1B－4 ハードウェア記述言語による superscalar 及び VLIW プロセッサの設計とその比較*

山崎浩太、森本貴之、中村宏、朴泰祐、中澤喜三郎†
筑波大学 電子・情報工学系‡

1. はじめに

単一プロセッサにおいて、命令レベル並列処理を実現する手法として、superscalar 方式及び VLIW 方式 [1] [2] [3] が提案されている。

superscalar 方式は、複数命令を同時に実行し、高速化を実現するプロセッサである。1度に複数の命令を取り込んで、なるべく多くの演算器が並列に動作できるように供給する制御をハードウェアで行う方式である。

これに対して VLIW 方式は、あらかじめ並列に処理できる部分をまとめて1つの超長形式命令とし、ハードウェアによる解析をより簡単に実現したものである。

この2つの方式を比較するため、本研究ではハードウェア記述言語 SFL [4] で、superscalar と VLIW プロセッサの方式設計を行った。SFL はハードウェア設計支援システム PARTHENON の動作記述言語であり、動作レベルの記述から論理素子を構成単位とする回路図を合成することを目的としている。これらプロセッサの設計、シミュレーションを通して、性能、設計記述量に関しての比較検討を行う。

2. 記述対象プロセッサ

ここで設計するプロセッサはいずれも命令パイプライン制御を行う。プロセッサの持つ演算パイプラインの本数によって、3種類のモデルに分類する。その分類については表1に示す。superscalar ,VLIW ともに表1に示した3種類のプロセッサを設計する。

表1: 各プロセッサモデルの持つ演算パイプライン

model	同時使用可能本数	パイプラインの種類
1	3	整数2, 小数1, ロードストア1
2	3	整数2, 小数1, ロードストア2
3	4	整数2, 小数2, ロードストア2

整数... 整数演算パイプライン
小数... 浮動小数点演算パイプライン
ロードストア... ロードストアパイプライン

ロードストア命令は、整数演算パイプラインを使用するため、各 model とも整数演算パイプラインとの組合せには制限が生ずる。整数演算のレイテンシは1 マシンサイクル(以下 MC)、浮動小数点演算のレイテンシは

field 1	field 2	field 3
整数演算	整数演算	浮動小数点演算
ロードストア		

図 1: VLIW model 1 のフィールド構成

3MC、ロード及びストアのレイテンシは 1MC(all cache hit を想定) と仮定する。

branch は整数演算命令とし、常に not taken の分岐予測をする。この予測が外れた場合のペナルティは 1MC となるようにしてある。delayed branch は採用しない。

同 model における superscalar と VLIW の違いは、命令のデコードと発行を行うパイプラインステージ (decode stage) に主に現れる。

2.1 superscalar

superscalar プロセッサは命令間の依存関係を動的に検出し、同時に発行する命令グループのスケジューリングを行う。

命令セットアーキテクチャは DLX アーキテクチャ [1] のサブセットとする。命令数は 13 である。このプロセッサは model 1,model 2 では最大3命令を同時に発行可能である。model 3 では最大4命令が同時に発行可能である。また model が変わっても、命令セットアーキテクチャは変わらないので、model 間でのコードの互換性がある。

2.2 VLIW

VLIW プロセッサでは同時に発行可能な命令をまとめて1つの超長形式命令とする。ここで設計する VLIW プロセッサの1つの命令のフォーマットは、model 1,model 2 では3つのフィールド、model 3 では4つのフィールドから構成される（図1)。

各フィールドには superscalar プロセッサと同じ命令が挿入されるとした。すなわち、superscalar プロセッサの命令を3個（model 3 では4個）連結し、1つの超長形式命令を構成する。model により、挿入できる命令の位置、命令長が異なるので、model 間のコードの互換性はない。

各フィールドに挿入できる命令の種類には制限がある。図1に model 1 の場合の制限を示す。また各フィールドの命令間には依存関係のないことが保証されておりハードウェアはその依存関係の検出を行わないものとした。命令間の依存関係により、同時に実行可能な命令が存在せず、フィールドに空きが生じてしまう時は、そのフィールドには nop 命令が挿入される。

*Design and comparison of superscalar and VLIW processors by using hardware description language

†Kohta Yamazaki, Takayuki Morimoto, Hiroshi Nakamura, Taisuke Boku, Kisaburo Nakazawa
‡Institute of Information Sciences and Electronics, University of Tsukuba

3. 考察

全6種類のプロセッサについてSFLを用いて全体の設計・記述を行いシミュレーションを行って、動作の正確さを確認した。本章では、シミュレーションと記述量に焦点を当てて検討を行う。

3.1 シミュレーション

SFLの各設計記述を動作シミュレータSECONDSを用いてシミュレーションを行い、性能を評価した。ターゲットプログラムにはSAXPYループを用いた。

3.1.1 ターゲットプログラム

SAXPYループは単精度のベクトルX,Yについて、$Y(i) = aX(i) + Y(i)$(a:定数)を行うループである。シミュレーションで用いるターゲットプログラムとして、superscalarプロセッサでは、命令レベルの並列性を考えず、単純にコーディングしたコード（最適化無）と各model用に最適化したコード(最適化有)を与える。VLIWプロセッサは各model用に最適化コード(最適化有)のみを与える。最適化はループアンローリング、ソフトウェアパイプライニング手法を用いて、手作業で行った。

各プロセッサごとに最適化したコードを用いるため、それぞれのプロセッサ用のターゲットプログラムはコードサイズが異なる。表2に各プロセッサに与えるコードのサイズを示す。

表2より、superscalarとVLIWの差は、最適化したコードにおいて、どのmodelともVLIWの方がサイズが大きい。この差はスケジューリングの際に同時に発行できる命令が見つからない時にnop命令を挿入するか否かの違いである。従って、スケジューリングの効率がコードサイズの差に現れる。つまり、アプリケーションとプロセッサの持つリソースとの兼ね合いによっては、VLIWのコードサイズが増大する可能性もある。

表2: ターゲットプログラムのコードサイズ

model	superscalar		VLIW
	最適化無	最適化有	最適化有
model 1	10	40	60
model 2	10	55	63
model 3	10	61	80

1wordを32bitとした時のword数

3.1.2 実行サイクル

シミュレーションにより各プロセッサでベクトル長が32の場合のSAXPYループを実行するのにかかるMCの測定を行った。測定の結果は表3に示す。

表3: 実行サイクル数(MC)

model	superscalar		VLIW
	最適化無	最適化有	最適化有
model 1	420	116	116
model 2	420	80	81
model 3	420	68	69

表3より最適化有コードではsuperscalar,VLIWの性能の差はほとんどないことが分かる。これはall cache hitの仮定をしているため、あらかじめ演算のレイテンシを正確に予測でき、動的に発生する依存関係が存在しないからである。しかし、演算のレイテンシが可変の場合では、動的にデータ依存性を検出できるsuperscalarの方が、柔軟なスケジューリングができ、性能に差が現れてくると思われる。

3.2 記述量

SFLで設計・記述をしたプロセッサの記述量を表4に示す。設計した記述についての論理合成は行っていないが、これらの記述量がプロセッサのハードウェア量を測定する上で、ある程度の目安になると思われる。

表4: 各プロセッサの記述量(行数)

model	superscalar	VLIW
model 1	669 / 2391	300 / 1984
model 2	669 / 2468	365 / 2134
model 3	939 / 2996	407 / 2403

decode stageの記述量/全体の記述量

この表より、VLIWの方が記述量が少ないことが分かる。特にdecode stageでは約2倍の差が出ている。VLIWにおいては、命令のフィールドに入る命令が固定されていること、さらにフィールド間の依存関係を検出しないためである。また、superscalarでは動的に命令のスケジューリングを行うことがあげられる。全体の記述量ではVLIWの方が、約20 %ほど少ない。

4. おわりに

ハードウェア記述言語を用いてsuperscalar,VLIWを設計し、シミュレーションを行い性能を評価した。superscalarとVLIWでの差はほとんどなかった。

記述量は、VLIWの方が20 %ほど少なくて済むことが分かったが、その一方で、superscalarの持つオブジェクトコードの互換性という利点を失っている。記述量は定量的に評価できるが、互換性という利点を定量的に評価することは難しい。

今回の研究では、cache、命令セット及びターゲットプログラムは簡単なものであった。また、論理合成を行わず、単に記述量のみの比較であった。今後は、より現実的な仮定のもとでの検討、論理合成を行うことにより記述量だけでなく回路素子量やクロック周波数も含めた評価、及び互換性という利点の評価方法、に関して研究を進めていく予定である。

謝辞

本研究に関し貴重な御意見をいただいた筑波大学西川博昭助教授並びに中澤研究室諸氏、PARTHENON仕様環境を提供して頂いた日本電信電話株式会社コミュニケーション科学研究所の皆様に深く感謝します。なお、本研究は一部文部省科学研究費(奨励研究(A) 06780228)による。

参考文献

[1] J.L.Hennessy, D.A.Patterson "Computer Architecture:A Quantitative Approach" Morgan Kaufmann Publishers,Inc (1990)

[2] M.Johnson "Superscalar Microprocessor Design" Prentice Hall(1991)

[3] 竹内陽一郎 "Superscalar と Very Long Instruction Word:どこが super, なぜ very long?" 情報処理 vol.35,no.12 p1128-1129, 情報処理学会(1994)

[4] 中村行宏、小野定康 "ULSIの効果的な設計法" オーム社(1994)