(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3587591号
(P3587591)
(45)発行日　平成16年11月10日(2004.11.10)　(24)登録日　平成16年8月20日(2004.8.20)

(51)Int.Cl.[7]　F I
G06F 12/08　G06F 12/08　509B
G06F 12/08　505Z
G06F 12/08　531Z

請求項の数 7　(全 17 頁)

(21)出願番号　特願平7-188826
(22)出願日　平成7年7月25日(1995.7.25)
(65)公開番号　特開平8-63393
(43)公開日　平成8年3月8日(1996.3.8)
審査請求日　平成13年10月30日(2001.10.30)
(31)優先権主張番号　286,121
(32)優先日　平成6年8月4日(1994.8.4)
(33)優先権主張国　米国(US)

(73)特許権者　398038580
ヒューレット・パッカード・カンパニー
HEWLETT-PACKARD COMPANY
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(74)代理人　100081721
弁理士　岡田　次生
(72)発明者　ジェームス・エス・フィンネル
アメリカ合衆国95129カリフォルニア州サンノゼ、エングルウッド・ドライブ　4920

最終頁に続く

(54)【発明の名称】キャッシュ・ミスを制御する方法およびその計算機システム

(57)【特許請求の範囲】
【請求項１】
プロセッサと、メイン・メモリと、第１レベル・キャッシュおよび第２レベル・キャッシュとを備える計算機システムにおいてキャッシュのミスを制御する方法であって、該第２レベル・キャッシュはデータ・ラインを含み、該第１レベル・キャッシュは、該第２レベル・キャッシュ内におけるデータ・ラインのフラグメントを含んでおり、
前記プロセッサによるキャッシュ参照の結果、前記第１レベル・キャッシュのミスおよび前記第２レベル・キャッシュのミスが発生したとき、
（ａ）新しいデータ・ラインを前記メイン・メモリから取り出すステップと、
（ｂ）前記ステップ（ａ）と同時に実行されるステップであって、前記第１レベル・キャッシュ内のどのエントリが無効であるかを判断するステップと、
（ｃ）前記新しいデータ・ラインが前記メイン・メモリから取り出されたとき、該新しいデータ・ラインから得られるできるだけ多くのフラグメントを前記第１レベル・キャッシュ内の無効エントリに入れるステップと、
（ｄ）前記ステップ（ｃ）と同時に実行されるステップであって、前記新しいデータ・ラインを前記第２レベル・キャッシュに入れ、該第２レベル・キャッシュ内の第２レベル犠牲キャッシュ・ラインを入れ換えるステップと、
を含む、キャッシュ・ミスを制御する方法。
【請求項２】
前記プロセッサがデータワードへのアクセスを試みることに応答して、前記第１レベル・

キャッシュにおいて該データワードを検索するステップと、
前記第１レベル・キャッシュのミスが発生したとき、前記第２レベル・キャッシュにおいて前記データワードを検索するステップと、
前記第２レベル・キャッシュのミスが発生したとき、前記ステップ（ａ）から（ｃ）を実行するステップと、を含み、
前記メイン・メモリからの前記新しいデータ・ラインは前記データワードを含んでおり、
前記第１レベル・キャッシュに入れられた前記新しいデータ・ラインからのフラグメントが前記データワードを含む、請求項１に記載のキャッシュ・ミスを制御する方法。
【請求項３】
前記ステップ（ｂ）において、前記無効エントリには、前記第１レベル・キャッシュ内の無効とマークされたエントリと、前記第２レベル・キャッシュ内の前記第２レベル犠牲キャッシュ・ラインからのフラグメントを含む第１レベル・キャッシュ内のエントリとが含まれる、請求項１または請求項２に記載のキャッシュ・ミスを制御する方法。
【請求項４】
それぞれの第２レベル・キャッシュのデータ・ラインは６４バイトを有し、第１レベル・キャッシュのデータ・ラインのフラグメントは１６バイトを有する、請求項１から請求項３のいずれかに記載のキャッシュ・ミスを制御する方法。
【請求項５】
プロセッサと、メイン・メモリと、第１レベル・キャッシュと、第２レベル・キャッシュと、キャッシュ・コントローラとを備える計算機システムであって、
前記第２レベル・キャッシュはデータ・ラインを含み、前記第１レベル・キャッシュは、前記第２レベル・キャッシュ内におけるデータ・ラインのフラグメントを含んでおり、
前記キャッシュ・コントローラは、
第１レベル・キャッシュのミスおよび第２レベル・キャッシュのミスが、プロセッサによるキャッシュ参照から生じたとき、前記メイン・メモリから新しいデータ・ラインを取り出す取り出し手段と、
前記第１レベル・キャッシュにおけるどのエントリが無効であるかを判断する判断手段と、
前記新しいデータ・ラインが前記メイン・メモリから取り出されたとき、該新しいデータ・ラインから得られるできるだけ多くのフラグメントを第１レベル・キャッシュ内の無効エントリに入れる挿入手段と、を有しており、
前記キャッシュ・コントローラは、前記挿入手段が前記新しいデータ・ラインからのできるだけ多くのフラグメントを第１レベル・キャッシュ内の無効エントリに入れるとき、前記第２レベル・キャッシュ内の第２レベル犠牲キャッシュ・ラインを入れ換える入れ替え手段を有する、計算機システム。
【請求項６】
前記判断手段によって無効と判断された無効エントリには、第１レベル・キャッシュ内の無効とマークされたエントリと、前記第２レベルキャッシュ内の前記第２レベル犠牲キャッシュ・ラインからのフラグメントを含む第１レベル・キャッシュ内のエントリとが含まれる、請求項５に記載の計算機システム。
【請求項７】
第２レベル・キャッシュのデータ・ラインのそれぞれは６４バイトを有し、第１レベルキャッシュのデータ・ラインのフラグメントのそれぞれは１６バイトを有する、請求項５または請求項６に記載の計算機システム。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】
本願発明は、多レベル・キャッシュ・システムにおけるキャッシュ・ミスを処理する方法に関する。
【０００２】

【従来の技術】
大部分の現代のコンピュータ・システムは、中央演算処理装置（ＣＰＵ）とメイン・メモリとを含む。ＣＰＵが命令およびオペランドを復号し実行できる速度は、その命令およびオペランドをメイン・メモリからＣＰＵへ送ることができる速度に依存する。多数のコンピュータ・システムは、ＣＰＵが命令およびオペランドをメイン・メモリから得るために必要な時間を短縮するために、ＣＰＵとメイン・メモリの間にキャッシュ・メモリを含む。

【０００３】
キャッシュ・メモリとは、小容量の高速バッファ・メモリであり、ＣＰＵによって近い将来に使用されると考えられるメイン・メモリの内容の部分を一時的に保持するために使用される。キャッシュの主要な目的は、データ取出しの場合でも、あるいは命令取出しの場合でも、メモリ・アクセスを実行するのに必要な時間を短縮することである。キャッシュ・メモリに位置する情報には、メイン・メモリに位置する情報よりもずっと短い時間でアクセスすることができる。したがって、キャッシュ・メモリを含むＣＰＵは、命令およびオペランドが取り出され、あるいは記憶されるのを待つために費やす必要のある時間がずっと短い。

【０００４】
キャッシュ・メモリは、１つまたは複数のデータ・ワードの多数のブロック（ラインとも呼ぶ）から成る。各ブロックには、それがどのブロックまたはメイン・メモリのコピーであるかを一意に識別するアドレス・タグが関連付けられている。プロセッサがメモリを参照するたびに、アドレス・タグの比較が行われ、要求されたデータのコピーがキャッシュ中に存在するかどうかが調べられる。所望のメモリ・ブロックがキャッシュ中にない場合、そのブロックは、メイン・メモリから検索され、キャッシュに記憶され、プロセッサに供給される。新しいキャッシュ・ラインＸをキャッシュに導入するとき、ターゲットのキャッシュ・エントリにすでに他のキャッシュ・ラインＹがある場合、存在するそのキャッシュ・ラインＹは「犠牲キャッシュ・ライン」になる。新しいキャッシュ・ラインＸのための空間を設けるには、犠牲キャッシュ・ラインＹをキャッシュから削除しなければならない。犠牲キャッシュ・ラインＹが、キャッシュに導入されてから修正されている場合、「ダーティ」と呼ばれ、メイン・メモリに書き直さなければならない。犠牲キャッシュ・ラインＹが修正されていない場合は、「クリーン」と呼ばれ、単に廃棄することができる。２レベル・キャッシュ・システムでは、「レベル１犠牲キャッシュ・ライン」と「レベル２犠牲キャッシュ・ライン」があってよい。

【０００５】
ＣＰＵは、キャッシュを使用してメイン・メモリからデータを検索するだけでなく、データを直接メモリに書き込む代わりにキャッシュに書き込むこともできる。プロセッサがデータをメモリに書き込みたいとき、キャッシュはアドレス・タグの比較を行い、データを書き込むべきデータ・ブロックがキャッシュに存在するかどうかが調べられる。データ・ブロックがキャッシュに存在する場合、データはキャッシュ中のそのデータ・ブロックに書き込まれ、そのデータ・ブロックのデータ「ダーティ・ビット」がセットされる。ダーティ・ビットは、データ・ブロック中のデータが修正されており、したがって、そのデータ・ブロックをキャッシュから削除する前に、修正済みデータをメイン・メモリに書き直しておかなければならないことを示す。データを書き込むべきデータ・ブロックがキャッシュに存在しない場合、そのデータ・ブロックをキャッシュに取り込み、あるいは、データを直接メイン・メモリに書き込まなければならない。

【０００６】
いくつかの応用例では、２つのキャッシュ・メモリが使用される。第１レベル・キャッシュ・メモリは通常、第２レベル・キャッシュ・メモリ中のデータのサブセットを有する。同様に、第２レベル・キャッシュ・メモリは通常、メイン・メモリ中のデータのサブセットを有する。一般に、第１レベル・キャッシュは、第２レベル・キャッシュと比べてサイズが小さい。第１レベル・キャッシュは、通常１プロセッサ・サイクルである高速アクセ

ス時間を有する。第２レベル・キャッシュは、たとえば、２サイクルないし３サイクルのある程度低速のアクセス時間を有する。第２レベル・キャッシュ・ライン・サイズは、メモリ・アクセス・オーバヘッド（ライン・サイズが大きければ大きいほど向上する）と、使用されないデータを取り出すことによってサイクルの無駄にすることを避けることとの間の最適な折合いである。通常、第２レベル・キャッシュ・ライン・サイズは、２ワードないし１６ワードである。第１レベル・キャッシュ・ライン・サイズは通常、１ワードから、第２レベル・キャッシュ用のキャッシュ・ラインのサイズまでの範囲である。

【０００７】

アクセスされたデータは、第１レベル・キャッシュ・メモリで探索される。第１レベル・キャッシュ・メモリでミスがあった場合、アクセスされたデータは、第２レベル・キャッシュ・メモリで探索される。第２レベル・キャッシュ・メモリでミスがあった場合、データはメイン・メモリから取り出される。

【０００８】

プロセッサ・システム用のある２レベル・キャッシュ構成では、第１レベル・キャッシュは、第２レベル・キャッシュの完全なサブセットである。第１レベル・キャッシュは、第２レベル・キャッシュよりも小規模であり、第２レベル・キャッシュよりもプロセッサに近い。第１レベル・キャッシュは、第２レベル・キャッシュよりも小規模であり、かつ第２レベル・キャッシュよりもプロセッサに近いため、一般に、第１レベル・キャッシュに伴う第２レベル・キャッシュやメモリよりもアクセス待ち時間が短いので改良された性能を提供する。

【０００９】

通常の動作では、あるメモリ参照を満たすのに必要ないくつかの異なる動作がある。メモリ参照とは一般に、ロード命令または記憶命令である。まず、最も簡単で最も高速の参照は、第１レベル・キャッシュでヒットすることができる。ヒットは、所望のメモリ・アドレスのデータが、検査中のキャッシュ（第１レベル・キャッシュまたは第２レベル・キャッシュ）に存在するときとして定義されている。ミスは、所望のメモリ・アドレスのデータが、検査中のキャッシュ（第１レベル・キャッシュまたは第２レベル・キャッシュ）に存在しないメモリ参照として定義されている。第１レベル・キャッシュでヒットがあったとき、ゼロ・サイクル・ペナルティがあり、処理ペナルティなしで参照が完了する。

【００１０】

次に高速のキャッシュは、第２レベル・キャッシュでヒットする第１レベル・キャッシュ・ミスである。この場合、単一の第１レベル・キャッシュ・ラインを第２レベル・キャッシュから得たデータで埋めるための動作シーケンスが実行される。続いて、データがプロセッサに供給され、あるいは、記憶が完了する。これは、中間速度動作であり、プロセッサは、メモリ参照が成し遂げられる間フリーズする。

【００１１】

最も低速の動作は、メモリ参照が第１レベル・キャッシュでも第２レベル・キャッシュでもミスしたときのものである。この例では、関連するラインをメイン・メモリから第２レベル・キャッシュに導入するための長い動作シーケンスが開始される。このデータがメモリから返され第２レベル・キャッシュにコピーされると、第１レベル・キャッシュが再び参照され、その結果、今度は、第１レベル・キャッシュ・ミスが発生し、次に、第２レベル・キャッシュでヒットが発生する。これによって、第２レベル・キャッシュ・ラインの関連する部分が第１レベル・キャッシュに書き込まれ、続いて、要求されたデータがプロセッサに供給され、あるいは、記憶が完了する。

【００１２】

１つの拡張は、第２レベル・キャッシュ・ラインが書き込まれるのと同時に、アドレスされた第１レベル・キャッシュ・ラインを埋めることである。これによって、第１レベル・キャッシュ・ミスのペナルティがなくなり、第２レベル・キャッシュ・ミスの性能ペナルティだけになる。マルチレベル・キャッシュ・システムに関する一般的な情報については、たとえば、Ｊ．　Ｌ．　Ｂａｅｒ、Ｗ．　Ｈ．　Ｗａｎｇ著「Ｍｕｌｔｉｌｅｖｅｌ

Ｃａｃｈｅ Ｈｉｅｒａｒｃｈｉｅｓ； Ｏｒｇａｎｉｚａｔｉｏｎｓ， Ｐｒｏｔｏｃｏｌｓ ａｎｄ Ｐｅｒｆｏｒｍａｎｃｅｓ」（Ｊｏｕｒｎａｌ ｏｆ Ｐａｒａｌｌｅｌ Ｄｉｓｔｒｉｂｕｔｅｄ Ｃｏｍｐｕｔｉｎｇ、第６巻、１９８９年、４５１～４７６ページ）、Ｗ． Ｈ． Ｗａｎｇ、Ｊ． Ｂａｅｒ、およびＨ． Ｌｅｖｙ著「Ｏｒｇａｎｉｚａｔｉｏｎ ａｎｄ Ｐｅｒｆｏｒｍａｎｃｅ ｏｆ ａ Ｔｗｏ－Ｌｅｖｅｌ Ｖｉｒｔｕａｌ－Ｒｅａｌ Ｃａｃｈｅ Ｈｉｅｒａｒｃｈｙ」（Ｐｒｏｃｅｅｄｉｎｇｓ ｏｆ ｔｈｅ １６ｔｈ Ａｎｎｕａｌ Ｉｎｔｅｒｎａｔｉｏｎａｌ Ｓｙｍｐｏｓｉｕｍ ｏｎ Ｃｏｍｐｕｔｅｒ Ａｒｃｈｉｔｅｃｔｕｒｅ、１９８９年、１４０～１４８ページ）、Ｊ． Ｌ． Ｂａｅｒ、Ｗ． Ｈ． Ｗａｎｇ著「Ｏｎ ｔｈｅ Ｉｎｃｌｕｓｉｏｎ Ｐｒｏｐｅｒｔｉｅｓ ｆｏｒ Ｍｕｌｔｉ－Ｌｅｖｅｌ Ｃａｃｈｅ Ｈｉｅｒａｒｃｈｉｅｓ」（Ｐｒｏｃｅｅｄｉｎｇｓ ｏｆ ｔｈｅ １５ｔｈ Ａｎｎｕａｌ Ｉｎｔｅｒｎａｔｉｏｎａｌ Ｓｙｍｐｏｓｉｕｍ ｏｎ Ｃｏｍｐｕｔｅｒ Ａｒｃｈｉｔｅｃｔｕｒｅ、１９８８年、７３～８０ページ）を参照されたい。

【００１３】

【発明が解決しようとする課題】

本願発明の目的は、単一のキャッシュ・ミスの処理中に将来のキャッシュ・ミスを処理するようにして、多レベル・キャッシュ・システムを含むコンピュータ・プロセッサでのメモリ参照の結果でおきるキャッシュ・ミス・ペナルティを統計的に減少させることである。

【００１４】

【課題を解決するための手段】

本願発明の好ましい実施例によれば、計算機システムは、プロセッサと、メイン・メモリと、第１レベル・キャッシュと、第２レベル・キャッシュとを含む。第２レベル・キャッシュはデータ・ラインを含む。第１レベル・キャッシュは、第２レベル・キャッシュ内のデータ・ラインのデータ・ライン・フラグメントを含む。好ましい実施例では、第２レベル・キャッシュに記憶されるデータ・ラインのサイズは６４バイト（１６ワード）であり、第１レベル・キャッシュに記憶されるデータ・ライン・フラグメントのサイズは１６バイト（４ワード）である。プロセッサがデータ・ワードにアクセスしようとしたことに応答して、キャッシュ制御装置が第１レベル・キャッシュでそのデータ・ワードを探索する。アクセスが試みられた結果、第１レベル・キャッシュ・ミスが発生すると、第２レベル・キャッシュでデータ・ワードが探索される。第２レベル・キャッシュ・ミスが発生すると、参照中のデータ・ワードを含む新しいデータ・ラインがメイン・メモリから取り出される。同時に、キャッシュ制御装置は、新しいメモリ・データ・ラインによってマップされる第１レベル・キャッシュ・エントリのうちのどれが無効であるかを識別する。

【００１５】

好ましい実施例では、無効なエントリには、たとえば、パージ、フラッシュ、またはマルチプロセッサのスヌープによって以前に無効のマークが付けられた第１レベル・キャッシュ・エントリと、レベル２犠牲キャッシュ・ラインのキャッシュ・ライン・フラグメントを含んでいた第１レベル・キャッシュ内のエントリとが含まれる。レベル２キャッシュ・ラインとは、メイン・メモリから取り出された新しいデータ・ラインで入れ換えられる第２レベル・キャッシュ内のキャッシュ・ラインである。新しいデータ・ラインは、メイン・メモリから取り出された後、第２レベル・キャッシュに入れられ、レベル２犠牲キャッシュ・ラインと入れ換える。また、新しいデータ・ラインのできるだけ多くのデータ・ライン・フラグメントが、第１レベル・キャッシュ中の無効なエントリに入れられる。第１レベル・キャッシュに入れられた新しいデータ・ラインのデータ・ライン・フラグメントのうちの１つは、現在参照中のデータ・ワードを含む。

【００１６】

【実施例】

図１は、本願発明の好ましい実施例による多レベル・キャッシュを含むプロセッサ・システムの概略ブロック図を示す。処理システムに対する命令を実行する中央演算処理装置（

ＣＰＵ）２１が示されている。メイン・メモリ２５は、処理システム用の記憶域を提供する。好ましい実施例では、各データ・ワードは４バイトを含む。２レベル・キャッシュ・システムは、第１レベル・キャッシュ２４と第２レベル・キャッシュ２３とを含む。

【００１７】

好ましい実施例では、第１レベル・キャッシュ２４は、キャッシュ・ライン・サイズが小さな４ワードである、小規模で高速の直接マップ・キャッシュである。第１レベル・キャッシュ２４は、第２レベル・キャッシュ２３のサブセットを含む。第１レベル・キャッシュ２４内の有効な各エントリは、第２レベル・キャッシュ２３内の有効なキャッシュ・ラインの４ワード・フラグメントを含む。好ましい実施例では、第１レベル・キャッシュ２４から第２レベル・キャッシュへの書込みは行われない。すなわち、記憶命令に応答して、データを第１レベル・キャッシュ２４に記憶することはできるが、第２レベル・キャッシュ２３に書き込むことはできない。したがって、第１レベル・キャッシュ２４は、第２レベル・キャッシュ２３よりも新しいデータを有することができる。第２レベル・キャッシュは、キャッシュ・ライン・サイズがより大きな１６ワードである、より大規模で低速の直接マップ・キャッシュである。

【００１８】

キャッシュ制御論理機構２２は、ＣＰＵ２１のために第１レベル・キャッシュ２４および第２レベル・キャッシュ２３にアクセスし、メモリ２５と第１レベル・キャッシュ２４と第２レベル・キャッシュ２３の間のデータ転送の調和をとる。キャッシュ制御論理機構２２とメモリ２５の間のデータ・アクセスは、メモリ・データ線４０を使用して実行される。ＣＰＵ２１は、アドレス線３１および第１レベル・アドレス線３７を使用してキャッシュ制御論理機構２２および第１レベル・キャッシュ２４にデータ・アドレスを送る。ＣＰＵ２１は、記憶データ線３３を使用して、記憶すべきデータをキャッシュ制御論理機構２２に送る。ＣＰＵ２１は、第１レベル・ロード・データ線３２を介してロード・データを受け取る。

【００１９】

ＣＰＵ２１は、アドレス線３１および第２レベル・アドレス線３４を使用してキャッシュ制御論理機構２２および第２レベル・キャッシュ２３にデータ・アドレスを送る。ＣＰＵ２１は、第２レベル・キャッシュ２３から第２レベル・データ出力線３５を介してロード・データを受け取る。第２レベル・キャッシュ２３は、第２レベル・データ入力線３６を介してデータを受け取る。第１レベル・キャッシュ２４は、第１レベル・データ入力線３９を介してデータを受け取る。第１レベル・キャッシュ２４は、第１レベル・データ出力線３８を介してキャッシュ制御論理機構２２にデータを送る。

【００２０】

図２、図３、図４、図５、および図６は、本願発明の好ましい実施例によるメモリ参照の結果として発生する、図１に示したプロセッサでのデータ・フローを示す。

【００２１】

図２中の点線は、ロードのためのメモリ参照の結果、第１レベル・キャッシュ２４でヒットが発生するときのデータ・フローを示す。図のように、ＣＰＵ２１は、アドレス線３１上でキャッシュ制御論理機構２２へアドレスを送り、キャッシュ制御機構２２はそのアドレスを第１レベル・アドレス線３７へ送り、さらに、第１レベル・キャッシュ２４へ送る。第１レベル・キャッシュ２４から読み取られたデータは、第１レベル・データ出力線３８上でキャッシュ制御論理機構２２へ送られ、第１レベル・ロード・データ線３２上でＣＰＵ２１へ送られ、ロード要求が完了する。

【００２２】

図３中の点線は、記憶のためのメモリ参照の結果、第１レベル・キャッシュ２４でヒットが発生するときのデータ・フローを示す。図のように、ＣＰＵ２１は、アドレス線３１上でキャッシュ制御論理機構２２へアドレスを送る。また、ＣＰＵ２１は、記憶データ線３３上でキャッシュ制御論理機構２２へ記憶データを送る。キャッシュ制御機構２２は、第１レベル・アドレス線３７上にアドレスを置き、さらに、そのアドレスを第１レベル・キ

ャッシュ２４上に置く。第１レベル・キャッシュ２４は、第１レベル・キャッシュ２４のそのアドレスにある現内容を第１レベル・データ出力線３８上でキャッシュ制御論理機構２２に返す。どんなタイプの記憶命令が関与するかに応じて、キャッシュ制御論理機構２２は、第１レベル・キャッシュ２４から返されたデータをＣＰＵ２１からの記憶データと選択的にマージする。マージされたデータは、第１レベル・データ入力線３９を使用して第１レベル・キャッシュ２４に書き込まれ、記憶要求が完了する。

【００２３】

図４中の点線は、記憶またはロードのためのメモリ参照の結果、第１レベル・キャッシュ２４でクリーン・ミスが発生し、第２レベル・キャッシュ２３でヒットが発生するときのデータ・フローを示す。クリーン・ミスが発生するのは、第１レベル・キャッシュ２４でアクセスされたデータが、要求されたキャッシュ・ラインの一部ではなく、かつ修正されていないときであり、したがって、この修正済みデータを第２レベル・キャッシュ２３に書き込むことも、あるいはメイン・メモリ２５に書き込むこともなく、このデータを第１レベル・キャッシュ２４から削除して、要求されたキャッシュ・ラインのための空間を設けることができる。アドレス・タグの比較によって、必要なデータ・ブロックが第１レベル・キャッシュ２４にないことが示されると、第２レベル・キャッシュ２３が検査される。第１レベル・キャッシュ２４中の対応するデータ・ブロックがダーティではなく、第２レベル・キャッシュ２３でヒットがあった場合、第１レベル・キャッシュ２４中の対応するデータ・ブロックは単に、無効化され第２レベル・キャッシュ２３から得たデータで上書きされる。ＣＰＵ２１は、第２レベル・キャッシュ２３にアクセスするために、キャッシュ制御論理機構２２へのアドレス線３１をドライブし、キャッシュ制御論理機構２２は第２レベル・アドレス線３４をドライブする。これに応答して、第２レベル・キャッシュ２３は、第２レベル・データ出力線３５上でデータを返す。また、キャッシュ制御論理機構２２は、第１レベル・アドレス線３７上で第１レベル・キャッシュ２４のアドレスをドライブする。第２レベル・キャッシュ２３からのデータは次いで、第２レベル・データ出力線３５および第１レベル・データ入力線３９を使用して、キャッシュ制御論理機構２２を介して第１レベル・キャッシュ２４に書き込まれる。記憶命令の場合、キャッシュ制御論理機構２２は、ＣＰＵ２１からのデータが第１レベル・キャッシュ２４に書き込まれるとき、ＣＰＵ２１からの記憶データを前記データとマージする。ロード命令の場合、ＣＰＵ２１は、第２レベル・データ出力線３５上で第２レベル・キャッシュ２３から直接ロード・データを得る。

【００２４】

図５および図６中の点線は、記憶またはロードのためのメモリ参照の結果、第１レベル・キャッシュ２４でダーティ・ミスが発生し、第２レベル・キャッシュ２３でヒットが発生するときのデータ・フローを示す。ダーティ・ミスが発生するのは、第１レベル・キャッシュ２４中のアクセスされたデータが、要求されたキャッシュ・ラインの一部ではなく、かつ修正されているときであり、したがって、この修正済みデータを第２レベル・キャッシュ２３またはメイン・メモリ２５にも書き込まない限り、第１レベル・キャッシュ２４からデータを削除することはできない。アドレス・タグの比較によって、必要なデータ・ブロックが第１レベル・キャッシュ２４にないことが示されると、第２レベル・キャッシュ２３が検査される。第１レベル・キャッシュ２４中の対応するデータ・ブロックがダーティであり、第２レベル・キャッシュ２３でヒットがあった場合、第１レベル・キャッシュ２４中の対応するデータ・ブロックは第２レベル・キャッシュ２３に書き込まれる。これを図５で示す。図５で、ＣＰＵ２１は、アドレス線３１上でキャッシュ制御論理機構２２へアドレスを送る。キャッシュ制御論理機構２２は、第１レベル・アドレス線３７上で第１レベル・キャッシュ２４のアドレスをドライブする。またＣＰＵ２１は、第２レベル・アドレス線３４をドライブするようにしむける。第１レベル・キャッシュ２４からのデータは次いで、第１レベル・データ出力線３８および第２レベル・データ入力線３６を使用して、キャッシュ制御論理機構２２を介して第２レベル・キャッシュ２３に書き込まれる。

【００２５】
ダーティ・データの書直しが完了した後、第１レベル・キャッシュ２４でのクリーン・ミスの場合と同様に参照が継続する。図６に示したように、ＣＰＵ２１は、第２レベル・キャッシュ２３にアクセスするために、キャッシュ制御論理機構２２へのアドレス線３１をドライブし、キャッシュ制御論理機構２２は、第２レベル・アドレス線３４をドライブする。これに応答して、第２レベル・キャッシュは、第２レベル・データ出力線３５上でデータを返す。キャッシュ制御論理機構２２は、第１レベル・アドレス線３７上で第１レベル・キャッシュ２４のアドレスをドライブする。第２レベル・キャッシュ２３からのデータは次いで、第２レベル・データ出力線３５および第１レベル・データ入力線３９を使用して、キャッシュ制御論理機構２２を介して第１レベル・キャッシュ２４に書き込まれる。記憶命令の場合、キャッシュ制御論理機構２２は、ＣＰＵ２１からのデータが第１レベル・キャッシュ２４に書き込まれるときに、ＣＰＵ２１からの記憶データを前記データとマージする。ロード命令の場合、ＣＰＵ２１は、第２レベル・データ出力線３５上で第２レベル・キャッシュ２３から直接ロード・データを得る。

【００２６】
第１レベル・キャッシュ２４と第２レベル・キャッシュ２３の両方でミスがあったとき、データはメイン・メモリ２５から導入しなければならない。第１レベル・キャッシュ２４でダーティ・ミスがあった場合、上述のように、ダーティ・データは第２レベル・キャッシュ２３に書き直される。第２レベル・キャッシュ２３でダーティ・ミスがあり、すなわち、第２レベル犠牲キャッシュ・ラインがダーティであるとき、第２レベル・コピー出力動作を介してキャッシュ・ライン全体をメイン・メモリ２５に書き直さなければならない。第２レベル犠牲キャッシュ・ラインとは、メイン・メモリ２５から取り出された新しいデータ・ラインで置換される第２レベル・キャッシュ２３内のキャッシュ・ラインである。第２レベル・コピー出力が実行される間、第２レベル犠牲キャッシュ・ラインの各部を保持していた第１レベル・キャッシュ・エントリまたは第２レベル・キャッシュ・エントリが無効化される。第２レベル・コピー出力動作が完了した後、メイン・メモリ２５からデータが読み取られ、第２レベル・キャッシュ２３に書き込まれる。

【００２７】
以下でさらに説明するように、本願発明の好ましい実施例では、メイン・メモリ２５からデータ・ラインが取り出されるとき、第１レベル・キャッシュ２４中の要求された１６バイト・ラインが埋められるだけでなく、第１レベル・キャッシュ２４中の最大３つの他の第１レベル・キャッシュ・ラインも埋められる。これらの他の３つのキャッシュ・ラインは、現在参照されていないが、キャッシュ参照の統計的局所性のために、何らかの他のキャッシュ・ミスによって入れ換えられる前に将来の命令によって参照される確率が高い。

【００２８】
図７、図８、図９、図１０は、本願発明の好ましい実施例によって第２レベル・キャッシュでのミスの後に第１レベル・キャッシュの埋込みを処理する様々な方法を示す。

【００２９】
第２レベル・キャッシュ２３および第１レベル・キャッシュ２４中のキャッシュ・ラインを図７に示す。第２レベル・キャッシュ２３は、１６個のキャッシュ・ラインを含む。図７は、１６個のキャッシュ・ラインのうちのキャッシュ・ライン６１、６２、６４、６５、６６、６７を示す。各キャッシュ・ラインは、１６個のデータ・ワードを含む。図７で、１６個のワード・キャッシュ・ラインは、４つの４ワード・ライン・フラグメントとして構成されたものとして示されている。各キャッシュ・ラインごとに、第１列５１は、キャッシュ・ライン・ワード０ないし３を含む。第２列５２は、キャッシュ・ライン・ワード４ないし７を含む。第３列５３は、キャッシュ・ライン・ワード８ないし１１を含む。第４列５４は、キャッシュ・ライン・ワード１２ないし１５を含む。第１レベル・キャッシュ２４は、８つの４ワード・ライン・フラグメントを含む。各４ワード・ライン・フラグメントは、特定の第２レベル・キャッシュ・エントリにマップされる。図７は、ライン・フラグメント７１、７２、７３、７４、７５、７６、７７、７８を示す。好ましい実施

例では、第２レベル・キャッシュ２３および第１レベル・キャッシュ２４は直接マップされる。各第２レベル・キャッシュ・ラインは、４つの４ワード・フラグメントとして示されている。第２レベル・キャッシュ・ラインの４つの４ワード・フラグメントはそれぞれ、特定の第１レベル・キャッシュ・エントリにマップされる。

【００３０】

図７で、１６ワード・キャッシュ・ラインＡは、第２レベル・キャッシュ２３の第２レベル・キャッシュ・エントリ６１に存在するものとして示されている。１６ワード・キャッシュ・ラインＢは、第２レベル・キャッシュ２３の第２レベル・キャッシュ・エントリ６２に存在するものとして示されている。１６ワード・キャッシュ・ラインＩは、第２レベル・キャッシュ２３の第２レベル・キャッシュ・エントリ６４に存在するものとして示されている。１６ワード・キャッシュ・ラインＪは、第２レベル・キャッシュ２３の第２レベル・キャッシュ・エントリ６５に存在するものとして示されている。１６ワード・キャッシュ・ラインＫは、第２レベル・キャッシュ２３の第２レベル・キャッシュ・エントリ６６に存在するものとして示されている。１６ワード・キャッシュ・ラインＰは、第２レベル・キャッシュ２３の第２レベル・キャッシュ・エントリ６７に存在するものとして示されている。

【００３１】

同様に、１６ワード・キャッシュ・ラインＡの最初の４つのワードは、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７１に存在する。１６ワード・キャッシュ・ラインＫのワード８ないし１１は、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７３に存在する。１６ワード・キャッシュ・ラインＡの最後の４つのワードは、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７４に存在する。１６ワード・キャッシュ・ラインＢの最初の４つのワードは、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７５に存在する。１６ワード・キャッシュ・ラインＪのワード４ないし７は、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７６に存在する。１６ワード・キャッシュ・ラインＪのワード８ないし１１は、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７７に存在する。１６ワード・キャッシュ・ラインＰの最後の４つのワードは、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７８に存在する。

【００３２】

第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７２中の見出しは、無効とマーク付けされている。キャッシュ・ラインは、パージ命令やフラッシュ命令やマルチプロセッサ・スヌープなどいくつかの手段を介して無効化することができる。

【００３３】

第１レベル・キャッシュ２４と第２レベル・キャッシュ２３の両方に関してミスが発生すると、第２レベル・キャッシュ２３に新しいラインが導入され、第１レベル・キャッシュ２４に適当な４ワード・ライン・フラグメントが書き込まれる。

【００３４】

たとえば、メイン・メモリ２５に存在するが、第１レベル・キャッシュ２４にも第２レベル・キャッシュ２３にも存在しない、ラインＱの第２ワードにＣＰＵ２１がアクセスする必要があると仮定する。さらに、ラインＱが、ラインＡと同じ第１レベル・キャッシュ・エントリおよび第２レベル・キャッシュ・エントリにマップされると仮定する。

【００３５】

ラインＱを第２レベル・キャッシュ２３に書き込み、適当な４ワード・フラグメントを第１レベル・キャッシュ２４に書き込む、ある方法では、ラインＡはまず、それがダーティであるときにメイン・メモリ２５に書き直される。次に、ラインＡの各部を保持するすべての第１レベル・キャッシュ・エントリおよび第２レベル・キャッシュ・エントリが無効化される。この場合、第２レベル・キャッシュ・エントリ６１、第１レベル・キャッシュ・エントリ７１、および第１レベル・キャッシュ・エントリ７４が無効化される。次いで、ラインＱが、メイン・メモリ２５から導入され、第２レベル・キャッシュ２３中の第２

レベル・キャッシュ・エントリ６１に書き込まれる。最後に、ラインＱの第１の４ワード・フラグメントが、第２レベル・キャッシュ２３から読み出され、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７１に書き込まれる。この結果を図８に示す。

【００３６】

上述の方法は、ラインＱの適当な４ワード・フラグメントがメイン・メモリ２５から第２レベル・キャッシュ２３に導入されるとき、第１レベル・キャッシュ２４に直接、前記フラグメントを入れることによって、改善することができる。これによって、ラインＱの第１の４ワード・フラグメントを、第２レベル・キャッシュ２３から読み出して第１レベル・キャッシュ２４に書き込むことができるようにするために、ラインＱが第２レベル・キャッシュ２３に書き込まれるのを待つことが不要になる。多数のシステムでは、バイパス経路のために、追加データ経路論理機構が必要になる可能性がある。しかし、本発明の好ましい実施例では、キャッシュ制御論理機構２２が、第１レベル・キャッシュ２４と第２レベル・キャッシュ２３の両方及び又はメイン・メモリ２５のインタフェースとして働き、追加データ経路論理機構を不要にする。

【００３７】

上述の方法に対する追加改良では、キャッシュ制御装置２２は、ラインＡが上書きされることによって無効化された第１レベル・キャッシュ・エントリを記憶することができる。メイン・メモリ２５からラインＱが導入されると、４ワード・フラグメントが、第１レベル・キャッシュ２４の第１レベル・キャッシュ・エントリ７１および第１レベル・キャッシュ・エントリ７４に書き込まれる。この結果を図９に示す。この改良の利点は、ラインＱの第４の４ワード・フラグメントへの以後のアクセスが第１レベル・キャッシュ２４でヒットすることである。ラインＱの第２または第３の４ワード・フラグメントへの以後のアクセスでは依然として、第２レベル・キャッシュから第１レベル・キャッシュへの移動が必要である。

【００３８】

本願発明の好ましい実施例では、上述の方法が再び改良される。この場合、どの第１レベル・キャッシュ・フラグメントがラインＡフラグメントによって無効化されたか記憶されるだけでなく、すでに他の理由で無効化されている可能性がある関連する第１レベル・キャッシュ・エントリに関する検査も行われる。メイン・メモリ２５からラインＱが導入されると、ラインＱのできるだけ多くの４ワード・フラグメントが第１レベル・キャッシュ２４に書き込まれる。この結果は、図１０で分かる。この「ブートレグ第１レベル・キャッシュ埋込み（ｂｏｏｔｌｅｇ　ｆｉｒｓｔ　ｌｅｖｅｌ　ｃａｃｈｅ　ｆｉｌｌ）」は、図１に示した本願発明のシステム構成によって実現可能になる。メイン・メモリ２５から第２レベル・キャッシュ２３に書き込まれるデータがキャッシュ制御論理機構２２を通過するため、データ経路ハードウェアを追加せずにこの好ましい方法を実施することができる。

【００３９】

図１１は、本願発明の好ましい実施例による方法の動作を要約したものである。この方法は、埋め込まれている位置が、要求されたものではない場合でも、必要に応じて第１レベル・キャッシュ・エントリを埋め込むことを含む。方法ステップ８１で、通常の第１レベル・キャッシュ・ミスが発生していて、第２レベル・キャッシュ２３が検査され、そして第２レベル・キャッシュ２３でもミスが発生し、メイン・メモリ２５の参照が開始される。方法ステップ８２で、メモリからの新しいデータ・ラインが要求される。

【００４０】

方法ステップ８３で、新しいメモリ・データを待つ間に、様々な動作が実行され、必要な第１レベル・キャッシュ・エントリが無効化されて第１レベル・キャッシュで空間が設けられる。第１レベル犠牲キャッシュ・ラインがダーティである場合、データは第２レベル・キャッシュに書き直される。必要な第２レベル・キャッシュ・エントリと、第２レベル犠牲キャッシュ・ラインのフラグメントを含む第１レベル・キャッシュ・エントリも無効

化され、第２レベル・キャッシュで空間が設けられる。第２レベル犠牲キャッシュ・ラインの一部がダーティである場合、データはメモリに書き直される。また、新しいメモリ・データによってマップされる無効な第１レベル・キャッシュ・エントリはすべて識別される。

【００４１】

方法ステップ８４で、メモリからの新しいデータ・ラインが第２レベル・キャッシュに書き込まれる間、データ・ラインのフラグメントは、上記ステップで判定された利用可能な第１レベル・キャッシュエントリのそれぞれに書き込まれる。この方法を使用すると、長い時間にわたって無効なままである第１レベル・キャッシュ・エントリはほとんどなくなる。最大３つの余分の第１レベル・キャッシュ・エントリを余分のペナルティなしで埋め込むことができる。

【００４２】

以上で説明を終了するが、本願発明には、例として次のような実施態様が含まれる。

【００４３】

（イ）プロセッサ（２１）と、メイン・メモリ（２５）と、第２レベルキャッシュ内のデータ・ラインのデータ・ライン・フラグメントを含む第１レベル・キャッシュ（２４）と、データ・ラインを含む前記第２レベル・キャッシュ（２３）とを含む計算機システムにおいて、プロセッサ（２１）によるキャッシュ参照の結果、第１レベル・キャッシュ（２４）のミスおよび第２レベル・キャッシュ（２３）のミスが発生したときに実行される方法であって、（ａ）新しいデータ・ラインをメイン・メモリ（２５）から取り出すステップと、（ｂ）ステップ（ａ）と同時に、第１レベル・キャッシュ（２４）中のどのエントリが無効であるかを識別するステップと、（ｃ）新しいデータ・ラインがメイン・メモリ（２５）から取り出されたとき、新しいデータ・ラインから得たできるだけ多くのデータ・ライン・フラグメントを第１レベル・キャッシュ（２４）中の無効なエントリに入れるステップとを含むことを特徴とする新しいワード・フラグメントの書き込み方法。

【００４４】

（ロ）上記ステップ（ｃ）と同時に、新しいデータ・ラインが第２レベル・キャッシュ（２３）に入れられ、第２レベル・キャッシュ（２３）内の第２レベル犠牲キャッシュ・ラインと入れ換えるステップを更に含むことを特徴とする上記（イ）に記載の方法。

【００４５】

（ハ）上記ステップ（ｂ）で、無効とマーク付けされた第１レベル・キャッシュ（２４）のエントリと、第２レベル・キャッシュ（２３）内の第２レベル犠牲キャッシュ・ラインから得たキャッシュ・ライン・フラグメントを含む第１レベル・キャッシュ（２４）内のエントリとが、無効なエントリに含まれることを特徴とする上記（イ）に記載の方法。

【００４６】

（ニ）各第２レベル・キャッシュ（２３）ラインが、１６個のワードを含み、各第１レベル・キャッシュ（２４）ライン・フラグメントが、４つのワードを含むことを特徴とする上記（イ）に記載の方法。

【００４７】

（ホ）計算機システムにおいて、プロセッサ（２１）と、メイン・メモリ（２５）と、第２レベル・キャッシュ（２３）内のデータ・ラインのデータ・ライン・フラグメントを含む第１レベル・キャッシュ（２４）と、データ・ラインを含む前記第２レベル・キャッシュ（２３）と、プロセッサ（２１）、メイン・メモリ（２５）、第１レベル・キャッシュ（２４）、および第２レベル・キャッシュ（２３）に結合されたキャッシュ制御装置（２２）とを備え、キャッシュ制御装置（２２）が、第１レベル・キャッシュ（２４）のミスおよび第２レベル・キャッシュ（２３）のミスが、プロセッサ（２１）によるキャッシュ参照の結果に伴って発生したとき、新しいデータ・ラインをメイン・メモリ（２５）から取り出す手段と、第１レベル・キャッシュ（２４）中のどのエントリが無効であるかを識別する手段と、新しいデータ・ラインがメイン・メモリ（２５）から得られたとき、新しいデータ・ラインから得たできるだけ多くのデータ・ライン・フラグメントを第１レベル

・キャッシュ（２４）中の無効なエントリに入れる手段とを備えることを特徴とする計算機システム。

【００４８】

（ヘ）上記キャッシュ制御装置（２２）がさらに、新しいデータ・ラインから得たできるだけ多くのデータ・ライン・フラグメントを第１レベル・キャッシュ（２４）中の無効なエントリに入れるのと同時に、第２レベル・キャッシュ（２３）内の第２レベル犠牲キャッシュ・ラインと入れ換える手段を備えることを特徴とする上記（ホ）に記載の計算機システム。

【００４９】

（ト）上記識別手段によって識別されるエントリには、無効とマーク付けされた第１レベル・キャッシュ（２４）のエントリと、第２レベル・キャッシュ（２３）内の第２レベル犠牲キャッシュ・ラインから得たキャッシュ・ライン・フラグメントを含む第１レベル・キャッシュ（２４）内のエントリとが含まれることを特徴とする上記（ホ）に記載の計算機システム。

【００５０】

（チ）各第２レベル・キャッシュ（２３）ラインが、１６個のワードを含み、各第１レベル・キャッシュ（２４）ライン・フラグメントが、４つのワードを含むことを特徴とする上記（ホ）に記載の計算機システム。

【００５１】

（リ）プロセッサ（２１）と、メイン・メモリ（２５）と、第２レベルキャッシュ内のデータ・ラインのデータ・ライン・フラグメントを含む第１レベル・キャッシュ（２４）と、データ・ラインを含む前記第２レベル・キャッシュ（２３）とを含む計算機システムにおける新しいワード・フラグメントの書き込み方法において、（ａ）プロセッサ（２１）がデータ・ワードにアクセスしようとしたことに応答して、第１レベル・キャッシュ（２４）でそのデータ・ワードを探索するステップと、（ｂ）ステップ（ａ）が実行された結果、第１レベル・キャッシュ（２４）のミスが発生したとき、第２レベル・キャッシュ（２３）でデータ・ワードを探索するステップと、（ｃ）ステップ（ｂ）が実行された結果、第２レベル・キャッシュ（２３）のミスが発生したとき、（ｃ．１）データ・ワードを含む新しいデータ・ラインをメイン・メモリ（２５）から取り出すサブステップと、（ｃ．２）サブステップ（ｃ．１）と同時に、第１レベル・キャッシュ（２４）中のどのエントリが無効であるかを識別するサブステップと、（ｃ．３）データ・ワードを含む新しいデータ・ラインがメイン・メモリ（２５）から取り出されたとき、新しいデータ・ラインから得たできるだけ多くのデータ・ライン・フラグメントを第１レベル・キャッシュ（２４）中の無効なエントリに入れるサブステップとを含むことを特徴とする新しいワード・フラグメントの書き込み方法。

【００５２】

（ヌ）上記サブステップ（ｃ．３）と同時に、新しいデータ・ラインが第２レベル・キャッシュ（２３）に入れられ、第２レベル・キャッシュ（２３）内の第２レベル犠牲キャッシュ・ラインと入れ換えるサブステップを更に含むことを特徴とする上記（リ）に記載の方法。

【００５３】

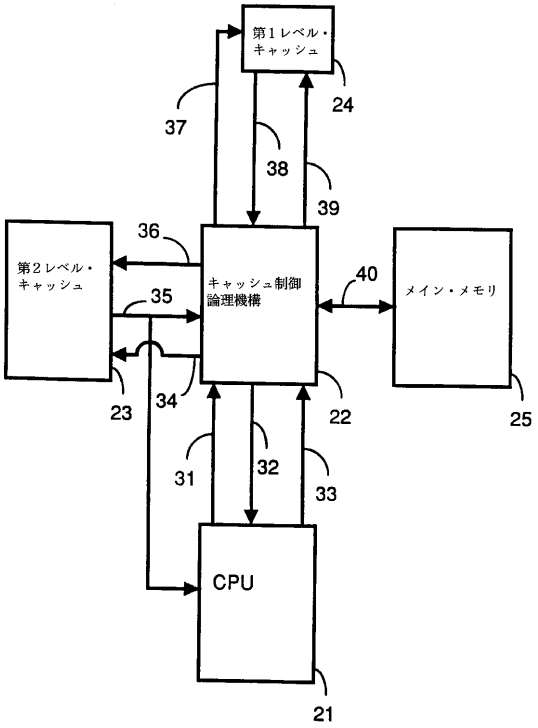
（ル）上記サブステップ（ｃ．２）で、無効とマーク付けされた第１レベル・キャッシュ（２４）のエントリと、第２レベル・キャッシュ（２３）内の第２レベル犠牲キャッシュ・ラインから得たキャッシュ・ライン・フラグメントを含む第１レベル・キャッシュ（２４）内のエントリとが、無効なエントリに含まれることを特徴とする上記（リ）に記載の方法。

【００５４】

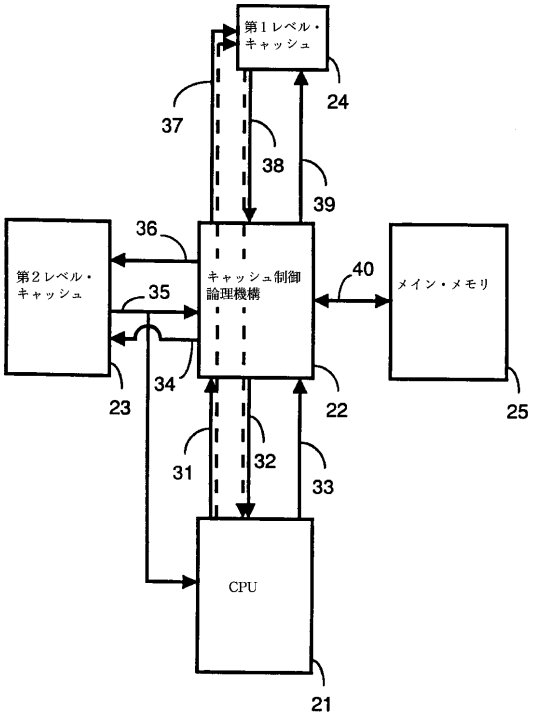
（ヲ）各第２レベル・キャッシュ（２３）ラインが、１６個のワードを含み、各第１レベル・キャッシュ（２４）ライン・フラグメントが、４つのワードを含むことを特徴とする上記（リ）に記載の方法。

【００５５】
前記の議論は、本願発明の典型的な方法および実施例を開示し説明したものに過ぎない。当業者には理解されるように、本願発明は、その趣旨または基本的な特徴から逸脱せずに他の特定の形で実施することができる。したがって、本願発明の開示は、特許請求の範囲に記載された本発明の範囲を例示するものであるが、制限するものではない。
【００５６】
【発明の効果】
以上の説明で明らかなように、本願発明は、単一のキャッシュ・ミスの処理中に将来のキャッシュ・ミスを処理するようにして、多レベル・キャッシュ・システムを含むコンピュータ・プロセッサでのメモリ参照の結果でおきるキャッシュ・ミス・ペナルティを統計的に減少させることが可能となる。
【図面の簡単な説明】
【図１】本願発明の好ましい実施例による多レベル・キャッシュを含むプロセッサ・システムの概略ブロック図である。
【図２】本願発明の好ましい実施例よるメモリ参照の結果として発生する、図１に示したプロセッサでのデータ・フローを示す図である。
【図３】本願発明の好ましい実施例よるメモリ参照の結果として発生する、図１に示したプロセッサでのデータ・フローを示す図である。
【図４】本願発明の好ましい実施例よるメモリ参照の結果として発生する、図１に示したプロセッサでのデータ・フローを示す図である。
【図５】本願発明の好ましい実施例よるメモリ参照の結果として発生する、図１に示したプロセッサでのデータ・フローを示す図である。
【図６】本願発明の好ましい実施例よるメモリ参照の結果として発生する、図１に示したプロセッサでのデータ・フローを示す図である。
【図７】第２レベル・キャッシュでのミスの後の第１レベル・キャッシュの埋込みを処理する様々な方法を示す図である。
【図８】第２レベル・キャッシュでのミスの後の第１レベル・キャッシュの埋込みを処理する様々な方法を示す図である。
【図９】第２レベル・キャッシュでのミスの後の第１レベル・キャッシュの埋込みを処理する様々な方法を示す図である。
【図１０】第２レベル・キャッシュでのミスの後の第１レベル・キャッシュの埋込みを処理する様々な方法を示す図である。
【図１１】本願発明の好ましい実施例よって第２レベル・キャッシュでのミスの後に第１レベル・キャッシュを埋める方法を示すフロー・チャートである。
【符号の説明】
２１　中央演算処理装置（ＣＰＵ）
２２　キャッシュ制御論理機構
２３　第２レベル・キャッシュ
２４　第１レベル・キャッシュ
２５　メイン・メモリ
３１　アドレス線
３２　第１レベル・ロード・データ線
３３　記憶データ線
３４　第２レベル・アドレス線
３５　第２レベル・データ出力線
３６　第２レベル・データ入力線
３７　第１レベル・アドレス線
３８　第１レベル・データ出力線
３９　第１レベル・データ入力線
４０　メモリ・データ線

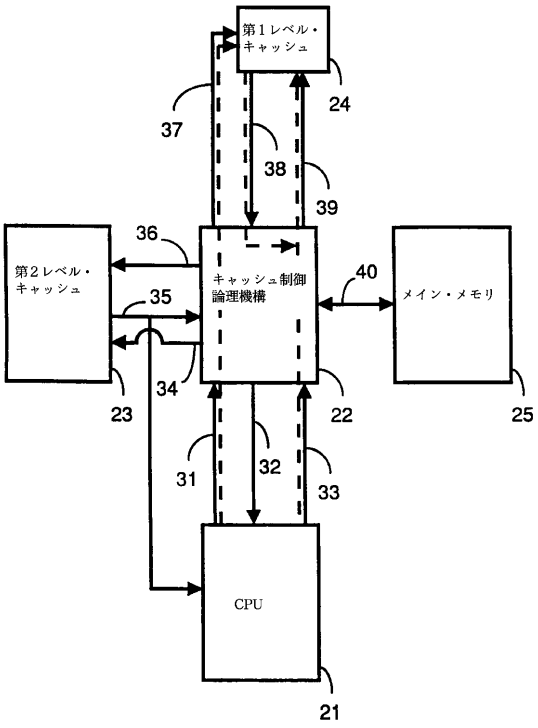
【図１】


第１レベル・
キャッシュ
24
37
38
39
36
第２レベル・
キャッシュ
キャッシュ制御
論理機構
40
メイン・メモリ
35
23
34
22
25
31
32
33
CPU
21

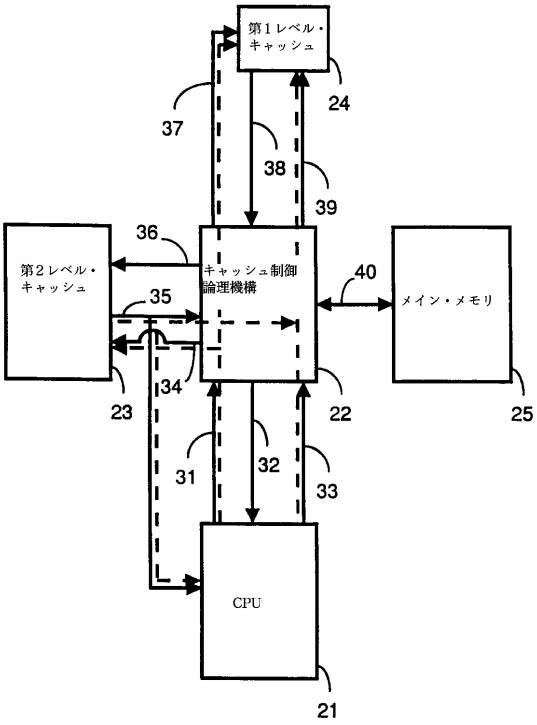
【図２】


第１レベル・
キャッシュ
24
37
38
39
36
第２レベル・
キャッシュ
キャッシュ制御
論理機構
40
メイン・メモリ
35
23
34
22
25
31
32
33
CPU
21

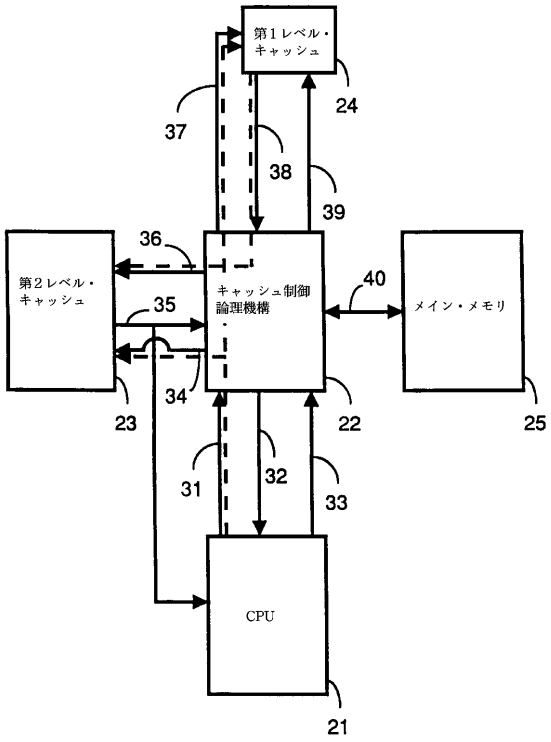
【図３】


第１レベル・
キャッシュ
24
37
38
39
36
第２レベル・
キャッシュ
キャッシュ制御
論理機構
40
メイン・メモリ
35
23
34
22
25
31
32
33
CPU
21

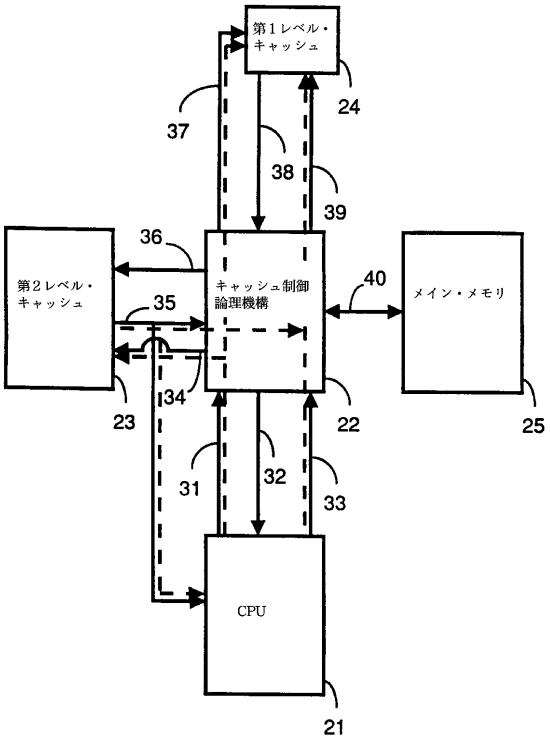
【図４】


第１レベル・
キャッシュ
24
37
38
39
36
第２レベル・
キャッシュ
キャッシュ制御
論理機構
40
メイン・メモリ
35
23
34
22
25
31
32
33
CPU
21

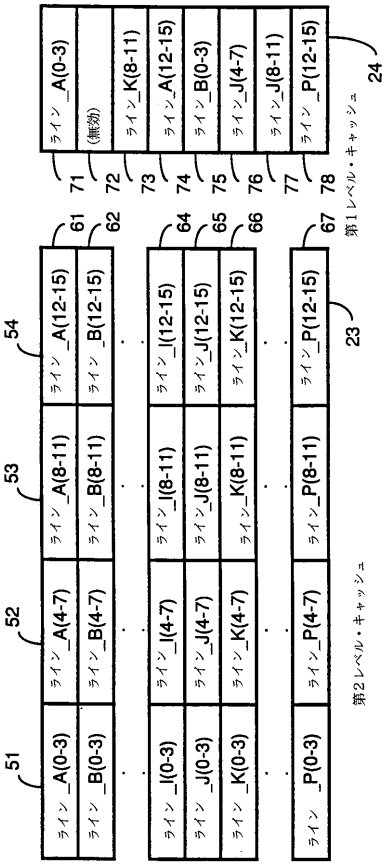
【図５】


第１レベル・
キャッシュ
24
37
38
39
36
第２レベル・
キャッシュ
キャッシュ制御
論理機構
40
メイン・メモリ
35
34
23
22
25
31
32
33
CPU
21

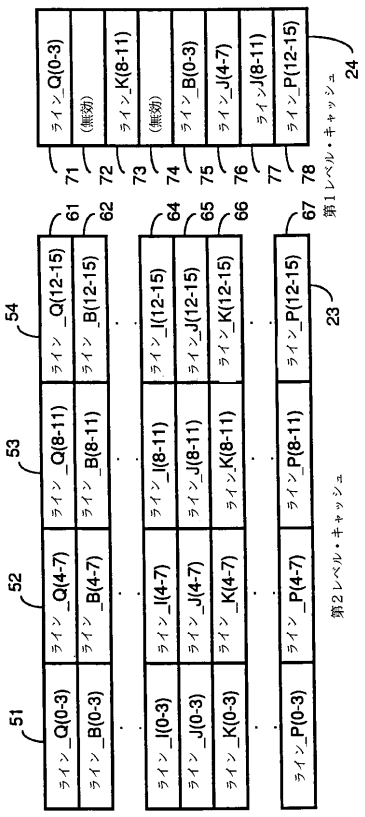
【図６】


第１レベル・
キャッシュ
24
37
38
39
36
第２レベル・
キャッシュ
キャッシュ制御
論理機構
40
メイン・メモリ
35
34
23
22
25
31
32
33
CPU
21

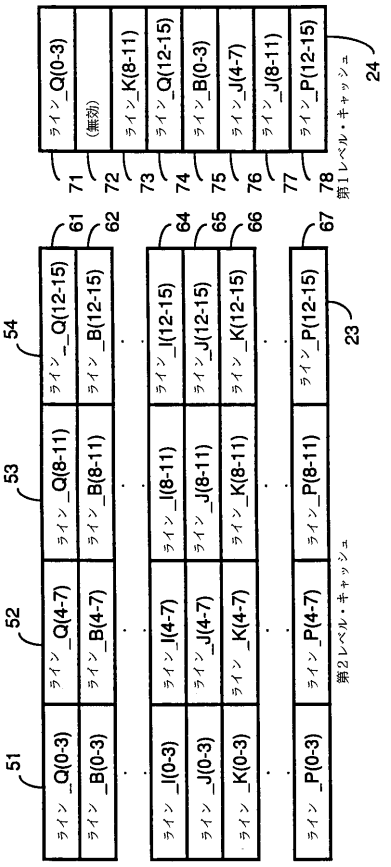
【図７】


ライン_A(0-3)
(無効)
ライン_K(8-11)
ライン_A(12-15)
ライン_B(0-3)
ライン_J(4-7)
ライン_J(8-11)
ライン_P(12-15)
71
72
73
74
75
76
77
78
24
第１レベル・キャッシュ
61
62
64
65
66
67
54
53
52
51
23
ライン_A(12-15)
ライン_B(12-15)
ライン_I(12-15)
ライン_J(12-15)
ライン_K(12-15)
ライン_P(12-15)
ライン_A(8-11)
ライン_B(8-11)
ライン_I(8-11)
ライン_J(8-11)
ライン_K(8-11)
ライン_P(8-11)
ライン_A(4-7)
ライン_B(4-7)
ライン_I(4-7)
ライン_J(4-7)
ライン_K(4-7)
ライン_P(4-7)
ライン_A(0-3)
ライン_B(0-3)
ライン_I(0-3)
ライン_J(0-3)
ライン_K(0-3)
ライン_P(0-3)
第２レベル・キャッシュ

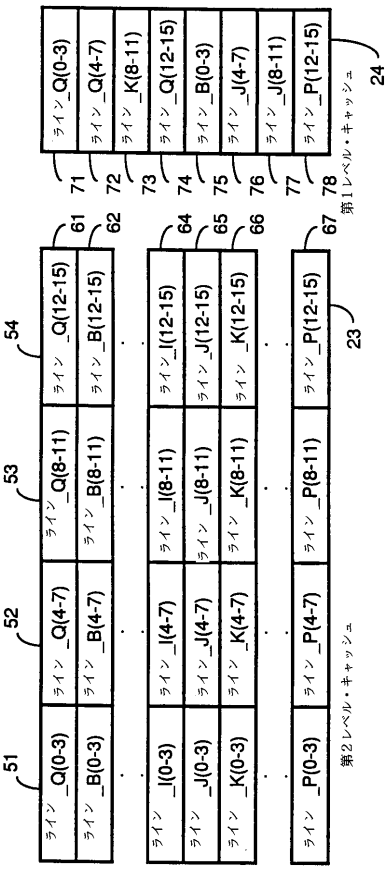
【図８】


ライン_Q(0-3)
(無効)
ライン_K(8-11)
(無効)
ライン_B(0-3)
ライン_J(4-7)
ライン_J(8-11)
ライン_P(12-15)
71
72
73
74
75
76
77
78
24
第１レベル・キャッシュ
61
62
64
65
66
67
54
53
52
51
23
ライン_Q(12-15)
ライン_B(12-15)
ライン_I(12-15)
ライン_J(12-15)
ライン_K(12-15)
ライン_P(12-15)
ライン_Q(8-11)
ライン_B(8-11)
ライン_I(8-11)
ライン_J(8-11)
ライン_K(8-11)
ライン_P(8-11)
ライン_Q(4-7)
ライン_B(4-7)
ライン_I(4-7)
ライン_J(4-7)
ライン_K(4-7)
ライン_P(4-7)
ライン_Q(0-3)
ライン_B(0-3)
ライン_I(0-3)
ライン_J(0-3)
ライン_K(0-3)
ライン_P(0-3)
第２レベル・キャッシュ

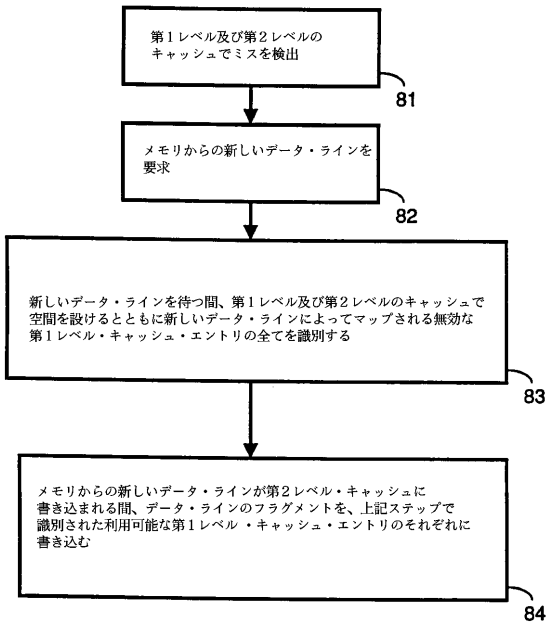
【図９】


ライン_Q(0-3)
(無効)
ライン_K(8-11)
ライン_Q(12-15)
ライン_B(0-3)
ライン_J(4-7)
ライン_J(8-11)
ライン_P(12-15)
第1レベル・キャッシュ
24
71
72
73
74
75
76
77
78
ライン_Q(0-3)
ライン_B(0-3)
ライン_Q(4-7)
ライン_B(4-7)
ライン_Q(8-11)
ライン_B(8-11)
ライン_Q(12-15)
ライン_B(12-15)
ライン_I(0-3)
ライン_J(0-3)
ライン_K(0-3)
ライン_I(4-7)
ライン_J(4-7)
ライン_K(4-7)
ライン_I(8-11)
ライン_J(8-11)
ライン_K(8-11)
ライン_I(12-15)
ライン_J(12-15)
ライン_K(12-15)
ライン_P(0-3)
ライン_P(4-7)
ライン_P(8-11)
ライン_P(12-15)
第2レベル・キャッシュ
23
51
52
53
54
61
62
64
65
66
67

【図１０】


ライン_Q(0-3)
ライン_Q(4-7)
ライン_K(8-11)
ライン_Q(12-15)
ライン_B(0-3)
ライン_J(4-7)
ライン_J(8-11)
ライン_P(12-15)
第1レベル・キャッシュ
24
71
72
73
74
75
76
77
78
ライン_Q(0-3)
ライン_B(0-3)
ライン_Q(4-7)
ライン_B(4-7)
ライン_Q(8-11)
ライン_B(8-11)
ライン_Q(12-15)
ライン_B(12-15)
ライン_I(0-3)
ライン_J(0-3)
ライン_K(0-3)
ライン_I(4-7)
ライン_J(4-7)
ライン_K(4-7)
ライン_I(8-11)
ライン_J(8-11)
ライン_K(8-11)
ライン_I(12-15)
ライン_J(12-15)
ライン_K(12-15)
ライン_P(0-3)
ライン_P(4-7)
ライン_P(8-11)
ライン_P(12-15)
第2レベル・キャッシュ
23
51
52
53
54
61
62
64
65
66
67

【図１１】


第1レベル及び第2レベルの
キャッシュでミスを検出
81
メモリからの新しいデータ・ラインを
要求
82
新しいデータ・ラインを待つ間、第1レベル及び第2レベルのキャッシュで
空間を設けるとともに新しいデータ・ラインによってマップされる無効な
第1レベル・キャッシュ・エントリの全てを識別する
83
メモリからの新しいデータ・ラインが第2レベル・キャッシュに
書き込まれる間、データ・ラインのフラグメントを、上記ステップで
識別された利用可能な第1レベル ・キャッシュ・エントリのそれぞれに
書き込む
84

フロントページの続き

(72)発明者 ディーン・エイ・ミュラ
アメリカ合衆国９５１１７カリフォルニア州サンノゼ、ハイビスカス・レーン ９８５

審査官 清木 泰

(56)参考文献 特開平２－３０１８４３（ＪＰ，Ａ）
特開昭６１－１５６３４６（ＪＰ，Ａ）
特開平５－３２４４６８（ＪＰ，Ａ）
特開平４－２８８６４４（ＪＰ，Ａ）
特開平６－１３９１４７（ＪＰ，Ａ）
PREFETCHING WITH INVALID CACHE ENTRIES, IBM Technical Disclosure Bulletin, 米国, International Business Machines Corporation, １９９０年 ８月, Volume.33, Number 3B, page:46

(58)調査した分野(Int.Cl.[7], ＤＢ名)
G06F12/08-12/12