計算機アーキテクチャ　113－7
（１９９５．　８．　２３）

投機的実行を支援するアーキテクチャのハードウエア設計

原　哲也　安藤秀樹　中西知嘉子　中屋雅夫

三菱電機（株）
システムＬＳＩ開発研究所
〒664　兵庫県伊丹市瑞原 4-1
e-mail:hara1@lsi.melco.co.jp

我々は、プレディケーティングと呼ぶ、制限のない投機的命令移動を可能にするハードウエア支援を提案している。このプレディケーティング方式を実現するプロセッサ“SPEV”は、4命令発行の VLIW マシンである。スカラ・マシン、投機的実行のハードウエア支援なしの VLIW マシン、SPEV のハードウエアの設計を行い、遅延解析ツールを用いて処理時間を調べた。その結果、VLIW マシンは、複数分岐命令実行と分岐予測による分岐処理時間の延びと、バイパス処理時間の延びが原因でスカラ・マシンに対して14%(1.1ns) サイクル時間が長くなる。SPEV マシンは、プレディケート付きレジスタ・ファイルが２つのデータ領域を持つため、データ読み出し時間が VLIW マシンのものより遅くなるが、プレディケート評価回路は単純であるためこれを含む分岐処理時間は VLIW マシンのそれよりも短くなり、SPEV のサイクル時間は VLIW マシンより僅かに (0.1ns) 延びるだけであることが分かった。

Hardware Design for an Architecture with Unconstrained Speculative Execution Support

Tetsuya Hara　Hideki Ando　Chikako Nakanishi　Masao Nakaya

Mitsubishi Electric Corporation
System LSI Laboratory
4-1 Mizuhara, Itami, Hyogo, 664 Japan
e-mail:hara1@lsi.melco.co.jp

We have proposed a new mechanism, called predicating, which removes restrictions that limit the compiler's ability for speculative execution. SPEV machine which supports predicating is a 4-issue VLIW machine. We compare the cycle time of a scalar machine, a VLIW machine without any hardware support for speculative execution, and SPEV machine using a static critical-path analyzer. Our evaluation shows that the cycle time of the VLIW machine is 1.14x longer than the cycle time of the scalar machine, due to increase of hardware complexity for branch handling (multi-way branch and dynamic branch prediction) and for bypass handling. SPEV machine contains a predicated register file which requires dual data-field. Although Predicated Register file makes register-reading time slower, the simple predicate evaluation logic compensates the register-reading extra time. Consequently, SPEV machine has little cycle time penalty against the VLIW machine.

1 はじめに

マイクロプロセッサは、動作周波数の向上と1サイクルで実行できる命令数を増やすことによってその性能を上げることができる。命令レベル並列処理は、複数の機能ユニットを搭載して、1サイクルで実行できる命令数を増やすことにより性能を向上させるものである。最新のマイクロプロセッサ [1] は、種々のスーパスカラ技術を用いて投機的実行[1]を行い、高い性能を実現している。しかし、命令の並列性を引き出すためのハードウエアが複雑な論理を必要とし、サイクル時間を長くするという欠点がある。

これに対して、VLIW (very long instruction word) マシンは命令のスケジューリングをコンパイラが行うため、単純なハードウェアしか必要とせず、サイクル時間に悪い影響を与えない。しかしながら、投機的実行の副作用をコンパイラだけで完全に処理することができないため、コンパイラだけで投機的実行を実現する単純な手法 [3] では高い並列性を達成することは難しい。

我々は、**プレディケーティング**と呼ぶ、制限のない投機的命令移動を可能にするハードウエア支援を提案している [4] [5]。プレディケーティングは投機的命令移動に課せられていた制限をほとんど取り除くことができるので、コンパイラはプログラムに存在する並列性を最大限に引き出し、コードを最適にスケジューリングすることができる。実行サイクル数比較による評価を行った結果、プレディケーティング方式は、スカラ・マシンの2.45倍の性能を得ることができた。これは、投機的実行のハードウエア支援がないVLIWマシンの性能(1.78倍)を大幅に上回り、プレディケーティング方式は高い並列性を得ることができることを確認している。

一方、動作周波数であるが、単純なVLIWプロセッサと同程度の動作周波数を達成できるように、プレディケーティング方式の支援機構は単純なハードウエアで実現できるように考えられている。

本稿では、プレディケーティングを実現するプロセッサ“SPEV(Speculative Execution VLIW)マシン”について、そのハードウエア構成、パイプライン処理過程を説明した後、VLIWとSPEVのハードウエア・オーバヘッドについて述べる。

2 SPEVマシンのアーキテクチャ

我々の方式では、全ての命令はプレディケートを持つ。即ち、命令は次のような形式を持つ。

プレディケート ? 操作

命令のプレディケートは、ブール値を持つ分岐条件の論理式である。プレディケートが真であるときのみ、操作部で示された操作の結果が有効となる。

1. 非数値計算応用のプログラムでは、基本ブロック内の命令レベル並列度は非常に小さく、制御依存が並列性を制限しているので [2]、投機的実行を行うことにより制御依存を取り除くことが、大幅な性能改善のためには必須である。

マシンは投機的実行を支援するために、2つのマシン状態を持つ。1つは、制御依存が解消している命令の実行結果によって作られる**逐次的状態**で、もう1つは、制御依存が解消していない命令の実行結果によって作られる**投機的状態**である。

命令のオペランドがどちらの状態にあるかは、コンパイラが明示的に、レジスタ番号に .s というサフィックスを付加して表す。

命令の発行点では、まず、プレディケートが評価される。その結果、値が真であれば、通常のマシンの命令と同様に、実行を行い逐次的状態を更新する。もしも、プレディケートの値が偽であれば、単純に命令は無効化される。これらのいずれでもない場合、即ち、プレディケート評価に必要な分岐条件の少なくとも一つが未定義であるために、プレディケートの値が未定値の場合は、命令の実行を行い投機的状態を更新する。更新の際、後のコミット制御のためにプレディケートも書き込む。

我々は、ハードウェア量と信号遅延時間の観点から、プレディケートに許される論理式として、否定を含む分岐条件の論理積に限定した。プレディケートのエンコーディングとしては、それぞれの要素が各分岐条件に対応するベクタとし、各要素にプレディケートを真にするために必要なブール値を持たせる。分岐条件に「冗長」も許すので、例えば、4つの分岐条件 c0,c1,c2,c3 に対してプレディケート c0&!c2&c3 は {1,X,0,1} とエンコードする。分岐条件が「冗長」の場合、プレディケートの評価では、その分岐条件に対する一致演算の結果はマスクする。これにより、プレディケート評価の論理は、単純に、CCR とのマスク付き一致演算で行うことができる。

プレディケーティングは、投機的結果をプレディケートによってラベルリングし、レジスタ・ファイルあるいはストア・バッファにバッファリングする。バッファの各エントリは、プレディケートの記憶とその値を計算するハードウェアを持ち、投機的実行の副作用を処理する。

レジスタ・ファイルの各エントリは、逐次的データと投機的データの2つのデータを記憶するために、2つのレジスタ（**逐次的レジスタ**と**投機的レジスタ**）を持つ。各エントリは、さらに、投機的データのコミット条件であるプレディケートを保持する。即ち、投機的書き込みの際には、プレディケートも同時に書き込む。

論理的には、各逐次的レジスタには異なるプレディケートを持つ複数の投機的状態が存在しうるので、複数の投機的レジスタが必要である。しかしながら、我々はハードウェア量を抑えるために各逐次的レジスタには唯1つの投機的レジスタを与えることとした。この制限によって、異なるプレディケートを持つ結果の書き込みの間で競合が生じるが、この競合による性能低下は僅かに0.1%であり、性能にほとんど影響を与えない。

メモリにおける投機的状態は、データ・キャッシュの前に置かれるストア・バッファに記憶する。ストア・バッファはFIFOで構成し、ストア操作が逐次的か投機的かにかかわらず、データはストア・バッファに一旦書き込まれ、FIFOの先頭のデータが有効な逐次的データであ

れば、データ・キャッシュに書き込まれる。レジスタと同様に、各エントリは投機的データに対してはプレディケートを共に保持し、状態遷移を制御する。

3 SPEV マシンの概要

本章では、SPEV マシンの構成と命令パイプライン処理過程について述べる。

3.1 SPEV マシンの構成

SPEV は非数値計算をターゲットとした試作プロセッサであるため、整数演算の処理のみを行い、浮動小数点演算、および、整数の乗算、除算はハードウエアでサポートしない。

算術論理演算命令およびロード・ストア命令は、MIPS R3000 の命令セットに準拠している。命令には実行条件があるため、分岐命令には条件分岐や無条件分岐などの区別がなく J(絶対アドレス指定)、Jrel(相対アドレス指定)、JR(Jump Register) の 3 つとしている。また、分岐条件を設定する分岐条件セット命令を新たに設けている。

SPEV のハードウエア構成を図 1 に示す。以下に SPEV 特有のユニットについて説明する。

(1) 実行状態制御ユニット

EXC および MEM ステージに存在する命令の実行状態の制御を行う。各命令の実行条件であるプレディケートとそのサイクルでの分岐条件を参照して評価を行い (**プレディケート評価**と呼ぶ)、命令の実行状態を設定する。

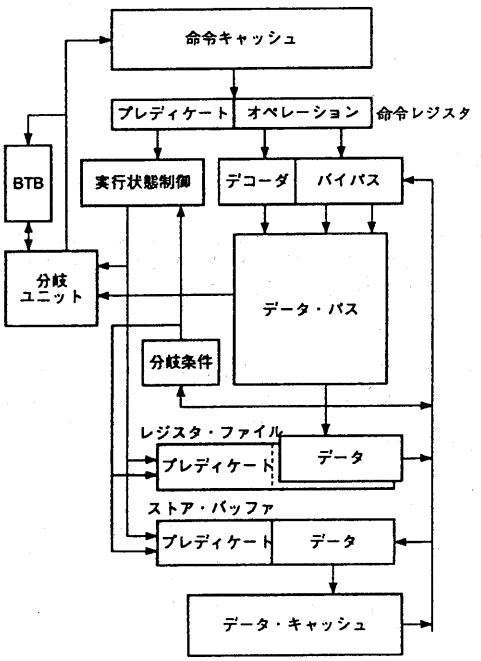


図 1　ハードウエア構成

プレディケートの評価値が真であればマシンの逐次的状態を更新する**逐次的実行状態**に、偽であればその実行結果をマシン状態に反映させない**無効状態**に命令の実行状態を割り当てる。未定値の場合は投機的状態を更新する**投機的実行状態**に割り当てる。これらの実行状態はここで一括管理する。

(2) 分岐条件ユニット

マシンの分岐条件を示す CCR(Condition Code Register) の設定を行う。CCR の値はデータ・パスで計算される。ただし、各パイプは処理する CCR が固定されており、CCR0 のセット命令はパイプ 0 に、CCR1 はパイプライン 1 に投入されるように命令コードはスケジューリングされている。

(3) プレディケート付きレジスタ・ファイル

図 2 にレジスタ・ファイルの構成を示す。レジスタ・ファイルのエントリ数は 32 である。各エントリにおいて、32bit 長の 2 つのデータ・フィールドの一方が逐次的レジスタであり、他方が投機的レジスタである。どちらが投機的レジスタであるかをフラグ W が示す。フラグ V は、投機的レジスタが有効な値を保持していることを示す。さらに、フラグ E は、処理されていない投機的例外が存在することを示す。また、パイプライン内で解消されなかった投機的状態の実行結果はレジスタ・ファイル内で更新を行うため、各エントリは、常に分岐条件を参照し、格納されたプレディケートを評価するハードウェアを持つ。ポート数は、書き込み 4 、読み出し 8 である。

(4) プレディケート付きストアバッファ

4 エントリのストアバッファで、各エントリはストア・アドレスとデータ以外にプレディケートとその評価回路を備える。

3.2 処理過程

命令パイプラインは、IF(命令フェッチ)、ID(命令デコード)、EXC(実行)、MEM(メモリ・アクセス)、WB(レジスタ書き込み) の 5 ステージ構成である。各命令のパイプライン処理過程を以下に説明する (図 3)。

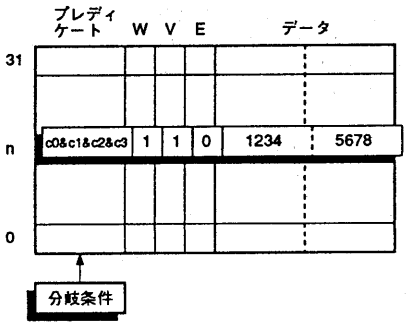


図 2　プレディケート付きレジスタ・ファイル

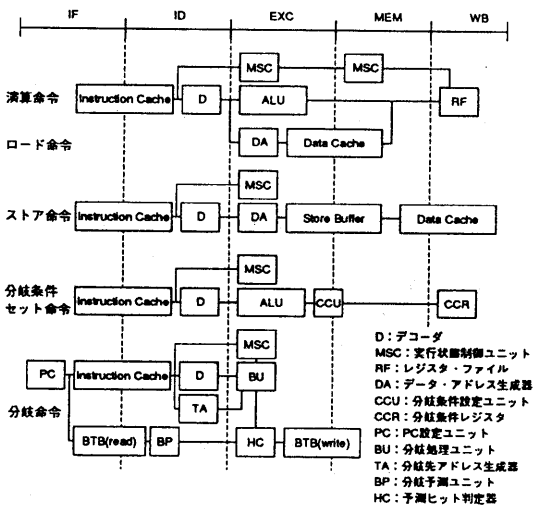


図3　パイプライン処理過程

3.2.1 演算命令およびロード命令

通常の命令フェッチ、デコードを行った後、EXC ステージでの ALU 実行（演算命令）あるいはアドレス生成（ロード命令）と並行して、命令の実行状態の設定を行う。プレディケート評価を行いその評価値が、真であればマシンの逐次的状態を更新する逐次的実行状態に、偽であればその実行結果をマシン状態に反映させない無効状態に、未定値の場合は投機的状態を更新する投機的実行状態に命令の実行状態を割り当てる。MEM ステージにおいても同様のプレディケート評価を行い、命令の実行状態の更新を行う。WB ステージでのレジスタ・ファイルへの書き込みは、書き込みを行う命令が逐次的実行状態であれば、その実行結果を逐次的レジスタに書き込む。一方、投機的実行状態であれば、投機的レジスタに実行結果を書き込むと同時にフラグ V をセットする。どちらのフィールドが逐次的レジスタ（投機的レジスタ）かの判断はフラグ W を参照することによって行う。レジスタ・ファイルの書き込みに引き続き、プレディケートの評価と投機的状態の更新を全エントリに対して行う。プレディケートが真と評価された場合、フラグ W を反転することによって投機的データのコミットを行う。偽と評価された場合、フラグ V をリセットすることによって投機的データの無効化を行う。未定の場合は状態の変化はない。

3.2.2 ストア命令

EXC ステージ前半までロード命令と同様の処理を行った後、無効状態でないストア命令はアドレス、データ、プレディケートをストア・バッファに書き込む。ストア・バッファも各エントリ毎にプレディケートの評価を行い、投機的状態の更新を行う。ストア・バッファはその先頭エントリが逐次的状態であれば、データ・キャッシュへの書き込みを行う。

3.2.3 分岐条件セット命令

EXC ステージの ALU 実行までは演算命令と同様処理を行い、コンディションを生成する。EXC ステージ終わりで、生成したコンディションを分岐条件に設定する。分岐命令が実行され他のリージョンに移行する場合は、もとのリージョンの分岐条件をリセットする。WB ステージでは設定した分岐条件を用いて CCR の更新を行う。

3.2.4 分岐命令

命令フェッチ、BTB を用いた分岐予測、および、デコードを行った後、EXC ステージでのプレディケート評価の結果を用いて分岐方向を決定する。プレディケート評価値が真ならその分岐命令は実行される命令であり Taken 実行となる。評価値が偽であればその分岐命令は無効であり Not-taken 実行となる。分岐命令は投機的状態にならぬようにスケジューリングされているので、評価値が未定ということはない。この分岐の実行結果と分岐予測結果から予測ヒット判定を行い PC の設定を行う。

4 ハードウエア

制限のない投機的命令移動を実現するために、プレディケートの評価などのハードウエア支援機構を SPEV は実装する。これらの機構は、ハードウエア・オーバヘッド、特に、マシン・サイクル・タイムを延ばさないように設計を行っている。

本章では、SPEV の設計を行い遅延解析を行った結果明らかになったクリティカル・パスである、レジスタ・ファイル、分岐処理系、バイパス処理系について、スカラ・マシン、および、SPEV と同じ機能ユニット構成でハードウエア・サポートを行わない VLIW マシンとの比較を行う。

4.1 評価条件

(1) モデル

スカラ・マシン

MIPS R3000 のアーキテクチャを実現するプロセッサ。1 命令発行で投機的実行のサポートはなく、分岐ペナルティは遅延分岐方式で対処する。

VLIW マシン

MIPS R3000 の命令セットを実行する、4 命令発行のプロセッサ。投機的実行のサポートはなく、2 つの分岐命令を同時に実行できる。分岐ペナルティへの対処は BTB 方式 [6] で行う。

SPEV

我々の評価結果 [4] によると、命令発行数は 4 で十分であるので（ループ展開を行わない場合）、SPEV は 4 命令発行とし 4 本のパイプラインを持つ。1 サイクルに発行可能な命令数は、算術論理演算命令が 4、ロード命令が 2、ストア命令が 1 とする。

プレディケーティングでは、ハードウエアで定めた分岐

条件の数だけ分岐条件を越える投機的命令移動を許す。分岐条件の数による性能を調べたところ、分岐条件数8を基準として、分岐条件数を4にしてもわずか1%しか性能は低下しないが、分岐条件数を3にすると14%性能が低下する。よって、分岐条件数は4とした。分岐条件数を3から4にするとプレディケート評価の処理時間が増加するが、1個の3入力ゲートが4入力になるだけで、その増加は僅かである。

分岐命令については、同時に実行できる分岐数を2から4にすると2.5%性能が向上するが、分岐処理時間の増加と分岐先アドレス計算器やアドレス・セレクタの増加などのオーバヘッドが大きい。このため、分岐命令数は2とする。

分岐ペナルティへの対処はBTB方式で行う。

図4にSPEVのフロア・プランを示す。

(2) 遅延の測定方法

トランジスタ(Tr)数から各ユニットの面積を見積り(一部はレイアウトにより求めた)、これに基づき配線容量を計算し、論理設計を行った回路に反映させて遅延値を求めた。使用プロセスは0.5μmCMOS 3層配線である。遅延値はEPIC社製の静的遅延解析ツール「PathMill」を使用して求めた。PathMillの精度はSpiceとの差が±10%以内である。キャッシュおよびBTBのアクセス時間は6.0ns、32ビットの加算が4.0nsとする。

4.2 レジスタ・ファイル

スカラ・マシンのレジスタ・ファイルは、2-read 1-writeのポート構成となるので、データの1ビット分に相当する**データ・セル**は9Trで構成できる。また、レジスタ・ファイルのアドレス・デコード回路も読み出し2、書き込み1で済む。

これに対して4命令発行のVLIWマシンでは、レジスタ・ファイルに8-read 4-writeポートが要求され、データ・セルは24Tr構成となる。さらに、アドレス・デコード回路も読み出し8、書き込み4が必要となる。

プレディケート付きレジスタ・ファイルを実現するには、同じ命令発行数のVLIWマシンのレジスタ・ファイルに対して図5で示すような、

- プレディケート保持および評価回路
- レジスタの状態フラグおよびその制御回路
- データ・フィールドのどちらに書き込み(読み出し)を行うかを指定する選択回路
- 投機的レジスタ

の追加ハードウエアが必要となる。上記に述べた追加のハードウエアのうち投機的レジスタを除く回路をSPEVのレジスタ制御回路とする。

図6にデータ・セルの構成を示す。SPEVは2つのデータ・フィールドを持つので、VLIWマシンの2倍の48Trが必要となる。

4.2.1 面積

表1に各レジスタ・ファイルにおけるTr数と面積を示す。面積は、データ・セルのアレイによりなる**データ領域**に対しては実際にレイアウトを行った結果であり、アドレス・デコーダ、レジスタ制御回路に対しては8000Tr/mm^{2}のTr密度を適用して算出している。

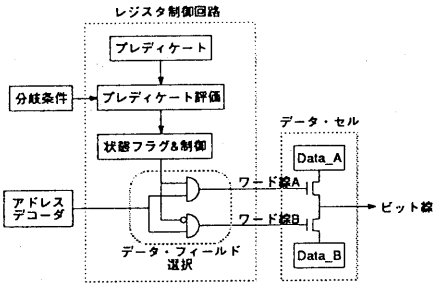


図5 プレディケート付きレジスタ・ファイルのハードウエア

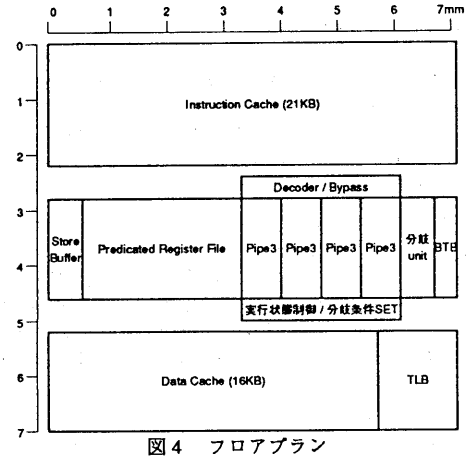


図4 フロアプラン

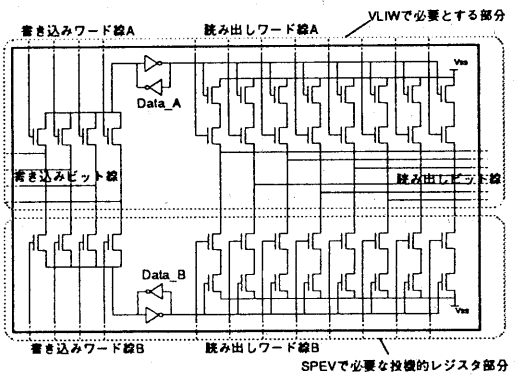


図6 プレディケート付きレジスタ・ファイルのデータ・セル構成

表1　レジスタ・ファイルのTr数と面積

	アドレス・デコーダ		レジスタ制御		データ領域		全体		
	Tr数	面積(mm²)	Tr数	面積	Tr数	面積	Tr数	サイズ(μm)	面積
スカラ	1618	0.20	-	-	9216	0.97	10834	1044x1120	1.17
VLIW	6272	0.78	-	-	24576	1.52	30848	1309x1760	2.30
SPEV	6272	0.78	11072	1.38	49152	2.37	66496	1655x2742	4.54

データ領域について

データ・セルはトランジスタ数を少なくし面積を抑えるためにビット線をドライブするトランジスタはサイズを小さくできるNチャネルTrのみで構成するが、データ保持にはインバータを使用するので、PチャネルTrが必要となる。スカラ・マシンのデータ・セルは、1セルあたりのTr数が少なく、ラッチアップ[7]防止のために設けるNチャネルTrとPチャネルTr間のマージン・スペースが占める割合が大きくなるのでTr密度が低くなっている。これに対してVLIWマシンではビット線のドライブTr数が多い、つまり、NチャネルTr数が多く、高密度に実装できるた。このため、データ・セルのTr数はスカラ・マシンの2.7倍であるが、その面積を約1.5倍に抑えることができる。SPEVのデータ・セルは、VLIWマシンの2倍のTrが必要であるが、VLIWマシンに比べNチャネルTrがさらに多く、ラッチアップ防止のスペースが占める割合がさらに小さくなるので、その面積をVLIWマシンの約1.5倍に抑えることができる。

アドレス・デコーダ、レジスタ制御回路について

VLIWマシンとSPEVはスカラ・マシンの4倍のポート数を持つため、アドレス・デコーダのTr数は4倍弱となっている。

さらにSPEVではレジスタ制御回路としてアドレス・デコーダの約2倍のハードウエアが必要である。これは、レジスタ制御回路は1エントリあたり364Trとそれほど多くないが、32エントリ分必要であることがハードウエア量が増える原因となっている。アドレス・デコーダやレジスタ制御回路はデータ・セルのように高密度に実装できないため、その面積は大きくなり、SPEVではデータ領域と同程度の面積が必要となる。

全体について

SPEVはVLIWマシンに比べ、データ領域では1.56倍程度であるが、レジスタ制御回路のオーバヘッドがあり、全体の面積では1.97倍となっている。しかしながら、SPEVの全体の面積(48.68mm²)からみると、VLIWに対する追加のハードウエア面積は4.6%となり許容できる量であると考えられる。

4.2.2 遅延時間

レジスタ・ファイルは、まず、入力されたソース・レジスタ番号のアドレス・デコードを行い読み出すレジスタ・エントリを決める。SPEVの場合はさらに、エントリのフラグの値に基づいて2本のワード線の一方をアクティブにし、2つのデータ・フィールド(逐次的レジスタと投機的レジスタ)の一方のデータをビット線に送る(データ読み出し)。ビット線はトライ・ステート・バッファ(TRB:Tri-State Buffer)の入力となり、このTRBがデータ・バスのソース・バスをドライブする。

表2に、アドレス・デコード時間、データ読み出し時間および、これらの合計であるレジスタ読み出し時間を示す。

表2　レジスタ・ファイルの処理時間

	アドレスデコード	データ読み出し	レジスタ読み出し
スカラ	2.03ns	0.97ns	3.00ns
VLIW	2.46	1.36	3.80
SPEV	2.68	1.71	4.39

VLIWマシンはスカラ・マシンに比べてレジスタ読み出し時間が18%(0.51ns)増加する。これは主に、ポート数を4倍に増やすことにより、データ・セルが大きくなりビット線の配線容量が増加したためである。

SPEVはVLIWマシンに比べてレジスタ読み出し時間が18%(0.58ns)増加する。そのうちわけは、アドレス・デコード時間が0.23ns、データ・アクセス時間が0.35nsである。アドレス・デコード時間の増加は、データ・フィールド選択処理の遅延によるもので、データ・アクセス時間の増加はデータ・フィールドが2つになったことでデータ・セルが大きくなり配線容量が増加したためである。

4.3 分岐処理系

スカラ・マシンは、分岐の条件判定として、2つのレジスタ値が一致するか、あるいは、1つのレジスタ値と0との大小比較を用意している。よって、分岐処理系のクリティカル・パスは、32ビットの一致比較→分岐判定→次フェッチ・アドレスの選択(パス①)、となる。

VLIWマシンは、スカラ・マシンと同じ条件判定を用い、さらに、2分岐実行、BTB方式を採用しているため、クリティカル・パスは、32ビットの一致比較→各分岐命令の分岐判定の後、どの分岐を実行するかの分岐セット→次フェッチ・アドレスの選択(パス①)、および、分岐予測ヒット判定→次フェッチ・アドレスの選択(パス②)の2つのパスとなる。

SPEVのクリティカル・パスは、プレディケート評価→各分岐命令の分岐判定を行った後、① どの分岐を実行す

表3　分岐処理時間

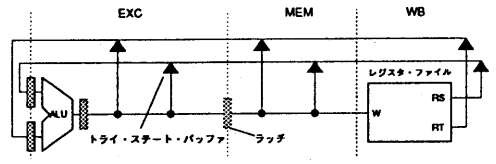
	分岐処理時間		
	スカラ	VLIW	SPEV
パス①	4.92ns	5.95ns	6.02ns
パス②	-	6.15ns	5.85ns

るかの分岐セット→次フェッチ・アドレスの選択(パス①)、および、分岐予測ヒット判定→次フェッチ・アドレスの選択(パス②)の2つのパスとなる。

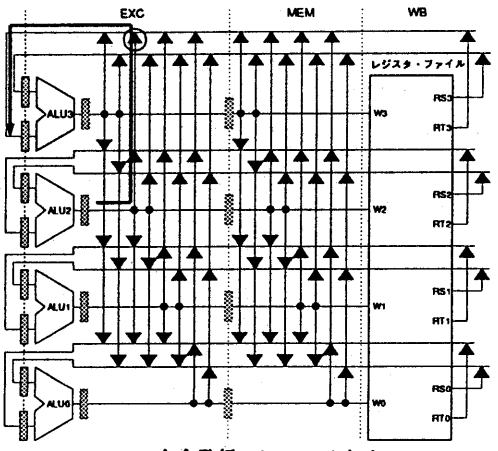
表3に、スカラ・マシン、VLIWマシン、および、SPEVの分岐処理系の処理時間を示す。

VLIWマシンはパス②がクリティカル・パスとなり、スカラ・マシンに比べて処理時間が25%(1.23ns)増加する。これは、BTB方式を採用しているため分岐予測のヒット判定のオーバーヘッドがあることが原因である。しかしBTB方式の採用をやめても、2分岐の並列実行を行うためにパス①がクリティカル・パスとなり、実行する分岐の選択とその分岐先の設定によって処理時間は21%増加となる。

SPEVはパス①がクリティカル・パスとなり、VLIWマシンとほぼ同じ処理時間となっている。これは、プレディケート評価に要する時間(2.96ns)がVLIWマシンの32ビット一致比較+分岐判定の処理時間とほぼ等しいためである。この結果より、プレディケートの評価を行うハードウェアは同じ機能ユニット構成のVLIWマシンに対して、サイクル・タイムを延ばすオーバヘッドとはならないことが分かる。



(a)　スカラ・マシンのバイパス



(b)　4命令発行マシンのバイパス

図7　バイパス回路

表4　バイパスの処理時間

	スカラ	VLIW	SPEV
バイパス処理時間	3.80ns	5.29ns	5.29ns

4.4 バイパス

各ステージの実行結果およびレジスタ・ファイルの読み出し線をトライ・ステート・バッファ(TRB)を介してデータ・パスのソース・パスに結合し、どのTRBをイネーブルにするかによって、データのバイパスを行う。

図7(a)にスカラ・マシン、図7(b)に4命令発行マシン(VLIW、SPEV)のバイパスを示す。例えば、ALU2のEXCステージでの実行結果は、図中の○でかこんだTRBをイネーブルにすることにより太線で示されるような経路を通りALU3の入力にバイパスさせる。4命令発行マシンでは、4つのパイプラインの3つのステージの実行結果が、各パイプラインの入力データとなるため、ソース・パスには9個のTRBが接続されるので、TRBの出力容量によってソース・パスの付加容量が大きくなる。さらに、9つの候補の中からEXC→MEM→レジスタ・ファイルの順番で優先順位付きの選択を行う必要があるので、そのロジックは複雑になる。

表4に、レジスタ番号が与えられてからソース・パスにデータが供給されるまでの遅延時間を示す。

VLIWマシンはスカラ・マシンに比べ、34%(1.33ns)処理時間が増加する。これは、バイパス選択処理がスカラ・マシンでは3つの中から選べば済むところを、VLIWマシンでは9個の中から選ばなければならず、その優先順位付き選択の処理時間が2.2倍になっていることと、ソース・パスの負荷容量が大きいのでバス・ドライブの遅延が1.5倍になっているからである。

SPEVとVLIWマシンのバイパスの違いは、レジスタ番号一致比較において、SPEVは実行状態の一致も調べるという点だけである。これは一致比較回路の構成法により遅延を増やさないようにできるので、SPEVとVLIWマシンの遅延時間は同じとなっている。

4.5 クリティカル・パスとサイクル・タイム

4.2-4.4節で調べた各パスの遅延時間より、各マシンのクリティカル・パスの遅延時間を求めた(図8)。

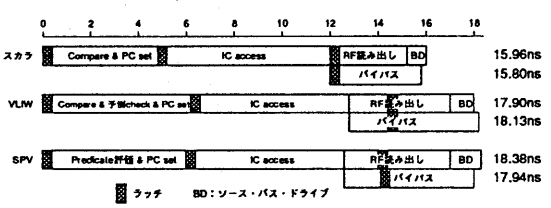


図8　クリティカル・パスと処理時間

図中のハッチング部分は信号がラッチを通過する時間である。

プロセッサのサイクル時間は、クリティカル・パス時間をその処理に必要なサイクル数で割った時間以上であり、かつ、ラッチを信号が通過する期間がクロックのイネーブル期間に入っているという、2つの条件を満たす時間で決定される。

クロックにはノンオーバーラップの2相クロックを使用する。クロックの立ち上がりエッジの1つの相はサイクルの始め、もう一方は中間点とし、イネーブル期間の幅はサイクル・タイムの1/4とする。

スカラ・マシンのクリティカル・パスはレジスタ・ファイル読み出しを含むパスで、その処理時間は15.96nsである。これが2サイクルに相当するので、サイクル時間は8.0nsとなる。

VLIWマシンのクリティカル・パスは、バイパスを含むパスであり、その処理時間は18.13nsである。これが2サイクルに相当するので、サイクル時間は9.1nsとなり、スカラ・マシンの14%(1.1ns)増しである。ただし、スカラ・マシンと同じパイプライン構成にするために、半サイクル(4.55ns)を越えるバイパス処理(5.29ns)の途中(オペランドと書き込みレジスタの一致比較を終えた点)にラッチを移動させている。また、レジスタ読み出しも同じ理由で、アドレス・デコード後にラッチを移動させている。

SPEVのクリティカル・パスはレジスタ・ファイル読み出しを含むパスで、その処理時間は18.38nsである。これが2サイクルに相当するので、サイクル時間は9.2nsとなり、スカラ・マシンの15%(1.2ns)増しであり、VLIWマシンとはほぼ同じである。VLIWマシンと同様にバイパスおよびレジスタ読み出し処理のラッチを移動させている。

プレディケーティングを支援するためにSPEVではプレディケート評価回路とプレディケート付きレジスタ・ファイルを設けている。プレディケート評価回路はシンプルであるため、これが含まれる分岐処理時間は、VLIWマシンのそれよりも短くなる。一方、プレディケート付きレジスタ・ファイルは、2つのデータ領域を持つため、データ領域の選択とレジスタ・アクセス時間が長くなり、データ読み出し時間が同じポート数を持つVLIWのレジスタ・ファイルより遅くなる。サイクル時間は、プレディケート付きレジスタ・ファイルの遅延を分岐処理時間の短縮でほぼ補うことができるので、VLIWマシンより僅かに(0.1ns)延びるだけである。

今回の評価では、VLIWマシンの分岐方式をcompare&branch方式としたが、branch-on-condition方式にすることで分岐処理時間を短くすることができる。ただし、分岐のパス長が増加するので実行サイクル数が増える。また、図8から分かるように、VLIW、SPEVは分岐処理、命令キャッシュ・アクセス、バイパス処理(あるいはレジスタ読み出し)の時間がほぼ等しい。そこで、それぞれの処理を1ステージに割り当てるパイプライン構成に変更することで、サイクル時間を短くすることができる。この場合、分岐ペナルティの増加により実行サイクル数が増える。したがって、それぞれの方式によるサイクル時間の短縮と実行サイクル数の増加を求めて検討する必要がある。

5 まとめ

本稿では、プレディケーティングを実現するプロセッサ"SPEVマシン"のハードウエア構成、パイプライン処理過程について説明し、スカラ・マシン、VLIWマシン、および、SPEVのハードウエア・オーバヘッドについて比較を行った。

ハードウエアの設計を行い、遅延解析ツールを用いて処理時間を調べた結果、スカラ・マシンに対してVLIWマシンは、複数分岐命令実行と分岐予測による分岐処理時間の延びと、バイパス処理時間の延びが原因で14%サイクル時間が遅くなる。

プレディケーティングを支援するSPEVマシンは、プレディケート付きレジスタ・ファイルが2つのデータ領域を持つため、データ読み出し時間がVLIWマシンのものより遅くなるが、プレディケート評価回路は単純であるためこれを含む分岐処理時間はVLIWマシンのそれよりも短くなり、SPEVのサイクル時間はVLIWマシンより僅かに(0.1ns)延びるだけであることが分かった。この結果より、プレディケーティングを実現するためのハードウエアが単純であり、プレディケーティング方式が有効であることを確認した。

参考文献

[1] 浅見 直樹、枝 洋樹,"次世代マイクロプロセサ、スーパスカラとVLIWが融合、" 日経エレクトロニクス、No. 627, pp.67-150, 1995年1月.

[2] D. W. Wall, "Limits of Instruction-Level Parallelism," In *Proc. Fourth Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, pp.272-282, April 1991.

[3] A. Nicolau, "Percolation Scheduling: A Parallel Compilation Technique," Computer Sciences Technical Report 85-678, Cornel University, May 1985.

[4] 安藤秀樹、中西知嘉子、原哲也、中屋雅夫、" プレディケート付き状態バッファリングによる投機的実行、" *1995年並列処理シンポジウム*、pp.107-114, 1995年5月.

[5] H. Ando, C. Nakanishi, T. Hara, and M. Nakaya, "Unconstrained Speculative Execution with Predicated State Buffering," In *Proc. 22nd Int. Symp. on Computer Architecture*, pp.126-137, June 1995.

[6] J. K. F. Lee, A. J. Smith, "Branch Prediction Strategies and Branch Target Buffer Design," *Computer* 17 (1), pp.6-22, January 1984.

[7] 富沢 孝、松山 泰男 監約,"CMOS VLSI設計の原理、" 丸善、1988年.