(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5068552号
(P5068552)
(45)発行日　平成24年11月7日(2012.11.7)　(24)登録日　平成24年8月24日(2012.8.24)

(51)Int.Cl.　F I
G06F 12/08 (2006.01)　G06F 12/08　505C

請求項の数 8　(全 33 頁)

(21)出願番号　特願2007-24090(P2007-24090)
(22)出願日　平成19年2月2日(2007.2.2)
(65)公開番号　特開2008-191824(P2008-191824A)
(43)公開日　平成20年8月21日(2008.8.21)
審査請求日　平成22年1月14日(2010.1.14)

(73)特許権者　302062931
ルネサスエレクトロニクス株式会社
神奈川県川崎市中原区下沼部１７５３番地
(74)代理人　100103894
弁理士　家入　健
(72)発明者　笹井　政尚
神奈川県川崎市中原区下沼部１７５３番地
ＮＥＣエレクトロニクス株式会社内

審査官　桜井　茂行

最終頁に続く

(54)【発明の名称】プリフェッチ方法、及びキャッシュ機構用ユニット

(57)【特許請求の範囲】
【請求項１】
メモリアドレスと、プログラムカウンタ値と、が与えられるステップと、
前記メモリアドレスに基づいて、リンクリストキャッシュテーブルから、前記メモリアドレスをインデックスとする第１のエントリを取得するステップと、
前記第１のエントリにデータが存在しない場合、又は前記第１のエントリにデータが存在し、かつ当該データが前記メモリアドレスとは異なるメモリアドレスである場合、
バススコアテーブルから第２のエントリを取得するステップと、
前記第２のエントリにデータが存在し、かつ前記第２のエントリの再ＰＣ到達サイクル値が０でなく、かつ前記与えられた前記プログラムカウンタ値から前記第２のエントリの最終ＰＣ到達サイクル値を減算したカウントサイクル値と、前記第２のエントリの前記再ＰＣ到達サイクル値と、が一致した場合、
前記リンクリストキャッシュテーブルから、前記第２のエントリのメモリアドレスをインデックスとするエントリを取得し、当該エントリのリンクインデックスとして、前記与えられた前記メモリアドレスをインデックスとするエントリのインデックスを設定するステップと、を有する
プリフェッチ方法。
【請求項２】
前記第１のエントリにデータが存在しない場合、又は前記第１のエントリにデータが存在し、かつ当該データが前記メモリアドレスとは異なるメモリアドレスである場合、

前記第１のエントリのリンクインデックスとして、前記第１のエントリ自身のインデックスを設定するステップをさらに有する
請求項１記載のプリフェッチ方法。
【請求項３】
前記与えられるステップでは、命令サイクルカウンタ値がさらに与えられ、
前記第２のエントリにデータが存在しない場合、
前記第２のエントリの前記メモリアドレスとして、前記与えられた前記メモリアドレスを設定し、前記第２のエントリの前記最終ＰＣ到達サイクル値として、前記与えられた前記命令サイクルカウンタ値を設定し、前記第２のエントリの前記再ＰＣ到達サイクル値として、０を設定する
請求項１又は２記載のプリフェッチ方法。
【請求項４】
前記与えられるステップでは、命令サイクルカウンタ値がさらに与えられ、
前記第２のエントリにデータが存在し、かつ前記第２のエントリの前記再ＰＣ到達サイクル値が０である場合、又は
前記第２のエントリにデータが存在し、かつ前記第２のエントリの前記再ＰＣ到達サイクル値が０でなく、かつ前記与えられた前記プログラムカウンタ値から前記第２のエントリの前記最終ＰＣ到達サイクル値を減算したカウントサイクル値と、前記第２のエントリの前記再ＰＣ到達サイクル値と、が異なる場合、
前記第２のエントリの前記メモリアドレスとして、前記与えられた前記第２メモリアドレスを設定し、前記第２のエントリの前記再ＰＣ到達サイクル値として、前記与えられた命令サイクル値から前記第２のエントリの前記最終ＰＣ到達サイクル値を減算した値を設定し、前記第２のエントリの前記最終ＰＣ到達サイクル値として、前記与えられた前記命令サイクルカウンタ値を設定する
請求項１乃至３いずれか１項記載のプリフェッチ方法。
【請求項５】
インデックスとしての第１メモリアドレスを含む第１のエントリを１以上含むリンクリストキャッシュテーブルと、
第２メモリアドレスと、プログラムカウンタ値と、命令サイクルカウンタ値と、を与えられるディスクリプタコントローラと、
インデックスとしてのプログラムカウンタ値と、前記ディスクリプタコントローラが当該プログラムカウンタ値と共に与えられた前記第２メモリアドレスであるメモリアドレスと、前記ディスクリプタコントローラが前回与えられた前記命令サイクルカウンタ値である最終ＰＣ到達サイクル値と、前記ディスクリプタコントローラが今回与えられた前記命令サイクルカウンタ値から前記最終ＰＣ到達サイクル値を減算した値である再ＰＣ到達サイクル値と、を含む第２のエントリを１つ含むバススコアテーブルと、を有し、
前記ディスクリプタコントローラは、
前記リンクリストキャッシュテーブルから、前記第２メモリアドレスをインデックスとする第１のエントリを取得し、
前記第１のエントリにデータが存在しない場合、又は前記第１のエントリにデータが存在し、かつ当該データが前記第２メモリアドレスとは異なるメモリアドレスである場合、
前記バススコアテーブルから前記第２のエントリを取得し、
前記第２のエントリにデータが存在し、かつ前記第２のエントリの前記再ＰＣ到達サイクル値が０でなく、かつ前記ディスクリプタコントローラが与えられた前記プログラムカウンタ値から前記第２のエントリの前記最終ＰＣ到達サイクル値を減算したカウントサイクル値と、前記第２のエントリの前記再ＰＣ到達サイクル値と、が一致した場合、
前記リンクリストキャッシュテーブルから、前記第２のエントリの前記メモリアドレスをインデックスとするエントリを取得し、当該エントリのリンクインデックスとして、前記ディスクリプタコントローラが与えられた前記第２メモリアドレスをインデックスとする前記エントリのインデックスを設定する

キャッシュ機構用ユニット。
【請求項６】
前記ディスクリプタコントローラはさらに、
前記第１のエントリにデータが存在しない場合、又は前記第１のエントリにデータが存在し、かつ当該データが前記第２メモリアドレスとは異なるメモリアドレスである場合、
前記第１のエントリのリンクインデックスとして、前記第１のエントリ自身のインデックスを設定する
請求項５記載のキャッシュ機構用ユニット。
【請求項７】
前記ディスクリプタコントローラはさらに、
前記第２のエントリにデータが存在しない場合、
前記第２のエントリの前記メモリアドレスとして、前記ディスクリプタコントローラが与えられた前記第２メモリアドレスを設定し、前記第２のエントリの前記最終ＰＣ到達サイクル値として、前記ディスクリプタコントローラが与えられた前記命令サイクルカウンタ値を設定し、前記第２のエントリの前記再ＰＣ到達サイクル値として、０を設定する
請求項５又は６記載のキャッシュ機構用ユニット。
【請求項８】
前記ディスクリプタコントローラはさらに、
前記第２のエントリにデータが存在し、かつ前記第２のエントリの前記再ＰＣ到達サイクル値が０である場合、又は
前記第２のエントリにデータが存在し、かつ前記第２のエントリの前記再ＰＣ到達サイクル値が０でなく、かつ前記ディスクリプタコントローラが与えられた前記プログラムカウンタ値から前記第２のエントリの前記最終ＰＣ到達サイクル値を減算したカウントサイクル値と、前記第２のエントリの前記再ＰＣ到達サイクル値と、が異なる場合、
前記第２のエントリの前記メモリアドレスとして、前記ディスクリプタコントローラが与えられた前記第２メモリアドレスを設定し、前記第２のエントリの前記再ＰＣ到達サイクル値として、前記ディスクリプタコントローラが与えられた命令サイクル値から前記第２のエントリの前記最終ＰＣ到達サイクル値を減算した値を設定し、前記第２のエントリの前記最終ＰＣ到達サイクル値として、前記ディスクリプタコントローラが与えられた前記命令サイクルカウンタ値を設定する
請求項５乃至７いずれか１項記載のキャッシュ機構用ユニット。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、プリフェッチ方法、及びキャッシュ機構用ユニットに関する。
【背景技術】
【０００２】
近年情報処理装置の高速化の進展が著しい。しかしながら、ＣＰＵの演算処理速度に比べて、メモリの動作速度は遅い。そして、メモリの動作速度は、ＣＰＵとメモリとを含んで構成されるシステム全体の高速化を妨げる要因となっている。
【０００３】
上述の問題に対する対策として、ＣＰＵにキャッシュ機構を付加する技術が知られている。キャッシュ機構は、ＣＰＵとメモリとの間のアクセスを監視し、そのアクセス情報を記憶する。そして、再度ＣＰＵからメモリにアクセスがあったとき、動作速度の遅いメモリの代わりに動作速度の速いキャッシュによりＣＰＵにデータが連絡される。
【０００４】
キャッシュは、小容量の記憶装置で構成され、その動作速度は高速である。しかし、キャッシュの容量は小さいため、メモリに比較して記憶できるデータ容量が少ない。従って、キャッシュには、メモリにあるデータのうち、必要となるデータのみが保持されることように構成する（特許文献１乃至４参照、非特許文献１参照）。

【特許文献１】特開２００３－２２８５１８号公報
【特許文献２】特開２０００－４７９４２号公報
【特許文献３】特開２００４－１６４６０７号公報
【特許文献４】特開２００４－１３３９３３号公報
【非特許文献１】ＰＯＷＥＲ４　プロセッサー　概要とチューニング・ガイド　２．４．８ハードウェアによるデータのプリフェッチ
【０００５】
ここで、まず、一般的なプリフェッチの方法として、ストライドによるプリフェッチについて説明する。なお、ストライドによるプリフェッチでは、２回のキャッシュミスに基づいて、その差分値を得る。つまり、１回目のキャッシュミス時のメモリのアドレス（以下、単にメモリアドレスとする）から２回目のキャッシュミス時のメモリアドレスの差分値を得る。そして、２回目のキャッシュミス時のメモリアドレスに、この差分値を加算する。そして、この加算値に対応するメモリアドレスにあるデータをキャッシュにフェッチする。
【０００６】
図１４に、ストライドによるプリフェッチのプリフェッチ機構５１０を示す。プリフェッチ機構５１０は、ＣＰＵ５００、メモリ５０１、キャッシュコントローラ５０２、キャッシュ５０３、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４、を有する。ＣＰＵ５００は、演算処理を実行する。メモリ５０１には、あらかじめデータが格納される。キャッシュ５０３には、プリフェッチされたデータが格納される。キャッシュコントローラ５０２は、ＣＰＵ５００とメモリ５０１との間のアクセスを監視し、そのアクセス情報をＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４に書き込む。キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４に書き込まれた情報に基づいて、適切にプリフェッチを行う。
【０００７】
図１５に、ストライドによるプリフェッチにおいて、配列式のプログラムが実行される場合の説明図を示す。なお、配列式のデータ構造のプログラムでは、プログラムの実行に伴って、順次一定数で増加するメモリアドレスにアクセスされる。
【０００８】
図１５に示すように、ステップ４において、１回目のキャッシュミスが発生する。キャッシュコントローラ５０２は、キャッシュミス発生時にアクセスされたメモリアドレス１０を、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＶａｌｕｅに書き込む。また、キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔａｔｅを１回目とする。
【０００９】
ステップ９において、２回目のキャッシュミスが発生する。キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔａｔｅから、このキャッシュミスが２回目であることを判断する。そして、今回のキャッシュミス時のメモリアドレス２０からＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＶａｌｕｅに設定された１回目のキャッシュミスが発生したメモリアドレス１０を減算した値（差分値）１０を求める。キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔｒｉｄｅに求めた差分値１０を設定し、そのＳｔａｔｅを２回目とする。また、キャッシュコントローラ５０２は、今回のメモリアドレス２０にＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔｒｉｄｅに設定された値１０を加えた３０に対応するメモリアドレス３０にあるデータをメモリ５０１からキャッシュ５０３にフェッチする。
【００１０】
ステップ１４において、メモリアドレス３０にアクセスがある。ステップ９での処理によって、キャッシュ５０３には、あらかじめメモリアドレス３０にあるデータがある。よって、ステップ１４では、メモリ５０１にアクセスする必要はなく、キャッシュ５０３にアクセスすればよい。つまり、キャッシュヒットが得られる。なお、ここでも、キャッシ

ュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔａｔｅから３回目であることを判断する。そして、キャッシュコントローラ５０２は、今回のアドレス３０にＳｔｒｉｄｅの１０を加えたアドレス４０にあるデータをメモリ５０１からキャッシュ５０３にプリフェッチする。

【００１１】

このようにして、それ以降においても、ステップ１４と同様に、キャッシュヒットを得ることができる。

【００１２】

しかしながら、プログラムのデータ構造には、上述の配列式のものではないリンク式のものもある。ストライドによるプリフェッチでは、配列式のプログラムに対しては有効である。しかし、リンクリスト式のプログラムに対しても有効なものであるとは言いがたい。以下、この点について説明する。

【００１３】

図１６に、リンクリスト式のプログラムのリストセルの配置サンプルを示す。図１６に示すように、不定数のリストセル１～５が、矢印に示すように、互いにリンクされて管理される。リストセルの実行は、そのリストセルの順番に従って行われる。ただし、リストセルの配列は、アクセスされるメモリアドレスに対応するものではない。ここでは、リストセルは、リストセル１、リストセル３、リストセル２、リストセル４、リストセル５の順番で配列されている。アクセスされるメモリアドレスのアドレス値は、リストセルの順番に従って、一定数で増加していない。

【００１４】

なお、リンクリスト式のプログラムにおいては、一般的に、互いにリンクされたリンクリストの途中でリストセルの挿入や削除も可能となっている。したがって、上述のように、リストセルとメモリのアドレスとの対応関係が取れないことが起こりえる。このような場合には、上述のストライドによるプリフェッチでは、十分な効果は得られない。

【００１５】

図１７に、ストライドによるプリフェッチにおいて、リンクリスト式のプログラムが実行される場合の説明図を示す。なお、実行されるプログラムは、図１６に示された配置サンプルであることを前提とする。

【００１６】

図１７に示すように、ステップ４において、１回目のキャッシュミスが発生する。キャッシュコントローラ５０２は、キャッシュミスが発生したときにアクセスされたメモリのアドレス１０を、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＶａｌｕｅに書き込む。また、キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔａｔｅを１回目とする。

【００１７】

そして、ステップ９において、２回目のキャッシュミスが発生する。キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔａｔｅから、このキャッシュミスが２回目であることを判断する。そして、メモリアドレス３０のアドレス値からＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＶａｌｕｅに設定された１回目のキャッシュミスが発生したメモリアドレス１０のアドレス値を減算した値（差分値）２０を求める。そして、キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔｒｉｄｅに求めた差分値２０を設定し、そのＳｔａｔｅを２回目とする。また、キャッシュコントローラ５０２は、今回のアドレス３０にＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔｒｉｄｅに設定された値２０を加えて得たアドレス５０にあるデータをメモリ５０１からキャッシュ５０３にフェッチする。

【００１８】

ステップ１４において、アクセスされるメモリアドレスは２０である。メモリ５０１のアドレス２０にあるデータは、２回目のメモリへのアクセス時における処理によって、キャッシュ５０３にあらかじめ格納されていない。つまり、キャッシュミスが発生する。

【００１９】
その後、キャッシュコントローラ５０２は、ＶａｌｕｅＨｉｓｔｏｒｙＴａｂｌｅ５０４のＳｔａｔｅから３回目であることを判断する。そして、キャッシュコントローラ５０２は、今回のアクセスされたメモリのアドレス２０からＶａｌｕｅにある２回目のアドレス３０を減算した－１０をＳｔｒｉｄｅに設定する。そして、Ｓｔａｔｅを２回目とする。そして、今回のアクセスされたメモリのアドレス２０にＳｔｒｉｄｅの－１０を加算して得たアドレス１０にフェッチしようとする。しかし、上述の１回目におけるメモリへのアクセスにより、そのデータが既にキャッシユにあるためフェッチは行わない。
【発明の開示】
【発明が解決しようとする課題】
【００２０】
上述の説明から明らかなように、従来のプリフェッチ方法では、リンクリスト式のプログラムに対応することはできなかった。
【課題を解決するための手段】
【００２１】
本発明にかかるプリフェッチ方法は、コンピュータを用いたリンクリスト構造を含むプログラムを処理する際のプリフェッチ方法であって、前記プログラムを実行し、当該プログラムの命令の実行に伴ってアクセスされたメモリアドレスの順番を記憶する第１実行ステップと、前記第１実行ステップにより記憶された前記順番に基づいて、事前に取得されるべきデータをメモリからキャッシュにフェッチし、前記プログラムを実行する第２実行ステップと、を具備する。
【００２２】
本発明にかかるプリフェッチ方法は、コンピュータを用いたリンクリスト構造を含むプログラムを処理する際のプリフェッチ方法であって、前記プログラムの実行は、複数のメモリアクセスを伴い、前記プログラムの第１実行時において、第２メモリアクセスがあったとき、当該第２メモリアクセス前に行われた第１メモリアクセス時のメモリアドレスをインデックスとするリンクリストキャッシュテーブルの第１エントリーに当該第２メモリアクセス時のメモリアドレスをインデックスとする前記リンクリストキャッシュテーブルの第２エントリーをリンクさせ、前記プログラムの第２実行時において、前記第１メモリアクセスに対応する第３メモリアクセスがあったとき、当該第３メモリアクセス時のメモリアドレスをインデックスとする前記第１エントリーにリンクされた前記第２エントリーのメモリアドレスを取得し、取得した当該メモリアドレスにあるデータをキャッシュにフェッチする。
【００２３】
本発明にかかるキャッシュ機構用ユニットは、インデックスとしてのメモリアドレスを含んで構成されるエントリーを複数含むリンクリストキャッシュテーブルと、インデックスとしてのプログラムカウンタのカウント値、当該カウント値に対応するメモリアドレスを含んで構成されるエントリーを有するバススコアテーブルと、第１メモリアクセス時のメモリアドレスをインデックスとするリンクリストキャッシュテーブルの第１エントリーに、第２メモリアクセス時のメモリアドレスをインデックスとする前記リンクリストキャッシュテーブルの第２エントリーをリンクさせるディスクリプタコントローラと、を備える。
【発明の効果】
【００２４】
リンクリスト式のプログラムであっても、プリフェッチによるキャッシュヒットの確率を高めることができる。
【発明を実施するための最良の形態】
【００２５】
以下、図面を参照しつつ、本発明の実施の形態について説明する。なお、図面は簡略的なものであるから、この図面の記載を根拠として本発明の技術的範囲を狭く解釈してはな

らない。また、同一の要素には、同一の符号を付し、重複する説明は省略するものとする。また、図面は、もっぱら技術的事項の説明のためのものであり、図面に示された要素の正確な大きさ等は反映していない。

【００２６】

〔第１の実施の形態〕

図１に、第１の実施の形態にかかるキャッシュ機構を説明するためのブロック図を示す。図１に示された演算処理機構１００は、メモリ１、キャッシュコントローラ２、ＣＰＵ３、キャッシュ４、プロファイラ５、プロファイルデータキュー６、ディスクリプタコントローラ７、バススコアテーブル８、リンクリストキャッシュテーブル９、キャッシュコントロールディスクリプタテーブル１０、を備える。

【００２７】

メモリ１には、実行されるプログラム、その他のデータが格納される。ＣＰＵ３は、メモリ１に格納されたプログラムを実行する。また、メモリ１に格納されたデータにアクセスする。キャッシュコントローラ２は、メモリ１のデータをキャッシュ４にフェッチする。また、キャッシュコントローラ２は、ＣＰＵ３－メモリ１間のアクセス、ＣＰＵ３－キャッシュ４間のアクセスを監視したりする。本実施形態におけるキャッシュコントローラ２は、キャッシュコントロールディスクリプタテーブル１０を参照して、適切なタイミングでプリフェッチする。従って、キャッシュ４には、プリフェッチにより必要なデータが格納される。

【００２８】

本実施形態においては、上述の、キャッシュコントローラ２、キュッシュ４、プロファイラ５、プロファイルデータキュー６、ディスクリプタコントローラ７、バススコアテーブル８、リンクリストキャッシュテーブル９、キャッシュコントロールディスクリプタテーブル１０からキャッシュ機構が構成される。

【００２９】

以下、キャッシュ機構に含まれる構成要素について説明する。

【００３０】

プロファイラ５は、メモリ１へのアクセスを監視し、メモリ１へのアクセスがないタイミングを検出する。そして、そのタイミングにおけるプログラムカウンタのカウント値（以下、単にＰＣとする）をＣＰＵ３から取得し、このＰＣをプリフェッチ可能ＰＣとしてプロファイルデータキュー６に連絡する。

【００３１】

プロファイラ５は、時間計測タイマを持ち、メモリ１からキャッシュ４へのデータの読み込みにどれだけの時間がかかるのかを計測する。そして、この計測結果に基づいて、プリフェッチを行うタイミングを検出する。プロファイラ５は、メモリ１へのアクセスが終了したとき、時間計測タイマを動作させる。また、メモリ１へのアクセスが開始したとき、時間計測タイマを停止する。時間計測タイマが動作中であれば、時間計測タイマのカウント値をダウンカウントする。時間計測タイマのカウント値が０であれば、記憶済みのバス開放時のＰＣをプリフェッチ可能ＰＣとして、プロファイルデータキュー６に連絡する。

【００３２】

プロファイラ５により行われる処理を図２のフローチャートに示す。プロファイラ５は、メモリ１へのアクセスを監視する。プロファイラ５は、メモリ１へのアクセスの開始を検出する（Ｓ１）。アクセスの開始を検出したとき、プロファイラ５は、時間計測タイマを停止させる（Ｓ２）。そして、プロファイラ５は、そのアクセスが、キャッシュコントローラ２によるプリフェッチ時のアクセスかどうかを判断する（Ｓ３）。つまり、キャッシュコントロールディスクリプタテーブル１０を参照して、キャッシュコントローラ２がメモリ１のデータをキャッシュ４にプリフェッチするときのアクセスであるか判断する。そして、プリフェッチ時のアクセスであれば、その時のＰＣをＣＰＵ３から取得し、そのＰＣをプリフェッチ可能ＰＣとしてプロファイルデータキュー６に登録する（Ｓ４）。そ

して、再び、開始点Ａに戻る。また、Ｓ３において、プリフェッチ時のアクセスではない場合にも、再び、開始点Ａに戻る。

【００３３】

Ｓ１において、メモリ１へのアクセスの開始を検出しないとき、プロファイラ５は、メモリ１へのアクセスの終了を検出する（Ｓ５）。そして、メモリ１へのアクセスの終了を検出したとき、その時点でのＰＣをＣＰＵ３から取得し、そのＰＣを記憶する（Ｓ６）。また、時間計測タイマを動作させる（Ｓ７）。

【００３４】

Ｓ５において、メモリ１へのアクセスの終了を検出しないとき、プロファイラ５は、時間計測タイマが停止中かを確認する（Ｓ８）。時間計測タイマが停止中であれば、開始点Ａに戻る。時間計測タイマが動作中であれば、ダウンカウントさせる（Ｓ９）。そして、時間計測タイマのカウント値が０であるかを確認する（Ｓ１０）。時間計測タイマのカウント値が０であれば、記憶済みのＰＣをプロファイルデータキューにプリフェッチ可能ＰＣとして登録する（Ｓ１１）。そして、その時のＰＣをバス開放時のＰＣとする（Ｓ１２）。そして、上述のＳ６、Ｓ７のステップを実行後、開始点Ａに戻る。Ｓ１０において、時間計測タイマのカウント値が０ではない場合には、開始点Ａに戻る。

【００３５】

また、プロファイラ５は、ディスクリプタコントローラ７にＣＰＵ３からメモリ１にアクセスがあったときの情報を与える。

【００３６】

プロファイラ５は、命令サイクルカウンタを持ち、ＣＰＵ３が命令を実行することを検出する度に、そのカウント値を１ずつ増加させる。また、プロファイラ５は、ＣＰＵ３のレジスタから、ＣＰＵ３によるメモリ１へのアクセス時（以下、メモリアクセス時と呼ぶこともある）のＰＣ、メモリ１のアドレス（以下、メモリアドレスと呼ぶ）を取得する。そして、この取得した情報（ＰＣ、メモリアドレス）と上述の命令サイクルカウンタのカウント値（以下、単にカウント値と呼ぶこともある）とを、ディスクリプタコントローラ７に連絡する。

【００３７】

プロファイラ５により行われる処理を図３のフローチャートに示す。プロファイラ５は、ＣＰＵ３が命令を１つ実行したことを検出する（Ｓ１）。そして、命令サイクルカウンタのカウント値を１増加させる（Ｓ２）。そして、そのＣＰＵ３において処理される命令が、ＣＰＵ３からメモリ１へのアクセスであるのか確認する（Ｓ３）。ＣＰＵ３からメモリ１へのアクセスである場合には、プロファイラ５は、ディスクリプタコントローラ７に情報を通知する（Ｓ４）。なお、通知される情報は、ＣＰＵ３によるメモリアクセス時のＰＣ、メモリアドレス、カウント値である。そして、開始点Ａに戻る。Ｓ３においてメモリアクセスではない場合には、開始点Ａに戻る。

【００３８】

上述のように、ＣＰＵからメモリへのアクセス時における所定の情報（ＰＣ、メモリアドレス、カウント値）が、プロファイラ５からディスクリプタコントローラ７に与えられる。換言すると、プロファイラ５は、ＣＰＵからメモリへのアクセスがあるたびに、所定の情報をディスクリプタコントローラ７に連絡する。そして、ディスクリプタコントローラ７は、プロファイラ５から与えられた情報に基づいて、リンクリストキャッシュテーブル９、バススコアテーブル８に後述の動作を行う。これによって、リンクリスト式のプログラム（リンクリスト構造を含むプログラム）であっても、キュッシュヒットを得る確率を高めることができる。

【００３９】

ディスクリプタコントローラ７は、プロファイルデータキュー６に格納されたプリフェッチ可能ＰＣをキャッシュコントロールディスクリプタテーブル１０に連絡する。また、リンクリストキャッシュテーブル９を参照した結果に基づいて、プリフェッチされるべきメモリアドレスをキャッシュコントロールディスクリプタテーブル１０に連絡する。

【００４０】
ここで、ディスクリプタコントローラ７の動作について詳述する前に、バススコアテーブル８、リンクリストキャッシュテーブル９について説明する。
【００４１】
図４に、バススコアテーブル８とリンクリストキャッシュテーブル９を説明する模式図を示す。図４に示すように、バススコアテーブル８は、エントリー８ｅを有する。エントリー８ｅは、インデックスとしてのＰＣ、メモリアドレスを格納するフィールドを有する。なお、格納されるＰＣとメモリアドレスとは、プロファイラ５から同時に与えられたときのものである。なお、バススコアテーブル８のエントリーには、データの在／無、メモリアドレスの有効／無効、最終ＰＣ到達サイクル、再ＰＣ到達サイクル、といったフィールドもある。最終ＰＣ到達サイクルは、プロファイラ５から与えられた最後のカウント値である。再ＰＣ到達サイクルは、今回のプロファイラ５から与えられたカウント値から、前回の最終ＰＣ到達サイクルを減算した値である。
【００４２】
図４に示すように、リンクリストキャッシュテーブル９は、エントリー９ｅ１、９ｅ２を有する。各エントリーは、インデックスとしてのメモリアドレス、リンクインデックスを有する。エントリー９ｅ１のリンクインデックスに他のエントリー９ｅ２のインデックスが設定されることにより、エントリー９ｅ１とエントリー９ｅ２とは互いに関連づけ（リンク）られる。なお、リンクインデックスには、他のエントリーのインデックスのほか、自身のエントリーのインデックスも設定される。また、各エントリーには、エントリーにおけるデータの有無、といった情報も格納される。
【００４３】
上述のように、ＣＰＵ３によるメモリ１へのアクセスがあったとき、ＰＣ、メモリアドレス、カウント値が、プロファイラ５からディスクリプタコントローラ７に連絡される。そして、ディスクリプタコントローラ７は、リンクリストキャッシュテーブル９に次のように動作をする。
【００４４】
ディスクリプタコントローラ７は、プロファイラ５から与えられたメモリアドレスに基づいて、そのメモリアドレスをインデックスとするリンクリストキャッシュテーブル９のエントリーを検索する。そして、取得したエントリーの状態に応じて、そのエントリーを次のように設定する。
【００４５】
１－２－Ａ：取得したエントリーにデータが無い場合。
ディスクリプタコントローラ７は、取得したエントリーのリンクインデックスに、取得したエントリー自身のインデックスを設定する。また、ディスクリプタコントローラ７は、プロファイラ５から与えられた情報に含まれるメモリアドレスを、キャッシュコントロールディスクリプタテーブル１０にプリフェッチされるべきメモリアドレス（プリフェッチ用メモリアドレス）として連絡する。
【００４６】
１－２－Ｂ：取得したエントリーにデータが在るが、プロファイラ５から与えられたメモリアドレスとは異なるアドレスである場合。
ディスクリプタコントローラ７は、取得したエントリーのリンクインデックスに、取得したエントリー自身のインデックスを設定する。また、ディスクリプタコントローラ７は、プロファイラ５から与えられたメモリアドレスを、キャッシュコントロールディスクリプタテーブル１０にプリフェッチ用メモリアドレスとして連絡する。
【００４７】
１－２－Ｃ：取得したエントリーにデータが在り、プロファイラ５から与えられたメモリアドレスと同じアドレスがある場合。
ディスクリプタコントローラ７は、取得したエントリーのリンクインデックスにより特定されるエントリーのメモリアドレスを、キャッシュコントロールディスクリプタテーブル

１０にプリフェッチ用メモリアドレスとして連絡する。換言すると、ディスクリプタコントローラ７は、プロファイラ５から与えられたメモリアドレスをキャッシュコントロールディスクリプタテーブル１０に連絡しない。なお、取得したエントリーのリンクインデックスには、あらかじめ、後述の２－２－Ｃ－ｂの手順によって、他のエントリーとリンクされている。

【００４８】

また、１－２－Ａ、１－２－Ｂの場合、ディスクリプタコントローラ７は、バススコアテーブル８に次のように動作をする。ディスクリプタコントローラ７は、プロファイラ５から与えられたＰＣに基づいて、そのＰＣをインデックスとするバススコアテーブル８のエントリーを取得する。そして、取得したエントリーを次のように設定する。

【００４９】

２－２－Ａ：バススコアテーブル８のエントリーにデータが無い場合。

プロファイラ５から与えられたメモリアドレスを、そのエントリーのメモリアドレスに設定する。また、プロファイラ５から与えられたカウント値を、そのエントリーの最終ＰＣ到達サイクルに設定する。また、そのエントリーの再ＰＣ到達サイクルを０とする。

【００５０】

２－２－Ｂ：バススコアテーブル８のエントリーにデータが在り、再ＰＣ到達サイクル０の場合。

プロファイラ５から与えられたメモリアドレスを、そのエントリーのメモリアドレスに設定する。また、プロファイラ５から与えられたカウント値から、そのエントリーにある最終ＰＣ到達サイクルの値を減算する。そして、この減算値を、そのエントリーの再ＰＣ到達サイクルに設定する。また、そのエントリーの最終ＰＣ到達サイクルに、プロファイラ５から与えられたカウント値を設定する。

【００５１】

２－２－Ｃ：バススコアテーブル８のエントリーにデータが在り、再ＰＣ到達サイクル０でない場合。

ディスクリプタコントローラ７は、プロファイラ５から与えられたカウント値から、そのエントリーにある最終ＰＣ到達サイクルを減算する。そして、減算して得た値（以下、カウントサイクル値と呼ぶ）と、そのエントリーの再ＰＣ到達サイクルを比較する。ディスクリプタコントローラ７は、比較した結果に基づいて、そのエントリーを次のように設定する。

【００５２】

２－２－Ｃ－ａ：カウントサイクル値と再ＰＣ到達サイクルの値が異なる場合。

ディスクリプタコントローラ７は、２－２－Ｂの場合と同じように動作する。すなわち、ディスクリプタコントローラ７は、プロファイラ５から与えられた命令サイクルカウントのカウント値から最終ＰＣ到達サイクルを減算する。そして、減算して得たカウントサイクル値を、そのエントリーの再ＰＣ到達サイクルに設定する。また、エントリーの最終ＰＣ到達サイクルに、プロファイラ５から与えられた命令サイクルカウンタのカウントを設定する。また、プロファイラ５から与えられたメモリアドレスを、そのエントリーのメモリアドレスとして設定する。

【００５３】

２－２－Ｃ－ｂ：カウントサイクル値と再ＰＣ到達サイクルの値が同じ場合。

ディスクリプタコントローラ７は、次のようにバススコアテーブル８のエントリーに動作する。そのエントリーにあるメモリアドレスに基づいて、そのメモリアドレスをインデックスとするリンクリストキャッシュテーブル９のエントリーを検索する。そして、取得したリンクリストキャッシュテーブル９のエントリーのリンクインデックスに、プロファイラ５から与えられたメモリアドレスをインデックスとするリンクリストキャッシュテーブル９のエントリーのインデックスを設定する。これによって、各エントリーはリンクされる。そして、リンクリストキャッシュテーブル９には、プログラムの実行に従ってアクセスされるメモリアドレスの順番が記憶される。この点は、以降の説明からも明らかとなる

。この後、プロファイラ５から与えられたメモリアドレスを、そのエントリーのメモリアドレスとして設定する。

【００５４】

さらに、ディスクリプタコントローラ７の動作を、図５のフローチャートを用いて説明する。図５に示すように、まず、ディスクリプタコントローラ７は、後述するキャッシュコントロールディスクリプタテーブル１０に対する処理を行う（Ｓ１）。次に、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを取得する（Ｓ２）。次に、取得したエントリーにデータが無いか確認する（Ｓ３）。エントリーにデータが無い場合、そのエントリーを２－２－Ａの設定をする（Ｓ４）。そして、開始点Ｂに戻る。

【００５５】

取得したエントリーにデータがある場合には、そのエントリーにある再ＰＣ到達サイクル０か確認する（Ｓ５）。その値が０である場合には、２－２－Ｂの設定する（Ｓ６）。

【００５６】

再ＰＣ到達サイクル０ではない場合（Ｓ５）、上述のカウントサイクル値（プロファイラ５から与えられたカウント値から、そのエントリーにある最終ＰＣ到達サイクルを減算して得た値）が、取得したエントリーの再ＰＣ到達サイクルと違うか確認する（Ｓ７）。値が異なる場合、２－２－Ｂの設定する（Ｓ６）。そして、開始点Ｂに戻る。

【００５７】

カウントサイクル値と取得したエントリーの再ＰＣ到達サイクルが同じ場合には、２－２－Ｃ－ｂのように処理を実行する（Ｓ８）。そして、開始点Ｂに戻る。

尚、Ｓ７において、カウントサイクル値と再ＰＣ到達サイクルの値が同じであれば、一定の間隔でメモリアクセスが行われていることを確認できる。よって、２－２－Ｃ－ｂの処理に基づいて記憶されるメモリアドレスの順番の信頼性を高めることができる。

【００５８】

次に、図５のＳ１（ディスクリプタコントローラ７によるキャッシュコントロールディスクリプタテーブル１０への動作）について説明する。図６のフローチャートに示すように、ディスクリプタコントローラ７は、プロファイラ５から与えられた情報に基づいて、ＣＰＵ３からメモリ１にリードアクセスがあったか判断する（Ｓ１）。リードアクセスがあった場合、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチ動作の開始点となるプリフェッチ開始ＰＣを得る（Ｓ２）。次に、プロファイラ５から与えられたＰＣに基づいて、バススコアテーブル８からエントリーを取得する（Ｓ３）。次に、取得したエントリーのメモリアドレスと、プロファイラ５から与えられたメモリアドレスとが同じか判断する（Ｓ４）。そして、アドレスが同じでない場合、プロファイラ５から与えられたメモリアドレスをプリフェッチ用のメモリアドレスとする（Ｓ５）。

【００５９】

取得したアドレスが同じ場合、次のようにプレフェッチ用メモリアドレスを設定する（Ｓ６）。上述の１－２－Ｂの場合、プロファイラ５から与えられたメモリアドレスを、プリフェッチ用のメモリアドレスとして設定する。上述の１－２－Ｃの場合、取得したバススコアテーブル８のエントリーのメモリアドレスをインデックスとするリンクリストキャッシュテーブル９のエントリーのリンクインデックスにリンクされたリンクリストキャッシュテーブル９のエントリーのメモリアドレスを、プリフェッチ用のメモリアドレスとして設定する。

【００６０】

そして、ディスクリプタコントローラ７は、上述のＳ５又はＳ６で設定したプリフェッチアドレスと、上述のＳ２で取得したプリフェッチ開始ＰＣとをキャッシュコントロールディスクリプタテーブル１０に連絡する（Ｓ７）。そして、上述の１－２－Ａ、１－２－Ｂの場合、取得したリンクリストキャッシュテーブル９のエントリーを上述したように設定する（Ｓ８）。なお、リードアクセスが無い場合（Ｓ１）には、上述のＳ８の処理が実行される。

【００６１】
次に、キャッシュコントロールディスクリプタテーブル１０について説明する。キャッシュコントロールディスクリプタテーブル１０は、ＰＣをインデックスとするエントリーがある。エントリーは、エントリーの有効／無効、プリフェッチ開始ＰＣ、プリフェッチ用メモリアドレス、のフィールドを有する。
【００６２】
上述のように、ディスクリプタコントローラ７は、キャッシュコントロールディスクリプタテーブル１０にプリフェッチ可能ＰＣを連絡する。また、リンクリストキャッシュテーブル９を参照した結果に基づいて、プリフェッチすべきメモリアドレスをキャッシュコントロールディスクリプタテーブル１０に連絡する。そして、キャッシュコントロールディスクリプタテーブル１０のエントリーには、プリフェッチ開始ＰＣとプリフェッチ用メモリアドレスが設定される。
【００６３】
なお、キャッシュコントロールディスクリプタテーブル１０は、ディスクリプタコントローラ７からの指令に基づいて、エントリーの登録、エントリーの削除が行われる。
【００６４】
次に、キャッシュコントローラ２について説明する。キャッシュコントローラ２は、ＣＰＵ３から与えられる実行中のＰＣをインデックスとして、上述のように設定されたキャッシュコントロールディスクリプタテーブル１０から該当するエントリーを探す。ＣＰＵ３から与えられるＰＣをインデックスとするエントリーがある場合、そのタイミングで、そのエントリーにあるメモリアドレスにアクセスし、そのアドレスにあるデータをメモリ１からキャッシュ３にフェッチする。
【００６５】
キャッシュコントローラ２の動作を図７のフローチャートに示す。図７に示すように、キャッシュコントローラ２は、ＣＰＵ３から与えられる実行中のＰＣに基づいて、キャッシュコントロールディスクリプタテーブル１０に該当するエントリーを探す（Ｓ１）。そして、該当するエントリーがあるかどうか判断する（Ｓ２）。該当するエントリーがある場合には、そのエントリーにあるメモリアドレスにアクセスし、そのアドレスにあるデータをメモリ１からキャッシュ３にフェッチする（Ｓ３）。フェッチした後は、開始点Ａに戻る。該当するエントリーがない場合には、開始点Ａに戻る。
【００６６】
本実施形態においては、ディスクリプタコントローラ７は、上述の２－２－Ｃ－ｂのように動作する。これによって、リンクリストキャッシュテーブル９には、実行されるリンクリスト式のプログラムの命令の実行に伴ってアクセスされるメモリアドレスの順番が記憶される。従って、次回又はそれ以降、同じプログラムが実行された場合には、リンクリストキャッシュテーブル９に保存されたメモリアドレスの順番を活用してプリフェッチする。これによって、リンクリスト式のプログラムにおいても、キャッシュヒットする確率を高めることができる。すなわち、リンクリスト式のプログラムのように、リストセルの順番に従ってアクセスされるメモリアドレスの値が一定数で増加していないものであっても、リンクリストキャッシュテーブル９に保存された情報を活用することにより、適切にプリフェッチできる。
【００６７】
本実施形態のキャッシュ機構の動作について補足する。上述の２－２－Ｃ－ｂ：カウントサイクル値と再ＰＣ到達サイクルの値が同じ場合では、ディスクリプタコントローラ７はバススコアテーブル８のエントリーにあるメモリアドレスに基づいて、そのメモリアドレスをインデックスとするリンクリストキャッシュテーブル９のエントリーを検索する。そして、取得したエントリーのリンクインデックスに、プロファイラ５から与えられたメモリアドレスをインデックスとするエントリーのインデックスを設定する。これによって、各エントリーはリンクされる。
【００６８】

プログラムが次回又はそれ以降に実行されるとき、ディスクリプタコントローラ７は、リンクリストキャッシュテーブル９にアクセスする。そして、１－２－Ｃ：取得したエントリーにデータが格納されており、メモリアクセス時のメモリアドレスと同じメモリアドレスがある場合が発生する。そして、ディスクリプタコントローラ７は、キャッシュコントロールディスクリプタテーブル１０に取得したエントリーのリンクインデックスにより特定されるエントリーにあるメモリアドレスをプリフェッチ用メモリアドレスとして連絡する。

【００６９】

本実施形態では、リンクリストキャッシュテーブル９に、プログラムの実行に従ってアクセスされるメモリアドレスの順番が記憶される。従って、ディスクリプタコントローラ７は、リンクリストキャッシュテーブル９を参照することで、アクセスされる順番どおりに、プリフェッチ用メモリアドレスを取得できる。ディスクリプタコントローラ７は、プリフェッチ開始ＰＣとともにプリフェッチ用メモリアドレスをキャッシュコントロールディスクリプタテーブル１０に連絡する。キャッシュコントローラ２は、キャッシュコントロールディスクリプタテーブル１０を参照することで、適切にプリフェッチを行うことができる。すなわち、本実施形態におけるキャッシュ機構は、リンクリスト式のプログラムにおいても、キャッシュヒットを得ることができる。

【００７０】

ここで、以下、図８に示されたプリフェッチの実行トレース表を用いて、本実施形態におけるプリフェッチ方法について説明する。なお、実行されるプログラムは、図１６に示された配置サンプルであることを前提とする。また、プロファイラ５は、ループごとに５行目のＰＣをプロファイルデータキュー６にプリフェッチ可能ＰＣとして連絡するものとする。

【００７１】

まず、リンクリスト式のプログラムの１回目の実行（プログラムの第１実行時）について説明する。なお、ループごとの実行結果については、図９を参照するものとする。

【００７２】

Ｐ＝１０のループ（サイクル２～８間のループ）では、リストセル１の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス１０である。

【００７３】

ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル７のとき、ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスを、メモリアドレス１０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【００７４】

また、ディスクリプタコントローラ７は、バススコアテーブル８に格納されたエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル７のとき、ディスクリプタコントローラ７は、バススコアテーブル８にインデックスＰＣ７とするエントリーを登録する。また、そのエントリーに、データが在るとする。また、そのエントリーのメモリアドレスに、メモリアドレス１０を設定する。また、再ＰＣ到達サイクルに０を設定する。また、最終ＰＣ到達サイクルに７を設定する。

【００７５】

Ｐ＝３０のループ（サイクル９～１５間のループ）では、リストセル２の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス３０である。

【００７６】

ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとして、リンク

リストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル１４のとき、ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス３０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【００７７】

また、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のサイクル７で登録したエントリーがある。そして、ディスクリプタコントローラ７は、そのエントリーにおける再ＰＣ到達サイクル０であることを確認する。そして、サイクル１４のとき、そのエントリーのＰＣを１４に変更する。また、そのエントリーのメモリアドレスに、メモリアドレス３０を設定する。また、そのエントリーの再ＰＣ到達サイクルに７を設定する。また、最終ＰＣ到達サイクルに１４を設定する。

【００７８】

Ｐ＝２０のループ（サイクル１６～２２間のループ）では、リストセル３の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス２０である。

【００７９】

ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル２１のとき、ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス２０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【００８０】

また、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のようにサイクル１４で設定したエントリーがある。そして、ディスクリプタコントローラ７は、現在のサイクル２１からそのエントリーの最終ＰＣ到達サイクルの１４を減算する。そして、減算して得たカウントサイクル値７と、そのエントリーにおける再ＰＣ到達サイクル７は等しいことを確認する。

【００８１】

そして、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス３０（前回のループにおいてアクセスされたメモリアドレス）に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。そして、取得したメモリアドレス３０をインデックスとするエントリー（前回のループにおいてアクセスされたメモリアドレスをインデックスとするエントリー）と、現在のメモリアドレス２０をインデックスとするリンクリストキャッシュテーブル９のエントリー（今回のループにおいてアクセスされたメモリアドレスをインデックスとするエントリー）とをリンクさせる。具体的には、メモリアドレス３０をインデックスとするエントリーのリンクインデックスに、メモリアドレス２０をインデックスとするエントリーのインデックスを設定する。

【００８２】

図９に示すように、メモリアドレス３０をインデックスとするエントリーにおけるリンクインデックスに設定された値が３０から２０に変更される。つまり、メモリアドレス３０をインデックスとするエントリーは、自身のエントリーにリンクされていた状態から、メモリアドレス２０をインデックスとするエントリーにリンクされた状態となる。

【００８３】

また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを２

０に設定する。また、バススコアテーブル８の最終ＰＣ到達サイクルに２１を設定する。
【００８４】
Ｐ＝４０のループ（サイクル２３～２９間のループ）では、リストセル４の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス４０である。
【００８５】
ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル２８のとき、ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス４０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。
【００８６】
また、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のようにサイクル２１で設定したエントリーがある。そして、ディスクリプタコントローラ７は、現在のサイクル２８からそのエントリーの最終ＰＣ到達サイクルの２１を減算する。減算して得たカウントサイクル値７と、そのエントリーにおける再ＰＣ到達サイクルの値７は等しいことを確認する。
【００８７】
そして、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス２０（前回のループにおいてアクセスされたメモリアドレス）に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。そして、その探したメモリアドレス２０をインデックスとするエントリー（前回のループにおいてアクセスされたメモリアドレスをインデックスとするエントリー）と、現在のメモリアドレス４０をインデックスとするエントリー（今回のループにおいてアクセスされたメモリアドレスをインデックスとするエントリー）とをリンクさせる。具体的には、メモリアドレス２０をインデックスとするエントリーのリンクインデックスに、メモリアドレス４０をインデックスとするエントリーのインデックスを設定する。
【００８８】
図９に示すように、メモリアドレス２０をインデックスとするエントリーにおけるリンクインデックスに設定されたインデックス値が２０から４０に変更される。つまり、メモリアドレス２０をインデックスとするエントリーは、自身のエントリーにリンクされていた状態から、メモリアドレス４０をインデックスとするエントリーにリンクされた状態となる。
【００８９】
また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを４０に設定する。また、バススコアテーブル８の最終ＰＣ到達サイクルに２８を設定する。
【００９０】
このようにして、リンクリスト式のプログラムが１回実行されることにより、メモリアドレス３０をインデックスとするエントリーのリンクインデックスには、メモリアドレス２０をインデックスとするエントリーがリンクされる。また、メモリアドレス２０をインデックスとするエントリーのリンクインデックスには、メモリアドレス４０をインデックスとするエントリーがリンクされる。すなわち、プログラムの命令の実行順に伴ってアクセスされるメモリアドレスの順番がリンクリストキャッシュテーブル９に記憶される。
【００９１】
したがって、次回以降、このプログラムが実行されるとき、メモリアドレス２０をメモリアドレス３０へのアクセス時にプリフェッチする。これによって、次段階で必要となるメモリアドレス２０のデータをキャッシュ４に用意することができる。よって、メモリ１

にＣＰＵ３はアクセスする必要はなく、キャッシュ４にアクセスすればよく、ＣＰＵ３は、必要なデータを高速に取得できる。また、メモリアドレス２０へのアクセス時に、メモリアドレス４０をプリフェッチする。これによって、次の段階で必要となるメモリアドレス４０のデータをキャッシュ４に用意することができる。よって、メモリ１にＣＰＵ３はアクセスする必要はなく、キャッシュ４にアクセスすればよい。そして、ＣＰＵ３は、必要となるデータを高速に取得できる。

【００９２】

次に、リンクリスト式のプログラムの２回目の実行（プログラムの第２実行時）について説明する。なお、ループごとの実行結果については、図１０を参照するものとする。また、２回目のループ実行時の状況は、初回のループサイクルからＣサイクル隔てているものとする。

【００９３】

Ｐ＝１０のループ（サイクル２～８間のループ）では、リストセル１の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス１０である。

【００９４】

ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。

【００９５】

また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス１０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。

【００９６】

そして、ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス１０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【００９７】

また、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、ＰＣ７をインデックスとするエントリーがある。そして、ディスクリプタコントローラ７は、現在のサイクル７＋Ｃから、そのエントリーの最終ＰＣ到達サイクル２８を減算する。減算して得たカウントサイクル－２１＋Ｃと、そのエントリーにおける再ＰＣ到達サイクル０とは異なることを確率的に認識する。なお、上述のようにＣサイクル隔てているため、現在のサイクルは７＋Ｃに設定されている。

【００９８】

そこで、ディスクリプタコントローラ７は、そのエントリーに次のように動作する。メモリアドレスを１０に設定する。また、再ＰＣ到達サイクルに、－２１＋Ｃを設定する。また、バススコアテーブル８の最終ＰＣ到達サイクルに７＋Ｃを設定する。

【００９９】

Ｐ＝３０のループ（サイクル９～１５間のループ）では、リストセル２の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス３０である。

【０１００】

ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。そして、ディスクリプタコントローラ７は、このエントリーのリンクインデックスにリンクされたメモリアドレス２０をインデックスとするエントリーを取得する。

【０１０１】

また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス２０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。

【０１０２】

そして、ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス３０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１０３】

また、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、ＰＣ７をインデックスとするエントリーがある。そして、ディスクリプタコントローラ７は、現在のサイクル１４＋Ｃからそのエントリーの最終ＰＣ到達サイクルの７＋Ｃを減算する。減算して得た値（カウントサイクル値）７と、そのエントリーにおける再ＰＣ到達サイクル－２１＋Ｃは異なることを確率的に認識する。

【０１０４】

そこで、ディスクリプタコントローラ７は、そのエントリーに次のように動作する。メモリアドレスを３０に設定する。また、再ＰＣ到達サイクルに７（１４＋Ｃから７＋Ｃを減算した値）を設定する。また、バススコアテーブル８の最終ＰＣ到達サイクルに１４＋Ｃを設定する。

【０１０５】

Ｐ＝２０のループ（サイクル１６～２２間のループ）では、リストセル３の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス２０である。

【０１０６】

ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。そして、ディスクリプタコントローラ７は、このエントリーのリンクインデックスにリンクされたメモリアドレス４０をインデックスとするエントリーを取得する。

【０１０７】

また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス４０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。

【０１０８】

そして、ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス２０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１０９】

また、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、ＰＣ７をインデックスとするエントリーがある。そして、ディスクリプタコントローラ７は、現在のサイクル２１＋Ｃからそのエントリーの最終ＰＣ到達サイクルの１４＋Ｃを減算する。減算して得た値（カウントサイクル値）７と、そのエントリーにおける再ＰＣ到達サイクル７とが同じ値であることを認識する。

【０１１０】

そこで、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス３０に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。すなわち、バススコアテーブル８は、メモリアドレス３０をインデックスとして、リンクリストキャッシュテーブル９の

エントリーを探す。そして、その探したメモリアドレス３０をインデックスとするエントリーと、現在のメモリアドレス２０をインデックスとするエントリーとをリンクさせる。具体的には、メモリアドレス３０をインデックスとするエントリーのリンクインデックスに、メモリアドレス２０をインデックスとするエントリーのインデックスを設定する。このようにして、メモリアドレス３０をインデックスとするエントリーにおけるリンクインデックスに設定されたインデックス値が３０から２０に変更される。

【０１１１】

また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを２０に設定する。また、バススコアテーブル８の最終ＰＣ到達サイクルに２１＋Ｃを設定する。

【０１１２】

Ｐ＝４０のループ（サイクル２３～２９間のループ）では、リストセル４の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス４０である。

【０１１３】

ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。そして、ディスクリプタコントローラ７は、このエントリーのリンクインデックスにリンクされたメモリアドレス４０をインデックスとするエントリーを取得する（つまり、自身のエントリーを取得する）。

【０１１４】

また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス４０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。

【０１１５】

そして、ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス４０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１１６】

また、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、ＰＣ７をインデックスとするエントリーがある。そして、ディスクリプタコントローラ７は、現在のサイクル２８＋Ｃからそのエントリーの最終ＰＣ到達サイクルの２１＋Ｃを減算する。減算して得た値（カウントサイクル値）７と、そのエントリーにおける再ＰＣ到達サイクルの値が７と同じ値であることを認識する。

【０１１７】

そこで、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス２０に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。すなわち、バススコアテーブル８は、メモリアドレス２０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、その探したメモリアドレス２０をインデックスとするエントリーと、現在のメモリアドレス４０をインデックスとするエントリーとをリンクさせる。具体的には、メモリアドレス２０をインデックスとするエントリーのリンクインデックスに、メモリアドレス４０をインデックスとするエントリーのインデックスを設定する。このようにして、メモリアドレス２０をインデックスとするエントリーにおけるリンクインデックスに設定されたインデックス値が２０から４０に変更される。

【０１１８】

また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを４０に設定する。また、バススコアテーブル８の最終ＰＣ到達サイクルに２８＋Ｃを設定す

る。
【０１１９】
このようにして、図１０に示すように、キャッシュコントロールディスクリプタテーブル１０には、適切なタイミングでプリフェッチを行う指示が設定される。そして、キャッシュコントローラ２は、キャッシュコントロールディスクリプタテーブル１０を参照して、適切なタイミングでプリフェッチする。
【０１２０】
〔第２の実施の形態〕
第１の実施の形態とは、ディスクリプタコントローラ７からバススコアテーブル８に対する動作が主に異なる。また、バススコアテーブル８のエントリーの内容も異なる。また、メモリ１の各メモリアドレスには、次にアクセスされるメモリアドレスを示すデータも格納されている。なお、本実施形態では、プロファイラ５は、第１の実施の形態のように、命令サイクルカウンタを有する必要はない。つまり、上記した図３におけるＳ２のステップは、本実施形態においては省略可能である。
【０１２１】
バススコアテーブル８は、第１の実施の形態と同様に、ＰＣをインデックスとするエントリーを有する。本実施形態におけるエントリーは、ステータス情報を有する。ステータス情報は、（３ａ）エントリーにデータが無い状態、（３ｂ）エントリーにデータが在って、後述のロードアドレスオフセットが無しの状態、（３ｃ）エントリーにデータが在って、後述のロードアドレスオフセットが在りの状態、の３つのステータスを示す。また、プリフェッチ開始ＰＣ、ロードアドレスオフセット、最終ロード値、を含む。
【０１２２】
なお、本実施形態では、メモリ１の各メモリアドレスには、次にアクセスされるメモリアドレスを示すデータが格納されている。そして、次にアクセスされるメモリアドレスを示すデータは、メモリアドレスから一定のロードアドレスオフセットを加算した位置からロードされる。よって、上述の最終ロード値は、次にアクセスされるメモリアドレスに相当する。ロードアドレスオフセットは、今回のメモリアクセス時のメモリアドレスから前回のメモリアクセス時に設定された最終ロード値を減算して得た値である。したがって、ループにおいて分岐処理が行われていたとしても、第１の実施の形態と同様に、好適にプリフェッチすることができる。
【０１２３】
ディスクリプタコントローラ７は、プロファイラ５から与えられたＰＣをインデックスとして、バススコアテーブル８からエントリーを得る。ディスクリプタコントローラ７は、エントリーの状態により、次のような動作をする。
【０１２４】
３ａ：エントリーにデータが無い場合。
ディスクリプタコントローラ７は、エントリーにメモリアドレスを設定する。また、最終ロード値に、最後にロードした値を設定する。エントリーのステータスを、エントリーにデータ在りでロードレスアドレスオフセット無しに設定する。
【０１２５】
３ｂ：エントリーにデータが在って、ロードアドレスオフセットがなしの場合。
ディスクリプタコントローラ７は、プロファイラ５から与えられたメモリアドレスからエントリーの最終ロード値を減算する。そして、この減算値を、そのエントリーのロードアドレスオフセットに設定する。最終ロード値に、最後にロードした値を設定する。メモリアドレスに、プロファイラ５から与えられたメモリアドレスを設定する。エントリーのステータスをエントリーにデータ在りで、ロードアドレスオフセット在りに設定する。
【０１２６】
３ｃ：エントリーにデータが在って、ロードアドレスオフセットがありの場合。
ディスクリプタコントローラ７は、プロファイラ５から与えられたメモリアドレスからエントリーの最終ロード値を減算する。そして、減算して得た値とロードアドレスオフセッ

トを比較する。ディスクリプタコントローラ７は、比較した結果に基づいて、次のように動作する。

【０１２７】

３ｃ－１：減算値とロードアドレスオフセットの値が異なる場合。

ディスクリプタコントローラ７は、３ｂ：エントリーにデータが在って、かつロードアドレスオフセットがなしの場合と同じ動作をする。

【０１２８】

３ｃ－２：減算値とロードアドレスオフセットの値とが同じ場合。

ディスクリプタコントローラ７は、バススコアテーブル８のエントリーにあるメモリもアドレスに基づいて、そのメモリアドレスをインデックスとして、リンクリストキャッシュテーブル９からエントリーを検索する。そして、取得したリンクリストキャッシュテーブル９のエントリーが持つリンクインデックスに、プロファイラ５から与えられたメモリアドレスをインデックスとするエントリーを指定するように設定する。これによって、リンクリストキャッシュテーブル９には、プログラムの実行に伴ってアクセスされるメモリアドレスの順番が記憶される。その後に、最終ロード値に、最後にロードした値を設定する。

【０１２９】

本実施形態におけるディスクリプタコントローラ７の動作を、図１１のフローチャートを用いて説明する。

【０１３０】

図１１に示すように、まず、ディスクリプタコントローラ７は、第１の実施の形態と同様に、キャッシュコントロールディスクリプタテーブル１０に対する処理を行う（Ｓ１）。次に、ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを取得する（Ｓ２）。次に、取得したエントリーにデータが無いか確認する（Ｓ３）。エントリーにデータが無い場合、そのエントリーを上述の３ａのように設定する（Ｓ４）。そして、開始点Ｃに戻る。取得したエントリーにデータがある場合には、上述の３ｂの場合であるか確認する（Ｓ５）。そして、上述の３ｂの場合であれば、上述の３ｂに記載したようにエントリーの設定を行う（Ｓ６）。上述の３ｂの場合でなければ、上述の３ｃ－１の場合であるのか判断する（Ｓ７）。上述の３ｃ－１の場合であれば、上述のＳ６を実行する。上述の３ｃ－１の場合でなければ、上述の３Ｃ－２のように処理を実行する（Ｓ８）。そして、開始点Ｃに戻る。Ｓ７において、減算値とロードアドレスオフセットの値とが同じであれば、実行されるループに分岐があったとしても問題は生じない。したがって、２－２－Ｃ－ｂの処理に基づいて、アクセスされるメモリアドレスの順番の記憶は問題なく行える。

【０１３１】

ここで、以下、再び図８に示されたプリフェッチの実行トレース表を用いて、本実施形態におけるプリフェッチ方法について説明する。なお、実行されるプログラムとしては、図１６に示された配置サンプルであることを前提とする。また、プロファイラ５は、ループごとに５行目のＰＣをプロファイルデータキュー６にプリフェッチ可能ＰＣとして連絡するものとする。

【０１３２】

まず、リンクリスト式のプログラムの１回目の実行（プログラムの第１実行時）について説明する。なお、ループごとの実行結果については、図１２を参照するものとする。

【０１３３】

Ｐ＝１０のループ（サイクル２～８間のループ）では、リストセル１の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス１０である。

【０１３４】

ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在し

ない。そこで、サイクル７のとき、ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス１０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１３５】

また、ディスクリプタコントローラ７は、バススコアテーブル８に格納されたエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル７のとき、ディスクリプタコントローラ７は、バススコアテーブル８にエントリーを登録する。そのエントリーに、インデックスとして、ＰＣ７を設定する。また、そのエントリーに、データが在るとする。また、そのエントリーのメモリアドレスに、メモリアドレス１０を設定する。また、ステータス情報を、エントリーにデータありでロードアドレスオフセット無しとする。また、最終ロード値に、最後にロードした値３０を記録させる。

【０１３６】

Ｐ＝３０のループ（サイクル９～１５間のループ）では、リストセル２の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス３０である。

【０１３７】

ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル１４のとき、ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス３０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１３８】

ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のように、サイクル７で登録したエントリーがある。ディスクリプタコントローラ７は、そのエントリーのステータス情報を確認する。そして、ステータス情報が、エントリーにデータがありでロードアドレスオフセット無しであることを認識する。そして、エントリーに次のように設定する。メモリアドレスに、プロファイラ５から与えられたメモリアドレス３０を設定する。エントリーのステータス情報に、エントリーにデータありでロードアドレスオフセットありを設定する。ロードアドレスオフセットに、現在のプロファイラ５から与えられたメモリアドレス３０から、エントリーに在る最終ロード値を減算した値０を設定する。また、最終ロード値に、最後にロードした値２０を設定する。

【０１３９】

Ｐ＝２０のループ（サイクル１６～２２間のループ）では、リストセル３の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス２０である。

【０１４０】

ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル２１のとき、ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス２０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１４１】

ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のように、サイクル１４で設定したエントリーがある。ディス

クリプタコントローラ７は、そのエントリーのステータス情報を確認する。そして、ステータス情報が、エントリーにデータがありでロードアドレスオフセット在りであることを認識する。そして、プロファイラ５から与えられたメモリアドレス２０から、そのエントリーにある最終ロード値２０を減算する。減算して得た値０と、そのエントリーに在るロードアドレスオフセット０が同じ値であることを確認する。

【０１４２】

そして、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス３０に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。すなわち、バススコアテーブル８は、メモリアドレス３０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、その探したメモリアドレス３０をインデックスとするエントリーと、現在のメモリアドレス２０をインデックスとするエントリーとをリンクさせる。具体的には、メモリアドレス３０をインデックスとするエントリーのリンクインデックスに、メモリアドレス２０をインデックスとするエントリーのインデックスを設定する。そして、図１２に示すように、メモリアドレス３０をインデックスとするエントリーにおけるリンクインデックスに設定されたインデックス値が３０から２０に変更される。

【０１４３】

また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを２０に設定する。また、エントリーのステータス情報に、エントリーにデータありでロードアドレスオフセットありを設定する。また、最終ロード値に、最後にロードした値４０を設定する。

【０１４４】

Ｐ＝４０のループ（サイクル２３～２９間のループ）では、リストセル４の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス４０である。

【０１４５】

ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。しかし、該当するエントリーは存在しない。そこで、サイクル２８のとき、ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとするエントリーを登録する。また、そのエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス４０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１４６】

ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のように、サイクル２１で設定したエントリーがある。ディスクリプタコントローラ７は、そのエントリーのステータス情報を確認する。そして、ステータス情報が、エントリーにデータがありでロードアドレスオフセット在りであることを認識する。そして、プロファイラ５から与えられたメモリアドレス４０から、そのエントリーにある最終ロード値４０を減算する。減算して得た値０と、そのエントリーに在るロードアドレスオフセット０が同じ値であることを確認する。

【０１４７】

そして、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス２０に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。すなわち、バススコアテーブル８は、メモリアドレス２０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、その探したメモリアドレス２０をインデックスとするエントリーと、現在のメモリアドレス４０をインデックスとするエントリーとをリンクさせる。具体的には、メモリアドレス２０をインデックスとするエントリーのリンクインデックスに、メモリアドレス４０をインデックスとするエントリーのインデックスを設定する。そ

して、図１２に示すように、メモリアドレス２０をインデックスとするエントリーにおけるリンクインデックスに設定されたインデックス値が２０から４０に変更される。

【０１４８】

また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを４０に設定する。また、エントリーのステータス情報に、エントリーにデータありでロードアドレスオフセットありを設定する。また、最終ロード値に、最後にロードした値０を設定する。

【０１４９】

このようにして、本実施形態の場合も、リンクリスト式のプログラムが１回実行されることにより、メモリアドレス３０をインデックスとするエントリーのリンクインデックスには、メモリアドレス２０をインデックスとするエントリーがリンクされる。実行されるプログラムにおいては、メモリアドレス３０にアクセスされた後に、メモリアドレス２０にアクセスされる。したがって、次回以降、このプログラムが実行され、メモリアドレス３０にアクセスされたとき、メモリアドレス２０にあるデータはキャッシュ４にフェッチされる。これによって、次の段階で必要となるメモリアドレス２０のデータをキャッシュ４に用意することができる。よって、ＣＰＵ３は、メモリ１にアクセスする必要はなく、キャッシュ４にアクセスして必要なデータを取得できる。すなわち、プリフェッチによりキャッシュヒットを得ることができる。

【０１５０】

また、メモリアドレス２０をインデックスとするエントリーのリンクインデックスには、メモリアドレス４０をインデックスとするエントリーがリンクされる。したがって、次回以降、このプログラムが実行され、メモリアドレス２０にメモリアクセスされたとき、メモリアドレス４０にあるデータはキャッシュ４にフェッチされる。これによって、次の段階で必要となるメモリアドレス４０のデータをキャッシュ４に用意することができる。すなわち、プリフェッチによりキャッシュヒットを得ることができる。よって、ＣＰＵ３は、メモリ１にアクセスする必要はなく、キャッシュ４にアクセスして、必要となるデータを取得することができる。

【０１５１】

次に、このリンクリスト式のプログラムの２回目の実行（プログラムの第２実行時）について説明する。なお、ループごとの実行結果については、図１３を参照するものとする。

【０１５２】

Ｐ＝１０のループ（サイクル２～８間のループ）では、リストセル１の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス１０である。

【０１５３】

ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。そして、ディスクリプタコントローラ７は、このエントリーのメモリアドレス１０をキャッシュコントロールディスクリプタテーブル１０に連絡する。

【０１５４】

また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス１０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。

【０１５５】

そして、ディスクリプタコントローラ７は、メモリアドレス１０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス１０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１５６】
ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のように、ＰＣ７をインデックスとするエントリーがある。ディスクリプタコントローラ７は、そのエントリーのステータス情報を確認する。そして、ステータス情報が、エントリーにデータがありでロードアドレスオフセット在りであることを認識する。そして、プロファイラ５から与えられたメモリアドレス１０から、そのエントリーにある最終ロード値０を減算する。減算して得た値１０と、そのエントリーに在るロードアドレスオフセット０が異なる値であることを確認する。
【０１５７】
そして、ディスクリプタコントローラ７は、そのエントリーに次のことを設定する。メモリアドレスに１０を設定する。エントリーのステータス情報を、エントリーにデータ在りでロードアドレスオフセット在りを設定する。ロードアドレスオフセットに、プロファイラ５から与えられたアドレス１０から、そのエントリーの最終ロード値を減算した値１０を設定する。最終ロード値に、最後にロードした値３０を設定する。
【０１５８】
Ｐ＝３０のループ（サイクル９～１５間のループ）では、リストセル２の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス３０である。
【０１５９】
ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。そして、ディスクリプタコントローラ７は、このエントリーのリンクインデックスにリンクされたメモリアドレス２０をインデックスとするエントリーを取得する。
【０１６０】
また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス２０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。
【０１６１】
そして、ディスクリプタコントローラ７は、メモリアドレス３０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス３０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。
【０１６２】
ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のように、ＰＣ７をインデックスとするエントリーがある。ディスクリプタコントローラ７は、そのエントリーのステータス情報を確認する。そして、ステータス情報が、エントリーにデータがありでロードアドレスオフセット在りであることを認識する。そして、プロファイラ５から与えられたメモリアドレス３０から、そのエントリーにある最終ロード値３０を減算する。減算して得た値０と、そのエントリーに在るロードアドレスオフセット１０が異なる値であることを確認する。
【０１６３】
そして、ディスクリプタコントローラ７は、そのエントリーに次のことを設定する。メモリアドレスに３０を設定する。エントリーのステータス情報を、エントリーにデータ在りでロードアドレスオフセット在りを設定する。ロードアドレスオフセットに、プロファイラ５から与えられたメモリアドレス３０から、そのエントリーの最終ロード値３０を減算した値０を設定する。最終ロード値に、最後にロードした値２０を設定する。
【０１６４】
Ｐ＝２０のループ（サイクル１６～２２間のループ）では、リストセル３の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス２

０である。
【０１６５】
ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。そして、ディスクリプタコントローラ７は、このエントリーのリンクインデックスにリンクされたメモリアドレス４０をインデックスとするエントリーを取得する。
【０１６６】
また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス４０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。
【０１６７】
そして、ディスクリプタコントローラ７は、メモリアドレス２０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス２０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。
【０１６８】
ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のように、ＰＣ７をインデックスとするエントリーがある。ディスクリプタコントローラ７は、そのエントリーのステータス情報を確認する。そして、ステータス情報が、エントリーにデータがありでロードアドレスオフセット在りであることを認識する。そして、プロファイラ５から与えられたメモリアドレス２０から、そのエントリーにある最終ロード値２０を減算する。減算して得た値０と、そのエントリーに在るロードアドレスオフセット０が同じ値であることを確認する。
【０１６９】
そして、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス３０に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。すなわち、バススコアテーブル８は、メモリアドレス３０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、その探したメモリアドレス３０をインデックスとするエントリーと、現在のメモリアドレス２０をインデックスとするエントリーとをリンクさせる。具体的には、メモリアドレス３０をインデックスとするエントリーのリンクインデックスに、メモリアドレス２０をインデックスとするエントリーのインデックスを設定する。このようにして、図１２に示すように、メモリアドレス３０をインデックスとするエントリーにおけるリンクインデックスに設定されたインデックス値が３０から２０に変更される。
【０１７０】
また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを２０に設定する。また、エントリーのステータス情報に、エントリーにデータありでロードアドレスオフセットありを設定する。また、最終ロード値に、最後にロードした値４０を設定する。
【０１７１】
Ｐ＝４０のループ（サイクル２３～２９間のループ）では、リストセル４の実行が行われる。このとき、ＣＰＵ３からアクセスされるメモリ１のアドレスは、メモリアドレス４０である。
【０１７２】
ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、該当するエントリーを取得する。そして、ディスクリプタコントローラ７は、このエントリーのリンクインデックスにリンクされたメモリアドレス４０をインデックスとするエントリーを取得する（つまり、

自身のエントリーを取得する）。

【０１７３】

また、このとき、ディスクリプタコントローラ７は、プロファイルデータキュー６からプリフェッチの開始点となるＰＣ５を取得する。そして、ディスクリプタコントローラ７は、ＰＣ５とメモリアドレス４０とをキャッシュコントロールディスクリプタテーブル１０に連絡する。

【０１７４】

そして、ディスクリプタコントローラ７は、メモリアドレス４０をインデックスとするエントリーにデータが在るとする。また、そのエントリーのメモリアドレスは、メモリアドレス４０とする。また、そのエントリーのリンクインデックスには、自身のエントリーのインデックスを設定する。

【０１７５】

ディスクリプタコントローラ７は、バススコアテーブル８のエントリーを探す。バススコアテーブル８には、上述のように、ＰＣ７をインデックスとするエントリーがある。ディスクリプタコントローラ７は、そのエントリーのステータス情報を確認する。そして、ステータス情報が、エントリーにデータがありでロードアドレスオフセット在りであることを認識する。そして、プロファイラ５から与えられたメモリアドレス４０から、そのエントリーにある最終ロード値４０を減算する。減算して得た値０と、そのエントリーに在るロードアドレスオフセット０が同じ値であることを確認する。

【０１７６】

そして、ディスクリプタコントローラ７は、次のように動作する。バススコアテーブル８のエントリーに在るメモリアドレス２０に基づいて、ディスクリプタコントローラ７はリンクリストキャッシュテーブル９のエントリーを探す。すなわち、バススコアテーブル８は、メモリアドレス２０をインデックスとして、リンクリストキャッシュテーブル９のエントリーを探す。そして、その探したメモリアドレス２０をインデックスとするエントリーと、現在のメモリアドレス４０をインデックスとするエントリーとをリンクさせる。具体的には、メモリアドレス２０をインデックスとするエントリーのリンクインデックスに、メモリアドレス４０をインデックスとするエントリーのインデックスを設定する。このようにして、図１２に示すように、メモリアドレス２０をインデックスとするエントリーにおけるリンクインデックスに設定されたインデックス値が４０に変更される。

【０１７７】

また、ディスクリプタコントローラ７は、バススコアテーブル８のメモリアドレスを２０に設定する。また、エントリーのステータス情報に、エントリーにデータありでロードアドレスオフセットありを設定する。また、最終ロード値に、最後にロードした値０を設定する。

【０１７８】

図１３に示すように、キャッシュコントロールディスクリプタテーブル１０には、適切なタイミングでプリフェッチを行う指示が設定されることが分かる。そして、キャッシュコントローラ２は、キャッシュコントロールディスクリプタテーブル１０を参照して、適切なタイミングでプリフェッチする。

【０１７９】

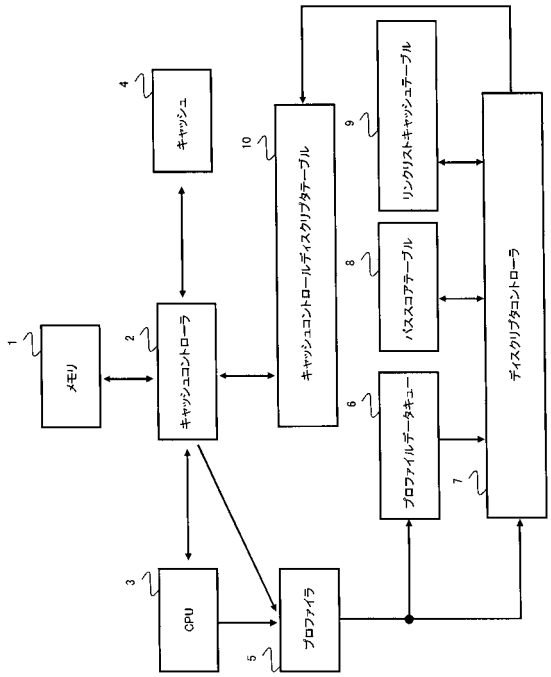
本発明の技術的範囲は、上述の実施の形態に限られない。リンクリストキャッシュテーブルのエントリー同士をリンクさせる具体的な方法は任意である。エントリーのリンクリストインデックスではなく、エントリーのメモリアドレスを利用して、各エントリーをリンクさせることもできる。

【０１８０】

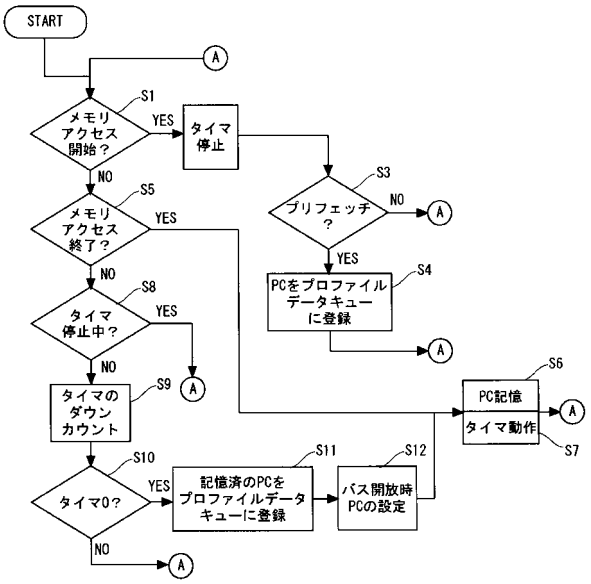
プロファイラは、ＣＰＵ３からメモリ１へのストア命令、リード命令のいずれかを検出しても良いし、双方を検出しても良い。プリフェッチ可能ＰＣとされるプロファイルキューに登録されたＰＣは、直近に登録されたＰＣである必要はない。プロファイルキュー自体は、プログラムの分岐命令等により適宜リフレッシュされる。

【図面の簡単な説明】
【０１８１】
【図１】第１の実施の形態におけるキャッシュ機構のブロック図である。
【図２】プロファイラ５の動作を示すフローチャートである。
【図３】プロファイラ５の動作を示すフローチャートである。
【図４】バススコアテーブル８、リンクリストキャッシュテーブル９の模式図である。
【図５】ディスクリプタコントローラ７の動作を示すフローチャートである。
【図６】図５のＳ１での処理を示すフローチャートである。
【図７】キャッシュコントローラ２の動作を示すフローチャートである。
【図８】プリフェッチの実行トレースを示す表図である。
【図９】プログラムの第１実行時におけるループごとの実行結果を示す表図である。
【図１０】プログラムの第２実行時におけるループごとの実行結果を示す表図である。
【図１１】第２の実施の形態におけるディスクリプタコントローラ７の動作を示すフローチャートである。
【図１２】第２の実施の形態におけるプログラムの第１実行時におけるループごとの実行結果を示す表図である。
【図１３】第２の実施の形態におけるプログラムの第２実行時におけるループごとの実行結果を示す表図である。
【図１４】従来のプリフェッチ機構を示す。
【図１５】配列式のプログラムが実行される場合の説明図である。
【図１６】リンクリスト式のプログラムのリストセルの配置サンプルである。
【図１７】ストライドによるプリフェッチにおいて、リンクリスト式のプログラムが実行される場合の説明図である。
【符号の説明】
【０１８２】
１　メモリ
２　キャッシュコントローラ
３　ＣＰＵ
４　キュッシュ
５　プロファイラ
６　プロファイルデータキュー
７　ディスクリプタコントローラ
８　バススコアテーブル
９　リンクリストキャッシュテーブル
１０　キャッシュコントロールディスクリプタテーブル

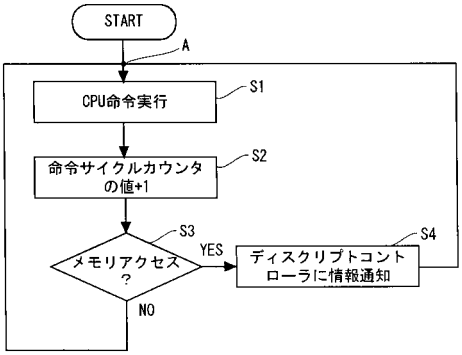
【図１】


メモリ
キャッシュコントローラ
CPU
プロファイラ
キャッシュ
キャッシュコントロールディスクリプタテーブル
リンクリストキャッシュテーブル
バススコアテーブル
プロファイルデータキュー
ディスクリプタコントローラ

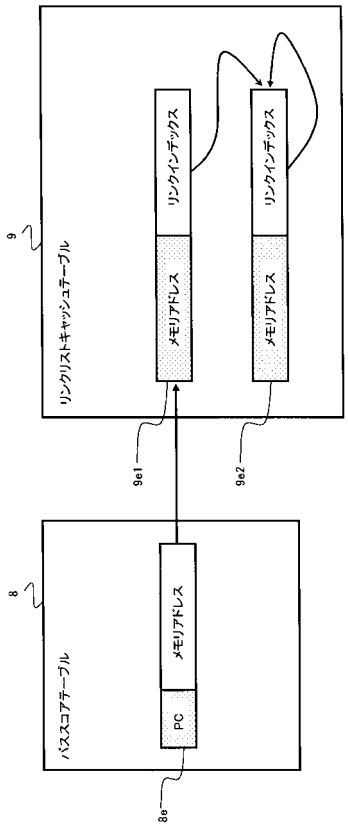
【図２】


START
メモリアクセス開始？
YES
NO
タイマ停止
メモリアクセス終了？
プリフェッチ？
PCをプロファイルデータキューに登録
タイマ停止中？
タイマのダウンカウント
タイマ0？
記憶済のPCをプロファイルデータキューに登録
バス開放時PCの設定
PC記憶
タイマ動作

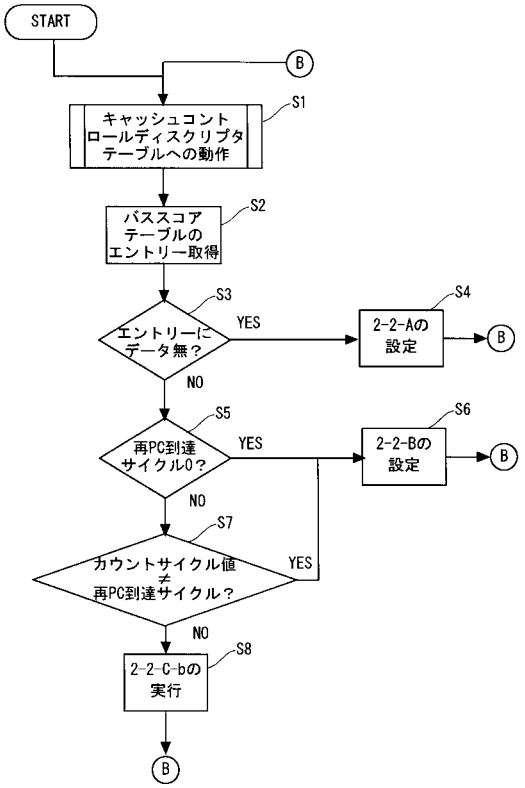
【図３】


START
CPU命令実行
命令サイクルカウンタの値+1
メモリアクセス？
YES
NO
ディスクリプトコントローラに情報通知

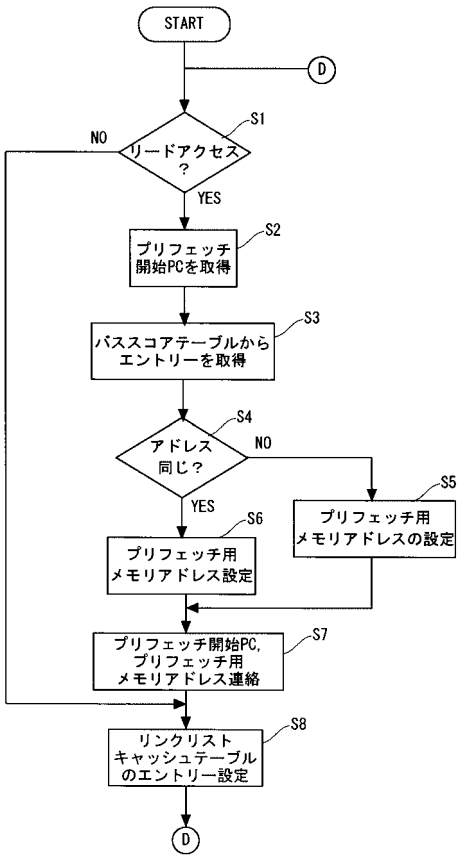
【図４】


リンクリストキャッシュテーブル
リンクインデックス
メモリアドレス
リンクインデックス
メモリアドレス
バススコアテーブル
メモリアドレス
PC

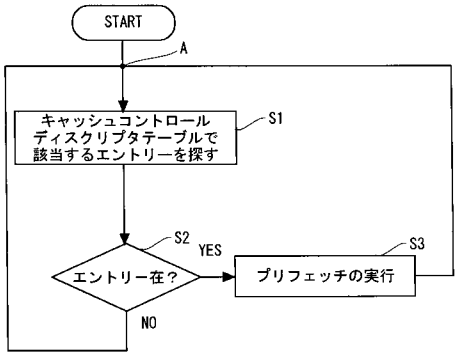
【図５】



【図６】



【図７】



【図８】

サイクル	PC	実行される命令	プログラムの第2実行時の説明
1	1	p=10;	
p=10のループ			
2	2	while(p)	
3	3	{	
4	4	<メモリアクセスのある処理>	
5	5	<メモリアクセスのない処理>	
6	6	a+=p->value;/*メモリアクセスのある処理*/	
7	7	p=p->next; /*メモリアクセスのある処理*/	PC=5でアドレス10にプリフェッチすると登録
8	8	}	
p=30のループ			
9	2	while(p)	
10	3	{	
11	4	<メモリアクセスのある処理>	
12	5	<メモリアクセスのない処理>	
13	6	a+=p->value;/*メモリアクセスのある処理*/	
14	7	p=p->next; /*メモリアクセスのある処理*/	PC=5でアドレス20にプリフェッチすると登録
15	8	}	
p=20のループ			
16	2	while(p)	
17	3	{	
18	4	<メモリアクセスのある処理>	
19	5	<メモリアクセスのない処理>	アドレス20をプリフェッチ
20	6	a+=p->value;/*メモリアクセスのある処理*/	ヒット
21	7	p=p->next; /*メモリアクセスのある処理*/	PC=5でアドレス40にプリフェッチすると登録
22	8	}	
p=40のループ			
23	2	while(p)	
24	3	{	
25	4	<メモリアクセスのある処理>	
26	5	<メモリアクセスのない処理>	アドレス40をプリフェッチ
27	6	a+=p->value;/*メモリアクセスのある処理*/	ヒット
28	7	p=p->next; /*メモリアクセスのある処理*/	
29	8	}	

【図９】

サイクル	データの在／無	有効／無効	メモリアドレス	プリフェッチ開始PC	消PC到達サイクル	最終PC到達サイクル	<リンクリストキャッシュテーブルのエントリーのリンクインデックス> メモリアドレス10	メモリアドレス20	メモリアドレス30	メモリアドレス40
初期状態	無	無効	0	無効	0	0	0			
7 (p=10)	在	有効	10	5	0	7	10			
14 (p=30)	在	有効	30	5	7	14	10		30	
21 (p=20)	在	有効	20	5	7	21	10	20	20	
28 (p=40)	在	有効	40	5	7	28	10	40	20	40

【図１０】

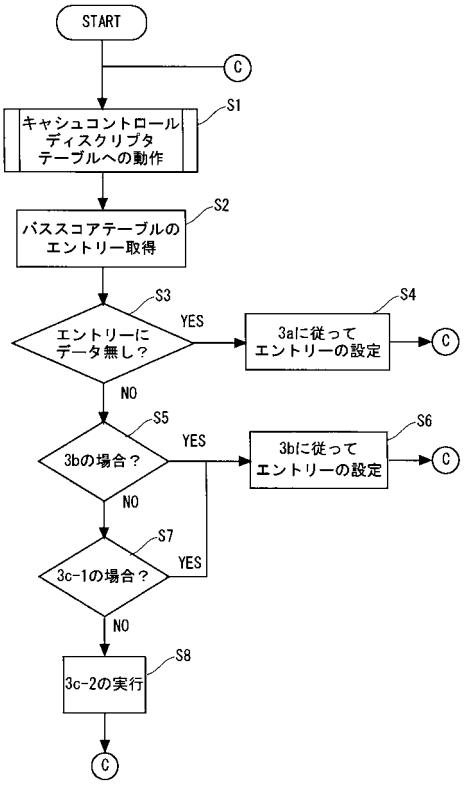
<バススコアテーブル>

サイクル	データの在／無	有効／無効	メモリアドレス	プリフェッチ開始PC	消PC到達サイクル	最終PC到達サイクル	<リンクリストキャッシュテーブルのエントリーのリンクインデックス> メモリアドレス10	メモリアドレス20	メモリアドレス30	メモリアドレス40
7+C (p=10)	在	有効	10	5	-21+C	7+C	10	40	20	40
14+C (p=30)	在	有効	30	5	7	14+C	10	40	30	40
21+C (p=20)	在	有効	20	5	7	21+C	10	20	20	40
28+C (p=40)	在	有効	40	5	7	28+C	10	40	20	40

<キャッシュコントロールディスクリプタテーブル>

サイクル	メモリアドレス10	メモリアドレス20	メモリアドレス30	メモリアドレス40
7+C (p=10)	PC=5			
14+C (p=30)		PC=5		
21+C (p=20)			PC=5	
28+C (p=40)				

【図１１】

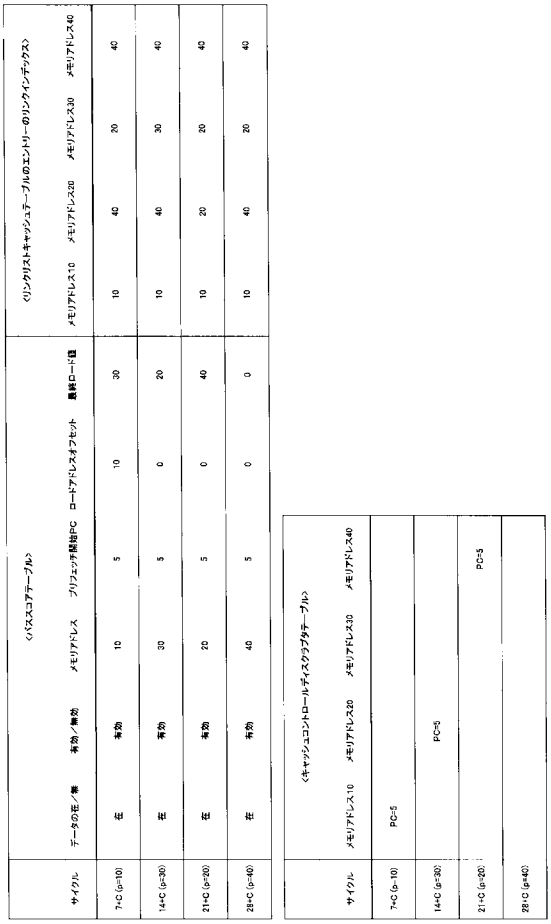


【図１２】

<バススコアテーブル>

サイクル	データの在／無	有効／無効	メモリアドレス	プリフェッチ開始PC	ロードアドレスオフセット	最終ロード値	<リンクリストキャッシュテーブルのエントリーのリンクインデックス> メモリアドレス10	メモリアドレス20	メモリアドレス30	メモリアドレス40
初期状態	無	無効	0	無効	不定	0	0			
7 (p=10)	在	有効	10	5	不定	30	10			
14 (p=30)	在	有効	30	5	0	20	10		30	
21 (p=20)	在	有効	20	5	0	40	10	20	20	
28 (p=40)	在	有効	40	5	0	0	10	40	20	40

【図１３】



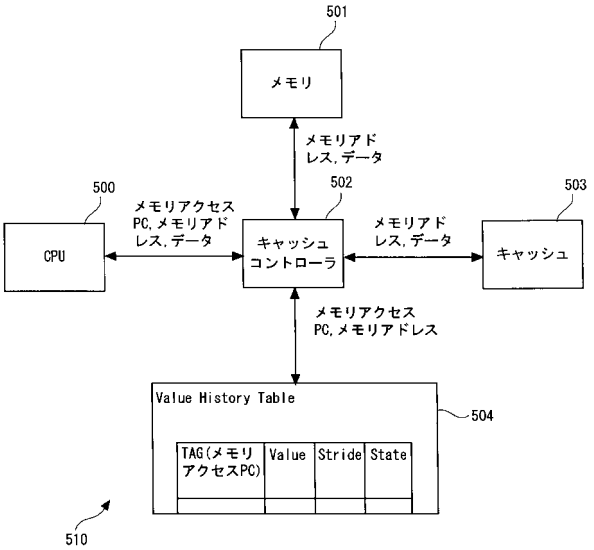
〈パスコアテーブル〉

	サイクル	データの生／無	有効／無効	メモリアドレス	プリフェッチ開始PC	ロードアドレスオフセット	最終ロード値	〈リンクリストキャッシュテーブルのエントリーのリンクインデックス〉 メモリアドレス10	メモリアドレス20	メモリアドレス30	メモリアドレス40
1	7+C (p=10)	生	有効	10	5	10	30	10	40	20	40
2	14+C (p=30)	生	有効	30	5	0	20	10	40	30	40
3	21+C (p=20)	生	有効	20	5	0	40	10	20	20	40
4	28+C (p=40)	生	有効	40	5	0	0	10	40	20	40

〈キャッシュコントロールディスクリプタテーブル〉

	サイクル	メモリアドレス10	メモリアドレス20	メモリアドレス30	メモリアドレス40
1	7+C (p=10)	PC=5			
2	14+C (p=30)		PC=5		
3	21+C (p=20)			PC=5	
4	28+C (p=40)				

【図１４】



【図１５】

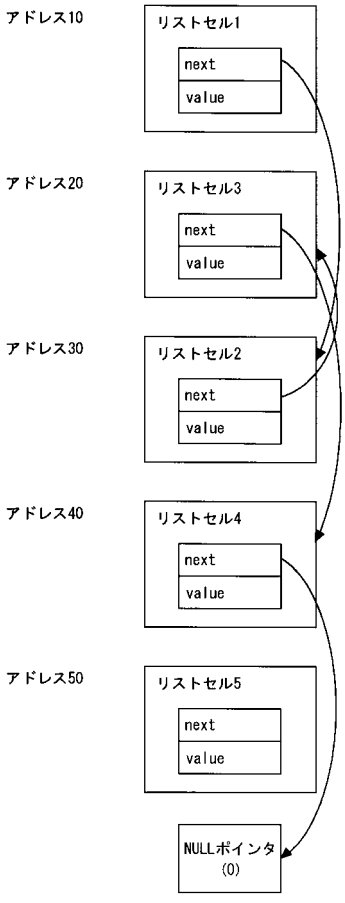
〈メモリアクセス〉

STEP	PC	命令	アドレス10	アドレス20	アドレス30
1	1	i=10			
2	2	while(10<N)			
3	3	{			
4	4	a+=mem[10];	ミス		
5	5	i+=10;			
6	6	}			
7	2	while(20<N)			
8	3	{			
9	4	a+=mem[20];		ミス	プリフェッチ
10	5	i+=10;			
11	6	}			
12	2	while(30<N)			
13	3	{			
14	4	a+=mem[30];			ヒット
15	5	i+=10;			
16	6	}			

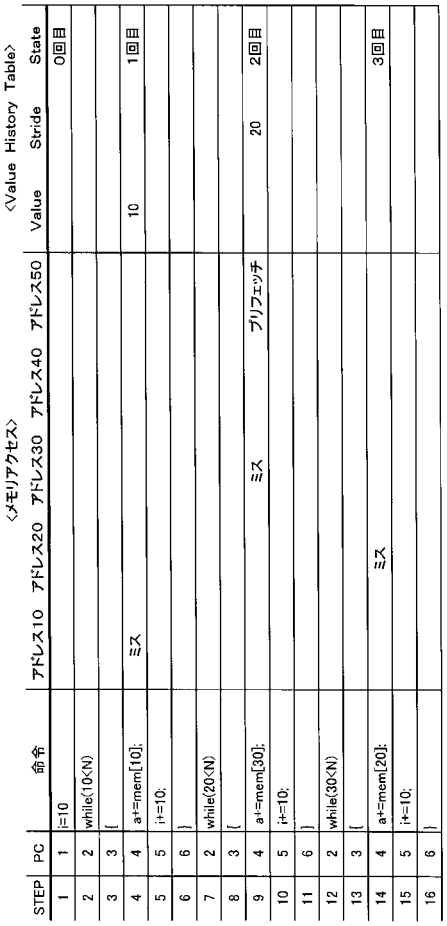
〈Value History Table〉

State	Stride	Value
0回目		
1回目		10
2回目	10	
3回目		

【図１６】



【図１７】



〈メモリアクセス〉

STEP	PC	命令	アドレス10	アドレス20	アドレス30	アドレス40	アドレス50
1	1	i=10					
2	2	while(10<N)					
3	3	{					
4	4	a+=mem[10];	ミス				
5	5	i+=10;					
6	6	}					
7	2	while(20<N)					
8	3	{					
9	4	a+=mem[30];			ミス	プリフェッチ	
10	5	i+=10;					
11	6	}					
12	2	while(30<N)					
13	3	{					
14	4	a+=mem[20];		ミス			
15	5	i+=10;					
16	6	}					

〈Value History Table〉

Value	Stride	State
		0回目
10		1回目
	20	2回目
		3回目

フロントページの続き

(56)参考文献 特開２００１－１８８７０６（ＪＰ，Ａ）
特開２００１－０６９４４０（ＪＰ，Ａ）
特開２００６－２６００６７（ＪＰ，Ａ）
特開平１１－１６７５２０（ＪＰ，Ａ）
特表２００６－５１００８２（ＪＰ，Ａ）
特開平０８－１９４６１５（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１２／０８－１２／１２