非同期式カスケードALUプロセッサの評価

3N－6

深作泉　上野洋一郎　中村宏†　南谷崇†

東京工業大学 情報理工学研究科　†東京大学 先端科学技術研究センター

1　はじめに

クロック分配に起因する同期式システム設計の限界を突破することを目指し、事象駆動原理に基づいた非同期式プロセッサの研究が進められている。

非同期式では、最大遅延によって性能が決定される同期式とは異なり、平均遅延によって性能が決定される。その特徴を有効に生かすアーキテクチャとしてカスケードALU方式が提案されている。[1]

本稿では、カスケードALU方式を既存の命令セットアーキテクチャに導入し、シミュレーションによって性能評価を行った結果を述べる。

2　カスケードALU方式

非同期式プロセッサの新しいアーキテクチャとしてカスケードALU方式が提案されている。この方式は、データの依存関係の無い命令列はVLIW、スーパスカラ方式と同様に並列実行が可能でああ。そして、依存関係のある命令に関してはALUをカスケード接続することにより依存関係を解決し、複数命令を1サイクルで実行することが可能なアーキテクチャである。

非同期式プロセッサに対しこのアーキテクチャを導入することにより、以下のような効果による高速化が望める。

1. 依存関係の無い命令列は並列実行が可能である。依存関係のある命令列に対しても命令発行が可能であり、In orderであるにも関わらずALU数と等しいだけの命令発行ウィンドウサイズのOut of order実行と同じ効果が望める。

2. 算術系の演算器をカスケード接続した場合の演算遅延は、接続に要する回路の遅延、配線遅延を無視すれば、ビットレベルの並列性により単独で2つの演算を行った場合の遅延時間の和以下となり、データによっては通常の1命令分＋α程度の時間で実行できることがある。よって、一命令あたりの実行時間を短縮する効果が望め、命令レベルの並列度が低いプログラムでも高速化が期待できる。

3. 全体のサイクルタイムがALUに比べて遅い場合に、論理演算系の命令の組合せなどは1命令分、論理演算と算術演算の組み合わせなども、通常の1命令分＋α程度の時間で実行できるため、一命令あたりの実行時間を短縮する効果が望め、命令レベルの並列度が低いプログラムでも高速化が期待できる。

4. 2線2相方式の非同期式回路を用いた場合には休止相時間が必要である。しかし、休止相の実行は命令の依存関係に依らず並列に行なうことができるため、1命令当りの休止相時間は1サイクルに投入される命令数に比例して減少する。

これらの特徴から、カスケードALU方式を採用したプロセッサは、命令レベルの並列度の低いオブジェクトから高い並列度を持つオブジェクトに至るまで高い性能を発揮できると考えられる。

本稿では、以上の高速化要素のうち、1、3、4の要素を考慮したシミュレーションによって性能評価を行なった。2の効果ついては、[2]で述べられているので参照されたい。

3　シミュレーション方法

カスケードALU方式の性能を評価するにあたり、以下に述べるような3種類のアーキテクチャのシミュレータを作成した。

- 単一命令実行パイプライン方式
- 静的な命令発行／実行方式の単純な4並列スーパスカラ方式
- ALU数4のカスケードALU方式

単一命令実行パイプライン方式では、命令フェッチ、命令デコード、演算実行、データメモリアクセス、ライトバックステージの5ステージのパイプラインプロセッサを想定した。

単純なスーパスカラ方式、カスケードALU方式では、ロード側のデータメモリアクセスを実行ステージに統合し、ストア側をライトバックステージに統合した4ステージからなるパイプラインプロセッサを想定した。演算実行ステージに関しては、同一の構成のALUを4つづつ持ち、データアクセス系命令以外の同一種類の演算を最大4命令同時に実行できるものとした。データアクセス系に関しては、データのロードとストアの同一サイクル中の実行は各々1回づつに制限した。

スーパスカラ方式、カスケードALU方式において、その性能を大きく左右するものに分岐命令による制御ハザードがある。シミュレーションにおいては、各々の方式の利点を生かすため既存の分岐予測技術を採用するものとし、その分岐予測のヒット率は比較的簡単な分岐予測方式でも到達可能である90％と仮定した。[4]

ヒット率100％の命令キャッシュとデータキャッシュを用いているものとし、命令の供給に要するメモリアクセス、及びデータのアクセスは常に一定の時間で行なわれるものと仮定した。

演算時間はTITAC-2[3]の設計で得たデータを参考に設定し、1サイクルの時間は、各パイプラインステージの演算時間の最大値を取るものとした。2線2相方式の非同期式回路で構成することを想定しているため、各ステージの実行時間には演算時間の他に休止相時間が存在する。

シミュレーションで仮定する命令セットアーキテクチャには、R2000に準拠した命令セットアーキテクチャを持つ非同期式プロセッサTITAC-2のものを用いた。

性能評価に用いたプログラムはDhrystone2.1、Byte benchmarkのハノイの塔、エラトステネスのふるい、マンデルブロ集合演算、Squeeze法によるデータ圧縮、プロセッサシミュレータの6種類を用いた。これらをTITAC-2用のGCCによってコンパイルし、生成されたバイナリプログラムをシミュレータの入力とした。

Performance evaluation for Asynchronous Cascade-ALU Processors

Izumi Fukasaku, Yoichiro Ueno
Tokyo Institute of Technology, Graduate School of Information Science and Engineering
Hiroshi Nakamura, Takashi Nanya
Research Center for Advanced Science and Technology, The University of Tokyo

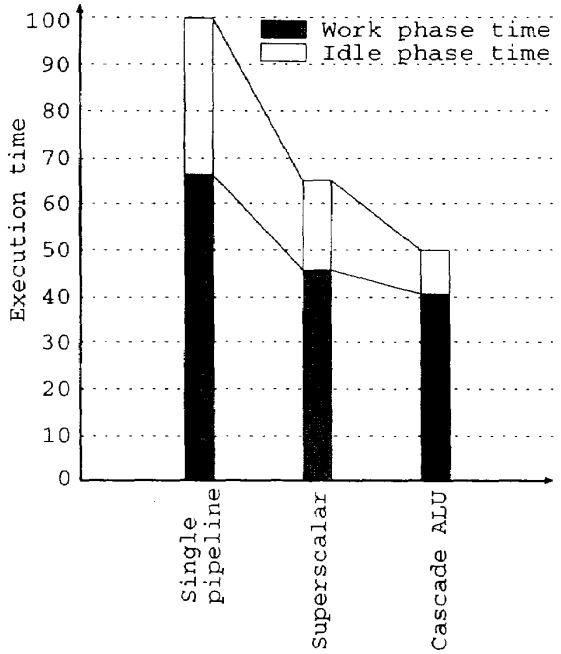

図 1: 各方式の性能比較（単一命令実行パイプライン方式を 100 とする）

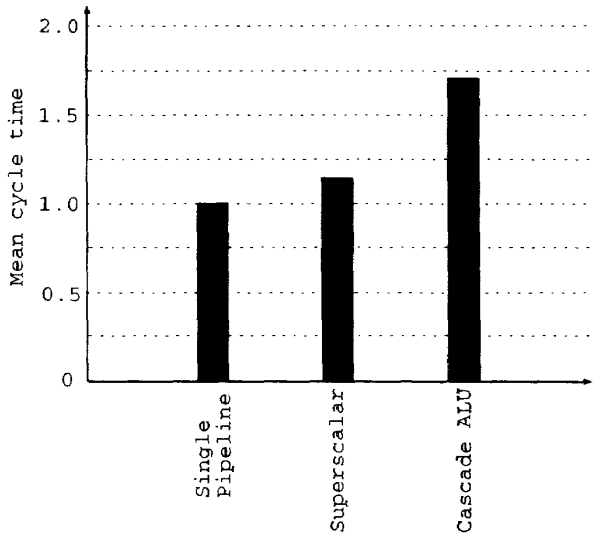

図 2: 各方式の平均サイクルタイムの比較（単一命令実行パイプライン方式を 1 とする）

4 シミュレーション結果

図 1に各方式の性能を比較した結果を示す。このグラフはプログラムの実行をシミュレーションし、その時のシミュレータ上のプログラム実行時間を単一命令実行パイプライン方式を基準とした相対性能で示している。相対実行時間は稼働相時間、休止相時間、およびそれらの合計に分かれている。

カスケード ALU 方式は他の 2 方式よりも性能が高いことがわかる。特に休止相時間低減の効果による性能向上が大きい。2 章のカスケード ALU 方式による高速化の効果の 4 に述べられているように、休止相時間低減の効果は 1 サイクルに投入される命令数に比例する。1 サイクル当たりに投入される命令数を分析すると、スーパスカラ方式では分岐等による制御ハザード、ロードストアユニット数の制限による構造ハザード、命令間のデータ依存の 3 種によって制限される。その結果、1 サイクルあたりの命令投入数の平均は 1.49 命令であった。一方、カスケード ALU 方式ではデータ依存のある命令はカスケード接続されるため、スーパスカラ方式よりも多くの命令を 1 サイクルに投入することが可能である。その結果、1 サイクルあたりの平均命令投入数は 2.81 命令となった。この差によって、カスケード ALU 方式はスーパスカラ方式よりも休止相時間が大幅に低減されている。

図 2に各方式の平均サイクルタイムを比較した結果を示す。このグラフは各プログラムの実行をシミュレーションした時のシミュレータ上の平均サイクルタイムを単一命令実行パイプライン方式を基準とした相対値で示している。

カスケード ALU 方式ではプログラムの実行時間で見た性能がもっとも高いにも関わらず、他の方式よりも平均サイクルタイムが大幅に大きくなっている。これは、複数の演算がカスケード接続されることにより発生する。カスケード接続される演算数を調べると、1 サイクル当たりの平均は 1.76 命令であった。2のスーパスカラ方式とカスケード ALU 方式での平均サイクルタイムの比は 1.48 である。カスケード接続される演算数ほどはサイクルタイムが悪化しないのは、2 章のカスケード ALU 方式による高速化の効果の 1 と 3 によるものである。

5 まとめ

カスケード ALU 方式と単一命令実行パイプライン方式、静的な命令発行／実行のスーパスカラ方式のシミュレータを作成し、その上でプログラムを実行させ比較を行った。その結果、今回の条件下でのシミュレーションでは、カスケード ALU 方式では平均サイクルタイムが大きく増大するにも関わらず性能が向上することが確認できた。

またこの結果から、非同期式プロセッサに対するカスケード ALU 方式の適用は、直接的に性能を向上させるばかりではなく、平均サイクルタイムの増大により各ステージの性能設計に余裕をもたらし、さらなる性能向上につながる様々な手法の適用を容易なものにすると考えられる。

今回の評価に用いたプログラムは、カスケード ALU 方式に適した最適化などは行なわれていない。カスケード ALU 方式に適した命令再配置等の最適化を行なうことでさらに高い性能が得られる可能性がある。今後、カスケード ALU 方式に適した最適化手法を確立する必要がある。

本研究の一部は科研費補助金基盤研究 (B)09480049、(株) 半導体理工学研究センターとの共同研究によるものである。

参考文献

[1] 上野, 深作, 中村, 南谷. 非同期式カスケード ALU アーキテクチャの提案. 情報処理学会第 56 回全国大会予稿集, Mar 1998

[2] 今井, 中村, 南谷. 非同期式プロセッサにおける複合演算の効果. 情報処理学会第 56 回全国大会予稿集, Mar 1998

[3] Akihiro Takamura, Masashi Kuwako, Masashi Ima, Taro Fujii, Motokazu Ozawa, Izumi Fukasaku, Yoichiro Ueno, and Takashi Nanya. TITAC-2: An asynchronous 32-bit microprocessor based on scalable-delay-insensitive model. In *Proc. International Conf. Computer Design (ICCD)*, pages 288–294, Oct 1997.

[4] 岩田 靖. 分岐パス情報に基づく分岐予測方式の提案. 計算機アーキテクチャ, Vol.117, pp.61-66, Mar 1996