(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3592640号
(P3592640)
(45)発行日　平成16年11月24日(2004.11.24)　(24)登録日　平成16年9月3日(2004.9.3)

(51)Int.Cl.[7]	F I		
G06F 3/06	G06F	3/06	301K
G06F 12/00	G06F	3/06	540
G06F 13/10	G06F	12/00	514A
	G06F	13/10	340B

請求項の数 28　(全 29 頁)

(21)出願番号　特願2001-1137(P2001-1137)
(22)出願日　平成13年1月9日(2001.1.9)
(65)公開番号　特開2002-207572(P2002-207572A)
(43)公開日　平成14年7月26日(2002.7.26)
審査請求日　平成13年1月9日(2001.1.9)

(73)特許権者　000003078
株式会社東芝
東京都港区芝浦一丁目１番１号
(74)代理人　100083161
弁理士　外川　英明
(72)発明者　水野　聡
東京都青梅市末広町２丁目９番地　株式会社東芝　青梅工場内

審査官　藤井　浩

最終頁に続く

(54)【発明の名称】ディスク制御システムおよびディスク制御方法

(57)【特許請求の範囲】
【請求項１】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、
前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、
前記ライトリクエストに応答して、前記複数のディスク上に設けられた管理データ領域に、前記上位ファイルシステムからの前記論理アドレスを論理アドレスログ情報として書き込む手段と、
前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知する手段とを具備することを特徴とするディスク制御システム。
【請求項２】
前記上位ファイルシステムからの前記ライトリクエストによって転送された複数のデータブロックに対して、１つのライトリクエストとしてのＩ／Ｏ要求を発行して、前記複数のデータブロックを同時に前記書き込み対象データストライプの空き領域に書き込むことを特徴とする請求項１記載のディスク制御システム。

【請求項３】
前記論理アドレスログ情報は、前記管理データ領域に１つ又は複数個設けられた論理アドレスログ領域に書き込まれることを特徴とする請求項１記載のディスク制御システム。
【請求項４】
前記複数個の論理アドレスログ領域で構成した場合、現在使っていないもので一番最近使用した論理アドレスログ領域を使用することを特徴とする請求項３記載のディスク制御システム。
【請求項５】
前記論理アドレスログ領域は、前記ディスク上に固定的に設けられていることを特徴とする請求項３記載のディスク制御システム。
【請求項６】
前記データ書き込み処理が完了しない状態で、システムダウンを起した場合、前記論理アドレスログ領域に記録した前記論理アドレスログ情報を用いてシステムリブート後のデータ回復を行なうことを特徴とする請求項３記載のディスク制御システム。
【請求項７】
前記管理データ領域には、書き込みが行われたストライプＩＤ、およびこれから書き込みが行われるストライプＩＤを記録するストライプＩＤログ領域を更に有することを特徴とする請求項１記載のディスク制御システム。
【請求項８】
前記書き込み対象データストライプの一部をタグ情報の書き込み領域として用い、前記書き込み対象データストライプの全領域のデータが揃った時の前記論理アドレスログ領域に書き込まれた論理アドレス情報が前記タグ情報として書き込まれることを特徴とする請求項１記載のディスク制御システム。
【請求項９】
主メモリの一部に書き込みデータに関する論理アドレスと物理アドレスの対応を記憶するアドレス変換テーブルキャッシュを設け、前記書き込み対象データストライプへのデータおよびタグ情報の書き込みが終了した時、前記アドレス変換テーブルキャッシュを前記タグ情報を基に更新することを特徴する請求項８記載のディスク制御システム。
【請求項１０】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、
主メモリ上に設けられたライトバッファに、複数のライトリクエストに対応する複数のブロックデータを書き込む手段と、
前記複数のライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に、前記ライトバッファに記憶した複数のブロックデータを一括ライトするデータ書き込み手段と、
前記複数のブロックデータに対応する前記上位ファイルシステムからの前記論理アドレスを論理アドレスログ情報として、前記複数のディスク上に設けられた管理データ領域に一括ライトするログ書き込み手段と、
前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに完了を通知する手段とを具備することを特徴とするディスク制御システム。
【請求項１１】
前記ライトリクエストの数を変数として管理し、前記変数が所定値以上の場合は、前記ライトリクエストをペンディングリクエストとして保留し前記データおよびログ書き込み手段による一括ライト処理を実行し、前記変数が所定値より小さい場合は、前記ライトリクエスト単位の書き込み処理を実行することを特徴とする請求１０記載のディスク制御シス

テム。
【請求項１２】
前記一括ライト処理の実行判断は、前記ペンディングリクエストのリクエスト数が所定値を越えた時、または前記ペンディングリクエストのデータサイズ合計が所定値を越えた時、または前記ペンディングリクエストのデータサイズ合計が前記書き込み対象データストライプの残りの空き領域を越えた時、または前記ペンディングしている時間が所定時間経過した時に行なわれることを特徴とする請求項１１記載のディスク制御システム。
【請求項１３】
前記論理アドレスログ情報は、前記管理データ領域に１つ又は複数個設けられた論理アドレスログ領域に書き込まれることを特徴とする請求項１０記載のディスク制御システム。
【請求項１４】
前記複数個の論理アドレスログ領域で構成した場合、現在使っていないもので一番最近使用した論理アドレスログ領域を使用することを特徴とする請求項１０記載のディスク制御システム。
【請求項１５】
前記論理アドレスログ領域は、前記ディスク上に固定的に設けられていることを特徴とする請求項１０記載のディスク制御システム。
【請求項１６】
前記データ書き込み処理が完了しない状態で、システムダウンを起した場合、前記論理アドレスログ領域に記録した前記論理アドレスログ情報を用いてシステムリブート後のデータ回復を行うことを特徴とする請求項１５記載のディスク制御システム。
【請求項１７】
前記管理データ領域には、書き込みが行われたストライプＩＤ、およびこれから書き込みが行われるストライプＩＤを記録するストライプＩＤログ領域を更に有することを特徴とする請求項１０記載のディスク制御システム。
【請求項１８】
前記書き込み対象データストライプの一部をタグ情報の書き込み領域として用い、前記書き込み対象データストライプの全領域のデータ揃った時の前記論理アドレスログ領域に書き込まれた論理アドレス情報が前記タグ情報として書き込まれることを特徴とする請求項１０記載のディスク制御システム。
【請求項１９】
主メモリの一部に書き込みデータに関する論理アドレスと物理アドレスの対応を記憶するアドレス変換テーブルキャッシュを設け、前記書き込み対象データストライプへのデータおよびタグ情報の書き込みが終了した時、前記アドレス変換テーブルキャッシュを前記タグ情報を基に更新することを特徴する請求項１８記載のディスク制御システム。
【請求項２０】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、
前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、
前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域に、前記上位ファイルシステムからの前記論理アドレス、および書き込みデータサイズ、および書き込みデータのチェックサムを論理アドレスログ情報として書き込む手段と、
前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知する手段とを具備することを特徴とするディスク制御システム。

【請求項２１】
書き込み中にシステムの障害が発生した場合には、書込み処理途中のストライプに関して、その前記論理アドレスログ領域に書き込まれているチェックサム値が前記データ領域の書き込みデータから求めたチェックサム値との一致を確認し、一致した時には、そのデータは有効なデータとして取り扱い、一致しなかった時には書き込みが完了しなかったとみなし、そのデータを捨てることを特徴とする請求項２０記載のディスク制御システム。
【請求項２２】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、
前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域に、前記上位ファイルシステムからの前記論理アドレス、書き込みデータサイズ、書き込みデータのチェックサム、および少なくとも１データブロックを論理アドレスログ情報として書き込む手段と、
前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、
前記上位ファイルシステムからのライトリクエストに対して、前記論理アドレスログ領域への前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知する手段とを具備することを特徴とするディスク制御システム。
【請求項２３】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、
前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域のヘッダ部に有効／無効を示すフラグ、ストライプＩＤ番号、および最終データの書き込みタイムスタンプを記録し、前記論理アドレスログ領域のエントリ部に書き込みリクエストによって処理される１つ又は複数のブロックデータのタイムスタンプ、１つ又は複数の論理アドレス、および１つ又は複数のチェックサムを論理アドレスログ情報として書き込む手段と、
前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、
書き込み中にシステムの障害が発生した場合には、書込み処理途中の有効なフラグがセットされているストライプに関して、その前記論理アドレスログ領域に書き込まれているチェックサム値が前記データ領域の書き込みデータから求めたチェックサム値との一致を確認し、一致した時には、そのデータは有効なデータとして取り扱い、一致しなかった時には書き込みが完了しなかったとみなし、そのデータを捨てることを特徴とするディスク制御システム。
【請求項２４】
前記チェックサムが一致した場合、前記データストライプのＴＡＧ領域に物理アドレスおよび論理アドレスの対応表を記録し、そのＴＡＧ領域の対応表を主メモリのアドレス変換テーブルキャッシュに反映させることを特徴とする請求項２３記載のディスク制御システム。
【請求項２５】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込

み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御方法において、
前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込み、
前記ライトリクエストに応答して、前記複数のディスク上に設けられた管理データ領域に、前記上位ファイルシステムからの前記論理アドレスを論理アドレスログ情報として書き込み、
前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知することを特徴とするディスク制御方法。

【請求項２６】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御方法において、
主メモリ上に設けられたライトバッファに、複数のライトリクエストに対応する複数のブロックデータを書き込み、
前記複数のライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に、前記ライトバッファに記憶した複数のブロックデータを一括ライトするデータ書き込み、
前記複数のブロックデータに対応する前記上位ファイルシステムからの前記論理アドレスを論理アドレスログ情報として、前記複数のディスク上に設けられた管理データ領域に一括ライトするログ書き込み、
前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに完了を通知することを特徴とするディスク制御方法。

【請求項２７】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御方法において、
前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込み、
前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域に、前記上位ファイルシステムからの前記論理アドレス、および書き込みデータサイズ、および書き込みデータのチェックサムを論理アドレスログ情報として書き込み、
前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知することを特徴とするディスク制御方法。

【請求項２８】
論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御方法において、
前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域のヘッダ部に有効／無効を示すフラグ、ストライプＩＤ番号、および最終データの書き込みタイムスタンプを記録し、前記論理アドレスログ領域のエントリ部に書き込みリク

エストによって処理される１つ又は複数のブロックデータのタイムスタンプ、１つ又は複数の論理アドレス、および１つ又は複数のチェックサムを論理アドレスログ情報として書き込み、
前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込み、
書き込み中にシステムの障害が発生した場合には、書込み処理途中の有効なフラグがセットされているストライプに関して、その前記論理アドレスログ領域に書き込まれているチェックサム値が前記データ領域の書き込みデータから求めたチェックサム値との一致を確認し、一致した時には、そのデータは有効なデータとして取り扱い、一致しなかった時には書き込みが完了しなかったとみなし、そのデータを捨てることを特徴とするディスク制御方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、ディスク制御システムおよびディスク制御方法に関し、特に、ＲＡＩＤ制御装置に用いられていた書き込みバッファやアドレス変換テーブルなどを構成する専用の不揮発性メモリを使用しないで、主メモリとディスク装置だけを用いて、ディスク上のストライプに連続的にライトリクエストのデータを書き込むディスク制御システムおよびディスク制御方法に関するものである。

【０００２】

【従来の技術】

ＲＡＩＤ構成のディスク制御システムのライト方式は、例えば特開平６－２１４７２０号公報や、特開平１１－５３２３５号公報に示すように、ランダムライトを１つの連続した領域としての物理ストライプにまとめて書き込む方式が提案されている。しかしながら、従来技術では、書き込むデータを一度不揮発性メモリ（以下、ＮｖＲＡＭと称する）又は揮発性メモリに書き込み、１ストライプ分の書込みデータが揃った時点で、ディスク装置上の物理ストライプに書込みを行っていた。また、上位ファイルシステムからの論理アドレスをディスク上の物理アドレスに変換したアドレス変換テーブルを上記ＮｖＲＡＭに記憶させていた。

【０００３】

この様に、一度、ＮｖＲＡＭにデータを書き込むことにより、ディスク装置上に書き込む前にシステムが異常停止したとしても、システムリブート後にＮｖＲＡＭを参照し、ＮｖＲＡＭ上の書きかけのデータを正しくディスク装置に書き込むことにより、書き込みを完了させることができる（書き込みデータを失わない）効果が有る。

【０００４】

即ち、ＮｖＲＡＭへのデータ書き込みが終れば、そのライトデータが失われることはない。このため、従来のディスク制御システムでは、書き込みデータをＮｖＲＡＭに書き込んだ時点で、そのライトリクエストに対して「完了」をホストに通知していた。

【０００５】

【発明が解決しようとする課題】

しかしながら、ＮｖＲＡＭはシステムに装着するＩ／Ｏカード（ハードウェア）で提供する必要が有り、その分コストがかかるという問題があり、また、ホスト計算機との相性の問題、互換性、保守等の手間もかかる等の問題がある。

【０００６】

本発明では、ＮｖＲＡＭを使わずに、主メモリとディスク装置だけを用いて、ディスク上のストライプに連続的にライトリクエストのデータを書き込むディスク制御システムおよびディスク制御方法を提供することを目的とする。また、ＮｖＲＡＭを使用せずとも異常停止時にデータを損失することが無く、システムの性能向上を実現させるものである。

【０００７】

【課題を解決するための手段】

上記目的を達成するために、発明のディスク制御システムは、論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、前記ライトリクエストに応答して、前記複数のディスク上に設けられた管理データ領域に、前記上位ファイルシステムからの前記論理アドレスを論理アドレスログ情報として書き込む手段と、前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知する手段とを具備することを特徴とする。

【０００８】

本発明によれば、不揮発性メモリに代わりに、ディスク上に設けられた管理データ領域１８ｂ、データ領域１８ｃなどを用いることにより、ランダムライトが物理的に連続した領域への書き込みとして実行することが出来る。したがって、シーク／回転待ちが少なくなり、不揮発性メモリを使用しなくても高速な書き込み処理が実現される。

【０００９】

また、本発明のディスク制御システムは、論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、主メモリ上に設けられたライトバッファに、複数のライトリクエストに対応する複数のブロックデータを書き込む手段と、前記複数のライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に、前記ライトバッファに記憶した複数のブロックデータを一括ライトするデータ書き込み手段と、前記複数のブロックデータに対応する前記上位ファイルシステムからの前記論理アドレスを論理アドレスログ情報として、前記複数のディスク上に設けられた管理データ領域に一括ライトするログ書き込み手段と、前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに完了を通知する手段とを具備することを特徴とする。

【００１０】

本発明によれば、不揮発性メモリに代わりに、ディスク上に設けられた管理データ領域１８ｂ、データ領域１８ｃなどを用いることにより、ランダムライトが物理的に連続した領域への書き込みとして実行することが出来る。また、複数のライトリクエストが１度のデータライト処理と１度の論理アドレスログ情報の書き込みとして処理することができるため、書き込み処理回数が減り、書き込みサイズが大きくなることでディスクへのトータルとして書き込みオーバヘッドが小さくなり、結果的に書き込み処理のスループットが向上する。

【００１１】

また、本発明のディスク制御システムは、論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域に、前記上位ファイルシステムからの前記論理アドレス、および書き込みデータサイズ、および書き

込みデータのチェックサムを論理アドレスログ情報として書き込む手段と、前記上位ファイルシステムからのライトリクエストに対して、前記データと前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知する手段とを具備することを特徴とする。

【００１２】

本発明によれば、ディスクのデータ領域へのデータ書き込み中にシステムの障害が発生した場合には、書き込み処理途中のストライプに関して、その論理アドレスログ領域に書き込まれているチェックサム値とデータ領域のデータから求めたチェックサム値とが一致するか否かを確認チェックすることにより、一致している場合には、データ書き込みが完了していたと判断し、データを有効とみなし残りの処理（アドレス変換テーブルへの登録等）を行ない、効率的な障害回復処理が実行できる。

【００１３】

また、本発明のディスク制御システムは、論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域に、前記上位ファイルシステムからの前記論理アドレス、書き込みデータサイズ、書き込みデータのチェックサム、および少なくとも１データブロックを論理アドレスログ情報として書き込む手段と、前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、前記上位ファイルシステムからのライトリクエストに対して、前記論理アドレスログ領域への前記論理アドレスログ情報の書き込み完了をもって、上位ファイルシステムに書き込み完了を通知する手段とを具備することを特徴とする。

【００１４】

本発明によれば、個々のライトリクエストに対して、論理アドレスログ領域に論理アドレスログ情報とライトデータａの書き込みが完了すれば、その時点でＯＳファイルシステムに対して完了通知することができる。つまり、ＯＳファイルシステムに対するレスポンスが早くなる。

【００１５】

また、本発明のディスク制御システムは、論理アドレスを用いた少なくとも１つのデータブロックからなるデータの書き込みを行うための上位ファイルシステムから出力されるライトリクエストに応答して、論理アドレスから物理アドレスへのアドレス変換を行い、複数のディスク装置により構成される書き込み領域であるデータストライプに対して連続的にライトリクエストのデータを書いていくディスク制御システムにおいて、前記ライトリクエストに応答して、前記複数のディスク上に設けられた論理アドレスログ領域のヘッダ部に有効／無効を示すフラグ、ストライプＩＤ番号、および最終データの書き込みタイムスタンプを記録し、前記論理アドレスログ領域のエントリ部に書き込みリクエストによって処理される１つ又は複数のブロックデータのタイムスタンプ、１つ又は複数の論理アドレス、および１つ又は複数のチェックサムを論理アドレスログ情報として書き込む手段と、前記ライトリクエストに応答して、前記複数のディスク上に設けられたデータ領域の割り当てられた書き込み対象データストライプの空き領域に前記データを書き込む手段と、書き込み中にシステムの障害が発生した場合には、書込み処理途中の有効なフラグがセットされているストライプに関して、その前記論理アドレスログ領域に書き込まれているチェックサム値が前記データ領域の書き込みデータから求めたチェックサム値との一致を確認し、一致した時には、そのデータは有効なデータとして取り扱い、一致しなかった時には書き込みが完了しなかったとみなし、そのデータを捨てることを特徴とする。

【００１６】

本発明によれば、書き込み中にシステムの障害が発生した場合には、書込み処理途中の有

効なフラグがセットされているストライプに関して、その論理アドレスログ領域に書き込まれているチェックサム値がデータ領域の書き込みデータから求めたチェックサム値との一致を確認し、一致した時には、そのデータは有効なデータとして取り扱い、一致しなかった時には書き込みが完了しなかったとみなし捨てることにより、障害回復処理を効率よく実行することが出来る。

【００１７】

【発明の実施の形態】

先ず、本発明で用いられる各種の用語について説明する。

（データストライプ）

ディスク制御システムが「まとめ書き」（又は「一括書き」とも言われる）するデータの単位を意味する。ディスクのパーティション上で、一続きの領域となっており、ＲＡＩＤコントローラが管理するストライプサイズの整数倍のサイズである。ＲＡＩＤ５に対しては、このサイズをパリティグループにする事により、いわゆる「ＲＡＩＤ５のライトペナルティ」を改善する事ができ、大幅な性能向上を実現できる。ストライプは複数のデータブロックから構成される。

【００１８】

（ストライプ番号）

パーティション上に並んでいる物理ストライプの通し番号を意味する。

【００１９】

（論理ブロック番号＝論理アドレス番号）

「論理ブロック番号」とは、上位ファイルシステムから見た時のディスクパーティション上のデータブロック番号である。ディスク制御システム方式では、上位ファイルシステムがアクセスを要求する時のブロック番号は、論理ブロック番号であり、仮想的なブロック番号である。この論理ブロック番号は、ディスク制御システムが管理する「アドレス変換テーブル」によって物理ブロック番号（物理パーティション上の物理的なブロック番号）と対応づけられる。（物理ブロック番号）×（ブロックサイズ［バイト］）よりパーティション上でのバイトオフセット値（アドレス）が求まる。

【００２０】

（アドレス変換テーブル：ＡＭＴ（Ａｄｄｒｅｓｓ　Ｍａｐｐｉｎｇ　Ｔａｂｌｅ））

ディスク制御システム方式では、上位ファイルシステムがアクセスを要求する時のブロック番号は、論理ブロック番号であり、仮想的なブロック番号である。この論理ブロック番号はディスク制御システムが管理する「アドレス変換テーブル」によって物理ブロック番号（物理パーティション上の物理的なブロック番号）と対応づけられる。本発明の実施形態では、アドレス変換テーブルは、対象パーティション上に有り、データ部と別の領域が割り当てられる。アドレス変換テーブルは、論理ブロック番号に対する物理ブロック番号が登録されたれたテーブルである。新たにデータブロックを書き込む時には、相当する論理ブロック番号のエントリに、そのブロックを書き込む物理ブロック番号（＝ディスク上のアドレス）を登録する。また、データブロックを参照する際には、その論理アドレスをインデックスとするエントリの値を求め、その値をディスク上の物理ブロック番号として実際のアドレスを求め参照する。

【００２１】

以下、図面を参照して本発明の実施形態について説明する。

図１には、本発明の一実施形態に係るディスク制御システムを適用した計算機システムの構成が示されている。この計算機システムは、例えばサーバ（ＰＣサーバ）として利用されるものであり、複数のＣＰＵ（＃１、＃２、＃Ｎ）１１を搭載することが出来る。これら各ＣＰＵ１１は図示のようにプロセッサバス１を介してブリッジ１２に接続されている。ブリッジ１２はプロセッサバス１とＰＣＩバス２を双方向で接続するためのブリッジＬＳＩであり、ここには主メモリ１３を制御するためのメモリコントローラも内蔵されている。主メモリ１３には、オペレーティングシステム（ＯＳ）、実行対象のアプリケーションプログラム、およびドライバなどがロードされている。また、本発明では、主メモリ１

３のドライバ作業領域１７に、ライトバッファ（まとめ書きバッファ）１７１、アドレス変換テーブル（以下、ＡＭＴキャッシュと称する）１７２、ストライプ管理テーブル１７３が設けられている。

【００２２】

ライトバッファ１７１は、書き込みデータブロックを蓄積するためのデータバッファであり、このライトバッファ１７１に例えば１物理ストライプ分の書き込みデータブロックが蓄積された時点でディスクアレイ１８への一括書き込みが開始されるが、本発明においてはそれに限定されるものではない。まとめ書きを行なう場合、物理ストライプは「まとめ書き」の単位であり、ディスクアレイ１８の全記憶領域に形成されたパーティション上で一続きの連続した領域から構成される。各物理ストライプは、ＲＡＩＤコントローラ１６が管理するストライプユニットの整数倍のサイズである。

【００２３】

ＡＭＴキャッシュ１７２は、論理アドレス空間を構成する複数の論理ブロック番号それぞれと、それら論理ブロック番号で指定されるデータブロックが存在するディスクアレイ１８上の物理位置を示す物理ブロック番号との対応関係を示すアドレス変換情報を記憶する。ＯＳファイルシステムがアクセスを要求するときのブロック番号は論理ブロック番号であり、これは仮想的な論理アドレスである。この論理ブロック番号はＡＭＴキャッシュ１７２によって物理ブロック番号（ディスクパーティション上の物理的なブロック番号）と対応づけられる。また、物理ブロック番号×ブロックサイズ（バイト）によって、ディスクパーティションの先頭位置からのバイトオフセット値が求められる。

【００２４】

また、ＡＭＴキャッシュ１７２は、論理ブロック番号それぞれに対応する複数のエントリを有している。新たにデータブロックを書き込むときには、書き込み要求された論理ブロック番号に対応するエントリには、そのデータブロックの実際に書き込まれる物理ブロック番号（物理アドレス）が登録される。データブロックの読み出し時には、読み出し要求された論理ブロック番号に対応するエントリから物理ブロック番号が調べられ、その物理ブロック番号（物理ブロック番号×ブロックサイズ）で指定されるディスクパーティション上の物理位置からデータブロックが読み出される。

【００２５】

ストライプ管理テーブル１７３は、各論理ストライプに関する情報を管理するためのものであり、それら各論理ストライプの情報はデータ再配置のための処理などに利用される。

【００２６】

ＰＣＩバス２には、ＲＡＩＤコントローラ１６が接続されている。ＲＡＩＤコントローラ１６によって制御される複数のディスク装置で構成されるディスクアレイ１８は、各種ユーザデータの記録等に用いられる。

【００２７】

ディスクアレイ１８は、ＲＡＩＤコントローラ１６の制御により、例えばＲＡＩＤ５構成のディスクアレイとして機能する。この場合、ディスクアレイ１８は、データ記憶用のＮ台のディスク装置にパリティ記憶用のディスク装置を１台付加したＮ＋１台（ここでは、ＤＩＳＫ０～ＤＩＳＫ４の５台）のディスク装置から構成される。これらＮ＋１台のディスク装置はグループ化され、単一の論理ディスクドライブとして使用される。

【００２８】

グループ化されたディスク装置には、図示のように、データ（ＤＡＴＡ）とそのパリティ（ＰＡＲＩＴＹ）とから構成される物理ストライプ（パリティグループ）が割り当てられ、かつ、パリティ位置は物理ストライプ毎にＮ＋１台のディスク装置間で順に移動される。例えば、物理ストライプＳ０では、ＤＩＳＫ０～ＤＩＳＫ３それぞれの同一位置に割り当てられたストライプ上のデータ（ＤＡＴＡ）群のパリティ（ＰＡＲＩＴＹ）は、ＤＩＳＫ４の対応するストライプ上に記録される。また、物理ストライプＳ１においては、そのストライプＳ１のデータに対応するパリティ（ＰＡＲＩＴＹ）は、ＤＩＳＫ３の対応するストライプ上に記録される。このようにパリティを物理ストライプ単位でＮ＋１台のディ

スク装置に分散させることによって、パリティディスクへのアクセス集中を防止するように構成されている。

【００２９】

主メモリ１３のドライバ作業領域１７は、ＲＡＩＤ高速化ドライバの実現のために使用される作業領域である。ＲＡＩＤ高速化ドライバは、ディスクアレイ１８に対する書き込み性能を向上させるために用いられる。本実施形態においては、ＯＳファイルシステム５０には改良を加えず、ＯＳに組み込んで用いられるドライバプログラムとドライバ作業領域１７とによってＲＡＩＤ高速化ドライバが実現されている。ここで、図２を参照して、ＲＡＩＤ高速化ドライバによるディスクアレイ１８の書き込み制御の原理について説明する。

【００３０】

ＯＳファイルシステム５０と物理ディスク（ディスクアレイ１８）の間に位置するフィルタドライバとしてＲＡＩＤ高速化ドライバ１００が設けられる。ＲＡＩＤ高速化ドライバ１００は、ファイルシステム５０からのライトリクエストに対して、▲１▼アドレス変換を行い、書き込み先物理アドレス（ＲＡＩＤ上の論理パーティション内アドレス）を書き込み対象データストライプの次の空き領域とし、実際に書き込む機能、▲２▼ライトリクエストの書き込み先アドレスをＡＭＴキャッシュ１７２に登録する機能を有する。

【００３１】

このＲＡＩＤ高速化ドライバ１００は、ＯＳファイルシステム５０からのライトリクエストに対して、そのリクエストされた論理アドレスを物理アドレスにＡＭＴキャッシュ１７２を用いて変換し、（例え、ターゲットアドレスが連続していない一連のライトリクエストに対しても）ディスクアレイ１８に形成されるデータストライプに転送された書き込みデータを書き込む。このデータストライプを構成するディスク上の連続したアドレスである連続領域は、ディスクに効率良く書き込めるアライン、及びサイズである。このアライン、サイズはディスクを直接操作するドライバやＲＡＩＤコントローラ１６に依存して決まる。特に、書き込み対象のディスクアレイ１８がＲＡＩＤ５で構成されている場合には、上記連続領域が「パリティグループ」、あるいはその整数倍である。

【００３２】

ＲＡＩＤ高速化ドライバ１００を使用した書き込み方式では、ホスト計算機（本実施形態ではＯＳファイルシステム５０）から書き込み要求された論理アドレスに従ってデータ書き込み位置を決定するのではなく、ホスト計算機から書き込み要求された順番で書き込みデータを順次蓄積することによって複数の書き込みデータブロックから構成される大きなデータブロックを構成し、その大きなデータブロックを一括してディスクアレイ１８の空き領域に上から順番に「まとめ書き」する。「まとめ書き」の単位は物理ストライプである。つまり、「まとめ書き」の度に１つの空き物理ストライプを生成し、そこに１物理ストライプ分のデータブロックがシーケンシャルに書き込まれる。これにより、ランダムなアクセスをシーケンシャルなアクセスに変換することが出来、書き込み性能を大幅に向上させることができる。しかしながら、本発明では、上記の「まとめ書き」による一括書き込みに限定されず、ＯＳファイルシステム５０からのライトリクエスト単位、又は複数のライトリクエスト単位に応じて書き込み処理を実行しても良い。

【００３３】

図２では、上記した「まとめ書き」による書き込み動作が示されている。ＯＳファイルシステム５０から送られてくる書き込みデータのブロックデータサイズが２ＫＢ、１ストライプユニットのデータサイズが６４ＫＢ、１物理ストライプ（パリティグループ）分のデータサイズが２５６ＫＢ（＝６４ＫＢ×４）の場合の例である。２ＫＢの書き込みデータブロックは、ＯＳに組み込んで用いられるＲＡＩＤ高速化ドライバ１００によって取得され、主メモリ１３のドライバ作業領域１７のライトバッファ１７１に蓄積される。

【００３４】

基本的には、２５６ＫＢ分のデータブロック（２ＫＢ×１２８個のデータブロック）がドライバ作業領域１７のライトバッファ１７１に蓄積された時点で、ＲＡＩＤ高速化ドライ

バ１００の制御の下、ディスクアレイ１８の１物理ストライプに対する書き込みが１度にまとめて行われる。この場合、ＲＡＩＤコントローラ１６は、２５６ＫＢ分の書き込みデータブロックのみからパリティを生成できるので、パリティ計算のために旧データを読み出す等の処理が不要となり、良く知られたＲＡＩＤ５の書き込みペナルティーを減らすことができる。

【００３５】

（第１の実施形態）

次に、図３を使ってＲＡＩＤ高速化ドライバ１００によるディスクアレイ１８への書き込み動作について説明する。この図３は、上位のファイルシステム５０から送られてきたライトリクエストＲｅｑ１，Ｒｅｑ２，Ｒｅｑ３，Ｒｅｑ４，‥‥をＲＡＩＤ高速化ドライバ１００が処理する様子を示している。

【００３６】

なお、図３では、１つのディスクに対して書き込み／参照処理をしているように示されているが、図３のディスク１８０は論理的なものであり、実際には図１又は図２に示すように複数の物理ディスク（ＤＩＳＫ０～ＤＩＳＫ４）から構成されるＲＡＩＤ構成をとっているものとする。そして、ディスク１８０全体が１つのパーティションを構成している様子を示している。

【００３７】

図３において、ディスク１８０のパーティション内は、アドレス変換テーブル領域（以下、ＡＭＴ領域と称する）１８ａと、ストライプＩＤログ領域１８ｂ１、論理アドレスログ領域１８ｂ２から成る管理データ領域１８ｂと、データ領域１８ｃに分けて管理されている。

【００３８】

例えば、現在の書き込み対象の物理データストライプがデータ領域１８ｃの「ストライプ３４」であると想定する。ＲＡＩＤ高速化ドライバ１００は、書き込み対象データストライプ３４に対して全ての領域に書き込みデータが割り当てられた時点で、新しい空ストライプが次の書込み対象ストライプとして割り当てる。以降、ライトリクエストデータのライト処理は、新たに割り当てたストライプの領域に対して行うようになる。

【００３９】

ＲＡＩＤ高速化ドライバ１００は、ＯＳファイルシステム５０からのライトリクエストReq1,Req2,Req3,Req4,‥に対して、ライトバッファ１７１に記憶したデータをディスク１８０のデータ領域１８ｃのストライプ３４に書き込むと共に、ＡＭＴキャッシュ１７２にライトした実際の物理アドレスを登録する。また、ＯＳファイルシステム５０からのリード要求に対しては、ＡＭＴキャッシュ１７２を参照して、指定された論理アドレスに対する物理アドレスを求め、ディスク１８０からその物理アドレスをリードした結果のデータをＯＳファイルシステム５０に返す。

【００４０】

データ領域１８ｃの各データストライプの最後の１ブロックには「ＴＡＧ領域」ＴＡＧが設けられる。各ストライプへの書き込みに際して、このＴＡＧ領域に、そのデータストライプに関して、▲１▼そのデータストライプを書き込んだ時刻（あるいは書込みに関するシーケンス番号としてのタイムスタンプＴＳ）、▲２▼有効データブロックに関してデータストライプの各ブロックの論理アドレスが記録保存されている。

【００４１】

また、データストライプへの書き込みに際し、管理データ領域１８ｂの論理アドレスログ領域１８ｂ２には、現在書き込み中のデータストライプに書き込んだデータブロックの「論理アドレス」がデータブロック毎に記録される。

【００４２】

また、管理データ領域１８ｂのストライプＩＤログ領域１８ｂ１には、書き込み対象データストライプとして選んだストライプＩＤが時系列順に記録される。この例では、ストライプＩＤ「３４」の書き込みの前に、ストライプＩＤ「３１」の書き込みが行なわれたこ

とが分かる。

【００４３】

図３の上部には、主メモリ１３上に確保したドライバ作業領域１７が示されている。この図３では、この領域１７にライトバッファ１７１、およびＡＭＴキャッシュ１７２の２つの領域が確保されていることを示している。ＡＭＴキャッシュ１７２は、ＡＭＴ領域１８ａに記憶されるアドレス変換テーブルの一部が記憶されており、システム稼働中はこのＡＭＴキャッシュ１７２がアクセスされる。ＡＭＴキャッシュ１７２に登録されていない論理アドレスによるアクセス要求があった場合、ＡＭＴ領域１８ａのアドレス変換テーブルから該当する変換テーブルの内容が読み出され、ＡＭＴキャッシュ１７２に登録される。ＡＭＴキャッシュの内容が更新された場合、あるいはシステムがシャットダウンされる場合には、ＡＭＴキャッシュ１７２の内容はＡＭＴ領域１８ａのアドレス変換テーブルに書き戻される。

【００４４】

ライトバッファ１７１は、ＯＳファイルシステム５０からの書き込みリクエストについて、そのデータを最初にバッファリングするための領域である。１つのライトバッファは１データストライプの大きさであり、図３では図面スペースの都合上１つのライトバッファしか記載していないが、複数のライトバッファが用意される。第１の実施形態の発明では、このライトバッファ１７１は必ずしも必要なものではない。例えば、１データブロックまたは複数データブロックを記憶するバッファメモリであっても構わない。

【００４５】

ＡＭＴキャッシュ１７２は、ディスク上のＡＭＴ領域１８ａのアドレス変換テーブルをキャッシュして記憶した領域である。ＲＡＩＤ高速化ドライバ１００は、ＡＭＴキャッシュ１７２を参照、または変更する場合には、決められた固定サイズ（たとえば４ＫＢ）の単位でＡＭＴ領域１８ａからＡＭＴキャッシュ１７２に読み込んで、主メモリ１３上で参照、または変更を行う。変更を行った場合には、そのキャッシュされたＡＭＴキャッシュ１７２は元のＡＭＴ領域１８ａに書き戻される。

【００４６】

次に、ＲＡＩＤ高速化ドライバ１００によるディスク１８０のパーティションに対するデータ書き込み処理の動作を図４に示すフローチャートを用いて説明する。なお、以下の説明では、書き込み対象データストライプを「ストライプ３４」であるとして説明する。

【００４７】

図４は、ライト処理のメインルーチンの処理を示す。ＯＳファイルシステム５０からライトリクエストＲｅｑｉ（ｉ＝１，２，３，４‥）が入力され、そのライトリクエストＲｅｑｉは、論理アドレスａｄｄｒｉから始まる２ＫＢ単位のライトブロックデータＢ０～Ｂｎからなるものとする（ステップＳ４０１）。

【００４８】

このライトリクエストReqiに対して、空き領域を有するストライプが、書込み対象ストライプとして割り当てられているか否かがチェックされ（ステップＳ４０２）、割り当てられていなければディスク上の空ストライプを１つ選択する（ステップＳ４０３）。そして、そのストライプＩＤ（この例では、ストライプ３４）をＩＤｋとして、管理データ領域１８ｂのストライプＩＤログ領域18b1に次のエントリとして書き込む（ステップＳ４０４）。ストライプＩＤログ領域18b1には、書き込み対象に選ばれたストライプのＩＤが時系列順に記録されていく。

【００４９】

次に、ＲＡＩＤ高速化ドライバ１００は、管理データ領域１８ｂの論理アドレスログ領域１８ｂ２を割り当てる（ステップＳ４０５）。論理アドレスログ領域１８ｂ２は、ライトリクエストＲｅｑｉのデータをデータストライプ３４に書き込んだ時に、そのデータの論理アドレスを憶えておくための領域である。この論理アドレスログ領域１８ｂ２の先頭には、書き込み対象のデータストライプＩＤが記録され、以降、ストライプを構成する物理ブロックがどの論理アドレスのデータを保持しているかが記録される。なお、ステップＳ

４０２で、書き込み対象データストライプ３４が既に割り当てられている場合には、ステップＳ４０３，Ｓ４０４，Ｓ４０５の処理が省略されて、ステップＳ４０６への進む。

【００５０】

次に、ＲＡＩＤ高速化ドライバ１００は、ライトリクエストＲｅｑｉによるライトデータを２ＫＢのブロック単位で書込み対象データストライプ３４の空き領域に割り当てる。これは、データストライプ３４の空き領域が無くなるまで、あるいは、ライトリクエストＲｅｑｉの全てのブロックを割り当てるまで行われる（ステップＳ４０６）。

【００５１】

次に、ＲＡＩＤ高速化ドライバ１００は、データストライプ３４に対するデータ書き込みのＩ／Ｏ要求を発行する（ステップＳ４０７）。この際、書き込みできるデータはライトバッファ１７１上で１つに纏められ、Ｉ／Ｏ要求が発行される。例えば、図３のライトリクエストＲｅｑ１は、２ＫＢのブロックデータを２個書き込むものであるが、これを１つのライトリクエストとしてＩ／Ｏ要求を発行する。これにより、ディスクの特性から効率の良い書き込みが行われる。更に、ライトデータに関する論理アドレス情報を、ステップＳ４０５で割り当てられた論理アドレスログ領域１８ｂ２に書き込むためのＩ／Ｏ要求が発行される。この例では、論理アドレスログ領域１８ｂ２の先頭にストライプＩＤ＝３４が書き込まれ、それ以降に書き込まれたデータの論理アドレスＶとして、ライトリクエストＲｅｑ１の２ブロックデータの書き込みに対応してＶ＝１００，１０２が、同様にＲｅｑ２の３ブロックデータに対応してＶ＝７６，７７，７８が、Ｒｅｑ３の１ブロックデータに対応してＶ＝６０が、Ｒｅｑ４の４ブロックデータに対応してＶ＝７９，８０，８１，８２が書き込まれる（ステップＳ４０８）。なお、上記したステップＳ４０７、４０８で発行したＩ／Ｏ要求は非同期に処理される。

【００５２】

次に、書き込み対象データストライプ３４に空き領域が有るか否かが調べられる（ステップＳ４０９）。空き領域が無い場合、つまり書き込み対象データストライプ３４の全容量分の書き込みデータが揃った場合には、ＴＡＧ情報（対象データストライプ３４に書き込んだブロックの論理ブロック番号テーブル）の書き込みＩ／Ｏ要求を発行する（ステップＳ４１０）。ＴＡＧ情報は原則として、ディスク１８０上のどこに配置しても良いが、この実施例ではデータストライプ３４の１つのブロックとして存在している。ステップＳ４１０のＴＡＧ情報の書き込み処理は非同期に行われる。そして、その書き込み対象データストライプ３４に書き込んだデータブロックのＴＡＧ情報をＡＭＴキャッシュ１７２に登録してＡＭＴキャッシュを更新する（ステップＳ４１１）。この処理についてもＡＭＴキャッシュ１７２の操作にＩ／Ｏ要求が伴う場合があり、非同期に処理が行われる。一方、ステップＳ４０９において、書き込み対象データストライプ３４に空き領域が有る場合には、上記ステップＳ４１０、Ｓ４１１の処理をせず、ステップＳ４１２に進む。

【００５３】

最後に、ＲＡＩＤ高速化ドライバ１００は、ライトリクエストＲｅｑｉに対して全てのデータブロックの書き込みＩ／Ｏ要求が発行されたか否かを判断する（ステップＳ４１２）。まだライトすべきデータが残っている場合には、ステップＳ４０２に戻り、新たなストライプを割り当てて、同様にライトリクエストを書いていく。

【００５４】

上記ステップＳ４１２で、ライトリクエストＲｅｑｉに対する全てのデータブロックの書き込みＩ／Ｏ要求が発行していた場合には、ディスクへの書き込み処理が終了する。

【００５５】

図５は、上記ステップＳ４０７、Ｓ４０８で発行したＩ／Ｏ要求の完了処理（通常は完了割り込みの処理である）の動作を示すフローチャートである。Ｉ／Ｏ完了処理１は、ステップＳ４０７に対応した完了処理であり、また、Ｉ／Ｏ完了処理２はステップＳ４０８に対応した完了処理である。

【００５６】

ＲＡＩＤ高速化ドライバ１００は、データ書き込み完了（Ｉ／Ｏ完了処理１）では、完了

したデータ書き込みに対応する論理アドレスログ番号の書き込み処理が完了しているか否かをチェックし（ステップＳ５０１）、一方、論理アドレスログ書き込み完了（Ｉ／Ｏ完了処理２）では、完了した論理アドレスログの書き込みに対応するデータ書き込み処理が完了したか否かをチェックする（ステップＳ５１０）。それらの書き込みが完了していれば、元々のライトリクエストＲｅｑｉに関して、全てのデータ書き込みと、論理アドレスログの書き込みとが完了しているか否かをチェックする（ステップＳ５０２）。それらが完了していれば、ＯＳファイルシステム５０にライトリクエストＲｅｑｉに対するディスクへの書き込みが全て完了したことを通知する（ステップＳ５０３）。即ち、全ての書き込みブロックがデータストライプ３４に書き込まれ、かつ、それらのブロックの論理アドレス情報が論理アドレスログ領域１８ｂ２に書き込まれた時点で、ＲＡＩＤ高速化ドライブ１００はライトリクエストＲｅｑｉの完了通知をリクエスト発行元のＯＳファイルシステム５０に通知する。

【００５７】

図６は、図４のステップＳ４１０における、ＴＡＧ書き込みＩ／Ｏ要求の完了実行処理の動作を示すフローチャートである。ＴＡＧ書き込みＩ／Ｏ要求の完了実行処理（Ｉ／Ｏ完了処理３）では、ＴＡＧ書き込みＩ／Ｏ要求処理が終ったストライプ３４に割り当てた論理アドレスログ領域を開放して（ステップＳ６００）、ＴＡＧ情報の書き込みを終了する。

【００５８】

上記した第１の実施形態によれば、本来、ランダムライトリクエストに関して、ディスク上には物理的に離れた領域に書き込みが起こり、ディスクのシークや回転待ちによる性能低下があった。本発明によれば、不揮発性メモリの代わりに、ディスク上に設けられたＡＭＴ領域１８ａ（主メモリ上のＡＭＴキャッシュ１７２を含む）、管理データ領域１８ｂ、データ領域１８ｃを用いることにより、ランダムライトが物理的に連続した領域への書き込みとして実行することが出来る。したがって、シーク／回転待ちが少なくなり、結果として高速な書き込み処理が実現される。ＲＡＩＤ５のディスクに対しては、パリティグループ単位での書き込みになるようにストライプ境界とサイズを合わせれば、ストライプへの書き込みが全て終った段階で、ＲＡＩＤコントローラのキャッシュメモリ上でパリティグループが揃うため、パリティ計算のためのディスクリード処理が不要となり、所謂「ＲＡＩＤ５のライトペナルティ」を無くすことが可能になり、結果として高速な書込み処理が実現される。

【００５９】

また、一連のデータ書き込み処理が完了しない状態で、不慮のシステムダウンによる処理停止が起こった場合、論理アドレスログ領域１８ｂ２に記録した論理アドレスログ情報を用いて、システムリブート後のデータ回復処理が実行できる。なお、論理アドレスログ情報は、書き込み対象データストライプ３４に書き込まれる全ての書き込みデータブロックの論理アドレスと物理アドレスの関係が、データストライプ３４のＴＡＧ情報として登録された時点で不要になる。

【００６０】

（第２の実施形態）

次に、本発明の第２の実施形態について説明する。

図４に示した第１の実施形態では、論理アドレスログ領域１８ｂ２の割り当ては、ステップＳ４０３～Ｓ４０５で空ストライプを割り当てる際に行っている。論理アドレスログ領域１８ｂ２は、図３に示すように固定的に１つ、あるいは複数用意されている。この第２の実施形態では、複数の論理アドレスログ領域１８ｂ３を用いて書き込み処理するものである。

【００６１】

先ず、図４のステップＳ４０５で行なった新しい論理アドレスログ領域を割り当てに際して、この実施形態では、複数の論理アドレスログ領域（１８ｂ２－１～１８ｂ２－ｍ）のうち、一番最近使用されて開放された論理アドレスログ領域を選ぶようにする。

【００６２】
これを実現するために、使用した論理アドレスログ領域１８ｂ２をＬＲＵで管理する仕組みを用意する。また、論理アドレスログ領域（１８ｂ２－１～１８ｂ２－ｍ）を示すＩＤ番号に関するスタック状のフリーリソース管理手段（図示せず）を用意する。なお、システム初期化時には、フリーリソース管理手段に全ての論理アドレスログ領域で示されるストライプＩＤが登録される。
【００６３】
図４のステップＳ４１０の処理によってＴＡＧ書き込みが完了した時点で、ＲＡＩＤ高速化ドライバ１００はそのストライプに割り当てていた論理アドレスログ領域を開放する（図６のステップＳ６００）。ここで、開放された論理アドレスログ領域（仮に１８ｂ２－１とする）を示すＩＤ番号は、フリーリソース管理手段のスタックに上から積まれる。図４のステップＳ４０５において、論理アドレスログ領域を新しく必要とする場合には、そのスタックの一番上に蓄積されたＩＤ番号が読み出され、そのアドレスに対応する論理アドレスログ領域１８ｂ２－１が確保される。即ち、一番最近開放された論理アドレスログ領域１８ｂ２－１が確保されることになる。このように管理することにより、常に「最も最近書き込み処理が行われ、そして開放された論理アドレスログ領域」が使用できるようになる。
【００６４】
一般的に、バッテリバックアップされたキャッシュを搭載したＲＡＩＤコントローラでは、繰り返し同じ領域に書込みを行うと、キャッシュに長い時間保持されるようになるために、引き続き発生するそれらの領域に対するライト処理は高速になる。この第２の実施形態では、この特性を利用することにより、上記のように論理アドレスログ領域１８ｂ２を管理して、なるべく同じ論理アドレスログ領域１８ｂ２－１を使用することにより、書き込みのオーバヘッドを少なくすることが可能となる。
【００６５】
この第２の実施形態の効果について更に解説する。
データ書き込みはパリティグループとしてのストライプ単位の連続領域への書き込みとなり効率が良いが、論理アドレスログ情報に関しては、パリティグループ単位での書き込みになるとは限らず、書き込み効率は悪い。ライトリクエストに対する完了通知は、「データの書き込み」と「論理アドレスログ情報の書き込み」の両方の完了をもって行われる。このため、論理アドレスログ情報の書き込みが遅くなると「書き込み処理全体」が遅くなってしまう。
【００６６】
一方、ディスクの読み書きデータのアドレスを管理するキャッシュメモリを有し、ライトバックキャッシュ機能を持つＲＡＩＤコントローラの場合は、同じアドレスに対するライトリクエストが続けて行われると、２回目以降のライトが早く完了することが多い。これは、ライトバックキャッシュに直前に書いたデータが記憶されており、同じアドレスへのデータは、キャッシュ上のデータの変更のみ完了するからである。したがって、実際のディスクへの書き込みを待たない。そのキャッシュ領域は、ＲＡＩＤコントローラのキャッシュ制御のポリシーにより、いつかはディスクに書き込まれ開放されるが、同じアドレスへのライトが短い間隔で定期的に続けば、恒常的にキャッシュ領域が確保されつづけ、比較的早くアクセスすることが可能になる。
【００６７】
第２の実施形態では、この特性を活かして論理アドレスログの書き込み処理を高速化する。例えば、（１）１ストライプのデータ書き込み分に相当するデータの論理アドレスログ情報をセーブするための領域（論理アドレスログ領域）を複数設ける。（２）書き込み対象データストライプを新規に割り当てる場合には、現在使っていないもので一番最近使った「論理アドレスログ領域」を１つ選び、ストライプにデータを書き込む際に、論理アドレスログ情報を書き込む領域として使用する。（３）対象データストライプに全てのデータを書き込んだ後には、それら書き込んだブロックを対象データストライプのＴＡＧ情報

に登録する。全てのブロックに付いて登録が完了した時点で使用していた「論理アドレスログ領域」を解放する。

【００６８】

上記のように「論理アドレスログ領域」をＬＲＵで管理することにより、論理アドレスログ情報を書き込む領域が、ディスク上の限定された固定領域（領域が複数の場合も含む）になり、論理アドレス領域への書込みが高速になる結果、書込み処理の完了タイミングも早くなる。

【００６９】

（第３の実施形態）

次に、本発明の第３の実施形態について説明する。

図７は、第３の実施形態の動作を示すフローチャートである。

この第３の実施形態では、複数のライトリクエストを１つに纏めて書き込み処理するものである。ＲＡＩＤ高速化ドライバ１００が処理している未完了のライトリクエスト数を変数（Ｏｕｔｓｔａｎｄｉｎｇ　Ｒｅｑ　Ｃｏｕｎｔ）で管理することにより実現する。Ｏｕｔｓｔａｎｄｉｎｇ　Ｒｅｑ　Ｃｏｕｎｔは初め「０」で初期化されているものとする。

【００７０】

図７において、ＯＳファイルシステム５０からのライトリクエストＲｅｑｉを受信すると（ステップＳ７０１）、ＲＡＩＤ高速化ドライバ１００はそのライトリクエストＲｅｑｉを変数としてＯｕｔｓｔａｎｄｉｎｇ　Ｒｅｑ　Ｃｏｕｎｔに登録する（ステップＳ７０２）。つまり、変数Ｏｕｔｓｔａｎｄｉｎｇ　Ｒｅｑ　Ｃｏｕｎｔ（カウンタ値）の内容が＋１される。そして、Ｏｕｔｓｔａｎｄｉｎｇ　Ｒｅｑ　Ｃｏｕｎｔの値がある定数「Ａ」より多いか否かがチェックされ（ステップＳ７０３）、定数「Ａ」より少ない場合には、そのライトリクエストＲｅｑｉはステップＳ７０４に進み、図４で示したライトリクエストＲｅｑｉのＩ／Ｏ要求の発行処理（図４のステップＳ４０７）が実行される。

【００７１】

一方、Ｏｕｔｓｔａｎｄｉｎｇ　Ｒｅｑ　Ｃｏｕｎｔがある定数「Ａ」より大きい場合には、そのライトリクエストＲｅｑｉは一旦保留（ペンディング）され、ペンディングリクエストキューＰＬｉｓｔに入れられる（ステップＳ７０５）。ペンディングリクエストキューＰＬｉｓｔに入れられたライトリクエストＲｅｑｉ（複数登録可能）は、以下の条件が満たされた時に一まとめにされて書き込み処理が行われる。

（ペンディングされたライトリクエストのＩ／Ｏ要求発行条件）

ａ）ペンディングしているリクエストＲｅｑｉの数Ｐｃｏｕｎｔが定数「Ｂ」以上になった（ステップＳ７０７のチェック）。

ｂ）ペンディングしているリクエストＲｅｑｉのデータサイズＰｓｉｚｅの合計が定数「Ｃ」以上になった（ステップＳ７０８のチェック）。

ｃ）ペンディングしているリクエストＲｅｑｉのデータサイズＰｓｉｚｅの合計が現在の書込み対象データストライプの残りの空き領域を越えた（ステップＳ７０９のチェック）。

ｄ）タイマ関数ＴｉｍｅｒＲｏｕｔｉｎｅ（）が指定時間を経過して起動された（ステップＳ７１０のチェック）。

【００７２】

ここでタイマ関数ＴｉｍｅｒＲｏｕｔｉｎｅ（）とは、時間と関数を指定すると、指定した時間が経過した時点で指定した関数を実行してくれるＯＳ（オペレーティングシステム）のサービスで登録した関数のことである。この場合、例えば「３０ｍｓ時間後に指定する関数ＴｉｍｅｒＲｏｕｔｉｎｅ（）を実行せよ」というようにＯＳに登録する。ステップＳ７１０でセットするタイマ関数ＴｉｍｅｒＲｏｕｔｉｎｅ（）の処理は図８に示している。

【００７３】

上記の条件ａ）～ｄ）を判定するために、図７では以下の変数で管理している。

ペンディングリクエストの数 ・・・ Ｐｃｏｕｎｔ
ペンディングリクエストのデータサイズの合計 ・・・ Ｐｓｉｚｅ
【００７４】
上記した条件ａ）～ ｄ）のいずれかが満たされた場合には（ステップＳ７０７のＹｅｓ、又はＳ７０８のＹｅｓ、又はＳ７０９のＹｅｓ、又はＳ７１０）、変数Ｐｃｏｕｎｔ、Ｐｓｉｚｅを０クリアする（ステップＳ７１１、Ｓ７１２、図８のステップＳ８０１、Ｓ８０２）。さらにタイマ関数Ｔｉｍｅｒ Ｒｏｕｔｉｎｅ（）がセットされているか否かがチェックされ（ステップＳ７１３）、タイマ関数Ｔｉｍｅｒ Ｒｏｕｔｉｎｅ（）がセットされていればそれをリセットする（ステップＳ７１４）。そして、ペンディングリストＰＬｉｓｔのライトリクエストを１つのライトリクエストＷＲｅｑ として、図４のステップＳ４０７の手順に従ってライト処理を行なう（ステップＳ７１５、図８のステップＳ８０３）。
【００７５】
なお、ここで、ペンディングリストＰＬｉｓｔにつながっているリクエストをまとめたＷＲｅｑ の書き込み対象データストライプ（例えばストライプ３４）へのデータ書き込みは、それを構成する複数のライトリクエスト毎にはライトせず、一まとめにして１つのライトリクエストで書き込む。また、論理アドレスログ領域１８ｂ２への書き込みに関しても、ＷＲｅｑ全体に関して１回の書き込みとする。これらの完了時処理を図９に示す。
【００７６】
図９において、ＲＡＩＤ高速化ドライバ１００はライトリクエストＲｅｑｉに関して、データ書き込み完了（Ｉ／Ｏ完了処理１）では論理アドレスログ番号の書き込み処理が完了しているか否かをチェックし（ステップＳ９０１）、一方、論理アドレスログ書き込み完了（Ｉ／Ｏ完了処理２）ではデータ書き込み処理が完了したか否かをチェックする（ステップＳ９１０）。それらの書き込みが完了していれば、元々のライトリクエストＲｅｑｉに関して全てのデータ書き込みと論理アドレスログの書き込みが完了しているものとしてＯＳファイルシステム５０にＩ／Ｏ完了を通知する（ステップＳ９０２）。そして、Ｏｕｔｓｔａｎｄｉｎｇ Ｒｅｑ Ｃｏｕｎｔの値から完了したライトリクエストＲｅｑｉを引いた値をＯｕｔｓｔａｎｄｉｎｇ Ｒｅｑ Ｃｏｕｎｔの値として更新記録して、処理を終了する（ステップＳ９０３）。
【００７７】
この第３の実施形態の具体的な動作の例について、図１０～図１１を用いて説明する。先ず、以下の動作例では、ＲＡＩＤ高速化ドライバ１００にＯＳファイルシステム５０からライトリクエストＲｅｑｉがマルチストリームで並行して多数到来してきている状況とする。図１０（ａ）は、この内４つのライトリクエストＲｅｑ１～Ｒｅｑ４が到来したところを示している。なお、この時点で、ＲＡＩＤ高速化ドライバ１００は既に複数のライトリクエストを処理中であり、これらリクエスト到着時には既にＯｕｔｓｔａｎｄｉｎｇ Ｒｅｑ Ｃｏｕｎｔ≧Ａの条件を満たしているものとする。即ち、図７のステップＳ７０３がＹｅｓの条件で進む。
【００７８】
図１０（ａ）で、各ライトリクエストＲｅｑ１～Ｒｅｑ４について、ライトサイズ、論理ブロック番号を示した。なお、ここではＲＡＩＤ高速化ドライバ１００のデータ管理単位は２ＫＢであるとし、ＯＳファイルシステム５０からの書き込みアドレスは、書き込み先の論理ブロック番号で示している。例えば、論理ブロック番号１００とは、論理アドレス（ＯＳファイルシステムが指定した書き込み先アドレス）で１００×２ＫＢ＝２００ＫＢのアドレスを意味する。また、ライトリクエストＲｅｑ１は書き込み先の論理アドレス番号１００に４ＫＢのデータを書き込むものであるのに対し、ＲＡＩＤ高速化ドライバ１００のデータ管理単位は２ＫＢであることから、このリクエストＲｅｑ１は論理アドレス番号Ｖとして、１００、１０１が割り当てられることを理解する。同様に、リクエストＲｅｑ２では６ＫＢであるから、論理アドレス番号Ｖ＝７６、７７、７８が割り当てられ、リ

クエストＲｅｑ３は２ＫＢであるから論理アドレス番号Ｖ＝６０が割り当てられ、リクエストＲｅｑ４は８ＫＢであるから、論理アドレス番号Ｖ＝７９、８０、８１、８２が割り当てられることを理解する。ここで、定数Ｂ＝４とし、ステップＳ７０７のＰｃｏｕｎｔ≧Ｂの条件が成立した場合を考える。

【００７９】

初め、ペンディングリクエストＰＬｉｓｔにライトリクエストが１つも無かった状態の場合に、ライトリクエストＲｅｑ１～Ｒｅｑ４がこの順に到来したとする。これらのリクエストは、先ずペンディングリクエストＰＬｉｓｔに順次格納され、ライトリクエストＲｅｑ４が来た時点で図１０（ｂ）の状態に格納される。

【００８０】

この時点で、図７のステップＳ７０７の条件が成立し、ステップＳ７１１に制御が移り、ステップＳ７１５で、ペンディングリストＰＬｉｓの４つのリクエストＲｅｑ１～Ｒｅｑ４（全２０ＫＢ）が１つのライトリクエストＷＲｅｑとして、図４で示した手順によりディスク１８０にライト処理される。この時、ライトリクエストＷＲｅｑによる２０ＫＢのデータが書き込み対象データストライプ（例えばストライプ３４）の残りの空き領域に入りきらない場合には、その空き領域に入るサイズ（例えば、１０ＫＢ）で切り、その単位で処理する。残りの１０ＫＢは、別な空き領域のストライプに、別なもう１つのライトリクエストＷＲｅｑに纏めてＩ／Ｏ要求を発行する。

【００８１】

この時のディスク１８０へのライト処理の様子を図１１に示す。ファイルシステムからのライトリクエストＲｅｑ１～Ｒｅｑ４は、本来４つのライトリクエストであるが、書き込み先アドレスは連続した１つのデータストライプ３４であるので、まとめて１つのライトリクエストＷＲｅｑによって書き込み処理が行われる。同時に、論理アドレスログ情報も、一括ライトを発行したデータに対応した論理アドレスログ領域１８ｂ２へ論理アドレス情報Ｖ＝１００，１０１，７６，７７，・・・，８２が１つのライトＩ／Ｏ要求によって書き込まれる。

【００８２】

この結果、書き込み対象データストライプ３４へのデータライトに４つのＩ／Ｏ要求が、また論理アドレスログ領域１８ｂ２への論理アドレスログ情報の書込みに４つのＩ／Ｏ要求が必要だったところが、ペンディングリストＰＬｉｓｔに格納して書き込むことにより、２つのライトＩ／Ｏ要求で書き込み処理を行なうことが可能となり、Ｉ／Ｏ要求の発行を少なくして、まとめ書きによる高速化が達成される。なお、上記した例では定数Ｂ＝４とし、Ｐｃｏｕｎｔ≧Ｂの条件が成立した場合を考えたが、上述した他の条件ｂ），ｃ），ｄ）の各条件が成立した場合（ステップＳ７０８、Ｓ７０９、Ｓ７１０）も、全く同様である。

【００８３】

この第３の実施形態によれば、ある時点でＲＡＩＤ高速化ドライバが処理中でまだ完了を通知していない、上位から送られてきたリクエスト（具体的には、ディスク制御システムドライバに入ったリクエストで、まだ完了通知を行っていないリクエスト）の個数をカウンタで管理することにより、任意時点での「ライトリクエストの同時Ｉ／Ｏ発行数」を得ることができる。この「ライトリクエストの同時Ｉ／Ｏ発行数」が所定値Ａよりも大きな値の時には、以降のライトリクエストに関して、データ及び論理アドレスログ情報の書き込みを一定時間待つ。そして、▲１▼一定以上の個数のライトリクエストが新たに到着した。▲２▼到着したライトリクエストのライトサイズの合計が、一定サイズを越えた。▲３▼到着したライトリクエストのライトサイズの合計が、現在書込み対象になっているストライプの残り空領域のサイズを越えた。▲４▼一定時間が経過した。の場合に、それまでペンディングしていたライトリクエストのデータを１まとめにして対象のストライプに書き込む。同時に書き込むデータブロック（複数）の論理アドレスログ情報をまとめて１回の書込みとして「論理アドレスログ領域」に書き込む。

【００８４】

この結果、複数のライトリクエストが１度のデータライト処理と１度の論理アドレスログ情報の書き込み処理となる。各データリクエストの開始は遅れるが、書き込み処理回数が減り、書き込みサイズが大きくなることでディスクへのトータルとして書込みオーバヘッドが小さくなり、結果的に書き込み処理のスループット向上を実現できる。

【００８５】

（第４の実施形態）

次に、本発明の第４の実施形態について説明する。

この第４の実施形態では、論理アドレスログ領域１８ｂ２に論理アドレスログ情報のみならず、書き込みデータサイズおよびチェックサムを併せて記録するものである。図１２は、ライトリクエストＲｅｑ１のライトデータａに対して、書き込み対象データストライプ３４に対応する論理アドレスログ領域１８ｂ２に、

▲１▼論理アドレス　Ｖ＝１００、１０１

▲２▼ライトサイズ　Ｓｉｚｅ＝４ＫＢ

▲３▼チェックサム　ＣｈｋＳｕｍ＝０ｘ１６ｆ９２ａａｂ

を書き込む様子を示している。このチェックサムＣｈｋＳｕｍは、ライトデータａを４バイト単位で加算した結果の値である。このように、ライトデータサイズおよびそのチェックサムを論理アドレスログ領域に記録することにより、障害回復を容易にする。

【００８６】

即ち、ディスク１８のデータ領域１８ｃへのデータ書き込み中にシステムの障害が発生した場合には、書き込み処理途中のストライプに関して、その論理アドレスログ領域１８ｂ２に書き込まれているチェックサム値とデータ領域１８ｃのデータから求めたチェックサム値とが一致するか否かを確認チェックする。一致した場合には、データ書き込みが完了していたと判断し、データを有効とみなし残りの処理（アドレス変換テーブルへの登録等）を行い、書き込み処理を完了させる。一致しなかった場合には、書き込みは完了しなかったとみなし、そのデータを捨てる。

【００８７】

（第５の実施形態）

次に、第５の実施形態について説明する。

第５の実施形態では、ライトデータａを書き込み対象データストライプ３４に書き込むだけでなく、論理アドレスログ領域１８ｂ２にも論理アドレスログ情報と一緒に書き込むものである。

【００８８】

図１３は、この第５の実施形態による論理アドレスログ領域１８ｂ２への書き込みデータの構成を示している。即ち、論理アドレスログ領域１８ｂ２には、

▲１▼論理アドレス　Ｖ＝１００、１０１

▲２▼ライトサイズ　Ｓｉｚｅ＝４ＫＢ

▲３▼チェックサム　ＣｈｋＳｕｍ＝０ｘ１６ｆ９２ａａｂ

▲４▼ライトデータａ（４ＫＢ）

が書き込まれる。実際には上記した▲１▼～▲４▼は１つのライトＩ／Ｏ要求で処理するものであり、これによりオーバヘッドを減らすことができる。

【００８９】

図１４は、この第５の実施形態によるＲＡＩＤ高速化ドライバ１００の動作手順を示すフローチャートである。ＯＳファイルシステム５０からライトリクエストＲｅｑｉ（ｉ＝１，２，３，４‥）が入力され、そのライトリクエストＲｅｑｉは、論理アドレスａｄｄｒｉから始まる２ＫＢ単位のライトブロックデータＢ０～Ｂｎからなるものとする（ステップＳ１４０１）。

【００９０】

このライトリクエストReqiに対して、空き領域を有するデータストライプが、書込み対象ストライプとして割り当てられているか否かがチェックされ（ステップＳ１４０２）、割り当てられていなければディスク上の空ストライプを１つ選択する（ステップＳ１４０３

）。そして、データストライプと同じサイズのバッファＷＢを主メモリ１３上に確保する（ステップＳ１４０４）。バッファＷＢはライトバッファ１７１に確保される。割り当てられたストライプＩＤをＩＤｋとして、管理データ領域１８ｂのストライプＩＤログ領域18b1に次のエントリとして書き込む（ステップＳ１４０５）。ストライプＩＤログ領域18b1には、書き込み対象に選ばれたストライプのＩＤが時系列順に記録（３１，３４，…）されていく。

【００９１】

次に、管理データ領域１８ｂの論理アドレスログ領域１８ｂ２を割り当てる（ステップＳ１４０６）。書き込み対象データストライプ（例えば、ストライプ３４）中の次の空き領域に、まだライト領域を割り当てていないデータストライプＢｊのための領域を可能な限り確保する（ステップＳ１４０７）。

【００９２】

ＲＡＩＤ高速化ドライバ１００は、このステップＳ１４０７で確保したサイズのデータを主メモリ１３に用意したバッファＷＢにコピーする（ステップＳ１４０８）。バッファＷＢは、ストライプ領域に等しい大きさを持った主メモリ１３上に確保したライトバッファである。

【００９３】

次に、ステップＳ１４０９では図１３のように、▲１▼論理アドレス情報と▲２▼ライトデータを１つのライトＩ／Ｏ要求によって論理アドレスログ領域１８ｂ２に書き込む。そして、データストライプに空き領域がなく、ストライプに対する書込みデータがバッファＷＢで全て揃った場合には（ステップＳ１４１０）、バッファＷＢの内容を書き込み対象データストライプ３４に一括ライトするＩ／Ｏ要求が発行され、ディスク１８０の書き込み対象データストライプ３４に書き込まれる（ステップＳ１４１１）。また、その書き込み対象データストライプ３４に書き込んだＴＡＧ情報をＡＭＴキャッシュ１７２に登録してＡＭＴキャッシュ１７２を更新する（ステップＳ１４１２）。一方、ステップＳ１４１０において、書き込み対象データストライプに空き領域がない場合には、上記ステップＳ１４１１、Ｓ１４１２の処理をせず、ステップＳ１４１３に進む。ステップＳ１４１３では、ライトリクエストＲｅｑｉに対して全てのデータブロックの書き込みＩ／Ｏ要求が発行されたか否かが判断される。まだライトすべきデータが残っている場合には、ステップＳ１４０２に戻り、新たなストライプを割り当てて、同様にライトリクエストを書いていく。

【００９４】

なお、この実施例では、書き込み対象データストライプ３４内の１データブロックをＴＡＧ領域としている。従って、ステップＳ１４１１において、予めＴＡＧデータをバッファＷＢ上にセットした状態で、バッファＷＢの書き込みが終了すれば、ＴＡＧデータの書き込みも同時に終了する。

【００９５】

図１５は、図１４のステップＳ１４０９、およびステップＳ１４１１のＩ／Ｏ完了処理の動作を示すフローチャートである。図１５（ａ）はステップＳ１４１１の書き込み対象データストライプにおける完了処理（Ｉ／Ｏ完了処理１）であり、書き込みが完了したライトリクエストＲｅｑｉの「完了」をＯＳファイルシステム５０に通知する（ステップＳ１５０１）。一方、図１５（ｂ）はステップＳ１４０９の論理アドレスログ領域１８ｂ２への書き込み完了処理（Ｉ／Ｏ完了処理２）であり、論理アドレスログ領域１８ｂ２への書き込みが完了すると、バッファＷＢを開放して（ステップＳ１５１０）、更に論理アドレスログ領域１８ｂ２を開放して終了する（ステップＳ１５１１）。

【００９６】

この第５の実施形態と、上述の第１の実施形態を比較すると、第５の実施形態では個々のライトリクエストＲｅｑｉに対して、論理アドレスログ領域１８ｂ２に論理アドレスログ情報とライトデータａの書き込みが完了すれば、その時点でＯＳファイルシステム５０に対して完了通知することができる。つまり、ＯＳファイルシステムに対するレスポンスが

早くなる。これに対し、第１の実施形態では、論理アドレスログ領域１８ｂ２への書き込みと、書き込み対象データストライプ３４への書き込みの、２つのライトＩ／Ｏ要求完了まで完了通知が待たされる。

【００９７】

一方で個々のライトデータに付いて考えると、第５の実施形態では、結果的には論理アドレスログ領域１８ｂ２と、データストライプ３４へ、２つの書き込みとなる。論理アドレスログ領域１８ｂ２は、ライトデータａを２重に書くというのは、実施形態２を用いて同じ領域への書き込みになるよう、キャッシュを有効利用したとしても、ＲＡＩＤコントローラ１６の特性によっては大きな負荷となる可能性がある。このため、第１の実施形態の方式にするか、第５の実施形態の方式にするかは、使用しているＲＡＩＤコントローラ１６の特性によって結果的に性能が良い方式を選ぶようにする。

【００９８】

（第６の実施形態）

ＲＡＩＤ高速化ドライバ１００が、書き込み処理中にシステムに障害が発生した場合には、データや論理アドレスログ情報等の書き込みが完了しない状態でシステムが立ち上がってくる場合がある。このようなときに、データ書き込みが完了していたものについては、書き込み対象データストライプに対応するＴＡＧ領域とＡＭＴキャッシュ１７２にそのブロックを登録し直すことにより、以降、そのデータへのアクセスが可能になる。ここで、データがディスク１８に正しく書き込めたか否かの判断は、論理アドレスログ情報としてセーブしたデータのチェックサム情報と、書き込み先のデータの領域から再計算したチェックサムが一致することが検出されれば、「正しいデータである」と判断出来る。その場合、論理アドレスログ領域１８ｂ２にセーブした（データのチェックサムの値を含む）値が、「本当に正しい値であるか」を保証する必要が有る。このため、例えば論理アドレスログ領域１８ｂ２の各エントリ毎に、

（ｘ）エントリ書き込み時に、特定の文字列（ｓｉｇｎａｔｕｒｅ）をエントリの領域の最初と最後に入れておき、読み出し時にチェックする。

（ｙ）エントリ全体のチェックサムを計算して記録する。

（ｚ）ターゲットのストライプＩＤを記録して、論理ストライプログ領域１８ｂ２の先頭のブロックに記録されるストライプＩＤとの一致を確認する。

等して、各エントリのデータ整合性を確認すれば良い。

【００９９】

上記した処理を、図１６のフローチャートに示す。なお、書き込み方式は第１の実施形態の方式であることを前提として説明する。

【０１００】

ステップＳ１６０１では、論理アドレスログ領域１８ｂ２を参照して使用中のものが有るか確認する。論理アドレスログ領域１８ｂ２が使用中か否かの判断は、例えば図１７に示すように、予め、論理アドレスログ領域１８ｂ２の先頭（ヘッダ部）に、使用中を示すフラグ（Ｖａｌｉｄ）を設け、論理アドレスログ領域１８ｂ２が使用中の間は、このフラグをセット（Ｖａｌｉｄ＝”１”）し、論理アドレスログ領域１８ｂ２を解放する時にリセット（Ｖａｌｉｄ＝”０”）すれば良い。図１７の例では、論理アドレスログ領域１８ｂ２が４つある例を示しており、この論理アドレスログ領域１８ｂ２の中では、使用中フラグ（Ｖａｌｉｄ＝”１”）を示している、論理アドレスログＬＡ１、ＬＡ２が使用中であると判断できる。これに従って、使用中リストＳにはＬＡ１、ＬＡ２として記録される。

【０１０１】

次に、ステップＳ１６０２では、上記のリストＳをヘッダ部のタイムスタンプＴＳの値によりソートする。このソートにより、結果的に時系列順にソートされる。図１７の例では、論理アドレスログＬＡ２のタイムスタンプＴＳ＝１１６７９が論理アドレスログＬＡ１のタイムスタンプＴＳ＝１１６８０より小さい（即ち、古い）ので、ステップＳ１６０２におけるソート結果のリストＳは、ＬＡ２、　ＬＡ１の順となる。

【０１０２】

次に、ステップＳ１６０３にて使用中の論理アドレスログ領域１８ｂ２（要素）があることを判断し、ステップＳ１６０４で、リストＳから一番古いタイムスタンプＴＳを持つ要素を選択（最初に論理アドレスログＬＡ２を選択）する。ステップＳ１６０５では、ＴＡＧイメージ用に確保した主メモリ１３の領域（ライトバッファ１７１に確保される）を初期化し、「対象データストライプの全ブロックが無効である」状態にする。そして、ステップＳ１６０６では、変数ｋ＝０に設定する。ステップＳ１６０７以降は、論理アドレスログＬＡ２内の各有効なエントリに対してデータのチェックサム値を読み出して、実際にストライプに書き込んであるデータのチェックサムと一致するか否かを確認する。有効なエントリの判断は、上述した（ｘ）～（ｚ）のようにすれば良い。

【０１０３】

図１７の論理アドレスログＬＡ２の例では、エントリＥ０、Ｅ１、Ｅ２、・・が有効なエントリであると想定している。エントリＥ０に注目すると、論理アドレスログの先頭から２つの物理ブロックに、論理アドレスＶ＝４６３，４６４と、それぞれのデータブロックのチェックサムＣＳ（ＣｈｋＳｕｍ０）＝ＣＳ０，ＣＳ１が登録されている。そこでステップＳ１６０８では、実際にデータストライプ３４（論理アドレスログＬＡ２に対応するデータストライプ）の１番目、２番目のデータブロックを読み出して、チェックサムＣｈｋＳｕｍ１＝ＣＳ０’，ＣＳ１’を求める。

【０１０４】

ステップＳ１６０９では、チェックサムＣｈｋＳｕｍ０とＣｈｋＳｕｍ１（ＣＳ０とＣＳ０’、およびＣＳ１とＣＳ２’）を比較して等しいか否かを判断する。その比較結果が一致するブロックについては正しいデータであると判断し、ステップＳ１６１０でデータストライプ３４のＴＡＧ領域にその論理ブロック番号を登録する。図１８は、このようにして、エントリＥ０、Ｅ１までの５データブロックを処理した時点のＴＡＧ領域の状態が示されている。なお、ここではデータストライプ３４の先頭ブロックの物理ブロック番号が２０００であると仮定している。そして、ステップＳ１６１１で、変数ｋ＝ｋ＋１とし、更にステップＳ１６１２にてｋ≦Ｎ（Ｎはデータストライプを構成するデータブロック数）の比較を行なう。

【０１０５】

論理アドレスログＬＡ２の全てのエントリに付いて、ステップＳ１６０７～Ｓ１６１２の処理を終えたら、ＴＡＧ情報が完成する。そして、ステップＳ１６１３およびＳ１６１４にて、ＴＡＧ情報を本来のアドレスに書き込み（ストライプ３４の場合、一番最後の物理ブロック）、同じ内容をＡＭＴキャッシュ１７２に登録してＡＭＴキャッシュ１７２を更新する。最後にＳ１６１５でＬＡ２の使用中フラグをリセット（ｖａｌｉｄ＝”０”）して、論理アドレスログＬＡ２の回復処理は終了する。同様の処理を論理アドレスログＬＡ１について行えば、障害発生時に書き込み途中であった論理アドレスログＬＡ１、ＬＡ２の書き込みを終了させることができる。

【０１０６】

なお、図１８のＴＡＧ領域には有効なデータに関しては、その論理アドレス値、無効なアドレスに関しては無効である旨を示す論理アドレス値を使って、ＴＡＧデータを作成しＴＡＧを書き込む。無効論理アドレス値としては、例えばそのパーティションに存在しない論理アドレス値を使用する。

【０１０７】

以上、本実施形態では、第１の実施形態の書き込み方式（データはリクエスト単位で対象ストライプに書き込む）を前提にしたが、第６の実施形態の書き込み方式（データは一旦論理アドレスログ領域に書き込み、バッファＷＢが一杯になった時点でバッファＷＢの単位で対象ストライプに書き込み）の場合でも、

（イ）ステップＳ１６０８で論理アドレスログ領域１８ｂ２の書き込みデータからＣｈｋＳｕｍ１を求める。

（ロ）ステップＳ１６０９、Ｓ１６１０の間で、書き込みデータを対象ストライプに書き込む。

という変更を行うことによりほぼ同様に処理できる。

【０１０８】

【発明の効果】

以上、発明によれば、ログ形式のデータブロック管理方式を採用したディスクシステムにおいて、ファイルシステムのライトアドレス情報をディスク上の固定領域に書き込むことによりＲＡＩＤコントローラのキャッシュを有効利用し書き込み時間を短縮することができる。アドレス変換テーブルの管理にチェックサムを導入することにより、不意のシステムダウン後の障害回復処理後等にアドレス変換テーブルの整合性を確認し信頼性を向上する。

【図面の簡単な説明】

【図１】本発明のディスク制御システムを適用した計算機システムの構成を示すブロック図。

【図２】本発明のＲＡＩＤ高速化ドライバによるディスクアレイの書き込み制御の原理を示すブロック図。

【図３】本発明の第１の実施形態における、ＲＡＩＤ高速化ドライバとディスクとの関係と、ＲＡＩＤ高速化ドライバによる書き込み制御を示すブロック図。

【図４】本発明の第１の実施形態における、ライト処理のメインルーチンを示すフローチャート。

【図５】本発明の第１の実施形態における、Ｉ／Ｏ完了処理１および２の手順を示すフローチャート。

【図６】本発明の第１の実施形態における、Ｉ／Ｏ完了処理３の手順を示すフローチャート。

【図７】本発明の第３の実施形態における、ライト処理のメインルーチンを示すフローチャート。

【図８】本発明の第３の実施形態における、タイマ関数のセット処理の手順を示すフローチャート。

【図９】本発明の第３の実施形態における、Ｉ／Ｏ完了処理１および２の手順を示すフローチャート。

【図１０】本発明の第３の実施形態における、ライトリクエストＲｅｑｉの構成、およびペンディングリストへの格納状況を示す図。

【図１１】本発明の第３の実施形態における、データストライプ、および論理アドレスログ領域への書き込み操作を示す図。

【図１２】本発明の第４の実施形態における、データストライプ、および論理アドレスログ領域への書き込み操作を示す図。

【図１３】本発明の第５の実施形態における、データストライプ、および論理アドレスログ領域への書き込み操作を示す図。

【図１４】本発明の第５の実施形態における、ライト処理のメインルーチンを示すフローチャート。

【図１５】本発明の第５の実施形態における、Ｉ／Ｏ完了処理１および２の手順を示すフローチャート。

【図１６】本発明の第６の実施形態における、ライト処理のメインルーチンを示すフローチャート。

【図１７】本発明の第６の実施形態における、論理アドレスログＬＡ０～ＬＡ３の構成例を示す図。

【図１８】本発明の第６の実施形態における、データストライプのＴＡＧ情報の書き込み状態を示す図。

【符号の説明】

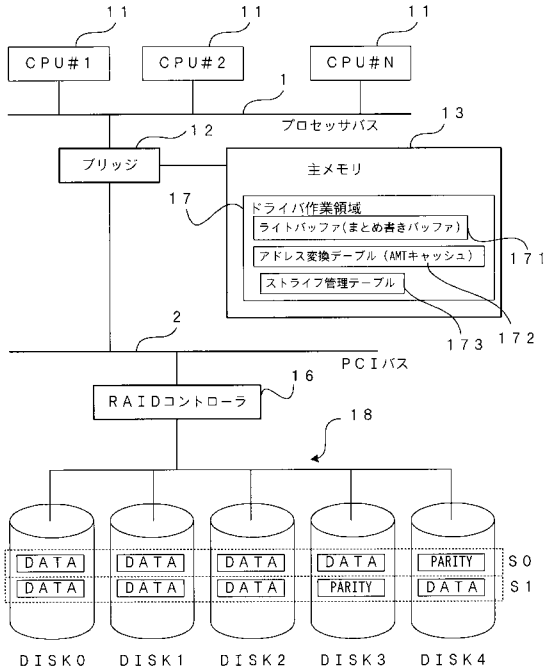
１…プロセッサバス

２…ＰＣＩバス

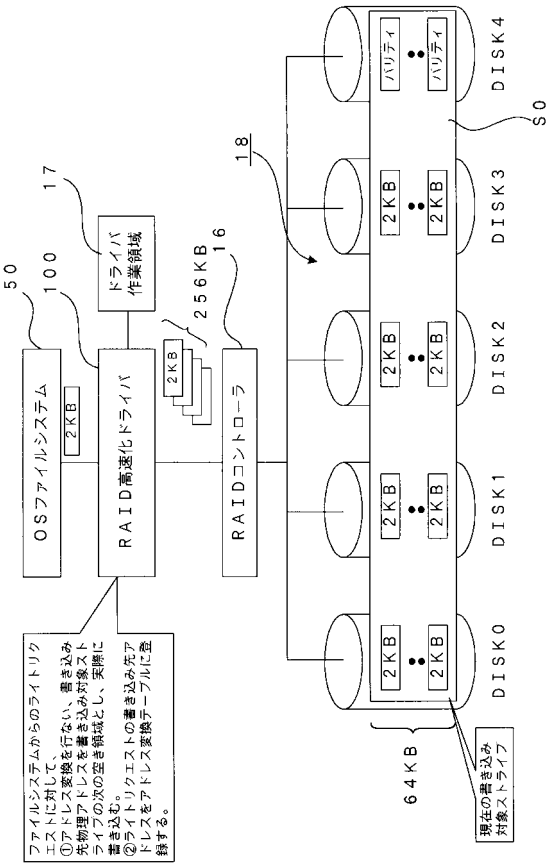
１１…ＣＰＵ＃１、＃２、＃３

１２…ブリッジ
１３…主メモリ
１６…ＲＡＩＤコントローラ
１７…ドライバ作業領域
１７１…ライトバッファ（ログメモリ）
１７２…アドレス変換テーブルＡＭＴ（キャッシュ）
１７３…ストライプ管理テーブル
１８…ディスクアレイ
１８ａ…ＡＭＴ領域
１８ｂ…管理データ領域
１８ｂ１…ストライプＩＤログ領域
１８ｂ２…論理アドレスログ領域
１８ｃ…データ領域
５０…ＯＳファイルシステム
１００…ＲＡＩＤ高速化ドライバ

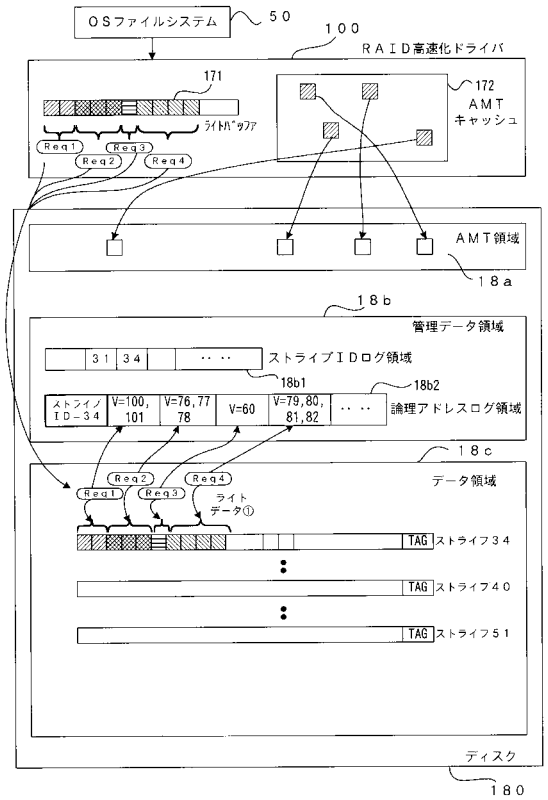
【図１】



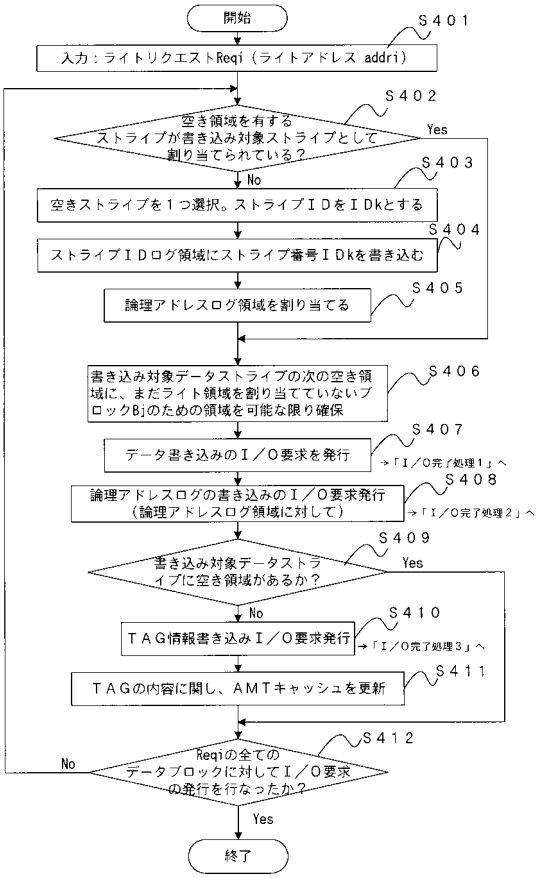
【図２】



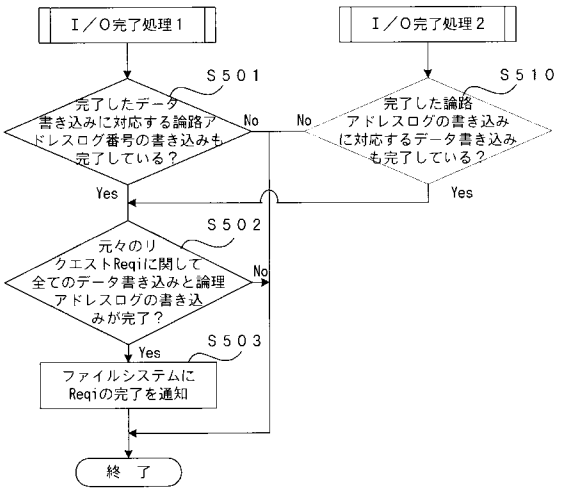
【図３】


OSファイルシステム
50
100
RAID高速化ドライバ
171
ライトバッファ
Req1
Req3
Req2
Req4
172
AMT
キャッシュ
AMT領域
18a
18b
管理データ領域
31
34
ストライプIDログ領域
18b1
18b2
メトライノ ID=34
V=100, 101
V=76,77 78
V=60
V=79,80, 81,82
論理アドレスログ領域
18c
データ領域
ライト データ①
TAG
ストライプ34
ストライプ40
ストライプ51
ディスク
180

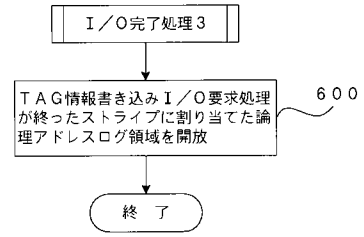
【図４】


開始
S401
入力：ライトリクエストReqi（ライトアドレス addri）
S402
空き領域を有するストライプが書き込み対象ストライプとして割り当てられている？
Yes
No
S403
空きストライプを１つ選択。ストライプＩＤをＩＤkとする
S404
ストライプＩＤログ領域にストライプ番号ＩＤkを書き込む
S405
論理アドレスログ領域を割り当てる
S406
書き込み対象データストライプの次の空き領域に、まだライト領域を割り当てていないブロックBjのための領域を可能な限り確保
S407
データ書き込みのＩ／Ｏ要求を発行
→「Ｉ／Ｏ完了処理１」へ
S408
論理アドレスログの書き込みのＩ／Ｏ要求発行（論理アドレスログ領域に対して）
→「Ｉ／Ｏ完了処理２」へ
S409
書き込み対象データストライプに空き領域があるか？
Yes
No
S410
ＴＡＧ情報書き込みＩ／Ｏ要求発行
→「Ｉ／Ｏ完了処理３」へ
S411
ＴＡＧの内容に関し、ＡＭＴキャッシュを更新
S412
Reqiの全てのデータブロックに対してＩ／Ｏ要求の発行を行なったか？
No
Yes
終了

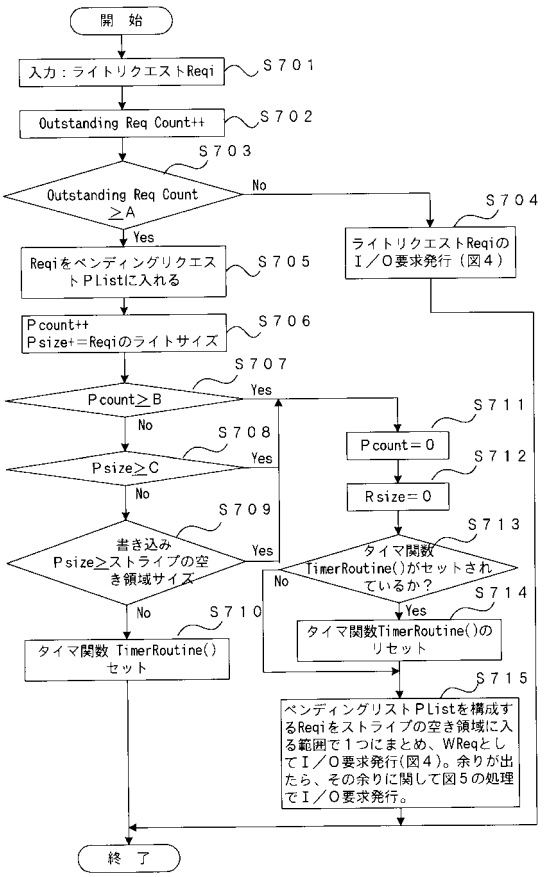
【図５】


Ｉ／Ｏ完了処理１
Ｉ／Ｏ完了処理２
S501
完了したデータ書き込みに対応する論路アドレスログ番号の書き込みも完了している？
No
No
S510
完了した論路アドレスログの書き込みに対応するデータ書き込みも完了している？
Yes
Yes
S502
元々のリクエストReqiに関して全てのデータ書き込みと論理アドレスログの書き込みが完了？
No
Yes
S503
ファイルシステムにReqiの完了を通知
終 了

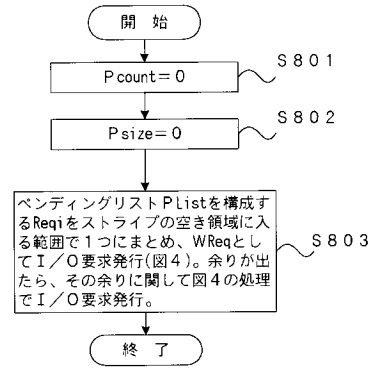
【図６】


Ｉ／Ｏ完了処理３
ＴＡＧ情報書き込みＩ／Ｏ要求処理が終ったストライプに割り当てた論理アドレスログ領域を開放
600
終 了

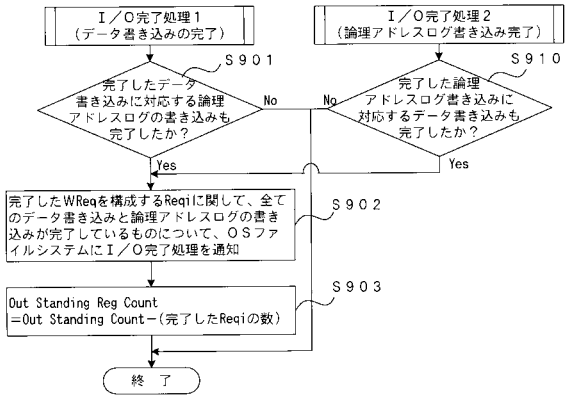
【図７】


開 始
入力：ライトリクエストReqi
S701
Outstanding Req Count++
S702
S703
Outstanding Req Count ≧A
No
Yes
S704
ライトリクエストReqiのＩ／Ｏ要求発行（図４）
S705
ReqiをペンディングリクエストPListに入れる
S706
Pcount++
Psize+＝Reqiのライトサイズ
S707
Pcount≧B
Yes
No
S708
Psize≧C
Yes
No
S709
書き込みPsize≧ストライプの空き領域サイズ
Yes
No
S710
タイマ関数 TimerRoutine()セット
S711
Pcount＝0
S712
Rsize＝0
S713
タイマ関数TimerRoutine()がセットされているか？
No
Yes
S714
タイマ関数TimerRoutine()のリセット
S715
ペンディングリストPListを構成するReqiをストライプの空き領域に入る範囲で１つにまとめ、WReqとしてＩ／Ｏ要求発行(図４)。余りが出たら、その余りに関して図５の処理でＩ／Ｏ要求発行。
終 了

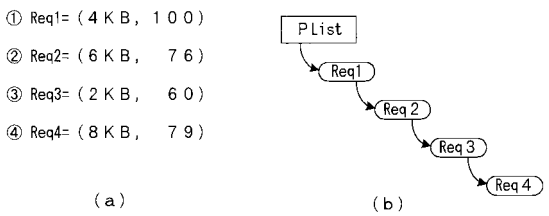
【図８】



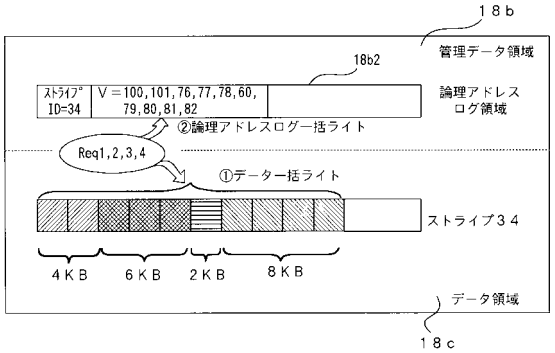
【図９】



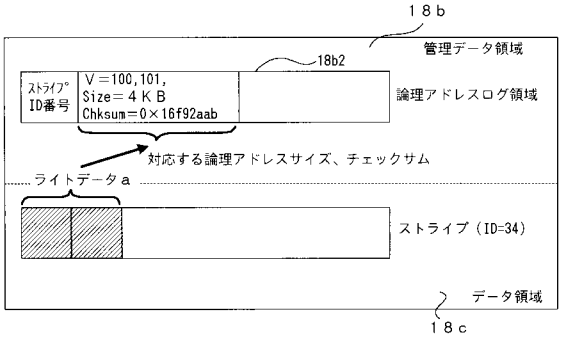
【図１０】



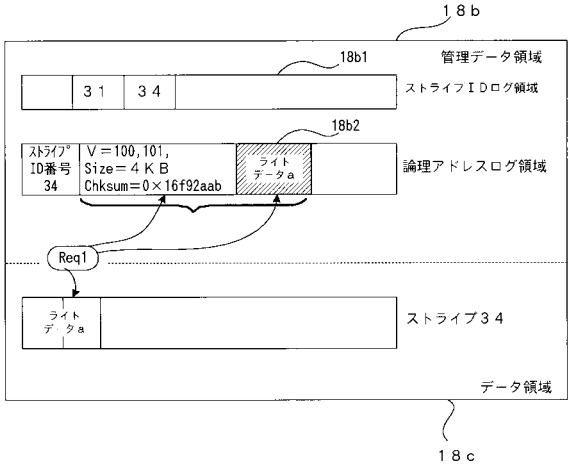
【図１１】



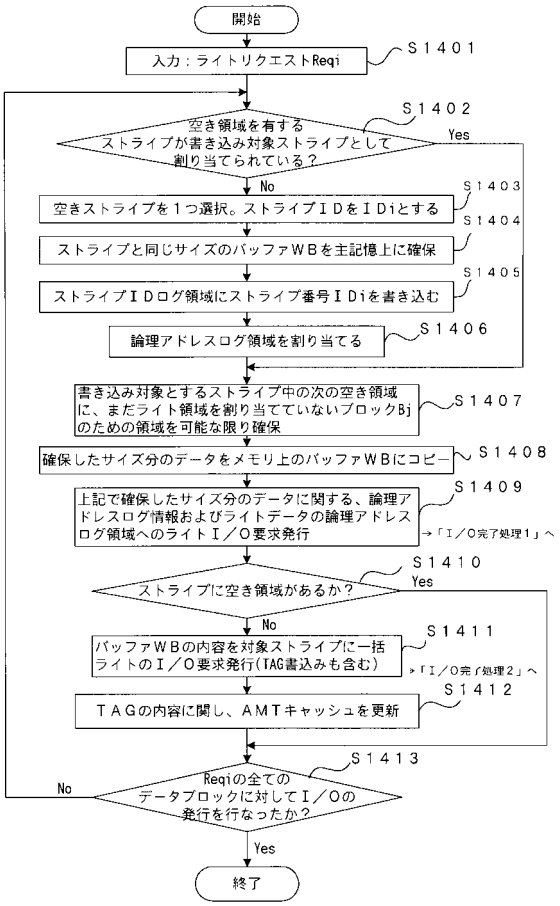
【図１２】



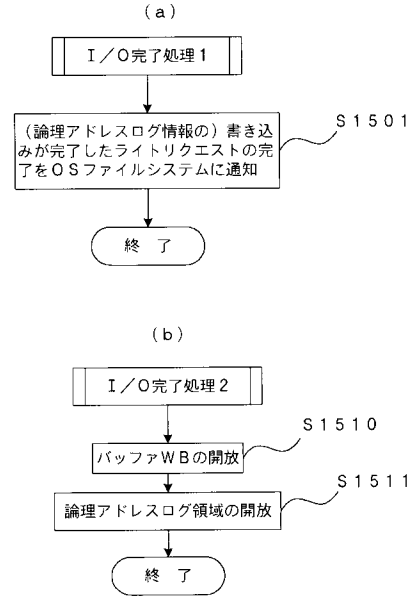
【図１３】



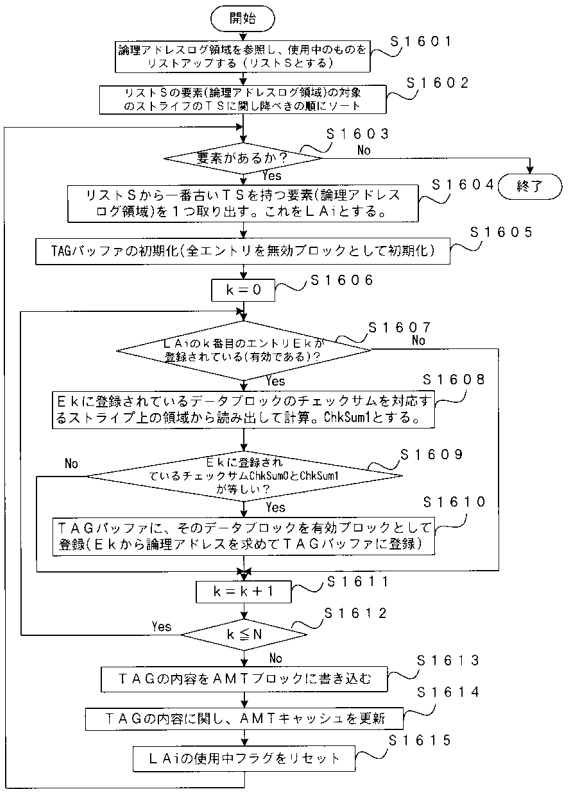
【図１４】



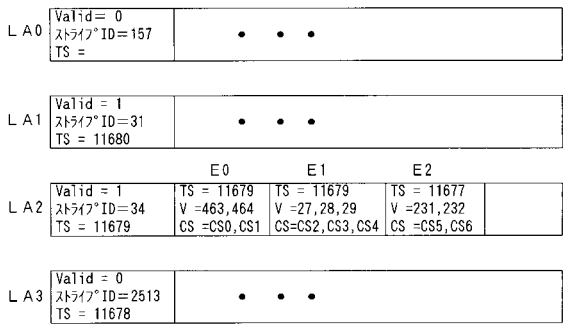
【図１５】



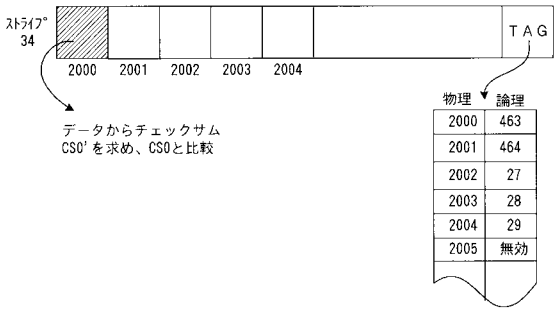
【図１６】



【図１７】



【図１８】



フロントページの続き

(56)参考文献 特開平１１－０５３２３５（ＪＰ，Ａ）
特開２０００－０４７８３２（ＪＰ，Ａ）
関戸一紀、水野聡，ＲＡＩＤ高速化技術ＲＡＩＤ　ＢＯＯＳＴＥＲ，東芝レビュー，日本，(株)東芝　デジタルメディア機器社，１９９９年１２月　１日，Vol.54 No.12，P.13-17

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
G06F 3/06-3/08
G06F 12/00-12/16
G06F 13/10-13/14
G11B 20/00-20/18