(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第5627330号
(P5627330)

(45)発行日　平成26年11月19日(2014.11.19)　(24)登録日　平成26年10月10日(2014.10.10)

(51)Int.Cl.　　F I
G06F 9/38 (2006.01)　　G06F 9/38 310X
G06F 12/08 (2006.01)　　G06F 12/08 551Z
G06F 12/08 507Z
G06F 12/08 573
G06F 12/08 559D

請求項の数 15　(全 33 頁)

(21)出願番号　特願2010-173961 (P2010-173961)
(22)出願日　平成22年8月2日 (2010.8.2)
(65)公開番号　特開2012-33112 (P2012-33112A)
(43)公開日　平成24年2月16日 (2012.2.16)
審査請求日　平成25年6月25日 (2013.6.25)

(73)特許権者　000001007
キヤノン株式会社
東京都大田区下丸子３丁目３０番２号
(74)代理人　100076428
弁理士　大塚　康徳
(74)代理人　100112508
弁理士　高柳　司郎
(74)代理人　100115071
弁理士　大塚　康弘
(74)代理人　100116894
弁理士　木村　秀二
(74)代理人　100130409
弁理士　下山　治
(74)代理人　100134175
弁理士　永川　行光

最終頁に続く

(54)【発明の名称】情報処理装置、キャッシュ装置およびデータ処理方法

(57)【特許請求の範囲】
【請求項１】
複数の第１ノードを有し、当該第１ノードの保持する第１データを第１方向に位置する隣のノードへ移動させる第１パイプラインと、
前記第１パイプラインの第１ノードの各々に対応する複数の第２ノードを有し、当該第２ノードの保持する第２データを前記第１方向と逆の第２方向に位置する隣のノードへ移動させる第２パイプラインと、
前記第１データと前記第２データとを用いてデータ処理を行う処理手段と、
前記第２パイプラインの出力した第２データの属性情報に基づき、前記出力した第２データの何れかを保持する保持手段と、
を備え、
前記第２データの属性情報に基づき、前記保持手段で保持した第２データを前記第２パイプラインに入力することを特徴とする情報処理装置。
【請求項２】
前記保持手段では、前記第２パイプラインからの破棄と復帰を決定するための属性情報に基づいて、保持している第２データの優先順位が決定され、
前記優先順位に基づいて、前記保持手段が保持する第２データと前記第２パイプラインが保持する第２データとが入れ替えられることを特徴とする請求項１に記載の情報処理装置。
【請求項３】

前記処理手段のデータ処理の結果に基づき、前記属性情報を前記保持手段で算出することを特徴とする請求項１又は２に記載の情報処理装置。
【請求項４】
前記処理手段のデータ処理の結果に基づき、前記属性情報を前記第２パイプラインで算出することを特徴とする請求項１又は２に記載の情報処理装置。
【請求項５】
前記処理手段のデータ処理は、前記第１パイプラインの第１データと、前記第２パイプラインの第２データとが一致するか否かを判定する処理であり、前記属性情報は、一致したときの回数であることを特徴とする請求項１から４のいずれか１項に記載の情報処理装置。
【請求項６】
情報処理装置であって、
複数の第１ノードを有し、当該第１ノードの保持する第１データを第１方向に位置する隣のノードへ移動させる第１パイプラインと、
前記第１パイプラインの第１ノードの各々に対応する複数の第２ノードを有し、当該第２ノードの保持する第２データを前記第１方向と逆の第２方向に位置する隣のノードへ移動させる第２パイプラインと、
前記第１データと前記第２データとを用いてデータ処理を行う処理手段と、
前記第２パイプラインの出力した第２データの属性情報に基づき、前記出力した第２データの何れかを保持する保持手段と、
を備え、
前記属性情報は、前記情報処理装置の外部から設定されることを特徴とする情報処理装置。
【請求項７】
前記保持手段が複数の第２データを保持するときは、前記保持した第２データのうち、前記属性情報が最小の第２データを低い優先順位と定め、該第２データを破棄して新たに入力される第２データと交換することを特徴とする請求項１から６のいずれか１項に記載の情報処理装置。
【請求項８】
前記保持手段が複数の第２データを保持するときは、前記保持した第２データのうち、前記属性情報が最大の第２データを高い優先順位と定め、第２パイプラインへ入力することを特徴とする請求項１から６のいずれか１項に記載の情報処理装置。
【請求項９】
複数の第１ノードを有し、当該第１ノードの保持する第１データを第１方向に位置する隣のノードへ移動させる第１パイプラインと、
前記第１パイプラインの第１ノードの各々に対応する複数の第２ノードを有し、当該第２ノードの保持する第２データを前記第１方向と逆の第２方向に位置する隣のノードへ移動させる第２パイプラインと、
前記第１データと前記第２データとを用いてデータ処理を行う処理手段と、
前記第２パイプラインの出力した第２データの属性情報に基づき、前記出力した第２データの何れかを保持する保持手段と、
を備え、
前記属性情報は優先順位を示す情報を含むことを特徴とする情報処理装置。
【請求項１０】
対象データのキャッシュを管理するキャッシュ装置であって、
複数の第１ノードを有し、当該第１ノードの保持する第１データを第１方向に位置する隣のノードへ移動させる第１パイプラインと、
前記第１パイプラインの第１ノードの各々に対応する複数の第２ノードを有し、当該第２ノードの保持する第２データを前記第１方向と逆の第２方向に位置する隣のノードへ移動させる第２パイプラインと、

前記第１データと前記第２データとを用いてデータ処理を行う処理手段と、
前記第２パイプラインの出力した第２データの属性情報に基づき、前記出力した第２データの何れかを保持する保持手段と、
を有する情報処理装置を備え、
前記第１パイプラインで前記対象データのアドレスを移動し、前記第２パイプラインでキャッシュタグを移動することにより、前記アドレスと前記キャッシュタグとが一致するか否かにより、キャッシュヒットを判定することを特徴とするキャッシュ装置。
【請求項１１】
前記キャッシュタグは、前記第２パイプラインと前記保持手段に分割して保持されることを特徴とする、請求項１０に記載のキャッシュ装置。
【請求項１２】
対象データのキャッシュを管理するキャッシュ装置であって、
複数の第１ノードを有し、当該第１ノードの保持する第１データを第１方向に位置する隣のノードへ移動させる第１パイプラインと、
前記第１パイプラインの第１ノードの各々に対応する複数の第２ノードを有し、当該第２ノードの保持する第２データを前記第１方向と逆の第２方向に位置する隣のノードへ移動させる第２パイプラインと、
前記第１データと前記第２データとを用いてデータ処理を行う処理手段と、
前記第２パイプラインの出力した第２データの属性情報に基づき、前記出力した第２データの何れかを保持する保持手段と、
を有する情報処理装置を備え、
前記第１パイプラインで前記対象データのアドレスを移動し、前記第２パイプラインでキャッシュタグを移動することにより、前記保持手段が、前記アドレスと前記キャッシュタグとが一致しない場合に、前記出力した第２データを保持し、一致した場合には、前記保持した第２データを前記第２パイプラインに入力することを特徴とするキャッシュ装置。
【請求項１３】
前記キャッシュ装置は、該キャッシュ装置の一部に予め必要とするデータに対して格納アドレスを識別して格納し、
前記保持手段が、前記第１パイプラインの前記識別された格納アドレスと前記キャッシュタグとが一致しない場合にも、前記第２パイプラインから前記第２データを一時的に退避させて、前記第２パイプラインに入力することを特徴とする請求項１０に記載のキャッシュ装置。
【請求項１４】
データ処理装置において実行されるデータ処理方法であって、
複数の第１ノードを備える第１パイプラインが、第１ノードの保持する第１データを第１方向に位置する隣のノードへ移動させる工程と、
前記第１パイプラインの第１ノードの各々に対応する複数の第２ノードを備える第２パイプラインが、第２ノードの保持する第２データを前記第１方向と逆の第２方向に位置する隣のノードへ移動させる工程と、
処理手段が、前記第１データと前記第２データとを用いてデータ処理を行う工程と、
保持手段が、前記第２パイプラインの出力した第２データの属性情報に基づき、前記出力した第２データの何れかを保持する工程と、
を有し、
前記第２データの属性情報に基づき、前記保持手段で保持した第２データを前記第２パイプラインに入力することを特徴とするデータ処理方法。
【請求項１５】
請求項１４に記載のデータ処理方法の各工程をコンピュータに実行させるためのプログラム。
【発明の詳細な説明】

【技術分野】
【０００１】
本発明は、複数のノードを有する２つのデータ・ストリーム間で相互に接続されたノードのデータ要素を処理する情報処理技術に関する。
【背景技術】
【０００２】
複数のノードを有する２つのデータ・ストリームにおいて、それぞれのデータ・ストリームの各ノードで保持するデータ要素の各々を比較するアプリケーションがある。例えば、複数のデータを有する第１データ・ストリームの要素について、第２データ・ストリームの要素の少なくとも１つと一致するかどうかを判定する場合について、総当りで比較する処理がある。このようなアプリケーションでは、ある方向にデータ要素を移動させる第１データ・ストリームと、ある方向と逆方向にデータ要素を移動させる第２データ・ストリームの各々のデータ要素を比較する。しかしながら後述するように、相互に逆の方向にデータ要素を移動させる２つのデータ・ストリームにおいて、各ノードにおけるデータ要素の比較が正しく動作しないケースが起きる。
【０００３】
特許文献１には、相互に逆の方向にデータ要素を移動させる２種類のパイプライン・データストリームにおいて、パイプラインの各ステージで各々のデータ要素を比較する２方向パイプライン技術（カウンタフロー・パイプライン技術）が記載されている。この特許文献１では、各ノードにおけるデータ要素の比較が正しく動作しないケースに対し１つの解決策を開示している。
【０００４】
ここで、図１３を用いてこのアプリケーションのデータ要素の比較が正しく動作しないケースについて説明する。図１３には、互いに逆方向にデータ要素を移動させる２つのパイプライン回路の一部が記載されている。下側の第１パイプラインは、「上手」である図面上の左側から「下手」である図面上の右側に向かってデータ要素が移動する。一方、上側の第２パイプラインは「上手」である図面上の右側から「下手」である図面上の左側に向かってデータ要素が移動する。
【０００５】
図１３（ａ）は、データ要素の比較が正しく行われ、前述の問題が発生しないケースである。第１パイプラインが動作することでデータ要素が移動し、第２パイプラインが停止することでデータ要素が移動しないケースを示している。図１３（ａ－１）は、時刻［Ｔ］の状態を示し、図１３（ａ－２）は、時刻［Ｔ］からある一定時間後の時刻［Ｔ＋１］の状態を示し、図１３（ａ－３）は、時刻［Ｔ＋１］からさらに一定時間後の時刻［Ｔ＋２］の状態を示す。
【０００６】
いま、第１パイプラインが動作し、「上手」である図面上の左側から「下手」である図面上の右側にパイプライン・ステージに保持されているデータ要素である、Ｗ、Ａ（０）、Ａ（１）、Ａ（２）、Ｂ，Ｃが移動している。ここでＡ（０）、Ａ（１）、Ａ（２）は、説明上区別するために括弧の０，１，２で表記しているが、他のデータと同等のデータと考えてよい。第１パイプラインと第２パイプラインとが対応する各ステージは、比較して同じか否かを判定する判定ステージ９０１～９０４により互いに接続されている。
【０００７】
図１３（ａ－１）の時刻［Ｔ］のとき、各ステージの判定結果は、第１パイプラインの「下手」から順に以下のようになる。まず、第１パイプラインの一番下手の判定ステージ９０１では、データ要素ＷとＡの比較となって一致しないので、＜偽＞と判定する。以降の判定ステージ９０２～９０４についても、データ要素Ａ（０）とＺと、データ要素Ａ（１）とＹと、データ要素Ａ（２）とＸと、を各々比較して、一致しないので＜偽＞と判定する。
【０００８】

次に時間が経過し、図１３（ａ－２）の時刻［Ｔ＋１］のとき、第１パイプラインのデータ要素は、１ステージだけ「下手」に移動する。各判定ステージ９０１～９０４の判定結果は、第１パイプラインの「下手」から順に、以下のようになる。まず、判定ステージ９０１で、データ要素Ａ（０）とＡとの比較となり、＜真＞と判定する。以降の判定ステージ９０２～９０４では、データ要素Ａ（１）とＺと、データ要素Ａ（２）とＹと、データ要素ＢとＸと、を各々比較して、＜偽＞と判定する。

【０００９】

さらに時間が経過し、図１３（ａ－３）の時刻［Ｔ＋２］のとき、第１パイプラインのデータ要素は、さらに１ステージだけ「下手」に移動する。各判定ステージ９０１～９０４の判定結果は、第１パイプラインの「下手」から順に以下のようになる。まず、判定ステージ９０１でのデータ要素Ａ（１）とＡの比較となり、＜真＞と判定する。以降の判定ステージ９０２～９０４では、データ要素Ａ（２）とＺと、データ要素ＢとＹと、データ要素ＣとＸと、を各々比較して、＜偽＞と判定する。

【００１０】

以上のように、第１パイプラインのデータ要素は時間の経過とともにステージを移動するが、第２パイプラインの「上手」にあるデータ要素Ａとの比較は、第１パイプラインのデータ要素Ａ（０）、Ａ（１）共に正しく比較される。このように第１パイプラインと第２パイプラインで一方のみが動作し、一方が停止しているケースでは、データ要素の比較は、正しく行われる。

【００１１】

次に、第１パイプラインと第２パイプラインの両方が動作するケースを図１３（ｂ）に示す。図１３（ｂ－１）～（ｂ－３）は、図１３（ａ－１）～（ａ－３）と同様の時刻における状態を示している。第１パイプラインは、図１３（ａ）と同様に動作するので説明を省略する。一方、第２パイプラインは、図１３（ａ）とは異なる。すなわち、第２パイプラインでは、「上手」である図面上の右側から「下手」である図面上の左側にパイプライン・ステージに保持されているデータ要素Ｘ、Ｙ、Ｚ、Ａ、Ｂ、Ｃも移動する。以下、図１３（ａ）と同様に、第１パイプラインと第２パイプラインの対応する各ステージにおける判定ステージの判定結果を説明する。

【００１２】

図１３（ｂ－１）の時刻［Ｔ］のとき、各ステージの判定結果は第１パイプラインの「下手」から順に以下のようになる。まず、第１パイプラインの一番下手の判定ステージ９０１は、データ要素ＷとＡとを比較し、一致しないので＜偽＞と判定する。以降の判定ステージ９０２～９０４でも、判定（比較）回路は、データ要素Ａ（０）とＺと、データ要素Ａ（１）とＹと、データ要素Ａ（２）とＸと、を各々比較し、一致しないので＜偽＞と判定する。

【００１３】

次に時間が経過し、図１３（ｂ－２）の時刻［Ｔ＋１］のとき、第１パイプラインのデータ要素と第２パイプラインのデータ要素は、１ステージだけ各々の下手に移動する。各判定ステージ９０１～９０４の判定結果は、第１パイプラインの「下手」から順に以下のようになる。まず、判定ステージ９０１は、データ要素Ａ（０）とＢとを比較し、一致しないので＜偽＞と判定する。次の判定ステージ９０２では、データ要素Ａ（１）とＡとを比較し、一致するので＜真＞と判定する。以降のステージ９０３、９０４では、データ要素Ａ（２）とＺと、データ要素ＢとＹと、を各々比較して、一致しないので＜偽＞と判定する。

【００１４】

さらに時間が経過し、図１３（ｂ－３）の時刻［Ｔ＋２］のとき、第１パイプラインのデータ要素と第２パイプラインのデータ要素は、１ステージだけ各々の「下手」に移動する。各判定ステージ９０１～９０４の判定結果は、第１パイプラインの「下手」から順に以下のようになる。まず判定ステージ９０１は、データ要素Ａ（１）とＣと比較し、一致しないので＜偽＞と判定する（ただし、Ａ（１）は、図１３（ｂ－２）のときに判定ステ

ージ９０２で＜真＞と判定済みである）。以降の判定ステージ９０２～９０４で、データ要素Ａ（２）とＢと、データ要素ＢとＡと、データ要素ＣとＺと、を各々比較し、一致しないので＜偽＞と判定する。

【００１５】

以上のように、第１パイプラインと第２パイプラインが同時に移動する場合、第２パイプラインの「上手」にあるデータ要素Ａと第１パイプラインのデータ要素Ａ（１）は、比較された。しかし、データ要素Ａ（０）とＡ（２）は第２パイプラインのデータ要素Ａと比較されることがなかった。例えば、第１パイプラインの各々のデータ要素が、第２パイプラインのデータ要素の各々の少なくとも１つと一致するとき、図１３（ｂ－１）～（ｂ－ｃ）の処理では正しい判定ができていないことがわかる。これは、第１パイプラインと第２パイプラインの両方が互いに逆方向に移動するため、両パイプラインの相対移動速度が２倍になることに由来する。

【００１６】

実際、図１３（ａ）のケースでは、時刻［Ｔ］～時刻［Ｔ＋２］の間に、第１パイプラインのデータ要素、Ｗ、Ａ（０）、Ａ（１）の比較が完了するだけである。これに対し、図１３（ｂ）のケースでは第１パイプラインのデータ要素、Ｗ、Ａ（０）、Ａ（１）、Ａ（２）、Ｂの比較が完了している。このように、図１３（ｂ）のケースは、図１３（ａ）のケースと比較して判定時間が短くなる代わりに、データ要素の比較漏れが発生してしまうのである。

【００１７】

特許文献１に記載の技術は、データ要素が移動する際に通過する特定のステージおよびデータ要素に対し、実行予定の動作に応じてデータ要素を不規則なスケジュールで移動させることで、この問題を解決している。具体的には、第１パイプラインと第２パイプラインの各ステージにおいて、ステージ状態を監視する。そしてこの問題が起こる特定ステージにおいて、２方向のパイプラインのデータ要素の移動を停止し、比較が完了してから、その特定ステージのデータ要素の移動を許可するのである。この工夫により、比較の完了前に２方向のパイプライン上で関連するデータ要素が、判定されずに通り過ぎることを防いでいる。

【００１８】

しかしながら、このような構成では、問題が起こりうる特定ステージにおいて、停止、比較の完了、およびデータ要素の移動という動作を伴うため、その結果、各ステージ状態に合わせてデータ要素が不規則なスケジュールで移動と停止を繰り返してしまう。この不規則なスケジュールの繰り返しにより、データ処理のためのパイプラインは乱れ、パイプライン構成を採用する利点であるスループットの向上が妨げられてしまう。

【００１９】

一方、上記のようなデータ処理技術を用いた場合の最良の応用例として、非常に多くのデータ要素をパイプライン構成で高速にデータ処理するような用途が考えられる。例えば、このようなデータ処理を行う分野として画像処理分野やグラフィックス処理分野がある。近年、このような分野では製品の高画質化や機能強化のために種々な画像処理が行われており、複数コンテンツ（画像、映像）を時分割多重で同時並行的に画像処理したり、半透明のテクスチャデータを複数枚、重ね合わせて表示したりしている。このような画像処理では、ＤＲＡＭなどの外部メモリから適宜必要な画像や映像を取り出して処理し、外部メモリからのデータ取得を、キャッシュ装置を介して行うことが一般的である。

【００２０】

しかしながら、時分割多重の画像処理や半透明テクスチャを複数枚重ね合わせると、様々な画像や映像を同時にしかも並行して処理することになり、１つの対象データである画像や映像データは、細切れに外部メモリから取得されことになる。一般的なキャッシュ技術の実装で、様々な対象データである画像や映像データを細切れに外部メモリから取得すると、キャッシュ競合が頻発しキャッシュ性能を著しく低下させてしまう。このキャッシュ競合を回避するため、キャッシュタグのウェイ数を増やすことができるが、一般的なキ

ャッシュ技術の実装で多数のウェイが存在する場合、判定ステージでのセレクタの論理段数が多くなり、各ステージでのタイミング収束が難しくなる。このため、高い動作周波数で動作することが難しくなるという課題がある。また、同時にウェイ数を増やすことにより、キャッシュ競合時の置き換え制御も複雑となり、キャッシュ技術の実装も難しくなる。

【先行技術文献】
【特許文献】
【００２１】
【特許文献１】特許第３５８８４８７号公報
【発明の概要】
【発明が解決しようとする課題】
【００２２】
従って本発明は、パイプラインを用いて処理を行う情報処理装置において、繰り返し必要とされるデータが情報処理装置から破棄されないように、置き換え制御を工夫しておこなうことで、より効率的な技術を提供することを目的とする。
【課題を解決するための手段】
【００２３】
係る課題を解決するため、情報処理装置は、複数の第１ノードを有し、当該第１ノードの保持する第１データを第１方向に位置する隣のノードへ移動させる第１パイプラインと、前記第１パイプラインの第１ノードの各々に対応する複数の第２ノードを有し、当該第２ノードの保持する第２データを前記第１方向と逆の第２方向に位置する隣のノードへ移動させる第２パイプラインと、前記第１データと前記第２データとを用いてデータ処理を行う処理手段と、前記第２パイプラインの出力した第２データの属性情報に基づき、前記出力した第２データの何れかを保持する保持手段と、を備え、前記第２データの属性情報に基づき、前記保持手段で保持した第２データを前記第２パイプラインに入力することを特徴とする。
【発明の効果】
【００２４】
本発明によれば、パイプラインを用いて処理を行う情報処理装置において、繰り返し必要とされるデータが情報処理装置から破棄されないように、置き換え制御を工夫しておこなうことで、より効率的な技術を提供することが可能となる。
【図面の簡単な説明】
【００２５】
【図１】データ処理装置の基本構成の一例を示すブロック図。
【図２】データ処理装置を応用したキャッシュタグ判定部の一例を示すブロック図。
【図３】データ処理装置のリオーダ回路の一例を示すブロック図。
【図４】データ処理装置をリオーダ回路の処理の一例を示すフローチャート。
【図５】データ処理装置を適用した画像処理装置の一例を示すブロック図。
【図６】データ処理装置を適用したキャッシュ装置のキャッシュメモリ調停部の処理フローの一例を示す図。
【図７】データ処理装置のリオーダ回路の一例を示すブロック図。
【図８】データ処理装置をリオーダ回路の処理の一例を示すフローチャート。
【図９】データ処理装置を適用したキャッシュ装置のキャッシュメモリ調停部の処理フローの一例を示す図。
【図１０】データ処理装置に属性情報を組み込んだ基本構成の一例を示すブロック図。
【図１１】データ処理装置に属性情報を組み込んだ基本構成の一例を示すブロック図。
【図１２】データ処理装置に複数のリオーダバッファを備えるリオーダ回路の一例を示すブロック図。
【図１３】従来技術の動作例を説明する図。
【発明を実施するための形態】

【００２６】
以下、添付の図面を参照して本発明の好適な実施形態を説明する。
【００２７】
＜実施形態１＞
（基本構成）
図１は、実施形態１による情報処理装置（以降、データ処理装置と称す）の基本構成の一例を示すブロック図である。図に示すように、各パイプラインは、データの保持手段としての複数のノードを有している。第１パイプライン１００は、この場合、所定数の８つの第１ノード（第１保持手段）を有し、この中の１つを着目ノードとすると、着目ノードから第１の方向へ１つ下手の第１ノードへデータ要素（第１データ）を所定の周期で移動させる。このときの移動方向を「第１移動方向」、移動する行為を「第１移動工程」とする。また、第１パイプライン１００には、外部から、データ要素の有効信号である「ｖａｌｉｄ［ｌ－１］」１１２と、データ信号である「ｄａｔａ［ｌ－１］」１１４と、データ信号の処理結果である処理結果信号「ｔａｇ＿ｉｄ［ｌ－１］」１１６とが入力される。この一連の入力信号は、種々のデータ処理を行うデータ処理回路１２０で処理された後、パイプライン・レジスタ１１０にて一時的に記憶される。
【００２８】
このパイプライン・レジスタ１１０は、駆動信号である「ｅｎａｂｌｅ」１０２が有効（アサート状態）であるとき、データ処理後のデータ要素を一時的に記憶して、以前に記憶したデータ要素を更新する。ただし、「ｅｎａｂｌｅ」１０２が無効（ディアサート状態）であるときは、このパイプライン・レジスタ１１０は、データ処理後のデータ要素を記憶せずに、以前に記憶したデータ要素をそのまま保持するため、データ要素の更新は行わない。入力からパイプライン・レジスタ１１０に一時的に記憶されるまでの区間をステージ（第１ステージ）と呼ぶ。
【００２９】
次にパイプライン・レジスタ１１０から、データ要素の有効信号「ｖａｌｉｄ［ｌ］」１３２と、データ信号「ｄａｔａ［ｌ］」１３４と、処理結果信号「ｔａｇ＿ｉｄ［ｌ］」１３６とが出力される。そして、第１ステージと同様にデータ処理回路１４０で種々のデータ処理が行われる。更に、処理後のデータ要素が、第２パイプライン・レジスタ１３０にて一時的に記憶されるが、この記憶動作は第１ステージと同様である。パイプライン・レジスタ１１０からのデータ要素がパイプライン・レジスタ１３０に一時的に記憶されるまでの区間を第２ステージと呼ぶ。
【００３０】
さらにパイプライン・レジスタ１３０からデータ要素の有効信号「ｖａｌｉｄ［ｌ＋１］」１５２と、データ信号「ｄａｔａ［ｌ＋１］」１５４と処理結果信号「ｔａｇ＿ｉｄ［ｌ＋１］」１５６とが出力される。このような動作により、第１パイプライン１００では、データ要素（有効信号「ｖａｌｉｄ」とデータ信号「ｄａｔａ」と処理結果信号「ｔａｇ＿ｉｄ」）が「上手」である図面上の左側から「下手」である図面上の右側に移動する。
【００３１】
一方、本実施形態では、第１パイプライン以外に第２パイプラインが存在する。この第２パイプラインは、第１パイプラインの８つの第１ノードに対応付けられた第２ノード（第２保持手段）を有し、この第２ノードの保持するデータ要素（第２データ）を第１パイプラインの移動させる第１方向と逆方向（第２方向）にある第２ノードへ移動させる。詳細には、この第２パイプライン１６０は「上手」である図面上の右側から「下手」である図面上の左側に向けてデータ要素を移動させる。このときの移動方向を「第２移動方向」、移動する行為を「第２移動工程」とする。
【００３２】
第２パイプライン１６０には、データ要素の有効信号である「ｔａｇ＿ｖａｌｉｄ[ｉ＋２]」１７２と、データ信号である「ｔａｇ＿ｄａｔａ[ｉ＋２]」１７４が外部から入

力される。これらの入力信号は、前述のデータ処理回路１４０で使用される。その後、パイプライン・レジスタ１７０にて一時的に記憶される。

【００３３】

このパイプライン・レジスタ１７０は、駆動信号である「ｓｈｉｆｔ」１６２が有効であるとき、一連の入力信号である有効信号「ｔａｇ＿ｖａｌｉｄ[ｉ＋２]」１７２と、データ信号「ｔａｇ＿ｄａｔａ[ｉ＋２]」１７４とを一時的に記憶する。そしてパイプライン・レジスタ１７０は、以前に記憶したデータ要素を更新する。

【００３４】

一方、駆動信号「ｓｈｉｆｔ」１６２が無効（ディアサート状態）であるときは、一連の入力信号を記憶せずに、以前に記憶したデータ要素をそのまま保持するために、データ要素を更新しない。入力からパイプライン・レジスタ１７０に一時的に記憶されるまでの区間をステージ（第１ステージ）と呼ぶ。

【００３５】

次にパイプライン・レジスタ１７０から、データ要素の有効信号「ｔａｇ＿ｖａｌｉｄ［ｉ＋１］」１８２と、データ信号「ｄａｔａ［ｉ＋１］」１８４とが出力され、第１ステージと同様にデータ処理回路１２０に入力される。そして、データ処理後に、第２パイプライン・レジスタ１８０にて一時的に記憶されるが、この記憶動作は第１ステージと同様である。パイプライン・レジスタ１７０からデータ要素がパイプライン・レジスタ１８０に一時的に記憶されるまでの区間を第２ステージと呼ぶ。

【００３６】

さらに、パイプライン・レジスタ１８０からデータ要素の有効信号「ｔａｇ＿ｖａｌｉｄ[ｉ]」１９２と、データ信号「ｔａｇ＿ｄａｔａ[ｉ]」１９４とが出力される。このような動作により、第２パイプライン１６０では、データ要素である、有効信号「ｔａｇ＿ｖａｌｉｄ」とデータ信号「ｔａｇ＿ｄａｔａ」とが、「上手」である図面上の右側から「下手」である図面上の左側に移動する。

【００３７】

（データ処理回路）

次にデータ処理回路１２０および１４０について説明する。本実施形態のデータ処理では、２方向のデータ要素の「ｄａｔａ」と「ｔａｇ＿ｄａｔａ」とを比較する。もし両者が一致し、等しいと判断すければ、そのときの「ｔａｇ＿ｄａｔａ」の格納番号（前述の"［ｉ］"、"［ｉ＋１］"、"［ｉ＋２］"）を「ｔａｇ＿ｉｄ」として記憶する。そして「ｔａｇ＿ｉｄ」は、第１パイプラインのデータ処理結果として、「ｄａｔａ」と同期して、「上手」である図面上の左側から「下手」である図面上の右側に移動する。このようにして、「ｔａｇ＿ｉｄ［ｌ］」１３６には、第１パイプライン１００のｌ番目のデータ要素「ｄａｔａ［ｌ］」１３４と値が等しい第２パイプライン１６０のデータ要素の格納番号が設定されることになる。

【００３８】

具体的には、まずデータ処理回路１２０は、有効信号「ｔａｇ＿ｖａｌｉｄ[ｉ]」１９２が有効のときに、データ信号「ｄａｔａ［ｌ－１］」１１４と「ｔａｇ＿ｄａｔａ［ｉ］」１９４とを比較回路１２２（第１比較回路）で比較する。そして比較結果が等しければ、セレクタ１２６は「ｔａｇ＿ｄａｔａ［ｉ］」１９４の格納番号である「Ｎｏｄｅ＝ｉ」を選択する。この選択された値は、第１パイプライン１００の「ｄａｔａ［ｌ－１］」１１４と値が等しい第２パイプライン１６０のデータ要素の格納番号として、「ｔａｇ＿ｉｄ［ｌ－１］」１１６に設定される。

【００３９】

従来例で述べたように、第１パイプライン１００と第２パイプライン１６０が同時に動作する場合、比較に失敗するケースがある。これに対処するためにデータ処理回路１２０は、さらに有効信号「ｔａｇ＿ｖａｌｉｄ[ｉ＋１]」１８２が有効のときに、データ信号「ｄａｔａ［ｌ－１］」１１４と「ｔａｇ＿ｄａｔａ［ｉ＋１］」１８４とを比較回路１２４（第２比較回路）で比較する。そして比較の結果が等しければ、セレクタ１２６は、

「ｔａｇ＿ｄａｔａ［ｉ＋１］」１８４の格納番号である「Ｎｏｄｅ＝ｉ＋１」を優先的に選択する。

【００４０】

またセレクタ１２６は、この比較の結果、両者が等しくないときは、入力された処理結果信号「ｔａｇ＿ｉｄ［ｌ－１］」１１６を選択する。外部からの駆動信号「ｓｈｉｆｔ」１６２が有効（アサート状態）であり、第２パイプライン１６０が動作する場合は、データ要素は「下手」である図面上の左側に移動する。したがって、この場合は「ｔａｇ＿ｄａｔａ」の格納番号も１つだけ左の格納番号を指すことが正しい。そこでセレクタ１２６の選択結果からデクリメンタ（減算器）１２８を用いて格納番号を１だけ減算することで調整する。

【００４１】

ここで格納番号の選択手法について補足説明しておく。格納番号は、「Ｎｏｄｅ＝ｉ」、「Ｎｏｄｅ＝ｉ＋１」、「ｔａｇ＿ｉｄ［ｌ－１］」１１６のいずれかが選択されるが、その選択基準は、例えば「数の大きいものを優先的に選択する」という単純なものでよい。例えば、データ信号「ｄａｔａ［ｌ－１］」１１４と「ｔａｇ＿ｄａｔａ［ｉ＋１］」１８４が等しいときに、外部からの駆動信号「ｓｈｉｆｔ」１６２が有効の場合を考える。この場合、前述の比較に失敗するケースの問題を回避するために「Ｎｏｄｅ＝ｉ＋１」を選択することが重要であるが、この動作は「数の大きいものを優先的に選択する」という手法に合致している。

【００４２】

一方、外部からの駆動信号「ｓｈｉｆｔ」１６２が無効の場合は、前述の比較に失敗するケースの問題を回避する必要がなく、「Ｎｏｄｅ＝ｉ＋１」を選択する必要はない。しかしながら「Ｎｏｄｅ＝ｉ＋１」をここで選択するかどうかに関わらず、第１パイプラインの下手ステージである第２ステージで、もう一度データ信号「ｄａｔａ［ｌ］」１３４と「ｔａｇ＿ｄａｔａ［ｉ＋１］」１８４との比較が改めて評価される。このため、外部からの駆動信号「ｓｈｉｆｔ」１６２が無効の場合の、第１ステージの「ｔａｇ＿ｄａｔａ［ｉ＋１］」１８４との比較は、どちらでもよい。逆に言えば、「数の大きいものを優先的に選択する」という手法で選択しても何ら差し支えがないことになる。このようにして選択された値は、「ｄａｔａ［ｌ－１］」１１４と値が等しい第２パイプライン１６０のデータ要素の格納番号を示す「ｔａｇ＿ｉｄ［ｌ－１］」１１６に設定されることになる。

【００４３】

図１は、あくまで一例であり、もちろんセレクタ１２６に駆動信号「ｓｈｉｆｔ」１６２を代入して、外部からの駆動信号「ｓｈｉｆｔ」１６２が無効の場合、「Ｎｏｄｅ＝ｉ＋１」は、選択しないという制御を行ってもよい。実施形態１の例では、比較回路１２２と比較回路１２４の両方ともが＜偽＞であったとき、入力データ信号「ｔａｇ＿ｉｄ［ｌ－１］」１１６を選択している。ここでは、外部からの駆動信号「ｓｈｉｆｔ」１６２が有効の場合、格納場所が「下手」に移動することに対処するため、格納番号を１だけ減算する調整をいずれにしても行う必要がある。このため、「Ｎｏｄｅ＝ｉ＋１」は選択しないという制御は行わず、「Ｎｏｄｅ＝ｉ＋１」を選択してから、格納番号を１だけ減算する調整を改めて行っている。

【００４４】

データ処理回路１４０についても同様である。まず、有効信号「ｔａｇ＿ｖａｌｉｄ[ｉ＋１]」１８２が有効のときに、データ信号「ｄａｔａ［ｌ］」１３４と「ｔａｇ＿ｄａｔａ［ｉ＋１］」１８４とを比較回路１４２で比較する。そして比較結果が等しければ、セレクタ１４６は、「ｔａｇ＿ｄａｔａ［ｉ＋１］」１８４の格納番号である「Ｎｏｄｅ＝ｉ＋１」を選択する。第２パイプライン１６０の動作に備え、データ処理回路１４０は、さらに有効信号「ｔａｇ＿ｖａｌｉｄ[ｉ＋２]」１７２が有効のときに、データ信号「ｄａｔａ［ｌ］」１３４と「ｔａｇ＿ｄａｔａ［ｉ＋２］」１７４とを比較回路１４４で比較する。そして比較結果が等しければ、セレクタ１４６は、「ｔａｇ＿ｄａｔａ［ｉ

＋２］」１７４の格納番号である「Ｎｏｄｅ＝ｉ＋２」を優先的に選択する。またセレクタ１４６は、上記２種類の比較結果がどちらも等しくないときは、処理結果信号「ｔａｇ＿ｉｄ［１］」１３６を選択する。

【００４５】

外部から駆動信号「ｓｈｉｆｔ」１６２が有効であり、第２パイプライン１６０が動作する場合、「ｔａｇ＿ｄａｔａ」の格納番号は１つだけ「下手」である図面上の右側に移動する。このため、セレクタ１４６の選択結果からデクリメンタ（減算器）１４８を用いて格納番号を１だけ減算する。このように検出結果を調整し、簡易な処理により正確で高速なデータ処理を実現することができる。

【００４６】

以上のように本実施形態では、互に逆方向に移動させるデータ要素をパイプラインの各ステージで確実かつ高速に比較することが可能である。本実施形態のデータ処理回路１２０および１４０の各々は、第１パイプライン１００の１つのデータ要素につき、第２パイプライン１６０の比較対象であるデータ要素との比較回路を持っている。さらに、データ処理回路１２０および１４０の各々は、第２パイプライン１６０が動作することを想定して第２パイプライン１６０の比較対象であるデータ要素に対し第２パイプライン１６０の「上手」のデータ要素との比較回路を新たに設けている。これにより、特許文献１の構成で生じる、特定ステージ毎にインターロック（内部ストール）が発生するという問題は回避できるので、常にデータ処理を停止することなく、高処理性能を実現することができる。

【００４７】

また、本実施形態では、２つのデータ要素が等しいときの格納場所を算出するために、データ処理回路１２０および１４０の各々は、第２パイプライン１６０が動作することを想定し、処理結果を１だけ減算するためのデクリメンタ（減算器）を備えている。そして、第２パイプラインのデータ要素の格納番号は、第２パイプラインの「下手」から「上手」に向けて「０、１、２、・・・、ｉ、ｉ＋１、ｉ＋２、・・・、Ｎ－１（ｉ、Ｎは正の数、ｉ＜Ｎ）」と増加するように予め割り振られている。このように割り振る利点については、後述する。また、格納番号が逆順に割り振られている場合は、デクリメンタは当然、１だけ加算するためのインクリメンタになることは言うまでもない。

【００４８】

また図２に、図１の基本構成をＮ個（Ｎ：整数）の組み合せて、Ｎ＝８の場合（８ステージ）のデータ処理装置の構成の一例を記載する。図２のデータ処理装置２００の例では、第１パイプラインと第２パイプラインの８個のデータ要素を比較している。図２において、ＴａｇＳｌｏｔ[０]～ＴａｇＳｌｏｔ[７]の各々は、図１に記載の第２パイプラインのパイプライン・レジスタ１７０、１８０に対応する。また、ＤａｔａＳｌｏｔ［０］～ＤａｔａＳｌｏｔ［７］の各々は、図１に記載の第１パイプラインのパイプライン・レジスタ１１０、１３０に対応する。さらに、Ｊｕｄｇｅ［０］～Ｊｕｄｇｅ［７］の各々は、図１に記載のデータ処理回路１２０、１４０に対応する。このように、基本構成を連結することで多くのデータ要素をパイプライン動作で並列に分散比較することが可能となる。

【００４９】

図２のデータ処理装置２００の構成例では、８個のデータ要素との比較を８ステージのパイプラインで実現している。ここでは、さらに外部からの駆動信号「ｓｈｉｆｔ」が有効であり、第２パイプラインが「下手」へ移動しても、処理性能を低下させることなく完全比較を実現できる。

【００５０】

そこで図２に示すように、データ処理装置２００に交換機能付きキャッシュ判定装置２８０を追加することで高度な画像処理用途に使用可能な、高速のフル・セット・アソシアティブ方式のキャッシュ装置を実現することができる。すなわち、この場合、使用されているメモリがＤＲＡＭとすると、キャッシュは、全てのＤＲＡＭのタグ情報を持っている

。以下、このキャッシュ装置を組み込んだ画像処理装置と、このキャッシュ装置について説明する。

【００５１】

（画像処理装置）

図５は、実施形態１による画像処理装置の全体構成例を示すブロック図である。本実施形態の画像処理装置は、図２で説明したデータ処理装置をフル・セット・アソシアティブ方式のキャッシュ判定部５２０として応用している。

【００５２】

画像処理装置には、ＣＰＵ５６０と外部メモリであるＤＲＡＭ５６５とＤＲＡＭコントローラ５６６とシステムバス５６４とが含まれる。また、ＤＲＡＭ５６５に対するデータの読み書きのために、データ読み出し用のＤＭＡＣ（Ｄｉｒｅｃｔ Ｍｅｍｏｒｙ Ａｃｃｅｓｓ Ｃｏｎｔｒｏｌｌｅｒ）であるＲＤＭＡＣ５６２とデータ書き込み用のＷＤＭＡＣ５６３も含まれる。また画像処理装置には、画像処理などの処理を行う処理装置５０１と、本発明のキャッシュ装置５１０が含まれる。なお、画像処理装置に含める処理装置の数は任意であり、処理装置の各々は高速に固定処理するパイプライン回路で構成されてもよいし、低速ではあるが、柔軟に処理内容を変更可能なプロセッサとプログラムで構成されてもよい。

【００５３】

ＣＰＵ５６０は、制御バス５６１を介して、ＲＤＭＡＣ５６２、ＷＤＭＡＣ５６３、および処理装置５０１を制御し、画像処理装置全体を統括制御する。ＣＰＵ５６０の指示によりＲＤＭＡＣ５６２は、システムバス５６４とＤＲＡＭコントローラ５６６を介してＤＲＡＭ５６５に格納された画像データを読み出し、処理装置５０１に入力する。処理装置５０１は、所望の画像処理を行い、処理結果の画像データをＷＤＭＡＣ５６３に送り出す。このとき、ＷＤＭＡＣ５６３は、予めのＣＰＵ５６０からの指示に基づき、処理装置５０１から受け取った画像データをシステムバス５６４とＤＲＡＭコントローラ５６６を介してＤＲＡＭ５６５に格納する。画像処理装置は、前述の一連の動作を実行することにより画像処理を実施するのである。

【００５４】

前述の画像処理の過程で、処理装置５０１は、接続されたキャッシュ装置５１０を介して、ＤＲＡＭ５６５から各種必要な対象データ（画像、映像、設定値、テーブル、属性情報など）を読み出して画像処理に使用する。また処理装置５０１がプロセッサとプログラムで構成されている場合、キャッシュ装置５１０を介してプログラムを逐次読み出して処理を実行することも可能である。

【００５５】

（キャッシュ装置）

次にキャッシュ装置の動作について説明する。前述の処理装置５０１は、キャッシュ装置を介してＤＲＡＭ５６５からデータを読み出すときに、ＤＲＡＭ５６５上のデータの格納アドレス５１３を、Ｉ／Ｆ５１２を介してキャッシュ判定部５２０に入力する。そして入力された格納アドレス５１３をもとにキャッシュ判定部５２０でキャッシュのヒットもしくはキャッシュミスが判定される。

【００５６】

（キャッシュ判定部）

さらに図２を用いて図５に記載の画像処理装置におけるキャッシュ判定部５２０の回路構成例について説明する。キャッシュ判定部５２０は、比較結果とキャッシュタグ数の大小関係を調べて、キャッシュのヒットを判定する。キャッシュ判定部５２０は、図２のデータ処理装置２００と交換機能付きキャッシュ判定装置２８０とを備える。なお、以後の説明では、「交換機能付きキャッシュ判定装置２８０」を「キャッシュ判定装置２８０」と略称する。

【００５７】

前述の格納アドレス５１３には、アドレスの有効信号である「ｖａｌｉｄ」とアドレス

信号である「ｄａｔａ」とが含まれ、格納アドレス５１３は、データ処理装置２００の第１パイプラインのＤａｔａＳｌｏｔにより移動する。データ処理装置２００は、８個の「ｔａｇ＿ｄａｔａ」を持ち、この８個の「ｔａｇ＿ｄａｔａ」にキャッシュタグ情報が記憶される。データ処理装置２００の一例は、８つのインデックスを有するフル・セット・アソシアティブ方式のキャッシュ装置である。また、第２パイプラインの「上手」から順に０番から７番までの格納場所が、一筆書きの決められた順路で予め定められており、駆動信号「ｓｈｉｆｔ」が有効（アサート状態）のときに「下手」にデータ要素が移動するシフト構造を有している。このシフト構造により、最も古いキャッシュタグは、格納場所０番の「ｔａｇ＿ｄａｔａ」に格納され、最も新しいキャッシュタグは、格納場所７番の「ｔａｇ＿ｄａｔａ」に格納されている。

【００５８】

キャッシュのキャッシュミスが起こる度に、キャッシュタグは格納場所７番から格納場所０番の「ｔａｇ＿ｄａｔａ」へ順番に移動（シフト）され、やがて第２パイプラインから掃き出されてしまう。このキャッシュ判定部は、非常に単純な機構ながら、常に最も古いキャッシュタグやキャッシュデータから順番に捨てられる。このような単純な機構により、一般的なキャッシュ機構の複雑な置き換え（リプレイス）制御を行う必要はない。しかし、たとえ使用頻度（後述の「ヒットカウント値」）が多いキャッシュタグとキャッシュデータであっても、キャッシュミスの多発により、いずれ破棄されてしまう。

【００５９】

そこで、この問題を解決するために、本実施形態のキャッシュ装置では、第２パイプラインから破棄されるキャッシュタグと、キャッシュ判定装置２８０のリオーダバッファに一時的に格納されたキャッシュタグとを交換する機能を持つ。そして、リオーダバッファに一時的に格納されたキャッシュタグを第２パイプラインに復帰させることができる。これらの特徴については、後述の「破棄予定のキャッシュタグ／キャッシュデータの交換機能」と「破棄予定のキャッシュタグ／キャッシュデータの復帰機能」の項で説明する。

【００６０】

（キャッシュ判定）

次に、キャッシュのヒット／キャッシュミスの判定の手順を説明する。ヒット／キャッシュミス判定は、図２のキャッシュ判定装置２８０で行われる。キャッシュがヒットしたかどうかは、データ処理装置２００から出力される処理結果信号「ｔａｇ＿ｉｄ」（２の補数表記）の１ビットの符号ビットを調べることで判定する。データ処理装置２００の出力である有効信号「ｖａｌｉｄ」が有効（アサート状態）であり、かつ符号ビットが１であるとき、「ｔａｇ＿ｉｄ」は負の値であり、キャッシュ判定はキャッシュミスとする。また符号ビットが０であるとき、「ｔａｇ＿ｉｄ」は正の値であり、キャッシュ判定はヒットとする。

【００６１】

データ処理装置２００の出力であるデータ信号「ｄａｔａ」が、キャッシュ判定装置２８０に「ｔａｇ＿ｉｄ」と同期して入力される。これによりキャッシュミスと判定された場合、このデータ信号「ｄａｔａ」がキャッシュミス時のアドレスである「ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ」となる。そしてキャッシュ判定装置２８０では、キャッシュミスのときに駆動信号「ｓｈｉｆｔ」を有効にし、キャッシュミス時のアドレスである「ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ」をデータ処理装置２００のデータ信号「ｔａｇ＿ｄａｔａ」として入力する。キャッシュがキャッシュミスする度に、駆動信号「ｓｈｉｆｔ」が有効となり、前述で説明したように処理結果「ｔａｇ＿ｉｄ」は減算されていく。

【００６２】

始めは処理結果「ｔａｇ＿ｉｄ」に正の値が保持されていても、キャッシュミスが繰り返されると、第２パイプラインがシフトされ、キャッシュタグを表す「ｔａｇ＿ｄａｔａ」が第２パイプラインから掃き出されることもある。この掃き出されるときの処理結果「ｔａｇ＿ｉｄ」の値は、最も古いキャッシュタグの格納番号が０であることから、負の値であることは明らかである。前述のキャッシュ判定で「ｔａｇ＿ｉｄ」の符号を調べるだ

けでよいのは、これに由来する。すなわち、最も古いキャッシュタグが０番、最も新しいキャッシュタグがＮ－１番となるように、格納場所の番号の割り振り方を工夫している。これにより、キャッシュ判定は、最終ステージのデータ処理結果の符号を判別するだけでよい。したがって、本実施形態によれば、キャッシュ判定が非常に簡単である。

【００６３】

また格納番号を第２パイプラインの「上手」から「下手」に０番からＮ－１番に割り振った場合は、「ｔａｇ＿ｉｄ」の値が第２パイプラインの要素数であるＮより小さいかどうかでキャッシュ判定できることは言うまでもない。また前述のように常に最も古いキャッシュデータから順番に捨てる機構のため、本実施形態の一例のキャッシュメモリは、リング式ＦＩＦＯを用いればよい。この場合、キャッシュ判定部５２０とキャッシュメモリ（ＦＩＦＯ）５９０との同期が容易となる。なお、キャッシュ判定がヒットの場合は、「ｔａｇ＿ｉｄ」が指し示す位置のキャッシュメモリに所望のキャッシュデータが格納されていることとなる。

【００６４】

以上の処理により、キャッシュ判定部５２０から、入力された格納アドレス５１３をもとに判定結果５２５である次の信号が出力される。

・データ要素の有効信号「ｖａｌｉｄ」
・キャッシュミス時のＤＲＡＭのデータ格納先であるアドレス信号「ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ」
・キャッシュデータの格納先である「ｔａｇ＿ｉｄ」
・キャッシュ判定結果である「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」
・リオーダバッファ選択信号「ｒｅｏｒｄｅｒ」、またはリオーダバッファ復帰信号「ｒｅｂｉｒｔｈ」
・キャッシュデータの交換信号「ｅｘｃｈａｎｇｅ」

なお、リオーダバッファ選択信号「ｒｅｏｒｄｅｒ」、リオーダバッファ復帰信号「ｒｅｂｉｒｔｈ」、キャッシュデータの交換信号「ｅｘｃｈａｎｇｅ」の動作については、後述する。それぞれ、「破棄予定のキャッシュタグ／キャッシュデータの交換機能」および「破棄予定のキャッシュタグ／キャッシュデータの復帰機能」の項にて説明する。

【００６５】

本実施形態の画像処理装置では、キャッシュのキャッシュミス時のペナルティであるリフィル・レイテンシを隠蔽するためにノンブロッキングのキャッシュ機構を採用している。これは、たとえ判定結果５２５がキャッシュミスと判定されても、後に必要となる情報「ｔａｇ＿ｉｄ、ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ、ｒｅｏｒｄｅｒ（ｒｅｂｉｒｔｈ）、ｅｘｃｈａｎｇｅ」からなる判定結果５２５を待ち合せＦＩＦＯ５４０に退避する。そして、キャッシュミスのキャッシュデータをＤＲＡＭ５６５から読み出し、キャッシュメモリ（ＦＩＦＯ）５９０へ格納する処理が完了する前に、次の画素のキャッシュ判定処理を先行して実行する。このような処理を行うことでキャッシュミスのキャッシュデータをＤＲＡＭ５６５からキャッシュメモリ５９０（ＦＩＦＯ）へリフィルしている間に、続く画素に対するキャッシュ判定を行うことができる。したがって、キャッシュのキャッシュミス時の性能低下を抑制することが可能になる。

【００６６】

なお、後述するように、キャッシュがキャッシュミスしたときの格納アドレスは、アクセス調停部５３０により送信ＦＩＦＯ５５０へ順次格納される。ＤＲＡＭコントローラ５６６は、この送信ＦＩＦＯ５５０から格納アドレスを受け取り、所望のデータをＤＲＡＭ５６５から読み出して、受信ＦＩＦＯ５７０へ書き込んでいく。キャッシュメモリ調停部５８０は、待ち合せＦＩＦＯ５４０から「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」を取り出す。そして、ＤＲＡＭコントローラ５６６は、キャッシュ判定がキャッシュミスであったかヒットであったかを特定する。キャッシュメモリ調停部５８０は、キャッシュ判定がヒットのとき、キャッシュデータをキャッシュメモリ（ＦＩＦＯ）５９０から直接読み出してＩ／Ｆ５１６へ送り出す。一方、キャッシュ判定がキャッシュミスのとき、キャッシュデータ

を受信ＦＩＦＯ５７０から読み出して、キャッシュメモリ（ＦＩＦＯ）５９０へ書き込む。そして、キャッシュメモリ調停部５８０は、このキャッシュデータをＩ／Ｆ５１６へ送り出す。このように、キャッシュミスしたキャッシュデータをＤＲＡＭ５６５から読み出してキャッシュメモリ（ＦＩＦＯ）５９０に更新する一連の処理をリフィルという。

【００６７】

（アクセス調停部）

アクセス調停部５３０は、有効信号「ｖａｌｉｄ」が有効（アサート状態）のとき動作し、それ以外の場合は待機する。そしてアクセス調停部５３０は、キャッシュ判定結果「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」の有効（アサート状態）／無効（ディアサート状態）に応じて、以下の処理を行う。

・キャッシュ判定結果、「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が有効のとき、接続される３つのＦＩＦＯである、送信ＦＩＦＯ５５０、受信ＦＩＦＯ５７０および待ち合せＦＩＦＯ５４０の格納領域の空き状態を評価する。そして３つのＦＩＦＯすべてに空き領域がある場合、「ｔａｇ＿ｉｄ、ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ、ｒｅｏｒｄｅｒ（ｒｅｂｉｒｔｈ）、ｅｘｃｈａｎｇｅ」５３５を待ち合せＦＩＦＯ５４０に書き込む。また同時に、アドレス信号「ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ」５３２を送信ＦＩＦＯ５５０に書き込む。もし空き領域がない場合は、駆動信号である「ｅｎａｂｌｅ」を無効（ディアサート状態）にしてキャッシュ判定部５２０を停止（ストール）して、格納領域が空くまで待機する。

・キャッシュ判定結果、「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が無効のとき、待ち合せＦＩＦＯ５４０の空き状態を評価する。そして空き領域があれば、「ｔａｇ＿ｉｄ、ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ、ｒｅｏｒｄｅｒ（ｒｅｂｉｒｔｈ）、ｅｘｃｈａｎｇｅ」からなる判定結果５３５を待ち合せＦＩＦＯ５４０に書き込む。もし空き領域がない場合、駆動信号である「ｅｎａｂｌｅ」を無効（ディアサート状態）にしてキャッシュ判定部５２０を停止（ストール）し、格納領域が空くまで待機する。

【００６８】

（キャッシュメモリ調停部）

キャッシュメモリ調停部５８０は、接続された２つのＦＩＦＯ、すなわち受信ＦＩＦＯ５７０と待ち合せＦＩＦＯ５４０の各々の格納領域にデータがあるかどうかを評価する。キャッシュメモリ調停部５８０は、まず待ち合せＦＩＦＯ５４０から処理すべきキャッシュ判定結果から「ｔａｇ＿ｉｄ、ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ、ｒｅｏｒｄｅｒ（ｒｅｂｉｒｔｈ）、ｅｘｃｈａｎｇｅ」を取り出す。なお、待ち合せＦＩＦＯ５４０が空の場合、処理すべきキャッシュ判定結果がないため、キャッシュメモリ調停部５８０は、何もせず待機する。そして、キャッシュ判定結果「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が無効（ディアサート状態）であるか有効（アサート状態）あるかに応じて、次の処理を行う。

・キャッシュ判定結果「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が無効（ディアサート状態）のときは、キャッシュがヒットの状態である。そこで、待ち合せＦＩＦＯ５４０から同時に取り出した「ｔａｇ＿ｉｄ」とキャッシュメモリ（ＦＩＦＯ）５９０のライトポインタからキャッシュメモリ（ＦＩＦＯ）５９０の格納アドレスを算出する。その算出した格納アドレスをもとにキャッシュされているデータをキャッシュメモリ（ＦＩＦＯ）５９０から「ｒｅａｄ＿ｄａｔａ」５９２として直接読み出す。そして、キャッシュデータ「ｖａｌｉｄ、ｃａｃｈｅ＿ｄａｔａ」５８５としてＩ／Ｆ５１６へ送出する。

【００６９】

キャッシュ判定結果で「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が有効（アサート状態）のときは、キャッシュがキャッシュミスの状態である。キャッシュメモリ調停部５８０は、ＤＲＡＭ５６５からのキャッシュされていないデータが受信ＦＩＦＯ５７０に届いているかどうかを確認する。届いていない場合は、届くまで待機する。届いている場合、受信ＦＩＦＯ５７０から更新すべきキャッシュデータを取り出す。そして、キャッシュメモリ（ＦＩＦＯ）５９０のライトポインタの指す格納領域に、取り出したデータを「ｗｒｉｔｅ＿ｄａｔａ」５８２として書き込む。同時に、キャッシュデータ「ｖａｌｉｄ、ｃａｃｈｅ＿

ｄａｔａ」５８５としてＩ／Ｆ５１６へ送出する。そして最後に、キャッシュメモリ（ＦＩＦＯ）５９０のライトポインタを１だけインクリメントする。ＦＩＦＯの容量を越えた場合は、ライトポインタを０に戻す。

【００７０】

前述の「破棄予定のキャッシュタグ／キャッシュデータの交換機能」や「破棄予定のキャッシュタグ／キャッシュデータの復帰機能」を実現するためのキャッシュメモリ調停部５８０の追加機能については、後述する。

【００７１】

最後に処理装置に接続されたＩ／Ｆ５１６は、上記の過程で得たキャッシュデータ「ｖａｌｉｄ、ｃａｃｈｅ＿ｄａｔａ」を処理装置５０１に送出する。

【００７２】

（破棄予定のキャッシュタグ／キャッシュデータの交換機能）

例えばプリンタ画像処理では紙面の大半は白地であるケースがあり、この白地を印刷するために必要となる画像処理のデータは、繰り返し使用される。このような使用頻度の多いデータに対応するキャッシュタグとキャッシュデータは、キャッシュ装置に常に保持した方が、高速に画像処理をする上で有利である。従って本実施形態のキャッシュ装置には、使用頻度が多いキャッシュタグとキャッシュデータが、キャッシュミスの多発により破棄されないための機構を備える。以下で、本発明の特徴である、「破棄予定のキャッシュタグ／キャッシュデータの交換機能」について記載する。

【００７３】

前述のキャッシュ判定装置２８０は、キャッシュタグ交換回路２５０を内包する。図３にキャッシュタグ交換回路２５０のブロック図を示す。また図４にキャッシュタグ交換回路２５０の処理を説明するフローチャート４００を示す。図３のキャッシュタグ交換回路２５０には、第１パイプラインから次の信号が出力される。

・データ要素の有効信号「ｖａｌｉｄ」。
・キャッシュミス時のＤＲＡＭのデータ格納先であるアドレス信号「ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ」
・キャッシュデータの格納先である「ｔａｇ＿ｉｄ」。

【００７４】

また、キャッシュタグ交換回路２５０に第２パイプラインから次の信号が出力される。

・破棄予定のキャッシュタグの有効信号「ｓｗｅｅｐ」。
・破棄予定のキャッシュタグ「ｓｗｔａｇ」。

【００７５】

更に、キャッシュ判定装置２８０の判定部２６０で処理されたキャッシュ判定結果である「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が、キャッシュタグ交換回路２５０に入力される。

【００７６】

キャッシュタグ交換回路２５０は、キャッシュタグ毎に使用頻度をカウントするために、ヒットカウント値算出回路３００と、破棄予定のキャッシュタグの使用頻度が多いときに一時的に保持するリオーダ回路３２０を備える。キャッシュタグ交換回路２５０の入力である、データ要素の有効信号「ｖａｌｉｄ」が無効（ディアサート状態）のとき、何もせず待機する(フローチャート４００には不図示)。データ要素の有効信号「ｖａｌｉｄ」が有効（アサート状態）で、かつ、キャッシュ判定結果、「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が無効（ディアサート状態）の場合、データ処理装置２００の判定結果はキャッシュヒットを示す。

【００７７】

Ｓ４０５において、データ要素の有効信号「ｖａｌｉｄ」が有効（アサート状態）で、かつ、キャッシュ判定結果「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が有効（アサート状態）の場合、データ処理装置２００の判定結果は、キャッシュキャッシュミスを示す。キャッシュヒットの場合、Ｓ４１０において、ヒットカウント値算出回路３００は、図２の８個のＪｕｄｇｅにおいてタグデータとデータの一致した一致回数を積算するためのもので、デコ

ード回路３０２により、キャッシュデータの格納先である「ｔａｇ＿ｉｄ」の指し示すヒットカウンタ３０８のヒットカウント値３０４を１だけインクリメントする。キャッシュミスの場合、Ｓ４１５において、破棄予定のキャッシュタグ「ｓｗｔａｇ」の有効信号「ｓｗｅｅｐ」を確認して、交換機能を動作させるかどうかを判定する。

【００７８】

破棄予定のキャッシュタグの有効信号「ｓｗｅｅｐ」が無効（ディアサート状態）のとき、制御回路「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は、交換機能を動作させる必要がない。そこでＳ４９０において、制御回路「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は、シフト信号「ｓｈｉｆｔ」３０５を有効にし、ヒットカウント値３０４を「上手」から「下手」に移動する。また同時に、前述のデータ処理装置２００の第２パイプラインのキャッシュタグも「上手」から「下手」に移動する。実施形態１では、最も「上手」に位置するキャッシュタグには、キャッシュミス直後の「ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ」が格納される。このときのキャッシュタグのヒットカウント値３０４は必ず「０」でなければならないため、キャッシュタグが「上手」から「下手」へ移動するときに、そのヒットカウント値３０４は初期化され、カウント値の初期値としてゼロが与えられる。

【００７９】

一方Ｓ４１５において、破棄予定のキャッシュタグの有効信号「ｓｗｅｅｐ」が有効（アサート状態）のとき、制御回路「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は、交換機能を動作させ、Ｓ４２０において、リオーダバッファの空き状態を確認する。リオーダバッファの有効信号である「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ」３３２が無効（ディアサート状態）のとき、リオーダバッファである「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ」３３４は空き状態である。従ってＳ４２５において、「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は無条件で「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ」３３４に破棄予定のキャッシュタグ「ｓｗｔａｇ」を退避できる。また、最終段のノードの破棄予定のデータについてのヒット回数を計測し、そのカウント結果であるヒットカウント値３０７をリオーダバッファのヒットカウント値「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」３３６に格納し退避できる。これらの退避を実現するため、「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は、シフト信号「ｓｈｉｆｔ」３０５を有効（アサート状態）にして、ヒットカウント値３０４を「上手」から「下手」に移動する。また引き続きＳ４９０において、前述のデータ処理装置２００の第２パイプラインのキャッシュタグも「上手」から「下手」に移動する。

【００８０】

Ｓ４２０において、リオーダバッファの有効信号「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ」３３２が有効（アサート状態）のとき、既にリオーダバッファに退避された「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ」３３４が存在する。そこで、Ｓ４３０において、この「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ」３３４と入力された「ｄａｔａ」とを比較回路３６０で比較し、キャッシュのヒット判定を再評価する。再評価の結果、キャッシュヒットとなった場合（Ｓ４３０でＹＥＳ）、Ｓ４３５において「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は、ヒットカウンタ３３８のヒットカウント値「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」３３６を１だけインクリメントする。また入力された「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」を無効（ディアサート状態）にする。また、キャッシュメモリ調停部５８０でキャッシュデータをリオーダバッファか読み出すための切り替え信号「ｒｅｏｒｄｅｒ」を有効（アサート状態）にする。

【００８１】

Ｓ４３０における再評価の結果、キャッシュミスの場合、リオーダ回路３２０は、リオーダバッファに退避された「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ」３３４と入力された破棄予定のキャッシュタグ「ｓｗｔａｇ」との何れか一方を選択する。本実施形態では、使用頻度を表すヒットカウント値が、より大きいキャッシュデータは、後に再利用される可能性が高いと考える。そこで、Ｓ４４０において、破棄予定のキャッシュタグのヒットカ

ウント値「ｈｉｔ＿ｃｏｕｎｔ」３０７とリオーダバッファのヒットカウント値「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」３３６、すなわち第２パイプラインの最終段で計測された、最新の２つのカウント値を大小比較回路「Ｃｏｍｐａｒｅ」３２２により比較する。Ｓ４４０で、破棄予定のキャッシュタグのヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」３０７が大きい場合、Ｓ４４５に移行する。Ｓ４４５において、入力された破棄予定のキャッシュタグ「ｓｗｔａｇ」をリオーダバッファのキャッシュタグ「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ」３３４に上書きし、第２パイプラインの先頭のｔａｇ＿ｄａｔａとして戻す。また同時に、ヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」３０７をリオーダバッファのヒットカウント値「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」３３６に上書きする。そして交換したことを示す、交換信号「ｅｘｃｈａｎｇｅ」を有効（アサート状態）にして出力する。

【００８２】

Ｓ４４０で、破棄予定のキャッシュタグのヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」３０７が小さい場合、入力された破棄予定のキャッシュタグ「ｓｗｔａｇ」を破棄する。また同時に、そのヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」３０７を破棄する。最後に、Ｓ４９０において「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は、シフト信号「ｓｈｉｆｔ」３０５を有効（アサート状態）にし、ヒットカウント値３０４を「上手」から「下手」に移動する。また同時に、前述のデータ処理装置２００における第２パイプラインのキャッシュタグも「上手」から「下手」に移動することとなる。

【００８３】

以上のように、キャッシュタグのヒットカウント値の大きさにより、優先順位を定め、この優先順位に応じてリオーダ回路に保持するキャッシュタグを交換する。

【００８４】

（キャッシュメモリ調停部５８０の追加機能）

図６に、キャッシュメモリ調停部の処理フローの一例を示す。キャッシュメモリ調停部５８０の基本動作は前述の通りであるが、ここでは図６（ａ）を用いて、キャッシュタグ交換機能に連携する動作について説明する。

【００８５】

Ｓ６０５において、待ち合わせＦＩＦＯ５４０から「ｔａｇ＿ｉｄ、ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ、ｅｘｃｈａｎｇｅ、ｒｅｏｒｄｅｒ」を取り出す。Ｓ６２０でキャッシュ判定結果の「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が無効（ディアサート状態）のときは、Ｓ６２５において、リオーダバッファにキャッシュデータがあるかどうかを示す入力信号である「ｒｅｏｒｄｅｒ」を確認する。そして、Ｓ６２５で「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が有効（アサート状態）のとき、Ｓ６３５において、リオーダバッファに格納されたキャッシュデータを「ｒｅａｄ＿ｄａｔａ」５９２として直接読み出す。そして、Ｓ６９０においてキャッシュデータ「ｖａｌｉｄ、ｃａｃｈｅ＿ｄａｔａ」５８５としてＩ／Ｆ５１６へ送出する。

【００８６】

Ｓ６２０で、キャッシュ判定結果「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が有効（アサート状態）のとき、Ｓ６５０において、受信ＦＩＦＯ５７０から更新すべきキャッシュデータを取り出す。そしてＳ６９０において、キャッシュデータ「ｖａｌｉｄ、ｃａｃｈｅ＿ｄａｔａ」５８５としてＩ／Ｆ５１６へ送出する。

【００８７】

一方、Ｓ６５５で入力された交換信号「ｅｘｃｈａｎｇｅ」が有効（アサート状態）のとき、Ｓ６６０において、キャッシュメモリ（ＦＩＦＯ）５９０のライトポインタの指す格納領域のキャッシュデータを読み出し、リオーダバッファへ退避する。そして、Ｓ６７０において、受信ＦＩＦＯ５７０から取り出したデータをライトポインタの指す格納領域に「ｗｒｉｔｅ＿ｄａｔａ」５８２として書き込む。更に、Ｓ６８０において、キャッシュメモリ（ＦＩＦＯ）５９０のライトポインタを１だけインクリメントする。ＦＩＦＯの容量を越えた場合は、ライトポインタを０に戻す。

【００８８】
なお、図４のＳ４１５で、破棄予定のキャッシュタグの有効信号「ｓｗｅｅｐ」が有効で、Ｓ４２０でリオーダバッファの有効信号である「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ」３３２が無効のとき、キャッシュメモリ調停部５８０は以下の動作を行う。
【００８９】
まず、このケースにおいて、リオーダ回路の「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０は無条件で「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ」３３４に破棄予定のキャッシュタグ「ｓｗｔａｇ」を退避している。キャッシュメモリ調停部５８０は、これに連動するためにキャッシュメモリ（ＦＩＦＯ）５９０のライトポインタに着目する。まず、キャッシュ装置の初期化後、キャッシュメモリ調停部５８０のライトポインタは０である。そして、前述のようにキャッシュ判定結果がキャッシュミスになる毎に、ライトポインタは１だけインクリメントされる。ライトポインタがＦＩＦＯ容量を超えて０の状態のとき、次にキャッシュミスが起こると、ライトポインタが０のキャッシュデータが始めて破棄されることになる。つまり、キャッシュメモリ調停部５８０は、初めてライトポインタが０に戻るときのキャッシュデータを無条件でリオーダバッファに格納する。
【００９０】
本実施形態のキャッシュ装置によれば、比較的簡単な構成で、高速に処理可能なフル・セット・アソシアティブ方式のキャッシュ装置を実現できる。一般的なダイレクトマップ方式のキャッシュ判定部では、アドレスの下位ビットからキャッシュタグを管理するタグメモリの格納先を算出するため、相関性の低いアドレスに対しては容易にキャッシュ競合を引き起こしてしまう。セット・アソシアティブ方式のウェイ数を増やしていくことが、このキャッシュ競合の確率を低減させるための１つの解決策となる。しかしながら、処理装置の数が多くなると多数のウェイに対応する必要があり、一般的なキャッシュ装置の実装では、キャッシュ判定部のセレクタの論理段数が多く、タイミング収束が難しくなる。従って、動作周波数が高くなると動作させることができなくなる。これに対して、本実施形態のキャッシュ判定部は、パイプライン構成で判定するために、動作周波数がたとえ高くても確実に動作する。
【００９１】
また本実施形態のキャッシュ装置は、破棄予定のキャッシュタグとキャッシュデータをリオーダバッファに一時的に保存し、ヒットカウント値に応じて重要なキャッシュデータを常に保持するための交換機能を備える。そのため、繰り返し画像処理に使用されるキャッシュデータを優先的に残し、キャッシュ効率を向上させることができる。
【００９２】
＜実施形態２＞
さらに図７、図８および図９を用いて、実施形態２における、図５に記載のキャッシュ判定部５２０とキャッシュメモリ調停部５８０の構成の例を説明する。本実施形態のキャッシュ判定装置２８０は、リオーダバッファに一時的に退避された破棄予定のキャッシュタグをデータ処理装置２００の第２パイプラインへ復帰させる機能を有する。前述の実施形態１と同様の動作については、説明を省略する。
【００９３】
（破棄予定のキャッシュタグ／キャッシュデータの復帰機能）
実施形態１で説明した「破棄予定のキャッシュタグ／キャッシュデータの交換機能」から明らかなように、リオーダバッファには使用頻度を表すヒットカウント値の大きなキャッシュタグが残っている。これらをデータ処理装置２００の第２パイプラインへ復帰させることにより、第２パイプラインのキャッシュタグが循環して、使用頻度の少ないキャッシュタグを優先的に破棄することが可能となる。
【００９４】
この復帰機能の例では、図４または図８のフローチャートにおいて、前述のキャッシュのヒット判定を再評価でリオーダバッファに退避されたキャッシュタグがヒットしたときに実行する（Ｓ４３０でＹＥＳ）。ヒット時に復帰機能を実行することで、通常のキャッ

シュミス時のキャッシュタグの下手へのシフト動作と排他的に動作することが可能となる。この場合、図４のＳ４３５の処理を一部変更して、図８のＳ４３８に示す復帰信号「ｒｅｂｉｒｔｈ」を有効（アサート状態）にし、復帰機能が動作したことを後段に位置するキャッシュメモリ調停部５８０に伝える。

【００９５】

また、復帰機能を備えるキャッシュタグ交換回路２５０を図７に示す。図７では、前述の実施形態１の同回路を示す図３と違い、結線３８０により、リオーダバッファのヒットカウント値３３６をヒットカウント値算出回路３００の最も「上手」に位置するキャッシュタグのヒットカウント値３０４に復帰できる。また、この復帰機能の動作により、リオーダバッファのキャッシュタグ３３４とヒットカウント値３３６を保持するレジスタには空き状態となる。そこで、復帰機能の動作時に、リオーダバッファのキャッシュタグ３３４には、外部入力の破棄予定のキャッシュタグ「ｓｗｔａｇ」とその有効信号「ｓｗｅｅｐ」を格納する。また、リオーダバッファのヒットカウント値３３６には、最も「下手」に位置するヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」３０７を格納する。

【００９６】

なお、この復帰機能は、外部からの割り込み命令（トリガ）により実行されてもよい。また、この復帰機能は、破棄予定のキャッシュタグの「ｈｉｔ＿ｃｏｕｎｔ」３０７とリオーダバッファのキャッシュタグの「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」３２６を常に監視し、「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」３２６が大きければ実行してもよい。

【００９７】

以上のように、キャッシュタグのヒットカウント値の大きさにより、優先順位を定め、この優先順位に応じてリオーダ回路のキャッシュタグを第２パイプラインに復帰（再入力）する。

【００９８】

（キャッシュメモリ調停部５８０の追加機能）

キャッシュメモリ調停部５８０の基本動作は前述の通りであるが、ここではキャッシュタグ復帰機能に連携する動作について説明する。

【００９９】

キャッシュ判定結果「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が無効（ディアサート状態）のときは、リオーダバッファのキャッシュデータの復帰を示す入力信号である「ｒｅｂｉｒｔｈ」を確認する。そして、有効（アサート状態）のとき、リオーダバッファに格納されているキャッシュデータをキャッシュデータ「ｖａｌｉｄ、ｃａｃｈｅ＿ｄａｔａ」５８５としてＩ／Ｆ５１６へ送出する（Ｓ６９０）。また、ライトポインタの指す格納領域のキャッシュデータとリオーダバッファに格納されているキャッシュデータを交換する。そしてライトポインタを１だけインクリメントする。

【０１００】

本実施形態のキャッシュ装置によれば、破棄予定のキャッシュタグとキャッシュデータをリオーダバッファに一時的に保存し、ヒットカウント値に応じて重要なキャッシュデータを第２パイプラインに復帰する機能を備える。そのため、繰り返し画像処理に使用されるキャッシュデータを優先的に残し、使用頻度の少ないキャッシュデータを優先的に破棄し、キャッシュ効率を向上することができる。また、キャッシュミス時に前述の実施形態１の交換機能を実施し、ヒット時に本実施形態の復帰機能を実施することで、実施形態１と本実施形態を併用することができる。これらの併用により、さらにキャッシュ効率を向上することができる。

【０１０１】

＜実施形態３＞

さらに図１０を用いて、実施形態３における、図５に記載されたキャッシュ判定部５２０の別の構成例を説明する。本実施形態では、実施形態１のデータ処理装置２００の内部にヒットカウント値算出回路を備えている。実施形態１では、キャッシュタグ交換回路２

５０の１ステージで使用頻度をカウントする。この場合、データ処理装置２００の第２パイプラインのデータ要素数が増えることで、ヒットカウント値算出回路３００のデコード回路３０２の遅延が大きくなる。結果として、キャッシュ判定部５２０の動作周波数は向上し辛くなる。本実施形態では、データ処理装置２００の第２パイプラインに後述するヒットカウント値算出回路７３０や７５０が組み込まれており、上記のヒットカウント値算出回路３００が不要となる。そのため、ヒットカウント値算出のために動作周波数を向上しても動作にそれほど影響はない。

【０１０２】

なお、実施形態３の図１０に記載の基本構成は、実施形態１の図１に記載の基本構成に相当する。そして、この基本構成を実施形態１の図２のように複数連結し、データ処理装置２００のデータ要素の数を増やす使い方は、実施形態１の使い方と同様である。

【０１０３】

また図１０において、第１パイプライン７００は、図の「上手」である左側から「下手」である図面上の右側に向けてデータが移動する。また第２パイプライン７６０は「上手」である図面上の右側から「下手」である図面上の左側に向けてデータが移動している。この状態は、実施形態１と同様である。、図１０における各種の信号名称やその意味は、図１で説明した信号名称やその意味と同様であるので、同じ信号に対する説明は省略する。実施形態３では、基本構成でヒットカウント値を算出する仕組みが実施形態１と異なるので、これに関係するデータ処理回路の該当箇所について説明する。

【０１０４】

実施形態１では、後段に位置するキャッシュ判定装置２８０で判定を完了してから、ヒットカウント値を算出していた。そのため図１におけるデータ処理回路の比較回路１２２、１２４、１４２、１４４では、第１パイプライン１００の有効信号「ｖａｌｉｄ［ｌ－１］」１１２や「ｖａｌｉｄ［ｌ］」１３２を比較で考慮する必要がなかった。本実施形態では、比較と同時にヒットカウント値を算出するため、図１０の比較回路７２２、７２４、７４２、７４４は、第１パイプライン７００の有効信号「ｖａｌｉｄ［ｌ－１］」７１２や「ｖａｌｉｄ［ｌ］」７３２を考慮して比較を実行する。そして、比較回路７２２、７２４、７４２、７４４は、比較結果７２３、７２５、７４３、７４５、７６３を算出している。第２パイプライン７６０のパイプライン・レジスタ７８０と７７０には、実施形態１とは異なりヒットカウント値を保持するためのレジスタ７８１と７７１が追加されている。これらのレジスタ７８１、７７１は、実施形態１の図３におけるキャッシュタグ交換回路２５０のヒットカウント値算出回路３００のヒットカウント値３０４に相当する。これらのレジスタに保持されるヒットカウント値は、第２パイプラインの駆動信号「ｓｈｉｆｔ」７６２の状態によらず値が変化する。当然、有効信号「ｔａｇ＿ｖａｌｉｄ」とデータ信号「ｔａｇ＿ｄａｔａ」とは異なり、駆動信号「ｓｈｉｆｔ」７６２だけで「保持」と「更新」が切り替わる分けではない。

【０１０５】

また、第２パイプライン７６０には、これらのレジスタに保持するヒットカウント値を算出するためのヒットカウント値算出回路７３０と７５０を備える。ヒットカウント値算出回路７３０は、第２パイプライン７６０のパイプライン・レジスタ７７０からヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ［ｉ＋１］」７７８を、パイプライン・レジスタ７８０からヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ［ｉ］」７８８を受け取る。そしてヒットカウント値算出回路７３０は、比較結果７２３、７２５、７４３と外部入力の駆動信号「ｓｈｉｆｔ」７６２により制御されて、新たなヒットカウント値７７９を算出する。そして、新たに算出したヒットカウント値７７９をパイプライン・レジスタ７８０のレジスタ７８１に書き込む。

【０１０６】

同様に、ヒットカウント値算出回路７５０は、第２パイプライン７６０のパイプライン・レジスタ（不図示）からヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ［ｉ＋２］」７６８を受け取る。そして更にヒットカウント値算出回路７５０は、パイプライン・レジスタ７７

０から「ｈｉｔ＿ｃｏｕｎｔ［ｉ＋１］」７７８を受け取る。そしてヒットカウント値算出回路７５０は、比較結果７４３、７４５、７６３と外部入力の駆動信号「ｓｈｉｆｔ」７６２により制御されて、ヒットカウント値７６９を算出し、パイプライン・レジスタ７７０へ書き込む。

【０１０７】

ヒットカウント値算出回路７３０と７５０は同様に動作するため、ヒットカウント値算出回路７３０を用いて説明する。駆動信号「ｓｈｉｆｔ」７６２が無効（ディアサート状態）のときと有効(アサート状態)のときで動作が異なる。以下、順を追って説明する。

【０１０８】

（駆動信号「ｓｈｉｆｔ」７６２が無効（ディアサート状態）のとき）

駆動信号「ｓｈｉｆｔ」７６２が無効の場合、第２パイプライン７６０の各データ要素は「上手」から「下手」に移動しない。そのため、ヒットカウント値算出回路７３０の制御回路である「Ｃｏｎｔｒｏｌｌｅｒ」７３２は、セレクタ７３４を操作して「ｈｉｔ＿ｃｏｕｎｔ［ｉ］」７８８を、加算器である「ａｄｄｅｒ」７３８に代入する。このときの比較対象は、第２パイプラインのパイプライン・レジスタ７８０のデータ信号「ｔａｇ＿ｄａｔａ」と第１パイプラインのデータ信号「ｄａｔａ［ｌ－１］」であり、比較結果７２３に着目する。そして「Ｃｏｎｔｒｏｌｌｅｒ」７３２は、この比較結果７２３によりセレクタ７３６を操作する。

・比較結果７２３が有効のとき、セレクタ７３６は固定値「１」を選択し、「ａｄｄｅｒ」７３８により、「ｈｉｔ＿ｃｏｕｎｔ［ｉ］」７８８を１だけインクリメントする。

・比較結果７２３が無効のとき、セレクタ７３６は固定値「０」を選択し、「ｈｉｔ＿ｃｏｕｎｔ［ｉ］」７８８はそのままの値とする。

【０１０９】

（駆動信号「ｓｈｉｆｔ」７６２が有効(アサート状態)のとき）

駆動信号「ｓｈｉｆｔ」７６２が有効の場合、第２パイプライン７６０の各データ要素は「上手」から「下手」に移動する。そのため、「Ｃｏｎｔｒｏｌｌｅｒ」７３２は、セレクタ７３４を操作して「ｈｉｔ＿ｃｏｕｎｔ［ｉ＋１］」７７８を「ａｄｄｅｒ」７３８に代入する。そして「Ｃｏｎｔｒｏｌｌｅｒ」７３２は、判定結果７２５と７４３によりセレクタ７３６を操作する。

【０１１０】

判定結果７２５と７４３が共に有効のとき、第２パイプライン７６０のパイプライン・レジスタ７７０のデータ信号「ｔａｇ＿ｄａｔａ」と第１パイプラインの２つのデータ信号「ｄａｔａ［ｌ－１］」と「ｄａｔａ［ｌ］」とが共に等しい。そこで、セレクタ７３６は固定値「２」を選択し、「ａｄｄｅｒ」７３８により、「ｈｉｔ＿ｃｏｕｎｔ［ｉ＋１］」７７８を２だけ加算する。

【０１１１】

判定結果７２５と７４３の何れか１つが有効のとき、第２パイプライン７６０のパイプライン・レジスタ７７０のデータ信号「ｔａｇ＿ｄａｔａ」と第１パイプラインのデータ信号「ｄａｔａ［ｌ－１］」もしくは「ｄａｔａ［ｌ］」のどちらかと等しい。そこで、セレクタ７３６は固定値「１」を選択し、「ａｄｄｅｒ」７３８により、「ｈｉｔ＿ｃｏｕｎｔ［ｉ＋１］」７７８を１だけインクリメントする。

【０１１２】

判定結果７２５と７４３が共に無効のとき、第２パイプライン７６０のパイプライン・レジスタ７７０のデータ信号「ｔａｇ＿ｄａｔａ」と第１パイプラインの２つのデータ信号「ｄａｔａ［ｌ－１］」と「ｄａｔａ［ｌ］」が共に異なる。そこで、セレクタ７３６は固定値「０」を選択し、「ｈｉｔ＿ｃｏｕｎｔ［ｉ＋１］」７７８はそのままの値とする。そして、上記の手順で得られたヒットカウント値７７９を第２パイプラインのパイプライン・レジスタのヒットカウント値を格納するレジスタ７８１に書き込む。以上のような動作により、基本構成の中でヒットカウント値の算出が可能となる。

【０１１３】

しかしながら、実施形態３では実施形態１と異なり、以下で説明する制限事項を伴う。「Ｃｏｎｔｒｏｌｌｅｒ」７３２、７５２は、比較結果７２３、７２５、７４３、７４５、７６３に基づいてヒットカウント値を加算するが、この比較結果が最終的な比較結果と異なる場合、実施形態３ではヒットカウント値を過剰に見積もってしまう。

【０１１４】

すなわち、本実施形態のキャッシュ装置では、着目するデータ要素より先行するデータ要素の判定結果がキャッシュミスの場合、第２パイプラインのデータ要素は下流へ移動する。この移動が原因で、各々のデータ処理回路で着目するデータ要素が比較されたときに存在したはずの第２パイプラインのデータ要素が、着目するデータ要素が判定装置に入るまでに破棄されているケースが起こり得る。このケースでは、比較結果が有効でヒットカウント値を増加した後に判定装置でキャッシュミスと訂正されるため、ヒットカウント値は正しい値より大きな値となってしまう。

【０１１５】

しかしながら、上記の制限事項は、すべてのキャッシュデータに対してヒットカウント値が常に過剰に見積もられる傾向となるので、キャッシュデータにより過少に見積もられることはない。また本実施形態の目的は、ヒットカウント値の小さいキャッシュデータをキャッシュ装置から優先的に破棄し、なるべく使用頻度の多いキャッシュデータを残すことである。それには、ある程度、相対的にヒットカウント値の比較ができ、極端にヒットカウント値の小さいキャッシュデータを破棄できればよく、厳密にヒットカウント値を求めなくてもよい。

【０１１６】

仮に、過剰に見積もられたヒットカウント値が原因で、キャッシュ装置から破棄されるキャッシュデータが出たとすると、その後、そのキャッシュデータは再利用され、当然キャッシュミスとして判定される。そして、そのキャッシュデータは改めてリフィルされることとなる。つまり、このようなケースでは、キャッシュ効率が若干低下することが起きてしまうが、一方で画像処理装置の機能そのものが損なわれるわけではないとも言える。

【０１１７】

つまり、このような制限事項があったとしても、本発明の課題を解決するには十分であると言える。また、本実施形態ではこの制限事項と引き替えに回路の動作周波数を高められるという効果があり、動作周波数の向上により画像処理装置そのものの処理性能は向上できる。つまり、画像処理装置の総合的な処理性能は、きわめて高いと言える。

【０１１８】

＜実施形態４＞

さらに図１１を用いて、実施形態３の制限事項を緩和した基本構成の別の例を説明する。また図１１において、第１パイプライン１１００は、図の「上手」である左側から「下手」である図面上の右側に向けてデータが移動する。また第２パイプライン１２００は「上手」である図面上の右側から「下手」である図面上の左側に向けてデータが移動している。この状態は、実施形態１や３と同様である。図１１における各種の信号名称やその意味は、図１や図１０で説明した信号名称やその意味と同様であるので、同じ信号についての説明は省略する。実施形態３では、基本構成でヒットカウント値を算出する仕組みが実施形態１と異なるので、これに関係するデータ処理回路の該当箇所について説明する。

【０１１９】

実施形態４のデータ処理回路について説明する。なお図１０のデータ処理回路とそこに含まれる回路要素は、図１１では、比較回路１０２１～１０２９と選択および減算回路１０４０（セレクタ１０４６とデクリメンタ１０４８を含む）とに分かれている。

【０１２０】

本実施形態では、２方向の一方（第１パイプライン）のデータ要素である１つのデータ信号「ｄａｔａ」と、他方（第２パイプライン）のデータ要素である複数のデータ信号「ｔａｇ＿ｄａｔａ」１２２０～１２２７とを同時に比較する。もし１つでも等しい「ｔａｇ＿ｄａｔａ」がある場合、そのときの「ｔａｇ＿ｄａｔａ」の格納番号を「ｔａｇ＿ｉ

ｄ」として記憶する。等しい「ｔａｇ＿ｄａｔａ」が複数ある場合は、最も「数の大きいものを優先的に選択する」などとしてよい。

【０１２１】

具体的には、まず「ｖａｌｉｄ［ｌ］」と「ｄａｔａ［ｌ］」を、一度に比較したい「ｔａｇ＿ｄａｔａ」の数より１だけ大きいノード１０３０＿１～１０３０＿９まで複製する。図７（ｂ）に記載の一例では、比較対象のデータ要素「ｔａｇ＿ｄａｔａ」は８個存在するので、複製するノード数は９個となる。ここで１だけ余分にノードを作る意味は、実施形態３と同様に第２パイプラインのデータ要素が移動し、比較が失敗するケースに備えるためである。次に各ノードに接続される９個の比較回路１０２１～１０２９を設け、各ノード１０３０＿１～１０３０＿９と接続する。さらに９個の比較回路１０２１～１０２９の各々に、「ｔａｇ＿ｖａｌｉｄ［ｉ］」と「ｔａｇ＿ｄａｔａ［ｉ］」の組み１２２０から「ｔａｇ＿ｖａｌｉｄ［ｉ＋８］」と「ｔａｇ＿ｄａｔａ［ｉ＋８］」の組１２２７とを各々接続する。

【０１２２】

比較回路１０２１～１０２９の各々の動作は、実施形態３と同様であり、その９本の比較結果がセレクタ１０４６に入力される。そしてセレクタ１０４６は、比較結果が等しい「ｔａｇ＿ｄａｔａ」の格納番号である「Ｎｏｄｅ＝ｉ」から「Ｎｏｄｅ＝ｉ＋８」のいずれかを選択する。またセレクタ１０４６は、上記９種類の比較結果がいずれも等しくないときは、入力された処理結果信号「ｔａｇ＿ｉｄ［ｌ］」を選択する。

【０１２３】

実施形態３と同様に、格納番号を「数の大きいものを優先的に選択する」という方法で選択する。そして、外部からの駆動信号「ｓｈｉｆｔ」１２０２が有効の場合、セレクタ１０４６の選択結果からデクリメンタ（減算器）１０４８を用いて格納番号を１だけ減算することで調整する。

【０１２４】

そして、本実施形態では、第１パイプライン（１１００）の１区間において、第２パイプライン（１２００）のデータ要素のヒットカウント値を保持するために、８個のパイプライン・レジスタ１０６０～１０６７を備える。また、上記の比較結果を基にヒットカウント値を算出するための、８個のヒットカウント値算出回路１０５０～１０５７を備える。第２パイプライン（１２００）の８個のデータ要素１２２０から１２２７と、ヒットカウント値の８個のパイプライン・レジスタ１０６０～１０６７と、８個のヒットカウント値算出回路１０５０～１０５７の接続のし方は、実施形態３と同様でよい。

【０１２５】

上記のように、接続のし方を工夫することで、本実施形態では、実施形態３の図１０で示していた８区間（８ステージ）のパイプラインを１区間（１ステージ）で実現しており、１つの入力データに対する比較完了までのレイテンシが８から１に削減できる。そして、このようなレイテンシの削減により、本実施形態では、実施形態３で説明した制限事項の発生を非常に少なくすることができる。また図１１の第１パイプラインの１区間（１ステージ）を基本構成としてそれを連結することにより、非常に多くのデータ要素に対応できることは言うまでもない。

【０１２６】

＜実施形態５＞

さらに図１２を用いて、実施形態１および２で用いたリオーダバッファを複数バッファに拡張した別の構成例を説明する。本実施形態では、リオーダバッファに複数のキャッシュタグを格納できる構成となる。実施形態１および２では、データ処理装置２００に格納された８個のキャッシュタグとキャッシュタグ交換回路２５０のリオーダバッファに格納された１個のキャッシュタグがあった。つまり、実質的には９ウェイのフル・セット・アソシアティブ方式のキャッシュタグ判定部となっていた。

【０１２７】

実施形態５では、リオーダ回路３２０を複数のキャッシュタグに拡張することにより、

例えば、M個のリオーダバッファに拡張する場合、（８＋M）ウェイのフル・セット・アソシアティブ方式のキャッシュタグ判定部となる。実施形態５では、フル・セット・アソシアティブ方式のウェイ数をさらに増加してキャッシュ効率を向上することができる。

【０１２８】

また、既に開発済みのデータ処理装置２００を変更せずに、キャッシュ判定装置２８０のリオーダ回路を拡張することでキャッシュのウェイ数を拡張できる。本実施形態により、フル・セット・アソシアティブ方式のキャッシュ装置において、データ処理装置２００のデータ処理装置とキャッシュ判定装置２８０のリオーダ回路で所望のウェイ数を分割して実装できるようになる。

【０１２９】

なお、図１２における各種の信号の名称や意味は、図３と図１０を用いて説明した実施形態１および２と同様であり、同じ信号に対する説明はここでは省略する。

【０１３０】

図１２に記載の３００のブロックは、図３および図１０のヒットカウント値算出回路３００に相当する。また図１２に記載の８００のブロックが実施形態５でのリオーダ回路であり、図３および図１０のリオーダ回路３２０に相当する部分である。また、制御回路である「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」８５０は実施形態１および２の制御回路である「Ｒｅｏｒｄｅｒ ｃｏｎｔｒｏｌｌｅｒ」３５０に相当する。

【０１３１】

まず、本実施形態のリオーダ回路８００には、リオーダ回路３２０と異なり、以下に示すM組の回路を備える。

・「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［０］、ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ［０］」８１０＿０から「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［ｍ－１］、ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ［ｍ－１］」８１０＿Ｍ－１に示すM個のリーダバッファ。

・８２０＿０から８２０＿Ｍ－１に示すM個の比較回路。

・「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［０］とインクリメンタ」８３０＿０から「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［ｍ－１］とインクリメンタ」８３０＿Ｍ－１に示すM個のヒットカウンタ。

【０１３２】

また、本実施形態のリオーダ回路８００には、リオーダ回路３２０と異なり、上記M組みの回路の処理結果を基にリオーダ回路８００を制御する、以下の回路を備える。

・バッファの空き状態、最小、もしくは最大となるヒットカウント値のリオーダバッファ番号を検知する「Ｈｉｔ ｃｏｕｎｔ ｄｅｔｅｃｔｏｒ」８４０

・交換機能を実現する「Ｒｅｏｒｄｅｒ ｄｅｃｏｄｅｒ」８６０。

【０１３３】

以下、図１２を用いて「破棄予定のキャッシュタグ／キャッシュデータの交換機能」と「破棄予定のキャッシュタグ／キャッシュデータの復帰機能」について説明する。どちらの機能も前述の実施形態１および２と同様に、入力された「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」が有効（アサート状態）のときに動作する。

【０１３４】

（破棄予定のキャッシュタグ／キャッシュデータの交換機能）

まず、M個の比較回路８２０＿０から８２０＿Ｍ－１で、入力された「ｄａｔａ（ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ）」とM個のリオーダバッファ８１０＿０から８１０＿Ｍ－１を比較する。１つでも同一のリオーダバッファが存在した場合、「ｒｅｏｒｄｅｒ＿ｃｏｎｔｒｏｌｌｅｒ」８５０は、「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」を無効（ディアサート状態）にする。そして、「ｔａｇ＿ｉｄ」をリオーダバッファの番号に変更する。

【０１３５】

前述の実施形態１および２では、リオーダバッファが１つであった。そのため、リオーダバッファとキャッシュメモリを切り替える信号「ｒｅｏｒｄｅｒ」を有効（アサート状態）にするだけでキャッシュデータの格納位置を特定できた。しかしながら本実施形態の

場合、複数リオーダバッファの格納位置をキャッシュメモリ調停部５８０に伝える必要がある。

【０１３６】

また、１つも同一となるリオーダバッファが存在しなかった場合は、以下に示す交換機能が動作する。いま、リオーダバッファの有効信号である「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［０］」８１０＿０から「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［ｍ－１］」８１０＿Ｍ－１の何れかが無効（ディアサート状態）のときを考える。この場合、「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ［０］」８１０＿０から「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ［ｍ－１］」８１０＿Ｍ－１のリオーダバッファの何れかは空き状態であることを示す。

【０１３７】

当然、リオーダ回路８００は無条件で空き状態のリオーダバッファ「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ［０］」８１０＿０から「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｄａｔａ［ｍ－１］」８１０＿Ｍ－１の何れかに破棄予定キャッシュタグ「ｓｗｔａｇ」を退避できる。そこで、「Ｈｉｔ ｃｏｕｎｔ ｄｅｔｅｃｔｏｒ」８４０は空き状態のリオーダバッファを探す。具体的には「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［０］」８１０＿０から「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［ｍ－１］」８１０＿Ｍ－１のうち値が無効（ディアサート状態）のものを１つ探す。そして、「Ｈｉｔ ｃｏｕｎｔ ｄｅｔｅｃｔｏｒ」８４０は検知結果８４２を出力する。

【０１３８】

次に「Ｍｉｎ ｈｉｔ ｃｏｕｎｔ ｄｅｃｏｄｅｒ」８６０は、この検知結果８４２を受け取り、入力された「ｓｗｅｅｐとｓｗｔａｇ」を選択された１つのリオーダバッファに退避させる。更に、破棄予定のヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」を、対応するヒットカウント値「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［０］」８３０＿０から「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［ｍ－１］」８３０＿Ｍ－１の何れかに退避させる。

【０１３９】

一方、リオーダバッファの有効信号である「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［０］」８１０＿０から「ｒｅｏｒｄｅｒ＿ｔａｇ＿ｖａｌｉｄ［ｍ－１］」８１０＿Ｍ－１がすべて有効（アサート状態）のときを考える。この場合、１つもリオーダバッファに空きがない状態である、交換すべきリオーダバッファを選択する必要がある。

【０１４０】

そこで「Ｈｉｔ ｃｏｕｎｔ ｄｅｔｅｃｔｏｒ」８４０は、「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［０］」８３０＿０から「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［ｍ－１］」８３０＿Ｍ－１のうち、ヒットカウント値が最小のリオーダバッファを探す。そして、前記「Ｈｉｔ ｃｏｕｎｔ ｄｅｔｅｃｔｏｒ」８４０は、検知結果８４２と「ｍｉｎ＿ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」８４３を出力する。そして破棄予定のヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」とこの「ｍｉｎ＿ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」８４３を比較回路「Ｃｏｍｐａｒｅ」８６２で比較する。

【０１４１】

破棄予定のヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」が大きい場合、「Ｒｅｏｒｄｅｒ ｄｅｃｏｄｅｒ」８６０は、入力された「ｓｗｅｅｐ、ｓｗｔａｇ、ｈｉｔ＿ｃｏｕｎｔ」を、検知結果８４２を基に選択されたリオーダバッファに上書きする。破棄予定のヒットカウント値「ｈｉｔ＿ｃｏｕｎｔ」が小さい場合、入力された「ｓｗｅｅｐ、ｓｗｔａｇ、ｈｉｔ＿ｃｏｕｎｔ」を破棄する。

【０１４２】

キャッシュメモリ調停部５８０ではM個のリオーダバッファに対応するキャッシュデータの格納領域を設け、上記のリオーダバッファの番号を示す「ｔａｇ＿ｉｄ」と「ｒｅｏｒｄｅｒ」と「ｅｘｃｈａｎｇｅ」に従い、キャッシュデータを更新／交換すればよい。

【０１４３】

上記のような仕組みで複数のリオーダバッファについて、破棄予定のキャッシュタグ／

キャッシュデータの交換機能が実現できる。

【０１４４】

（破棄予定のキャッシュタグ／キャッシュデータの復帰機能）

前記の交換機能と同様に、Ｍ個の比較回路８２０＿０から８２０＿Ｍ－１で、入力された「ｄａｔａ（ｍｉｓｓ＿ｈｉｔ＿ａｄｄｒｅｓｓ）」とＭ個のリオーダバッファ８１０＿０から８１０＿Ｍ－１を比較する。１つでも同一のリオーダバッファが存在した場合は、「ｒｅｏｒｄｅｒ＿ｃｏｎｔｒｏｌｌｅｒ」８５０は、「ｍｉｓｓ＿ｈｉｔ＿ｆｌａｇ」を無効（ディアサート状態）し、「ｔａｇ＿ｉｄ」をリオーダバッファの番号に変更する。

【０１４５】

一方、「Ｈｉｔ ｃｏｕｎｔ ｄｅｔｅｃｔｏｒ」８４０は、「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［０］」８３０＿０から「ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ［ｍ－１］」８３０＿Ｍ－１のうち、ヒットカウント値が最大のリオーダバッファを探す。そして前記「Ｈｉｔ ｃｏｕｎｔ ｄｅｔｅｃｔｏｒ」８４０は、検知結果８４２と「ｍａｘ＿ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」８４４を出力する。

【０１４６】

ここで、キャッシュヒットしたリオーダバッファと最大のヒットカウント値を持つリオーダバッファが同一の場合、復帰機能が動作する。検知結果８４２により、「ｒｅｏｒｄｅｒ＿ｃｏｎｔｒｏｌｌｅｒ」８５０は、駆動信号「ｓｈｉｆｔ」を有効（アサート状態）にし、「ｔａｇ＿ｄａｔａ」を第２パイプラインへ書き込む。また同時に、「ｍａｘ＿ｒｅｏｒｄｅｒ＿ｈｉｔ＿ｃｏｕｎｔ」８４４をヒットカウント値算出回路３００へ書き込む。また、「Ｒｅｏｒｄｅｒ ｄｅｃｏｄｅｒ」８６０は、入力された「ｓｗｅｅｐ、ｓｗｔａｇ、ｈｉｔ＿ｃｏｕｎｔ」を、検知結果８４２を基に選択されたリオーダバッファに上書きする。

【０１４７】

キャッシュメモリ調停部５８０ではＭ個のリオーダバッファに対応するキャッシュデータの格納領域を設け、上記のリオーダバッファの番号を示す「ｔａｇ＿ｉｄ」と「ｒｅｂｉｒｔｈ」と「ｅｘｃｈａｎｇｅ」に従い、キャッシュデータを更新／交換すればよい。

【０１４８】

上記のような仕組みで複数のリオーダバッファについて、破棄予定のキャッシュタグ／キャッシュデータの復帰機能が実現できる。

【０１４９】

＜実施形態６＞

近年のキャッシュ装置では、キャッシュメモリの一部を、値を書き換えない非キャッシュ領域に指定することができる。先にプリンタ画像処理では紙面の大半は白地であるケースを課題として挙げたが、この白地を印刷するために必要となる画像処理のデータは、非キャッシュ領域に割り当てた方が処理効率の高いのは自明である。

【０１５０】

そこで、図１０のデータ処理装置の第２パイプライン７６０のように、前述のヒット数カウンタとヒットカウント値を備える代わりに、キャッシュタグの破棄を防止するフラグ（１ビットのパイプライン・レジスタ）を具備する。そして新たに具備した破棄防止フラグを用いてキャッシュメモリの一部を識別してその部分を非キャッシュ領域に指定する機能を追加する。第２パイプラインの破棄防止フラグ(レジスタ)が有効（アサート状態）となるキャッシュタグの示すキャッシュメモリの格納先が非キャッシュ領域となる。

【０１５１】

まず、データ処理装置は、画像処理を開始する前に、予めキャッシュメモリの一部に必要となるデータ（前述の白地を印刷するために必要となる画像処理のデータ等）を記憶する。そして、データ処理装置の第２パイプラインのデータ要素として、記憶したデータのキャッシュメモリ上の格納先(格納アドレス)をｔａｇ＿ｄａｔａ［ｉ］に記憶し、対応する有効信号ｔａｇ＿ｖａｌｉｄ［ｉ］と破棄防止フラグを有効（アサート状態）に設定す

る。

【０１５２】

次にデータ処理装置は、画像処理を開始し、所望の処理を実行する。処理の中で、この防止フラグが有効（アサート状態）なキャッシュタグが第２パイプラインから破棄されたとき、データ処理装置は、リオーダバッファに一時的にこのキャッシュタグを退避後、前述の動作のように、必ずリオーダバッファのキャッシュタグをデータ処理装置の第２パイプラインに復帰させる。このような動作を行うことで、キャッシュ装置のキャッシュメモリの一部を、非キャッシュ領域に指定することができる。

【０１５３】

また、例えば、ＣＰＵが予めデータ処理装置に、メモリの特定領域を示すアドレス上限と下限を設定し、この範囲のメモリを非キャッシュ領域としてもよい。この場合、フラグの有効（アサート状態）化は、キャッシュミスアドレスが特定領域内のアドレスであれば、キャッシュ判定装置２８０が自動的に有効化してもよい。

【０１５４】

これによりキャッシュメモリの一部に、画像処理で必要とするデータを保持するための非キャッシュ領域（バッファ）として使用することができる（キャッシメモリ内に任意の大きさのバッファ領域を確保できる）。そして、画像処理の処理内容に応じて、キャッシュと非キャッシュ領域（バッファ）という２種類のデータ保持の形態を適切に配分し、より効率的なキャッシュ機構を実現できることとなる。

【０１５５】

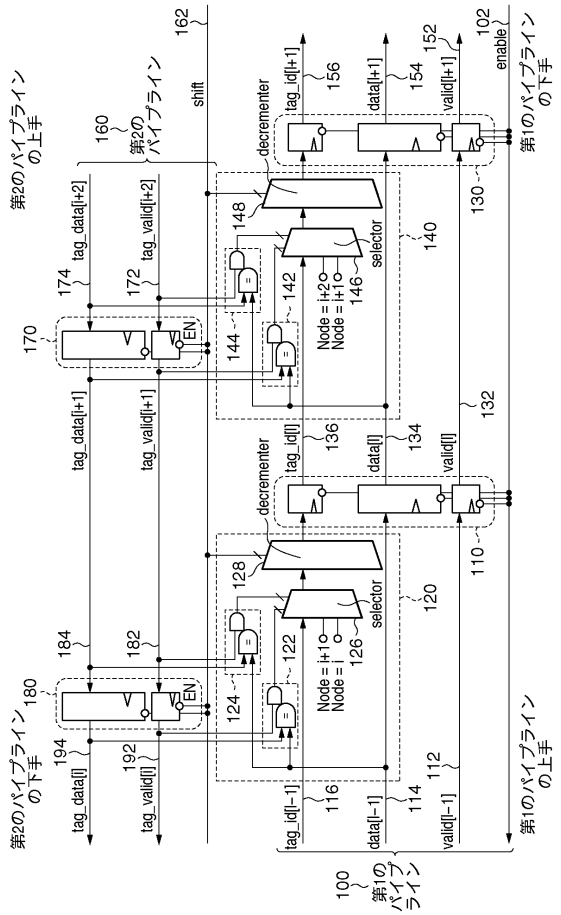
これまで説明したように、第２パイプラインのデータ要素にヒットカウント値などの統計情報やコンピュータ等の処理装置（ＣＰＵ、プロセッサ）から指定するフラグ等の属性情報を付加する。そして、付加した属性情報の示す優先順位に基づき、第２パイプラインに入力するデータ要素の破棄と復帰を制御する。これらの仕組みを通して、第２パイプライン上のデータ要素の優先順位を適応的に変更することが可能となる。

【０１５６】

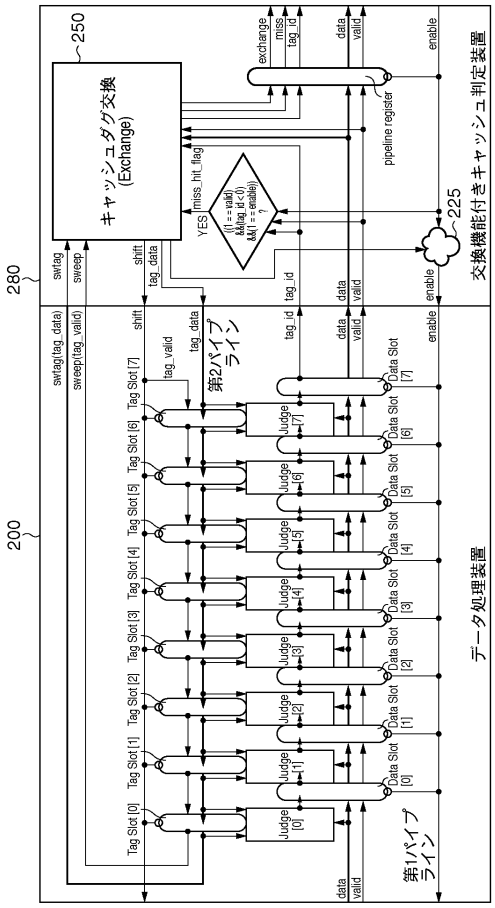
＜その他の実施形態＞

また、本発明は、以下の処理を実行することによっても実現される。即ち、上述した実施形態の機能を実現するソフトウェア（プログラム）を、ネットワーク又は各種記憶媒体を介してシステム或いは装置に供給し、そのシステム或いは装置のコンピュータ（またはＣＰＵやＭＰＵ等）がプログラムを読み出して実行する処理である。

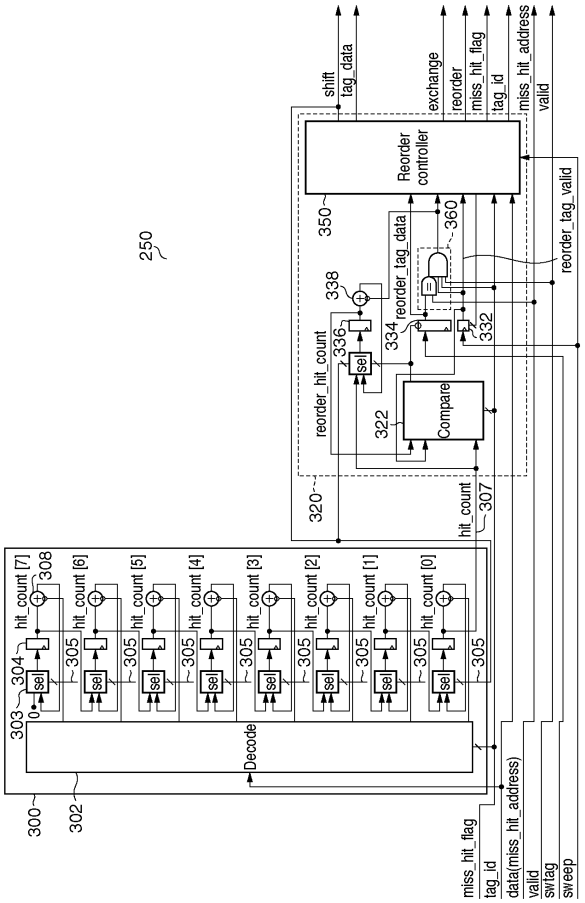
【図１】


第2のパイプラインの上手
第2のパイプライン
第1のパイプラインの下手
第2のパイプラインの下手
第1のパイプラインの上手
第1のパイプライン
shift
decrementer
selector
EN
Node = i+2
Node = i+1
Node = i
tag_id[i+1]
data[i+1]
valid[i+1]
tag_data[i+2]
tag_valid[i+2]
tag_data[i+1]
tag_valid[i+1]
tag_id[i]
data[i]
valid[i]
tag_data[i]
tag_valid[i]
tag_id[i−1]
data[i−1]
valid[i−1]
enable

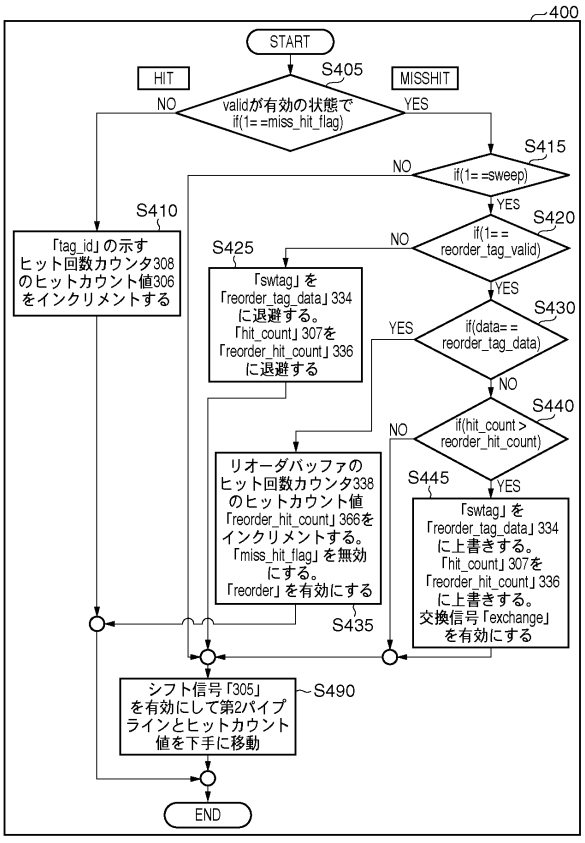
【図２】


キャッシュダグ交換
(Exchange)
交換機能付きキャッシュ判定装置
データ処理装置
第2パイプライン
第1パイプライン
pipeline register
Tag Slot
Data Slot
Judge
swtag
sweep
shift
tag_data
tag_id
data
valid
enable
exchange
miss
miss_hit_flag

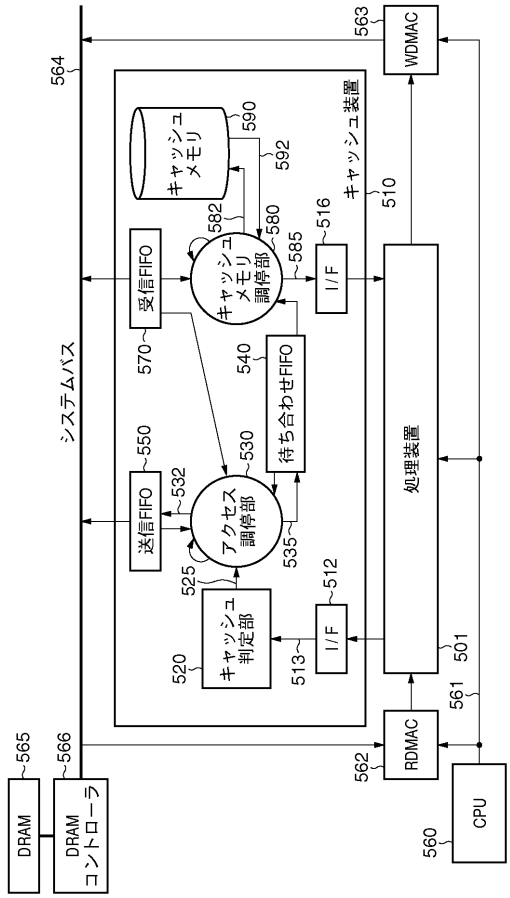
【図３】


Reorder controller
Compare
Decode
hit_count [7]
hit_count [6]
hit_count [5]
hit_count [4]
hit_count [3]
hit_count [2]
hit_count [1]
hit_count [0]
reorder_hit_count
reorder_tag_data
reorder_tag_valid
miss_hit_flag
tag_id
data(miss_hit_address)
valid
swtag
sweep
shift
tag_data
exchange
reorder
miss_hit_flag
tag_id
miss_hit_address
valid

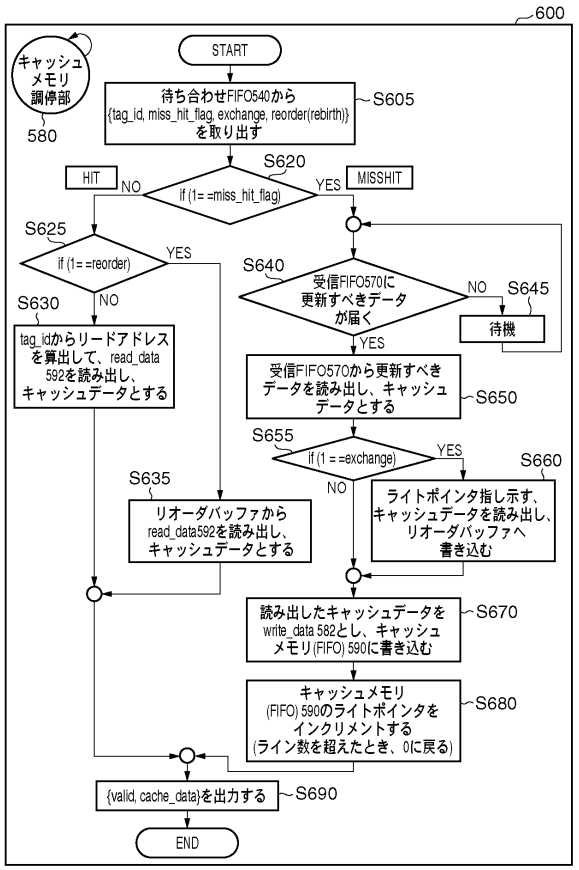
【図４】


START
HIT
MISSHIT
validが有効の状態で
if(1==miss_hit_flag)
NO
YES
if(1==sweep)
if(1==
reorder_tag_valid)
if(data==
reorder_tag_data)
if(hit_count>
reorder_hit_count)
「tag_id」の示す
ヒット回数カウンタ308
のヒットカウント値306
をインクリメントする
「swtag」を
「reorder_tag_data」334
に退避する。
「hit_count」307を
「reorder_hit_count」336
に退避する
リオーダバッファの
ヒット回数カウンタ338
のヒットカウント値
「reorder_hit_count」366を
インクリメントする。
「miss_hit_flag」を無効
にする。
「reorder」を有効にする
「swtag」を
「reorder_tag_data」334
に上書きする。
「hit_count」307を
「reorder_hit_count」336
に上書きする。
交換信号「exchange」
を有効にする
シフト信号「305」
を有効にして第2パイプ
ラインとヒットカウント
値を下手に移動
END

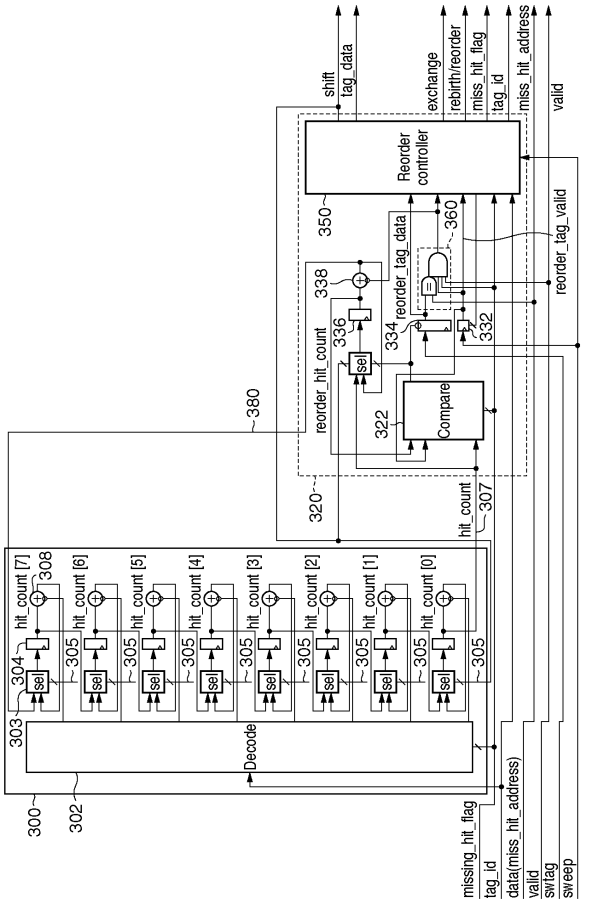
【図５】


システムバス
564
563
WDMAC
キャッシュ装置
510
キャッシュメモリ
590
592
582
580
516
I/F
585
受信FIFO
570
キャッシュメモリ調停部
540
待ち合わせFIFO
550
530
532
送信FIFO
アクセス調停部
535
525
512
キャッシュ判定部
520
513
I/F
処理装置
501
561
RDMAC
562
CPU
560
565
566
DRAM
DRAMコントローラ

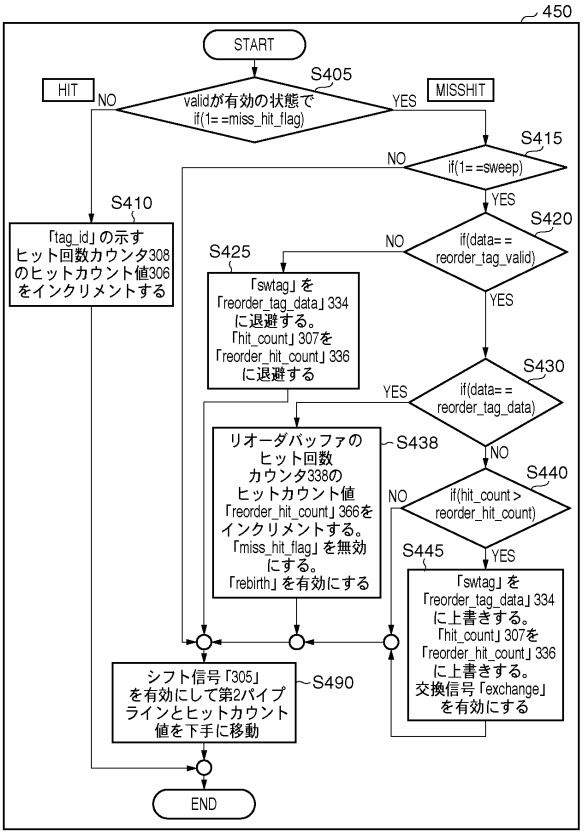
【図６】


600
キャッシュメモリ調停部
580
START
待ち合わせFIFO540から{tag_id, miss_hit_flag, exchange, reorder(rebirth)}を取り出す
S605
HIT
NO
if (1= =miss_hit_flag)
S620
YES
MISSHIT
S625
if (1= =reorder)
YES
NO
S630
tag_idからリードアドレスを算出して、read_data 592を読み出し、キャッシュデータとする
S640
受信FIFO570に更新すべきデータが届く
NO
S645
待機
YES
受信FIFO570から更新すべきデータを読み出し、キャッシュデータとする
S650
S655
if (1 = =exchange)
YES
NO
S660
ライトポインタ指し示す、キャッシュデータを読み出し、リオーダバッファへ書き込む
S635
リオーダバッファからread_data592を読み出し、キャッシュデータとする
読み出したキャッシュデータをwrite_data 582とし、キャッシュメモリ(FIFO) 590に書き込む
S670
キャッシュメモリ(FIFO) 590のライトポインタをインクリメントする(ライン数を超えたとき、0に戻る)
S680
{valid, cache_data}を出力する
S690
END

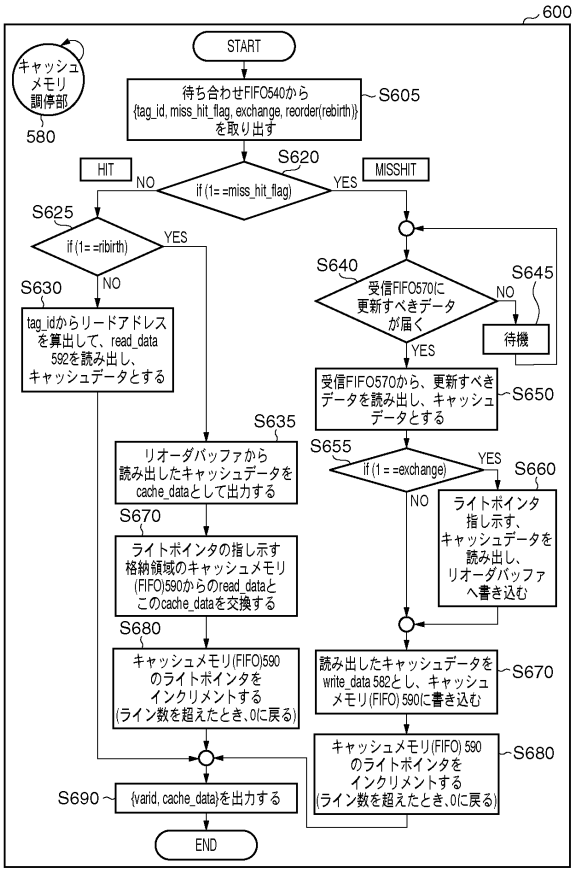
【図７】


shift
tag_data
exchange
rebirth/reorder
miss_hit_flag
tag_id
miss_hit_address
valid
Reorder controller
350
360
reorder_tag_data
reorder_tag_valid
338
336
334
332
reorder_hit_count
sel
Compare
322
380
320
hit_count
307
hit_count [7]
hit_count [6]
hit_count [5]
hit_count [4]
hit_count [3]
hit_count [2]
hit_count [1]
hit_count [0]
308
304
305
303
Decode
300
302
missing_hit_flag
tag_id
data(miss_hit_address)
valid
swtag
sweep

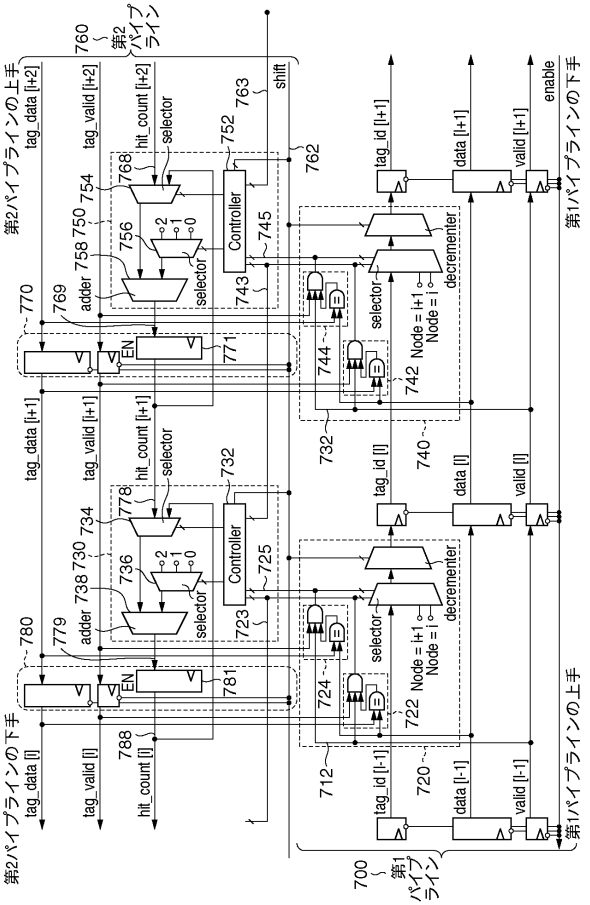
【図８】


450
START
HIT
NO
validが有効の状態で if(1= =miss_hit_flag)
S405
YES
MISSHIT
NO
if(1= =sweep)
S415
YES
S410
「tag_id」の示すヒット回数カウンタ308のヒットカウント値306をインクリメントする
NO
if(data= = reorder_tag_valid)
S420
S425
「swtag」を「reorder_tag_data」334に退避する。「hit_count」307を「reorder_hit_count」336に退避する
YES
YES
if(data= = reorder_tag_data)
S430
リオーダバッファのヒット回数カウンタ338のヒットカウント値「reorder_hit_count」366をインクリメントする。「miss_hit_flag」を無効にする。「rebirth」を有効にする
S438
NO
NO
if(hit_count > reorder_hit_count)
S440
YES
S445
「swtag」を「reorder_tag_data」334に上書きする。「hit_count」307を「reorder_hit_count」336に上書きする。交換信号「exchange」を有効にする
シフト信号「305」を有効にして第2パイプラインとヒットカウント値を下手に移動
S490
END

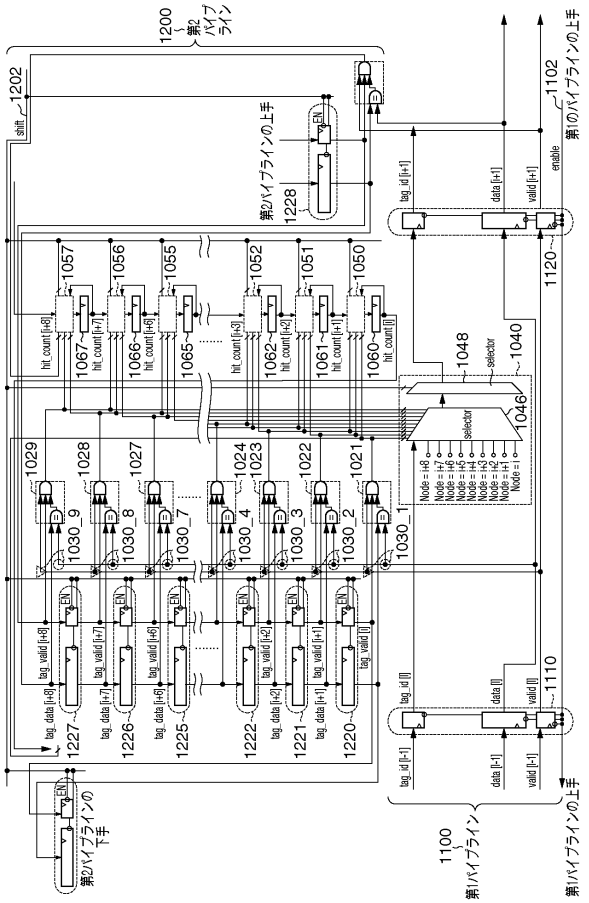
【図９】



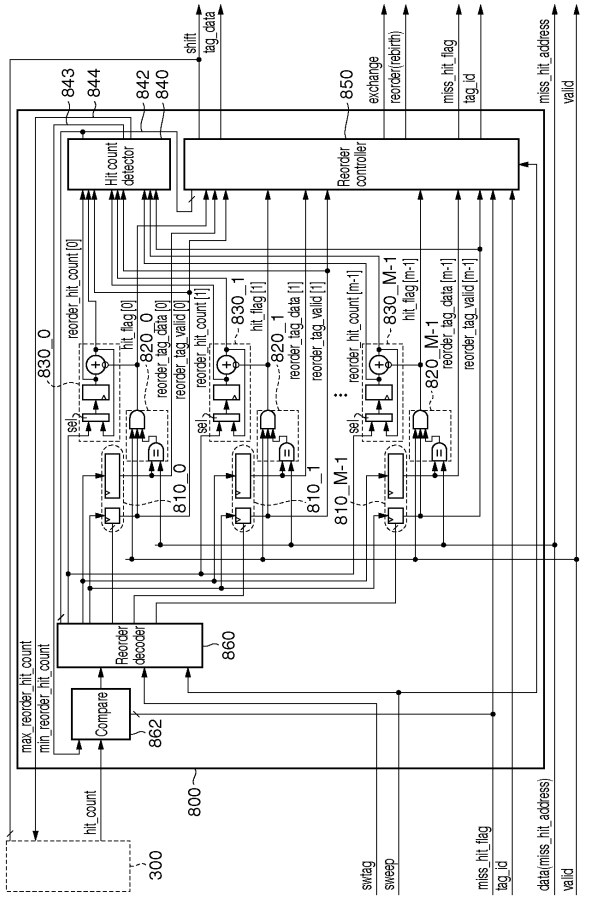
【図１０】



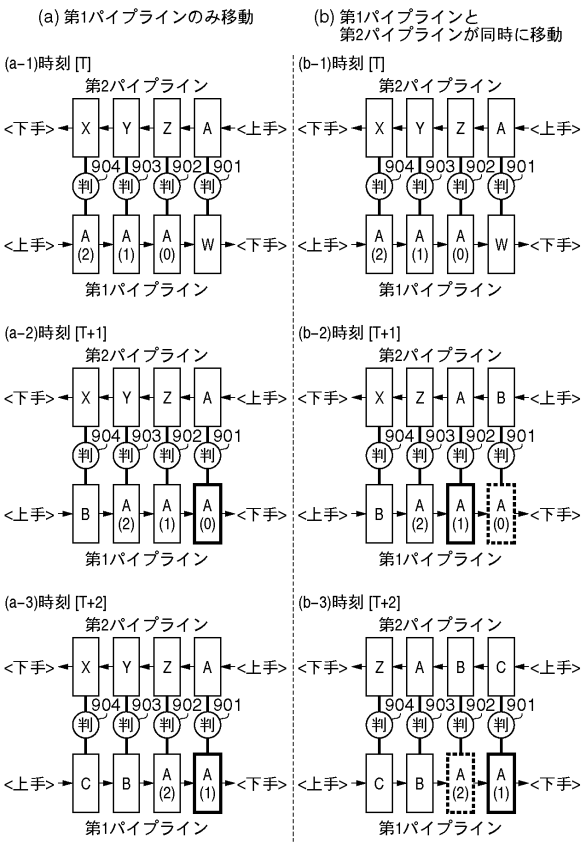
【図１１】



【図１２】



【図１３】


(a) 第1パイプラインのみ移動
(b) 第1パイプラインと
第2パイプラインが同時に移動
(a-1)時刻 [T]
(b-1)時刻 [T]
(a-2)時刻 [T+1]
(b-2)時刻 [T+1]
(a-3)時刻 [T+2]
(b-3)時刻 [T+2]
第2パイプライン
第1パイプライン
<下手>
<上手>
判
904
903
902
901
X
Y
Z
A
B
C
W
(2)
(1)
(0)

フロントページの続き

(72)発明者　伊藤　忠幸
東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

審査官　三坂　敏夫

(56)参考文献　特開平０４－１８４５３３（ＪＰ，Ａ）
特開平０４－１７８７５５（ＪＰ，Ａ）
特表２００１－５１４７８０（ＪＰ，Ａ）
米国特許第０６２４７１１５（ＵＳ，Ｂ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／３８
Ｇ０６Ｆ　１２／０８