C-2

アプリケーションプログラムを基にしたプロセッサアーキテクチャの自動生成

Automatic Generation of Processor Architecture Based on Application Program

栗崎 正和† 宮内 新† 荒井 秀一†
Masakazu Kurisaki Arata Miyauchi Syuichi Arai

1．はじめに

近年のFPGAを用いたプロセッサ開発分野では,汎用プロセッサの中で処理時間の短縮が必要とされる部分を,専用のハードウェアを追加実装して処理することによって,全体の高速化が行われている.しかし,汎用プロセッサ自身の高速化やソフトウェアアルゴリズムの最適化が行われると,その用途についての専用ハードウェアによる高速化は不要になるケースが考えられる.したがって,このような高速化を目的とした専用ハードウェアの開発は早いサイクルで行われなくてはならない.

そこで我々は,与えられたアプリケーションの実行を高速化するため,高級言語で記述されたプログラムから,演算が最適に並列化されたプロセッサの HDL 記述を生成する研究を行っている.

2．システムの構成

本研究で生成されるプロセッサは,VLIW 型で構成し明示的な並列演算を行なうことにする.

命令の組み合わせ頻度から命令や演算の合成を行ない,命令実行の時間的・空間的最適化を行なう.

この他のプロセッサ生成時の可変要素としては,ALU ユニット内の演算器の種類と数,レジスタの本数と幅,メモリの容量,アドレスバス等の幅,命令フォーマットを,アプリケーションに応じて変化させることにする.

アプリケーションプログラムからプロセッサを自動生成する流れとして,以下の図1に示す手順で処理を行う.

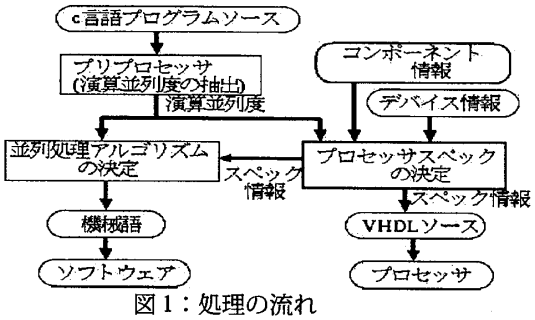

図1：処理の流れ

3．ベースプロセッサの仕様

ベースプロセッサは,実行並列度2のプロセッサと,3のプロセッサを用意した.これは外部メモリ(データキャッシュ)へのアクセスパスを2つに制限しているためである.ベースプロセッサはターゲットアルゴリズムのスケジューリングに合わせて並列度2または3のどちらかを選択して,高速化の基準とした.ベースプロセッサの主な仕様を以下に示す.

- IF・ID・EXE・MEM・WB の 5 ステージのパイプラインを持つ RISC プロセッサ
- VLIW 型による 2 並列または 3 並列の処理機構
- アドレスバス 32bit,命令フェッチ用データバス 64bit,データフェッチ用データバス 32bit
- 割り込み機構等は実装されていない

4．アルゴリズム向けプロセッサのスペック決定

4.1 命令セットの定義

アルゴリズム向けプロセッサを,ターゲットアルゴリズムが実行可能な最小限の命令セットにすることで,回路規模の縮小を図る方法もあるが,ターゲットアルゴリズム内で使用しない基本演算が検出された場合,その基本演算の演算器をプロセッサから外す事になる.しかし,その 1 つでも外した演算器の命令を使うアルゴリズムは,実行できなくなる.

本提案手法では,ターゲットアルゴリズム以外でも全てのアルゴリズムの実行を保証するため,最低限の汎用性を残し,ベースプロセッサとの互換性のあるアルゴリズム向けプロセッサを生成する方針である.アルゴリズム向けプロセッサに実装する命令セットは,基本命令・拡張命令・合成命令の3つに分類している.

4.1.1 基本命令

基本命令は,DLX や MIPS といった既存の RISC アーキテクチャ命令セットの中で多く使用されている命令や,一般的な演算処理を行う上での最低限の命令を選定したものである.ベースプロセッサは,これらの基本命令のみを実装している.

4.1.2 拡張命令

拡張命令郡とは,ターゲットアルゴリズムに有用と判断された拡張命令だけを,アルゴリズム向けプロセッサに実装するものである.

拡張命令はあまり使用されない命令や特定のアルゴリズムに有用と判断される命令が含まれる.

拡張命令では基本的に 16bit 乗算よりも遅延が小さいもので,また指定できるオペランドの数は3つとした.

4.1.3 合成命令

事前に定義しておいた拡張命令郡の中に,ターゲットアルゴリズムが短縮可能な命令がなかった場合,ベースプロセッサに拡張命令はまったく追加されない.

そこで,複数の演算を 1 つの命令で行うことによって,実行時間の短縮を図ることにする.この命令は,拡張命令と違って,スペックを決定する際に新しく定義する命令であり,本研究においては合成命令と呼ぶことにする.

合成命令には垂直型合成命令,水平型合成命令があるが,ここでは垂直型合成命令について図2に示す.

†武蔵工業大学,Musashi Institute of Technology

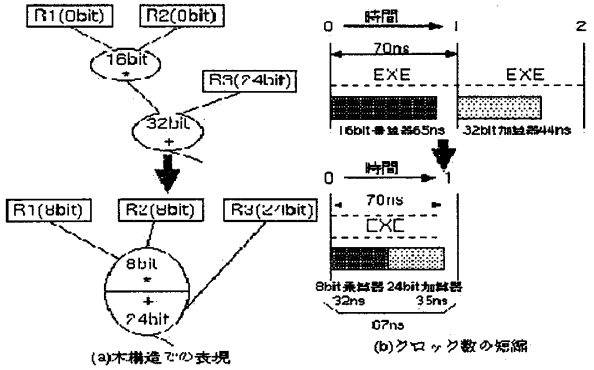

図2：垂直合成命令

垂直型合成命令とは,連続した演算の実行時間を短縮するものである.垂直型合成命令の定義は連続して行われる演算の組み合わせの頻度が高いものについて 1 つの命令発行で複数の演算を続けて行う命令を定義するものである.

なお,メモリアクセスユニット,分岐ユニットはそれぞれ演算ユニットからは独立しているため,メモリアクセス命令,分岐命令は合成命令の対象とはならず,それ以外の任意の演算命令について合成命令の対象となる.

5. アルゴリズム向けプロセッサの試作

5.1 拡張命令を実装したプロセッサの試作

拡張命令を実装するプロセッサの試作としてここでは DCT について解析を行った.しかし,拡張命令の実装方法についてはまだ検討段階であり,またコンパイラも完成していない為,ここでは C 言語ではなくアセンブリ言語にてプログラムを行い,拡張命令を実行するための演算器を作成した.その演算器の遅延を検証し,拡張命令をベースプロセッサに実装,その後,実行ステップ数,実行時間について検証した.

5.1.1 DCT

DCT では COS を求める演算が多く見られる.しかし COS を求める演算を基本命令のみでプログラムを行うとマクローリン展開などが必要なため,多くの命令が COS を求めるために必要である.そこで,テーブルを用いて COS を求める演算器を作成した.

この COS を求める拡張命令は論理合成の結果により 16bit 乗算の遅延よりも大きくなることから,1CLK ではなく 2CLK で COS を求める演算器も合わせて作成した.

また,DCT では C(u)C(v)を掛ける必要があり,これを基本命令のみでプログラムを行うと u,v の値により分岐を行う必要がある.そこで,この C(u)C(v)の乗算を 1 命令で行う拡張命令もあわせて作成した.

作成した拡張命令を実装した結果を次の表 1 に示す. 表 1 のようにベースプロセッサに COS を 1CLK で求める拡張命令を実装した場合,CLK 数は減少するが COS を求める拡張命令の遅延が大きいためプロセッサの速度が減少し,結果として実行時間が増えてしまう.そこで,COS を 2CLK で求めるようにすることで実行時間を減少することが出来る.

表 1：拡張命令を実装したプロセッサによる実行結果

	プロセッサ構成	周波数(MHz)	実行ステップ数	実行時間(μs)
DCT	ベースプロセッサ	28.9	23925	827
	cos=1CLKの拡張命令	21.6	19829	918
	cos=2CLKの拡張命令	29.4	20853	709

5.2 合成命令を実装したプロセッサ

垂直型合成命令を実装するプロセッサの試作としてここでは

- バブルソート
- 2 分探索
- 8 近傍画像重み付けフィルタ処理

の各アルゴリズムについて,基本命令セットのみを用いたアルゴリズムと垂直型合成命令を加えたアルゴリズムの実行時間の比較を行うことで,性能が向上したかどうかを判断する.次の表 2 に追加した垂直型合成命令を,表 3 に垂直型合成命令を実装したプロセッサによる実行結果を示す.

表 2：追加した合成命令

アルゴリズム	追加した合成命令
バブルソート	加算 16bit→乗算 16bit の垂直合成 32bit 即値代入→加算 32bit の垂直合成
2 分探索	加算 16bit→乗算 16bit の垂直合成
画像フィルタ処理	加算 32bit→乗算 32bit の垂直合成

表 3：合成命令を実装したプロセッサによる実行結果

アルゴリズム	プロセッサ構成	周波数(MHz)	実行ステップ数	実行時間
バブルソート	ベースプロセッサ	28.9	1014	37.3(μs)
	合成命令	23.8	810	34.0(μs)
2 分探索	ベースプロセッサ	28.9	61	2.16(μs)
	合成命令	24.7	57	2.31(μs)
画像フィルタ処理	ベースプロセッサ	28.9	3,229,117	118(ms)
	合成命令	23.7	2,132,345	89.8(ms)

以上の結果より,バブルソート,画像フィルタ処理に関しては高速化が行えたが,2 分探索では合成命令による実行ステップ数の減少があまり行えず,演算器を増やした為,周波数が遅くなり,実行時間が増大してしまった.

6. おわりに

今回は拡張命令,合成命令に焦点を当てた.この提案手法では拡張命令,合成命令により高速化がなされる.しかし,拡張命令,合成命令を実装した際の演算器の遅延の増加が原因となり,結果として実行時間が増大する可能性がある.

また,拡張命令は今の段階では手作業による追加を行っている.

以上のことより今後の課題として

- レイテンシの見積もり方法の検討
- 拡張命令の実装方法
- 拡張命令郡への多様な命令の拡充

などがあげられる.

参考文献

[1]松田 和幸,門脇 一馬,宮内 新,石川 知雄:"アプリケーションプログラムを基にしたプロセッサアーキテクチャの自動生成",SWEST3,2001.7.25

[2]門脇 一馬,松田 和幸,宮内 新,石川 知雄:"アプリケーションプログラムを基にしたプロセッサアーキテクチャの自動生成-プロセッサスペックの決定と処理アルゴリズムの決定-",第 5 回システム LSI ワークショップ,2001.11.27