C-003

ダイナミック・リコンフィギュラブル・プロセッサの応用と評価

宮本　悠生†　　小柳　滋†
†立命館大学大学院　理工学研究科

1. はじめに

近年、ハードウェア構成を動的に切り替えることのできる動的リコンフィギュラブルプロセッサに注目が集まっている。そこで本稿では、これの有用性を検証することを目的とし、IPFlex 社の DAPDNA-2 を用いて、キラーアプリケーションとされる画像処理、暗号処理アルゴリズムを設計する。検証の手段としては C 言語で作成したものとを比較することによる処理速度での観点から性能評価を行う。併せて DAPDNA-2 でのハードウェア設計による設計面からも評価を行う。

2. DAPDNA-2

DAPDNA-2 は IPFlex 社が開発した動的リコンフィギュラブルプロセッサであり、動的再構成を制御する 32bit の RISC プロセッサ(DAP)と動的再構成可能な演算コア(DNA)とを併せ持つマルチプロセッサである[1]。図 1 にその構成を示す。これはマルチコンテキストによる動的再構成を行い、最大で 4 つのコンフィギュレーションデータを保持することが可能である。DNA 内には 2 次元配列状に 32bit の PE(Processing Element)が配置されており、これを 6 つのセグメントに分割することで動作周波数の一定化を図っている。又、各データフローは並列度を自由に設計できる柔軟な構造となっている。

DAPDNA-2 の開発環境としては FW II が用意されており、3 種類の方法でコンフィギュレーションデータを設計できる。本稿ではその 1 つである DFC(Data Flow C)を用いる。DFC はデータフローを記述する拡張 C 言語である。又、制御部の DAP は従来の C 言語で記述する。DFC コードは DFC コンパイラによってコンフィギュレーションデータに変換され、続いて DNA コンパイラによって配置配線され、オブジェクトコードが生成される

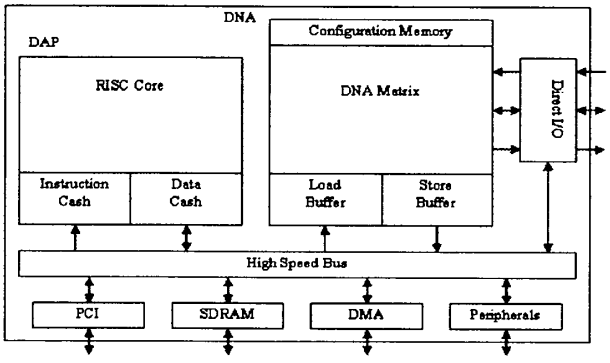

図 1　DAPDNA-2 の構成

The Application and Evaluate of Dynamic Reconfigurable Processor
† Yuki Miyamoto, † Shigeru Oyanagi
†Department of Computer Science, Graduate School of Science and Engineering, Ritsumeikan University

3. 離散コサイン変換の設計

離散コサイン変換(DCT)は JPEG や MPEG といった画像圧縮に用いられるアルゴリズムの 1 つである[2]。その際には 2 次元に拡張し 8x8 画素のブロック毎に分割して処理を行うが、これは 1 次元 DCT を 2 回繰り返すことにより実現可能である。それに則って 1 次元 DCT のコンフィギュレーションデータを設計し、それを 2 回呼び出すこととした。又、固定小数点演算で以ってこれを設計し、入力画素データは DAP の内部メモリに格納する。

当初は 8×8 の画素データを 1 単位として DNA で処理させるという仕様であったが、これでは DNA 呼び出し回数はデータ数に比例して増加するので、処理時間も増大した。そこで一度に全画素データを DNA で処理をさせることを試みたが、エラーが生じたため、DNA が処理できる様、入力データ、8x8 画素を 1 単位とし、それを 16x16 四方（128x128 の画素）用意し、それを 1 パックとして DNA に転送し、DCT の処理をさせた（図 2 参照）。

このパックは一度に 128×128 のデータを DNA に転送するので、DNA 呼び出し回数は当初のものより、1/256 となった。但し、その分 DNA 側での処理量も増加するため、処理時間も比例して 1/256 となった訳ではなく、画像サイズ 1600×1200 においては、約 1/15 の処理時間となった。

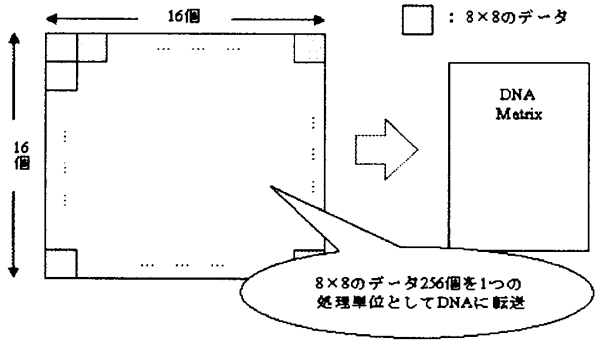

図 2　DCT のデータパック

4. モンゴメリ乗算の設計

モンゴメリ乗算は RSA 暗号といった公開鍵暗号の剰余演算で多用されるアルゴリズムである。これは除算をビットシフトに置き換え、積和演算の繰り返しで剰余乗算を行うものであり、ハードウェア化に適している[3]。

このモンゴメリ乗算では、入力値の 1 つの事前計算として拡張ユークリッド互除法を用いる必要があるが、DFC では文法上記述することができない。そこでこれも含めた他の入力データ（被乗数、乗数、法）を DAP 側で計算し、その値を DNA 側に引数として渡す。DAPDNA-2 は乗算に関しては 16bit までしか扱えないので、1 つのデータサイズを 16bit とし、それを 4bit 毎に 4 分割する、小規模のモンゴメリ乗算を行うことにした。通常 1024bit のデータサイズを用いるのが主流であるが、本研究の目的はソフトとの

比較であるため、16bit のデータサイズとした。具体的な設計手順としては、まず乗数を 4bit 毎に分割と、被乗数の最下位 4bit を得る。続いて積和演算部を静的コード反復である *seq* を用いて並列展開して計算を行う様にし、その演算結果の符号調節をした後に P を出力させるように設計した（図 3 参照）。

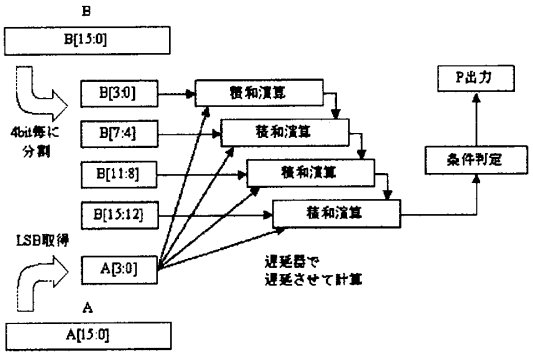

図 3　モンゴメリ乗算の設計方法

5. 評価

5.1 画像処理の性能評価

実行環境は、DFC は動作周波数 166MHz、FW II 上にて、C は動作周波数 3.2GHz、PC 上にて実行し処理時間による比較を行う。下図 4 に結果を示す。なお本稿では DCT のみ記載したが、他の画像処理アルゴリズムも併せて記載する。

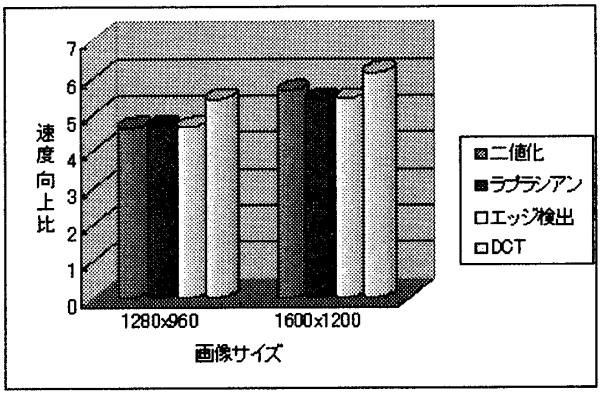

図 4　DAPDNA-2 の PC に対する処理時間比

上図 4 から DFC の方が、動作周波数比が約 1/20 なのに対し、約 5 から 6 倍の速度向上が得られており、画素データの多いほうが大きな速度向上を得ていることも分かる。これはデータが多い分それだけ、DNA 側の処理時間が増えたためではないかと考えられる。

5.2 モンゴメリ乗算の性能評価

こちらは DFC、C ともに FW II 上で実行し、処理に要するクロック数での比較を行う。図 5 に結果を示す。

速度向上比は緩やかに高くなっていることが分かるが、データ数が少ないとそれ程速度向上が得られていないことが分かる。又、データ数が 10 の時は圧倒的に C の方が処理クロック数は少ない。このことからデータ数がそれ程多くないものに関しては DNA 側ではなく、DAP 側で処理をさせた方が、処理時間が短く済むことが分かる。

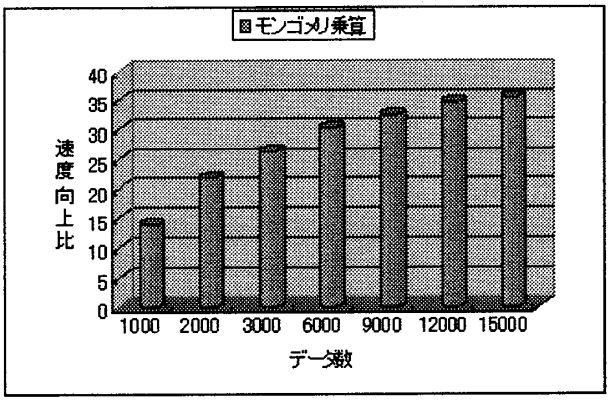

図 5　DNA の DAP に対する処理クロック数比

5.3 設計評価

DFC を用いたハードウェア設計の特徴としては、C 言語ベースであるため、C からの多少の変更でハードウェア設計が可能であるので、TAT の削減、設計効率の高さが挙げられる。又、生成されるコンフィグレーションデータは DFC ソースコードに依存することも挙げられる。

DNA マトリックスには 1 つのコンフィギュレーションデータに最大 4 つまでの別のアルゴリズムを適応することができるが、DFC による設計ではコンフィギュレーションデータには 1 つのアルゴリズムしか設計することができない。又、これは経験則によるものだが、主に使用する DNA マトリックスのセグメントも 3 つまでしか使用しない。別の設計手法であるスケマテック設計ならば、この両者の問題を解決することができると思われる。そのため DFC による設計ではリソース使用率を向上させることは難しいと考えている。

6. おわりに

本稿では動的リコンフィギュラブルプロセッサの応用ということで画像、暗号処理アルゴリズムを設計し、各々に対して処理速度に重みを充てた性能評価、DAPDNA-2 における高位言語によるハードウェア設計の評価を行った。このことから現行の CPU よりも大きな性能を得ていることが分かる。しかし、今後も CPU の動作周波数は向上することが予測される。そのため動的リコンフィギュラブルプロセッサも並行して周波数向上を果たす必要がある。又、C 言語からの多少の変更で様々なアルゴリズムを設計することができ、ソフトウェア設計の様にハードウェア設計を行うことができた。

しかし今回設計したものは、開発ツールの機能を必要最低限用いたものであり、それ故まだ性能向上の余地はあると考えている。

参考文献

[1] IPFlex 社：”DAPDNA ダイナミック・リコンフィギュラブル・プロセッサ DAPDNA-FW II BASIC コース資料”, 2004

[2] 安居院猛、長尾智晴：”C 言語による画像処理入門”、昭晃堂 2001

[3] 鈴木大輔、市川哲也、粕谷智巳：”FPGA 向けモンゴメリ乗算アーキテクチャの提案”、第 1 回リコンフィギュラブルシステム研究会 2003