

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4547028号  
(P4547028)

(45) 発行日 平成22年9月22日(2010.9.22)

(24) 登録日 平成22年7月9日(2010.7.9)

(51) Int.Cl.

F I

G 0 6 F 12/00 (2006.01)

G 0 6 F 12/00 5 4 2 D

G 0 6 F 12/00 5 9 7 U

請求項の数 9 (全 26 頁)

(21) 出願番号	特願2008-525178 (P2008-525178)	(73) 特許権者	506197901
(86) (22) 出願日	平成18年8月1日(2006.8.1)		サンディスク コーポレーション
(65) 公表番号	特表2009-503745 (P2009-503745A)		アメリカ合衆国、95035、カリフォル
(43) 公表日	平成21年1月29日(2009.1.29)		ニア州、ミルピタス、マッカシー プルバ
(86) 国際出願番号	PCT/US2006/030228		ード 601
(87) 国際公開番号	W02007/019217	(74) 代理人	100075144
(87) 国際公開日	平成19年2月15日(2007.2.15)		弁理士 井ノ口 壽
審査請求日	平成21年8月3日(2009.8.3)	(72) 発明者	シンクレア、アラン ダブリュー、
(31) 優先権主張番号	60/705,388		イギリス連邦共和国、FK2 OBU、フ
(32) 優先日	平成17年8月3日(2005.8.3)		ォールカーク、マディストン、キャンディ
(33) 優先権主張国	米国(US)		、ブロードヘッド、ザ コテージース
(31) 優先権主張番号	60/746,740	(72) 発明者	ライト、バリー
(32) 優先日	平成18年5月8日(2006.5.8)		イギリス連邦共和国、スコットランド、E
(33) 優先権主張国	米国(US)		H6 6DE、エディンバラ、ヘンダーソ
			ン ストリート、40 フラット 4
			最終頁に続く

(54) 【発明の名称】 ブロック管理を伴う不揮発性メモリ

(57) 【特許請求の範囲】

【請求項1】

再プログラム可能な不揮発性メモリシステムであって、  
不揮発性メモリ回路であって、

メモリセルのブロックにグループ分けされ、前記ブロックのページにデータを再プログラムする前に同時に消去されるメモリセルアレイであって、前記ブロックが消去単位であるとともに物理ブロックアドレスによって識別され、前記ページが書き込み単位である、メモリセルアレイと、

前記メモリセルに接続可能なことにより、前記メモリセルを消去しかつ再プログラムすることができる読み出し回路、書き込み回路および消去回路と、を備える不揮発性メモリ回路と、

前記メモリセルアレイと動作可能に接続している制御回路であって、

前記メモリ回路とメモリシステム外部との間のデータ転送を制御するとともに、前記メモリ回路上でホストデータを記憶することも制御し、かつ

前記メモリ回路とメモリシステム外部との間でデータを転送するためのインターフェイス回路と、

処理回路とを備え前記処理回路は、

前記インターフェイス回路を介してメモリシステム外部からデータを受信し、そして前記受信済みデータをブロックのうちの1つ以上のブロックに記憶し、

メモリセルアレイのブロック内にマネジメントデータを保持し、前記マネーজে

10

20

ントデータは、

複数のブロック用の複数のレコードであって、前記レコードが、前記ブロックのアドレスによって識別されるように前記レコードと関係のある複数のブロックのうちの1つのブロックに記憶されるデータの異なった態様の複数の記述子の値を個々に含み、前記制御回路が、対応するブロック内の有効データ量であるレコード内の第1の記述子の値を書き込み、かつ対応するブロック内の消去済み容量の量であるレコード内の第2の記述子の値を書き込むように操作する、複数のレコードと、

複数のブロックのアドレスによるポインタを含む第1のリストであって、前記ポインタが、前記1つのブロックのアドレスを受信することに応じて、ブロックのうちの1つのブロック用のレコードのうちの1つのレコードがメモリセルアレイ内に位置するように個々に指示し、かつ前記ポインタをそれぞれの複数のブロックのアドレスの順序にしたがってポインタのリスト内に並べる、第1のリストと、

複数のブロックの対応するレコードが少なくとも第1と第2の記述子の値を含んでいる複数のブロックのアドレスを含む第2のリストであって、前記第2のリスト内の複数のブロックのアドレスをその対応するレコード内の第1の記述子の値の順序にしたがって並べる、第2のリストと、

複数のブロックの対応するレコードが対応するブロック内の少なくとも有効データ量と対応するブロック内の用済みデータの量とである少なくとも記述子の値を含んでいる複数のブロックのアドレスを含む第3のリストであって、前記制御回路がその対応するブロック内の有効データ量の値の順序にしたがって前記第3のリスト内の複数のブロックのアドレスを並べるように操作する、第3のリストとを含み、

前記処理回路を動作させ、前記第2のリストに出現する順序にしたがって前記第2のリスト内の複数のブロックのうちの個々のブロックに記憶されるデータで前記読み出し回路、書き込み回路および消去回路のうちの1つ以上を利用して動作させる、メモリシステム。

#### 【請求項2】

請求項1記載のメモリシステムにおいて、

複数のブロックのうちの個々のブロックに記憶されるデータ上での制御回路の動作が、個々のブロックからの有効データを消去済み空間を有する他のブロックにコピーし、その後その個々のブロックを消去することを含むメモリシステム。

#### 【請求項3】

請求項1記載のメモリシステムにおいて、

前記制御回路を、第2の記述子の値を含む対応するレコードを有する前記第3のリストからブロックを省くように操作するメモリシステム。

#### 【請求項4】

再プログラム可能な不揮発性メモリシステムであって、

不揮発性メモリ回路であって、

メモリセルのブロックにグループ分けされ、前記ブロックのページにデータを再プログラムする前に同時に消去されるメモリセルアレイであって、前記ブロックが消去単位であるとともに物理ブロックアドレスによって識別され、前記ページが書き込み単位である、メモリセルアレイと、

前記メモリセルに接続可能なことにより、前記メモリセルを消去しかつ再プログラムすることができる読み出し回路、書き込み回路および消去回路と、を備える不揮発性メモリ回路と、

前記メモリセルアレイと動作可能に接続している制御回路であって、

前記メモリ回路とメモリシステム外部との間のデータ転送を制御するとともに、前記メモリ回路上でホストデータを記憶することも制御し、かつ

前記メモリ回路とメモリシステム外部との間でデータを転送するためのインターフェイス回路と、

処理回路とを備え前記処理回路は、

10

20

30

40

50

前記インターフェイス回路を介して前記メモリシステムによって受信された複数のブロック内のデータを記憶するように動作し、

メモリセルアレイのブロック内のマネージメントデータを保持するように動作し、前記マネージメントデータは、

複数のブロック用の複数のレコードであって、前記レコードが、前記ブロックのアドレスによって識別されるように前記レコードと関係のある複数のブロックのうちの個々のブロック内の有効データ量、用済みデータの量、および消去済み容量の量の表示を個々に含む、複数のレコードと、

複数のブロックのアドレスによるポインタを含む第1のリストであって、前記ポインタが、前記1つのブロックのアドレスを受信することに応じて、ブロックのうちの1つのブロック用のレコードのうちの1つのレコードがメモリセルアレイ内に位置するように個々に指示し、かつ前記ポインタをそれぞれの複数のブロックのアドレスの順序にしたがってポインタのリスト内に並べる、第1のリストと、

複数のブロックの対応するレコードが少なくとも有効データと消去済み容量の正の値を含んでいる複数のブロックのアドレスを含む第2のリストであって、前記第2のリスト内の複数のブロックのアドレスをその対応するレコード内の有効データ量の順序にしたがって並べる、第2のリストと、

複数のブロックの対応するレコードが少なくとも有効データと用済みデータの正の値を含むが、消去済み容量の量は含んでいない複数のブロックのアドレスを含む第3のリストであって、前記第3のリスト内の複数のブロックのアドレスをその対応するレコード内の有効データ量の順序にしたがって並べる、第3のリストとを含み、

前記処理回路は、前記ブロックのアドレスが第2および第3のリスト上にある前記ブロックを、前記第2および第3のリストに出現する順序にしたがって複数のブロックのうちの個々のブロックに記憶されるデータで前記メモリシステムを動作させるために前記読み出し回路、書き込み回路および消去回路のうちの1つ以上を利用してアクセスさせる、メモリシステム。

#### 【請求項5】

メモリ回路と制御回路とを有する不揮発性メモリシステムを操作する方法であって、前記メモリ回路がメモリセルのブロックにグループ分けされ、前記ブロックのページにデータを再プログラムする前に同時に消去されるメモリセルを含み、前記ブロックが消去単位であるとともに物理ブロックアドレスによって識別され、前記ページが書き込み単位であり、前記メモリ回路が前記メモリセルに接続可能な読み出し回路、書き込み回路および消去回路を備え、前記制御回路が前記メモリシステムと通信するホストと前記メモリ回路との間のデータ転送を制御するとともに、前記メモリ回路上でホストデータを記憶することも制御する、前記方法において、

前記制御回路を介して前記メモリシステムによって受信された複数のブロック内のデータを前記書き込み回路を利用して記憶するステップと、

複数のブロックのうちの対応するブロック用の複数のレコードを前記制御回路によって前記メモリ回路の第1のブロック内に保持するステップであって、前記レコードが、前記ブロックのアドレスによって識別されるように前記レコードと関係のある複数のブロックのうちの1つのブロックに記憶されるデータの異なった態様の少なくとも第1、第2、および第3の記述子の値を個々に含み、前記第1の記述子が対応するブロック内の有効データ量であり、前記第2の記述子が対応するブロック内の消去済み容量の量であり、前記第3の記述子が対応するブロック内の用済みデータの量である、保持するステップと、

複数のブロックのアドレスによるポインタを含む第1のリストを前記制御回路によって前記メモリ回路の第1のブロック内に保持するステップであって、前記ポインタが、前記1つのブロックのアドレスを受信することに応じて、ブロックのうちの1つのブロック用のレコードのうちの1つのレコードがメモリシステム内に位置するように個々に指示し、かつ前記ポインタをそれぞれの複数のブロックのアドレスの順序にしたがってポインタのリスト内に並べる、保持するステップと、

複数のブロックの対応するレコードが少なくとも第1と第2の記述子の値を含んでいる  
複数のブロックのアドレスを含む第2のリストを前記制御回路によって前記メモリ回路の  
第1のブロック内に保持するステップであって、前記第2のリスト内の複数のブロックの  
アドレスをその対応するレコード内の第1の記述子の値の順序にしたがって並べる、保持  
するステップと、

複数のブロックの対応するレコードが少なくとも第1と第3の記述子の値を含んでいる  
複数のブロックのアドレスを含む第3のリストを前記制御回路によって前記メモリ回路の  
第1のブロック内に保持するステップであって、前記第3のリスト内の複数のブロックの  
アドレスをその対応するレコード内の第1の記述子の値の順序にしたがって並べる、保持  
するステップと、

前記第2のリストに出現する順序にしたがって前記第2のリスト内の複数のブロックの  
うちの個々のブロックに記憶されるデータ上で前記読み出し回路、書き込み回路および消  
去回路のうちの1つ以上を利用することによって前記メモリシステムの動作をするステッ  
プと、

を含む方法。

#### 【請求項6】

請求項5記載の方法において、

前記メモリシステムの動作を遂行するステップが、個々のブロックからの有効データを  
 消去済み空間を有する他のブロックにコピーし、その後その個々のブロックを消去するこ  
とを含む方法。

#### 【請求項7】

請求項5記載の方法において、

前記第3のリストでは、対応するレコードが第2の記述子の値を含んでいる複数のブロ  
 ックのアドレスが除外される方法。

#### 【請求項8】

請求項5記載の方法において、

前記第2のリストでは、対応するレコードが第3の記述子の値を含んでいる複数のブロ  
 ックのアドレスが除外される方法。

#### 【請求項9】

メモリ回路と制御回路とを有する不揮発性メモリシステムを操作する方法であって、前  
 記メモリ回路がメモリセルのブロックにグループ分けされ、前記ブロックのページにデー  
 タを再プログラムする前に同時に消去されるメモリセルを含み、前記ブロックが消去単位  
であるとともに物理ブロックアドレスによって識別され、前記ページが書き込み単位であ  
り、前記メモリ回路が前記メモリセルに接続可能な読み出し回路、書き込み回路および消  
去回路を備え、前記制御回路が前記メモリシステムと通信するホストと前記メモリ回路と  
の間のデータ転送を制御するとともに、前記メモリ回路上でホストデータを記憶すること  
も制御する、前記方法において、

前記制御回路を介して前記メモリシステムによって受信された複数のブロック内のデー  
タを前記書き込み回路を利用して記憶するステップと、

複数のブロック用の複数のレコードを前記制御回路によって前記メモリ回路の第1のブ  
 ロック内に保持するステップであって、前記レコードが、前記ブロックのアドレスによっ  
 て識別されるように前記レコードと関係のある複数のブロックのうちの個々のブロック内  
の有効データ量、用済みデータの量、および消去済み容量の量の表示を個々に含む、保持  
するステップと、

複数のブロックのアドレスによるポインタを含む第1のリストを前記制御回路によって  
 前記メモリ回路の第1のブロック内に保持するステップであって、前記ポインタが、前記  
 1つのブロックのアドレスを受信することに応じて、ブロックのうちの1つのブロック用  
のレコードのうちの1つのレコードがメモリシステム内に位置するように個々に指示し、  
かつ前記ポインタをそれぞれの複数のブロックのアドレスの順序にしたがってポインタの  
リスト内に並べる、保持するステップと、

10

20

30

40

50

複数のブロックの対応するレコードが少なくとも有効データと消去済み容量の正の値を含んでいる複数のブロックのアドレスを含む第2のリストを前記制御回路によって前記メモリ回路の第1のブロック内に保持するステップであって、前記第2のリスト内の複数のブロックのアドレスをその対応するレコード内の有効データ量の順序にしたがって並べる、保持するステップと、

複数のブロックの対応するレコードが少なくとも有効データと用済みデータの正の値を含むが、消去済み容量の量は含んでいない複数のブロックのアドレスを含む第3のリストを前記制御回路によって前記メモリ回路の第1のブロック内に保持するステップであって、前記第3のリスト内の複数のブロックのアドレスをその対応するレコード内の有効データ量の順序にしたがって並べる、保持するステップと、

10

前記ブロックのアドレスが第2および第3のリスト上にある前記ブロックを、前記第2および第3のリストに出現する順序にしたがって複数のブロックのうちの個々のブロックに記憶されるデータ上で前記メモリシステムの動作をするために前記読み出し回路、書き込み回路および消去回路のうちの1つ以上を利用することによってアクセスするステップと、

を含む方法。

【発明の詳細な説明】

【技術分野】

【0001】

本願は、半導体フラッシュメモリ等の再プログラム可能な不揮発性メモリシステムの動作に関し、より具体的にはメモリアレイの個別に消去可能なブロックの管理に関する。

20

【背景技術】

【0002】

ホストシステム、メモリシステム、その他の電子システムの外部インターフェイスを通じて通信されるデータをアドレス指定するには主に2つの手法がある。そのうちの一手法では、システムによって生成または受信されるデータファイルのアドレスが、データの論理ブロックとして設定された別個の連続論理アドレス空間範囲にマップされる（これ以降「LBAインターフェイス」と称する）。アドレス空間の大きさは通常、システムで取り扱うことのできるアドレス全域を十分にカバーする。一例において、磁気ディスク記憶ドライブは、このような論理アドレス空間を通じてコンピュータをはじめとするホストシステムと通信する。このアドレス空間は、ディスクドライブの全データ記憶容量を十分にアドレス指定する大きさを持つ。

30

【0003】

フラッシュメモリシステムは多くの場合、パーソナルコンピュータやカメラ等の様々なホストと着脱可能な状態で接続するメモリカードまたはフラッシュドライブの形で提供されるが、このようなホストシステムの中に埋め込まれることもある。ホストは通常、メモリへデータを書き込む場合に、メモリシステムの連続する仮想アドレス空間の中でセクタ、クラスタ、あるいはその他のデータ単位に一意な論理アドレスを割り当てる。ホストは、ディスクオペレーティングシステム(DOS)のように、メモリシステムの論理アドレス空間の中のアドレスにデータを書き込み、かつこれからデータを読み出す。メモリシステムの中のコントローラは、ホストから受け取った論理アドレスをメモリアレイの中のデータが実際に記憶される物理アドレスに翻訳し、これらのアドレス翻訳の経緯を把握する。メモリシステムのデータ記憶容量は少なくとも、メモリシステムのために規定される論理アドレス空間全体にわたってアドレス指定可能なデータ量に相当する。

40

【0004】

現在の商用フラッシュメモリシステムにおける消去単位のサイズは、複数セクタのデータを十分に記憶するメモリセルブロックまで拡大している。事実、多数のデータページが1つのブロックに記憶され、1ページで複数セクタのデータを記憶できる。さらに、多くの場合、2つ以上のブロックがまとめてメタブロックとして扱われ、このようなブロックのページはまとめてメタページとして論理的にリンクされる。多数のセクタのデータを含

50

むデータのページまたはメタページがまとめて読み書きされることにより、操作の並列性を高めている。そのような大容量操作単位には、それらを効率的に操作するという課題がともなう。

【0005】

説明を平易にするため、別段の断りがない限り、ここで用いる用語「ブロック」は、ある特定のシステムでメタブロックが使われるか否かに応じて、ブロック消去単位または多重ブロック「メタブロック」を指すことを意図する。同様に、ここで言及する「ページ」は、システム構成に応じて、単一のブロックまたはメタブロック内の「メタページ」の中でのプログラミングの単位を指すことがある。

【0006】

現在普及しているメモリスシステムのLBAインターフェイスが使われる場合、メモリの接続先にあたるホストによって生成されるファイルには、インターフェイスの論理アドレス空間の中で一意なアドレスが割り当てられる。次いで、メモリスシステムは普通、論理アドレス空間とメモリの物理ブロックのページとの間でデータをマッピングする。メモリスシステムは論理アドレス空間から物理メモリへのマッピングを絶えず把握するが、ホストはこれを知らない。ホストは、論理アドレス空間の中でそのデータファイルのアドレスを絶えず把握するが、メモリスシステムはこのマッピングについて無知かまたはほぼ無知の状態で作動する。

【特許文献1】米国特許出願第11/382,224号

【特許文献2】米国特許出願第11/382,228号

【特許文献3】米国特許出願第11/382,232号

【特許文献4】米国特許出願第11/382,235号

【発明の開示】

【0007】

使用されている2通りのシステムインターフェイス手法の2番目の手法では、電子システムによって生成または受信されたデータファイルが一意に識別され、そのデータはファイル内でのオフセットによって論理的にアドレスされる。コンピュータをはじめとする他のホストシステムと「スマートカード」として知られる着脱可能なメモリカードとの間では、このアドレス指定方法の形態が使われている。スマートカードは通常、識別、バンキング、販売時点購入、ATMアクセス等のために消費者に利用され、フラッシュメモリカードやフラッシュドライブに比べて少量のメモリを内蔵する。

【0008】

前に相互参照した特許出願において、大容量フラッシュメモリスシステムではホストによって割り当てられたファイル名によってデータファイルが識別され、このファイルのデータはファイル内のオフセットアドレスによってアクセスされる（これ以降「直接データファイルインターフェイス」と称する）。次いで、メモリスシステムは、各セクタまたはその他のデータ単位が帰属するホストファイルを知る。ここで論じるファイル単位は、例えば連続する論理オフセットアドレスによって序列される1組のデータであり、メモリスシステムの接続先にあたるホスト演算システムで動作するアプリケーションプログラムによって作成され、一意に識別される。

【0009】

前に援用されている4つの米国特許出願（2006年5月8日に出願された米国特許出願第11/382,224号（特許文献1）、第11/382,228号（特許文献2）、第11/382,232号（特許文献3）、第11/382,235号（特許文献4））では、メモリスシステムの動作中にアクセスされるブロックのリストを保守する手法が説明されている。リスト上の個々の項目は、メモリブロックのパラメータとそれぞれのアドレスとを含む。リスト内の項目は、これに対応するブロックの物理アドレスか、またはこの項目の中に記録されたブロックパラメータの値のいずれかに従って選択できる。本願明細書は、フラッシュメモリの指定されたブロックの中でデータブロックの1つ以上のリストを記憶することに、またリスト内の特定の項目に、この項目の中にあるフィールドの情

10

20

30

40

50

報内容に基づきアクセスすることに関する。

【0010】

その手法では、2つ以上のリスト上で全ブロックの情報を収容する整列されていないレコードを記憶するため、指定されたブロック内の論理ページのいくつかを使用する。その他のページは、ブロックの内容アドレス指定可能なパラメータと、ブロックの完全レコードへのポインタだけを収容する項目のリストのために使用する。これらのリストの中ではパラメータ値の順序で項目が記憶され、ある特定のパラメータ値を持つ項目には速やかにアクセスできる。これらのリストは、整列されていないブロックレコードに対し一種のディレクトリとしての役割を果たす。

【0011】

ブロック消去可能な不揮発性メモリでは、個々のブロックとそこに収容されたデータについて最新の情報を保守すれば、効率的なメモリ管理が可能となる。1つ以上のファイルの有効データと、用済みデータと、消去済みの空間とが混在するブロックの場合、これは非常に複雑である。しかし、このようなデータをブロック単位で把握すれば、ある種のメモリシステム操作は、特にメモリアレイにおける使用されていない空間の再生は、効率的に遂行できる。使用されていない空間（用済みデータを収容する空間または使用されていない消去済みの空間）の再生はブロック単位で行われる（ここでブロックは消去の単位である）。有効データのコピーを減らすため、ブロックに対するブロック再生操作は効率的な順序で遂行することが望ましい。ブロック再生の順序を決定するには、1つ以上の記述子値の順序でブロックを格付けすると有益であり、ここで、記述子値は、ブロックに記憶されるデータの少なくとも1つの態様を記述する。例えば、ブロックは、そこに収容された有効データの量によって序列でき、そのようにして収容する有効データ量が最も少ないブロックを最初に再生するべく速やかに識別できる。

【0012】

第1の例において、必ずしも全てのブロックではないが、メモリアレイの様々なブロックのために、レコードを保守する。有効データを収容するブロックは、それらが消去済みの空間を収容するか否かに従い（不完全ブロック）、収容しない場合にはそれらが用済みデータを収容するか否かに従い（用済みブロック）、分類される。これとは別に、用済みデータだけを収容するブロック（無効ブロック）と消去済みの空間だけを収容するブロック（消去済みブロック）とが分類される。これらの分類のいずれか1つに該当する各ブロックにつきレコードを保守する。レコードは専用のレコードブロックにて保守される。加えて、分類されたブロックはリストに列挙され、このリストは記述子値によって序列される。記述子値は、ブロック内の有効データ量、ブロックのアドレス、または他の何らかの記述子値であってもよい。リスト項目は、第1の例において、対応するレコードのポインタを収容する。

【0013】

第2の例では、様々な分類に該当するブロックのためのレコードも保守する。2つ以上のファイルのデータを収容し、かつ用済みデータもあるいは消去済みの空間も収容しないブロックのために分類「コンプリート共通ブロック」が加えられる。レコードは、専用レコードブロックの中の専用レコードページにて保守される。レコードは、ブロックのブロックアドレス、ブロックに収容された有効データ量、書き込みポインタの位置等を含む、対応するブロックについて様々な情報を収容できる。

【0014】

個々のレコードを速やかに見つけることを可能にするためにディレクトリが提供される。ディレクトリの中の項目は、レコードページと、レコードページにおける対応するレコードの位置とを識別する。ディレクトリはディレクトリブロックの中の専用ディレクトリページにて保守され、ディレクトリの項目はブロックアドレスによって序列される。ディレクトリページは重複しないブロックアドレス範囲を収容する。1つのディレクトリページで複数のレコードページのポインタを収容できる。しかし、1つのディレクトリページの項目によって指示されるレコードは1レコードページに収容されたレコードだけである

10

20

30

40

50

。したがって、レコードページの中でレコードが更新されるときには、ただひとつのディレクトリページを更新すればよい。

【0015】

ディレクトリブロックはまた、1つ以上のリストを収容できる。リスト内の項目はブロックアドレスと記述子値とを収容し、リスト内の項目は記述子値によって序列される。ブロック内の有効データ量は代表的な記述子値である。再生またはその他の目的でブロック内のデータに基づきブロックを選択するにあたって、リストは簡便な手段を提供する。例えば、有効データ量を記述子値として使用する用済みブロックのリストでは、有効データ量が最も少ない用済みブロックが最初の項目になる。このブロックは再生のためにリストから速やかに識別でき、かくしてリストは再生操作のためのキューを提供する。別の例では、ある特定の記述子値を持つブロックを探すことができる。これは、適切なリストページでバイナリ探索を遂行することによって速やかに識別できる。

10

【0016】

第3の例では、不揮発性メモリアレイの各ブロックにつき常時レコードを保守する。この場合のディレクトリページは固定ブロックアドレス範囲の項目を収容し、項目はブロックアドレスによって順次に序列される。これは、特定のブロックの項目をディレクトリページの中で所定のオフセットのところに提供するため、項目の参照先にあたるブロックのブロックアドレスを項目に収容する必要はない。

【0017】

ここではフラッシュデータファイルインターフェイスとともに動作するフラッシュメモリに使われるリストを管理しこれにアクセスする手法を説明するが、これらの手法は様々な用途でタイプの異なるリストに役立てることもできる。

20

【発明を実施するための最良の形態】

【0018】

詳細な第1の例

前述した特許出願で説明されている直接データファイル記憶法はブロックのリストを作成し、個々のブロックの使用に関する所定の値を持つ項目がこのリストから選択される。ここでは、これらのブロックリストの内容アドレス指定可能な探索のための手法を説明する。本願明細書では、添付の図1～3を参照する。

【0019】

30

図1は、メモリセルブロックの、この内容に基づく、分類を示す表である。事実上メモリシステムの全ブロックが図に示されたとおりに分類される。この例では3つの別々のリスト、すなわち不完全ブロックのリストと、用済みブロックのリストと、消去済みブロックのリストとを保守する。

【0020】

この例で認識されるブロックのタイプは次のとおりである。

「プログラムブロック」は部分的にプログラムされ、かつただひとつのファイルの有効データを収容する。このブロックにはある程度の消去済み容量が残存する。これはある程度の用済みデータを収容することもある。

「共通ブロック」は部分的にプログラムされ、かつ2つ以上のファイルの有効データを収容する。ある程度の消去済み容量が残存する。ある程度の用済みデータを収容することもある。

40

「フル共通ブロック」は完全にプログラムされ、かつ2つ以上のファイルの有効データを収容する。ある程度の用済みデータを収容することもある。

「ファイルブロック」は完全にプログラムされ、かつ1つのファイルの有効データを収容する。ある程度の用済みデータを収容することもある。

「無効ブロック」は有効データを収容しない。無効ブロックは少なくともある程度は用済みデータを収容し、消去済み容量を収容することもあるが、有効データは収容しない。

「消去済みブロック」はブロックの全容量がプログラムされておらず、データを受け付けることができる。消去済みブロックの中にデータはない。メモリがデータで満たされて

50



いるかまたはほぼ満たされている場合は通常、使用中のブロックの中にある使われていない容量を連続的に再生することによって特定の最小消去済みブロック数のプールを維持する。

#### 【 0 0 2 1 】

この例を目的とし、前述したタイプを持つブロックは次に示す3つの分類に分類される。

「不完全ブロック」はある程度のプログラムされていない容量を収容し、1つ以上のファイルの有効データを収容し、かつある程度を用済みデータを収容することがある。プログラムブロックと共通ブロックは不完全ブロックの例である。

「用済みブロック」は、ある程度を用済みブロックを収容するファイルブロックまたはフル共通ブロックである。用済みブロックは消去済み容量を収容せず、有効データと用済みデータの両方を収容する。

「消去済みブロック」の定義は同一名のブロックタイプと同じであり、つまりデータがないブロックである。

#### 【 0 0 2 2 】

メモリアレイの全てのブロックがこれらの分類に入るわけではないことと、ブロックによってはこれらの分類から外れることがあることと、またこのようなブロックがこの例のリストに出現しないこととに留意すべきである。

#### 【 0 0 2 3 】

記憶容量を再生するためのブロックを選択する場合、主な要因となるのは、ブロック内の有効データ量（有効データ容量）である。ブロックに対する再生操作にあたってはこれの有効データを別のブロックへコピーする必要があるから、有効データ量が最も少ないブロックが最初に選ばれる。というのは、データコピーには時間がかかり、データをプログラムし読み出すにあたってメモリシステムの効率的な動作に支障をきたすおそれがあるからである。容量の大半が消去済みでデータを記憶するために使用できるブロックの利点は、いくぶんこれに相反する。有用な消去済み記憶容量を持つブロックを再生することによって得られる記憶空間の増大はさほど大きくないから、そのようなブロックは再生操作にとって好ましくない。この特定の例の不完全および用済みブロックリストは、ブロックに収容された有効データ量（ページ数）と消去済みの量（ページ数）とに基づきブロックを分類するために使用される。ここで説明するこれらのリストの目的は、再生操作に供されるブロックを一度に1つずつ選択することにある。

#### 【 0 0 2 4 】

##### ブロックリストのタイプとアクセス

不完全ブロックリスト（Pリスト）は、システム内の全ての不完全ブロックにつき、すなわちある程度の有効データとある程度の消去済み容量の両方を収容する全ブロックにつき、項目を収容する。ある程度を用済みデータを収容することもある。

#### 【 0 0 2 5 】

用済みブロックリスト（Oリスト）は、不完全ブロックリストに項目を持たない、用済みデータを収容する、システム内の全ブロックにつき項目を収容する。

#### 【 0 0 2 6 】

消去済みブロックリスト（Eリスト）は、システム内の全消去済みブロックにつき項目を収容する。

#### 【 0 0 2 7 】

フラッシュメモリ操作の一具体例において、次のブロックリストを保守する。

P（V）リスト：ブロックに記憶される有効データ容量に従って項目が序列される不完全ブロックリスト

P（A）リスト：ブロックのブロックアドレスに従って項目が序列される不完全ブロックリスト

O（V）リスト：ブロックに記憶される有効データ容量に従って項目が序列される用済みブロックリスト

O ( A ) リスト：ブロックのブロックアドレスに従って項目が序列される用済みブロックリスト

E リスト：消去済みブロックリスト

【 0 0 2 8 】

P ( V ) リスト、O ( V ) リスト、P ( A ) リスト、O ( A ) リストとは、ブロックレコードの共通セットに対し「ディレクトリ」の役割を果たす。再生ブロックを選択するには、有効データ容量値が最も低い項目が P ( V ) リストと O ( V ) リストとから読み出される。アクティブブロックを選択するには、最も高い有効データ容量値が目標値に満たないかまたはこれに等しい項目が P ( V ) リストから読み出される。消去済みブロックを選択するには、E リスト上の最初の項目が読み出され、リストから削除される。

10

【 0 0 2 9 】

P リストまたは O リストで項目を更新するには、ターゲットブロックアドレスを持つ項目が P ( A ) リストまたは O ( A ) リストから読み出される。

【 0 0 3 0 】

#### ブロックリストの記憶法

前述したセクションで明らかにしたブロックリスト上のブロックに関する情報を記憶するために、1 つ以上のブロックまたはメタブロックが割り当てられる。これらはレコードブロックとして知られ、書き込みまたは更新が 1 ページ単位で行われる。レコードブロック内の情報のために内容アドレス指定が使用される。

【 0 0 3 1 】

20

各々のレコードブロックは一定数の論理ページを収容し、論理ページの各々は、使用可能な次の物理ページへこれを再度書き込むことによって更新できる。リスト内のブロックについて情報を収容するレコードを記憶するため、複数の論理ページが割り当てられる。これらはレコードページとして知られている。

【 0 0 3 2 】

レコードは、ブロックリスト内の全ブロックにつき存在する。レコードページとこれの中にあるレコードはどんな順序でもよい。異なるブロックリストの中にあるブロックのレコードを別々にする必要はない。レコードページの中には用済みレコードが存在することがあり、これは、ブロックリストへ追加されたブロックの新しいレコードに置き換えることができる。レコードは、ブロックの物理アドレスと、ブロックの中にある有効データ量 ( 有効データ容量 ) を定める値とを、他の情報と併せて収容する。

30

【 0 0 3 3 】

個々のブロックリストの 1 ブロックリストの中でブロックを識別する項目を記憶するため、1 つ以上の論理ページが割り当てられる。これらはリストページとして知られている。異なるブロックリストのための別々のリストページが使用される。リストページ内の各項目は、記述子値を、レコードページ内のブロックレコードのポインタと併せて収容し、記述子値は、対応するブロックの物理アドレスか、またはブロック内の有効データ容量の値 ( またはブロック内のデータに関連する他の何らかの値 ) である。レコードに対応するブロックが 2 つのブロックリストに現れる場合は、異なるブロックリストの異なるリストページにある 2 つのリスト項目が同じレコードを指示する。

40

【 0 0 3 4 】

ブロックリスト内の項目は、ブロックのレコードの共通セットに対し「ディレクトリ項目」の役割を果たす。

【 0 0 3 5 】

不完全ブロックリストまたは用済みブロックリストのリストページでは、それらの記述子の順序で項目が記憶される。項目は稠密な形式で書き込まれ、リストページ内の書き込まれていない項目位置は通常、書き込まれた最後の項目の後ろにある。リストページは重複しない記述子範囲を記憶する。リストページが満杯になると、新しい論理ページをリストページとして割り当てることによってその項目を 2 つのリストページに分けることができる。同様に、記述子範囲が隣接する 2 つのリストページの項目数がしきい値を下回る場合

50

は、2つのリストページを1つにまとめることができる。レコードブロックのインデックスは、各リストページの最初の項目の記述子を収容する。

#### 【0036】

不完全ブロックリストまたは用済みブロックリストの中でブロックアドレスかまたは有効データ容量の目標値を持つ項目は、レコードブロックインデックスから該当する記述子範囲を持つリストページを識別することにより、そのページをフラッシュから読み出し、次いでそのページの中で目標項目のリニア探索かまたはバイナリ探索を遂行することにより、見つけることができる。そして、この項目より識別されるレコードページからターゲットブロックのレコードを読み出すことができる。

#### 【0037】

消去済みブロックリスト内のブロックの項目はリストページに記憶でき、ここでそれらは、それらが書き込まれた順序を保つ。消去済みブロックは常にリストの中で最も古い項目として選択される。

#### 【0038】

#### レコードブロックの構造

ブロックリストの中で参照されるブロックの全ての項目とインデックス情報は、フラッシュメモリの1つ以上のレコードブロックに収容される。レコードブロックはメタブロックであり、ページ単位で更新される。

レコードブロックは次の特性を持つ。

1. あらゆるタイプのブロックリストをまとめて1つのレコードブロックに記憶できる。
2. 必要に応じ、複数のレコードブロックを使用できる。
3. レコードブロックは特定の数の論理ページを持ち、これはこの例においてブロック内の物理ページ数の25%に規定される。
4. 最後に書き込まれたページのレコードブロックインデックスセクションは、各論理ページから物理ページへのマッピングを提供する。
5. 論理ページは、使用可能な次の物理ページにこれを再度書き込むことによって更新できる。
6. ブロックリストのため、またはブロックのレコードのために使用されるあらゆるタイプのページに論理ページを割り当てることができる。
7. レコードブロックは、満杯のときに圧縮され消去済みブロックに再度書き込まれる。

#### 【0039】

図2にはレコードブロックの一例の構造が示されている。

#### 【0040】

#### レコードブロックインデックス

レコードブロックインデックスは、レコードブロックの中にある各ページの1セクションとして存在する。これは最後に書き込まれたページの中にあるものだけが有効である。レコードブロックインデックスは可能な各論理ページにつき項目を収容し、論理ページ番号に従って序列される。各項目は次に示す3つのフィールドを持つ。

1. ページタイプを識別する数字コード
  - a. P(V)リストページ
  - b. P(A)リストページ
  - c. O(V)リストページ
  - d. O(A)リストページ
  - e. Eリストページ
  - f. レコードページ
  - g. 割り当てされていない論理ページ
2. ページ内の最初の項目の値(これにより、P(V)、P(A)、O(V)、O(A)リストページの各タイプで値の範囲を設定し、キャッシュできる。)
3. 論理ページのマップ先にあたる物理ページのポインタ

#### 【0041】

## レコードページ

レコードはそのリストのうちの 1 つの項目をブロックに關係する必要情報を全て収容し、レコードページに記憶される。レコードページは次に示す 3 つのセクションに細分される。

1. 項目状態
2. レコード
3. 共通ブロックレコード

### 【 0 0 4 2 】

項目状態セクションは、レコードが使用中か、それとも新しいブロックに割り当てることができるかを指示するビットマップを備える。レコードセクションはリスト内の全ブロックにつき一定サイズの項目を有し、フィールドはブロックの属性を次のとおりに規定する。

1. ブロックアドレス
2. ブロック内の有効データ容量
3. ブロック内のページ書き込みポインタの位置
4. ブロック内の最初のデータグループのファイル ID
5. ブロック内にデータが存在するファイルの合計数
6. ブロックの共通ブロックレコードへのオフセット ( 値 0 は共通ブロックレコードが存在しないことを意味する。 )

### 【 0 0 4 3 】

共通ブロックレコードセクションの項目のサイズは不定であり、フィールドは共通ブロック内の他のファイル ID を次のとおりに規定する。

1. 共通ブロックにおける後続データグループのファイル ID
2. レコード終了インジケータ

### 【 0 0 4 4 】

ブロックリストからブロックが削除される場合、レコードページは用済み項目を収容する。このようなレコードは、リストに追加される新しいブロックに再び割り当てることができる。

### 【 0 0 4 5 】

ブロックに割り当てられるレコードブロックの中の論理ページ番号とそのページ内のレコード番号は、レコードを参照するためにリストページによって使用されるため、通常は変更されない。しかし、ブロックのレコードを同一のリストページ内の別のレコード番号、別のリストページ、または別のレコードブロックへ移すことは許される。ブロックのレコードが移される場合、いずれかのリスト項目でそのポインタを相応に更新しなければならない。

### 【 0 0 4 6 】

レコードページが修正され再度書き込まれる場合、用済みの空間をなくすために共通ブロックレコードを圧縮でき、共通ブロックレコードへのオフセットの変化を反映させるためにレコードを更新できる。

### 【 0 0 4 7 】

レコードと共通ブロックレコードとの間の境界は動的である。

### 【 0 0 4 8 】

図 3 にはブロックレコードページの一例の構造が示されている。

### 【 0 0 4 9 】

## リストページ

リストページは、1 組のブロックの項目を、記述子値によって規定される順序で収容する。P ( V ) リストと O ( V ) リスト内の項目における記述子は有効データ容量であり、P ( A ) リストと O ( A ) リスト内の項目における記述子はブロックアドレスである。

### 【 0 0 5 0 】

有効項目はリストページの中で一連の項目位置を占めるが、ページ全体を埋め尽くす必

10

20

30

40

50

要はない。この中に用済み項目はなく、記述子値は連続しなくともよい。

【 0 0 5 1 】

リストページ内の項目は、記述子フィールド内の値の順序に並ぶ。記述子値の範囲は、他のリストページ内の記述子値の範囲と重複しない。

【 0 0 5 2 】

ブロックリストへ追加されたブロックのための項目を挿入する必要がある場合、その新しいブロックの記述子値を記述子範囲に含むリストページを識別する。新しい項目は記述子範囲内の適切な位置に挿入され、リストページは再度書き込まれる。項目を削除しなければならない場合、その項目を省いてリストページが圧縮され、再度書き込まれる。

【 0 0 5 3 】

満杯になったリストページへ追加を行わなければならない場合、空き論理ページが新しいリストページとして割り当てられ、満杯のリストページの記述子範囲は、2つのほぼ等しい重複しない範囲に分割され、それらは2つの使用可能なリストページに書き込まれる。

【 0 0 5 4 】

記述子範囲が隣接する2つのリストページで総有効項目数がしきい値（この例では、リストページにおける項目位置数の70%）を下回ると、2つのリストページの範囲は統合され、2つのリストページのうちの一方に書き込まれる。次いで、他方の使用されていないページは空き論理ページとなる。

【 0 0 5 5 】

P(V)リストとO(V)リストの場合、リストページ内の項目内のフィールドは次のとおりである。

1. ブロック内の有効データ容量
2. レコードページ内のブロックのレコードを指すポインタ（レコードページはリストページと同じレコードブロックの中になくてもよい。）

【 0 0 5 6 】

P(A)リストとO(A)リストの場合、リストページ内の項目内のフィールドは次のとおりである。

1. ブロックアドレス
2. レコードページ内のブロックのレコードを指すポインタ（レコードページはリストページと同じレコードブロックの中になくてもよい。）

【 0 0 5 7 】

Eリストの場合、リストページ内の項目内のフィールドは次のとおりである。

1. ブロックアドレス

【 0 0 5 8 】

レコードのアクセスシーケンス

PリストまたはOリストで目標ブロックのレコードにアクセスするには、次のステップからなるシーケンスを使用する。

1. P(V)リスト、O(V)リスト、P(A)リスト、またはO(A)リストを目標リストとして設定する。
2. レコードブロックの中で最後に書き込まれたページからレコードブロックインデックスを読み出す。この情報はあらかじめキャッシュの中に存在することがある。
3. ステップ1で規定した目標リストで目標記述子値のリストページに割り当てられた論理ページ番号を判定する。
4. ステップ3で判定した論理ページ番号をレコードブロックから読み出す。
5. 目標ブロックの項目を読み出すために、ステップ4で読み出したリストページを探索する。
6. ステップ5で読み出した項目によって規定されるレコードブロックからレコードページを読み出す。
7. ステップ6で読み出したレコードページからターゲットブロックのレコードを読み出

10

20

30

40

50

す。

#### 【 0 0 5 9 】

##### 詳細な第 2 の例

第 2 の例では前述した第 1 の例と同じく、特定のブロックが、そこに収容されるデータに従い、それぞれ分類され、これらのブロックのためのレコードが保守される。リストは、ブロックに記憶されるデータに関する記述子値に従って序列され、保守される。ブロックの中にある有効データ量は、このような記述子値の一例である。しかし、この第 2 の例で、ブロックの管理に用いる構造および方法のいくつかは異なる。第 1 および第 2 の例の構造および方法は代案とみなすべきものであり、両例の構造および / または手法の様々な組み合わせもまた本発明の開示の一部とみなされる。第 2 の例は、第 1 の例との違いに焦点をあてながら説明する。よって、両例に共通の要素は第 2 の例に関して重ねて詳述しない場合がある。

#### 【 0 0 6 0 】

第 2 の例のブロック分類は第 1 の例のブロック分類と同じであり、「コンプリート共通ブロック分類」が追加される。図 4 は、追加の分類「コンプリート共通ブロック」を除いて図 1 とほぼ同じブロック分類表を示す。「不完全ブロック」、「用済みブロック」、「消去済みブロック」、または「コンプリート共通ブロック」のブロック分類を有する各ブロックにつきレコード項目を保守する。分類「コンプリート共通ブロック」は、消去済み容量を収容せず、かつ用済みデータを収容しない共通ブロックに使われる。コンプリート共通ブロック ( C C B ) のレコードは、全コンプリート共通ブロックのブロックレコードにより保守される。再生のために使用できる空間を持たない ( 消去済みの空間も用済みの空間も持たない ) コンプリート共通ブロックは通常、再生操作に供されない。しかし、コンプリート共通ブロックの中である程度のデータが用済みになると、そのブロックは用済みブロックとして再分類され、再生操作に供されることがある。そのような再分類が行われる場合には、ブロックに記憶されるデータの関連情報を収容するブロックレコードが必要である。この情報はブロックの既存の C C B レコード項目から使用可能である。したがって、コンプリート共通ブロックのレコードを保守すれば、ブロックのレコードを生成するために情報を探索するという重い負担を伴わずにコンプリート共通ブロックから用済みブロックへの移行を果たすことができる。

#### 【 0 0 6 1 】

図 1 および 4 の分類方式は例示的なものであり、これ以外の方式も予期される。( 後ほど詳述する ) 一例において、メモリアレイの中にある全ブロックのために常時レコードを保守する。この場合は、用済みデータを持たないファイルブロックのための追加のブロック分類を図 4 の表に加えることができる。別の例では、図 4 の分類のいくつかは必要とされない。例えば、消去済みブロックや無効ブロックのレコードは保守しなくてよい。さらに別の例では、図 4 とは異なるブロックタイプにブロックを分けることができる。図 4 のブロックタイプが特定のメモリ管理方式にとって好都合であるが、他のメモリ管理方式で異なるブロックタイプを使用することも理解できよう。

#### 【 0 0 6 2 】

図 4 の表に記載されたブロック分類のいずれか 1 つに該当する全ブロックにつきレコードを保守する。ブロック分類が異なるブロックのレコードを同じページにまとめて記憶できる。この例では、ブロックレコードだけを記憶する専用のブロックレコードブロックを保守する。レコードブロック内の個々のレコード項目へのアクセスを促進するため、ブロックディレクトリを保守する。( ブロックディレクトリとブロックレコードの両ページをただひとつのブロックで収容する第一の例と違って ) ブロックディレクトリとブロックレコードは、フラッシュメモリの中で別個の 1 組のブロックに記憶される。図 5 は、ディレクトリブロック 5 0 3 の中にあるブロックディレクトリ項目 5 0 1 を示し、このブロックディレクトリ項目は、ブロックレコードブロック 5 0 9 の中にある対応するブロックレコード 5 0 7 の位置を指すポインタ 5 0 5 を含む。ブロックディレクトリ項目 5 0 1 はブロックアドレス 5 0 6 をも含む。

## 【 0 0 6 3 】

ブロックディレクトリは、ブロックレコードに項目が存在する各ブロックにつき1つのブロックディレクトリ項目を収容する。ブロックディレクトリ項目は重複しないブロックアドレス値範囲の中で記憶され、各々の範囲は別々のブロックディレクトリページに割り当てられる。範囲の中ではブロックアドレス値に従って項目が序列される。目標ブロックアドレスのブロックディレクトリ項目は、1つのブロックディレクトリページを読み出すことにより、またそのページの中でバイナリ探索を遂行することにより、見つけることができる。このようにしてブロックディレクトリは、ブロックアドレスに従って特定のブロックレコード項目を見つける簡便な手段を提供する。

## 【 0 0 6 4 】

用途によっては、ブロックアドレス以外の基準によってブロックを探索することが望ましい。場合によっては、ブロックに記憶されるデータに関連する記述子値をこのような探索に使用できる。例えば、再生の目的で、有効データが最も少ない不完全ブロックを識別することが望まれるかもしれない。このようなブロックを見つける一方法として、収容された有効データ量が最も少ないブロックがどれなのかを判定するために全ブロックのレコード項目を探索する。しかし、そのような探索によって多大な負担が加わるかもしれない。一代案では、収容される有効データ量（有効データ容量）に従ってブロックが序列されるブロックのリストを保守する。この場合は収容された有効データ量の順にブロックがリストされるから、有効データ量が最も少ないブロックを識別するには、そのリストの中の最初（または最後）の項目を読み出すだけでよい。同様に、一定量の有効データを持つブロックが必要ならば、バイナリ探索でこのようなブロックを速やかに識別できる。ブロックに記憶される有効データ量はリスト項目の中の記述子値によって与えられる。このような記述子値は、ブロックに記憶されるデータを記述する。これは、ブロックの物理位置を記述する（第一の例で記述子値として使われる）ブロックアドレスと対照をなす。

## 【 0 0 6 5 】

レコードのフィールドの中で規定されるブロック内の有効データ量の記述子値に従ってブロックレコードにアクセスするため、2段階探索プロセスを取り入れたメカニズムが提供される。この内容アドレス指定メカニズムを使用できるブロック分類の場合、有効データ容量値を収容する項目は、ブロックディレクトリの中で別々のリストページに記憶される。これらのブロックリスト項目は重複しない有効データ容量値範囲の中で記憶され、各々の範囲は別々のブロックリストページに割り当てられる。範囲の中では有効データ容量値に従って項目が序列される。各々のブロックリスト項目はブロックアドレスを収容し、そのアドレスは明確にブロックディレクトリ項目を識別する。ブロックリスト項目511はブロックアドレス513を収容し、これはブロックアドレス506と同じであり、よってブロックディレクトリ項目501を識別する。ブロックリスト項目511はまた、ブロックアドレス513を有するブロックの有効データ容量515を収容する。目標有効データ容量値のブロックディレクトリ項目は、1つのリストページを読み出すことにより、そのページの中でバイナリ探索を遂行して目標有効データ容量値を持つ項目を見つけることにより、さらに1つのディレクトリページを読み出すことにより、次いでそのディレクトリページの中でバイナリ探索を遂行して目標ブロックアドレスを持つ項目を見つけることにより、見つけることができる。

## 【 0 0 6 6 】

バイナリ探索は、単にページの記述子値範囲の中間にある項目を調べることを意味する場合がある。この項目の記述子値と探している記述子値との比較に基づき、探索はページの半分に限定される。そしてこの半ページの間にある項目も同様に調べ、探索はページの4分の1に限定される。連続するステップの後には、探している記述子値を有する1つ以上の項目が見つかる。別の例では、探している記述子値がリストの中で最下位（または最上位）であるため、バイナリ探索は必要ない。したがって、リスト内の最初（または最後）の項目が選択される。

## 【 0 0 6 7 】

ブロックレコードはブロックディレクトリの中の項目によって直接アドレス指定できる。ブロックレコードのページは、同じまたは別のブロックレコードブロック内のプログラムされていない位置へページを移す読み出し / 修正 / 書き込み操作によって更新される。ページ内のブロックレコードはどれも、同じブロックディレクトリページの中にある項目に対応しなければならない。しかし、1つのディレクトリページで複数のブロックレコードページの項目を収容できる。よって、ブロックレコードページの更新にあたっては、ただひとつのブロックディレクトリページを修正すればよい。

【0068】

#### ブロックディレクトリ

ブロックディレクトリは、ブロックアドレスによってブロックを識別し、対応するブロックレコードの位置を指示する、序列された項目の集まりである。ブロックレコードの保守が行われる各ブロックにつき1項目がブロックディレクトリの中に存在する。この例では、不完全ブロック、用済みブロック、コンプリート共通ブロック、および消去済みブロックの各ブロックにつき1項目が存在する。ブロックディレクトリは1つ以上のディレクトリブロックに収容される。

【0069】

図6は、用済みブロックと不完全ブロックのリストページを収容するディレクトリブロック621を示している。図6はまた、ディレクトリブロックの中にあるディレクトリページを示している。各ディレクトリブロックは一定数の論理ページを収容し、この論理ページの各々は、これを使用可能な次の物理ページへ再度書き込むことによって更新できる。有効項目を収容するページには論理ページ番号が割り当てられる。この例で、ディレクトリブロック内の論理ページ数はブロック内の物理ページ数の25%に指定される。他の例ではこれとは別の制限を指定できる。ディレクトリブロックの最後のページが書き込まれた後には、有効ページの全てを消去済みブロックへ書き込むことにより、そして元のディレクトリブロックを消去することにより、ブロックが圧縮される。

【0070】

#### ブロックディレクトリページ

ブロックディレクトリページは、1組のブロックディレクトリ項目を、それらのブロックアドレス値の順に収容する。図7Aにはブロックディレクトリページ731の一例が示されている。有効ブロックディレクトリ項目733はブロックディレクトリページ731の中で一連の項目位置を占めるが、ページ全体を埋め尽くす必要はないため、消去済みの空間が残ることがある。各々のブロックディレクトリページはレコードインデックス（後述する）を収容する。ここで、ブロックディレクトリページ731はレコードインデックス735を収容する。ブロックディレクトリ項目733の中に用済み項目はなく、ブロックアドレス値は連続しなくともよい。ある1つのブロックディレクトリページにおけるブロックアドレス値の範囲は、別のブロックディレクトリページにおけるブロックアドレス値の範囲に重複しない。

【0071】

ブロックの項目を挿入する必要がある場合は、その新しいブロックのブロックアドレス値をブロックアドレス範囲に含むブロックディレクトリページが、ディレクトリインデックスの情報から識別される。新しい項目はブロックアドレス範囲の中の適切な位置に挿入され、ブロックディレクトリページは再度書き込まれる。項目を削除しなければならない場合は、その項目を省いてブロックディレクトリページが圧縮され、再度書き込まれる。

【0072】

満杯になったブロックディレクトリページへ追加を行わなければならない場合は、空き論理ページが新しいブロックディレクトリページとして割り当てられ、満杯になったブロックディレクトリページのブロックアドレス範囲は2つのほぼ等しい重複しない範囲に分割され、それらは2つの使用可能なブロックディレクトリページに書き込まれる。

【0073】

ブロックアドレス範囲が隣接する2つのブロックディレクトリページの総有効項目数が

10

20

30

40

50



しきい値（この例では、1ブロックディレクトリページにおける項目位置数の70%）を下回ると、2つのブロックディレクトリページの範囲は統合され、2つのブロックディレクトリページのうちの一方に書き込まれる。次いで、他方の未使用ページは空き論理ページとなる。

#### 【0074】

この例のブロックディレクトリ項目737は2つのフィールド、すなわち（1）ブロックアドレスと、（2）対応するブロックレコードのポインタとを収容する。このポインタは、ブロックレコードページの論理識別子と、ページ内のある特定のブロックレコードのバイトオフセットとを識別する。ブロックレコードページ論理識別子は、同じブロックディレクトリページの中にある項目によって参照される最高16個のブロックレコードページのうちの1つを識別する。これは、その項目を収容するブロックディレクトリページの中のレコードインデックスフィールドによって物理ブロックアドレスとページ番号とへ変換される。バイトオフセットは、識別されたブロックレコードページの中でブロックレコードの位置を識別する。

#### 【0075】

各々の有効ブロックディレクトリページとブロックリストページの中には単独の有効レコードインデックスフィールドが存在する。これは、ブロックレコードページの論理識別子をブロックレコードページが位置するところのページ番号と物理ブロックアドレスとに変換するために使われる。レコードインデックス735は、ブロックディレクトリページ731のいずれかの項目の中で使われる各論理識別子につき1つの項目（例えば、項目739）を収容する。1つのブロックディレクトリページの中でブロックディレクトリ項目により最高16個のブロックレコードページが参照される。したがって、4ビット論理識別子が使用される。かくして個々のブロックディレクトリ項目は、対応するブロックレコードページの長い物理ページ位置の代わりに4ビット識別子を使用できる。レコードインデックスフィールドは、ブロックディレクトリページ内の全項目のためにこれらの論理識別子を変換する役割を果たす。

#### 【0076】

有効ディレクトリインデックスフィールド741は、最後に書き込まれたブロックディレクトリまたはブロックリストページにのみ存在する。それ以前に書き込まれたページ内のディレクトリインデックスフィールドの情報は用済みである。この目的は、ブロックディレクトリ項目とブロックリスト項目の序列と、論理ページから物理ページへのマッピングとを支援することにある。これが提供する構造では、ブロックディレクトリブロック内の個々のページに関する最新データが記憶される。ディレクトリインデックスは、論理ページ番号に従って序列される項目743などの項目を各可能な論理ページにつき収容する。各項目には4つのフィールドがある。

（1）論理ページの割り当て状態フラグ

（2）ページのタイプ、例えばブロックディレクトリ、PBリスト、またはOBリスト

（3）ブロックディレクトリページ内の最初の項目のブロックアドレス、またはリストページ（PBまたはOB）内の最初の項目の有効データ容量値（これにより各論理ページでブロックアドレスまたは有効データ値の範囲を確立し、キャッシュできる。）

（4）論理ページのマッピングにあたるブロックディレクトリの中の物理ページへのポインタ

#### 【0077】

#### ブロックリストページ

ブロックリストページは、1ブロック分類につき1組のブロックリスト項目を、ブロックに収容されたデータ（例えば、ブロックに収容された有効データ量）を記述する記述子値の順に、収容する。図7Bにはブロックリストページ751の一例が示されている。この例で、ブロックリストページはPBリストページまたはOBリストページである。図7BはOBリストページ751を示している。有効ブロックリスト項目753はブロックリストページ751の中で一連の項目位置を占めるが、ページ全体を埋め尽くす必要はない

10

20

30

40

50

。ブロックリストページの中には通常、用済み項目はない。ブロックリスト項目 7 5 3 は記述子値によって序列されるが、記述子値は連続しなくともよく、繰り返すことができる。この例で、有効データ容量値は連続しなくともよく、繰り返すことができる。ある 1 つのブロックリストページの記述子値の範囲は、同一ブロック分類の別のブロックリストページの記述子値の範囲に重複しない。

#### 【 0 0 7 8 】

この例で使われる記述子値は有効データ容量であるが、別の記述子値を使用することもできる。例えば、ブロック内の消去済み容量を記述子値として使用できる。再生のためブロックを所望の順序でリストするため、有効データ量と消去済み容量の組み合わせから記述子値を導き出すことができる。リストは、場合によっては重複してよい。この場合、2 つの異なるリストの中に同じブロックが現れることがある。例えば、ブロックを、そこに収容される有効データ量に従ってリストすると同時に、（別のリストでは）そこに収容される消去済みの空間の量に従ってリストすることができる。

#### 【 0 0 7 9 】

ブロックの項目を挿入する必要がある場合は、その新しいブロックの有効データ容量値を有効データ容量範囲に含むブロックリストページが、ディレクトリインデックスの情報から識別される。ブロックリストページは再度書き込まれ、新しい項目は、この有効データ容量値に従ってページの適切な位置に挿入される。項目を削除しなければならない場合、その項目を省いて、新しい物理位置にブロックリストページが再度書き込まれる。

#### 【 0 0 8 0 】

満杯になったブロックリストページへ追加を行わなければならない場合、空き論理ページが新しいブロックリストページとして割り当てられ、満杯になったブロックリストページの有効データ範囲は 2 つのほぼ等しい重複しない範囲に分割され、それらは 2 つの使用可能なブロックリストページに書き込まれる。

#### 【 0 0 8 1 】

範囲が隣接する 2 つのブロックリストページの総有効項目数が所定のしきい値（この例では、1 ブロックリストページにおける項目位置数の 7 0 % ）を下回ると、2 つのブロックリストページの範囲は統合され、2 つのブロックリストページのうちの一方に書き込まれる。そして、他方の未使用ページは空き論理ページとなる。

#### 【 0 0 8 2 】

ブロックリスト項目 7 5 5 は 2 つのフィールド、すなわち（ 1 ）ブロックアドレスと、（ 2 ）記述子値とを収容し、この例における記述子値は、ブロック内の有効データ量を示す値である。第 1 の例と違って、ブロックアドレスによって序列されるリストはない（しかし、ディレクトリはブロックアドレスによって序列される）。この例において、リストはブロックアドレスを含み、このブロックアドレスからディレクトリ項目が見つかり、さらにこのディレクトリ項目が対応するレコードの位置を順に指示する。よって、この例において、リスト項目 7 5 3 はレコードを直接指示しない。ブロックリストはディレクトリインデックスを収容できるが、有効なディレクトリインデックスを収容するのは最後に書き込まれたページだけである。図 7 B の O B リストページは用済みのディレクトリインデックス 7 5 7 を収容する。

#### 【 0 0 8 3 】

#### ブロックレコード

ブロックレコードはレコードの集まりであり、各レコードは、ブロックアドレスによって識別されるブロックの情報を収容する。各ブロックディレクトリ項目につき 1 つのレコードが存在する。ブロックレコードはブロックディレクトリ項目によってアドレス指定され、ブロックレコードページが修正されるときにはブロックディレクトリページを修正しなければならない。

#### 【 0 0 8 4 】

ブロックレコードは、図 8 に示すレコードブロック 8 6 1 などのような、1 つ以上の専用レコードブロックに収容される。第 1 の例と違って、ブロックリストとブロックレコー

10

20

30

40

50

ドは同じブロックの中にまとめて記憶されない。ただひとつのブロックレコードブロックがプログラムされていないページを収容し、ここへブロックレコードを書き込むことができる。ブロックレコード情報はどれも、ブロックレコード書き込みポインタによって識別されるこのブロックの中のプログラムされていない次のページ位置にプログラムされる。ブロック内の最後のページがプログラムされると、ブロックレコード書き込みポインタは消去済みブロックの最初のページへ移される。ブロックレコードブロックは、ブロックレコードページが再度書き込まれるときに用済みページを収容することができる。いくつかの実施形態において、有効ブロックレコードページは、ブロックレコードページが再度書き込まれてページ内のレコードが常に用済みになるときに用済みレコードを収容しない。別の実施形態において、用済みレコードが有効ブロックレコードページに残ることがある。しかし、用済みレコードのディレクトリ項目は削除されるかまたは有効レコードを指示する項目に置き換えられるので、用済みレコードはアクセスされない。レコードページの中の用済みレコードは、レコードページが再度書き込まれるときにコピーされない。

10

#### 【0085】

ブロックレコードページは、1つのブロックディレクトリページの中にあるブロックディレクトリ項目によって参照される1組のブロックレコードを、このブロックディレクトリページ内の項目と同じ順序で収容する。図9は、ブロックレコードページ965の一例を示している。ブロックディレクトリページは複数のブロックレコードページを参照できる。ブロックレコードページの修正にあたっては、ただひとつのブロックディレクトリページを修正すればよい。第1の例と違って、ブロックレコードページはブロックディレクトリ項目によって直接識別されるため、この例のブロックレコードページはブロックレコードインデックスを含まない。

20

#### 【0086】

ブロックレコードページは、これを読み出すことにより、そして1つ以上のブロックレコードを更新または追加することにより修正できる。いずれの用済みブロックレコードもページの圧縮によって削除され、ページは、ブロックレコード書き込みポインタによって識別される位置にプログラムされる。

#### 【0087】

ブロックレコードページヘッダは、ブロックレコードページに関連するブロックディレクトリページへの参照符と、このブロックレコードページにおけるブロックレコード情報の長さとを記憶する。ブロックレコードページヘッダはまた、ブロックレコードページが書き込まれたときにブロックレコードブロックの各々に存在する用済みページ数のレコードを記憶する。この情報は、最後に書き込まれたブロックレコードページヘッダの中にあるものだけが有効である。

30

#### 【0088】

個々のブロックレコード項目のサイズは不定である。よって、コンプリート共通ブロックのレコードは消去済みブロックのレコードより大きくなることもある。第1の例と違って、別個の共通ブロックレコード領域は必要ない。この例のレコードブロックは、ブロックの属性を定義する次のフィールドを持つ。

(1) ブロックアドレス

40

(2) ブロックのタイプ、PB、OB、CCB、またはEB

(3) ブロック内の有効データ容量

(4) ブロック内のページ書き込みポインタの位置

(5) ブロック内にデータが存在するファイルの合計数

(6) ブロック内にデータが存在する各ファイルのファイルID

#### 【0089】

他の例では、異なる記述子値を含む異なるフィールドをレコードに収容してよい。

#### 【0090】

### 第2の例におけるブロックレコードの再生プロセス

ブロックレコードは1つ以上のレコードブロックの中に収容され、ブロックディレクト

50

リ項目によって直接アドレス指定される。新規または更新済みブロックレコードをプログラムするにあたって使用できる消去済みページは、ただひとつのレコードブロックに収容される。それ以外の全てのレコードブロックで全てのページがプログラムされていたとしても、完全に用済みまたは部分的に用済みのページがブロックに収容されることがある。用済みブロックレコードによって占められる容量の再生は、再生される次のブロックとして1つのレコードブロックを指定することにより、そしてこの再生ブロックからブロックレコード書き込みポインタによって現在指定されているページへページを徐々にコピーすることにより、その後再生ブロックを消去することにより、遂行される。

#### 【0091】

レコードブロックに対する前の再生プロセスが完了し、前の再生ブロックが消去されると、再生ブロックが選択される。用済みページ数が最高のレコードブロックが再生ブロックとして選択される。各レコードブロックの用済みページ数は、最後に書き込まれたブロックレコードページのブロックレコードページヘッダに記録される。ブロックレコード書き込みポインタを収容するレコードブロックは再生のために選択されないことがある。選択されたレコードブロックは、再生プロセスが完了してブロックが消去されるまで再生ブロックであり続ける。次いで、消去されたブロックは消去済みブロックのプールへ加えられ、ホストデータを含むあらゆる類のデータを記憶するために再び使用できる。よって、ブロックは暫くの間専用レコードブロックであり続けるが、レコードブロックとして永久的に指定されるのではない。

#### 【0092】

用済みページを収容するブロックレコードブロックを再生するプロセスでは、ブロックから、有効ブロックレコードを収容する少数のページを、ブロックレコード書き込みポインタによって指定されるページへ、スケジュールされた間隔で段階的にコピーする。良好な性能のため、1段階にコピーされるページの数にはメタページに収容されるページの数になる。しかし、場合によっては、これよりも少ないページが書き込まれる。ブロックレコード書き込みポインタのところでページをプログラムすることは、メタページに対する1回のプログラミング操作として遂行できる。再生ブロックにおけるページの段階的コピー操作は、4つのメタページに収容されるページ位置数を通過するブロックレコード書き込みポインタの進捗によって規定される間隔でスケジュールできる。メタページをプログラムするには、書き込みポインタは、段階的コピー操作がスケジュールされるときに物理メタページの最初のページを指示しなければならない。

#### 【0093】

前述したブロックレコードブロックの再生プロセスとは対照的に、ディレクトリブロックの再生プロセスでは、ディレクトリブロックに消去済みの空間が残っていないときにそのブロックディレクトリブロックを圧縮するだけでよい。常時最大25%（この例の論理容量は物理容量の25%）の有効ページを持つブロックディレクトリブロックを圧縮すると、少なくとも75%の消去済みの空間を有するブロックになる。圧縮が行われると、データのコピー元にあたるブロックは無効ブロックとなり、消去されて消去済みブロックとなる。次いで、このブロックは消去済みブロックのプールへ加えられ、ホストデータを含むあらゆる類のデータを記憶するために使用できる。よって、ブロックは暫くの間専用ディレクトリブロックであり続けるが、ディレクトリブロックとして永久に指定されるのではない。

#### 【0094】

### 第2の例の構造の更新

メモリアレイでブロックにデータが書き込まれるとき、またはファイルが削除されるときには、1つ以上のブロックレコードを更新する必要がある。また、対応するブロックディレクトリおよびリスト項目を更新する必要がある。図10に示された次のプロセスは、第2の例の各種構造の更新を説明するものであり、このとき（既にある程度の用済みデータを収容する）不完全ブロックには追加の有効データが記憶され、ブロックは用済みブロックになる。

- ( 1 ) 追加データを記憶するブロックのアドレスを受け取る。
- ( 2 ) このブロックのディレクトリ項目 1 7 5 を収容するディレクトリページ 1 7 3 の物理ページ位置を判定するため、ブロックディレクトリブロック 1 7 0 の最終書き込みページでディレクトリインデックス 1 7 1 を調べる。
- ( 3 ) ブロックの項目 1 7 5 を見つけるため、ディレクトリページ 1 7 3 の中でバイナリ探索を遂行する。
- ( 4 ) 項目 1 7 5 の論理識別子からレコードインデックス 1 7 7 の情報に沿って、ブロックレコードブロック 1 8 1 の中でこのブロックのレコード 1 8 3 を収容する物理ページ 1 7 9 を判定する。受け取ったアドレスに対応する正しいレコード 1 8 3 を見つけるため、ディレクトリ項目 1 7 5 の中のオフセットを使用する。 10
- ( 5 ) レコード 1 8 3 からブロックの分類を判定する。追加データの記憶によって分類が変化するかどうかを判定する。ここでブロックは不完全ブロックであり、このブロックに残された空間は追加データで満たされるから、このブロックは用済みブロックになる。
- ( 6 ) レコードページ 1 8 3 の内容を、書き込みポインタ 1 8 5 によって指示される位置へコピーし、対象ブロックのレコード 1 8 3 は、異なるタイプのブロック、有効データ、ページ書き込みポインタの位置等を反映させるために更新される。
- ( 7 ) ディレクトリページ 1 7 3 をブロックディレクトリブロック 1 7 0 の使用可能な次の物理ページへコピーする。レコードページの新しい物理位置を反映させるため、更新された項目とレコードインデックスを新しいディレクトリページに書き込む。また、新しい物理ページでディレクトリインデックスを更新する。 20
- ( 8 ) ブロックの項目 1 8 9 を収容する不完全ブロックリストページ 1 8 7 の物理ページ位置を判定するため、ディレクトリインデックスを再び調べる（場合によっては複数のリストを調べる）。
- ( 9 ) 記述子値が対象ブロックのそれに等しいリスト項目 1 8 9 を見つけるため、ページ 1 8 7 の中でバイナリ探索を遂行する。複数のリスト項目が記述子値を有する場合は、ブロックアドレスにより一致する全ての項目を探索する。
- ( 1 0 ) 不完全ブロックリストページ 1 8 7 を新しい位置へコピーし、対象ブロックの項目 1 8 9 は削除される。新しい不完全ブロックページは、不完全ブロックページの新しい物理位置で更新されたディレクトリインデックスを含む。
- ( 1 1 ) 対象ブロックの有効データ量を含む、ブロック範囲当たりの有効データをカバーする用済みブロックリストページの物理ページ位置を判定するため、ディレクトリインデックスを再び調べる（図示せず）。 30
- ( 1 2 ) 用済みブロックリストページをコピーし、対象ブロックの新しい項目を、ブロック内の現在の有効データ量に基づき、適切なオフセットのところに追加する。新しい用済みブロックリストページは、用済みブロックリストの新しい位置を示す更新済みディレクトリインデックスを含む（図示せず）。

#### 【 0 0 9 5 】

前述したステップは必ずしも示されたとおりの順序で遂行されない。いくつかのステップは並行して遂行できる。例えば、更新済みの用済みブロックリストページと更新済み不完全ブロックリストページの書き込みは、メタブロック書き込みの一部として並行してよい。 40

#### 【 0 0 9 6 】

##### 詳細な第 3 の例

第 3 の例では、メモリアレイの各ブロックにつき常時レコードを保守する。これには 1 つ以上の追加のブロック分類が関係し、例えば用済みデータを収容しないファイルブロックのための追加の分類を、図 4 に示された第 2 の例の分類に加えることができる。各ブロックごとにレコードを保守すると保守する各レコードの合計数が増えるが、より簡素な構造を使用することができる。第 3 の例は、ブロックごとにレコードを用意する点を除き、第 2 の例と同様に進行する。

#### 【 0 0 9 7 】

例えば、全てのブロックのためのレコードを保守する場合は、ディレクトリ項目も全てのブロックのために保守する。ディレクトリ項目は一定の均等サイズを有する。したがって、ディレクトリページには、ブロックアドレスによって順次に序列される一定数の項目が収容できる。よって、各ディレクトリページは一定のブロックアドレス範囲をカバーする。このようなページの中の項目はそれぞれのブロックアドレスに従って所定のオフセットをとるため、各項目の中で別々にブロックアドレスを記録する必要はない。ある特定のブロックのディレクトリ項目を見つけるには、そのブロックを含むブロックアドレス範囲をカバーするディレクトリページを見つけ、そのページの中で、ディレクトリページの最初の項目のブロックアドレスと所望のブロックアドレスとの差によって決まるオフセットのところにある項目まで進めばよい。したがって、ディレクトリページのバイナリ探索は

10

【0098】

メモリ内の各ブロックにつきレコードを保守することは、いくつかの用途にとって好都合である。例えば、ブロックの物理的特性について記述子を保守できる。このような記述子の一例として消去カウントがある。消去カウントは、消去され損耗レベル化の目的に使用できる特定のブロックの消去回数を示す。ブロックの消去カウントに従って1つ以上のリストでブロックを序列できる。また、ブロックが最後に消去されたときのタイムスタンプをレコードに含むことができ、こうすれば最後の消去から経過した時間に従ってブロックをリストの中で序列できる。

20

【0099】

これまで本発明の様々な態様をその例示的な実施形態との関係で説明してきたが、添付の特許請求の範囲の全範囲内においてその権利が保護されるべきであることが理解できよう。

【図面の簡単な説明】

【0100】

【図1】第1の例によるブロック分類の表を示す。

【図2】第1の例によるレコードブロックのページを示す。

【図3】図2のレコードブロックの個々のページの詳細図を示す。

30

【図4】第2の例によるブロック分類の表を示す。

【図5】ブロックレコードブロックにある対応するブロックレコードを指示するブロックディレクトリブロック内のブロックディレクトリ項目を示す。

【図6】ディレクトリブロックのページ構造の詳細を示す。

【図7A】レコードインデックスとディレクトリインデックスとを含む、図6のブロックディレクトリページの構造を示す。

【図7B】図6の用済みブロックリストページの構造を示す。

【図8】レコードブロックのページ構造の詳細を示す。

【図9】図8のレコードページの構造をより詳細に示す。

【図10】ブロックレコードと関係する構造との更新過程にあるディレクトリブロックとレコードブロックとを示す。

40

【図 1】

ブロック タイプ	ブロック内容			ブロック 分類
	有効データ	消去済み容量	用済みデータ	
プログラム	含む	含む	顧慮しない	不完全ブロック
共通	含む	含む	顧慮しない	不完全ブロック
フル共通	含む	含まない	含まない	なし
			含む	用済みブロック
ファイル	含む	含まない	含まない	なし
			含む	用済みブロック
無効	含まない	含む	含む	用済みブロック
消去済み	含まない	含む	含まない	消去済みブロック

Table of Block Classifications

Block Type	Block Contents			Block Classification
	Valid Data	Erased Capacity	Obsolete Data	
Program	Yes	Yes	Don't Care	Partial Block
Common	Yes	Yes	Don't Care	Partial Block
Full Common	Yes	No	No	None
			Yes	Obsolete Block
File	Yes	No	No	None
			Yes	Obsolete Block
Invalid	No	Yes	Yes	Obsolete Block
Erased	No	Yes	No	Erased Block

FIG. 1

【図 2】

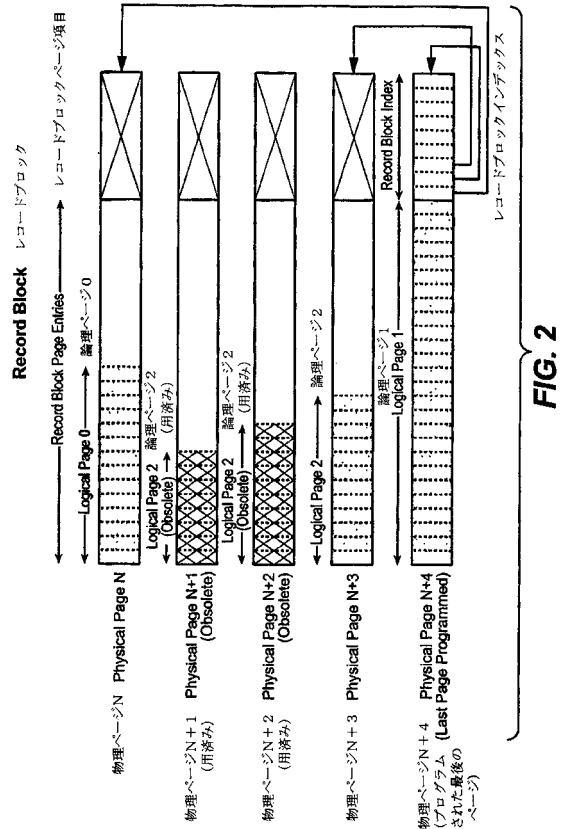


FIG. 2

【図 3】

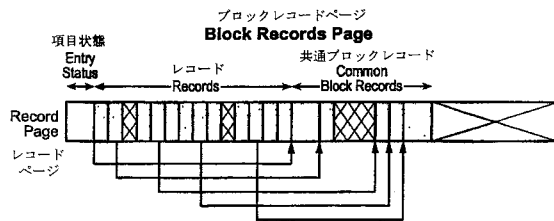


FIG. 3

【図 4】

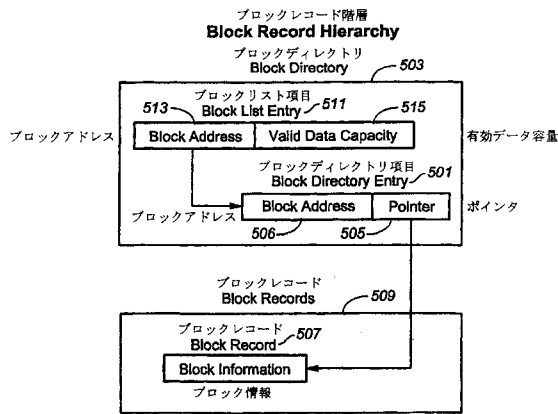
ブロック タイプ	ブロック内容			ブロック 分類
	有効データ	消去済み容量	用済みデータ	
プログラム	含む	含む	顧慮しない	不完全ブロック
共通	含む	含む	顧慮しない	不完全ブロック
フル共通	含む	含まない	含まない	コンプリー 共通ブロック
			含む	用済みブロック
ファイル	含む	含まない	含まない	なし
			含む	用済みブロック
無効	含まない	含む	含む	用済みブロック
消去済み	含まない	含む	含まない	消去済みブロック

Block Classifications

Block Type	Block Contents			Block Classification
	Valid Data	Erased Capacity	Obsolete Data	
Program	Yes	Yes	Don't Care	Partial Block
Common	Yes	Yes	Don't Care	Partial Block
Full Common	Yes	No	No	Complete Common Block
			Yes	Obsolete Block
File	Yes	No	No	None
			Yes	Obsolete Block
Invalid	No	Yes	Yes	Obsolete Block
Erased	No	Yes	No	Erased Block

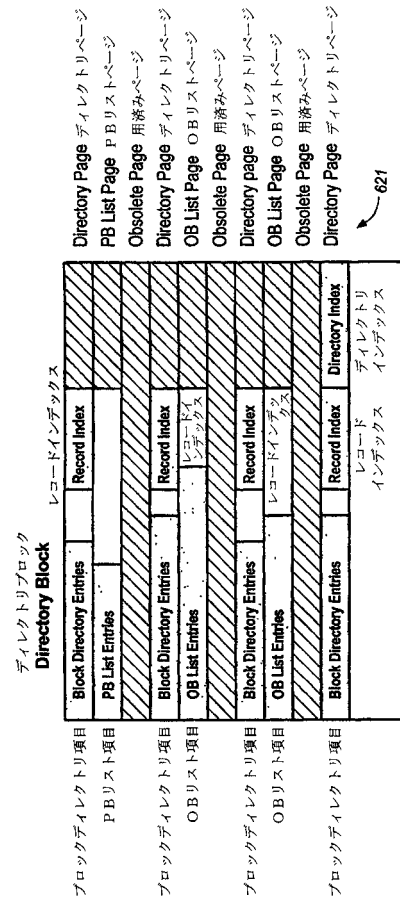
FIG. 4

【 図 5 】



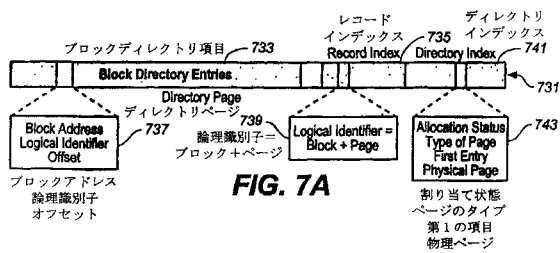
**FIG. 5**

【 図 6 】



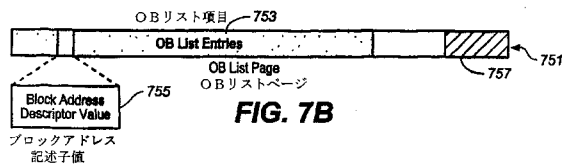
**FIG. 6**

【 図 7 A 】



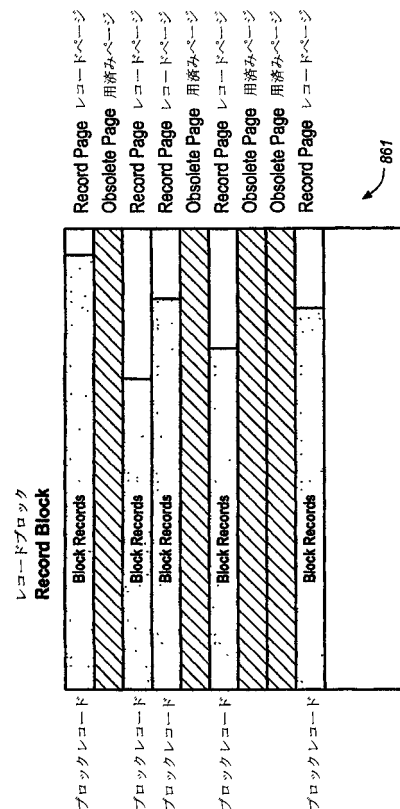
**FIG. 7A**

【圖 7 B】



**FIG. 7B**

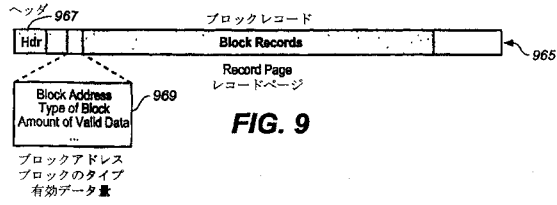
【 図 8 】



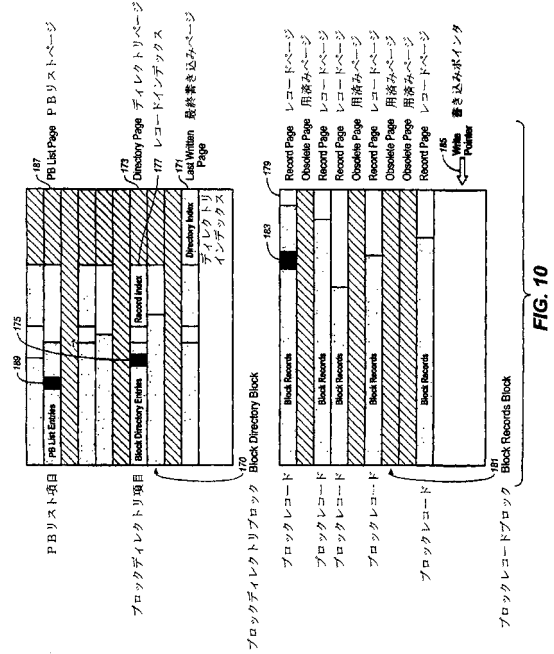
**FIG. 8**



【図 9】



【図 10】



---

フロントページの続き

- (31)優先権主張番号 11/459,260  
(32)優先日 平成18年7月21日(2006.7.21)  
(33)優先権主張国 米国(US)  
(31)優先権主張番号 11/459,268  
(32)優先日 平成18年7月21日(2006.7.21)  
(33)優先権主張国 米国(US)

## 早期審査対象出願

審査官 上嶋 裕樹

- (56)参考文献 国際公開第03/102782(WO, A1)  
特開2003-228513(JP, A)  
特開平07-072989(JP, A)  
特開2005-122439(JP, A)  
特開2004-310573(JP, A)  
国際公開第2004/040453(WO, A2)  
特開2002-366423(JP, A)  
国際公開第2005/066793(WO, A2)  
特表2004-526233(JP, A)  
特開2003-208352(JP, A)  
特開平10-326227(JP, A)

- (58)調査した分野(Int.Cl., DB名)  
G06F 12/00