1Q－8　命令ストリーミング：複数パスの投機処理に適した命令列構成方式

上田 英明, 中村 友洋, 吉瀬 謙二, 辻 秀典, 安島 雄一郎, 高峰 信, 坂井 修一, 田中 英彦
東京大学大学院 工学系研究科 *

1　はじめに

マイクロプロセッサの性能はデバイス技術やアーキテクチャの改良などにより向上し続けている。また将来のプロセッサにおいては、複数のパスを投機的に処理するものが考えられており、今後より高いフェッチ能力が要求される。

本稿では、メモリからの連続的転送という側面に着目し、複数の基本ブロックからなる命令列を単位とすることで、一度に多数の命令を読み込めるようにする。これによりメモリアクセス時のレイテンシの影響を減らし、スループットの向上が見込める連続的転送を活用することにより、命令フェッチ能力の向上を目指す。

2　メモリデバイス

2.1　レイテンシ

メモリデバイスは大容量化という要求のもと3年で4倍という Moore の法則に従い大容量化しているが、速度に関してはあまり大きな向上は見られない。この傾向は今後も続くと思われ、プロセッサの速度向上に比較してメインメモリのアクセス速度の向上はあまり望めない（図1）。このため、メモリアクセス時には速度ギャップから生まれるレイテンシが問題となる [2]。従来はプロセッサとメモリの速度差をキャッシュによって吸収してきた。しかし、プロセッサとメモリの速度ギャップが大きくなるにつれ、キャッシュ・ミスが発生した場合にはミス・ペナルティが大きくなる。そのためレイテンシを隠蔽することが必要になる。

2.2　連続転送

Rambus DRAM に代表されるデータの連続的転送によって、データ転送速度という点に関しては大幅に向上を見込むことができる。例えばシンクロナス DRAM は、アクセス時間に関しては DRAM と同程度の 60ns であるが、バースト転送時の動作周波数は 100MHz であり、通常の DRAM の動作周波数 20MHz に対して5倍速くなる。また、Rambus DRAM は最大アクセス時間が

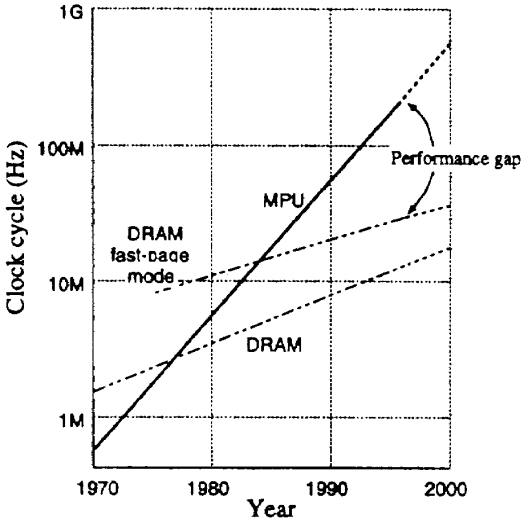

図 1: プロセッサとメモリ間の速度ギャップ

40ns であるのに対して、バースト転送時の動作周波数は 800MHz（サイクル時間 1.25ns）であり、DRAM に対して数十倍の周波数で動作し、スループットの向上をはかっている。これらのことより連続的転送を用いることで、スループットに関しては大幅な向上を見込むことができる。

3　ストリーミングの提案

3.1　ストリーミングの定義

ストリーミングとは、ストリーミング・コードを作成し、それに対してフェッチ操作を行なうことである。ストリーミング・コードは静的な段階で作成し、いくつかの基本ブロックをまとめたストリームの集合体からなる。

ストリームは基本ブロックを単位として、複数組み合わせて構成する。図2において1、2、3でくくられる丸はそれぞれ基本ブロックを示す。また、一つの矢印が一命令を示し、枝別れした矢印は一分岐命令を示す。1の基本ブロックにおける分岐命令によって、制御の流れは2と3の基本ブロックに分かれることになる。そこで2と3の基本ブロックを単一のストリームに含めることによって、両側のパスの命令を同時にフェッチできるようにする。このようにして、基本ブロックを単位として複数の基本ブロックからなるストリームを構成する。また、ストリームにはヘッダとして、ストリーム内の基本ブロックを管理するための情報と、次のストリームをフェッチ

---

* "Instruction Streaming:Instruction sequence adapted for speculative multi path processing"

Hideaki Ueda, Tomohiro Nakamura, Kenji Kise, Hidenori Tsuji, Yuichiro Ajima, Makoto Takamine, Shuichi Sakai Hidehiko Tanaka
University of Tokyo, School of Engineering, Graduate School of Engineering,
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

するために必要な情報をつけておく。

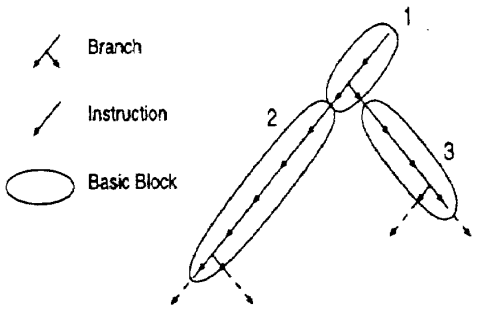

図 2: ストリームの構成方法

3.2 ストリームのフェッチ

現在フェッチされているストリームに含まれていない命令、すなわち、まだフェッチされていない命令をフェッチする場合には、次のストリームをフェッチして、コントロール・フロー・グラフを再構築することで行なわれる。ただし、必要な命令を含んだストリームだけをフェッチするとは限らない。実行される可能性がある命令を、事前に十分にフェッチするために、投機的にストリームをフェッチするという方法が考えられる。

3.3 ストリーミングによる利点

命令列が連続的に配置されていれば、連続的転送を活用することで、命令フェッチ能力の向上が期待できる。ストリーミングでは複数の基本ブロックをまとめて1かたまりにしたストリームを考え、命令列を十分長く連続化する。これによってメモリからの連続的転送を活用し、スループットの向上を得ることができる。また、ストリームは複数のパスを含んでおり、分岐がはずれた場合のペナルティを少なくすることができる。複数パスの命令を連続的転送を用いて一度にフェッチすることができるため、複数パスを投機的に処理するプロセッサに対して、効率的なフェッチ方法を提供することができる。

4 課題

ストリーミングにおいては、どのようにしてストリームを構成するか、またそのストリームに対してどのようにフェッチを行なうかということが課題となる。

1回のフェッチ操作でより多くの命令をフェッチすることを目的とするため、ストリーム単体のコード・サイズは大きくなる。ただし、いたずらに基本ブロックを多くしても、ストリーム中に占める有効な命令の割合は減少してしまい、結果的にムダにコード量が増えてしまう。これは分岐命令がある度に実行される可能性のあるパスは指数関数的に増えていくが、実際に実行されるパスは一つだけだからである。したがって、単純にストリームに含める基本ブロックを多くするわけにはいかない。また、間接分岐によって分岐先が特定できない場合にはその分岐命令から先の基本ブロックをストリームに含められなくなる。このとき間接分岐を含んだパスが実行される可能性が大きい場合には、それ以外の命令の部分は無駄なフェッチとなる。この点でも、単純にストリーム・サイズを大きくするわけにはいかないということがわかる。そこで、プロファイルを用いて分岐命令における分岐確率を求め、より実行される可能性が高いパスを優先的に含めることにより、より有効な命令からなるストリームを構築する方法が考えられる。

また、どのようにフェッチ操作を行なうかということであるが、基本的には次に分岐するストリームをフェッチすることになる。しかし、必要になった段階で該当するストリームのフェッチを開始すると、メモリのレイテンシの問題が発生してしまう。そこで、次にどのストリームの命令が実行されるかまだ確定していない段階で、投機的にストリームをフェッチする方法が考えられる。これによってメモリのレイテンシを隠蔽することで、フェッチ能力の向上が期待できる。

5 まとめ

本稿ではフェッチ能力を向上させるために、連続的転送を用いてメモリデバイスのレイテンシの問題を回避し、スループットの向上を目指すストリーミングを提案し、検討を行なう。

参考文献

[1] 中村友洋, 吉瀬謙二, 辻秀典, 安島雄一郎, 田中英彦. 大規模データパスプロセッサの構想, ARC124-3, pp. 13-18, 1997.

[2] Masaki Kumanoya, Toshiyuki Ogawa, Kazunari Inoue, Advances in DRAM Interfaces, IEEE MICRO, Vol. 15, No. 6, pp. 30-36, November 1995.

[3] Semiconductor Industry Association(SIA), *The National Technology Roadmap for Semiconductors*, 1994.

[4] 福田 昭, 枝 洋樹, "決断を迫られる高速DRAMへの切り替え", 日経BP, 日経エレクトロニクス 1995 7-31, pp. 99-125