(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4491798号
(P4491798)
(45)発行日　平成22年6月30日(2010.6.30)　(24)登録日　平成22年4月16日(2010.4.16)

(51)Int.Cl.　F I
G06F 17/14 (2006.01)　G06F 17/14 S

請求項の数 10　外国語出願　(全 29 頁)

(21)出願番号　特願2007-218719(P2007-218719)
(22)出願日　平成19年8月24日(2007.8.24)
(65)公開番号　特開2008-117368(P2008-117368A)
(43)公開日　平成20年5月22日(2008.5.22)
審査請求日　平成19年8月24日(2007.8.24)
(31)優先権主張番号　11/510218
(32)優先日　平成18年8月25日(2006.8.25)
(33)優先権主張国　米国(US)

(73)特許権者　501261300
エヌヴィディア　コーポレイション
アメリカ合衆国，カリフォルニア　９５０５０，サンタ　クララ，サン　トーマス　エクスプレスウェイ　２７０１
(74)代理人　100094318
弁理士　山田　行一
(74)代理人　100123995
弁理士　野田　雅一
(74)代理人　100107456
弁理士　池田　成人
(72)発明者　ラヴィ　ブルス
アメリカ合衆国，　カリフォルニア州，　サンタ　クララ，　ナンバー３１０，　マンション　コート　５３０

最終頁に続く

(54)【発明の名称】低消費パワーでデータ値配列に２次元変換を実行する方法及びシステム

(57)【特許請求の範囲】
【請求項１】
入力データ値をもつ行及び列を含む複数の入力データブロックの系列中における各入力データブロックに、行変換及び列変換を含む２Ｄ変換を実行するように構成されたシステムであって、
前記複数の入力データブロックの各々に応答して、前記入力データブロックにおける各々の行に前記行変換を実行することにより、部分的に変換されたデータをもつブロックを生成するように構成された行変換回路と、
前記部分的に変換されたデータをもつブロックにおける各々の列に前記列変換を実行するように構成された列変換回路と、
を備え、
前記行変換回路が、一の行のデータ値の第１の部分集合が少なくとも一つの零値を含むか否かを判定するように構成され、
前記行変換回路が、前記行のデータ値の第２の部分集合が少なくとも一つの零値を含むか否かを判定するように更に構成され、
前記行変換回路が、前記行変換を実行するときに、零値を含む少なくとも一つのデータ値であって、該少なくとも一つのデータ値は、前記行のデータ値の前記第１の部分集合及び前記行のデータ値の前記第２の部分集合からなる群から選択されるデータ値を含む、少なくとも一つのデータ値に関して選択される操作を実行するように更に構成されている、システム。

【請求項２】
前記選択される操作が、少なくとも一つの変換演算を回避することを含む、請求項１に記載のシステム。
【請求項３】
前記選択される操作が、零値を含む前記少なくとも一つのデータ値に対する少なくとも一つの変換演算を電力が削減されるように実行することを含む、請求項１に記載のシステム。
【請求項４】
前記行変換回路及び前記列変換回路に接続されているバッファを更に備え、
前記行変換回路が、前記部分的に変換されたデータをもつブロックを前記バッファに記憶するように構成され、
前記列変換回路が、前記部分的に変換されたデータをもつブロックを前記バッファから取り出すように構成されている、
請求項１に記載のシステム。
【請求項５】
前記行変換回路が、各々の前記行に対する前記行変換を実行するときに、並べ替え演算を実行するように構成されており、
前記並べ替え演算は、前記行のデータ値を並べ替えることによって、並べ替えられたデータ値を生成し、
前記行変換回路が、前記並べ替えられたデータをもつ第１の部分集合が少なくとも一つの零値を含むか否か、及び、前記並べ替えられたデータをもつ第２の部分集合であって前記第１の部分集合とは別個の該第２の部分集合が少なくとも一つの零値を含むか否かを判定するように更に構成されている、
請求項１に記載のシステム。
【請求項６】
データ値をもつ行及び列を含む複数の入力データブロックの系列に対して、行変換及び列変換を含む２Ｄ変換を、行変換回路と列変換回路とを備えるシステムが実行する方法であって、
前記行変換回路が、前記複数の入力データブロックのうちの一つブロックにおける各々の行に対する前記行変換を、各々の前記行の入力データ値に変換演算を実行することを含めて実行し、部分的に変換されたデータをもつブロックを生成するステップと、
前記列変換回路が、前記部分的に変換されたデータブロックにおける各々の列のデータ値に付加的な変換演算を実行することを含めて、各々の前記列に前記列変換を実行するステップと、
を含み、
前記行変換を実行するステップは、
一の行のデータ値の第１の部分集合が少なくとも一つの零値を含むか否かを判定するステップと、
前記行のデータ値の第２の部分集合が少なくとも一つの零値を含むか否かを判定するステップと、
零値を含む少なくとも一つのデータ値であって、該少なくとも一つのデータ値は、前記行のデータ値の前記第１の部分集合及び前記行のデータ値の前記第２の部分集合からなるグループから選択されるデータ値を含む、少なくとも一つのデータ値に関して選択される操作を実行するステップと、
を含む方法。
【請求項７】
前記行変換を実行するステップが、第１の入力及び第２の入力を有する乗算回路を作動させて、前記第１の入力にアサートされた第１の値が前記第２の入力にアサートされた第２の値と乗算される乗算演算を実行するステップを更に含み、
前記選択される操作を実行するステップが、零値を含む前記少なくとも一つのデータ値

に対する前記乗算演算を、前記第１の入力及び前記第２の入力からなる群から選択される入力の切り替えを阻止することによって、電力が削減されるように実行するステップを更に含む、
請求項６に記載の方法。
【請求項８】
前記選択される操作は、少なくとも一つの変換演算を回避する工程及び零値を含む前記少なくとも一つのデータ値に対する変換演算を電力が削減されるように実行する工程からなる群から選択される、請求項６に記載の方法。
【請求項９】
前記列変換を実行するステップが、
一の列のデータ値の第１の部分集合が少なくとも一つの零値を含むか否かを判定するステップと、
前記列のデータ値の第２の部分集合が少なくとも一つの零値を含むか否かを判定するステップと、
少なくとも一つの変換演算を回避するステップと、
を含む請求項６に記載の方法。
【請求項１０】
前記列変換を実行するステップが、
一の列のデータ値の第１の部分集合が少なくとも一つの零値を含むか否かを判定するステップと、
前記列のデータ値の第２の部分集合が少なくとも一つの零値を含むか否かを判定するステップと、
零値を含む前記少なくとも一つのデータ値に対する前記付加的な変換演算のうちの少なくとも一つを、電力が削減されるように実行するステップと、
を含む請求項６に記載の方法。
【発明の詳細な説明】
【発明の分野】
【０００１】
本発明は、２次元（２Ｄ）変換（例えば、逆離散コサイン変換又は逆アダマール変換）をデータ値の２Ｄ配列に実行する方法及びシステムに関するものである。典型的な実施形態では、本発明は、各値が相当な確率で零値となる２Ｄ配列の画像データ値に対する２Ｄ直交変換の逆変換を、零値を変換するためには不要のパワー消費型の演算を回避することによって消費パワーを削減するように、実行する方法及び回路に関するものである。
【発明の背景】
【０００２】
本開示の全体を通じて、ビデオデータの「ブロック」との用語を使用して、ビデオデータのフレームを構成するデータの部分集合であって、フレームの矩形領域内に空間位置を有する部分集合を表す。ビデオデータのブロックは、圧縮された（又は、さもなければ符号化された）ビデオデータで構成され得るが、ビデオデータで構成されていなくてもよい。ビデオデータのブロックの例は、従来の定義によるＭＰＥＧ符号化ビデオフレームのマクロブロックである。
【０００３】
多くの従来のアプリケーションにおいて、画像データ（例えば、ビデオデータ）又はその他のデータは２次元（「２Ｄ」）変換を受け、変換されたデータは元データを復元するために後に逆変換される。このような変換の例には、２Ｄ離散コサイン変換（２次元「ＤＣＴ」）、２Ｄアダマール変換、及び、２Ｄフーリエ変換がある。
【０００４】
本開示の全体を通じて、演算を「回避」する（そうしなければ、演算出力値を生成する）との表現は、実際に演算を実行することなく（演算出力値の代わりに）代替出力値を生成又はアサートすることを表す。乗算回路の入力に零値「ｚ１」及び非零定数「ｃ」をア

サートして当該回路にその出力への電流「ｃｚ１」をアサートさせ、第２の乗算回路の入力に別の零値「ｚ２」及び異なる非零定数「ｄ」をアサートして当該回路にその回路の出力への電流「ｄｚ２」をアサートさせ、電流「「ｃｚ１」及び「ｄｚ２」」に応答してノード（グラウンド電位上の零ボルトに等しい）に出力電圧「ｃｚ１＋ｄｚ２」をアサートするよう加算回路を動作させるオペレーションを「回避」する例は、乗算回路及び加算回路において実際に乗算ステップ及び加算ステップを実行することなく、ノードを接地する（それによって、ノードを強制的にグラウンド電位にする）ことであろう。

【０００５】

本発明は、２D配列のデータ値（すなわち、データ値の行及び列からなる配列）に２D変換を実行する改良された方法及びシステムに関するものであり、ここで、各値は相当に高い確率で零値となるものである。典型的な実施形態では、本発明は、２Ｄ配列のデータ値に２Ｄ直交変換の逆変換（例えば、２Ｄ逆離散コサイン変換又は逆アダマール変換）を実行する改良された方法及びシステムに関するものであり、ここで、各値は零値となる相当に高い確率を有するものである。あるクラスの好ましい実施形態では、本発明は、ＤＣＴ係数に２次元ＩＤＣＴ（２Ｄ逆離散コサイン変換）を実行する改良された方法及びシステムに関するものである。ＤＣＴ係数は、ビデオデータ（又はその他の画像データ）の配列に２Ｄ離散コサイン変換を実行することにより生成され、その各々は零値となる相当に高い確率を有している。

【０００６】

本開示の全体を通じて、「零値」（又は「零データ値」）という表現は、値零を示すデータを意味する。同様に、「零入力データ値」という表現は、値零を示す入力データを意味する。例えば、零入力値は、値零をもつ入力データ（例えば、ＤＣＴ係数、又は、ビデオデータのピクセルの色成分）を示す用語である。

【０００７】

本開示の全体を通じて、「疎」データ（例えば、逆変換を受けるデータの疎ブロック）との表現は、零値である可能性が高い値を示すデータを意味する。例えば、比較的多数の零値及び比較的少数の非零値を示す入力データのブロック（例えば、ＤＣＴ係数のブロック）は、データの疎ブロックである。

【０００８】

逆変換の実施は、通常、ビデオ圧縮及び伸長標準に準拠するシステムの実施の主要部分である。この逆変換は、計算集約的なプロセスであり、処理サイクル及び消費パワー要件に著しく関与する。ビデオ圧縮及び伸長標準を実施するモバイル機器（例えば、携帯型メディアプレーヤー）は、特に厳しい処理サイクル及び消費パワーの要件があり、アプリケーションによって設定された厳しい性能要件を満たし、非常に低いパワーを消費してバッテリー寿命を最大限に延ばす必要があり、変換エンジンは、通常、多数の圧縮標準及びこれらの標準に伴う様々な要件をサポートする必要がある。

【０００９】

データのブロックへの（２Ｄ逆変換を含む）２Ｄ変換の典型的な従来の実施は、性能を改善し、又は、パワーを削減するために、様々な組み合わせで以下の技術を使用する。

【００１０】

１．（変換エンジンに供給される各入力ブロックが外部手段によって符号化されている又は符号化されていないブロックとして特定される場合に）外部手段によって符号化されていないブロックとして特定されたブロックの変換を回避する。しかし、この技術には不必要な変換演算（例えば、符号化されたブロックであるとして特定されているが、零ＤＣ係数だけからなるブロックの変換）の遂行を招くことがあるといった不利な点がある。

【００１１】

２．完全に零値により構成された各入力データブロックの行又は列（「零行」又は「零列」）の全体を特定し、（例えば、零行又は零列毎に、所定の値、通常は「零」を出力することにより）このような行又は列に実行されることになる通常の変換演算を回避する。零行及び零列は、外部機器によって指定されるか、又は、変換エンジンによって内部で特

定される。しかし、この従来技術は、行（又は列）が零行（又は零列）ではないが、非常に少数の非零値しか含まない疎行（又は疎列）であるという多数の一般的な状況において、性能を改善しないか、又は、パワーを削減しない。

【００１２】

３．（入力データ行又は列内のデータ値を乗算する際に使用するために事前に決定された）同じ係数が後続の入力データ行又は列内のデータ値を乗算するために使用されるべきであることを示す条件を（入力データから）特定し、さもなければ後続の入力データ行又は列内のデータ値を乗算するために新しい係数を決定するよう実行されることになるこのような係数の更新を回避する。

【００１３】

４．分散算術変換を実施する（ルックアップテーブルに基づく２Ｄ変換の実施）。典型的なルックアップテーブルベースの実装は、ブロックを変換するために実行されるべき乗算演算の個数を削減することにより、オーバーヘッドを削減する。しかし、このような実装を設計することは通常、非常に複雑である。その理由は、非常に大規模なＲＯＭテーブル及び同様にマルチポート型ＲＯＭが通常必要とされ、設計制約によって、通常、達成されるべき消費パワーの改善が制限されるからである。

【００１４】

米国特許出願公開第２００５／００３３７８８号及び関連の米国特許第６，７９９，１９２号に記載されている別の従来の２Ｄ変換では、（ＩＤＣＴの列変換フェーズを実行するときに）データのブロックの各列内の最後の非零エントリーが決定され、そして、変換システムが、各列を逆変換するためにソフトウェアでＩＤＣＴ演算を実行する８個の異なる「特殊ＩＤＣＴ」プログラムルーチンの中の適切な一つへ分岐する。明らかに、（より少数の乗算演算及び加算演算を必要とする）より簡単な変換演算が、（最後の非零値がより高い位置にあることによって示されるように）比較的多数の零を有する列を処理するために利用され、（より多数の乗算演算及び加算演算を必要とする）より複雑な変換演算が、（最後の非零値がより低い位置にあることによって示されるように）より少数の零を有する列を処理するために利用される。これら参考文献は、（列変換フェーズ後に）ＩＤＣＴの行変換フェーズを実行するときに、ブロックの各行内の最後の非零エントリーが決定され、変換システムはその後に、各行を逆変換するためにソフトウェアでＩＤＣＴ演算を実施する８個の異なる「特殊ＩＤＣＴ」プログラムルーチンのうちの適切な一つへ分岐することも教示している。

【００１５】

米国特許出願公開第２００５／００３３７８８号及び米国特許第６，７９９，１９２号に記載された技術には、当該技術が、零値と非零値をもつ典型的なパターンを有する多数の列及び行を処理する場合に、性能を改善しないか、又は、消費パワーを削減しないという意味で非効率的であることを含めて、多数の問題及び制限がある。例えば、変換されるべき列又は行が零（特に、多数の零）を含むが、非零である最後のエントリーを有する場合に、この技術は、列又は行を変換するために大量のパワーを消費する複雑な（例えば、最も複雑な）「特殊ＩＤＣＴ」ルーチンを選択する。これに対して、本発明の好適な実施形態は、零値で構成されている列の部分及び行の部分（例えば、零値により構成されている行の半分又は列の半分の各々、若しくは、行の４分の１又は列の４分の１の各々）への変換演算を回避することにより、或いは、電力が削減されるようにこのような変換演算を実行することにより、性能を改善し消費パワーを削減する。本発明の一部の好ましい実施形態は、変換されるべき行又は列内の個々の零値への変換演算を回避することにより（或いは、電力が削減されるように行又は列内の個々の零値への変換演算を実行することにより）性能を改善し消費パワーを削減する。

【００１６】

米国特許出願公開第２００５／００３３７８８号又は米国特許第６，７９９，１９２号には、これら文献に記載されている技術によって得られる性能改善及び低消費パワー化の利益が、変換される各行又は列の部分集合を独立に処理することによって増大されること

についての示唆、並びに、どのようにそれを実行するか、又は、そのようにすることが可能であるか否かに関する示唆はない。これに対して、本発明の好適な実施形態は、変換すべき（例えば、逆変換すべき）各行又は列の異なる部分集合に同じ演算を順次に実行可能であり、ここで、各行又は列の部分集合が行又は列の部分を定めるものであり、このような実施形態によって達成可能な性能改善及び低消費パワー化の利益は、かかる部分を定める部分集合のサイズを減少させることだけによって増大することができる。例えば、本発明の一部の好ましい実施形態は、変換されるべき８Ｎビットの各行又は各列のうちの２Ｎビットの部分集合に演算の組（１行又は１列当たり４組の演算の組）を順次に実行して優れた性能改善及び低消費パワー化の利益を達成し、本発明のその他の好ましい実施形態は、変換されるべき８Ｎビットの行又は列のうちのＮビットの部分集合に演算の組（１行又は１列当たりに８組の演算の組）を順次に実行して更に優れた性能改善及び低消費パワー化の利益を達成する。

【００１７】

別の従来の２Ｄ変換は、ＲｏｈｉｎｉＫｒｉｓｈｎａｎらによる“Ｄｅｓｉｇｉｎｏｆａ２Ｄ ＤＣＴ／ＩＤＣＴ Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ ＶＬＩＷ ＰｒｏｃｅｓｓｏｒＳｕｐｐｏｒｔｉｎｇ Ｓｃａｌｅｄ ａｎｄＳｕｂ－ｓａｍｐｌｅｄ Ｂｌｏｃｋｓ”, １６ｔｈ ＩｎｔｅｒｎａｔｉｏｎａｌＣｏｎｆｅｒｅｎｃｅ ｏｎ ＶＬＳＩ Ｄｅｓｉｇｎ, ｓｉｘ ｐａｇｅｓ（２００３）という題名の論文に記載されている。この論文は、ＩＤＣＴ回路へのデータブロック全体のスケール縮小版（例えば、８×８形のブロックの偶数行を廃棄することにより生成された８×４形のブロック）をアサートし、さもなければブロック全体を逆変換するために使用されることになるＩＤＣＴ回路の一部を回避すること等によってスケール縮小されたブロックを逆変換するようＩＤＣＴ回路を動作させることを教示する。この方法は、最終的に廃棄される出力値の計算を回避できるが、最終結果に決して寄与することがない演算を検出し省略することがない。

【００１８】

別の従来の２Ｄ変換は米国特許第５，８８３，８２３号に記載されている。この変換は、変換されるべき入力ブロックの領域を特定し、各領域を別々に処理する（例えば、あるＩＤＣＴは一部の領域の全要素に実行され、あるＩＤＣＴはその他の領域の非零要素だけに実行される）。例えば、ＵＳ５，８８３，８２３は、「局所的」ＩＤＣＴ計算が８×８形のブロックの一つの象限（すなわち、最低周波数領域に対応する４×４形の象限）の（零と非零とを問わずに）全要素に実行され、別のＩＤＣＴ計算が８×８形のブロックのその他のそれぞれ３つの象限（すなわち、より高い周波数領域に対応する３つの４×４形の象限）の非零要素だけに実行されるＩＤＣＴ計算を（第１０欄第５３行から第１１欄第２６行に）明確に教示している。しかしながら、ＵＳ５，８８３，８２３は、ＩＤＣＴ演算処理が非零要素だけに実行される各領域の非零要素を特定する方式（若しくは、このような非零係数を効率的に特定する方式）、又は、ブロックの領域の非零要素だけにＩＤＣＴ計算を実行する方式、又は、このようなＩＤＣＴ計算をこのような非零要素だけに効率的に（しかも、低消費パワー化されるように）実行する方式を教示又は示唆していない。

【発明の概要】

【００１９】

あるクラスの実施形態では、本発明は、入力データブロックの系列中の各ブロックに２Ｄ変換（例えば、逆離散コサイン変換）を実行するように構成されたシステムである。ここで、各ブロックは、入力データ値の行と列とにより構成されており、２Ｄ変換は、行変換及び列変換を含むものである。これらの実施形態では、システムは、各ブロックの全ての行に行変換を実行して部分的に変換されたデータのブロックを生成し、次いで、部分的に変換されたデータのブロックの各列に列変換を実行することによって、又は、各ブロックの全ての列に列変換を実行して部分的に変換されたデータのブロックを生成し、次いで、部分的に変換されたデータのブロックの各行に行変換を実行することによって、２Ｄ変換を実行するように構成されている。説明を簡単にするために、このクラスにおける実施形態を、各ブロックの全ての行に行変換を実行して部分的に変換されたデータのブロック

を生成し、次いで、部分的に変換されたデータのブロックの各列に列変換を実行することにより、２Ｄ変換を実行するように構成されたものとして説明する。「行」及び「列」への参照を全て、それぞれ、「列」及び「行」への参照によって置き換えて上記クラスの他の実施形態を説明することができることを理解されたい。

【００２０】

本明細書中、データ値の集合（例えば、行又は列）の「部分集合」との用語を広義に使用しており、当該用語は、データ値の行（又は列）、データ値の行（又は列）の偶数要素、データ値の行（又は列）の奇数要素、データ値の行（又は列）におけるＮ個毎に一つのデータ値、（ＦＦＴバタフライに適している）ビット反転順のデータ値の行又は列の偶数要素、又は、データ値の別の部分集合を表すことができる。

【００２１】

行変換を実施するために、上記のクラスにおけるシステムは、入力データブロックにおける各行の入力データ値に（通常、乗算及び加算を含む）変換演算を実行して部分的に変換されたデータのブロックを生成するように構成された回路を備える。システムは、通常、部分的に変換されたデータが記憶されるバッファを備える。列変換を実施するために、システムは、部分的に変換されたデータのブロックにおける各列のデータ値に（通常、乗算及び加算を含む）変換演算を実行するように構成された回路を備える。上記クラスにおける各実施形態は、以下の様に構成される。即ち、入力データの各行に行変換を実行するときに、その行の第１の区画を構成するデータ値の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否か（例えば、零値から成るか否か）を判定し、当該行の第１の区画の第１の部分集合中の異なる部分集合の各々が少なくとも一つの零値を含むか否かを判定し、当該行の第１の区画の少なくとも一つの別の部分集合中の異なる部分集合の各々が少なくとも一つの零値を含むか否かを判定し、少なくとも一つの零入力データ値及び少なくとも一つの非零入力データ値を含む上記行毎に行変換を実行する場合に、少なくとも一つの上記零入力データ値（かつ、好ましくは、上記零入力データ値の各々）に対する少なくとも一つの変換演算を回避させるか、又は、電力が削減されるように実行する（例えば、乗算及び加算回路を使用して実行されることになる少なくとも１回の乗算又は加算が回避される）。ここで、かかる変換演算は、零値が非零値であるならば全パワーを消費するように実行されることになるものである。少なくとも一つ零入力データ値及び少なくとも一つの非零入力データ値を含む各行に行変換を実施するときに、零入力値に変換演算を実行する一部のこのような実施形態の回路は、その入力のうちの少なくとも一つを更新することなく作動され、各入力を切り替えるために消費されることになるパワー消費を回避する。

【００２２】

好ましくは、上記クラスにおける実施形態は、以下の様に構成される。即ち、部分的に変換されたデータの各列に列変換を実行するときに、列の第１の区画を構成するデータ値の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否か（例えば、零値により構成されているか否か）を判定し、列の第１の区画の第１の部分集合中の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否かを判定し、列の第１の区画の少なくとも一つの別の部分集合中の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否かを判定し、部分的に変換されたデータの少なくとも一つの零値及び部分的に変換されたデータの少なくとも一つの非零値を含む上記列毎に列変換を実行するときに、少なくとも一つの変換演算を回避する（又は、電力が削減されるように実行する）。なお、この変換演算は、値が非零値であるならば、部分的に変換されたデータ値のうちの少なくとも一つの上記零値（かつ、好ましくは、上記零値の各々）に対して全パワーを消費する方式で実行されるものである（例えば、そうでなければ乗算及び加算回路を使用して実行されることになる少なくとも１回の乗算又は加算）。例えば、少なくとも一つの零データ値及び少なくとも一つの非零データ値を含む各列に列変換を実行するときに、一つのこのような実施形態は、零値が非零値であるならば実行されることになる列の零値への変換演算を回避するように構成されている（例えば、この実施形態は変換演算を実行する回路を回避する）

。

【００２３】

本発明の好ましい実施形態は、変換されるべきデータの各ブロックが完全に零値から成るか否かを判定する。ブロックが完全に零値から成るものと判定されると、ブロックの値に関する変換演算（行変換演算と列変換演算の双方）は、回避されるか、又は、電力が削減されるように実行される。これらの好ましい実施形態は、変換されるべきデータのブロックの各行又は各列中における複数の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否かを順次に（例えば、反復的に）判定する。このような実施形態の実施例は、値x_iから成るデータの行（又は列）（ここで、ｉは０≦ｉ≦Ｎ－１の範囲に入る整数であり、Ｎが偶数である）と、データ値の第１の部分集合及び第１の部分集合とは別のデータ値の第２の部分集合への行（又は列）の一区画と、データ値の第３の部分集合及び第３の部分集合とは別個の第４の部分集合への第１の部分集合の一区画と、データ値の第５の部分集合及び第５の部分集合とは別個の第６の部分集合への第２の部分集合の一区画とを参照する。例示の実施形態は、第１の部分集合が完全に零値により構成されているか否か、及び、第２の部分集合が完全に零値により構成されているか否かを判定する。通常、行（又は列）が（ビデオデータのフレームにＤＣＴを実行することにより生成された）コサイン変換係数により構成されている場合、第１の部分集合が低周波係数（ｉが０≦ｉ≦（Ｎ／２）－１の範囲に入る整数である場合の値x_i）により構成され、第２の部分集合が高周波係数（ｉがＮ／２≦ｉ≦Ｎ－１の範囲に入る整数である場合の値x_i）により構成され、第２の部分集合が零値だけにより構成されている可能性が高い（そして、零値だけにより構成されている可能性は第１の部分集合より遙かに高い）。

【００２４】

例示の実施形態は、第１の部分集合が完全に零値により構成されていることを判定すると、第１の部分集合中の値への変換演算を回避するか、又は、この変換を電力が削減されるように実行する（例えば、これらの演算を実行する回路が回避されるか、又は、回路が更新されていない入力のうちの少なくとも一つを用いて作動され、このような各入力をトグルする（（切り替える）ために消費されることになるパワー消費を回避する）。第２の部分集合が完全に零値により構成されていることを判定すると、第２の部分集合中の値への変換演算は回避されるか、又は、電力が削減されるように実行される。

【００２５】

第１の部分集合及び第２の部分集合のそれぞれが少なくとも一つの非零値を含むことを判定すると、例示の実施形態は、第３の部分集合、第４の部分集合、第５の部分集合及び第６の部分集合のそれぞれが完全に零値により構成されているか否かを判定する。第１の部分集合が低周波係数（ｉが０≦ｉ≦（Ｎ／２）－１の範囲に入る整数である場合の値x_i）により構成される典型的な実施例では、第２の部分集合が高周波係数（ｉがＮ／２≦ｉ≦Ｎ－１の範囲に入る整数である場合の値x_i）により構成され、第３の部分集合が第１の部分集合中の偶数値（ｉが０≦ｉ≦（Ｎ／２）－１の範囲に入る偶数整数である場合の値x_i）により構成され、第４の部分集合が第１の部分集合中の偶数値（ｉが０≦ｉ≦（Ｎ／２）－１の範囲に入る奇数整数である場合の値x_i）により構成され、第５の部分集合が第２の部分集合中の偶数値により構成され、第６の部分集合が第２の部分集合中の奇数値により構成される。完全に零値により構成されていることが判定された第３の部分集合、第４の部分集合、第５の部分集合及び第６の部分集合のそれぞれに関して、このような部分集合の値についての変換演算は、回避されるか、又は、電力が削減されるように実行され（例えば、これらの演算を実行する回路は回避されるか、又は、回路は入力のうちの少なくとも一つを更新することなく作動されて、このような各入力をトグルするために消費されることになるパワー消費を回避する）。少なくとも一つの非零値を含むことが判定された第３の部分集合、第４の部分集合、第５の部分集合及び第６の部分集合のそれぞれに関して、変換演算はこのような部分集合の値に全パワーを消費する方式で実行される。

【００２６】

上述した例示の実施形態の変形例（及び本発明のその他の実施形態）では、変換されるべきブロックの各行又は各列（又は、行又は列の部分集合）からなるデータ値は、行又は列（又はこれらの部分集合）の一部分からなる各個別の部分集合が完全に零値により構成されているか（又は零値であるか）否かに関する判定の前又は判定中に、並べ替えられる。例えば、行が値x_iにより構成され、ここで、ｉが０≦ｉ≦Ｎ－１の範囲に入る整数であり、Ｎが偶数整数である場合に、
区画はその行中の別々の第１の部分集合及び第２の部分集合により構成され、
第１の部分集合はその行中の別々の第３の部分集合及び第４の部分集合により構成され、
第２の部分集合はその行中の別々の第５の部分集合及び第６の部分集合により構成され、
第１の部分集合はｉが０≦ｉ≦（Ｎ／２）－１の範囲に入る整数である場合の値x_iにより構成され、
第２の部分集合はｉが高周波係数（Ｎ／２≦ｉ≦Ｎ－１の範囲に入る整数である場合の値x_i）により構成され、
第３の部分集合は、ｉが０≦ｉ≦（Ｎ／４）－１の範囲に入る整数である場合に、並べ替えを行なった第１の部分集合の偶数値である値x_iにより構成され、
第４の部分集合は、ｉがＮ／４≦ｉ≦（Ｎ／２）－１の範囲に入る整数である場合に、並べ替えを行なった第１の部分集合の奇数値である値x_iにより構成され、
第５の部分集合は、ｉがＮ／２≦ｉ≦（３Ｎ／４）－１の範囲に入る整数である場合に、並べ替えを行なった第２の部分集合の偶数値である値x_iにより構成され、
第６の部分集合は、ｉが３Ｎ／４≦ｉ≦Ｎ－１の範囲に入る整数である場合に、並べ替えを行なった第２の部分集合の奇数値である値x_iにより構成される。

【００２７】

別のクラスの実施形態では、本発明は、入力データブロックの系列に２Ｄ変換を実行する方法であり、ここで、各ブロックは入力データ値の行及び列からなり、２Ｄ変換は行変換及び列変換を含むものである。このクラスの典型的な実施形態では、各ブロックは、ビデオデータにＤＣＴを実行することにより生成されたＤＣＴ係数のブロックであり、２Ｄ変換は逆離散コサイン変換である。これらの典型的な実施形態及びこのクラス内のその他の実施形態では、系列中の殆ど又は全部のブロック（例えば、少なくとも実質的に全部のブロック）は疎データのブロックである。２Ｄ変換は、各入力データブロックの全行に行変換を実行して部分的に変換されたデータのブロックを生成し、次いで、部分的に変換されたブロックの各列に列変換を実行する各ステップ、又は、各入力データブロックの全列に列変換を実行して部分的に変換されたデータのブロックを生し、次いで、部分的に変換されたデータのブロックの各行に行変換を実行する各ステップを含む。説明を簡単にするため、このクラスの実施形態を、２Ｄ変換が各入力データブロックの全行に行変換を実行して部分的に変換されたデータのブロックを生成し、部分的に変換されたデータのブロックの各列に列変換を実行する各ステップを含むものとして、説明する。「行」及び「列」への参照を全て、「列」及び「行」それぞれへの参照によって置き換えてクラス内の他の実施形態を説明することができることを理解されたい。

【００２８】

上記のクラス内の一部の実施形態では、本方法は、以下の各ステップを含む。即ち、
（ａ）入力データブロックのうちの一つのブロックの各行に対する行変換を、各行における入力データ値に対する変換演算を実行することによることを含めて実行し、部分的に変換されたデータブロックを生成する、
（ｂ）部分的に変換されたデータブロックの各列に対する列変換を、各列のデータ値に対する更なる変換演算を実行することによることを含めて、実行する、
各ステップを含む。ここで、ステップ（ａ）は、以下の各ステップを含む。即ち、
上記各行の第１の区画を構成するデータ値の異なる部分集合のそれぞれが少なくとも一つの零値を含むか（例えば、零値により構成されているか）否かを判定し、第１の区画の第１の部分集合中の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否かを判定し、第１の区画の少なくとも一つのその他の部分集合中の異なる部分集合のそれぞれが少

なくとも一つの零値を含むか否かを判定し、
少なくとも一つの零入力値及び少なくとも一つの非零入力値を含む上記各行に行変換を実行するときに、少なくとも一つの上記各零入力値が非零値であるならば、少なくとも一つの上記零入力データ値（そして、好ましくは、上記零入力データ値毎）に全パワーを消費する形で実行されることになる少なくとも１回の変換演算を回避する（又は、電力が削減されるように実行する）（例えば、行の少なくとも一つの上記零入力データ値（そして、好ましくは、上記零入力データ値毎）に乗算及び加算回路を使用して実行されることになる少なくとも１回の乗算又は加算を回避する）、
各ステップを含む。

【００２９】

例えば、少なくとも一つの零入力データ値及び少なくとも一つの非零入力値を含む各行に行変換を実施するときに、ステップ（ａ）は、乗算演算（第１の入力にアサートされた零入力値が第２の入力にアサートされた第２の値によって乗算される演算）を実行するために、第２の入力をトグルするために消費されることになるパワー消費を回避するよう第２の入力にアサートされた値を更新することなく、第１の入力及び第２の入力を有する乗算回路を作動するステップを含む。

【００３０】

好ましくは、ステップ（ｂ）は、
上記各列における区画を構成するデータ値の異なる部分集合のそれぞれが少なくとも一つの零値を含むか（零値により構成されているか）否かを判定し、当該区画の第１の部分集合中の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否かを判定し、当該区画の少なくとも一つのその他の部分集合中の異なる部分集合のそれぞれが少なくとも一つの零値を含むか否かを決定するステップと、
少なくとも一つの零入力データ値及び少なくとも一つの非零入力データ値を含む上記各列に列変換を実行するときに、上記各零入力値が非零値であるならば、少なくとも一つの上記零入力データ値（そして、好ましくは、上記零入力データ値毎）に全パワーを消費する形で実行されることになる少なくとも１回の付加的な変換演算を回避する（又は、電力が削減されるように実行する）（例えば、列の少なくとも一つの上記零入力データ値（そして、好ましくは、上記零入力データ値毎）に乗算及び加算回路を使用して実行されることになる少なくとも１回の乗算又は加算を回避する）ステップと、
を含む。

【００３１】

本発明の典型的な実施形態により実施される変換回路の利点には、
冗長又は別の不必要な計算又は計算ステップの回避に起因した（低周波数の動作を可能とする）改善された（好ましくは、最適化された）計算効率（例えば、入力データ値のブロックを変換する際の本発明の変換エンジンの典型的な実施形態の性能は、１ブロック当たりの非零入力データ値の個数に直接的に対応する。これは、入力データ値のブロックを変換する際の性能が１ブロック当たりに少なくとも一つの非零値を含む行の行数に対応する従来の変換エンジンより粒度のレベルが細かい）
低減されたスイッチング動作が低消費パワー化すること
（好ましい実施形態における）インテリジェントな中間バッファメモリマネージメントの提供
冗長又は別の不必要な計算又は計算ステップの回避（例えば、不必要な乗算器入力トグリングの回避）による整数変換の高速化性能
が含まれる。

【００３２】

一部の実施形態では、本発明のシステムは、本発明によって実施される変換エンジンを含むビデオ処理システム（例えば、パイプライン方式ビデオ復号システム）である。一部のこのような実施形態では、ビデオ処理システムは、携帯型メディアプレーヤーのビデオ処理サブシステムとして動作可能であるように構成される。その他の実施形態では、本発

明のシステムは、本発明によって実施される変換エンジンを含むビデオ処理サブシステムを含む携帯型メディアプレーヤーである。

【００３３】

本発明のその他の態様は、本発明のシステムの実施形態で使用される変換エンジン及び変換エンジン回路と、本発明のシステムの実施形態の動作中に実行される方法である。

【好ましい実施形態の詳細な説明】

【００３４】

以下、本発明のシステムの実施形態を、図１、２、３、４、５及び６を参照して説明する。

【００３５】

図１は、パイプライン方式ビデオ復号システムのシステムレベルブロック図であり、当該システムは、ビットストリームエンジン１、（本発明に従って実施される）変換エンジン３、及び、ピクセル再構成エンジン５を備えており、これらは、図示するようにシステムバスに沿って接続されている。ビットストリームエンジン１は、符号化されたビデオデータを受信する。この符号化されたビデオデータは、周波数領域係数の処理済みブロックを含んでおり、当該周波数領域係数は、ビデオピクセルのブロックに２Ｄ変換（例えば、２次元ＤＣＴ）を実行することにより生成されている。ビットストリームエンジン１は、通常、アンパッキング及び逆量子化を実行することによって符号化されたビデオデータを処理し、得られる処理済みの周波数領域の係数のブロックの系列を変換エンジン３へアサートする。変換エンジン３は、系列中の各ブロックに２Ｄ逆変換を実行して部分的に復号されたビデオピクセルのブロックを生成し、部分的に復号されたビデオピクセルを更なるビデオ復号のためピクセル再構成エンジン５へアサートする。ピクセル再構成エンジン５は、部分的に復号されたビデオピクセルに応じて完全に復号されたピクセルを生成し、完全に復号されたピクセルをシステムバスへアサートする。

【００３６】

図１のシステムは、ビデオ復号機能を有する携帯型メディアプレーヤー又は別の携帯型データ処理システム（又は、携帯可能であることが目的とされていないデータ処理システム）のサブシステムであってもよく、このようなサブシステムは、本発明による変換エンジン３を実装することにより、本発明に従って実施される。このような処理システムは、図１には示していない更なる要素を備えることがあり、通常は更なる要素を含むものである。

【００３７】

本発明の一部の実施形態は、図１に示す構造を持たないシステムによって、又は、図２に示す構造を持たない処理システムによって実施されることも考えられる。

【００３８】

図２は、図１の変換エンジン３、又は、データ値のブロックに２次元変換（例えば、２Ｄ逆変換）を実行する別の装置の典型的な実施例のブロック図である。図２は、図示の通りに接続された行／列変換回路７、記憶バッファ９、及び、列／行変換回路１１を備えている。一部の実施例では、図２の回路７は、回路７によって受信された入力データのブロックの系列中の各ブロックの全行に行変換を実行し、得られたデータを記憶バッファ９へアサートするように構成され、回路１１は、バッファ９からこのようなデータの列を読み出し、行変換から得られたデータの各ブロックの全列に列変換を実行するように構成されている。その他の実施例では、図２の回路７は、回路７によって受信された入力データのブロックの系列中の各ブロックの全列に列変換を実行し、得られたデータを記憶バッファ９へアサートするように構成され、回路１１は、バッファ９からこのようなデータの行を読み出し、列変換から得られたデータの各ブロックの全行に行変換を実行するように構成されている。

【００３９】

図３は、図２の実施例の回路７に入力され得るタイプの入力データブロック（「Ｉ」と呼ばれる）と、（例えば、図２のこのような実施例の回路７において）ブロックＩに行変

換を適用したブロック（「ＩＩ」と呼ばれる）と、（例えば、図２のこのような実施例の回路７及び回路１１において）ブロックＩに行変換及び列変換を適用したブロック（「ＩＩ」と呼ばれる）の図である。図３のブロックＩは、４行４列の係数からなる周波数領域の係数の４×４形のブロックであり、零値を有する各係数は「Ｏ」で表され、非零値を有する各係数は「Ｘ」で表されている。行は参照シンボル「Ｒ１」乃至「Ｒ４」によって特定され、列は参照シンボル「ａ」乃至「ｄ」によって特定されている。

【００４０】

典型的な従来の変換エンジンは、完全に零値により構成された（変換されるべき各ブロックの）全行又は全列（すなわち、「零行」又は「零列」）を特定し、特定された各零行又は零列に実行されることになる通常の変換演算を回避するように構成されている。このような従来の変換エンジンは、ブロックＩの行Ｒ１及びＲ３を零行として特定し、行Ｒ１及びＲ３に実行することになる変換計算を回避する。

【００４１】

これに対して、本発明の変換エンジンの典型的な実施形態（例えば、本発明を具現化する図２のエンジンの実施例）は、変換されるべき各ブロックの零行及び零列（例えば、ブロックＩの行Ｒ１及びＲ３）を特定し、（行を零行として特定しなかった場合に）零行に実行されることになる行変換計算を回避し、（列を零列として特定しなかった場合に）零列に実行されることになる列変換計算を回避するだけでなく、非零行及び非零列への変換演算（例えば、ブロックＩの行Ｒ２及びＲ４上の行変換計算）中に付加的なパワーを節約するようにも構成されている。この付加的なパワー節約は、行における完全に零データ値により構成された部分集合（及び列の部分集合）（例えば、列ｃ及びｄ内の係数により構成されたブロックＩの行Ｒ２の部分集合、又は、列ｂ及びｄ内の係数により構成されたブロックＩの行Ｒ４の部分集合）を特定し、部分集合が完全に零データ値だけにより構成されていないならば、このような各部分集合に実行されることになる変換計算を回避する（又は、電力が削減されるように実行する）ことにより達成される。

【００４２】

本発明の変換エンジンのこのような実施形態が図２に示すような構造を有し、４×４のＤＣＴ変換係数をもつブロックの系列中の各ブロックにＩＤＣＴ（逆ＤＣＴ）変換を実行するように構成されている場合を考える。このような各ブロックの各行は、２個の低周波係数（例えば、図３における各ブロックの列「ａ」及び「ｂ」内の低周波係数）と、２個の高周波係数（例えば、図３における各ブロックの列「ｃ」及び「ｄ」内の高周波係数）とにより構成されている。このようなブロックの各列は、２個の低周波係数（例えば、図３における各ブロックの行「Ｒ１」及び「Ｒ２」内の低周波係数）と、２個の高周波係数（例えば、図３における各ブロックの行「Ｒ３」及び「Ｒ４」内の高周波係数）とにより構成されている。この場合には、各行及び各列の係数は、周波数が高いほど、低周波係数の場合より零値だけによって構成されている可能性が高くなる。

【００４３】

本発明の変換エンジンのこのような実施形態の回路７は、ＤＣＴ変換係数の各ブロック中の各行の２個の最低周波係数が完全に零値により構成されているか否かを判定し、このような各行の２個の最高周波係数が完全に零値により構成されているか否かを判定し、完全に零値により構成されているならば、各行の２個の最低周波係数への行変換計算を回避し（このような計算は、完全に零値により構成されていないならば、これらの係数に全パワーを消費するような形で実行されることになる）、又は、このような計算を電力が削減されるように実行し、完全に零値により構成されているならば、各行の２個の最高周波係数への行変換計算を回避し（このような計算は、完全に零値により構成されていないならば、これらの係数に全パワーを消費するような形で実行されることになる）、又は、このような計算を電力が削減されるように実行するように構成されている。例えば、このような実施形態の回路７は、ブロックＩの行Ｒ１、Ｒ２、Ｒ３及びＲ４のそれぞれの２個の最高周波係数への行変換計算を回避するように構成される。

【００４４】

本発明の変換エンジンのこのような実施形態の回路７は、
アサートされたＤＣＴ変換係数の各ブロック中の各行の２個の最低周波係数のうちの偶数係数（例えば、図３のブロックＩの列「ｂ」内の各係数）が零値であるか否かを判定し、アサートされたＤＣＴ変換係数の各ブロック中の各行の２個の最低周波係数のうちの奇数係数（例えば、図３のブロックＩの列「ａ」内の各係数）が零値であるか否かを判定し、アサートされたＤＣＴ変換係数の各ブロック中の各行の２個の最高周波係数のうちの偶数係数（例えば、図３のブロックＩの列「ｄ」内の各係数）が零値であるか否かを判定し、アサートされたＤＣＴ変換係数の各ブロック中の各行の２個の最高周波係数のうちの奇数係数（例えば、図３のブロックＩの列「ｃ」内の各係数）が零値であるか否かを判定するようにも構成される。このような実施形態の回路７は、零値であると判定されたこのような各係数に（より多くのパワーを消費する形で）実行されることになる乗算計算を電力が削減されるように実行して行を変換するようにも構成される。通常、このような低パワー化方式の各乗算は、第１の入力及び第２の入力を有する乗算回路を作動させて、
（第１の入力へアサートされた）零値に（第２の入力へアサートされた）第２の値を、当該第２の入力へアサートされた値を更新することなく、乗算することによって実行され、これによって、第１の入力へアサートされた値が非零値である場合に（そして、第２の入力における更新された第２の値がこのような非零値と乗算される必要がある場合に）、第２の入力をトグルするために消費されることになるパワーの消費を回避する。

【００４５】

本発明の変換エンジンのこのような実施形態の回路１１は、好ましくは、バッファ９から読み出される、部分的に変換された係数の４×４形の各ブロック中の各列における前半分（すなわち、最初の２個のデータ値）が完全に零値により構成されているか否かを判定し、このような各列における後半分（最後の２個のデータ値）が完全に零値により構成されているか否かを判定し、完全に零値により構成されているならば、（完全に零値により構成されていないならば、これらの値に全パワーを消費する形で実行されることになる）各列の前半分への列変換計算を回避するか、又は、このような計算を電力が削減されるように実行し、完全に零値により構成されているならば、（完全に零値により構成されていないならば、これらの値に全パワーで実行されることになる）各列の後半分への列変換計算を回避するか、又は、このような計算を電力が削減されるように実行するようにも構成される。

【００４６】

本発明の変換エンジンのこのような実施形態の回路１１は、バッファ９から読み出される、部分的に変換された係数の４×４形の各ブロック中の各列の前半分内の偶数データ値（例えば、図３のブロックＩＩ中の行Ｒ２内の各値）が零値であるか否かを判定し、このような各列の前半分内の奇数データ値（例えば、図３のブロックＩＩ中の行Ｒ１内の各係数）が零値であるか否かを判定し、このような各列の後半分内の偶数データ値（例えば、図３のブロックＩＩ中の行Ｒ４内の各係数）が零値であるか否かを判定し、このような各列の後半分内の奇数データ値（例えば、図３のブロックＩＩ中の行Ｒ３内の各係数）が零値であるか否かを判定するようにも構成される。このような実施形態の回路１１は、零値であると判定された各データ値に（より多くのパワーを消費する形で）実行されることになる乗算計算を電力が削減されるように実行して列を変換するようにも構成される。通常、このような各低パワー化乗算は、第１の入力及び第２の入力を有する乗算回路を作動させて、（第１の入力へアサートされた）零値に（第２の入力へアサートされた）第２の値を、当該第２の入力にアサートされた値を更新することなく、乗算することにより実行され、これによって、第１の入力にアサートされた値が非零値である場合に（そして、第２の入力で更新された第２の値がこのような非零値で乗算される必要がある場合に）、第２の入力をトグルするために消費されることになるパワーの消費を回避する。

【００４７】

各行及び各列が８個のデータ値により構成される上記実施例の変形例では、行（又は列）における前半分（又は後半分）内の奇数（又は偶数）データ値が零値であるか否かを判

定する各ステップは、行（又は列）における上記半分のうちの前半分（又は後半分）内のデータ値が完全に零値により構成されているか否かを判定するステップによって置き換えられ、エンジンは、完全に零値により構成される各半行（又は各半列）における半分それぞれの行（又は列）変換計算を回避するか（このような変換計算は、完全に零値により構成されていないならば、これらの値に全パワーを消費する形で実行されることになるものである）、又は、このような計算を電力が削減されるように実行するように構成される。好ましくは、エンジンは、このような各半行（又は半列）中の奇数データ値が零値であるか否かを判定し、このような各半行（又は半列）中の偶数データ値が零値であるか否かを判定し、零値であることが判定された各データ値に（より多くのパワーを消費する形で）実行されることになる乗算計算を電力が削減されるように実行して、関連の行又は列を変換するようにも構成される。

【００４８】

上記の何れかの実施例の変形例では、回路７又は１１によって変換されるべきブロックの各行又は各列（又は、行又は列の部分集合）を構成するデータ値は、行又は列（又はこれらの部分集合）における区画を構成する各個別の部分集合が完全に零値により構成されているか（又は零値であるか）否かに関する判定前又は判定中に並べ替えられる。

例えば、行（又は列）が値x_iにより構成されており、ここで、ｉが０≦ｉ≦Ｎ－１の範囲に入る整数であり、Ｎが偶数整数である場合に、

この区画は行（列）中の別個の第１の部分集合及び第２の部分集合により構成され、

第１の部分集合は行（列）中の別個の第３の部分集合及び第４の部分集合により構成され、

第２の部分集合は行（列）中の別個の第５の部分集合及び第６の部分集合により構成され、

第１の部分集合はｉが０≦ｉ≦（Ｎ／２）－１の範囲に入る整数である場合の値x_iにより構成され、

第２の部分集合はｉがＮ／２≦ｉ≦Ｎ－１の範囲に入る整数である場合の値x_iにより構成され、

第３の部分集合は、ｉが０≦ｉ≦（Ｎ／４）－１の範囲に入る整数である場合の値x_iにより構成されるか、第１の部分集合の偶数値により構成されるか、又は、第１の部分集合を並べ替えた部分集合の偶数値により構成され、

第４の部分集合は、ｉがＮ／４≦ｉ≦（Ｎ／２）－１の範囲に入る整数である場合の値x_iにより構成されるか、第１の部分集合の奇数値により構成されるか、又は、第１の部分集合を並べ替えた部分集合の奇数値により構成され、

第５の部分集合は、ｉがＮ／２≦ｉ≦（３Ｎ／４）－１の範囲に入る整数である場合の値x_iにより構成されるか、第２の部分集合の偶数値により構成されるか、又は、第２の部分集合を並べ替えた部分集合の偶数値により構成され、

第６の部分集合は、ｉが３Ｎ／４≦ｉ≦Ｎ－１の範囲に入る整数である場合に値x_iにより構成されるか、第２の部分集合の奇数値により構成されるか、又は、第２の部分集合を並べ替えた部分集合の奇数値により構成される。

【００４９】

図３のブロックＩの行Ｒ４を再び参照する。このような行Ｒ４内の偶数係数は零値であり、このような行内の奇数係数は完全に零値により構成されていない。このような行Ｒ４を変換するため、本発明の変換エンジンの一部の実施形態は、行の偶数係数に行変換演算を実行する乗算器への入力がこのような行変換演算の実行中に切り替わることを阻止することにより、消費パワーを削減することになる。これは、変換中に乗算器によって消費されるスイッチングパワーを削減することに（すなわち、動的消費パワーを削減するために）役立つ。

【００５０】

図３を再び参照する。ブロックＩの行Ｒ４では、行全体中の最初の係数だけが非零（符号化された）係数である。本発明の変換エンジンの好ましい実施形態は、このタイプの行

又は列（すなわち、最初のデータ値だけが非零である行又は列）を認識し、完全に非零値により構成されている行又は列を変換するために必要とされるパワーより少ないパワーを消費する形でこのような行又は列を変換するように構成される。例えば、（図２に示す構造を有する本発明の変換エンジンの実施形態の）回路７の好ましい実施例は、最初の値が非零である入力データ行に対する行変換を、行の非零係数だけに応じて、行の全データ値に通常の行変換（全パワーを消費することになる変換）を実行することなく、電力が削減されるように実行し、バッファ９に得られる部分的に変換されたデータ値を記憶するように構成される。

【００５１】

本発明の変換エンジンの好ましい実施形態は、ブロックの個別のデータ値が変換されるべきか否かを検出し（例えば、係数バイアスを用いて係数に逆直接コサイン変換を施すためにＤＣＴ係数のブロックの個別の係数に零検出を実行し）、零値であることが判定された全データ値に関して、少なくとも一つの上記零値を変換するために利用されるべき少なくとも一つの乗算器への少なくとも一つ（そして、好ましくは、一つずつ）の入力がその先行値から切り替わらないことを保証することにより消費パワーを削減する。これはエンジン内の動的消費パワーを削減する。

【００５２】

図２に示す構造を有する本発明の変換エンジンの実施形態の好ましい実施例では、回路７は、この回路によって生成される、部分的に変換されたデータ値を内部バッファ９に記憶する。回路１１は、バッファ９からこのデータを読み出し、データの各列（又は行）に列（又は行）変換を実行する。通常、回路１１は、データの少なくとも一つの列（行）がバッファ９内に存在し、変換のために利用可能になると直ぐに、列（行）変換計算を実行する。よって、好ましい実施形態では、回路７は、この回路７が実行する行（列）変換演算を並び替えて、できる限り迅速に、データの完全な列（行）に列（行）変換を実行するために十分な回路１１用の部分的に変換されたデータを生成する（バッファ９内に記憶する）ように構成される。このように並び替える方式（例えば、インターリーブする方式）で、行（列）変換を実行するように回路７を構成することによって、システムによる消費パワーを（例えば、内部バッファ記憶のため消費されるパワーを最小限に抑えることにより）低減することができる。

【００５３】

或いは、対称変換が図２に示す構造を有する本発明の変換エンジンの実施形態に従って実行される場合には、回路７は、変換の対称的な性質を利用する方式で、付加的な行変換計算を実行する前に２列のデータを生成する（そして、バッファ９内に記憶する）ために十分な行変換計算を実行するように、又は、付加的な列変換計算を実行する前に２行のデータを生成する（そして、バッファ９内に記憶する）ために十分な列変換計算を実行するように構成される。回路７のこのような実施形態によって実行される行（又は列）変換は、一時記憶されたデータの更なる完全な列（行）に列（行）変換を実行するために十分な回路１１用の部分的に変換されたデータを、できる限り迅速に生成し（そして、バッファ９内に記憶し）続ける。行変換計算を実行するために回路７を、付加的な行変換計算を実行する前に、１列のデータを出力するように実施するか又は２列のデータを出力するように実施するかに関する判定は、性能要件とバッファメモリ要件との間のトレードオフに依存する。バッファ９が１列の部分的に変換されたデータを記憶する容量を保有することだけが必要とされる方式で回路７が行変換を実行するように実施される場合には、バッファ９の要求された容量を最小限に抑える利点は、典型的に、性能低下（例えば、データ値の完全なブロックに逆変換を実行するために本発明の変換エンジンのサイクル数が増加すること）を犠牲にして得られることになることが理解されるべきである。

【００５４】

本発明に従って従来のＨ２６４に基づく変換（本当の乗算を用いることなく実行され得る整数変換）を実行するために、本発明の変換演算は乗算器無しで実施することができる。Ｈ２６４に基づく変換は、乗算器ではなくシフト器を使用して実施し得る。本発明によ

る変換回路の回避は、これら実施例であっても、変換計算を高速化する（そして、変換計算によって消費されるエネルギーを低下させる）。

【００５５】

次に、図４のフローチャートを参照して本発明の一つのクラスの実施形態をより詳細に説明する。図４のフローチャートは、入力データブロックの系列中の各ブロックに実行される２Ｄ変換（例えば、逆離散コサイン変換）に関するものであり、各ブロックは入力データ値をもつ行及び列を有しており、２Ｄ変換は行変換及び列変換を含んでいる。本発明によれば、２Ｄ変換は、各ブロックの全行に行変換を実行して部分的に変換されたデータのブロックを生成し、次いで、部分的に変換されたデータのブロックの各列に列変換を実行するか、又は、各ブロックの全列に列変換を実行して部分的に変換されたデータのブロックを生成し、次いで、部分的に変換されたデータのブロックの各行に行変換を実行することによって実行される。図４のフローチャートは、データ値の行へのこのような行変換、又は、データ値の列へのこのような列変換について記載している（例えば、本発明を具現化する図２の回路７の実施による周波数係数の行又は列の変換、又は、本発明を具現化する図２の回路１１の実施による、バッファ９から読み出された部分的に変換されたデータの行又は列の変換について記載している）。簡単にするために、図４の以下の説明は、データ値の行が変換されることを仮定している。この説明中の「行」への全ての参照を、「列」への参照によって置き換えて、本発明のその他の実施形態を説明することができることを理解されたい。

【００５６】

本発明に従って逆変換されるべき典型的なブロックは、（例えば、逆変換がＨ２６４に基づく変換であるならば）１行当たりに４個の係数を有し、又は、（例えば、逆変換が周波数係数の８×８形のブロック上のＩＤＣＴであるならば）１行当たりに８個の係数を有する。図４は何れのケースでも適用可能である。

【００５７】

図４に示すステップを実行する前に、本発明の典型的な実施形態の変換エンジンは、変換されるべきデータのブロックが完全に零値により構成されているか否かを判定する。ブロックが完全に零値により構成されていることを判定すると、変換エンジンは、このブロックへの変換演算（行変換演算と列変換演算の双方）を回避させる（例えば、変換エンジンは、ブロックの値に実際の変換演算を実行することなく、ブロックに応答して零を出力する）。変換エンジンは、変換されるべきデータのブロックが少なくとも一つの非零値を含むと判定したならば、行変換を開始する（すなわち、変換エンジンは図４のステップ２０を実行する）。

【００５８】

図４のフローチャートに従って実行されるべき行変換の開始後（ステップ２０）、変換されるべき第１行のデータ値がバッファメモリから読み出される（ステップ２２）。バッファメモリは、（回路７が図４のフローチャートを実施するならば）図２の回路７の入力に接続されているバッファメモリ、又は（回路１１が図４のフローチャートを実施するならば）図２のバッファメモリ９である。

【００５９】

ステップ２４において、変換エンジンは、現在の行（ステップ２２の最新の実行中に読み出された行）が少なくとも一つの非零値を含むか否かを判定する。もし含まないならば、変換エンジンは、（行のデータ値に実際の行変換演算を実行することなく）その行を行変換した結果を示す零を出力し（ステップ２５）、バッファメモリから次の行を読み出す（ステップ２２の次の実行）。現在の行が少なくとも一つの非零値を含むならば、エンジンは現在の行の前半分（例えば、４個のデータ値により構成される行の最初の２個のデータ値）が少なくとも一つの非零値を含むか否かを判定する（ステップ２６）。

【００６０】

現在の行の前半分が少なくとも一つの非零値を含まないならば、エンジンは、（行の前半分のデータ値に実際の行変換演算を実行することなく）その行の前半分を行変換した結

果を示す零を出力し（ステップ２８）、現在の行の後半分（例えば、４個のデータ値により構成される行の後半の２個のデータ値）が少なくとも一つの非零値を含むか否かを判定する（ステップ４０）。

【００６１】

現在の行の前半分が少なくとも一つの非零値を含むならば、エンジンは現在の行の前半分中の前半分が少なくとも一つの非零値を含むか否かを判定し（ステップ３０）、現在の行の前半分中の後半分が少なくとも一つの非零値を含むか否かを判定する（ステップ３２）。

【００６２】

現在の行の前半分中の前半分が少なくとも一つの非零値を含むならば（例えば、単一の非零値により構成されるか、又は、少なくとも一つの非零値を含む２個の値により構成されるならば）、エンジンは、行の前半分中の前半分のデータ値に行変換演算を実行し（ステップ３６）、得られる変換されたデータをバッファメモリに記憶する（ステップ３９）。ステップ３９を実行するために利用されるバッファメモリは、（回路１１が図４のフローチャートを実施するならば）図２の回路１１の出力に接続されたバッファメモリであってもよく、（回路７が図４のフローチャートを実施するならば）図２のバッファメモリ９であってもよい。

【００６３】

現在の行の前半分中の前半分が非零値を含まないならば、エンジンはステップ３７を実行する。そのステップ３７において、エンジンは、

（ａ）現在の行の前半分中の前半分を行変換した結果を示す一つ以上の零を出力する（現在の行の前半分中の前半分における一つ以上のデータ値に実際の行変換計算を実行することなく）。このような各零値は、その後にバッファメモリに記憶される（ステップ３９）。或いは、エンジンは、

（ｂ）現在の行の前半分中の前半分における（複数の）データ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各出力値はその後にバッファメモリに記憶される（ステップ３９）。例えば、エンジンは、これを、少なくとも一つの定数がアサートされる（少なくとも一つの入力を備える）第１の入力セットと現在の行の前半分中の前半分における各データ値がアサートされる（少なくとも一つの入力を備える）第２の入力セットとを有する乗算回路を作動させて、少なくとも、現在の行の前半分中の前半分における各データ値に対する乗算演算を、第１の入力セットと第２の入力セットの少なくとも一方にアサートされた値を更新することなく実行することによって行なうことができ、さもなければ乗算回路の（複数の）関連の入力を切り替えるため消費されることになるパワーを消費することを回避することができる。エンジンがＩＤＣＴを実行する場合には、第１の入力セットにアサートされた各定数はコサイン定数であり、一実施例では、第２の入力セットにアサートされたデータ値が零値であるときに、乗算の結果はコサイン定数の値とは無関係に零になるので、エンジンは上記零値で乗算されるべきコサイン定数が更新されること（それ以前の値から変更されること）を阻止する。別の実施例では、エンジンは、（何れかの乗算器入力の切り替えを阻止するために）（零データ値で乗算されるべき）第１の入力セットにアサートされるコサイン定数の更新を阻止し、第２の入力セットの対応の入力にアサートされているデータ値の更新を阻止し、（実際に２個のオペランドを一緒に乗算することなく）零データ値とコサイン定数とを乗算した結果を示す零値をアサートする（例えば、零値を処理パイプラインの出力に多重化する）。

或いは、エンジンは、

（ｃ）現在の行の前半分中の前半分における少なくとも一つの値を行変換した結果を示す一つ以上の零を出力し（一つ以上のデータ値に実際の行変換計算を実行することなく）、現在の行の前半分中の前半分における少なくとも一つのその他のデータ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各零値及び出力値はその後にバッファメモリに記憶される（ステップ３

９）。

【００６４】

現在の行の前半分中の後半分が少なくとも一つの非零値を含むならば（例えば、単一の非零値により構成されるか、又は、少なくとも一つの非零値を含む２個の値により構成されているならば）、エンジンは、行の前半分中の後半のデータ値に行変換演算を実行し（ステップ３４）、得られる変換されたデータをバッファメモリに記憶する（ステップ３９）。

【００６５】

現在の行の前半分中の後半分が非零値を含まないならば、エンジンはステップ３５を実行する。そのステップ３５において、エンジンは、

（ａ）現在の行の前半分中の後半分を行変換した結果を示す一つ以上の零を出力する（現在の行の前半分中の後半分における一つ以上のデータ値に実際の行変換計算を実行することなく）。このような各零値は、その後にバッファメモリに記憶される（ステップ３９）。

或いは、エンジンは、

（ｂ）現在の行の前半分中の後半分における（複数の）データ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各出力値はその後にバッファメモリに記憶される（ステップ３９）。例えば、エンジンは、これを、少なくとも一つの定数がアサートされる（少なくとも一つの入力を備える）第１の入力セットと現在の行の前半分中の後半分における各データ値がアサートされる（少なくとも一つの入力を備える）第２の入力セットとを有する乗算回路を作動させて、少なくとも、現在の行の前半中の後半の各データ値に対する乗算演算を、第１の入力セットと第２の入力セットの少なくとも一方にアサートされた値を更新することなく実行することによって行うことができ、さもなければ乗算回路の（複数の）関連の入力を切り替えるために消費されることになるパワーを消費することを回避する。エンジンがＩＤＣＴを実行するケースでは、第１の入力セットにアサートされた各定数はコサイン定数であり、一実施例では、第２の入力セットにアサートされたデータ値が零値であるときには、乗算の結果はコサイン定数の値とは無関係に零になるため、エンジンは上記零値で乗算されるべきコサイン定数が更新されること（それ以前の値から変更されること）を阻止する。別の実施例では、エンジンは、（何れかの乗算器入力の切り替えを阻止するために）（零データ値で乗算されるべき）第１の入力セットにアサートされるコサイン定数の更新を阻止し、第２の入力セットの対応の入力にアサートされているデータ値の更新を阻止し、（実際に２個のオペランドを一緒に乗算することなく）零データ値とコサイン定数とを乗算した結果を示す零値をアサートする（例えば、零値を処理パイプラインの出力に多重化する）。

或いは、エンジンは、

（ｃ）現在の行の前半分中の後半分における少なくとも一つの値を行変換した結果を示す一つ以上の零を（一つ以上のデータ値に実際の行変換計算を実行することなく）出力し、現在の行の前半分中の後半分における少なくとも一つのその他のデータ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各零値及び出力値はその後にバッファメモリに記憶される（ステップ３９）。

【００６６】

ステップ３４及び３６、ステップ３４及び３７、ステップ３５及び３６、又は、ステップ３５及び３７で生成又はアサートされた全データ値がバッファメモリに記憶された後（ステップ３９）、エンジンは、現在の行の後半分が少なくとも一つの非零値を含むか否かを判定する（ステップ４０）。現在の行の後半分が少なくとも一つの非零値を含まないならば、変換エンジンは、行の後半分を行変換した結果を示す零を（行の上記後半分のデータ値に実際の行変換演算を実行することなく）出力し（ステップ４１）、バッファメモリから変換されるべき次の行を読み出す（ステップ２２の別の実行）。

【００６７】
現在の行の後半分が少なくとも一つの非零値を含むならば、エンジンは、現在の行の後半分中の前半分が少なくとも一つの非零値を含むか否かを判定し（ステップ４２）、現在の行の後半分中の後半分が少なくとも一つの非零値を含むか否かを判定する（ステップ４４）。
【００６８】
現在の行の後半分中の前半分が少なくとも一つの非零値を含むならば（例えば、単一の非零値により構成されるか、又は、少なくとも一つの非零値を含む２個の値により構成されるならば）、エンジンは、行の後半分中の前半分におけるデータ値に行変換演算を実行し（ステップ４８）、得られる変換されたデータをバッファメモリに記憶する（ステップ５０）。ステップ５０を実行するために利用されるバッファメモリは、（回路１１が図４のフローチャートを実施するならば）図２の回路１１の出力に接続されたバッファメモリであってもよく、（回路７が図４のフローチャートを実施するならば）図２のバッファメモリ９であってもよい。
【００６９】
現在の行の後半分中の前半分が非零値を含まないならば、エンジンはステップ４９を実行する。そのステップ４９において、エンジンは、
（ａ）現在の行の後半分中の前半分を行変換した結果を示す一つ以上の零を（現在の行の後半中の前半の一つ以上のデータ値に実際の行変換計算を実行することなく）出力する。このような各零値は、その後にバッファメモリに記憶される（ステップ５０）。
或いは、エンジンは、
（ｂ）現在の行の後半分中の前半分における（複数の）データ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各出力値はその後にバッファメモリに記憶される（ステップ５０）。例えば、エンジンは、これを、少なくとも一つの定数がアサートされる（少なくとも一つの入力を備える）第１の入力セットと現在の行の後半分中の前半分の各データ値がアサートされる（少なくとも一つの入力を備える）第２の入力セットとを有する乗算回路を作動させて、少なくとも、現在の行の後半分中の前半分の各データ値に対する乗算演算を、第１の入力セットと第２の入力セットの少なくとも一方にアサートされた値を更新することなく実行することによって行うことができ、さもなければ乗算回路の（複数の）関連の入力を切り替えるために消費されることになるパワーを消費することを回避する。エンジンがＩＤＣＴを実行するケースでは、第１の入力セットにアサートされた各定数はコサイン定数であり、一実施例では、第２の入力セットにアサートされたデータ値が零値であるときには、乗算の結果はコサイン定数の値とは無関係に零になるため、エンジンは上記零値で乗算されるべきコサイン定数が更新されること（それ以前の値から変更されること）を阻止する。別の実施例では、エンジンは、（何れかの乗算器入力の切り替えを阻止するために）（零データ値で乗算されるべき）第１の入力セットにアサートされるコサイン定数の更新を阻止し、第２の入力セットの対応の入力にアサートされているデータ値の更新を阻止し、（実際に２個のオペランドを一緒に乗算することなく）零データ値をコサイン定数で乗算した結果を示す零値をアサートする（例えば、零値を処理パイプラインの出力に多重化する）。
或いは、エンジンは、
（ｃ）現在の行の後半分中の前半分の少なくとも一つの値の行変換バージョンを示す一つ以上の零を（一つ以上のデータ値に実際の行変換計算を実行することなく）出力し、現在の行の後半分中の前半分の少なくとも一つのその他のデータ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各零値及び出力値はその後にバッファメモリに記憶される（ステップ５０）。
【００７０】
現在の行の後半中の後半が少なくとも一つの非零値を含むならば（例えば、単一の非零値により構成されるか、又は、少なくとも一つの非零値を含む２個の値により構成されて

いるならば）、エンジンは、行の後半中の後半のデータ値に行変換演算を実行し（ステップ４６）、結果として得られる変換されたデータをバッファメモリに記憶する（ステップ５０）。

【００７１】

現在の行の後半分中の後半分が非零値を含まないならば、エンジンはステップ４７を実行する。そのステップ４７において、エンジンは、

（ａ）現在の行の後半分中の後半分を行変換した結果を示す一つ以上の零を（現在の行の後半分中の後半分における一つ以上のデータ値に実際の行変換計算を実行することなく）出力する。このような各零値は、その後にバッファメモリに記憶される（ステップ５０）。

或いは、エンジンは、

（ｂ）現在の行の後半分中の後半分における（複数の）データ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各出力値はその後にバッファメモリに記憶される（ステップ５０）。例えば、エンジンは、これを、少なくとも一つの定数がアサートされる（少なくとも一つの入力を備える）第１の入力セットと現在の行の後半分中の後半分における各データ値がアサートされる（少なくとも一つの入力を備える）第２の入力セットとを有する乗算回路を作動させて、少なくとも、現在の行の後半分中の後半分における各データ値に対する乗算演算を、第１の入力セットと第２の入力セットの少なくとも一方にアサートされた値を更新することなく実行することによって行うことができ、さもなければ乗算回路の（複数の）関連の入力を切り替えるために消費されることになるパワーを消費することを回避する。エンジンがＩＤＣＴを実行するケースでは、第１の入力セットにアサートされた各定数はコサイン定数であり、一実施例では、第２の入力セットにアサートされたデータ値が零値であるときには、乗算の結果はコサイン定数の値とは無関係に零になるため、エンジンは上記零値で乗算されるべきコサイン定数が更新されること（それ以前の値から変更されること）を阻止する。別の実施例では、エンジンは、（何れかの乗算器入力のトグリングを阻止するために）（零データ値で乗算されるべき）第１の入力セットにアサートされるコサイン定数の更新を阻止し、第２の入力セットの対応する入力にアサートされているデータ値の更新を阻止し、（実際に２個のオペランドを一緒に乗算することなく）零データ値をコサイン定数で乗算する結果を示す零値をアサートする（例えば、零値を処理パイプラインの出力に多重化する）。

或いは、エンジンは、

（ｃ）現在の行の後半分中の後半分における少なくとも一つの値を行変換した結果を示す一つ以上の零を（一つ以上のデータ値に実際の行変換計算を実行することなく）出力し、現在の行の後半分中の後半分における少なくとも一つのその他のデータ値に行変換計算を電力が削減されるように実行することにより生成された少なくとも一つのデータ値を出力する。このような各零値及び出力値はその後にバッファメモリに記憶される（ステップ５０）。

【００７２】

ステップ４６及び４８、ステップ４６及び４９、ステップ４７及び４８、又は、ステップ４７及び４９で生成又はアサートされた全データ値がバッファメモリに記憶された後（ステップ５０）、エンジンは、行変換が現在のブロックの全行に実行されたか否かを判定する（ステップ５２）。行変換が現在のブロックの全行に実行されているならば、エンジンは、行変換演算を終了する状態に入る（ステップ５４）。その後、エンジンは、変換されるべきデータ値をもつ次のブロックの処理を開始するためにステップ２０を再び実行可能とする。行変換が現在のブロックの全行に対して実行されていないならば、エンジンはバッファメモリから（変換されるべき現在のブロックの）次の行を読み出す（ステップ２２の別の実行）。

【００７３】

図５は、本発明を具現化する図２の回路７（又は回路１１）の典型的な実施に利用され

る変換回路のブロック図である。４×４形のデータのブロックを変換するため設計された回路７（又は１１）の実施例は、通常、変換されるべき各行又は各列の４個のデータ値Ｉ０、Ｉ１、Ｉ２及びＩ３を変換するための図５の回路の一つのユニットを備える。８×８形のデータのブロックを変換するため設計された回路７（又は１１）の実施例は、通常、図５の回路の２個のユニットを備えることになり、一方のユニットは変換されるべき各行又は列の前半分の４個のデータ値Ｉ０、Ｉ１、Ｉ２及びＩ３を変換し、同一のユニットは変換されるべき各行又は列の後半分の４個のデータ値（Ｉ４、Ｉ５、Ｉ６及びＩ７）を（図５に示す定数Ｃ０、Ｃ１、Ｃ２及びＣ３の代わりに第２のコサイン定数の集合Ｃ４、Ｃ５、Ｃ６及びＣ７を使用して）変換する。

【００７４】

図５の乗算ユニットＭ１、Ｍ２、Ｍ３及びＭ４のそれぞれは、（コサイン定数Ｃ０、Ｃ１、Ｃ２及びＣ３のうちの一つがアサートされる）第１の入力と、（データ値Ｉ０、Ｉ１、Ｉ２及びＩ３のうちの一つがアサートされる）第２の入力と、２個のオペランドの積を示すデータ値をアサートする出力と、を有する。図５の回路は、（前処理ロジック１００と呼ぶ）前処理及び制御ロジック１００と、加算回路Ａ１、Ａ２、Ａ３、Ａ４及びＡ５と、減算回路Ｓ１と、（フリップフロップとして実施された）遅延回路Ａ及びＢとを更に備えており、これらは図示したように接続されており、それにアサートされた４個の入力値の各組にパイプライン方式行変換又は列変換を実行して、各入力値に応じて出力値Ｙ０（又はＹ１）を生成するように構成されている。動作中に、遅延回路Ａは、通常、加算回路Ａ５、減算回路Ｓ１、及び、加算回路Ａ３のうちの（ロジック１００によってアサートされた制御信号によって決定されるように）選択された一つずつへの加算回路Ａ３の出力のアサーションを、パイプライン方式変換の１サイクルずつ遅延させる。動作中に、遅延回路Ｂは、典型的に、加算回路Ａ５、減算回路Ｓ１、及び、加算回路Ａ４のうちの（ロジック１００によってアサートされた制御信号によって決定されるように）選択された一つずつへの加算回路Ａ４の出力のアサーションをパイプライン方式変換の１サイクルずつ遅延させる。

【００７５】

図５の回路が４×４形のデータのブロックを変換するように設計された回路７（又は１１）の実施例に含まれている場合には、前処理ロジック１００は、図４の零（非零）値判定演算を実行するように構成される。図５の回路の２個のユニットが８×８形のデータのブロックを変換するように設計された回路７（又は１１）の実施例に含まれている場合には、各ユニット１００は、変換されるべきデータ値をもつ各行又は列の異なる部分集合（例えば、半分）に図４の零（非零）値判定演算を実行するように構成される。回路７（又は１１）の何れの実施例であっても、前処理ロジック１００は、変換されるべき入力データ値を関連のバッファメモリ（例えば、図２のバッファ９、又は図２の回路７の入力に接続されたバッファメモリ）から読み出し、図４によって要求されるその他の演算を実施するために必要とされる制御信号を生成する（そして、図５のその他のエレメントのうちの適切なエレメントへアサートする）ように構成される。例えば、ロジック１００が（例えば、図４のステップ３０中に）係数Ｉ０は零値であると判定したならば、ロジック１００は、前の動作サイクル中に乗算器Ｍ１へアサートされていた係数Ｃ０を、乗算器Ｍ１が零値係数Ｉ０に定数を乗算する後続の各サイクル中に乗算器Ｍ１へアサートさせ続けるように制御信号をアサートすることができる。

【００７６】

典型的な実施例では、前処理ロジック１００は、（例えば、バッファ９から）変換されるべき４個の新しい入力データ値をもつ各集合を受信し次第、出力値Ｙ０及びＹ１を零にリセットする。このケースでは、前処理ロジック１００が（例えば、図４のステップ２５、２８、４１、３５、３７、４７又は４９において）入力値のうちの一つに応じて零値（例えば、零値のＹ０又はＹ１）が出力されるべきであると判定したときに、前処理ロジックは、これら回路エレメントＭ１、Ｍ２、Ｍ３、Ｍ４、Ａ１、Ａ２、Ａ３、Ａ４、Ａ、Ｂ、Ａ５及びＳ１の入力の全ての切り替えを、さもなければ回路要素がこの入力値を変換す

るために動作することになる（複数の）サイクルにおいて阻止し、代わって、Ｙ０（又はＹ１）のリセットされた零値を関連のバッファ（例えば、バッファメモリ９、又は回路１１の出力に接続されているバッファメモリ）へアサートさせる。

【００７７】

変換されるべきデータ値の並べ替えを必要とする実施形態では、前処理ユニット１００が、好ましくは、このような並べ替えを実行するように構成される。例えば、図４を参照して説明した変換に類似しているが、半行又は半列（又は、半行又は半列の半分）の前半分（又は後半分）中のデータ値が少なくとも一つの非零値を含むか否かを判定する各ステップが、半行又は半列（又は、半行又は半列の半分）の偶数（又は奇数）データ値が少なくとも一つの非零値を含むか否かを判定するステップによって置き換えられている変換を実行するために、前処理ユニット１００は、必要な判定を実行するために、アサートされるべきデータ値を並べ替え可能である。例えば、ユニット１００は、並べ替えられた半行又は並べ替えられた半列（又は、並べ替えられた半行又は並べ替えられた半列の半分）の前半分（又は後半分）中のデータ値が少なくとも一つの非零値を含むか否かを判定し、それによって、並べ替えられていない場合の半行又は半列の（又は、並べ替えられていない場合の半行又は半列の半分）中の偶数（又は奇数）データ値が少なくとも一つの非零値を含むか否かを判定可能である。前処理ユニット１００の一部の実施例は、半行（又は部分行の半分）又は半列（又は部分列の半分）ではない行又は列（又は、部分行又は部分列）の部分集合の並べ替えを実行することも考えられる。

【００７８】

図６は、本発明のパイプライン方式ビデオ復号システムの実施形態の要素のブロック図である。図６に示す設計に関する多数の変形が本発明のその他の実施形態での実施用に考えられる（例えば、零検出回路１３２、１３４及び１３５－１３８が省かれ、その機能は零検出回路１３０又はその他の零検出回路によって実行される）。図６のシステムは、前処理ユニット１２０、変換エンジン１２２、及び、後処理ユニット１２４を備えており、これらは、図示したように接続されている。

【００７９】

ユニット１２０はまた、入力データ値（例えば、ビデオピクセルのブロックに２次元ＤＣＴ又は逆ＤＣＴ、若しくはその他の２Ｄ変換を実行することにより生成された入力周波数領域係数）をもつブロックの系列を受信するよう接続されている。ユニット１２０は、図６のシステムを通る１回目のパスの後に、後処理ユニット１２４から出力された（図６において、「第１ステージからの出力」及び「第１ステージ出力」として特定されている）データ値を受信するように接続されてり、その結果、図６の回路を通る２回目のパス中に（オプションとして、ユニット１２４からの出力後、かつ、ユニット１２０へのアサーション前にバッファメモリに一時記憶されている）これらのデータ値に第２ステージの処理が施される（例えば、データ値の列が図６の回路を通る１回目のパス中に列変換を施されているときにデータ値の行に行変換を実行するため）。

【００８０】

ユニット１２０のマルチプレクサ１２９は、（図６における処理が施されていない）入力データ値、又は、（図６のシステムを通る先行のパスにおいて処理が施されている）後処理ユニット１２４から出力されたデータ値の何れかを選択し、選択された値を零検出ロジック１３０へアサートする。選択された値は（オプションとして、ロジック１３０が選択された値は完全に零値により構成されていると判定したときを除いて）零検出ロジック１３０を通過する。オプションとして、その後に、ユニット１２０内のその他の回路は、ロジック１３０を通された値に予備的な処理演算を実行して、前処理済みのデータ値を生成する（これらの演算は図６のシステムを使用して実施されるべき特有の変換を実施することが要求されているときに実行される）。ユニット１２０は、（予備的な処理がロジック１３０の出力に実行されていないならば、ロジック１３０の出力と同一である）前処理済みのデータ値を変換エンジン１２２へアサートする。

【００８１】

前処理ユニット１２０は、通常、アサートされたデータ値の各集合をエンジン１２２が実施すべき変換を実行するのに適した部分集合に分離するように構成されている。ユニット１２０が部分集合に分離する方式は、実施されるべき変換（例えば、８×８形ＤＣＴ、８×８形ＩＤＣＴ、又は、８×８形アマダール）に依存し、外部ユニットからユニット１２０へアサートされた制御信号によって決定することができ、その結果、部分集合選択は、実施されるべき変換に基づいてカスタマイズすることが可能となる。

【００８２】

変換エンジン１２２は、ユニット１２０から受け取るデータ値の各ブロックに２Ｄ変換を実行して、変換済みのデータ値をもつブロック（例えば、部分的に復号されたビデオピクセルをもつブロック）を生成し、変換済みのデータ値をオプションとしての更なる処理のための後処理ユニット１２４へアサートする。より詳細には、変換エンジン１２２は、ユニット１２０から変換エンジンへアサートされた４個のデータ値の集合毎にパイプライン方式行変換又は列変換を実行して、４個のデータ値のそれぞれに応じて出力値を生成するように構成されている。このような４個のデータ値からなる各集合は、通常、データ値のブロックの行（又は列）であるか、又は、データ値のブロックの行（又は列）の部分集合である。

【００８３】

後処理ユニット１２４は、オプションとして（すなわち、図６のシステムによって実行されている特有の変換に適しているときに）、エンジン１２２から受信された変換済みの値に後処理演算を実行して、出力データ値を生成する。出力データ値は、ユニット１２４からシステムバスへアサートされ、及び／又は、ユニット１２０の少なくとも一つの入力へ（又は、ユニット１２０の少なくとも一つの入力へアサートされる起点となるバッファメモリへ）戻される。通常、ユニット１２４は、（必要に応じて）丸め、飽和、及び、最終的なパッキングを実行する。ユニット１２４によって実行される特定の後処理演算は、図６の回路がユニット１２２への入力に対して第１ステージ処理を実施するか、又は第２ステージの処理を実施するかに依存する。

【００８４】

図６の変換エンジン１２２は、定数Ｃ０、Ｃ１、Ｃ２及びＣ３の集合を使用して変換されるべき各行又は列の４個のデータ値（Ｉ０、Ｉ１、Ｉ２及びＩ３）を変換することを始めとして、４×４形のデータのブロックを変換するように設計されている。本発明の一部のその他の実施形態では、８×８形のデータのブロックを変換する回路は２個の変換エンジン１２２を含むことになり、一方のエンジンは変換されるべき各行又は列の第１の部分集合（例えば、前半分）の４個のデータ値Ｉ０、Ｉ１、Ｉ２及びＩ３を変換し、同一のユニットは（図６のブロック１２２に示す定数Ｃ０、Ｃ１、Ｃ２及びＣ３の代わりに第２の定数の集合Ｃ４、Ｃ５、Ｃ６及びＣ７を使用して）変換されるべき各行又は列の第２の部分集合（例えば、後半分）の４個のデータ値（Ｉ４、Ｉ５、Ｉ６及びＩ７）を変換する。利用されるべき定数（例えば、Ｃ０、Ｃ１、Ｃ２及びＣ３）は、外部ユニットからエンジン１２２へアサートされた制御信号によって決定され、その結果、定数の選択は実行されるべき変換に基づいてカスタマイズ可能である。

【００８５】

通常、前処理ユニット１２０の零検出ロジック１３０は、変換されるべきデータをもつブロック全体（変換されるべきデータの４×４形のブロックの４行又は４列の全て）が完全に零値により構成されているか否かを判定するように構成されている。例えば、零検出ロジックは、ブロックの第１行（列）を下流の回路へ送る前に、ブロックの全行（又は全列）にこのような判定を実行するのに十分なレイテンシーを提供するシフトレジスタを有する。ブロックが完全に零値により構成されていることを判定すると、ロジック１３０は、図６のシステムのその他の要素へ適切な制御ビットをアサートし、（例えば、上述したように）エンジン１２２によってブロックに対して後に実行されることになる変換演算を回避させ、また、通常、エンジン１２２及びユニット１２４の一方又は双方によるブロックの値への実際の変換演算を実行することなく、ブロックに応じて所定の値（例えば、零

）をユニット１２４のマルチプレクサ１３９から出力させる。
【００８６】
前の段落に記載したように、前処理ユニット１２０の零検出ロジック１３０がデータのブロック全体が完全に零値により構成されているか否かを判定するように構成されているか否かとは無関係に、ロジック１３０は、マルチプレクサ１２９からロジック１３０へアサートされた個別の行（又は列）のデータ値に零検出を実行して、行（又は列）の全データ値が零値であるか否かを判定するように構成されている（すなわち、ロジック１３０は図４のステップ２４を実行する）。このような各行（又は列）が４個のデータ値により構成されているならば、ロジック１３０は、行（列）が４個の零ビットにより構成されているか否かを判定する。ロジック１３０が、行（又は列）は完全に零ビットにより構成されていると判定するならば、ロジック１３０は、適切な制御ビットをアサートし、当該制御ビットが、行（列）に対してエンジン１２２によって後に実行されることになる変換演算を回避させ、通常、エンジン１２２及びユニット１２４の一方又は双方による行（列）の値への実際の変換演算を実行することなく、行（列）に応じて所定の値（例えば、零）をユニット１２４のマルチプレクサ１３９から出力させる。
【００８７】
一部の実施例では、ロジック１３０は、マルチプレクサ１２９からロジック１３０へアサートされたデータ値の個別の行（又は列）の１番目のデータ値が非零値であり、行（又は列）のその他の全値が零値であるか否かを判定する。最初の非零値を除き零値により構成された行（列）の特定に応答して、このようなロジック１３０の実施例は、適切な制御ビットをアサートし、当該ビットが、行（列）に対してエンジン１２２によって後で実行されることになる変換演算を回避させ、通常、エンジン１２２及びユニット１２４の一方又は双方による行（列）の値への実際の変換演算を実行することなく、行（列）に応じて所定の値（例えば、零）をユニット１２４のマルチプレクサ１３９から出力させる。
【００８８】
エンジン１２２の零検出ロジック１３２は、ユニット１２０からエンジン１２２へアサートされた各行（又は列）の第１の部分集合（例えば、最初の２個のデータ値である前半分）に零検出を実行し、このような第１の部分集合の全データ値が零値であるか否かを判定するように構成されている（例えば、零検出ロジックは図４のステップ２６を実行する）。ロジック１３２が、第１の部分集合（例えば、半行又は半列）は完全に零ビットにより構成されていると判定するならば、ロジック１３２は、適切な制御ビットをアサートし、当該制御ビットが、第１の部分集合に対してエンジン１２２によって後に実行されることになる変換演算を回避させ、通常、エンジン１２２による第１の部分集合中の値への実際の変換演算を実行することなく、第１の部分集合に応じて所定の値（例えば、零）をエンジン１２２のマルチプレクサ１４０から出力させる。ロジック１３２が、現在の行（列）の第１の部分集合（例えば、前半）は少なくとも一つの非零値を含むと判定するならば、零検出ロジックユニット１３５及び１３６は、第１の部分集合中の１番目の値及び２番目の値に零検出を実行する。
【００８９】
ロジック１３５が現在の行（列）の第１の部分集合中の１番目のデータ値は零値であると判定するならば、ロジック１３５は、（第１の入力に変換定数Ｃ０がアサートされ、第２の入力に現在の行又は列の零データ値がアサートされている）乗算回路１４１に制御ビットをアサートし、回路１４１に、その第１の入力にアサートされた前の値を更新することなく、その第２の入力にアサートされた値への乗算演算を実行させ、さもなければ第１の入力を更新された定数Ｃ０の値に切り替えるために消費されることになるパワー消費を回避する。
【００９０】
同様に、ロジック１３６が現在の行（列）の第１の部分集合中の２番目のデータ値は零値であると判定するならば、ロジック１３６は、（第１の入力に変換定数Ｃ１がアサートされ、第２の入力に現在の行又は列の零データ値がアサートされている）乗算回路１４２

に制御ビットをアサートし、回路１４２に、その第１の入力にアサートされた前の値を更新することなく、その第２の入力にアサートされた値への乗算演算を実行させ、さもなければ第１の入力を更新された定数Ｃ１の値に切り替えるために消費されることになるパワー消費を回避する。

【００９１】

エンジン１２２の零検出ロジック１３４は、ユニット１２０からエンジン１２２へアサートされた各行（又は列）の第２の部分集合（例えば、もう一つの２個のデータ値である後半）に対して零検出を実行し、このような第２の部分集合の全データ値が零値であるか否かを判定するように構成されている（例えば、零検出ロジックは図４のステップ４０を実行する）。ロジック１３４が第２の部分集合は完全に零ビットにより構成されていると判定するならば、ロジック１３４は、適切な制御ビットをアサートし、当該制御ビットが、さもなければエンジン１２２によって第２の部分集合（例えば、半行又は半列）に対して後に実行されることになる変換演算を回避させ、通常、エンジン１２２による第２の部分集合中の値への実際の変換演算を実行することなく、第２の部分集合に応じて所定の値（例えば、零）をエンジン１２２のマルチプレクサ１４１から出力させる。ロジック１３４が現在の行（列）の第２の部分集合は少なくとも一つの非零値を含むと判定するならば、零検出ロジックユニット１３７及び１３７は、第２の部分集合中の１番目の値及び２番目の値に零検出を実行する。

【００９２】

ロジック１３７が現在の行（列）の第２の部分集合中の１番目のデータ値は零値であると判定するならば、ロジック１３７は、（第１の入力に変換定数Ｃ２がアサートされ、第２の入力に現在の行又は列の零データ値がアサートされている）乗算回路１４３に制御ビットをアサートし、回路１４３に、その第１の入力にアサートされた前の値を更新することなく、その第２の入力にアサートされた値への乗算演算を実行させ、さもなければ第１の入力を更新された定数Ｃ２の値に切り替えるために消費されることになるパワー消費を回避する。

【００９３】

同様に、ロジック１３８が現在の行（列）の第２の部分集合中の２番目のデータ値は零値であると判定するならば、ロジック１３８は、（第１の入力に変換定数Ｃ３がアサートされ、第２の入力に現在の行又は列の零データ値がアサートされている）乗算回路１４４に制御ビットをアサートし、回路１４４に、その第１の入力にアサートされた前の値を更新することなく、その第２の入力にアサートされた値への乗算演算を実行させ、さもなければ第１の入力を更新された定数Ｃ３の値に切り替えるために消費されることになるパワー消費を回避する。

【００９４】

図６の各マルチプレクサ１２９、１４０、１４１、１５０、１５１、１５２及び１５３は、図６のシステムのパイプライン方式演算の現在のサイクル中にその入力にアサートされている値のうち一つをその出力へ通過させるよう動作中に制御される。図６の各回路１５６及び１５７は、図６のシステムの現在の動作サイクル中にその入力にアサートされている値の和をその出力にアサートするよう動作中に制御される加算回路である。図６の回路１５４及び１５５は、図６のシステムによって実行されている特定の変換に適したレイテンシーを有する（フリップフロップとして実施された）遅延回路である。

【００９５】

本発明の幾つかの代替実施形態は、図６のロジック回路１３２、１３４、１３５、１３６、１３７及び１３８が省かれ、省かれた回路の機能が図６のロジック回路１３０を修正した回路によって実行される点を除いて、図６と同じ回路により構成されるか、又は、図６と同じ回路を備える。具体的には、このようなロジック回路１３０を修正した回路は、通常、変換されるべきデータをもつブロック全体（すなわち、変換されるべきデータの４×４形のブロックの４行又は４列の全て）が完全に零値により構成されているか否か、変換されるべきブロックのデータの各行（又は列）が完全に零値により構成されているか否

か、変換されるべきブロックのデータの各行（又は列）の半分ずつが完全に零値により構成されているか否か、及び、変換されるべきブロックのデータの各半行（又は半列）の各データ値が零値であるか否かを判定するように構成され、このような判定に応じて、図６のロジック回路１３０、１３２、１３４、１３５、１３６、１３７及び１３８によってアサートされる上記のタイプの制御信号を（システムの他の要素へ）アサートする。

【００９６】

図６のシステムは、携帯型メディアプレーヤー又は別の携帯型データ処理システム（又は、携帯可能であることが目的とされていないデータ処理システム）、例えば、ビデオ復号機能を有するシステムのサブシステムであってもよい。このようなメディアプレーヤー又はその他の処理システムは、図６に示していない付加的な要素を有することがあり、通常、図６に示していない付加的な要素を含むであろう。

【００９７】

多種多様なフォーマットをもつデータのブロックに対して多種多様な２Ｄ変換を実行するように本発明の種々の実施形態を実施するために、図４を参照して示し説明した特定のステップ、及び、図５及び６を参照して示し説明した特定の構造に関する変形が実施されるであろうことを理解されたい。

【００９８】

本発明の一部の実施形態を本明細書中に示して説明したが、本発明は、請求項によって定められものであり、説明して示した特定の実施形態に限定されないことも理解されたい。

【図面の簡単な説明】

【００９９】

【図１】本発明の実施形態に従って実施される変換エンジンを含むパイプライン方式ビデオ復号システムのブロック図である。

【図２】本発明に従って実施される（ビデオ復号能力を有する）処理システムのブロック図である。

【図３】入力データブロックと、行変換が施されたブロックと、行変換及び列変換が施されたブロックの図である。

【図４】（本発明のシステムのあるクラスの実施形態によって実行される）本発明の方法の実施形態のステップのフローチャートである。

【図５】本発明を具現化する図２の回路７（又は回路１１）の典型的な実施例に利用される変換回路のブロック図である。

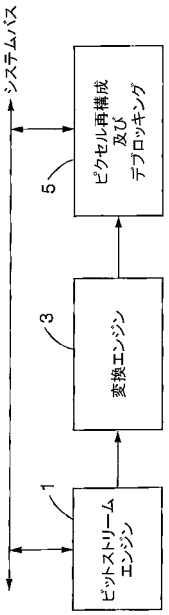
【図６】本発明のパイプライン方式ビデオ復号システムの実施形態の要素のブロック図である。

【符号の説明】

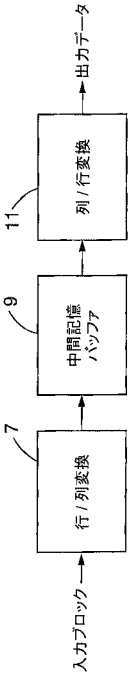
【０１００】

１…ビットストリームエンジン、３…変換エンジン、５…ピクセル再構成及びデブロッキング、７…行／列変換、９…中間記憶バッファ、１１…列／行変換。

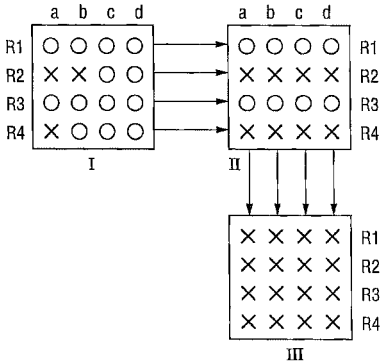
【図１】


システムバス
ビットストリーム
エンジン
1
変換エンジン
3
ピクセル再構成
及び
デブロッキング
5

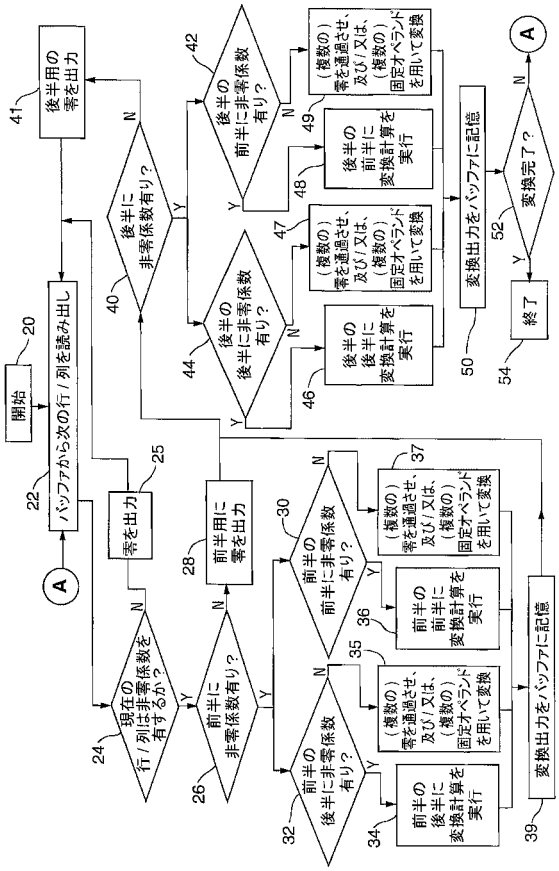
【図２】


入力ブロック
行 / 列変換
7
中間記憶
バッファ
9
列 / 行変換
11
出力データ

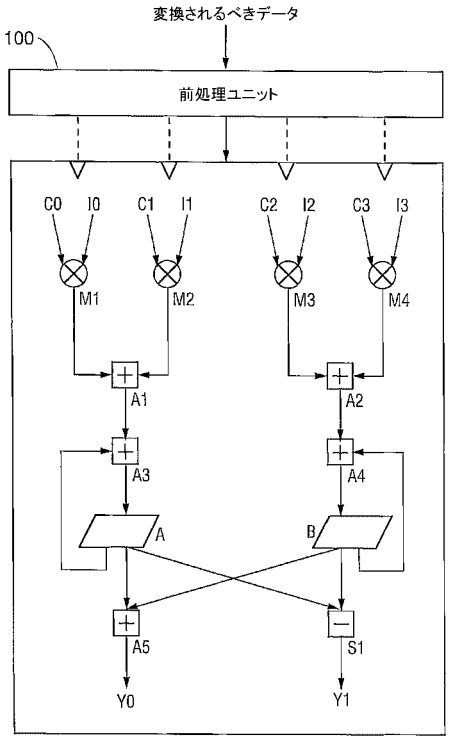
【図３】


a b c d
R1
R2
R3
R4
I
II
III

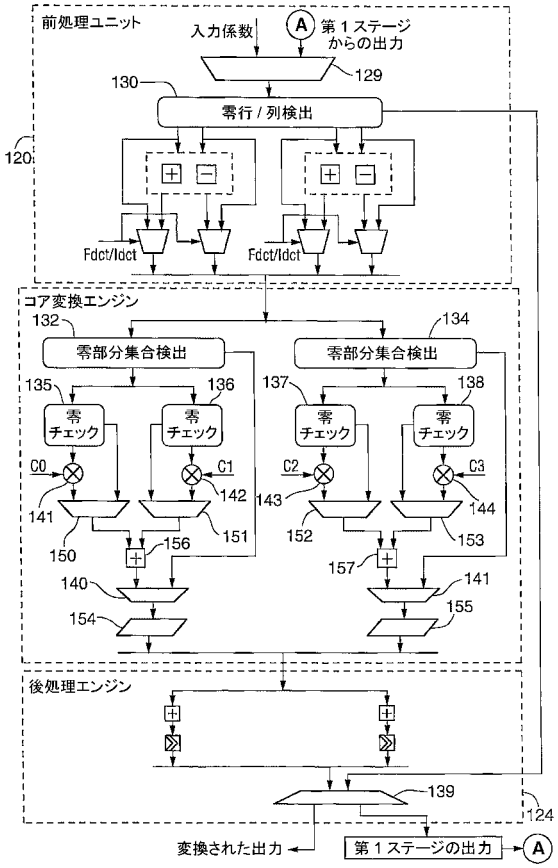
【図４】


開始
20
バッファから次の行 / 列を読み出し
22
現在の行 / 列は非零係数を有するか？
24
零を出力
25
後半用の零を出力
41
後半に非零係数有り？
40
前半用に零を出力
28
前半に非零係数有り？
26
後半の前半に非零係数有り？
42
後半の後半に非零係数有り？
44
前半の前半に非零係数有り？
30
前半の後半に非零係数有り？
32
後半の後半に変換計算を実行
46
後半の前半に変換計算を実行
48
前半の前半に変換計算を実行
36
前半の後半に変換計算を実行
34
（複数の）零を通過させ、及び / 又は、（複数の）固定オペランドを用いて変換
47
（複数の）零を通過させ、及び / 又は、（複数の）固定オペランドを用いて変換
49
（複数の）零を通過させ、及び / 又は、（複数の）固定オペランドを用いて変換
35
（複数の）零を通過させ、及び / 又は、（複数の）固定オペランドを用いて変換
37
変換出力をバッファに記憶
39
変換出力をバッファに記憶
50
変換完了？
52
終了
54
A
A
Y
N

【図５】


変換されるべきデータ
100
前処理ユニット
C0
I0
C1
I1
C2
I2
C3
I3
M1
M2
M3
M4
A1
A2
A3
A4
A
B
A5
S1
Y0
Y1

【図６】


前処理ユニット
入力係数
A
第１ステージからの出力
129
130
零行 / 列検出
120
Fdct/Idct
Fdct/Idct
コア変換エンジン
132
134
零部分集合検出
零部分集合検出
135
136
137
138
零チェック
零チェック
零チェック
零チェック
C0
C1
C2
C3
141
142
143
144
150
151
152
153
156
157
140
154
155
141
後処理エンジン
139
124
変換された出力
第１ステージの出力
A

フロントページの続き

(72)発明者　パルサ　スリラム
アメリカ合衆国,　カリフォルニア州,　ロス　アルトス,　アーモンド　アヴェニュー　９３

審査官　大塚　良平

(56)参考文献　特開平１１－９６１３８（ＪＰ，Ａ）
国際公開第２００５／０３８７１９（WO，Ａ２）
特表２００４－５１１１３９（ＪＰ，Ａ）
特開平１０－３２２６９９（ＪＰ，Ａ）
特開平３－２７０５７１（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１７／１０　－　１７／１８