スーパスカラプロセッサにおけるレジスタ分割方式

7X-1

丸島 敏一　　西 直樹
日本電気(株)　C&Cシステム研究所

1 はじめに

RISCプロセッサを中心とした近年のスカラプロセッサの高速化に伴い、より高速なスカラ処理を目指して、命令レベルの並列性を利用したアーキテクチャが各所で検討されている[1]。特に、所謂スーパスカラプロセッサはVLIWのように1サイクルに複数の命令を発行することができ、さらにオブジェクトレベルでの従来方式との親和性があることから注目されている[2]。しかしながら、スーパスカラプロセッサでは複数の同時書き込みを可能とするレジスタセットを必要とし、また、実行時にデータ依存関係を解析していくための時間的コストが高いという問題点がある。本稿ではこれらを解決するためにレジスタ分割方式を提案し、その得失について評価する。

2 スーパスカラプロセッサ

本稿で扱うスーパスカラプロセッサは、以下の特徴を持つものと仮定する[3]。

(1) パイプライン化された複数の処理ユニット

(2) 複数命令同時発行 (multiple instruction issue)

(3) out-of-order 実行

(4) 分岐予測機構

(1)(4)はメインフレームに、(2)はVLIWに、(3)はデータフロープロセッサに見られる技術である。

スーパスカラプロセッサで特に問題となるのは、1つは、(2)でN個の命令を同時に発行しようとすると、2 Nread-Nwrite可能な共有レジスタセットが必要となることである。このことは通常ハード作成上の困難が伴うばかりでなく、性能的にも悪影響を及ぼす[4]。

もう一つの問題は、(3)でN個の命令のデータ依存関係を同時に解析しなければならないことである。Tomasulo方式[5]のout-of-order実行では命令発行に先立って、書込みレジスタを登録し、読出しレジスタの値の到着をチェックする。依存関係があるかもしれないN個の命令に対して、N個同時に依存関係を解析し、N個同時に書込み登録を行なうことは、時間的にもハード量的にもコスト高になる。

3 レジスタ分割方式

上記の問題点を解決するために、レジスタセットを分割し、各演算器毎に独自の別セットのレジスタセットを設ける。各々のレジスタセットは2 read 1 writeの機能を持ち、それ以上の要求は遅らせる。異なるセットのレジスタセット間の転送は機械語レベルで明示的に行なう。このレジスタ分割により、命令コードによって書込みレジスタセットを区別できるようになり、書込みレジスタセットの異なる命令同士は独立して並列に依存関係解析をできるようになる。

このレジスタ分割方式の利点は次のようになる。

- 複数命令同時発行のために高価な共有レジスタセットを使用することなく、見かけ上2 Nread-Nwrite機能を提供する。
- 実行時の依存関係チェックが簡潔化され、より高速なクロックサイクルを使用できる。

一方、欠点としては、

- 依存関係チェックに伴うコンパイラの負担が大きくなる。
- レジスタのコピー操作が機械語レベルで見えてしまう。また、コピー操作自体のオーバヘッドが生じ得る。
- オブジェクト量が増加する。

という問題がある。

4 性能比較

4.1 比較条件

レジスタ分割方式の評価をするために、以下の比較モデルを設定する(図1)。

(1) シングルパイプ・シングル発行 (SS方式)
(1命令パイプライン, 1命令発行/サイクル, 1書き込み/サイクル)

(2) シングルパイプ・マルチ発行 (SM方式)
(1命令パイプライン, 演算器毎同時発行可, 1書き込み/サイクル)

(3) マルチパイプ・マルチ発行 (MM方式)
(4命令パイプライン, 演算器毎同時発行可, 演算器毎同時書き込み可)

(4) レジスタ分割方式
(4命令パイプライン*, 演算器毎同時発行可, 演算器毎同時書き込み可)

*但し依存関係解析は 独立に可能なもののみ同時に処理

処理ユニットとしては、固定加算器系、固定乗算器系、浮動小数点加算器系、浮動小数点乗算器系とロードストアユニットを持ち、各々はパイプライン動作する。out-of-order実行としてはTomasulo方式を使用し、SS方式を除いて分散Reservation Station (各4エントリ)とした。これにより、複数同時命令発行が出来るものとする。MM方式とレジスタ分割方式では、複数同時書込みが可能なため、命令供給も複数パイプラインとした。但し、レジスタ分割方式では書込みレジスタセットの異なる命令同士のみ同時に依存関係解析するものとする。

評価に用いたオブジェクトコードは、リバモア14ループのNo.3 (総和) とNo.8 (偏微分方程式求解) をロード・ストア型の命令セットにハンドコンパイルしたものである(表1)。これらについて、CRAY-1のスカラプロセッサにおける処理時間[6] (例えば、浮動小数点加算6サイクル) に準じて机上シミュレーションした。また、分割レジスタセット間の転送は1サイクルとした。

Register Splitting for Superscalar Processors
Toshikazu MARUSHIMA and Naoki NISHI
NEC Corporation

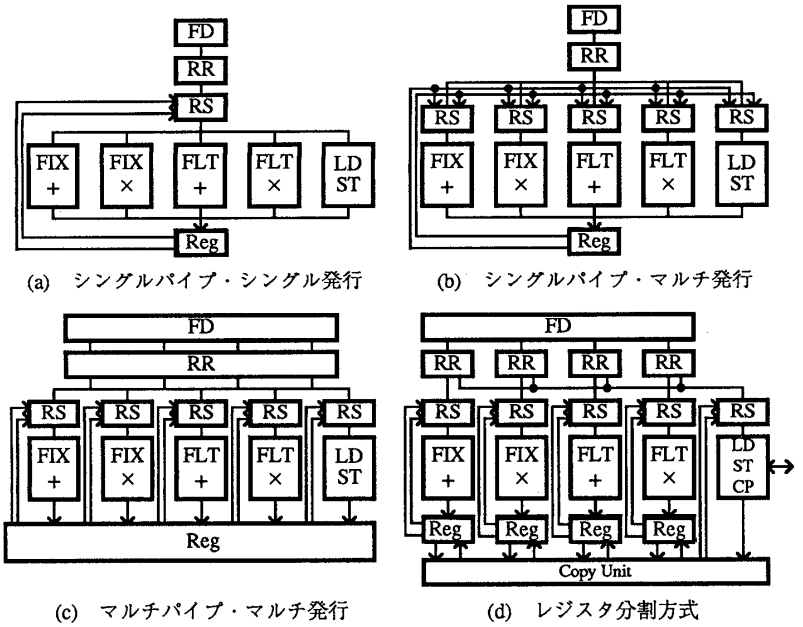

図1　性能比較モデル

表1　比較対象コードの命令数

	No.3 (×1000回)	No.8 (×20回)
従来コード	8	76
分割レジスタ向きコード	9	94
増加率	1.13	1.27

表2　評価結果

	No.3		No.8	
	総サイクル数	サイクル短縮率	総サイクル数	サイクル短縮率
SS方式	12511	base	1902	base
SM方式	12010	1.04	1601	1.19
MM方式	9013	1.39	964	1.97
レジスタ分割方式	10013	1.25	1243	1.53

4.2　結果

机上シミュレーションによる結果を表2に示す。これらより、レジスタ分割方式の実行サイクル数が、SS方式やSM方式のものよりも短縮されていることがわかる。また、MM方式の性能が良さそうに見えるが、これは10read5writeの共有レジスタセットが利用でき、さらに4命令同時に依存関係解析できると仮定したものである。実際のインプリメントを考慮すると、他のものに比較して遅いクロックサイクルでしか動作できないため、総合的には高速動作できないものと考えられる。

表1からオブジェクト量の比較をすると、プログラムの規模が大きくなると分割レジスタ方式のオブジェクト増加が顕著になっている。今回のように、複数命令パイプラインとすることにより命令供給のバランスを保つことが必要になるが、この条件が満たされれば、オブジェクト量が増すにもかかわらず高い性能を保持することができる。

5　おわりに

複数命令同時発行を可能とするスーパスカラプロセッサにおいて、レジスタ分割方式を提案した。これにより、低コストで見かけ上のレジスタ同時書込みが可能となり、また、各演算器が独自のレジスタを持つために実行時の依存関係解析が簡単化される。机上評価の結果、本方式ではオブジェクト量が増加するにもかかわらず、総合的に高い性能を示すことがわかった。しかしながら、レジスタコピーが見えてしまう使いにくさや、コンパイラに対する負荷の増加については今後検討すべき課題である。

謝辞　日頃御指導頂く日本電気C&Cシステム研究所大野部長、中崎課長に感謝致します。

参考文献

[1] R.Acosta,J.Kjelstrup,H.Torng:"An Instruction Issuing Approach to Enhancing Performance in Multiple Functional Unit Processors", IEEE Trans. on Computers, Vol.C-35, No.9 (1986)

[2] N.Joupi,D.Wall:"Available Instruction-Level Parallelism for Superscalar and Superpipelined Machines", ASPLOS III (1989)

[3] M.Smith,M.Johnson,M.Horowitz:"Limits on Multiple Instruction Issue", ASPLOS III (1989)

[4] M.Breternitz,A.Nicolau:"Tradeoffs between Pipelining and Multiple Functional Units in Fine-grain Parallelism Exploitation", Int. Conf. on Supercomputing (1989)

[5] R.Tomasulo:"An Efficient Algorithm for Exploiting Multiple Arithmetic Units", IBM J. of R&D (1967)

[6] R.Russell:"The CRAY-1 Computer System", CACM, Vol.21, No.1 (1978)