計算機アーキテクチャ 135－4
（1999. 11. 26）

Reconfigurable Computing Systemの暗号処理への適用

山口晃由† 橋山智訓†† 大熊繁†

†名古屋大学大学院工学研究科電気工学専攻
〒464-8603 名古屋市千種区不老町
Tel:052-789-2777 Fax:052-789-3140
Email:teru@okuma.nuee.nagoya-u.ac.jp
††(財) 名古屋産業科学研究所

あらまし 近年、ユーザが再構成可能な論理回路素子であるFPGAの性能向上と、ハードウェア記述言語によりハードウェアの高速性と並列性、ソフトウェアの柔軟性を兼ね備えた計算機システムが構成可能になりつつある。この手法はリコンフィギャラブルコンピューティング(RC)とよばれる。RCでは、論理演算等のFPGA向きの処理を必要に応じて回路を再構成しながら計算を進めることで、汎用プロセッサと協調し高速かつ柔軟な演算を実現できる。

本研究では高速に回路書換えを行なうFPGAシステムを製作し、暗号処理を例に提案システムの性能を評価した。

キーワード
リコンフィギャラブルコンピューティング FPGA 動的再構成 暗号処理

A Study on Reconfigurable Computing System for Encryption

Teruyoshi Yamaguchi† Tomonori Hashiyama†† Shigeru Okuma†

†Dept. of Electrical Engineering Nagoya University
Furo-cho,Chikusa-ku,Nagoya 464-8603 Japan
Tel:+81-52-789-2777 Fax:+81-52-789-3140
Email:teru@okuma.nuee.nagoya-u.ac.jp
††Nagoya Industrial Science Research Institute

abstract This paper introduces an implementation of Reconfigurable Computing technique for encryption processing. The Reconfigurable Computing (RC) is capable of accelerating the computational processing using dynamic reconfiguration of Field Programmable Gate Arrays (FPGAs). By deviding the target ploblems into hardware and software appropriately, the computation time will become much faster.

The authors implemented RC onto FPGA system. To examine the feasibility of this system, they apply it to encryption processing.

keyword

Reconfigurable Computing FPGA Dynamic Reconfiguration Encryption Processing

1 はじめに

近年、高度な情報処理の需要、また実時間処理への要望は増加する一方である。情報処理の高速化には、一般的に専用 IC（ASIC:Application Specific IC）が用いられることが多いが、一度設計し実装された回路は、その用途以外では使用できず、些少の仕様変更にも柔軟に対応することができない。

近年、ユーザが自由に回路書き換え可能な論理回路素子である FPGA(Field Programmable Gate Array) や、PLD (Programmable Logic Device) が登場し、ASIC の試作用として開発コストの削減に大きく貢献した。さらに、ここ数年の技術革新により、FPGA は高集積化し、高速書き換えが可能となった。加えて、ハードウェアを記述する言語であるハードウェア記述言語により、ハードウェアの高速性と並列性、ソフトウェアの柔軟性を兼ね備えた計算機システムが構成可能になりつつある。計算コストが大きい場合、必要な回路を書き換えながら計算を実現することで、高速な処理を実現する Reconfigurable Computing(RC)[1] が提案されている。従来、画像処理および FFT(高速フーリエ変換:Fast Fourier Transform) 等の高速化には ASIC を必要とした。RC では、論理演算等の FPGA 向きの処理を必要に応じて回路を再構成しながら計算を進めることで、汎用プロセッサと協調し高速な演算を実現できる。これまでにも DISC[2][3]、RIFLE-62[4] 等のシステムも開発され、様々な成果を挙げている。

我々は CPU に付加することで CPU では負荷のかかる演算を高速に行なうコプロセッサに注目し、ステレオビジョンにおいて計算に時間のかかるパターンマッチングを FPGA で高速に行なうことを試みた [5]。

しかし、従来の FPGA ボードで RC を実現した場合、FPGA-メモリおよび FPGA-CPU 間のデータ転送のオーバーヘッドが実際の処理時間を大幅に上回っており、RC の高速演算のメリットが活かせなかった。そこで本研究では、データ転送のボトルネックを軽減し高速に回路書き換えを行なうことのできるシステムを製作する。また、暗号処理を例にシステムの性能を検証する。

2 Reconfigurable Computing

ノイマン型計算機は、ソフトウェアによるプログラムを用い様々な処理を行なうことができる。しかしながら、実時間処理が必要な問題では、ASIC を開発して高速処理を実現する必要がある。ASIC は、特定用途に特化した処理回路であるため、高速な処理を実現することができるが、仕様の変更への対応や拡張性に関して問題が残る。

複雑な処理を実時間でで処理する要求に対し、ハードウェアの高速性とソフトウェアの柔軟性を兼ね備えた計算機システムが望まれ、Reconfigurable Computing(RC) が注目されている。

RC とは、対象とするアプリケーションに応じて FPGA の回路を変更することによって、高速処理を図る手法であり、FPGA でなければ実現できないシステムである。また、従来の ASIC を開発する方法と比べて、仕様の変更や拡張に対して柔軟であり、それぞれのアプリケーションに応じた構成をとることができる。このことにより、専用ハードウェアの高速性と、汎用計算機の柔軟性を両立することができる。

RC の概念を図 1に示す。RC の利用形態は以下のように分類できる。

- アタッチド・プロセッサ
- リコンフィギャラブル・コプロセッサ
- リコンフィギャラブル・プロセッサ

本稿では特にリコンフィギャラブル・コプロセッサについて検討する。RC では処理を必要に応じてハードウェアとソフトウェアに適切に分割することで汎用 CPU と ASIC の両特長を持つことができる。

図 1では直列に描いてあるが CPU と FPGA を並列実行可能な場合は、さらに高速化が可能である。

コプロセッサ型の RC は、近年の PC とインターネットの急速な普及を考慮すると、PCI ボード形式で容易に付加できるシステムが望まれる。

3 RC の実装上の問題点

コプロセッサ型の RC システムを PCI などの汎用ボードの形で実装を考えた場合の最も大きな問題点は、データ転送のオーバーヘッドがボトルネックとなることである。特に回路データ転送のボトルネックが FPGA の再構成時間の大半を占める。

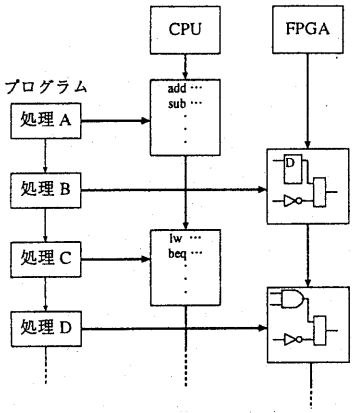


図 1: RC の概念図

通常の FPGA システムの概略を図 2に示す。回路データ・演算データは、ホスト PC からバスを通じて FPGA に送られる。特に回路データの転送はパラレルポートを用いることが多く、実際の回路書き換えには十数秒を要する。また、回路書換えを行なう度にデータを転送する必要があるため、頻繁に回路書換えを行なう RC においては、書き換え時間のオーバーヘッドのためにシステム全体としての性能が大きく低下する。

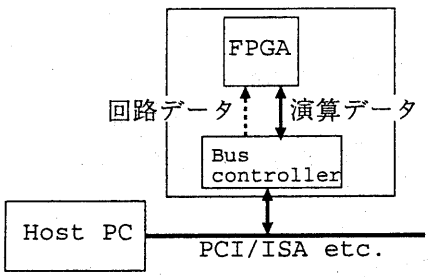


図 2: 従来の FPGA システムの概略

頻繁に回路書換えを行なう RC においては高速に FPGA の回路書換えを行なうことが必要である。高速に回路を書き換えることが可能な LSI としては、複数の回路データを内部 RAM に格納し、それらを切替えて再構成を行なうマルチコンテキスト FPGA が提案されているが、未だ研究段階であり入手が困難であること、実装できる回路量に限界があることが問題である。

そこで本研究では、市販の FPGA ボードを用いて高速に回路書き換えを行なうことのできるシステムを試作したので報告する。

4 RC システムの実装

今回試作したシステムの全体図を図 3に示す。FPGA を高速に書き換えるために、複数の Config.EPROM を用いて、これらを必要に応じて切替えて回路の再構成を行なう。また FPGA による演算を行なっている間に、Config.EPROM を書き換えることで回路データの転送のオーバーヘッドを軽減することができる。

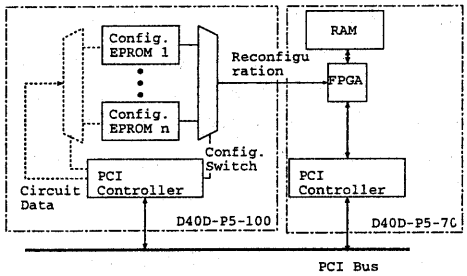


図 3: 提案する FPGA システム

本システムではアリテック社の FPGA 搭載 PCI ボード D40D-P5 シリーズを 2 枚使用した。本来は図 3の左側に点線で示されている部分は、必要な回路データを必要な時に Config.EPROM にダウンロードすべきであるが今回はあらかじめ保持しておくこととした

試作したシステムの性能を検証するため、暗号処理へ適用した。

5 暗号処理への適用

5.1 暗号処理

近年のインターネットの急速な普及により、ネットワークを通じて多くの情報を容易に得ることができるようになった。しかし一方で、いかにセキュリティを保つかも問題になっている。データを暗号化して転送や保管を行なうことが重要となり、様々な暗号処理法が提案されている。

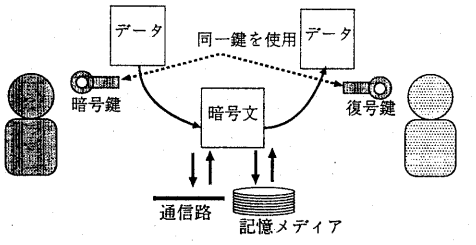
堅固な暗号を使用することは、複雑な処理を多く行なう必要があり、実時間処理を実現するためには

専用ハードウェアを用いることが多い。

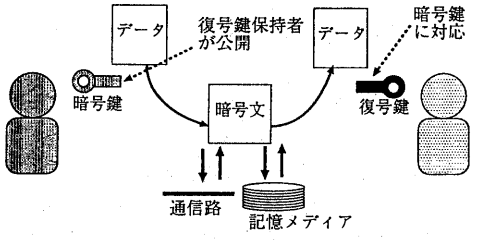
データを暗号化する手法は、暗号に必要な鍵と呼ばれるビット列の形態に応じて、大きく2つに分けることができる。それぞれ「共通鍵暗号方式」「公開鍵暗号方式」と呼ばれている。

共通鍵暗号方式の概略を図4(a)に示す。共通鍵暗号方式は暗号に必要な鍵（暗号鍵）と復号に必要な鍵（復号鍵）が同一の暗号方式であり、高速に暗号・復号が行なえるのが特徴である。しかし、多人数と通信する場合においては隠匿すべき鍵の量が増えるといった鍵保管の問題がある。例えば100人と暗号通信を行なう場合には、100個の鍵を隠匿する必要がある。共通鍵暗号方式の中には、数bit単位のブロック単位で暗号を行なうブロック暗号と1bitづつ暗号を行なうストリーム暗号がある。

一方、公開鍵暗号方式の概略を図4(b)に示す。公開鍵暗号方式は暗号鍵と復号鍵が異なっており、隠匿すべき鍵は自分の復号鍵のみであるので鍵保管の問題は無いが、共通鍵暗号方式に比べ処理速度が1000分の1程度と遅いのが欠点である。これらの特徴をふまえて公開鍵暗号方式は共通鍵暗号の鍵の配送や認証に、共通鍵暗号方式は実際のデータの暗号化に使われていることが多い。



(a) 共通鍵暗号方式



(b) 公開鍵暗号方式

図 4: 暗号方式の概略

5.2 共通鍵ブロック暗号 DES

共通鍵暗号方式のアルゴリズムとしては DES (Data Encryption Standard , FIPS 46-2) が有名である。DES は1974年に米国IBM社により開発され1977年に米国政府により標準暗号に指定された共通鍵ブロック暗号である。

図5に DES 暗号の概略を示す。DES においては、平文を64bitのブロック単位で64ビット(うち8ビットはパリティ)の暗号鍵を用いて暗号処理を行なう。まず、鍵生成部により暗号鍵から16個の副鍵を生成する。その後、入力の平文のビットを一定のマトリクスに従って入れ換え(初期転置)、その後32ビットずつに分割し、生成した副鍵を用いて非線形関数fにより暗号化していく。それを16回繰り返した後、初期転置を戻すマトリクスによりビットを入れ換える(逆初期転置)。

DES は開発されてから20年以上にわたって多くの暗号学者により解読が試みられたが、未だ有効な解読法が見つかっていないという実績により、欧米の金融機関を中心に使用されてきた。しかし、近年のコンピュータの計算能力の向上に伴い DES 暗号の安全性が低下しており、これに代わる暗号アルゴリズムが求められている。

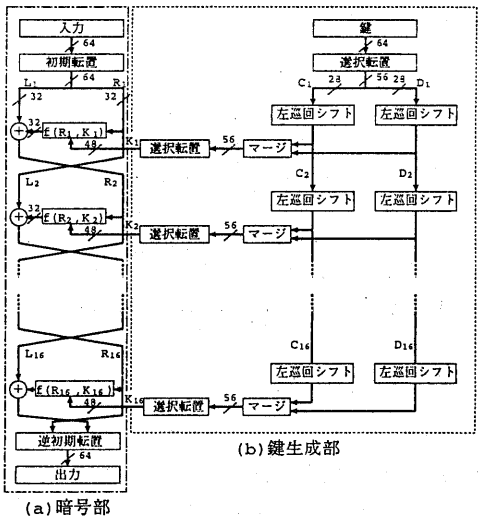


図 5: DES 暗号

現在、米国の NIST (National Institute of Standards and Technology) を中心に AES (Advanced Encryption Standard) と呼ばれる新しい政府標準

暗号の策定が行なわれている [6]。

一方で DES に代わるアルゴリズムとして Triple DES が金融機関を中心に使用されている。Triple DES は 2 つないし 3 つの暗号鍵を用いて 3 度 DES 暗号を施すことで、従来の DES 暗号に比べ格段に強度を上げている。しかし 3 度 DES 暗号を施すため、暗号処理に時間がかかるといった問題がある [7]。

5.3 試作システムの暗号処理への適用

一般的な共通鍵ブロック暗号アルゴリズムの概略を図 6に示す。一般に暗号を解読する手法としては、大きくブルートフォース法と統計的手法の 2 つに分けられる。前者は考えられる全ての鍵を試して解読する手法であり、後者は暗号文- 平文のペアの統計的性質を基に鍵を絞り込む手法である。前者に対しては主に鍵のビット長を増やすことで強度を増強でき、後者に対してはラウンド関数を複雑にするか、暗号化段数を増やすことで強度を増加できる。

しかしハードウェアによる処理において、これらの変更はハードウェア構成の大幅な変更を伴い、必要なハードウェア量は格段に増える。そこで本研究では、図 7に示すように暗号化アルゴリズムを複数に分割し、動的に書き換えることで大規模な暗号を小さいハードウェア量で実装することを考える。また、試作した RC システムを用いることで暗号処理自体を高速に行なうことを試みる。

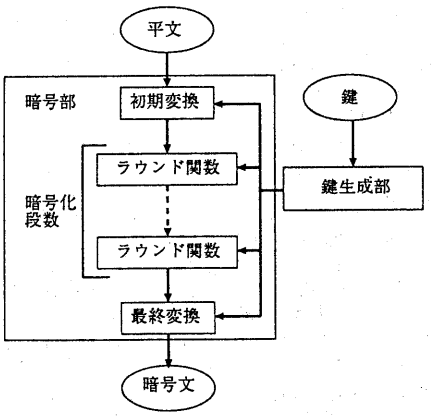


図 6: 共通鍵ブロック暗号アルゴリズムの概略

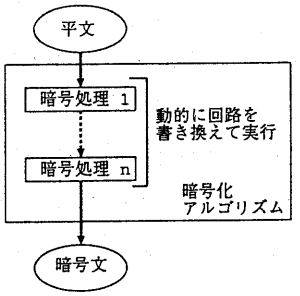


図 7: 暗号処理の分割

6 実験

6.1 提案するシステムの実装

4章で提案したシステムをアリテック社製 FPGA 搭載 PCI ボードに実装し、試作したシステムによる書き換えと回路データのダウンロードによる書き換えとで回路書き換え時間の比較を行なった。回路書き換えを行なった FPGA は Altera flex10k70 であり、Config.EPROM としてはワンタイム PROM である Altera EPC1 を用いた。また、回路データのダウンロードによる書き換えとは、Altera bitblaster 相当のダウンロードケーブルを用いて、直接 FPGA を再構成することを示す。再構成用のデータは 108KB のデータを持つ。結果を表 1に示す。

表 1: 回路書き換え時間の比較結果

	書き換え時間 [kbps]
ダウンロード	54.3
提案システム	5.51×10^3

表 1より、提案システムは従来に比べ約 100 倍の速度で回路書換えを行なえることがわかる。

6.2 暗号処理のシミュレーション

提案システムを暗号処理へ適用した際の全体としての処理時間を検証するシミュレーションを行なった。暗号化ブロック 64bit、暗号化段数 48 段の DES 暗号を行なった場合を想定し、RC を用いた場合とソフトウェアで同等の処理を行なった場合とでの比較を行なった。暗号化を行なうデータは 1MB とした。今回は図 8の様に暗号器を 2 つに分け、2 度回

路書換えを行なった場合を想定した。また、回路はあらかじめ Config.PROM にダウンロードすることとしている。シミュレーション条件を以下に示す。また、結果を表 2に示す。

- WS : UltraSPARC-I 167MHz (SunOS 5.5.1)
- PC : Pentium 200MHz (Windows 98)
- RC : FPGA Altera flex10k 19MHz

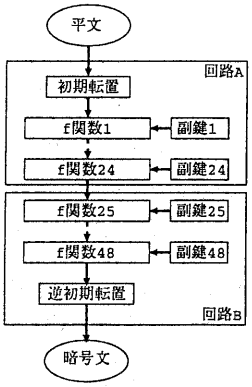


図 8: DES 暗号の分割方法

表 2: シミュレーション結果

	処理時間 (s)	書換え時間 (s)	計 (s)
WS	60.09	—	60.09
PC	35.27	—	35.27
RC	0.072	0.32	0.392

表 2より処理時間のみでは RC は WS の 835 倍、PC の 490 倍の性能向上が見られる。回路書き換え時間を含めても RC においては WS の 153 倍、PC の 90 倍の性能向上が認められる。ブロック暗号においてはブロック間に依存関係が無いので、暗号を施すデータの量によらず回路書き換え回数を一定にすることができる。よって、暗号化を行なうデータが増えるほど RC によるメリットが活きてくることもわかる。

また今回試作したシステムによる RC においては回路書き換え時間がボトルネックになっているが、コンフィギュレーションクロックの速い FPGA を用いることで高速化が可能である。今回用いた flex10k のコンフィギュレーションクロックは 10MHz であるが [8]、66MHz の FPGA も存在する。このため書き換え時間も現状でも約 6 倍は高速化が可能である。

7 まとめ

RC を従来の FPGA システムで実装した際には、データの転送のオーバーヘッドが RC の性能を大きく低下させていた。特に頻繁に回路書換えを行なう RC において、回路データの転送のオーバーヘッドは RC の性能を大きく低下させる要因であった。

そこでデータ転送のボトルネックを軽減し、高速に回路書換えを行なうシステムを製作した。従来の手法に比べ約 100 倍の速さで回路書換えを行なうことができることを確認した。また、暗号処理を例にシステムの性能を検証した。書き換え時間を考慮しても試作したシステムは WS の約 153 倍、PC の約 90 倍の性能向上が認められることをシミュレーションにより確認した。

今後は RC のメリットをより一層活かせるアプリケーションの検討を行なう予定である。

なお、本研究は（財）名古屋産業科学研究所「創発型ソフトコンピュータの開発」プロジェクトの一部として行なわれた。

参考文献

[1] 末吉敏則" リコンフィギャラブルロジック" 電子情報通信学会誌 Vol.81 No.11 pp1100-1106, 1998.

[2] Michael J.Wirthlin, Brad L.Hutchings, "Sequencing Run-Time Reconfigured Hardware with Software" in *ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pp.122-128, 1996.

[3] Michael J.Wirthlin, Brad L.Hutchings, "A Dynamic Instrubtion Set Computer" in *FCCM'95.IEEE Computer Society*, 1996.

[4] Milan Vasilko, David Long, "RIFLE-62:A Flexible Environment for Prototyping Dynamically Reconfigurable Systems" in *Proceedings of 9th IEEE International Workshop on Rapid System Prototyping (RSP '98)*, Leuven, Belgium, June 3-5, pp. 130-135, 1998.

[5] 山口晃由、橋山智訓、大熊繁"Reconfigurable Computing の画像処理への適用に関する基礎検討" 第 38 回計測自動制御学会学術講演予稿集 pp. 325-326, 1999.

[6] 宇根正志" 最近の AES を巡る動向について" 日本銀行金融研究所ディスカッション・ペーパー・シリーズ No.98-J-21

[7] 谷口文一・太田和夫・大久保美也子"Triple DES を巡る最近の標準化動向について" 日本銀行金融研究所ディスカッション・ペーパー・シリーズ No.99-J-6

[8] " FLEX10K Embedded Programmable Logic Family Data Sheet ver. 4.01"June 1999 Altera Corp.