

ショートノート**TLB とキャッシュメモリの統一的管理方式**鈴木 健一[†] 小林 広明[†] 中村 維男[†]

本論文は、アドレス変換だけに使用されていた TLB をキャッシュと統一的に管理する TLB-Unified Cache (TUC) を提案する。TUC では、キャッシュのタグに、TLB に格納されているページ番号へのポインタを書き込むことで、キャッシュされるデータとアドレスを間接的に関係付ける。これにより、高速メモリアレイの容量を大幅に削減することができる。また、TLB ミスの生じたエントリと関係するキャッシュエントリを高速に無効化するために、Black and White 無効化法を提案する。シミュレーションにより、TUC は、メモリアレイの大幅な削減にもかかわらず、従来の方法と同等のキャッシュミス率を示すことが明らかになった。

A TLB-Unified Cache Management SchemeKEN-ICHI SUZUKI,[†] HIROAKI KOBAYASHI[†] and TADAO NAKAMURA[†]

This paper proposes the TLB-Unified Cache (TUC), which integrates the management of a cache and a translation-lookaside buffer (TLB). In the TUC, a pointer to an entry of the TLB is stored as a cache tag instead of an address. Therefore, cached data and its address are indirectly related, and the space for cache tags is drastically reduced. This paper also proposes Black and White Invalidation for the fast invalidation of the cache entries pointing a missed TLB entry. Simulation results show that, in spite of the space saving, the TUC has the same performance in terms of cache miss ratio as conventional caches.

1. 緒言

TLB (Translation Lookaside Buffer) は、アドレス変換テーブルのエントリを保持する一種のキャッシュメモリであり、仮想アドレスから物理アドレスへの変換を高速に実行するためにメモリ管理機構により使用される¹⁾。本論文は、TLB とキャッシュを統一的に管理する TLB-Unified Cache (TUC) を提案する。TUC は、TLB とキャッシュの間で重複していたアドレス情報を単一化することにより、必要となる高速メモリアレイの容量を大幅に削減することを目的とする新しいメモリ管理機構である。最近では、TLB と仮想アドレスキャッシュを CPU と同一のチップ上に置くことが多いため、本方式により削減されたメモリアレイをレジスタ等に振り替えることが可能であり、その結果、計算機全体としての性能向上が期待できる。

2. TLB-Unified Cache の概要

図 1 は、アドレス変換と仮想アドレスキャッシュへのアクセスの一般的な機構である²⁾。通常、アドレス変換は、主記憶中の幾つかの表を逐次的に参照することにより行われる。そして、参照の結果は TLB に新しいエントリとしてキャッシュされる。多くの場合、アドレス変換は TLB を一度参照するだけで完了する。

TLB とキャッシュは、いずれも最近参照された、あるいは、これから参照される確率の高いアドレスに関する情報を格納するためのものであり、非常によく似た構造を持っている。しかしながら、一般的なアドレス管理機構では、TLB とキャッシュは独立に管理されている。

本論文で提案する TLB-Unified Cache (TUC) は、TLB とキャッシュタグアレイを統合するものである。図 2 に TUC のアドレス変換とキャッシュアクセスの機構を示す。TLB は従来のもと同じであるが、キャッシュのタグアレイは、アドレスの上位ビットでは

[†] 東北大学大学院情報科学研究科
Department of Computer and Mathematical Sciences,
Graduate School of Information Sciences,
Tohoku University

なく、TLB のエントリへのポインタとなっている。したがって、キャッシュミスの際に新しいエントリが格納される場合、そのアドレスが所属するページを格納している TLB エントリの番号 (図 2 の TLB entry number) をキャッシュタグに書き込むことになる。キャッシュのヒット/ミス判定は、キャッシュのタグが指し示す TLB エントリに格納されているページ番号を、アドレスの上位の幾つかのビットと比較することにより行う。

例として、仮想メモリ空間を 4G バイトとし、キャッシュサイズ 4k バイトのダイレクトマップ方式の仮想アドレスキャッシュを持つプロセッサを考えると、通常の方式では、キャッシュアドレスタグとして 1 エントリ当たり 20 ビットが必要である。それに対し、TUC では、高々 128 エントリ程度の TLB へのポインタで十分であるため、1 エントリ当たり 7 ビット程度で済むことになり、大幅な削減が可能となる (図

3)。

本方式のキャッシュタグは TLB エントリを指し示すポインタであるため、あるアクセスが TLB ミスを起こした場合、必ずキャッシュミスも発生する。その場合、まず主記憶上のページ交換テーブルを参照し、新しいエントリとして TLB に登録する。そして、新たに登録された TLB エントリへのタグを持つキャッシュエントリは以前の TLB エントリと関連付けられたものであるために、これらのキャッシュを無効化する。この無効化手法については、次章で詳しく述べる。以上の操作の終了後、ページ交換で決定された物理ページ番号により、主記憶からデータをフェッチし、対応する TLB エントリ番号をタグとしてキャッシュに格納する。

3. Black and White 無効化法

3.1 Black and White 無効化法の原理

2 章で述べたように、TLB がミスヒットした場合、その TLB エントリへのポインタを持っているすべてのキャッシュエントリを無効化する必要がある。しかし、無効化されるべきエントリを線形探索するならば、その処理速度が性能の低下の原因になると考えられる。これを避けるため、すべてのキャッシュエントリに、タグのビット数に応じた比較器を設けることも考えられるが、これではハードウェア量が莫大なものになってしまうため好ましくない。本論文で提案する Black and White 無効化法は、この問題を解決するものである。

Black and White 無効化法は、TLB のエントリを、頻繁にアクセスの起こっているページのもの、最近アクセスが少ないページのものに分割して管理す

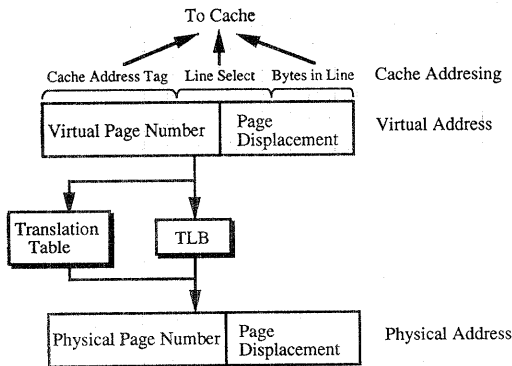


図 1 通常のアドレス変換機構と仮想アドレスキャッシュ

Fig. 1 Address translation mechanism and virtually addressed cache.

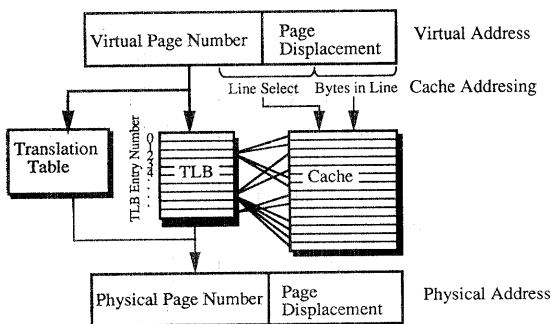
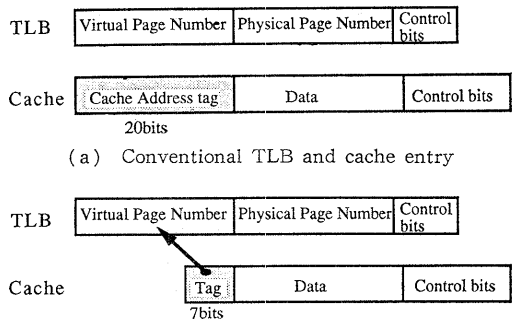


図 2 TUC のアドレス変換とキャッシュアクセス機構

Fig. 2 Address translation and cache access mechanism in the TUC.



(b) TLB and cache entry of TUC

図 3 通常のエントリおよび TUC のエントリ

Fig. 3 Cache entries of a conventional cache and the TUC.

る。そして、最近アクセスされていないページに所属するキャッシュエントリを一斉に無効化する。線形探索が不要なため、その速度は十分に速くすることができる。しかも、無効化は最近使われていないと思われるデータに対して行われるので、それによるキャッシュミスの増加はきわめて少ないと考えられる。

3.2 Black and White 無効化法の実現

Black and White 無効化法では、TLB とキャッシュの全エントリのそれぞれの制御用ビットとして1ビットずつの一貫性フラグ (Black, White の値をとる) を用意する。また、通常の制御用ビットに含まれている有効 (Valid) フラグも使用する。有効フラグは、「有効」、「無効」の値をとる。以下では、各エントリの一貫性フラグの値 (Black/White) をそのエントリの「色」と呼ぶ。

Black and White 無効化法では、キャッシュのヒット/ミス判定の際に、アドレス、タグ、および、通常用いられる有効フラグの比較だけでなく、キャッシュタグで対応付けられた TLB とキャッシュエントリのそれぞれの一貫性フラグの比較も行う。一貫性フラグの比較は表1によって解決される。対応する一貫性フラグの色が一致しなければ、仮に有効フラグが「有効」であっても、そのキャッシュエントリは無効とみなされる。したがって、TLB の一貫性フラグを反転するだけで、反転前の色を持ったキャッシュエントリをすべて同時に無効化できる。例えば、ある TLB エントリの色が Black であるとする、これを White に反転すれば、それまでそのエントリへのポインタを持っていた Black のキャッシュエントリをすべて一度に無効化することができる。

しかし、TLB の一貫性フラグを反転する場合に注意しなければならないことがある。それは、反転後の色を持ったキャッシュエントリが偶然その TLB エントリへのポインタを保持している可能性があるという

ことである。したがって、これから反転する先の色のキャッシュエントリの有効フラグを「無効」に変える必要がある。例えば、Black から White へ反転する場合には、White のキャッシュエントリの有効フラグをすべて「無効」に変えることになる。一度「無効」にしてしまえば、続いて他の TLB エントリがその方向へ反転するとしても、有効フラグを「無効」に変える必要はない。これを示すために、1ビットの方向フラグを用意する。これは、エントリごとではなく、キャッシュ全体に対して1ビットだけで十分である。方向フラグは White か Black の値をとる。方向フラグと逆の色への反転の場合、上で述べたような有効フラグの「無効」化を行う。方向フラグの色への反転の際には、この「無効」化は不要である。図4に状態遷移図を示した。この「無効」化の操作は、全キャッシュエントリのそれぞれについて、一貫性ビットと方向ビットの一致を求め、一致→「有効」、不一致→「無効」として各エントリの有効フラグに書き込むことで完了する。エントリごとに独立した操作なので、線形探索は不要であり、また、必要なハードウェア量もわずかである。

4. 性能評価

ワークステーション (Sun SPARCstation IPX) 上にC言語を用いてシミュレータを構築した。GNU のデバッガ GDB により TeX の100万命令をトレースし、そのときのキャッシュのミス率を TUC と従来のキャッシュについて測定した。図5に結果を示す。ここで、TLB ライン数 64、ページサイズ 4KB、キャッシュラインサイズ 64 バイトとし、また、TLB とキャッシュはいずれも 2-way セットアソシアティブ方式とした。

本シミュレーション結果からわかるように、本方式

表 1 一貫性フラグの比較によるキャッシュ管理
Table 1 Cache management by consistency flags.

Flags				data status
TLB consistency	cache consistency	cache valid		
any	any	invalid	→	invalid
White	White	valid	→	valid
White	Black	valid	→	invalid
Black	White	valid	→	invalid
Black	Black	valid	→	valid

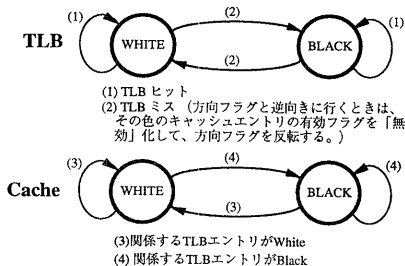


図 4 Black and White 無効化法の状態遷移図
Fig. 4 State transition diagram of Black and White Invalidation.

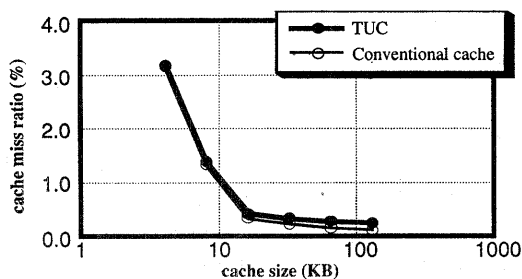


図5 シミュレーション結果
Fig. 5 Simulation results.

は、タグアレイの量を大幅に削減したにもかかわらず、従来のキャッシュとほぼ等しいミス率を保つことが可能である。

5. 結 言

本論文では、TLB-Unified Cache (TUC) による TLB とキャッシュの統一的な管理手法を提案した。TUC はアドレス情報の重複を取り除くことにより、必要なメモリ量を大幅に削減するものである。また、TUC で必要となる TLB ミス時のキャッシュエントリの高速な無効化のために、Black and White 無効化法を提案した。シミュレーションの結果から、本方式が、必要なメモリ量を大幅に削減したにもかかわらず、従来の方法とほぼ同程度のキャッシュミス率を持つことが示された。

今後、Black and White 無効化法を改良し、キャッシュミス率の低減を図る予定である。また、ハードウェアコスト量の検討を行うとともに、さまざまなベンチマークプログラムを用いたシミュレーションにより TUC の有効性を評価する予定である。

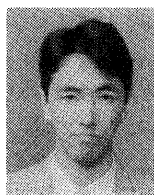
謝辞 本研究を進める上で多くのご助言をいただいた日本 IBM 東京基礎研究所の大庭信之博士に深く感謝いたします。

参 考 文 献

- 1) Smith, A. J.: Cache Memories, *Comput. Surv.*, Vol. 14, No. 3, pp. 473-530 (1982).
- 2) Teller, P. J.: Translation-Lookaside Buffer Consistency, *Computer*, Vol. 23, No. 6, pp. 26-36 (1990).

(平成 5 年 9 月 13 日受付)

(平成 6 年 2 月 17 日採録)



鈴木 健一 (正会員)

1969 年生。1992 年東北大学工学部機械工学科卒業。1994 年同大学院工学研究科機械工学専攻博士前期課程修了。現在、同情報科学研究科情報基礎科学専攻博士後期課程在学中。キャッシュメモリ、分散処理システムの研究に従事。



小林 広明 (正会員)

昭和 36 年生。昭和 58 年東北大学工学部通信卒業。昭和 63 年同大学院博士課程修了。工学博士。同年東北大学工学部機械助手。平成 3 年同大機械知能講師。平成 5 年同大情報科学研究科助教授。現在に至る。並列計算機アーキテクチャ、コンピュータグラフィックスの研究に従事。IEEE, ACM, 電子情報通信学会各会員。



中村 維男

昭和 19 年生。昭和 47 年東北大学大学院博士課程修了。工学博士。同年東北大学助手。昭和 53 年同大学助教授。昭和 63 年同大学教授。現在に至る。研究分野は、計算機アーキテクチャ。The Visual Computer の編集委員。IEEE Communication Software 委員会委員。IEEE Senior Member, 電子情報通信学会会員。