関数型プログラムの実行に適した マルチスレッド型プロセッサ・アーキテクチャに 関する研究

伊藤 英治

北陸先端科学技術大学院大学 情報科学研究科

1997 年 2 月 14 日

キーワード： マルチスレッド型プロセッサ・アーキテクチャ, 関数型プログラム, ハザード回避.

1 はじめに

MOS デバイスの微細化が進むと、プロセッサの動作クロックは配線遅延によって制限されることになる。

この状況下でプロセッサの高性能化を図るための 1 つの方法は、パイプラインを細分化することによって配線遅延を減少させ、動作クロックを向上させることである。また、細分化したパイプラインの性能を引き出すためには、単一ストリームだけを扱う従来のプロセッサよりも、同時に多数のストリームを扱うマルチスレッド型プロセッサが適している。

このような点から、今後の LSI 製造技術の進歩を生かすプロセッサ・アーキテクチャは、マルチスレッド型プロセッサ・アーキテクチャであると考えられる。

本論文では、マルチスレッド型プロセッサ・アーキテクチャを並列処理構造を持つ関数型プログラムの実行に適したアーキテクチャにすることによって、簡単なハードウェアで実現することができる並列処理のための高性能なプロセッサ・アーキテクチャを提案する。

2 マルチスレッド型プロセッサと関数型プログラムとの関係

関数プログラムの特徴は “多数の並列実行可能なストリームを供給することが可能である” という点である。

一方、マルチスレッド型プロセッサは、プロセッサ自身が持つ性能（パイプラインの高いスループット）を引き出すために、複数のスレッドを必要とする。

そこで、マルチスレッド型プロセッサに対して多数のスレッドを供給することによって、マルチスレッド型プロセッサの高性能化を実現することが可能になる。

3 マルチスレッド型ウルトラパイプライン・プロセッサ・アーキテクチャ

マルチスレッド型プロセッサ・アーキテクチャと関数型プログラムの特徴を組み合わせると、次の 2 つの事が可能になる。

1. 各パイプライン・ステージをすべて異なるスレッドからの命令で埋めることが可能

   パイプライン中の各々の命令が互いに依存関係を持たないので、データハザードや制御ハザードが発生しない。このため、プロセッサは 1 サイクル 1 命令の速度で命令を実行することができる。また、イプライン制御も非常に簡単化することができる。

2. パイプラインを長くすることが可能

   パイプラインを長くすることができるということは、パイプラインの各ステージを細分化し動作クロックを高速化することができ、パイプラインのスループットを向上させることができる。

上述の二つの利点を生かしたマルチスレッド型・プロセッサ・アーキテクチャが、本論文で提案するマルチスレッド型ウルトラパイプライン・プロセッサ・アーキテクチャ（MUP）である。MUP の特徴を以下に示す。

パイプライン・ステージ数と同数のレジスタセット プロセッサがパイプライン・ステージと同数のスレッドを同時に扱うことを可能にするために、パイプライン・ステージと同数のレジスタセットを用意する。さらに、構造ハザードを回避するために、レジスタセットは各スレッド毎に独立した形をとる。

スレッド ID 用パイプライン 発行された命令がどのスレッドから発行されたのかを識別するために、命令およびオペランドが流れるパイプラインとは別にスレッド ID が流れるパイプラインを持つ。

ラウンドロビン選択方式による実行スレッドの切り替え 各パイプライン・ステージをすべて異なるスレッドからの命令で埋めるために、ラウンドロビン選択方式による実行スレッドの切り替えを行なう。この切り替え方式によって、同一スレッドの命令実行は逐次実行の形態になる。

命令キャッシュとデータキャッシュの分離 ハーバード・アーキテクチャによって構造ハザードの発生を回避する。

キャッシュメモリとレジスタファイルのパイプライン化 MUP は、高速なクロックで動作するため、レジスタファイルやキャッシュメモリに対して高いスループットを求める。この要求を満たすために、レジスタファイルやキャッシュメモリのパイプライン化を行なう

複数スレッド間でのキャッシュメモリの共有 関数起動の時間的局所性を生かしキャッシュメモリのミス率を下げるためと、キャッシュメモリの容量性ミスの発生を減らすために、MUP では複数のスレッドで 1 つのキャッシュメモリを共有する形態をとる。

大容量キャッシュの搭載 MUP では、実行スレッドを毎サイクル切り替えるため、メモリアクセスの局所性がなくなるという問題が生じる可能性がある。この問題に対処するために、MUP は扱うスレッド数に応じた大容量のキャッシュメモリを持つ。

パイプラインをストールしないキャッシュミスの取り扱い MUP では、高いスループットを保つために、あるスレッドの命令がキャッシュミスを起こしてもパイプラインをストールすることなく他のスレッドの命令を実行し続ける。

これらの特徴によって、MUP はデータハザード、分岐ハザード、および構造ハザードを回避し、動作クロックの高速化による高いスループットを可能とする。

4 設計と性能見積り

4.1 設計

MUP の動作クロックとハードウェア量を見積もるために、具体的な MUP の設計を行なった。MUP の具体的な設計は以下の通りである。

命令セット： 整数除算・乗算命令、浮動少数点に関する命令を除く、一般的な命令を持つ。

ハードウェア構成： MUP を構成するハードウェア・ユニットは、スレッド選択ユニット、PC 群、命令キャッシュとデータキャッシュ、キャッシュミス処理ユニット、レジスタファイル群、符号拡張ユニット、2 ビットシフトユニット、演算ユニット、アドレス計算ユニット、および例外処理ユニットである。

パイプライン構成： パイプラインは、キャッシュメモリのメモリセルのアクセスタイムを考慮して、各ステージの論理回路の段数が 6 段以下になるように分割した。その結果、MUP のパイプラインは 17 ステージで構成することになった。

4.2 性能見積り

設計した MUP の動作クロックおよびハードウェア量を見積もるために、ゲートレベルでの論理合成を行なった。その結果、MUP のハードウエア量は次のようになる。

ゲート数： 48,705（フリップフロップ 5,059 個を含む）

レジスタファイルなどのビット数： 20,228

また、MUP をウェーハ上に実現する技術として、$0.1\mu m$ のプロセスルール、アルミ配線で設計するものと仮定し、$0.1\mu m$ デクノロジの回路パラメータを、“素子遅延：$12psec$”、“アルミ配線のシート抵抗：$0.6\Omega/sq$” および “アルミ配線の単位面積当たりの容量：$88.8fF/\mu m^2$” とする。これらの仮定から論理回路 1 段当たりの遅延を $114psec$ と見積もることによって、MUP は $1GHz$ のクロックで動作することができるといえる。

5 結論

本論文では、マルチスレッド型プロセッサ・アーキテクチャが LSI 製造技術の進歩を生かすプロセッサ・アーキテクチャであると考え、このプロセッサ・アーキテクチャと関数型プログラムの特徴を組み合わせることにより、並列処理のための高性能なプロセッサ・アーキテクチャであるマルチスレッド型ウルトラパイプライン・プロセッサ・アーキテクチャ（MUP）を提案した。

MUP は、スレッド選択ユニット、各スレッド毎のレジスタセット、命令キャッシュとデータキャッシュの分離などによって、データハザード、制御ハザードおよび構造ハザードを回避する。また MUP は、スレッド ID 用パイプライン、パイプライン化されたレジスタファイルやキャッシュを持つ。これらの特徴によって、MUP は高速な動作クロックによる高いスループットを実現する。

動作クロックとハードウェア量の見積りを行なうために、MUP を具体的に設計しゲートレベルでの論理合成を行なった。その結果、MUP はゲート数 “48k”、レジスタファイル等のビット数 “20k” で構成することができ、$1GHz$ のクロックで動作可能であることを示した。

今後は、プロセッサのパイプラインをどこまで細分化できるかという検討、およびプロセッサの総合的評価を行なう環境整備を行なう予定である。