非同期式プロセッサ TITAC-2 のキャッシュ構成

1G－5　石川 誠　桑子 雅史　山崎 淳　上野 洋一郎　南谷 崇†
東京工業大学 工学部　†東京大学 先端科学技術研究センター

1 はじめに

VLSI 製造技術の進歩によりチップ面積は増加し、素子速度は高速になっている。同期式回路は配線遅延の増大により、素子遅延に見合った高速なクロックをチップ全体に分配することが困難になってきている。そこで我々はクロックを用いない非同期式回路で、高速かつ実用的なプロセッサ TITAC-2 の設計を行なった [1]。

高速なプロセッサを構成するにあたって高性能な命令キャッシュは必須である。本稿では非同期式プロセッサ TITAC-2 に組み込まれた命令キャッシュの構成とその高速化手法の紹介をすると共に、ラインフェッチの方式、ラインサイズなどの設計要素が性能に与える影響について述べる。

2 命令キャッシュの方式

命令キャッシュにはキャッシュメモリの容量、連想度 (ウェイ数)、ラインサイズ、ラインフェッチ方式、外部データバスのビット幅など多くの設計要素があり、これらの選択はプロセッサの性能に大きな影響を与える。ここでは代表的なラインフェッチ方式として以下の 3 方式を扱う [2]。

1. Blocking 方式
2. Early Restart 方式
3. Non Blocking 方式

1 はラインサイズ分のデータをキャッシュメモリに転送し終わるまでプロセッサをストールさせる。2 はラインフェッチの終了を待たずに、目的のデータが転送され次第処理を再開する。その後ラインフェッチは命令キャッシュと並列動作するが、処理中のラインと異なるラインでキャッシュヒットした場合にはラインフェッチ終了までストールする。3 は 2 に加えてラインフェッチ中でもキャッシュヒットした場合にはプロセッサをストールさせずに動作する。

3 TITAC-2 への実装

3.1 非同期式回路に適した方式

同期式回路では最大遅延で動作サイクルが決定されるが、非同期式回路の性能は平均遅延で決定される [3]。TITAC-2 ではこの性質を利用してキャッシュメモリにラインサイズ分のバンド幅を持たせ、キャッシュヒット時には 1 ライン分のデータを同時に読み出す。その後同じラインを続けて使用する場合には目的のデータは既にキャッシュメモリから出力されているため、タグチェックおよびキャッシュメモリの読み出しを省き、マルチプレクサの選択だけで命令フェッチを行なう。タグチェックの必要性は上位モジュールが判断をし、必要な場合には "Tag Check 要求"、不必要な場合には "Select 要求" を命令キャッシュに送る。この方法ではキャッシュ動作の約 70%が高速なセレクト動作になり、動作時間の平均値を大幅に低減することができる。

3.2 構成とその動作

TITAC-2 はキャッシュ容量 8KB、連想度 1(ダイレクトマッピング)、ラインサイズ 8 ワード、Early Restart 方式の命令キャッシュを搭載し、32 ビットの外部データバスをもつ。命令キャッシュのブロック図を図 1に示す。

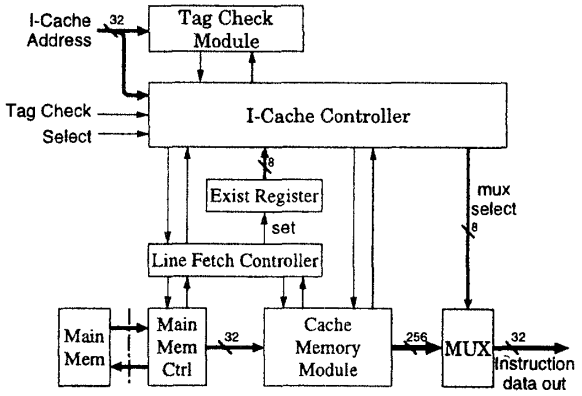

図 1: 命令キャッシュのブロック図

I-Cache Controller は命令キャッシュ全体を制御する。"Tag Check 要求" が発生すると Cache Memory Module の読み出し動作と Tag Check Module のタグ比較及び有効ビットの検査を同時に行ない、ヒットした場合には読み出し結果の出力を、ミスした場合には Line Fetch Controller に対してラインフェッチの要求を送る。Line Fetch Controller は Main Mem Ctrl を介して外部メインメモリから命令データを読み出し、Cache Memory Module に書き込みを行なう。

Early Restart 方式はラインフェッチを並列動作させるため、目的の命令データがキャッシュメモリに転送されていることを確認する手段が必要になる。TITAC-2 ではラインサイズと同じビット幅の記憶を持つ Exist Register を用意することでこの問題を解決した。Exist Register はラインフェッチの開始前にリセットされ、Line Fetch Controller によって 1 ワードの転送終了ごとに対応するビットがセットされる。I-Cache Controller はこのレジスタのセットを待ち合わせてマルチプレクサ (MUX) に選択信号を送る。"Select 要求" の発生時にはこの待ち合わせ動作のみを行なう。

3.3 ラインフェッチの高速化

キャッシュミス発生時にはラインサイズ分の命令データをキャッシュメモリに転送するが、この動作は内部回路の動作に比べて非常に遅い。またメインメモリはデータアクセスでも頻繁に使用されるため、その競合を避ける意味でもラインフェッチの高速化が重要である。

TITAC-2 ではメインメモリの読み出しデータを一旦ラッチすることでキャッシュメモリへの書き込みを並列動作させている。メインメモリの読み出しはキャッシュメモリの書き込みよりも遅いため、ラインフェッチ時間はメインメモリのアクセス時間のみに依存することになる。

3.4 メインメモリアクセスの高速化

TITAC-2 は基本的に 2 線 2 相方式 [3] で動作するが、外部メインメモリは同期式回路で用いられる SRAM を使用する。そのためメインメモリ周辺は任意のビット数のデータに時間情報を表す信号線を 1 本追加する束データ方式を採用し、Main Mem Controller、Line Fetch Controller は 2 線式データとのインターフェースを受け持つ。2 つのコントローラ間は 2 相方式の制御をするため、要求信号を出してから応答信号が帰るまでの稼働相、要求を取り下げてから

Instruction Cache for Asynchronous Processor TITAC-2
Makoto Ishikawa, Masashi Kuwako, Atsushi Yamazaki, Yoichiro Ueno
Faculty of Engineering, Tokyo Institute of Technology
Takashi Nanya
Reserch Center for Advanced Science and Technology, University of Tokyo

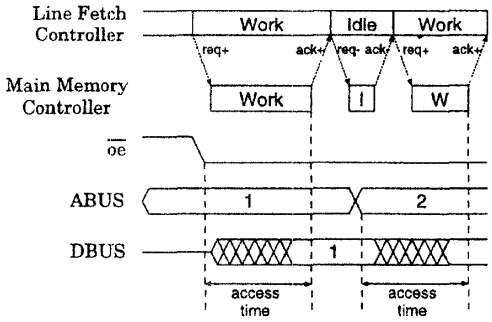

図 2: メモリアクセスのタイミング

応答信号が取り下げられるまでの休止相が存在する。しかし外部メインメモリには休止相が存在しないため、無駄な時間が生じる。これを改善するためにラインフェッチの期間は外部メインメモリの読み出し信号 ($\overline{oe}$) を 0 に固定し、アドレスバスを変化させることでメモリの読み出しを行なう。

図 2はメモリアクセスのタイミングを示したもので Work(W) は稼働相を、Idle(I) は休止相を表す。初回の読出しはアドレスバスの確定後、Line Fetch Controller の稼働相と共に始まり、読み出し完了後に Main Memory Controller が完了信号を返すことで休止相に入る。休止相の間に次のアドレスが確定するため、外部メモリは Main Memory Controller の稼働相の前に読み出し動作を開始する。この制御により外部メモリのアクセス時間を有効に使うことができる。

4 命令キャッシュの構成による性能変化

TITAC-2 ではパイプラインのステージ間ラッチに FIFO を採用し [1]、一時的に遅延が増加しても他のステージは FIFO 中に蓄えられたデータを使って動作を続けられる。この非同期式回路の特徴を活かせる条件下で、命令キャッシュの構成と性能の関係をシミュレーションによって調べた。シミュレーションではレジスタトランスファーレベルの命令キャッシュを使用し、連想度は 1 に固定、ラインサイズを 2、4、8、16、ラインフェッチ方式を 2節で述べた 3 種類、外部データバス幅を 32 ビット、64 ビットに変化させた。

4.1 シミュレーション環境

使用プログラムは動的辞書法による文字列の圧縮・展開を行なうものだが、キャッシュ容量を 8KB から 1KB に減少させ、[2] で統計に調べられているミス率[1]に近くなるように調整した。表 1はこのプログラムを実行した時のキャッシュの動作内容を示したものである。

表 1: 命令キャッシュの動作内容 (トータル 17227 命令)

ラインサイズ	2	4	8	16
Tag Miss	3316	1976	1272	799
Tag Hit	6248	3706	2187	1281
Sel(Hit)	7663	11545	13668	15167
ミス率 (%)	19.3	11.5	7.4	4.6

4.2 シミュレーション結果

シミュレーション結果を図 3に示す。縦軸は実行時間、横軸はラインサイズ、凡例の括弧内はデータバスの幅を表わす。

Blocking 方式ではラインフェッチ終了までに長期間 FIFO が空になり、他の方式に比べて性能が大幅に低下する。Early Restart 方式で、"ラインフェッチが並列動作している期間に別ラインのキャッシュ読み出しがブロックされる" 時間の総和は総実行時間の 0〜5%程度である。比率が小さ

[1]TITAC-2 は DLX に類似したアーキテクチャを持つため、DLX のミス率を参考にした

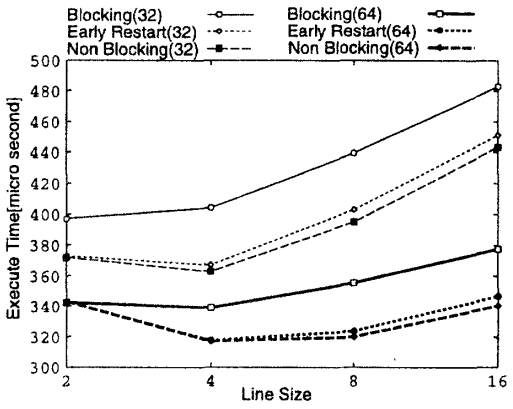

図 3: 圧縮・展開プログラムの実行時間

い理由として、TITAC-2 のメインメモリが通常 2 次キャッシュで用いられる高速 SRAM で構成され、比較的ラインフェッチ時間が短いこと、データキャッシュを搭載しないためにキャッシュメモリの競合がないことが考えられる。さらに、この時間は FIFO のバッファ効果によって短縮されるため、Non Blocking 方式との間に大きな差が見られない。

ラインサイズは命令構成を大きく変化させる (表 1)。ラインサイズ 2 ワードでは動作時間の長い Tag Miss、Tag Hit の回数が増加し、命令キャッシュの性能が低下する。ラインサイズ 16 ワードの場合には高速な Sel 動作が大半を占めるが、1 回のラインフェッチにかかる時間が増加し、その期間は次のラインフェッチ要求や ME ステージでのメモリアクセス要求がブロックされるため、プロセッサ全体の性能が落ちる。ラインサイズ 4 ワードでは、これらの性能低下が最も小さくなり、最大の性能を示している (実際の TITAC-2 ではラインサイズ 8 ワードの決定を参考文献 [2] のヒット率のみで行なったため、最適な設計になっていない)。

データバスの幅が性能に最も大きな影響を与えるのはグラフから明らかである。データバスを 64 ビットにすることでラインフェッチ時間が半分になり、命令キャッシュ自体の高速化を図れるとともに、データアクセスによって発生するメインメモリの競合を減らすことができるためである。

5 まとめ

本稿では、TITAC-2 に採用された命令キャッシュの構成と非同期式回路の特徴を利用した高速化手法について説明した。また非同期式回路の性能を引き出すためには'FIFO として動作するステージ間ラッチを空にしないことが重要であり、命令キャッシュそのものの動作時間だけでなく、長時間のメインメモリの占有が他のステージの動作を妨げることで性能低下を招くことを示した。

なお、本研究の一部は、新エネルギー・産業技術総合開発機構 (NEDO) 提案公募型・最先端分野研究開発事業受託研究 C-026、並びに科学研究費補助金 (試験研究 B)0755 8036 によって行われたものである。

参考文献

[1] 高村明裕, 桑子雅史, 南谷崇. 非同期式プロセッサ TITAC-2 の論理設計における高速化手法. 信学論 (D-I), Vol. J80-D-I, No. 3, March 1997. (掲載予定).

[2] John L. Hennesy and David A. Patterson. *Computer Architecture A Quantitive Approach Second Edition.* MORGAN KAUFMANN, 1996.

[3] 南谷崇. 非同期式プロセッサ — 超高速 VLSI システムを目指して —. 情報処理, Vol. 34, No. 1, pp. 72–80, January 1993.