画像処理システムの新設計手法の検討

A New Design Environment for Image Processing Systems

山　本　直　人＊　　船　山　　智＊
Yamamoto, Naoto　　Funayama, Satoshi

Image processing systems have grown large and complicated, and limitations to the usual methods of developing these systems are now being encountered because the scale of image processing systems has come to exceed the productivity of conventional development tools. Reported here is a new design environment that cuts the development time of image processing systems in half.

1　はじめに

従来の画像処理システムの開発は、まずC/C++などの高級言語でアルゴリズムを設計・検証し、その後アルゴリズムを組み込んだシステム・アーキテクチャ設計を行い、ハードウェア部（ASIC、FPGA）とソフトウェア部（DSP、MPU）に切り分けて開発を進め、最後に実機による評価・検証を行うという手法をとってきた。しかしながら、近年大規模・複雑化する画像処理システムの設計においては、このような開発方法では限界が見えてきた。開発するシステムの規模と設計生産性のギャップが大きくなり始めたからである。また、短期間で製品を開発し、かつ品質を保証するために、膨大なデータを使った高速な検証が必要になり、上流（アルゴリズム）から下流（ハードウェア）まで効率よくモデリングできる設計環境が不可欠となってきた。（Fig.1）

そこで、本検討では各段階をシームレスに設計・検証できる環境を新規設計手法として導入し、適用検討例として画像処理アルゴリズムのハードウェア化を行い、その有効性を確認したので報告する。

2　新しい開発手法の必要性

例えばデジタルカメラやプリンタなどの画像情報機器における画像処理システムの構成は、機器固有の入力デバイス（DSCの場合はCCDなどの撮像素子）や出力デバイス（プリンタの場合は作像デバイス、像形成材料）の特性などに合わせて決定される。採用される画像処理システムは、画質はもちろんのこと、処理に必要なリソースやコストとのトレードオフに関しても十分考慮されなければならない。

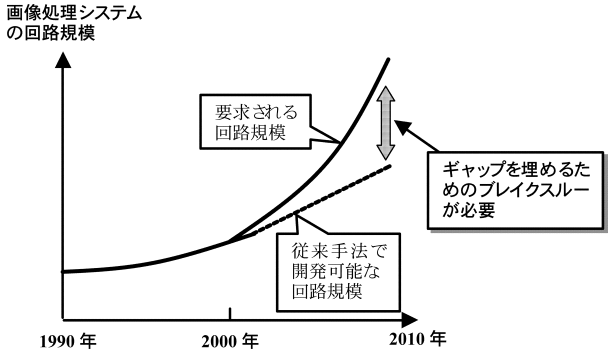

Fig.1　設計生産性の限界

システム全体が小規模な場合は、従来の開発方法で特に問題はなかった。しかしながら、処理画像容量の拡大、処理方法の複雑化により、「開発期間の長期化」、「開発費用の高騰」などの問題が顕在化し始めた。この問題に対処する方法として、さらなる人的資源を投入するより、開発手法そのものを改善するという機運が高まってきた。画像処理システム全体を抽象的なレベルで最適化し、システムのパフォーマンスを早い段階で見積もり、設計の効率化を図るという「システムレベル設計」という手法である。

本検討で取り上げたシステムレベル設計手法は、設計の上流から下流までシームレスに設計・検証が可能であること、画像処理に特化した豊富なライブラリを用いて同一環境下においてアルゴリズムの設計・検証ができること、さらに合成可能なHDL（Hardware Description Language）の生成ができ、HDLシミュレータによりハードウェア化した際の検証も同一環境上から行えることを特徴とする。新しいシステム設計のフローを新規手法とし、従来手法と比較した結果をFig.2に示す。

＊中央研究所　第3開発グループ

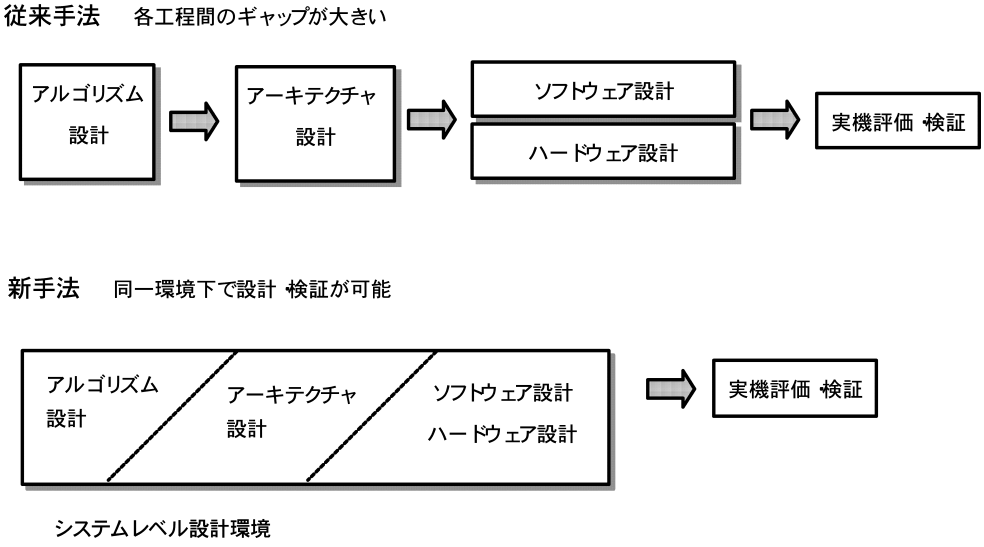

Fig.2 新規手法と従来手法の比較

3　画像処理アルゴリズムの適用検討

今回、「システムレベル設計」の有用性を確認するために適用検討例として誤差拡散アルゴリズムを取り上げた。以下、その過程と結果を示す。

3.1　アルゴリズムの概要

誤差拡散法は擬似中間調を生成する一手法であり、表示可能な階調数が限られるプリンタなどで階調を効果的に変換するために利用されることが多い。その原理は次のとおりである。

元画像の画素濃度と表示画像の画素濃度との差を2次元マトリクスにより重み付けをして周辺画素に拡散させ、局所的な平均誤差を小さくしようとする方法である。ここで使用される2次元マトリクスの重み付けの係数は、注目画素に距離的に近いものほど大きな値を与えられる。Fig.3に誤差拡散法に使用される2次元マトリクスの一例とその処理の様子を示す。

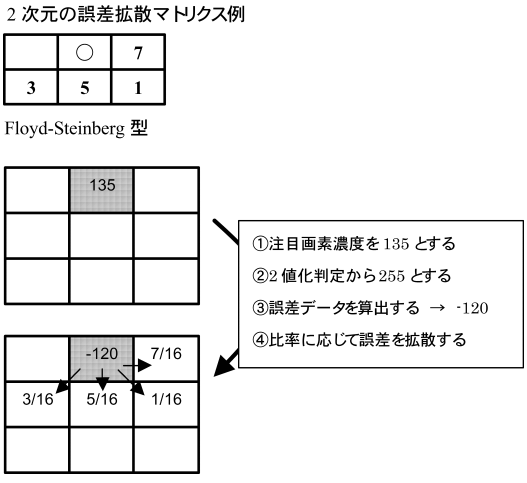

Fig.3 誤差拡散法の原理

3.2　従来型（C/C++→HDL）の設計・検証

Fig.3に示すような画像処理システムを従来手法で設計する場合、C/C++で設計した抽象的なアルゴリズムをHDLで記述し、ハードウェアに変換する工数が最も多い。ハードウェア特有の概念（クロック、並列処理）を導入し、性能とリソースのトレードオフを考慮し、かつC/C++との処理結果の等価性を保証しなければならないからである。

特に誤差拡散法のハードウェア化は、2次元マトリクス演算のためにラインメモリや積和演算を必要とするため、ハードウェア規模や速度の要求仕様が厳しい。

3.3　システムレベル設計環境上での設計・検証

従来手法に対して「システムレベル設計」手法では、Fig.4に示すように抽象度を大きく3段階に分けて設計・検証することができる。

3.4　アルゴリズムベース……設計と検証

アルゴリズムベースでの設計・検証は、最も抽象度が高く、データフローを時間的な概念不要で記述できる。例えば画像処理システムの場合フレーム単位で処理するため、機能をどのようなアーキテクチャで実現するかを考慮する必要がないため、検証時間が最も短くて済む。記述方法としては、C/C++を利用する事が多い。

さらに本検討で利用した設計環境では、Fig.5に示すような設計・検証に必要な画像処理ブロックが提供されており、ユーザが定義したブロックを「カスタムコードブロック」として設計環境に組み込むためのC/C++のテンプレートも用意されているため、設計者は一からコードを記述しなくても、必要なアルゴリズムの設計に注力できる。実際に、記述コード量としては従来手法の1/10で済んでいる。

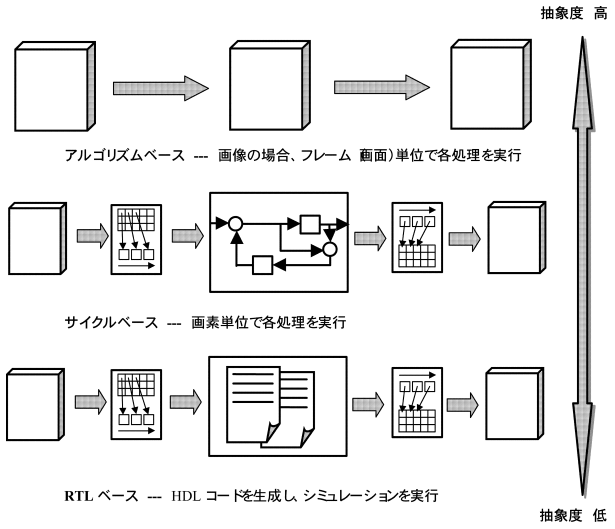

Fig.4 システムレベル設計環境の抽象度

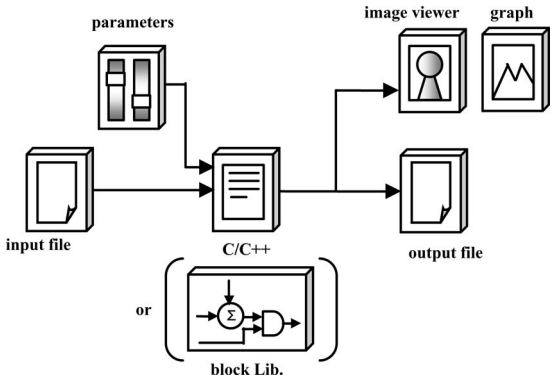

Fig.5 アルゴリズムベースの設計・検証

3.5 サイクルベース --- 設計と検証

アルゴリズムベースでの検証後はシステムへインプリメントするために、抽象度を下げていく。具体的にはシステムの性能、リソースを考慮しながら、ハードウェアまたはソフトウェアのどちらにインプリメントするのかを決定する。

ハードウェア化するブロックは、ハードウェア化を容易にするため処理を画面（フレーム）単位から画素（ピクセル）単位へ変更し、delay などの遅延ブロックを挿入することでクロックサイクル毎に処理を行うサイクルベースモデルを作成し、検証を行う。(Fig.6)

サイクルベースモデルでは、アルゴリズムベースより検証速度は低下するが、既にクロックの概念が導入されているため HDL 化およびその検証の障壁を従来手法より小さくすることができる。

ソフトウェアとしてインプリメントするブロックは、MPU や DSP のプロセッサモデルを使用してハードウェアとの協調シミュレーションするか、ハードウェア動作に必要な信号を生成するためのブロックとして再利用が可能である。

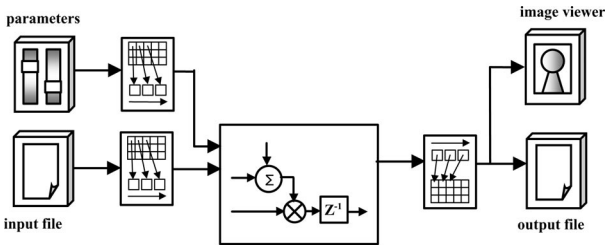

Fig.6 サイクルベースの設計・検証

3.6 HDL ベース --- 設計と検証

サイクルベースでの設計・検証は、必要に応じて浮動小数点精度のモデル、固定小数点精度のモデルの何れかで行うことができるが、ハードウェア化には固定小数点精度のモデルが必要となる。ここでは、固定小数点ライブラリを使用して演算ビット数や量子化の影響を検証し、画質の最適化を行う。

その後、サイクルベースで設計した固定小数点のモデルから合成可能な HDL を生成することで、ハードウェア（ASIC、FPGA）に直接インプリメントすることができる。

さらに HDL シミュレータとのリンクにより、同一環境上でアルゴリズムベース、サイクルベースでの結果と生成した HDL での結果との等価性検証も可能である。(Fig.7)

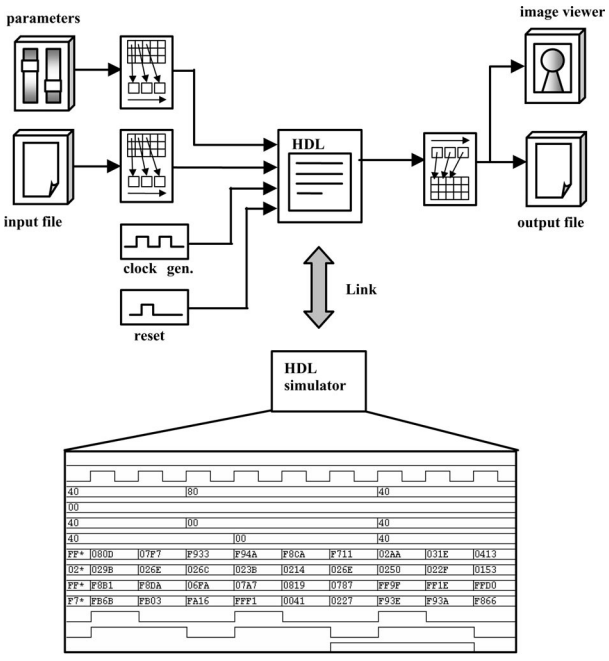

Fig.7 RTL ベースの設計・検証

3.7 効　果

「システムレベル設計」の適用により、アルゴリズム開発からハードウェア検証までに要する期間を、従来手法と比較して約1/2に短縮できた。特にサイクルベース検証を行うことでハードウェア化へのギャップを小さくでき、HDLを新たに書き起こすことなく、C/C++で記述したアルゴリズムの結果との等価性を検証できたことによる工数削減の効果は大きい。また同一環境上で設計・検証できるため、アルゴリズムベースのテストベンチをサイクルベース、HDLベースの各段階で使用でき、新たに書き起こす工数が不要であった。

アルゴリズムから固定小数点精度のサイクルベースモデル、HDL生成まで、共通の環境でグラフィカルに設計できるため、各レベルの設計者間（アルゴリズム、システム・アーキテクチャ、ハードウェア、ソフトウェア）で情報を共有しやすいという利点もあった。

3.8 課　題

- アルゴリズムベースの画面単位からサイクルベースの画素単位に変更して等価性を検証するのに時間を要した。詳細なハードウェア構成を意識するHDLベースほどではないが、やはり、アルゴリズムレベルとクロックサイクルを意識するサイクルベース設計を考慮した設計とのギャップは大きい。
- 最適化も含めたデータパス系の設計、検証には有効な手段である。しかし、実際のシステムは複雑な制御系を含む場合が多く、その設計には別の設計環境を利用した方が効率的であった。単体でのシステムレベル設計としては十分な機能を有しているとは言えず、適用範囲に応じていくつかの設計環境を併用する必要がある。

4　まとめ

「システムレベル設計」環境は、大規模化が進む画像処理システムの品質と設計生産性を向上させるために、多様なアプローチが現在でも試みられている。本報告で紹介した方法のほかにも、上流（アルゴリズム）から下流（ハードウェア）までC/C++をベースとした言語で設計する手法もいくつか登場している。

しかしながら、ハードウェアとソフトウェアをバランス良く融合させて設計できる環境、手法の本命は未だ出ていないのが現状である。特に課題でも挙げたが、サイクルベースへの変換のギャップは思いのほか大きく、劇的な効果を期待するにはサイクルベースに替わる動作レベルでのもっと効果的な手法が必要である。

抽象度と検証速度（シミュレーション速度）のトレードオフ、新しい設計言語とそれに対応する合成ツール、などクリアしなければならない問題は多く存在するが、システムレベルで考える設計手法は間違いなく今後の主流になると思われる。

●参考文献

1）"新版　画像電子ハンドブック"、画像電子学会編、コロナ社（1993）
2）"ファインイメージングとハードコピー"、コロナ社(1999)
3）"続　電子写真技術の基礎と応用"、電子写真学会編、コロナ社（1996）