図 6　Pipeline0(暗号化)

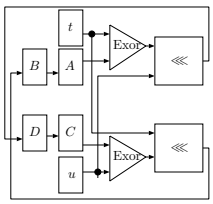

図 7　Pipeline1(暗号化)

クの処理をオーバラップさせていくことで, 乗算器を効率よく使用した.

こうして各暗号化器は入力 FIFO から 2 つの平文ブロックを取り出し, 暗号化処理を行い, 結果を出力 FIFO に格納する. このようにして, 4 つの処理部がそれぞれ 2 つのブロックをパイプライン処理する事で, 合計 8 つのブロックを並列に処理する事ができる. 復号も暗号化と同様の構成で処理を行う.

4 3　各コンテキストの概要

上述の RC6 プロセッサを DRP 上に実装するにあたって, 各処理を図 8 に示すように各コンテキストに分割した.

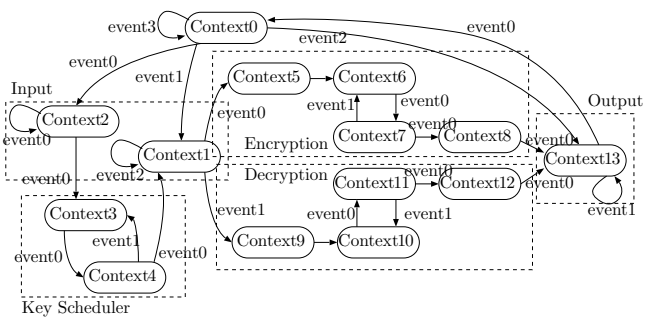

図 8　RC6 プロセッサのコンテキスト分割

各コンテキストの概要は以下の通りである.

Context0:　idle ステージ. メモリ, レジスタの初期化を行うと同時に, ユーザの入力する操作モードおよびラウンド数をデコードする. デコードが完了したら入力インタフェースまたは出力インタフェースに遷移する. ユーザからの要求がない場合は, 待機する.

Context1,2:　入力インターフェース. 鍵スケジュールを行う場合は, ユーザが指定する b バイトの秘密鍵を外部から読みだし Vmem で構成される c 個のワードサイズのアレイ L に格納する. また, 暗号化/復号を行う場合は, 平文/暗号文を最大 8 ブロックまで外部から読みだし入力 FIFO に格納する. 入力が終了するとーザが指定したモードに従い, 鍵スケジュール, 暗号化, 復号のいずれかの処理に遷移する.

Context3,4:　鍵スケジュール部. RC6 の鍵スケジュールを実行し, ラウンド鍵 S を生成し, 暗号化処理部または復号処理部に遷移する.

Context5,6,7,8:　暗号化処理部. ラウンド処理は主に乗算・固定長巡回シフトを処理する部分 (Pipeline 0) と, データ依存を含む可変長巡回シフトを処理する部分 (Pipeline 1) に分割される. 暗号化終了後は出力 FIFO に暗号文を格納し, 出力処理部に遷移する.

Context9,10,11,12:　復号処理部. 暗号化処理部と同様, ラウンド処理は主に乗算・固定長巡回シフトを処理する部分 (Pipeline 0) と, データ依存を含む可変長巡回シフトを処理する部分 (Pipeline 1) に分割される. 復号終了後は出力 FIFO に平文を格納し, 出力処理部に遷移する.

Context13:　出力インタフェース. 暗号化または復号されたメッセージを出力 FIFO から外部に出力する. すべてのメッセージを出力後は, idle に戻り待機する.

5　評　　価

全節で示した RC6 プロセッサを DRP のシミュレーション環境と自動配置配線ツールを用いて実装した. 以下にその評価結果を示す.

5 1　使用リソース

表 1 は, 各コンテキスト毎に必要な演算リソースの量をまとめたものである.

表 1　使用リソース

Context	Alu	Dmu	Ffu	Rfu	PE	Vmem	Hmem	Mul
0	5	17	27	5	27	0	0	0
1	27	147	137	3	147	0	0	0
2	8	5	0	3	8	8	0	0
3	20	9	24	2	24	8	0	0
4	32	10	25	2	32	8	0	0
5	32	72	72	2	72	4	0	8
6	162	137	136	2	162	4	0	8
7	164	137	136	3	164	4	0	8
8	32	2	40	0	40	4	0	0
9	99	73	168	2	168	4	0	8
10	163	137	160	2	163	4	0	8
11	161	136	160	2	161	4	0	8
12	128	64	168	0	168	4	0	0
13	10	116	138	16	138	4	0	0
MAX	**164**	**147**	**168**	**16**	**168**	**8**	**0**	**8**

特に注目すべきコンテキストは, 暗号化および復号のラウンド処理を行う Context6, 7, 11, 12 である. Alu の主な使用用途は 32 ビットの整数加算および整数減算であり, Dmu の主な仕様用途は 32 ビットの可変長および固定長巡回シフトおよび 32 ビットのセレクタである.

これらの処理に要したリソースは, 全 PE 数 (512) の約 33% であり, 配線に使用される部分を含めると約 60% に及び, 面積

効率の面では非常に効率良く使用されていると言える.

しかしながら, さらなる並列処理を実行するためには, 現在の DRP-1 では, 4 並列で 8 ブロック同時処理が可能であったが, 各ラウンド処理に必要な 2 つの乗算に対応して, 乗算器数が 16 個まで搭載されれば, さらに並列処理を行うことで性能は 2 倍程度まで向上すると考察される.

5.2 実行速度

表 2 は, 暗号化および復号のスループットを, 様々なプラットフォーム上で実行したものと比較したものである. 今回実装した RC6 プロセッサの最大動作遅延は 30874[psec] であり, 最大動作周波数は 32.4 MHz である.

表 2　暗号化および復号に対するソフトウェアとの性能比較

Platform	Mode	clocks/block	Mbits/sec
DRP(32.4MHz)	Encryption	5	829.4
	Decryption	5	829.4
MIPS64(500MHz)	Encryption	443	144.5
	Decryption	465	137.6
TI DSP C6713	Encryption	765	37.6
(225MHz)	Decryption	746	38.6

まず, DRP 上に実装した RC6 と同様, non-feedback モードで連続して処理するプログラムを組み込み CPU 上で実行した. この CPU での実行には 500MHz MIPS 64 アーキテクチャを用いた. この MIPS64(500MHz) は 4 issue in-order SuperScalar で, 各々 32KB のデータキャッシュと命令キャッシュ, 512 KB の L2 キャッシュをもつ. MIPS 用 gcc クロスコンパイラを用いてコンパイルしたものである. コンパイルオプションには -O3 を用いた. この評価プログラムは 10000 個の平文および暗号文ブロックを処理した際の最少クロック数からスループットを計測した.

次に, 同様のプログラムを DSP 上で実行したものと比較した. DSP には TEXAS INSTRUMENTS の TMS320C6713 を使用した. これは VLIW アーキテクチャを採用した浮動小数点 DSP で, 最大 225 MHz で動作する. 各々4KB の命令キャッシュとデータキャッシュ, 計 256KB の L2 キャッシュをもつ. 実行したプログラムは MIPS の評価で用いたものと同じ C コードを, 専用に提供されている開発環境 Code Composer Studio C6713 Version2.20.05 によりコンパイルしたものである. この評価も MIPS の場合と同様, 10000 個のブロックを処理した際の最少クロック数からスループットを計測した.

表 2 より, DRP 上に実装した RC6 プロセッサは, MIPS64 アーキテクチャ上のソフトウェアで実行した場合よりも約 6 倍のスループットを引き出す事ができた. また, DSP 上でのソフトウェアと比較した場合, DRP 上で実装した RC6 プロセッサは約 22 倍のスループットを達成する事ができた. それと同時に必要サイクル数もソフトウェアでの実行の場合よりもかなり少ないので, 消費電力を低く抑える事も可能である.

また, 動作周波数は配置配線の最適化を施すことで, 50MHz 程度までの動作が実現可能であると考えられる. また, さらなる高速化のためには, 最も遅延および面積の大きい可変長巡回シフトの最適化を施すことで, さらなる高速化が期待できる. これらの処理は細粒度のロジックで実装した方が遅延を抑える事が期待できるため, DRP の Tile の一部に細粒度のロジックを置く事で最適化できると思われる.

6 結　　論

本稿では, NEC 社のマルチコンテキストリコンフィギャラブルプロセッサである DRP 上に, ブロック暗号 RC6 を処理する RC6 プロセッサを設計, 実装し評価を行った. シミュレーションの結果より組み込み CPU や DSP 上で動作させたプログラムよりも高いスループットを達成する事ができた.

最近の暗号化方式の標準化動向からもわかるように, 今後特に組み込み分野での暗号化処理の必要性は高まる事が予想される. 本稿で示したように, 必要な性能を満足し, かつ高い柔軟性と面積効率, 低消費電力が期待できる事から, リコンフィギャラブルプロセッサ上で暗号化処理を行う事には大きな利点があるといえる.

謝　　辞

DRP およびその開発環境を提供して頂き, 本研究に対する多くのアドバイスをしてくださった NEC エレクトロニクスおよび NEC マルチメディア研究所の皆様に深く感謝致します.

文　　献

[1] M.Motomura. A Dynamically Reconfigurable Processor Architecture, Microprocessor Forum, Oct. 2002.

[2] Fujii, T., Furuta, K., Motomura, M., Nomura, M., Mizuno, M., Anjo, K., Wakabayashi, K., Hirota, Y., Nakazawa, Y., Ito, H. and Yamashina, M. A Dynamically Reconfigurable Logic Engine with a Multi-Context Multi-Mode Unified-Cell Architecture, *Proc. Intl. Solid-State Circuits Conf.*, pp. 360–361 (1999).

[3] X.-P. Ling, H. Amano. WASMII: A Data Driven Computer on a Virtual Hardware, *Proc. of the IEEE Symposium on FPGAs for Custom Computing Machines(FCCM'93)*, pp. 33–42, 1993.

[4] A. Alsolaim, J. Becker, M. Glesner, and J. Starzyk. Architecture and Application of a Dynamically Reconfigurable Hardware Array for Future Mobile Communication Systems, *Proc. of FCCM*, pp.205-214, 2000.

[5] G. J. M. Smit, P. J. M. Havinga, L. T. Smit, P. M. Heysters. Dynamic Reconfiguration in Mobile Systems, *Proc. of FPL2002*, pp.162-170, 2002.

[6] QuickSilver Technology, http://www.quicksilvertech.com/.

[7] IP FLEX DAP/DNA, http://www.ipflex.com/.

[8] S. Trimberger, D. Carberry, A. Johnson and J. Wong. A Time-Multiplexed FPGA *Proc. of the IEEE Symposium on FPGAs for Custom Computing Machines(FCCM'97)*, pp.22-28, (1997).

[9] R. Sidney R. Rivest, M. Robshaw and Y. L. Yin. The RC6 Block Cipher. in *The First Advanced Encryption Standard (AES) Candidate Conference*, June, 1998.

[10] R. Rivest. The RC5 Encryption Algorithm. In B. Preneel, editor, *Fast Software Encryption*, vol 1008 of *Lecture Note in Computer Science*, pages 86–95,1995. Springer Verlag.

[11] SO/IEC JTC 1/SC 27. http://www.din.de/ni/sc27/

[12] Cryptrec. http://www.ipa.go.jp/security/enc/CRYPTREC/

[13] John Daemen, Vincent Rijmen. AES Proposal: Rijndael, http://csrc.nist.gov/encryption/eas/rijndael/Rijndael.pdf, 1999.