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

特許第4422299号
(P4422299)

(45)発行日　平成22年2月24日(2010.2.24)　(24)登録日　平成21年12月11日(2009.12.11)

(51)Int.Cl.　F I
G06F 17/10 (2006.01)　G06F 17/10 D
G06F 7/523 (2006.01)　G06F 7/523 Z

請求項の数 8　(全 61 頁)

(21)出願番号　特願2000-174354(P2000-174354)
(22)出願日　平成12年6月9日(2000.6.9)
(65)公開番号　特開2001-56808(P2001-56808A)
(43)公開日　平成13年2月27日(2001.2.27)
審査請求日　平成13年9月26日(2001.9.26)
審判番号　不服2006-3617(P2006-3617/J1)
審判請求日　平成18年2月28日(2006.2.28)

(73)特許権者　500587067
アギア　システムズ　インコーポレーテッド
アメリカ合衆国．１８１０９　ペンシルヴァニア，アレンタウン，アメリカン　パークウェイ　エヌイー　１１１０
(74)代理人　100064447
弁理士　岡部　正夫
(74)代理人　100085176
弁理士　加藤　伸晃
(74)代理人　100096943
弁理士　臼井　伸一
(74)代理人　100101498
弁理士　越智　隆夫

最終頁に続く

(54)【発明の名称】行列計算を行うためのプロセッサ素子のパイプライン処理線形アレー

(57)【特許請求の範囲】
【請求項１】
少なくとも一つのデジタル・プロセッサ素子を含む装置であって、
前記デジタル・プロセッサ素子が、
少なくとも二つの複素数の実数部および虚数部に、別の少なくとも二つの複素数の実数部と虚数部とを掛け、それにより少なくとも１６個の部分的積を形成するよう動作し、また前記部分的積の、各々が実数または虚数を表わす一つまたはそれより多くの加法組合せを形成するよう動作する演算回路と、
少なくとも第一のポートおよび第二のポートを有するレジスタ・ファイルであって、これらポートの各々が、前記レジスタ・ファイルへ、または前記レジスタ・ファイルから、二つの複素数を書き込むことができ、または二つの複素数を読み出すことができ、これらポートが、演算回路に接続されており、この演算回路は、前記複素数を前記ポートに供給し、前記ポートから前記実数または虚数を受け取るようになっているレジスタ・ファイルとを含み、
さらに、前記プロセッサ素子の線形アレーを備え、前記線形アレーにおける前記各プロセッサ素子が、前記アレーの少なくとも一つの他のプロセッサ素子に接続しており、および
前記プロセッサ素子の前記線形アレーが、さらに、一つのヘッド・プロセッサ素子と、複数の通常のプロセッサ素子とを備え、前記ヘッド・プロセッサ素子が、前記各通常のプロセッサ素子の機能のスーパーセットを供給するようになっている装置。

【請求項２】
請求項１に記載の装置において、前記演算回路が、一つの演算の開始から終了までの待ち時間が、新しい演算をスタートすることができる周期よりも長くなるように構成されたパイプライン処理ロジックを含み、それにより、前記プロセッサ素子が、同時に、各演算が異なる開始時間を持つ一つ以上の演算を計算することができる装置。
【請求項３】
請求項１に記載の装置において、前記レジスタ・ファイルの前記第一または第二のポートのうちの少なくとも一つが、読み出しまたは書き込みによりアクセスされる前記二つの複素数の所与の組が、一対の各語線からの同じ語線または別の語線からのものとなるように、制御可能に語線を共にペアにするロジックを含む装置。
【請求項４】
請求項１に記載の装置において、前記演算回路が、さらに、前記部分的積を受け取ることができる入力を持つ少なくとも八つの加算装置を含み、前記加算装置が、前記部分的積の一つまたはそれ以上の加法組合せを形成することができ、前記各加算装置が、一つの実数または虚数に対応し、前記八つの加算装置が一緒になって、複素数の四つの和を形成することができ、また、各加算装置が、新しい入力数と以前の和と合計に対応する新しい和を累算することができるように、その入力にもどるフィードバック接続を持つ装置。
【請求項５】
請求項１に記載の装置において、前記プロセッサ素子が、さらに、非線形関数ゼネレータを含む装置。
【請求項６】
請求項５に記載の装置において、
前記非線形関数ゼネレータが、ある変数を固定の領域内にシフトすることにより、前記変数をスケールことができる正規化回路であって、その正規化ロジックが、前記変数をどれだけシフトしたかを示すべき指数値を出力する正規化回路と、
少なくとも１つの非線形関数に対する補間係数の数値のテーブルを記憶するためのメモリと、
選択した数の前記変数の最上位ビットだけを取り上げ、前記変数のところの前記関数における値が、前記補間係数を使用して補間を計算することにより近似できるように、選択した数の最上位ビットを、前記テーブル内の前記関数の補間係数を発見するために、アドレスとして使用するロジック回路とを備える装置。
【請求項７】
請求項１に記載の装置において、前記プロセッサ素子が、さらに、パイプライン処理遅延ラインであって、前にその入力のところに現われた同じシーケンスのデジタル語を出力することができる機能を有し、前記シーケンスの複数の要素が一度に種々の段における遅延ラインを横切るパイプライン処理遅延ラインと、
前記遅延ラインの出力、または前記遅延ラインの遅延のない入力シーケンスとして、その出力を選択することができるマルチプレクサであって、前記選択が前記プロセッサ素子の入力により制御されるマルチプレクサとを備える装置。
【請求項８】
線形アレーに配置されている複数のデジタル・プロセッサ素子であって、その各々が前記アレーの少なくとも１つの別のプロセッサ素子に接続されているようなデジタル・プロセッサ素子を含む装置において、少なくとも１つの前記デジタル・プロセッサ素子が、
少なくとも二つの複素数の実数部および虚数部に、別の少なくとも二つの複素数の実数部と虚数部とを掛け、それにより少なくとも１６個の部分的積を形成するよう動作し、また前記部分的積の、各々が実数または虚数を表わす一つまたはそれより多くの加法組合せを形成するよう動作する演算回路と、
少なくとも第一のポートおよび第二のポートを有するレジスタ・ファイルであって、これらポートの各々が前記レジスタ・ファイルへ、または前記レジスタ・ファイルから、二つの複素数を書き込むことができ、または二つの複素数を読み出すことができるようにな

っており、これらのポートが、演算回路に接続されており、この演算回路が前記複素数を前記ポートに供給し、前記ポートから前記実数または虚数を受け取るレジスタ・ファイルとを含み、

さらに、前記プロセッサ素子の線形アレーを備え、前記線形アレーにおける前記各プロセッサ素子が、前記アレーの少なくとも一つの他のプロセッサ素子に接続しており、および

前記プロセッサ素子の前記線形アレーが、さらに、一つのヘッド・プロセッサ素子と、複数の通常のプロセッサ素子とを備え、前記ヘッド・プロセッサ素子が、前記各通常のプロセッサ素子の機能のスーパーセットを供給するようになっている装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、概して、プロセッサ素子および技術に関し、特に行列計算を行うためのプロセッサ素子のアレーを備えるパイプライン処理デバイスに関する。

【０００２】

【従来の技術、及び、発明が解決しようとする課題】

＜優先権の主張＞

本出願は、「ＱＲ分解用のパイプライン処理線形プロセッサ・アレー」という名称の、発明者Ａ．Ｊ．グリーンバーガーの１９９９年６月１０日付の米国仮出願６０／１３８，４２３号の優先権を主張する。

【０００３】

技術が進歩するにつれて、メーカーは、一つの集積回路上により多くのワイヤおよびトランジスタを集積することができるようになり、今までは実現不可能であったデジタル信号処理動作を行う新しい機会が到来している。

【０００４】

そのような新規な用途の一例としては、無線通信システムの基地局でのアンテナのアレー用の適応ビーム形成の使用がある。この場合の目標は、所与のセル内により多くの移動局ユーザを収容し、より高速なデータ速度を実現できるようにすることである。この目標は、問題の移動局の方向を指していて、干渉ソースから全然影響を受けない放射パターンを形成するために、複数のアンテナにより受信した複数の信号を時変複合加重により加算することにより達成される。このタイプの問題を最適化するための種々の解決方法は周知であり、複素数の行列およびベクトルによる計算の実行を含む。例えば、１９９６年に、プレンティスホール社発行のＳ．ハイキンの「適応フィルタ理論」第３版参照。所与の時間において、一回の複素数乗算を行うだけではすまず、非常に多くの回数、複素数乗算を行わなければならない。

【０００５】

このタイプのもう一つの新しい用途は、直接シーケンス符号分割多元接続（ＤＳ／ＣＤＭＡ）コード化を使用する無線システム基地局に対するマルチユーザの検出である。これらのアルゴリズムの場合には、例えば、１９９６年１０月号のＩＥＥＥ通信マガジン掲載の、Ｓ．モシャビの「ＤＳ－ＣＤＭＡ通信用のマルチユーザ検出」の１２４ページに記載されているように、相互に干渉している種々の移動局について、リアルタイムで入手した集合的な知識が、上記干渉を解決し、個々の各移動局の検出を改善するために使用される。これらのアルゴリズムも、複素数の行列の集中的な計算を含む。

【０００６】

無線基地局に対する上記の新しい用途は、多くの場合、基本的な行列計算を使用する周知のアルゴリズムの修正したものを使用する。代表的な最も興味のある四つの演算は下記の通りである。

【０００７】

１．ｎ個の未知数を含む一組のｎ個の一次方程式の解。一つの可能なアプローチは、ギブンズ回転により、ＱＲ分解を使用して上記一組の方程式を三角形に変換し、その後で、逆

置換により三角形に変換した一組の方程式を解く方法である。
２．マトリックス反転。一つの可能なアプローチは、ギブンズ回転によりＱＲ分解を使用して一組の方程式を三角形に変換し、その後で、多重逆置換により、逆数について上記一組の三角形に変換した方程式を解く方法である。
３．マトリックスーマトリックス乗算。
４．共分散および相互相関。後でさらに処理が行われる行列およびベクトルを形成するために、入力信号のベクトルに関する上記統計を計算する必要がある。
【０００８】
これらの演算を行うための従来の技術については、以下にさらに詳細に説明する。
一組の一次方程式を解く場合には、行列表示は下記式のようになる。
$$Ax = y \qquad (1)$$
ここで、Ａは複素数値の既知の正方行列であり、ｙは複素数値の既知のベクトルであり、ｘは未知の複素数ベクトルである。上記式を数値解法するには多くの技術が使用される。しかし、これらの技術のうちのいくつかは、数値が不安定であるという欠点がある。式（１）を解くための数値的に安定な技術は、ギブンズ回転によるＱＲ分解またはＱＲ因数分解と呼ばれ、例えば、１９９６年に、ジョンズ・ホプキンス大学出版部が発行した、Ｈ．ゴルブおよびＣ．Ｆ．バン・ローンの「行列計算」第３版に記載されている。このアプローチは、行列Ａを積に因数分解するプロセスを含む。
$$A \equiv QR \qquad (2)$$
【０００９】
ここで、Ｑはユニタリー行列であり、Ｒは右上の三角行列である。「右上の三角」という表現は、主要な対角線の下のＲのすべての構成要素がゼロであることを意味する。それ故、式（１）の両辺には、Q^Hで示すＱの共役転置行列（すなわち、随伴エルミート行列）が掛けられる。
$$Q^H A x = Q^H y \qquad (3)$$
式（２）を式（３）に代入すると、下記式が得られる。
$$Q^H Q R x = Q^H y \qquad (4)$$
$Q^H Q$は、単位行列に等しいので、式（４）は下記式のようになる。
$$Rx = Q^H y \qquad (5)$$
【００１０】
式（５）は「逆置換」と呼ばれる技術を使用して、各構成要素ｘについて簡単に解くことができる式である。「逆置換」という用語は、ほとんどがゼロで、一つだけが未知数である、式（５）の行列の一番下の行が、最初に解かれる反復法を指す。この場合、解は一つ上の行に代入され、その行が解かれる。このプロセスは、式（５）の行列のすべての行の解が得られるまで継続して行われる。
【００１１】
ユニタリー行列Q^Hを発見するための、有効で数値的に安定している方法は、個々のいわゆる「ギブンズ回転」の積としての方法である。この場合、上記各ゼロは、前に置かれたゼロをそのままにしておいて、前の行列の一つの要素を追い出す。例えば、１９５８年３月出版の、「Ｊ．Ｓｏｃ．Ｉｎｄｕｓｔ．Ａｐｐｌ．Ｍａｔｈ」６巻、１号の２６ページ掲載のＷ．ギブンズの「一般行列と三角形に変形する平面ユニタリー回転の計算」を参照してほしい。当業者にとっては、ギブンズ回転が、帰納的最小自乗適応アルゴリズムを行うために、役に立つことは周知である。
【００１２】
図１－図５を参照しながら、ギブンズ回転を使用する小さな行列式の三角形への変換について以下に説明する。下記式（６）は、上記例に対する個々の行列およびベクトル要素を示す式（１）の展開である。
【数１】

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \qquad (6)$$

【００１３】

図１は、対応するギブンズ回転のいくつかの段を示す。図１の六つの段の場合には、５×４のアレーにゼロが連続して並んでいる。各アレーの頭のところの文字は、四つの列が、変換中の行列Ａの要素であること、および５番目の列が変換中のベクトルｙの要素を持っていることを示す。太字で示すゼロは、各段内に現在挿入中のゼロを示す。下記式（７）は、図１の段１での最初の単一変換の方法をより詳細に示す。より詳細に説明すると、一つの要素を強制的にゼロにするために、２×２の複素数のユニタリー行列が、最初の二つの行に適用され、他の要素すべては、新しい数値に変換される。「*」記号は、複素数の共役を示す。

【数２】

$$\begin{pmatrix} c^* & s^* \\ -s & c \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & y_1 \\ a_{21} & a_{22} & a_{23} & a_{24} & y_2 \end{pmatrix} = \begin{pmatrix} a'_{11} & a'_{12} & a'_{13} & a'_{14} & y'_1 \\ 0 & a'_{22} & a'_{23} & a'_{24} & y'_2 \end{pmatrix} \qquad (7)$$

強制的にゼロにされる要素を発見するために、行列を乗算することにより下記式が得られる。

【数３】

$$-s\,a_{11} + c\,a_{21} = 0 \qquad (8)$$

単一制約により、

【数４】

$$|s|^2 + |c|^2 = 1 \qquad (9)$$

下記式（１０）は、これら式を満足する。

【数５】

$$c = \frac{a_{11}}{\sqrt{|a_{11}|^2 + |a_{21}|^2}} \qquad (10)$$

そして、

【数６】

$$s = \frac{a_{21}}{\sqrt{|a_{11}|^2 + |a_{21}|^2}} \qquad (11)$$

【００１４】
ゼロでない変換された要素a_{11}'は、下記式の実数により表わされる。
【数７】

$$a_{1'1} = \frac{|a_{11}|^2 + |a_{21}|^2}{\sqrt{|a_{11}|^2 + |a_{21}|^2}} \quad , \qquad (12)$$

【００１５】
この行列は、二つの列のすべての他の要素を「回転」させる。
【００１６】
すでに説明したように、ギブンズ回転を使用するＱＲ分解は、三角形の形をしている二つの次元の収縮アレーで手早く解くことができる。対応する１９８１年発行の、発明者Ｗ．Ｍ．ジェントルマンおよびＨ．Ｔ．クングの「収縮アレーによるマトリックス三角形への変換」のＳＰＩＥ　Ｖ２９８の１９ページを参照してほしい。
【００１７】
図２－図５は、プロセッサ素子（ＰＥ）の三角形のアレー上の５×４行列式を三角形にする手順の始まりを示す。図２－図５の丸いボックスは、ｃ、ｓおよび上記式（１０）－（１２）に示すように更新した要素を計算する特殊な「ヘッド」要素である。四角いボックスは、ｃおよびｓを使用する二つの要素を回転する「通常の」要素である。これらも、ｃおよびｓを右に移動させる。図２に示す、ギブンズ回転のステップ１においては、第一のｃ１およびｓ１が計算される。図３に示すステップ２においては、第二のｃ２およびｓ２が計算され、要素の一番上の行の第一の四角いボックスは、回転させるためにｃ１およびｓ１を使用する。図４に示すステップ３においては、第三のｃ３およびｓ３が計算され、第一の四角いボックスは回転するために、ｃ２およびｓ２を使用し、第二の四角いボックスは、回転するためにｃ１およびｓ１を使用する。図５に示すステップ４においては、第二の行のヘッド要素は、第四のｃ４およびｓ４の計算を帰納的に開始することができる。第二の行のヘッド要素の上の四角いボックスは、依然として回転を続けている。計算は、三角形のアレーを通して収縮するように進行する。
【００１８】
この問題の場合には、帰納的な手順により、計算の待ち時間を短くする機会があることに留意されたい。所与の列に二つのゼロを並べてから、元の列に続けて他のゼロを並べながら、次の列に右に向かってゼロを並べ始めることができる。使用できる平行度の程度は、処理中の行列が大きくなるにつれて増大する。
【００１９】
計算が収縮的にではなく、平行して行われる上記アーキテクチャの一つの修正アーキテクチャがある。この修正アーキテクチャの場合には、ｃ１およびｓ１が分かった場合には、それらを、すべての四角いボックスのヘッド要素の右に同時に送り、ｃ１およびｓ１のこれらの数値を使用するすべての回転を並列に行うことができる。ヘッド要素でのｃおよびｓの計算は、集中的なものであることに留意されたい。何故なら、絶対値の自乗および逆数の平方根が必要であるからである。

【００２０】
図６は、ギブンズ回転を使用するＱＲ分解用の二つのアーキテクチャで、時間の経過とともに行われるデータの処理方法の全体を示す。並列の場合には、プロセッサ素子のほぼ水平なバンドが一度に動作する。時間の経過につれて、上記バンドは、図に示す三角形のアレーの中を下向きに移動する。収縮の場合には、能動状態のプロセッサ素子のある角度を持ったバンドが、アレーをその下向きの角度を維持しながら、三角形のアレーの中を下向きに移動する。両方のケースの場合、プロセッサ素子の使用が効率的でないことは明らかである。大型の行列のＱＲ分解の場合には、大部分のプロセッサ素子は、任意の所与の時間アイドル状態になっている。例えば、計算のピーク時の、縦１７×横１６の大きさの行列式の場合には、三角形内のプロセッサ素子の価値ある五つまでの行は、一度に能動状態になることができ、待ち時間は短くなり最適状態になる。しかし、平均して、プロセッサ素子のうちの僅かな素子だけを稼働状態に維持することができる。
【００２１】
それ故、上記の二次元の三角形のアレーよりも効率的な、ＱＲ分解および他のタイプの行列計算を行うためのアーキテクチャを発見することが望ましい。一つのアプローチは、一つの一次元のプロセッサ・アレーを考慮の対象にして、シーケンシャルに仮想の二つの次元のアレーの動作を、物理的に一次元のアレー上にマップする方法である。ＩＣＡＳＳＰ９９、Ｇ．ライトボディ、Ｒ．ワルキ、Ｒ．ウッドおよびＪ．マクキャニの「線形ＱＲアーキテクチャの新規なマッピング」；１９９９年の、ＩＥＥＥ国際会議アコウスト、演説、信号処理、１９３３ページに、上記アプローチが記載されている。このアプローチは、二次元の三角アレーからプロセッサ素子を効率よく使用する線形アレーにマッピングするために、再マッピングおよびスケジューリング・スキームを使用する。しかし、このアルゴリズムの全待ち時間は、置き換えた三角アレーの全待ち時間より遥かに長い。また、このアプローチにより、１００％ハードウェアを利用するには、第二の三角形への変換問題が、第一の三角形への変換問題が半分終了した場合にスタートする、二つの独立の三角形への変換問題のステップをインターリーブする必要がある。
【００２２】
１９９２年マサチューセッツ州、ボストンのクルワ・アカデミック出版発行の、Ｊ．Ｈ．モレノおよびＴ．ラングの「収縮タイプのアレーに関する行列計算」の２０６－２１０ページには、他のアプローチが記載されているが、このアプローチは、高い効率でこのタイプの、行列アルゴリズムを実行するために、パイプライン処理プロセッサ素子の線形アレーを使用する。
【００２３】
上記の適応アンテナ・アレー問題を処理するために、他の方法を使用することもできる。例えば、１９８３年の、Ｐｒｏｃ．ＳＰＩＥ　Ｖ．４３１、１０５ページ掲載の、Ｊ．Ｇ．マックハータの「収縮性アレーを使用する帰納的最小自乗最小化」に記載の「帰納的最小自乗」（ＲＬＳ）と呼ばれる計算がもっと簡単なアルゴリズムは、適応アンテナ問題に適用することができる。このアプローチの場合には、最適な結果には及ばない結果しか得られないが、計算が簡単になるという利点がある。ＲＬＳアルゴリズムは、上記逆置換を使用しなくてもすむ。二次元三角収縮アレー上で、迅速にＲＳＬアルゴリズムを実行する方法は周知である。この二つの二次元アレーに基づく、ハードウェアによる解決方法については、例えば、１９９８年の放送通信に関するＩＥＥＥ１９９８年国際チューリッヒ・セミナーの２９ページ掲載の、Ｂ．ハラーの「移動通信における、ＲＬＳをベースとする、時間参照ビーム形成についてのアルゴリズムおよびＶＬＳＩアーキテクチャ」が記載している。
【００２４】
上記の従来の逆置換プロセスについて、以下にさらに詳細に説明する。すでに説明したように、ＱＲ分解が終了した後で、三角形に変換された一組の式は、式（５）の形をしていて、この式はこの実施形態のために展開される。
【数８】

$$\begin{aligned} r_{11}x_1 + r_{12}x_2 + r_{13}x_3 + r_{14}x_4 &= y_1 \\ r_{22}x_2 + r_{23}x_3 + r_{24}x_4 &= y_2 \\ r_{33}x_3 + r_{34}x_4 &= y_3 \\ r_{44}x_4 &= y_4 \end{aligned} \qquad (13)$$

【００２５】
上記のギブンズの手順の後で、この例においては、r_{11}、r_{22}およびr_{33}である最後のものを除いて、Ｒ行列の対角線方向のすべての要素は実数であり、またこの例の場合はr_{44}である最後の対角線方向の最後の要素は複素数である。逆置換プロセスを使用すれば、この一組の式（１３）を解くことができる。より詳細に説明すると、一つの未知数x_4しか含んでいない四つの式の最後の式を最初に解くことができる。例えば、y_4をr_{44}で割ることにより、x_4を得ることができる。別の方法としては、x_4を発見するために、逆数、$1/r_{44}$を計算し、その後で、y_4を掛けることもできる。その後で、x_4は四つの式のうちの第三の式に代入される。r_{33}の逆数を計算する事ができる。その後で、x_3を、$(1/r_{33})(y_3 - r_{34}z_4)$の形で、発見することができる。その後で、このプロセスは、スペースの未知数が発見されるまで、式を通して、反対方向に継続して実行される。

【００２６】
行列の逆数は、上記の一連の逆置換を含む手順により、発見することができる。行列Ａの逆数A^{-1}を発見したい場合を考えてみよう。定義により、下記式のようになる。

$$AA^{-1} = I \qquad (14)$$

【００２７】
ここで、Ｉは単位行列である。上記説明と同様に、下記式のようなユニタリー行列Ｑが存在する。

$$A = QR \qquad (15)$$

【００２８】
ここで、Ｒは右の三角形である。これを式（１４）に代入すると、下記式が得られる。

$$QAR^{-1} = I \qquad (16)$$

【００２９】
その後で、先に進んで、一連のギブンズ回転の結果としての、Ｑ、Q^Hの随伴エルミート行列を発見することができる。式１６の両辺にQ^Hを掛けると、下記式が得られる。

$$RA^{-1} = Q^H \qquad (17)$$

【００３０】
この時点で、ＲおよびQ^Hは、両方とも発見されるが、A^{-1}は未知数であることを思いだしてほしい。式（１７）の左辺の各列は、右辺の対応する列と等しくなけてはならないので、式（１７）は一連の式に分割することができる。

$$Rx_j = y_j \qquad (18)$$

【００３１】
ここで、x_jはA^{-1}の列であり、y_jはQ^Hの対応する列である。ｊの異なる数値に対する一連の式（１８）の各式は、式（５）の逆置換問題に似ている。これらすべての逆置換問題は、共通の右の三角形の行列Ｒを共有している。それ故、行列の逆数は、ギブンズ回転によりＱＲ分解を行った後で、多くの逆置換を行うことにより発見することができる。

【００３２】
信号のアレーの高度の処理の際に行われるもう一つの共通な演算は、一つの行列に他の行列を掛けることである。この場合、各行列の要素は複素数値を持つ。行列ＡおよびＢの乗算による行列Ｃの構成を考えてみよう。この例の場合には、三つの各行列は、複素数要素を含む三つの４×４行列である。式（１９）は、この例を展開して、行列要素を示す。

【数９】

$$\begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{pmatrix} \equiv \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{pmatrix} \quad (19)$$

【００３３】

Ｃの１６の各複素数要素は、Ａの一つの行からの要素、およびＢの一つの列からの要素の四回の複素数乗算の結果得られたものである。

【数１０】

$$c_{23} = a_{21}b_{13} + a_{22}b_{23} + a_{23}b_{33} + a_{24}b_{43} \quad (20)$$

【００３４】

それ故、必要な複素数乗算の回数は、全部で４×４×４＝６４回である。式（１９）のマトリックス－マトリックス乗算は、多くの異なるハードウェア・アーキテクチャにより、多くの方法で計算することができる。通常のプログラム可能なデジタル・コンピュータの場合には、ある実数に他の実数を掛けるためのハードウェアを備えているが、通常、それ以上のハードウェアの支援はない。プログラムは、通常、四つの実数乗算の合計として、各複素数乗算を行う。

【数１１】

$$a_{21}b_{13} = (a_{21\text{real}}b_{13\text{real}} - a_{21\text{imaginary}}b_{13\text{imaginary}}) + i\,(a_{21\text{real}}b_{13\text{imaginary}} + a_{21\text{imaginary}}b_{13\text{real}}) \quad (21)$$

【００３５】

現在の多くのデジタル・コンピュータの場合には、乗算および累算用の演算ハードウェアが、前の項がその乗算および累積を終了する前に、新しい項が乗算を始めることができるように、すでにパイプライン処理を行っていて、それにより、式（２０）および（２１）に示すように、計算の処理能力が改善される。

【００３６】

現在のベクトル・スーパコンピュータの場合には、プログラム制御によるハードウェアが、メモリから（この例の場合には、a_{21}、a_{22}である）オペランドを、ベクトル・レジスタにロードするのを助け、そのため、上記オペランドを、パイプライン処理乗算ハードウェアの入力に迅速に送ることができる。

【００３７】

最近のプログラム可能なデジタル信号プロセッサ（ＤＳＰ）のあるものは、一つ以上の乗算装置を含んでいる。例えば、ペンシルバニア州アレンタウンのマイクロエレクトロニック・グループの、ルーセント・テクノロジーズ社のスターコアＳＣ１４０ ＤＳＰは、それぞれが、二つの実数を掛け合わせ、各サイクル毎にその結果を累算することができる、四つの演算ユニットを含む。正しくプログラムされている場合には、ＳＣ１４０は、式（２０）の演算を行っている際に、サイクル毎にピーク速度で、一つの複素数乗算および累算を行うことができる。

【００３８】

マトリックス－マトリックス乗算のような演算を行うための、「行列計算用の収縮アレー装置」という名称の、Ｈ．Ｔ．クングおよびＣ．Ｅ．レイザソンの、１９８５年の米国特許第４，４９３，０４８号は、プロセッサ素子の二次元収縮アレーを提案している。この場合、各プロセッサ素子は、二つの数の乗算、およびその後のサイクル毎の累算からなる

、「ステップ」演算を行うことができる。これらのアレーにおいては、ＡおよびＢの要素は、特定のパターンに従って、アレー周辺のプロセッサ素子に送られる。Ｃの要素は周辺のプロセッサ素子からのものである。例えば、１９８８年プレンテイス・ホール社発行のＳ．Ｙ．クングの「ＶＬＳＩアレー・プロセッサ」の２１３ページには、他の二次元アレーの例が記載されている。このようなアレーが、複素数の乗算および累算を行うことができる各プロセッサ素子と一緒に組み立てられた場合には、このアレーは、非常に高性能を持つことができる。しかし、このようなアレーは、非常に高価であり、データを入力したり、取り出したりするのが難しい。

【００３９】

上記適応アンテナ用途の場合には、多数の空間的に分離しているアンテナは、すべて同じ一組の必要な移動局ユーザおよび不必要な干渉源から信号を受信する。上記アンテナから受信した信号に、特定の組の複素数加重値を加えることにより、適応アンテナ・アレーは、一つの移動局の方向に高い利得を持ち（指向性）、干渉源の方向の利得をゼロにすることができる。ｘをアンテナ・アレーから受信したベクトルと仮定しよう。必要な信号y_dを検出するための最適加重W_{opt}のベクトルは、ウイナ・ホプト式により入手できるが、このベクトルは、静止している一組の移動局および干渉源にとっては理想的なものであり、また、あまり急速に移動しない、静止していない組にとっては実際的なものである。

$$R_x W_{opt} = r_{xd} \qquad (22)$$

【００４０】

この場合、

$$R_x = E\{x x^H\} \qquad (23)$$

【００４１】

上記式は、複素数共分散行列である。

$$r_{xd} = E\{x y^*_d\} \qquad (24)$$

【００４２】

上記式は、複素数相互相関ベクトルである。Ｅ｛｝演算は、時間経過中の統計的平均をとることを示す。実際には、上記平均は、最近受信したデータに対して加重をより重くし、古いデータに対して加重を軽くすることができる。ある実際のシステムの場合には、送信機は、周期的に、既知の信号を送信するが、システムをトレーニングし、W_{opt}を発見するために、y_dに対して使用されるのはこの既知の信号である。

【００４３】

それぞれ、式（２３）の共分散行列、および式（２４）の相互相関ベクトルを、かなり頻繁に計算しなければならない。これらの両方の手順は、上記マトリックス－マトリックス乗算と比較すると、計算は簡単であるが、従来の技術の場合には、必要な計算効率をあげることはできない。

【００４４】

上記のことから考えて、ＱＲ分解を行い、その後で、逆置換を行うことにより、一組の式の解のような演算、およびマトリックス反転、マトリックス－マトリックス乗算、および共分散および相互相関のような、その他の行列関連の演算を実行する際に使用するのに適している、もっと効率的なプロセッサ素子アーキテクチャの開発が待望されているのは明らかである。

【００４５】

【課題を解決するための手段】

本発明は、種々の行列演算を行うのに特に効率の高いパイプライン処理を行うプログラム可能なプロセッサ素子（ＰＥ）、および上記プロセッサ素子の線形アレー用のアーキテクチャを提供する。本発明によれば、各プロセッサ素子は、（ｉ）少なくとも二つの複素数の実数部と虚数部に、少なくとも他の二つの複素数の実数部および虚数部を掛け合わせることができ、それにより、少なくとも１６の部分的積を形成することができ、（ｉｉ）各加法の組合せが実数または虚数を表わす、上記部分的積の一つまたはそれ以上の加法の組合せを形成することができる演算回路を含む。各プロセッサ素子内のレジスタ・ファイルは、第一のポートと第二のポートとを含み、各第一のポートおよび第二のポートは、二つ

の複合語を読み取ることができ、またはレジスタ・ファイルへまたはレジスタ・ファイルから二つの複合語を書き込むことができる。上記ポートは、そこに複合語を供給し、そこから実数または虚数を受け取るための演算回路に接続している。

【００４６】

他の観点から見ると、本発明の行列計算を行うのに適しているプロセッサ素子の線形アレーは、ヘッド・プロセッサ素子、一組の通常のプロセッサ素子を含み、上記ヘッド・プロセッサ素子は、通常のプロセッサ素子の機能的スーパーセットであり、アレー内の最も近い隣接プロセッサ素子と相互接続していて、隣接していない通常のプロセッサ素子から、ヘッド・プロセッサ素子へのフィードバック経路を持つ。ヘッド・プロセッサ素子は、さらに、非線形関数ゼネレータを含む。各プロセッサ素子は、終了する算術演算の待ち時間を、それにより新しい演算をスタートする周期の倍数になるようにパイプライン処理を行う。アレーを制御するために、超長命令語（ＶＬＩＷ）プログラム、またはその他のタイプのプログラムを使用することができる。このアレーは、例えば、一組の一次方程式、マトリックス反転、マトリックス－マトリックス乗算、および共分散および相互相関の計算のような、複雑な行列演算を行う際に特に高い効率を示す。

【００４７】

都合のよいことに、本発明を使用した場合には、二次元三角アレーを使用してすでに実行した行列計算を、処理素子のパイプライン処理線形アレーを使用して、効率的に実行することができる。それ故、本発明を使用すると、パイプライン処理のために、待ち時間が若干長くなるだけで、かなり簡単なハードウェアにより、行列計算を行うことができる。

【００４８】

本発明の上記および他の機能および利点は、添付の図面を参照し、下記の説明を読めば理解することができるだろう。

【００４９】

【発明の実施の形態】

本発明は、例えば、ギブンズ回転によるＱＲ分解、逆置換による一組の三角形の式の解法、マトリックス－マトリックス乗算、および共分散および相互相関の形成を含む、種々の行列演算を行う際に、高い効率を示す制御を備えている、パイプライン処理を行うプログラム可能なプロセッサ素子（ＰＥ）、および上記プロセッサ素子の線形アレー用のアーキテクチャを提供する。

＜プロセッサ素子アーキテクチャ＞

【００５０】

図７は、本発明のプロセッサ素子（ＰＥ）１００の例示としての実施形態である。すでに説明したように、プロセッサ素子のパイプライン処理アレーは、通常、通常のプロセッサ素子とヘッド・プロセッサ素子の両方を含む。図７のプロセッサ素子１００は、本発明のヘッド・プロセッサ素子の一例と見なすことができる。このプロセッサ素子は、対応する通常の素子の機能のスーパーセットを持つ。この例示としての実施形態の場合には、ヘッド・プロセッサ素子は、図に示すすべての素子を含むが、一方、対応する通常のプロセッサ素子は、一点鎖線で示す非線形関数ゼネレータ１０２以外のすべての素子を含む。

【００５１】

図７のプロセッサ素子１００は５段のパイプラインで構成されているが、同じアーキテクチャを、他の長さのパイプラインを使用して実行することもできる。

【００５２】

プロセッサ素子１００の素子１０４は、以下に説明する方法で、二つの複素数と二つの複素数との間の乗算／累算を行う、複素数乗算装置部分である。上記素子１０４は、また、読出し専用メモリ（ＲＯＭ）テーブル駆動補間エンジンとして実行することができる、非線形関数ゼネレータ１０２を含む。例示としての実施形態で、ゼネレータ１０２にょり発生させることができる、二つの非線形関数は、逆平方根および逆数である。非線形関数ゼネレータ１０２については、以下にさらに詳細に説明する。

【００５３】

図７のプロセッサ素子１００は、また、三つのポートを持つレジスタ・ファイル１０６を含む。上記レジスタ・ファイル１０６の、これらポートのうちの二つ、すなわち、ポートｒ１およびｒ２は、計算中の一時的な記憶のソースおよび宛先として使用される。このレジスタ・ファイル１０６の第三のポートｒ３は、演算対象のデータの数値を書き込み、その結果を読み出すことができる、ホスト・プロセッサ専用のポートである。上記ホスト・プロセッサは、この第三ポートを、そのメモリ・アドレスおよびデータ・バスに直接インターフェースする、メモリとして使用することができる。もう一つの例としては、第三のポートは、デジタル・プロセッサ設計分野の当業者であれば周知の、直接メモリ・アクセス（ＤＭＡ）コントローラを通して、インターフェースすることもできる。ＤＭＡコントローラおよび関連するインターフェース・ロジックを使用すると、レジスタ・ファイルとホスト・プロセッサとの間のメモリ転送が容易になり、必要に応じて、語を再度並べ変えることができる。

【００５４】

複素数乗算装置部分１０４に左には、それぞれを、ｉｌｃ０（１）およびｉｌｓ０（１）で示す、一組の入力レジスタ１０８－１および１０８－２が位置する。これらレジスタは、横の入力ｉｌｃおよびｉｌｓからシフトされたか、またはレジスタ・ファイル１０６のポート２からロードされたか、または入力ｉｔａおよびｉｔｂからロードされた数値を保持することができる。これら二つのレジスタは、また、ある種の処理アルゴリズムの場合に必要な収縮パイプライン処理演算を維持するために、遅延素子１１１を通して遅延出力信号を供給する、一点鎖線で示す遅延ライン１１０を供給する。

【００５５】

図７の右側においては、出力右信号ｏｒｃおよびｏｒｓを、それぞれ、マルチプレクサ１１２－１および１１２－２により選択した、三つの可能な信号のうちの一つにより駆動することができる。より詳細に説明すると、ｏｒｃ出力信号は、ｉｌｃ０（１）、ｉｌｃ０（５）またはｏｒｔｃ（４）のいずれかであってもよいし、ｏｒｓ出力信号は、ｉｌｓ０（１）、ｉｌｓ０（５）またはｏｒｔｓ（４）のいずれかであってもよい。信号ｉｌｃ０（１）およびｉｌｓ０（１）は、マトリックス－マトリックス乗算を行うために使用するような、１サイクル収縮移送アルゴリズム用に使用される。信号ｉｌｃ０（５）およびｉｌｓ０（５）は、遅延ライン１１０の出力であり、ＱＲ分解を行うために使用されるアルゴリズムにような、多重サイクル収縮移動アルゴリズム用に使用される。信号ｏｒｔｃ（４）およびｏｒｔｓ（４）は、ゼネレータ１０２が発生する非線形関数を表わす。

【００５６】

プロセッサ素子１００は、また、保持レジスタｇ１、ｈ１、ｇ２およびｈ２、および種々の信号を、複素数乗算装置部分１０４のＹ１およびＹ２被乗数に適用することができる、多数のマルチプレクサ１１４－１、１１４－２、１１４－３、１１４－４、１１６－１および１１６－２を含む。

【００５７】

上記式（７）のところですでに説明したように、基本的なギブンズ回転は、下記式のような積の組合せにより、二つの複素数行列要素を回転するために、二つの複素数量ｃおよびｓを使用する。

【数１２】

$$a_{1'2} = c^* a_{12} + s^* a_{22}$$
$$a_{2'2} = -s\, a_{12} + c\, a_{22} \qquad (25)$$

【００５８】

それ故、プロセッサ素子１００が行う演算のうちの一つは、式（２５）のようなギブンズ

回転である。式（２５）中の演算は、通常、従来のデジタル・コンピュータまたはＤＳＰによる、１６の実数の乗算および多数の加算を必要とする。その代わりに、本発明のプロセッサ素子１００は、もっと効率的にギブンズ回転を行うために、二つの複素数と二つの複素数を使用する乗算装置を使用する。引用によって本明細書の記載に援用した、「複素数乗算装置回路」という名称の、発明者Ａ．グリーンバーガーの、１９９９年６月１４日付の米国特許出願第０９／３３３，０７１号が、この乗算装置について詳細に記載している。

【００５９】

さらに、複素数乗算装置部分１０４は、最後の加算装置段の入力に戻される出力からのフィードバックを含む。より詳細に説明すると、ｚ１（５）－Ｚ４（５）で示す複素数出力は、図に示すように、マルチプレクサ１１８－１および１１８－２を通して、レジスタ・ファイル１０６に書き込むことができる。ｚ１（５）出力は、可能な被乗数入力としてフィードバックされる。ｚ２（４）およびｚ４（４）出力は、それぞれ、マルチプレクサ１２０－１および１２０－２およびレジスタ１２２－１および１２２－２を通して、ｏｌａおよびｏｌｂとして、近隣のプロセッサ素子の左に送ることができる。

【００６０】

レジスタ・ファイル１０６のポートｒ１は、入力ｉｒａおよびｉｒｂを通して、プロセッサ素子からその右に送られる、最高二つの複素数語により、書き込むことができる。ｉｒａおよびｉｒｂ入力の数値は、ｏｌａおよびｏｌｂを通って左に進む。入力ｉｔａおよびｉｔｂにより、データは、近隣のプロセッサ素子を通らないで、それぞれ、マルチプレクサ１２４－１および１２４－２を通って、直接プロセッサ素子１００に入力することができる。

【００６１】

複素数乗算装置部分１０４の入力ａ２およびａ４は、それぞれ、マルチプレクサ１２６－１および１２６－２の出力により駆動される。乗算装置部分１０４のｚｌ（４）－ｚ４（４）出力は、図に示すように、レジスタ１２７－１、１２７－２、１２７－３および１２７－４に送られる。

【００６２】

図８は、図７の複素数乗算装置部分１０４で実行することができる、２複素数×２複素数用の乗算装置１５０のブロック・アーキテクチャである。図８の左側には、乗算処理される各複素数Ｘ１およびＸ２を保持する、複数の組の入力レジスタ１５２－１および１５２－２が位置する。本明細書においては、複素数Ｘ１およびＸ２を「乗数」と呼ぶ。図８の頂部には、乗算処理される各複素数Ｙ１およびＹ２を保持する、複数の組の入力レジスタ１５４－１および１５４－２が位置する。複素数Ｙ１およびＹ２は、「被乗数」と呼ばれる。

【００６３】

乗算装置１５０は、再コード化要素１５６－１および１５６－２を使用して、左側で多重ビット再コード化を行い、また、被乗数多重要素１５８－１および１５８－２を使用して、頂部で被乗数多重形成を行う。左側の四つの実数と頂部の四つの実数の１６の部分的積は、中央のボックス１６０内で作られる。中央のボックス内の各部分的積ボックス１６２－１、１６２－２、．．．、１６２－１６は、マルチプレクサおよび加算装置を含む。

【００６４】

図８の底部には、最高四つまでの複素数の結果、ｚ１－ｚ４の実数部および虚数部を形成するための、最後の八つの加算装置を含む、もう一つの一組の加算装置回路１６４が位置する。図８のｚ１（４）の「（４）」は、計算の際のパイプラインの段を示す。図８にはハッキリと示していないが、適当な部分的積の出力は、最後の八つの加算装置の入力に送られる。

【００６５】

また、図８に示すように、ｚ２－４（４）およびｚ１（４）マルチプレクサへの入力から、前段のパイプライン段に戻るフィードバック経路が設置されていて、そのため、累算を

行うことができる。前の段のパイプライン段のところで加算することができる、別々の入力、ａ２およびａ４となるの可能性を持つ、ｚ２（４）およびｚ４（４）用の最後の加算装置も設置されている。図７のプロセッサ素子１００においては、マルチプレクサ１２６－１および１２６－２により、これら二つの別々の入力を、ｉｌｃ、ｒ２およびａ２およびｉｌｓ用のｉｒａ、ｒ２およびａ４用のｉｒｂから、出力するようにすることができる。

【００６６】

図７の非線形関数ゼネレータ１０２について、さらに詳細に説明する。すでに説明したように、行列計算のために必要な関数は、通常、逆平方根および逆数である。これら関数を計算するには、整級数展開、反復帰納的フォーミュラおよび補間を含む、多くの周知の数値法がある。１９６８年に、プレンティス・ホール社発行の、Ｃ．Ｔ．ファイクの「数学的関数のコンピュータ評価」を参照してほしい。集積回路技術が進歩したために、チップをベースとするメモリ上に、大きな「参照」テーブルを記憶することができるようになったために、補間法がもっと実用的なものになった。これらのテーブルは、通常、点の格子上の関数に対するデータを含み、そのため、必要な点の関数を補間法により計算することができきる。この補間技術の一つの利点は、広い範囲にわたって効果的に機能する任意の関数に適用できることである。そのため、異なる機能を発生するために、同じまたは類似のハードウェア・エンジンを使用することができる。

【００６７】

図９は、非線形関数ゼネレータ１０２の例示としての実施形態である。ゼネレータ１０２は、式（１０）、（１１）および（１２）が示すギブンズ回転を計算するために使用される。図９の場合には、陰をつけた要素は、複素数を処理し、陰をつけていない要素は実数だけを処理する。ゼネレータ１０２は、対応する実数部および虚数部の平方の合計として、それぞれ、複素数Ｙ１およびＹ２の絶対値の平方を生成する、要素１７０－１および１７０－２を含む。ギブンズ回転の場合には、合計要素１７２が、平方計算要素１７０－１および１７０－２の出力の合計、$|a_{11}|^2+|a_{21}|^2$を発生する。この合計は、逆平方根ゼネレータ１７４に入力として送られる。入力合計の２進小数点が位置する場所を示すために、数ｑも、平方根ゼネレータ１７４に入力とし送られる。入力合計は、２進小数点の右のすべてのビットが、その数の小数部分、すなわち、その数の１より小さい部分を示す、「固定小数点」フォーマットの２進数である。

【００６８】

図１０は、図９の逆平方根ゼネレータ１７４の詳細図である。この実施形態の場合には、平方根ゼネレータ１７４は、二次補間装置１８０として実行される。正規化要素１８２は、その数を、１より大きいかまたは等しく、４より小さい領域内に正規化する。この正規化機能を実行するのに適している技術については、図２５、２６および２７を参照しながら、詳細に説明する。

【００６９】

正規化要素１８２の出力は、領域１／２－１内の符号がついていない正規化した数値であり、および正規化した数値を入手するために、入力のシフトしなければならないビット数を示す、符号がついている指数である。図１０に示すように、正規化した数値の多数の最上位のビットは「オフセット」と呼ばれ、予め計算したテーブルを含むメモリ１８４へのアドレスとして使用される。メモリ１８４の出力は、補間の際に使用される三つの係数ａ、ｂおよびｃである。オフセットの右側の正規化した数値の最下位のビットは、「ｄｘ」と呼ばれる。要素１８６が発生したｄｘおよびその平方は二次補間装置１８０に送られ、この補間装置は、$dx^2+bdx+c$の形の出力仮数を計算する。

【００７０】

結果として得られた仮数の精度は、メモリ１８４の大きさの関数である。オフセットのためにもっと多くのビットを使用する場合には、もっと大きなメモリが必要になり、その結果、仮数の精度は向上する。テーブル１は、異なるテーブル・サイズで達成可能な精度を示す。エラーは全領域にわたる、最悪ミニマックス相対エラーである。例えば、（１４＋

１７＋２４）ビットの９６語の容量を持つメモリは、３．１×１０$^{-7}$の最悪の場合のエラーを生み出すことができる。

【００７１】

図１０の同じハードウェアの大部分は、逆数計算に再使用することがある。もちろん、逆数用のテーブルを記憶するには、別のメモリが必要になる。

【表１】

テーブル・サイズ	y1内のエラー	a の精度	b の精度	c の精度
48	2.4E-6	13	16	21
96	3.1E-7	14	17	24
192	3.9E-8	16	19	27
384	4.9E-9	15	22	30
768	6.2E-10	17	25	33
1536	7.6E-11	18	26	37
3072	1.0E-11	19	28	39

表1-逆平方根の二次補間

【００７２】

再び図９について説明すると、非線形関数ゼネレータ１０２は、さらに、別の組の乗数１８８－１、１８８－２および１８８－３を含む。これら乗数は、式（１０）－（１２）で必要な三つの数値を生成する。固定小数点乗数の後で、対応するたる形シフタ１９０－１、１９０－２および１９０－３は、結果を非正規化（denormalize）するために、上記指数を使用する。一番下の右の「ｚ１（４）へ」のところの出力は、図８の一番下に「非線形から」と表示されているマルチプレクサへ送られる。

【００７３】

図９の非線形関数ゼネレータ１０２は、一段の回路の形をしてて、数値はＹ１およびＹ２入力に入力され、保持され、最終的には三つの結果が表示される。しかし、例示としての実施形態の場合には、上記回路は、好適には、パイプライン処理され、そのため、他のいくつかのものを計算しながら、新しい逆平方根の計算を始めることができるものであることが好ましい。

【００７４】

図１１は、図９の非線形関数ゼネレータ１０２の、上記パイプライン処理の実施形態の一例を示す。この実施形態の場合には、ゼネレータ１０２は、垂直方向の矢印で示す場所にレジスタを設置することにより、または、例えば、１９９６年の、ＩＥＥＥ　Ｊ半導体回

路３１巻、３号の３７６ページ掲載の、Ｍ．ハシャミおよびＢ．Ａ．ウーリイの「２５０ＭＨｚの傾斜クロックパイプライン処理接続バッファ」に記載されている技術のような、当業者であれば周知のもっと進んだ技術を使用してパイプライン処理される。各パイプライン段の待ち時間はφである。全計算を通しての全待ち時間は５φであるけれとも、各φ毎に新しい計算を始めることができる。それ故、計算のピーク時には、ｃ、ｓの五つの重畳している計算を行うことができ、式（１０）－（１２）の更新した数値が同時に計算されるが、φの倍数により時間的に分離している。

【００７５】

レジスタ・ファイル１０６のアーキテクチャについて、さらに詳細に説明する。必要な行列演算のあるものを行うためには、サイクル毎に四つの複素数語を処理する速度で、レジスタ・ファイル１０６に記憶している数値にアクセスしなければならない。例えば、二つの複素数乗数、および二つの複素数被乗数を、図８の乗算装置１５０のＸおよびＹ入力に供給しなければならない。図７のプロセッサ素子アーキテクチャの場合には、これら複素数語のうちの二つは、レジスタ・ファイル１０６のポートｒ１から供給され、これら複素数語のうちの他の二つは、レジスタ・ファイル１０６のポートｒ２から供給される。それ故、これらポートのうちの所与の一つは、同時に二つの複素数語を取り出すように設計されている。

【００７６】

図１２は、所与のプロセッサ素子の、レジスタ・ファイル１０６内の語１９０のアレーである。図１２は、また、三つのポートのうちの一つを解読するための語線も示す。この実施形態のレジスタ・ファイル１０６は、各語が、例えば、１６ビットのような、所与の数を保持する、四つのリアルな語の幅を持つように構成される。語は、簡単に図示してあるように、０から３で示す四つの列に常駐する。一つのポート用の語線デコーダ２００は、行アドレスおよび制御情報を取入れ、その左側の一本の語線、およびその右側の一本の語線を能動状態にすることができる。上記目的のために、水平方向の一点鎖線で示すように、二つの行はペアになっている。デコーダ２００は、同じ二重の行の左側の一本の語線および右側の一本の語線を駆動しさえすればよい。列０および１、または列２および３の複素数語のビットの、物理的レイアウトは図示する必要はないが、必要な任意の順列に並べられている左から右へのある順序を持つことができる。

【００７７】

所与の複素数語は、二つの隣接する実数語を含み、一方の語は実数部を表わし、他方の語は虚数部を表わす。この例の場合には、図１２の列０の各語は実数であり、列１の同じ行の各隣接する語は、関連する虚数部である。同様に、列２は実数部を保持し、列３は他の複素数語の虚数部を保持する。

【００７８】

テーブル２は、８×８行列の要素の行列図である。各要素は複素数である。行列の二つの隣接する列のすべての要素は、一つのプロセッサ素子のレジスタ・ファイル１０６に記憶される。陰をつけた部分は、第一のプロセッサ素子の、レジスタ・ファイルに記憶される、第一の二つの列を示す。

【表２】

A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{16}	A_{17}	A_{18}
A_{21}	A_{22}	A_{23}	A_{24}	A_{25}	A_{26}	A_{27}	A_{28}
A_{31}	A_{32}	A_{33}	A_{34}	A_{35}	A_{36}	A_{37}	A_{38}
A_{41}	A_{42}	A_{43}	A_{44}	A_{45}	A_{46}	A_{47}	A_{48}
A_{51}	A_{52}	A_{53}	A_{54}	A_{55}	A_{56}	A_{57}	A_{58}
A_{61}	A_{62}	A_{63}	A_{64}	A_{65}	A_{66}	A_{67}	A_{68}
A_{71}	A_{72}	A_{73}	A_{74}	A_{75}	A_{76}	A_{77}	A_{78}
A_{81}	A_{82}	A_{83}	A_{84}	A_{85}	A_{86}	A_{87}	A_{88}

表2-行列の内容

【００７９】
テーブル３は、テーブル２の陰をつけた要素が、図１２のレジスタ・ファイル１０６の行および列にどのように記憶されるかを示す。テーブル３においては、上記二つの「行列図」の列が、どのようにしてレジスタ・ファイルに、靴紐のパターンで記憶されるかを示すために、テーブル２の第一の列からの要素だけに陰がつけてある。
【表３】

	0-1	2-3
0	A_{11}	A_{12}
1	A_{22}	A_{21}
2	A_{31}	A_{32}
3	A_{42}	A_{41}
4	A_{51}	A_{52}
5	A_{62}	A_{61}
6	A_{71}	A_{72}
7	A_{82}	A_{81}

表3−二つの複素数コラムの記憶のレジスタ･ファイルの内容

【００８０】

テーブル３の記憶スキームの利点は、このスキームを使用した場合には、各ポートが、テーブル２の「行列図」の同じ行または同じ列上の、隣接する二つの複素数語にアクセスすることができることである。以下に説明するように、この機能は、プロセッサ素子のアレー上での、効率的なマトリックス－マトリックス乗算の際に使用される。テーブル４は、同じ結果、すなわち、同じ「行列図」の行または列上のとちらかの隣接する、二つの複素数語にアクセスすることができる、レジスタ・ファイル１０６の第一の二つの行用の別の記憶スキームである。

【表４】

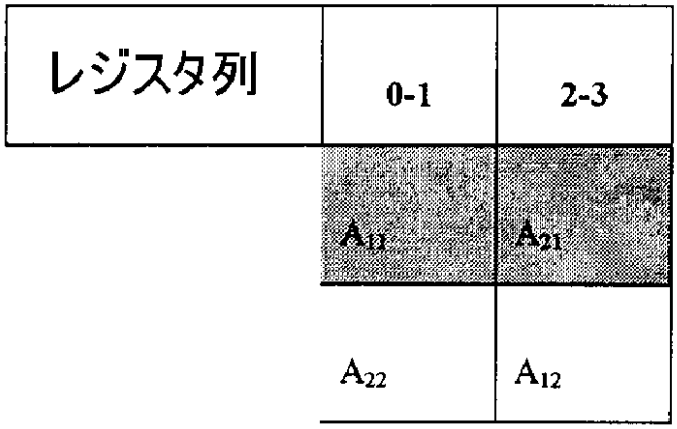


表4−二つの複素数語の別の配置

【００８１】
図１３は、レジスタ・ファイル１０６の所与のポートに関連するクロスバー・スイッチ２１０である。このスイッチは、図に示すように配置されている、マルチプレクサ２１２－１、２１２－２、２１２－３および２１２－４、およびドライバ２１４－１、２１４－２、２１４－３および２１４－４を含む。図の一番下の部分には、テーブル３に示すレジスタ・ファイルの列のビット線に取り付けられているバスが位置する。この図の一番上の部分には、ｃｍｐｌｘａおよびｃｍｐｌｘｂと表示されている、二つの複素数語の幅を持つバスが位置している。上記バス、ｃｍｐｌｘａおよびｃｍｐｌｘｂは、両方で図７のポートｒ１またはｒ２のうちの一方を形成する。
【００８２】
クロスバー・スイッチ２１０は、マルチプレクサ２１２－１、２１２－２、２１２－３および２１２－４の制御入力に供給された「逆」信号を通して、二つの複素数語の順序を反対するオプションを含む、図の一番上および一番下のところの二組のバスの二方向相互接続を行う。「逆」ロジック信号だと断定されない場合には、コラム０オプション１はｃｍｐｌｘａに接続され、列２および３はｃｍｐｌｘｂに接続される。「逆」ロジック信号と断定される場合には、相互接続は反対になる。クロスバー・スイッチ２１０の方向性は、「書込み」信号および「読出し」信号により制御される。一度に断定されるのは、これら信号のうちの一方だけである。「書込み」信号と断定された場合には、データはスイッチ２１０を通って下方に流れ、レジスタ・ファイルの内容はそのポートを通して書き込まれる。「読出し」信号であると断定された場合には、データはスイッチ２１０を通って上方に流れ、レジスタ・ファイル１０６の内容は、そのポートを通して読み出される。
【００８３】
テーブル３の記憶スキームと、図１３のクロスバー・スイッチ２１０とを組み合せることにより、「行列図」の行または列どちらかの上の、隣接する二つの複素数語に順次アクセスすることができ、以下に説明するように、高性能のマトリックス－マトリックス乗算を行うことができる。テーブル５は、二つの行２、３上に常駐する、種々のペアにアクセスするためのレジスタ・ファイル１０６およびクロスバー・スイッチ２１０の使用方法のいくつかの例を示す。
【表５】

複素数語のペア	列0－1の行	列2－3の行	逆
A_{31}, A_{32}	2	2	no
A_{31}, A_{41}	2	3	no
A_{41}, A_{42}	3	3	yes
A_{32}, A_{42}	3	2	yes

表5-アクセス例

【００８４】
テーブル６は、図の一番上の部分において、図１２のレジスタ・ファイル１０６に入る可能な制御フィールドを示す。ｄｒｏｗの他に、二つの行のうちのどれを使用すべきかを示す多重ビット・フィールド、およびアクセスは単一複素数語に対してのものなのか、または二重複素数語に対してのものなのかを示す１ビット・フィールドｓｄ、およびｓｄ＝単一である場合に、列番号を示し、ｓｄ＝二重である場合に、二つの複素数語が、同じ行上に存在するのか、同じ列（行列図）上に存在するのかを示す１ビット・フィールドｘ、および二つの可能な列の列またはスタート列を示す１ビット・フィールドｃが存在する。
【表６】

ｄｒｏｗ多重ビット	ｓｄ １ビット	ｘ １ビット	ｃ １ビット	
	単一	行	列	単一複素数語
	二重	0	スタート列	行上の 二重複素数語
	二重	1	スタート列	列上の 二重複素数語

表６－レジスタ・ファイル・アクセス・モードに対するアドレス・コード化

【００８５】
本発明は、複素数計算、すなわち、複素数を含む計算を実行する際に使用するのに特によく適している。しかし、本発明は実数値だけを含む計算にも使用することができる。例えば、図１２に示すようなレジスタ・ファイルが、実数だけを含んでいると仮定しよう。当業者であれば、複素数計算に関連する上記技術を、簡単な方法で、一つの行または一つの列上の、四つの連続している実数値にアクセスできるように、修正することができることは周知であろう。より詳細に説明すると、図１２のデコーダ２００は、一つを各列内の各語にというふうに、語線の数を、二重に出力するように構成することができる。テーブル

７は、実数の行列の四つの列の語を、これを行うためのレジスタ・ファイルのコラム０－３に、どのような方法で記憶させることができるかを示す。

【表７】

レジスタ列	0	1	2	3
	A_{11}	A_{21}	A_{31}	A_{41}
	A_{42}	A_{12}	A_{22}	A_{32}
	A_{33}	A_{43}	A_{13}	A_{23}
	A_{24}	A_{34}	A_{44}	A_{14}

表7-実数の配置

【００８６】

この実数計算をサポートするためのクロスバー・スイッチは、図１３のスイッチ２１０より大きい。また、マトリックス－マトリックス乗算が、レジスタ・ファイル１０６および乗算ハードウェアを完全に使用することができるように、図８の乗算装置１５０の一番下の最後の加算装置の数は１６に増大しなければならない。

【００８７】

＜プロセッサ素子の線形アレー＞

高い効率で行列演算を行うために、上記のように構成されている多数のプロセッサ素子は、線形アレーの形に相互接続される。異なるプロセッサ素子相互接続技術は、通常、例えば、ＱＲ分解、逆置換、マトリックス反転、マトリックス－マトリックス乗算、および共解散および相互相関の形成のような、異なるタイプの計算のために使用される。

【００８８】

図１４は、左側に一つのヘッド・プロセッサ素子２２２、その右側に１６の通常のプロセッサ素子を含むプロセッサ素子の１７の素子の線形アレー２２０である。各プロセッサ素子は、その最も近い隣接プロセッサ素子と二方向相互接続を持つ。各プロセッサ素子は、また、図７の（入力頂部用の）表示「ｉｔ」に対応する、下向き矢印で示す頂部からの外部入力を持つ。この線形アレー構成は、１７×１６行列式上でＱＲ分解を行ったり、逆置換を行ったりするのに適している。これについては、以下にさらに詳細に説明する。

【００８９】

図１５は、左側に一つのヘッド・プロセッサ素子２３２、およびその右側に三つの通常のプロセッサ素子を含むプロセッサ素子の４素子線形アレー２３０である。図は、時計方向の円形相互接続を示す。この構成は、以下に説明する方法により、マトリックス－マトリックス乗算を行うことができる。この場合、一番左側のプロセッサ素子２３２は、通常、ヘッド・プロセッサ素子に関連する上記の非線形関数能力を持つ必要はない。

【００９０】

図１６は、左側に一つのヘッド・プロセッサ素子２４２を持ち、その右側に四つの通常のプロセッサ素子を持つプロセッサ素子の５素子線形アレー２４０である。時計方向の円形

相互接続は、最も左側の四つのプロセッサ素子、および四番目のプロセッサ素子から五番目のプロセッサ素子への、左から右への一つの追加の接続用に形成される。この構成は、以下に説明するように、最も左の四つのプロセッサ素子において、共分散行列の計算を、最も右のプロセッサ素子において、相互相関関係ベクトルの計算を行うことができる。この場合もまた、最も左のプロセッサ素子２４２は、通常、ヘッド・プロセッサ素子に関連する非線形関数能力を持つ必要はない。

【００９１】

図１４－図１６に示すような、本発明のプロセッサ素子の線形アレーは、通常、種々の演算を行うように構成されているポート、演算回路、およびレジスタ・ファイルを含む。図１４－図１６に示すタイプのプロセッサ素子のアレーは、所与の時間に、アレー内の各プロセッサ素子が、それ自身のデータに対して、個々の演算または命令を実行することができることを意味する、多重命令多重データ（ＭＩＭＤ）と呼ばれる、クラスであると見なすことができる。アレーの機能を制御するために、プログラムを使用することができる。このプログラム制御は、例えば、１９９１年9月１３日付のサイエンス２５３巻、１２３３ページ掲載の、Ｊ．Ａ．フイッシャおよびＢ．Ｒ．ラウの「命令レベルの並列処理」記載の、超長命令語（ＶＬＩＷ）と呼ばれるクラスのものであってもよい。一つの長い命令語は、アレーの各プロセッサ素子を制御するための、個々の動作を含むことができる。

【００９２】

超長語命令は、周知の従来の技術を使用する、プロセッサの線形アレーを制御するために順番に並べることができる。一組の超長語命令を含むプログラムは、メモリに常駐している。一つの集積回路上のインスタンシエーションのために、一つのメモリ語に全部の超長語命令を保持するために、十分なビットからなる非常に広いアクセス・サイズを持つように、メモリを構成することができる。上記メモリから対応する線形アレーへの、超長語命令の転送は、図２に示すような、従来の二次元アレーの制御のために必要な転送より、簡単であることに留意されたい。

【００９３】

すでに説明したように、線形アレー上での異なるアルゴリズムの実行は、プロセッサ素子間の異なる相互接続を必要とする場合がある。例えば、ＱＲ分解は、通常、アレー内の素子の数が行列に１を足した次元に等しい場合には、右端の素子からヘッド素子へのフィードバックを必要としない。マトリックス－マトリックス乗算は、通常、使用中の右側の最後の素子からヘッド素子への、フィードバック接続を必要とする。共分散および相互相関は、通常、最後の素子の隣の素子からヘッド素子への、フィードバック経路を必要とする。

【００９４】

プロセッサ素子の線形アレーを備える集積回路が、異なるアルゴリズムを実行できるようにするために、追加ロジックを追加することができる。図１７は、上記のような追加ロジックを含む、５プロセッサ素子線形アレー２５０を示す。プロセッサ素子２５２はヘッド・プロセッサ素子を表わす。三状態ゲート２５４－１、２５４－２、２５４－３および２５４－４は、左から右へのバス接続を、ヘッドプロセッサ素子２５２に戻る、フィードバック・バス２５６に接続している。三状態ゲートのうちの一つだけしか能動状態になることができず、すべての他のゲートは能動状態ではないので、フィードバック・バス２５６を駆動することはできない。同様に、マルチプレクサ２５８－１、２５８－２、２５８－３および２５８－４は、右から左への接続とヘッドプロセッサ素子２５２により駆動されるフィードバック・バス２６０の間で選択を行う。

【００９５】

三状態ゲート２５４－１、２５４－２、２５４－３および２５４－４、およびマルチプレクサ２５８－１、２５８－２、２５８－３および２５８－４の制御は、アレーの相互接続を形成するために、所与のアルゴリズムの冒頭のところで、制御レジスタに静的に書込むことができる。これは、「構成可能な計算」と呼ばれるアプローチの一例である。

【００９６】

複素数行列により種々の演算を行うためのプロセッサ素子のアレーは、通常、アレーにデータを供給し、アレーにどの計算を行うのかを知らせ、計算終了後に結果を受け取るための、もっと大きなシステムへのインターフェースを必要とする。例えば、アレーは、特定用途向けロジックにインターフェースを通して、接続することもできるし、またはホスト・プロセッサにインターフェースを通して、接続することもできる。ホスト・プロセッサ自身は、マイクロプロセッサ、デジタル信号プロセッサ、ワークステーション、または他の任意のタイプのデジタル処理デバイスを使用することができる。

【００９７】

図１８は、共プロセッサ３０２およびホスト・プロセッサ３０４を含む、処理システム３００の一例である。共プロセッサ３０２は、一つのヘッドプロセッサ素子３０６および四つの通常のプロセッサ素子を含む５プロセッサ素子線形アレーを含む。線形アレーは、すでに説明したような方法で、アレー内でデータを送るために、最も近くの隣接するアレー間の接続を含む。プログラム・メモリ３０８は、上記のように、アレーの個々の素子の動作を制御するための、一つまたはそれ以上のプログラムを含む。アレーへの命令の適用は、周知の従来のタイプのものであってよい、プログラム・シーケンサ３１０により制御される。この例の各プロセッサ素子は、三つのポートを含むレジスタ・ファイルを含む。所与のプロセッサ素子の、レジスタ・ファイルのポートのうちの二つのポートは、プログラム・メモリの命令により制御されるが、第三のポートは、インターフェース・ロジック・ブロック３１２により独立に制御される。より詳細に説明すると、インターフェース・ロジック・ブロック３１２は、各プロセッサ素子用のポートｒ３のアドレスを制御する。インターフェース・ロジック・ブロック３１２は、各レジスタ・ファイルに、データを書込むことができるし、または各プロセッサ素子のポートｒ３を通して、各レジスタ・ファイルから、データを読み出すことができる。

【００９８】

図に示すように、一つのインターフェース・ポート３１３が、インターフェース・ロジック・ブロック３１２を、ホスト・プロセッサ３０４の直接メモリ・アドレス（ＤＭＡ）ポート３１４に接続している。インターフェース・ポート３１３を使用することにより、ホスト・プロセッサ３０４のメモリと、インターフェース・ロジック・ブロック３１２との間で、データをやりとりすることができる。もう一つのインターフェース・ポート３１５により、どんなデータを書き込んだり、読み出しするのかを指定するために、ホスト・プロセッサ３０４からプログラム・シーケンサ３１０に制御を渡すことができる。インターフェース・ロジック・ブロック３１２は、ホストＤＭＡポートへの、またはホストＤＭＡポート３１４からのデータ転送を、アレーのレジスタ・ファイルに必要な分散記憶することができる。必要なマッピングの一例は、ホスト・プロセッサ・メモリのテーブル２から、所与のプロセッサ素子のレジスタ・ファイルのテーブル３へ送られるものである。さらに、ホスト・プロセッサ３０４は、プログラム・シーケンサ３１０に、どんなプログラムを実行するのか、その実行を何時開始するのかを示すコマンドを送ることができる。プログラム・シーケンサ３１０は、例えば、ホスト・プロセッサ３０４に、特定のプログラムの実行を何時終了するのかを知らせるために、状態をホスト・プロセッサ３０４に送ることもできる。ホスト・プロセッサ３０４は、また、例えば、プログラム・メモリ３０８に、異なるプログラムをダウンロードするために、第二のＤＭＡポート３１６および接続３１７を通して、共プロセッサ３０２のプログラム・メモリ３０８と直接通信することもできる。接続３１７は、この接続がオプションであることを示すために一点鎖線で表わす。

【００９９】

共プロセッサ３０２は、例えば、汎用ホスト・コンピュータに、インターフェースを通して接続していて、実行する異なるプログラムとともに動的にロードすることができる、柔軟な共プロセッサとして、読出し専用メモリに、または多数の他の構成に記憶しているそのプログラムにより、一つの固定アルゴリズムを実行する特定用途向けＩＣ（ＡＳＩＣ）に埋設されている一つのモジュールの形で実行することができる。埋設したＡＳＩＣ用途の場合には、データにとっては、図１８のレジスタ・ファイルのポートｒ３を通してより

も、各プロセッサ素子上のポートを通して直接アレーを入力し、出力したほうが便利である。

【０１００】

ＱＲ分解、逆置換、マトリックス反転、マトリックス－マトリックス乗算、および共分散および相互相関の形成のいくつかの例のための、本発明のプロセッサ素子の線形アレーの動作を以下に詳細に説明する。

【０１０１】

＜ＱＲ分解＞

すでに説明したように、ＱＲ分解は、二次元三角収縮アレーを使用して、従来技術により達成される。この節においては、図１４に示す本発明の線形アレー上で、同じアルゴリズムを、かなり少ないハードウェアを使用して、ほとんど同じ性能で実行する方法について以下に説明する。

【０１０２】

図２示すものから図１４に示すものへの、相互接続のマッピングについて、最初に説明する。以下に説明するように、仮想の二次元三角アレーは、物理的な一次線形アレーにマッピングすることができる。三角形のアレーの丸い（ｒｏｎｕｄ）各ヘッドプロセッサ素子は、線形アレーの一つのヘッドプロセッサ素子にマッピングされる。丸いプロセッサ素子の右に隣接する、三角アレーの四角い（ｓｑｕａｒｅ）通常の各プロセッサ素子は、丸いプロセッサ素子の右に隣接する線形アレーの通常のプロセッサ素子にマッピングされる。このパターンは継続して行われる。図１４の線形アレー２２０のプロセッサ素子間の相互接続は、二方向の相互接続である。二次元三角アレーの一つの素子は、データをすぐ下の列の送り、これは線形アレーでマッピングされ、データを左に送る。このようにして、二次元三角アレー上で行われたすべての計算は、線形アレーにマッピングされるが、例外が一つだけある。すなわち、二次元の場合に可能な本当の平行状態ではなく、線形アレーは、行列の新しいコラム上で計算がスタートする度に、計算を位相φだけずらさなければならないことである。そうすることにより、以下により詳細に説明するように、ＱＲ分解の全待ち時間が若干長くなる。

【０１０３】

この例の場合には、図７のレジスタ・ファイルの唯一の使用方法は、結果が発生した場合、その結果を書き込むことである。上記結果はレジスタ・ファイル内に残り、プロセッサにより書き戻す際には示されない。

【表８】

命令 ニーモニック	丸い 要素	四角い 要素	説　明
ｃｓｒｇ	ｙ		右およびｇ１レジスタからの入力、ｃおよびｓの計算
ｃｓｒｖ	ｙ		右および最後の数値からの入力、ｃおよびｓの計算
ｃｓｔｇ	ｙ		頂部およびｇ１レジスタからの入力、ｃおよびｓの計算
ｃｓｔｖ	ｙ		頂部および最後の数値からの入力、ｃおよびｓの計算
ｌｄｒｔ	ｙ	ｙ	ｇ１レジスタを右端からロード
ｌｄｒｗ	ｙ	ｙ	ｌｄｒｔおよびｒｇｖｗ
ｌｄｔｐ	ｙ	ｙ	ｇ１レジスタを一番上からロード
ｒｇｒｗ	ｙ	ｙ	レジスタ・ファイルの右からの書き込み
ｒｇｖｗ	ｙ	ｙ	レジスタ・ファイルの最後の数値からの書き込み
ｒｏｒｇ	ｙ	ｙ	右およびｇ１レジスタからの入力の回転
ｒｏｒｖ	ｙ	ｙ	右および最後の数値からの入力の回転
ｒｏｔｇ	ｙ	ｙ	頂部およびｇ１レジスタからの入力の回転
ｒｏｔｖ	ｙ	ｙ	頂部および最後の数値からの入力の回転

表８－命令ニーモニック

【０１０４】
図８は、図１４の線形アレー２２０のプロセッサ素子のための、可能な一組の命令を示す。ニーモニックは最初のコラム内に表示されている。第二および第三のコラム内の「ｙ」は、ニーモニックが、それぞれ、丸いまたは四角いプロセッサ素子に、適用することができることを示す。第四のコラムは、命令の実行内容を示す。説明が示すように、頂部または右側からロードすることができる各プロセッサ素子の入力のところに、「ｇ１」レジスタが位置するものと仮定する。
【０１０５】
テーブル９は、テーブル８に示す命令を使用して、ＱＲ分解を行うプログラムである。「ｃｓｔｖ２」の２ように、ニーモニックに続く数字は、プロセッサ素子により無視されるもので、プログラムの理解を助けるために、ソース・コードだけに存在するものである。入力ばブランクである場合には、「ｎｏｏｐ」命令お実行すべきことを示す。「ｎｏｏｐ」は、新しい計算をスタートしない。しかし、進行中の任意の前の計算に対しては、パイプライン処理を続行する。一番上の行は列の意味を示す。第一列は、ステップおよびサブステップで測定した経過時間のカウントを示す。サブステップは、テーブルの１行を占有し、パイプラインの説明の上に示す長さの時間を必要とする。この例の場合、命令の終了の待ち時間は５φであり、一つのステップと等しい。第二から第八までの列は、図１４の１７のプロセッサ素子を制御する命令に対応する。
【０１０６】
このテーブルの一つの行は、超長語命令と見なすことができる。プロセッサ素子の線形アレーを制御するための超長語命令を順番に並べる方法は、当業者にとっては周知のものである。

【０１０７】
テーブル９のプログラムの開始の近くの、各ステップの第一のサブステップ上でだけ、命令をスタートさせることができることに留意されたい。何故なら、次の命令は、前の命令の結果を必要とし、待ち時間が一つのステップであるからである。しかし、ステップ４のサブステップ２に到達した場合には、ゼロがすでに二つ存在し、プログラムは、パイプラインに相互に位置する二つの「ｃ」と「ｓ」の計算を開始することができる。計算が、プロセッサ素子の二次元三角アレー上で行われている場合には、「ｃｓｔｖ４」ｐ「ｃｓｒｇ１６」は、同時に実行することができる。本発明の場合には、これら演算は完全には同時に行われないで、サブステップφだけずれる。このずれは、従来の二次元三角アレーから、本発明の一次元線形アレーへ、移行するさいに必要なプロセッサ素子の、かなりの低減からみればささいなことである。プログラムが進行するにつれて、ますます多くのパイプライン処理スロットが使用されるのを観察することができる。ステップ１３により、パイプラインは「ｃ」および「ｓ」の５回の計算で一杯になる。プログラムが終了するまでには、パイプラインをもはや満たすことはできない。テーブル９の最後の行は、各プロセッサ素子の使用の百分率を示す。

【表９】

表9-パイプライン処理接続線形アレー上での、17×16QR分解

step	rnd	sqr2	sqr3	sqr4	sqr5	sqr6	sqr7	sqr8	sqr9	sqr10	sqr11	sqr12	sqr13	sqr14	sqr15	sqr16	sqr17
0/1	ldtp																
/2																	
/3																	
/4																	
/5																	
1/1	cstg1	ldtp															
/2																	
/3																	
/4																	
/5																	
2/1	cstv2	rotg1	ldtp														
/2																	
/3																	
/4																	
/5																	
3/1	cstv3	rotv2	rotg1	ldtp													
/2	ldrt																
/3																	

/4																	
/5																	
4/1	cstv4	rotv3	rotv2	rotg1	ldtp												
/2	csrg16	ldrt															
/3																	
/4																	
/5																	
5/1	cstv5	rotv4	rotv3	rotv2	rotg1	ldtp											
/2	csrv17	rorg16	ldrt														
/3																	
/4																	
/5																	
6/1	cstv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp										
/2	csrv18	rorv17	rorg16	ldrt													
/3	ldrt																
/4																	
/5																	
7/1	cstv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp									
/2	csrv19	rorv18	rorv17	rorg16	ldrt												
/3	csrg30	ldrt															
/4																	
/5																	
8/1	cstv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp								

/2	csrv20	rorv19	rorv18	rorv17	rorg16	ldrt											
/3	csrv31	rorg30	ldrt														
/4																	
/5																	
9/1	cstv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp							
/2	csrv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt										
/3	csrv32	rorv31	rorg30	ldrt													
/4	ldrt																
/5																	
10/1	cstv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp						
/2	csrv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt									
/3	csrv33	rorv32	rorv31	rorg30	ldrt												
/4	csrg43	ldrt															
/5																	
11/1	cstv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp					
/2	csrv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt								
/3	csrv34	rorv33	rorv32	rorv31	rorg30	ldrt											
/4	csrv44	rorg43	ldrt														
/5																	
12/1	cstv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp				
/2	csrv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt							
/3	csrv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt										
/4	csrv45	rorv44	rorg43	ldrt													
/5	ldrt																

13/1	cstv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp			
/2	csrv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt						
/3	csrv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt									
/4	csrv46	rorv45	rorv44	rorg43	ldrt												
/5	csrg55	ldrt															
14/1	cstv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp		
/2	csrv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt					
/3	csrv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt								
/4	csrv47	rorv46	rorv45	rorv44	rorg43	ldrt											
/5	csrv56	rorg55	ldrt														
15/1	cstv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp	
/2	csrv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt				
/3	csrv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt							
/4	csrv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt										
/5	csrv57	rorv56	rorg55	ldrt													
16/1	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1	ldtp
/2	csrv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt			
/3	csrv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt						
/4	csrv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt									
/5	csrv58	rorv57	rorv56	rorg55	ldrt												
17/1	csrg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2	rotg1
/2	csrv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt		
/3	csrv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt					

/4	csrv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt								
/5	csrv59	rorv58	rorv57	rorv56	rorg55	ldrt											
18/1	csrv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3	rotv2
/2	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	ldrt	
/3	csrv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt				
/4	csrv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt							
/5	csrv60	rorv59	rorv58	rorv57	rorv56	rorg55	ldrt										
19/1	csrv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4	rotv3
/2	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	rorg16	
/3	csrv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt			
/4	csrv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt						
/5	csrv61	rorv60	rorv59	rorv58	rorv57	rorv56	rorg55	ldrt									
20/1	csrv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5	rotv4
/2	csrg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	rorv17	
/3	rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30	ldrt		
/4	csrv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt					
/5	csrv62	rorv61	rorv60	rorv59	rorv58	rorv57	rorv56	rorg55	ldrt								
21/1	csrv70	rorv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6	rotv5
/2	csrv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	rorv18	
/3		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31	rorg30		
/4	csrv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt				
/5	csrv63	rorv62	rorv61	rorv60	rorv59	rorv58	rorv57	rorv56	rorg55	ldrt							
22/1	csrv71	rorv70	rorv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7	rotv6
/2	csrv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	rorv19	

/3	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32	rorv31		
/4	rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43	ldrt			
/5	csrv64	rorv63	rorv62	rorv61	rorv60	rorv59	rorv58	rorv57	rorv56	rorg55	ldrt						
23/1	csrv72	rorv71	rorv70	rorv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8	rotv7
/2	csrv79	rorv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	rorv20	
/3	csrg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33	rorv32		
/4		rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44	rorg43			
/5	csrv65	rorv64	rorv63	rorv62	rorv61	rorv60	rorv59	rorv58	rorv57	rorv56	rorg55	ldrt					
24/1	csrv73	rorv72	rorv71	rorv70	rorv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9	rotv8
/2	csrv80	rorv79	rorv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	rorv21	
/3	csrv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34	rorv33		
/4			rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45	rorv44			
/5	rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61	rorv60	rorv59	rorv58	rorv57	rorv56	rorg55	ldrt				
25/1	csrv74	rorv73	rorv72	rorv71	rorv70	rorv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10	rotv9
/2	csrv81	rorv80	rorv79	rorv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	rorv22	
/3	csrv87	rorv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35	rorv34		
/4	ldrt			rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46	rorv45			
/5		rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61	rorv60	rorv59	rorv58	rorv57	rorv56	rorg55				
26/1	csrv75	rorv74	rorv73	rorv72	rorv71	rorv70	rorv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11	rotv10
/2	csrv82	rorv81	rorv80	rorv79	rorv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	rorv23	
/3	csrv88	rorv87	rorv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36	rorv35		
/4	csrg93	ldrt			rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47	rorv46			
/5			rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61	rorv60	rorv59	rorv58	rorv57	rorv56				

27/1	rgvw5	rorv75	rorv74	rorv73	rorv72	rorv71	rorv70	rorv69	rorv68	rorv67	rorg66	ldrw0	rotv15	rotv14	rotv13	rotv12	rotv11
/2	csrv83	rorv82	rorv81	rorv80	rorv79	rorv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	rorv24	
/3	csrv89	rorv88	rorv87	rorv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37	rorv36		
/4	csrv94	rorg93	ldrt			rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48	rorv47			
/5				rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61	rorv60	rorv59	rorv58	rorv57				
28/1		rgvw5	rorv75	rorv74	rorv73	rorv72	rorv71	rorv70	rorv69	rorv68	rorv67	rorg66	rgvw0	rotv15	rotv14	rotv13	rotv12
/2	csrv84	rorv83	rorv82	rorv81	rorv80	rorv79	rorv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	rorv25	
/3	csrv90	rorv89	rorv88	rorv87	rorv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38	rorv37		
/4	csrv95	rorv94	rorg93	ldrt			rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49	rorv48			
/5	ldrt				rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61	rorv60	rorv59	rorv58				
29/1			rgvw5	rorv75	rorv74	rorv73	rorv72	rorv71	rorv70	rorv69	rorv68	rorv67		rgvw0	rotv15	rotv14	rotv13
/2	rgvw6	rorv84	rorv83	rorv82	rorv81	rorv80	rorv79	rorv78	rorv77	rorg76	ldrt	rgvw1	rorv29	rorv28	rorv27	rorv26	
/3	csrv91	rorv90	rorv89	rorv88	rorv87	rorv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39	rorv38		
/4	csrv96	rorv95	rorv94	rorg93	ldrt			rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50	rorv49			
/5	csrg100	ldrt				rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61	rorv60	rorv59				
30/1				rgvw5	rorv75	rorv74	rorv73	rorv72	rorv71	rorv70	rorv69	rorv68			rgvw0	rotv15	rotv14
/2		rgvw6	rorv84	rorv83	rorv82	rorv81	rorv80	rorv79	rorv78	rorv77	rorg76		rgvw1	rorv29	rorv28	rorv27	
/3	csrv92	rorv91	rorv90	rorv89	rorv88	rorv87	rorv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40	rorv39		
/4	csrv97	rorv96	rorv95	rorv94	rorg93	ldrt			rgvw3	rorv54	rorv53	rorv52	rorv51	rorv50			
/5	csrv101	rorg100	ldrt				rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61	rorv60				
31/1					rgvw5	rorv75	rorv74	rorv73	rorv72	rorv71	rorv70	rorv69				rgvw0	rotv15
/2			rgvw6	rorv84	rorv83	rorv82	rorv81	rorv80	rorv79	rorv78	rorv77			rgvw1	rorv29	rorv28	
/3	rgvw7	rorv92	rorv91	rorv90	rorv89	rorv88	rorv87	rorv86	rorg85	ldrt		rgvw2	rorv42	rorv41	rorv40		
/4	csrv98	rorv97	rorv96	rorv95	rorv94	rorg93	ldrt			rgvw3	rorv54	rorv53	rorv52	rorv51			

/5	csrv102	rorv101	rorg100	ldrt				rgvw4	rorv65	rorv64	rorv63	rorv62	rorv61				
32/1	ldrt					rgvw5	rorv75	rorv74	rorv73	rorv72	rorv71	rorv70					rgvw0
/2				rgvw6	rorv84	rorv83	rorv82	rorv81	rorv80	rorv79	rorv78				rgvw1	rorv29	
/3		rgvw7	rorv92	rorv91	rorv90	rorv89	rorv88	rorv87	rorv86	rorg85			rgvw2	rorv42	rorv41		
/4	csrv99	rorv98	rorv97	rorv96	rorv95	rorv94	rorg93	ldrt			rgvw3	rorv54	rorv53	rorv52			
/5	csrv103	rorv102	rorv101	rorg100	ldrt				rgvw4	rorv65	rorv64	rorv63	rorv62				
33/1	csrg106	ldrt					rgvw5	rorv75	rorv74	rorv73	rorv72	rorv71					
/2					rgvw6	rorv84	rorv83	rorv82	rorv81	rorv80	rorv79					rgvw1	
/3			rgvw7	rorv92	rorv91	rorv90	rorv89	rorv88	rorv87	rorv86				rgvw2	rorv42		
/4	rgvw8	rorv99	rorv98	rorv97	rorv96	rorv95	rorv94	rorg93	ldrt			rgvw3	rorv54	rorv53			
/5	csrv104	rorv103	rorv102	rorv101	rorg100	ldrt				rgvw4	rorv65	rorv64	rorv63				
34/1	csrv107	rorg106	ldrt					rgvw5	rorv75	rorv74	rorv73	rorv72					
/2						rgvw6	rorv84	rorv83	rorv82	rorv81	rorv80						
/3				rgvw7	rorv92	rorv91	rorv90	rorv89	rorv88	rorv87					rgvw2		
/4		rgvw8	rorv99	rorv98	rorv97	rorv96	rorv95	rorv94	rorg93				rgvw3	rorv54			
/5	csrv105	rorv104	rorv103	rorv102	rorv101	rorg100	ldrt				rgvw4	rorv65	rorv64				
35/1	csrv108	rorv107	rorg106	ldrt					rgvw5	rorv75	rorv74	rorv73					
/2	ldrt						rgvw6	rorv84	rorv83	rorv82	rorv81						
/3					rgvw7	rorv92	rorv91	rorv90	rorv89	rorv88							
/4			rgvw8	rorv99	rorv98	rorv97	rorv96	rorv95	rorv94					rgvw3			
/5	rgvw9	rorv105	rorv104	rorv103	rorv102	rorv101	rorg100	ldrt				rgvw4	rorv65				
36/1	csrv109	rorv108	rorv107	rorg106	ldrt					rgvw5	rorv75	rorv74					
/2	csrg111	ldrt						rgvw6,	rorv84	rorv83	rorv82						

/3						rgvw7	rorv92	rorv91	rorv90	rorv89							
/4				rgvw8	rorv99	rorv98	rorv97	rorv96	rorv95								
/5		rgvw9	rorv105	rorv104	rorv103	rorv102	rorv101	rorg100					rgvw4				
37/1	csrv110	rorv109	rorv108	rorv107	rorg106	ldrt					rgvw5	rorv75					
/2	csrv112	rorg111	ldrt						rgvw6	rorv84	rorv83						
/3							rgvw7	rorv92	rorv91	rorv90							
/4					rgvw8	rorv99	rorv98	rorv97	rorv96								
/5			rgvw9	rorv105	rorv104	rorv103	rorv102	rorv101									
38/1	rgvw10	rorv110	rorv109	rorv108	rorv107	rorg106	ldrt					rgvw5					
/2	csrv113	rorv112	rorg111	ldrt						rgvw6	rorv84						
/3	ldrt							rgvw7	rorv92	rorv91							
/4						rgvw8	rorv99	rorv98	rorv97								
/5				rgvw9	rorv105	rorv104	rorv103	rorv102									
39/1		rgvw10	rorv110	rorv109	rorv108	rorv107	rorg106										
/2	csrv114	rorv113	rorv112	rorg111	ldrt						rgvw6						
/3	csrg115	ldrt							rgvw7	rorv92							
/4							rgvw8	rorv99	rorv98								
/5					rgvw9	rorv105	rorv104	rorv103									
40/1			rgvw10	rorv110	rorv109	rorv108	rorv107										
/2	rgvw11	rorv114	rorv113	rorv112	rorg111	ldrt											
/3	csrv116	rorg115	ldrt							rgvw7							
/4								rgvw8	rorv99								
/5						rgvw9	rorv105	rorv104									
41/1				rgvw10	rorv110	rorv109	rorv108										

/2		rgvw11	rorv114	rorv113	rorv112	rorg111											
/3	csrv117	rorv116	rorg115	ldrt													
/4	ldrt								rgvw8								
/5							rgvw9	rorv105									
42/1					rgvw10	rorv110	rorv109										
/2			rgvw11	rorv114	rorv113	rorv112											
/3	rgvw12	rorv117	rorv116	rorg115	ldrt												
/4	csrg118	ldrt															
/5								rgvw9									
43/1						rgvw10	rorv110										
/2				rgvw11	rorv114	rorv113											
/3		rgvw12	rorv117	rorv116	rorg115												
/4	csrv119	rorg118	ldrt														
/5																	
44/1							rgvw10										
/2					rgvw11	rorv114											
/3			rgvw12	rorv117	rorv116												
/4	rgvw13	rorv119	rorg118	ldrt													
/5	ldrt																
45/1																	
/2						rgvw11											
/3				rgvw12	rorv117												
/4		rgvw13	rorv119	rorg118													

/5	esrg120	ldrt															
46/1																	
/2																	
/3					rgvw12												
/4			rgvw13	rorv119													
/5	rgvw14	rorg120	ldrt														
47/1																	
/2																	
/3																	
/4				rgvw13													
/5		rgvw14	rorg120														
48/1	rgrw15																
/2																	
/3																	
/4																	
/5			rgvw14														
49/1		rgrw15															
util	62%	62%	62%	60%	59%	57%	54%	51%	48%	44%	40%	36%	31%	26%	20%	14%	7%

【０１０８】
上記例の場合には、三角形に変換される式の行列要素は、アレーの頂部のところの入力を通して、図１４の線形アレー２２０に入力される。他の実施形態の場合には、三角形に変換される行列要素は、アレーで行われた前の計算により、そこに残留している状態で、アレーのレジスタ・ファイルに予め位置することができる。もう一つの他の実施形態の場合には、三角形に変換される行列要素は、図７のポートｒ３を通して、ＤＭＡインターフェースのような他のインターフェースにより、レジスタ・ファイルに導入される。
【０１０９】
すでに説明したように、二次元三角アレーを通して線形アレーへ行くための待ち時間の延長は、毎回、アルゴリズムにおいて、新しい列をゼロにするプロセスを開始する機会であり、線形アレーは、一つのサブステップφだけ遅延する。テーブル９の第二の列を見れば、例示としてのプログラムの場合には、１５の新しい列をゼロに変換するための機会の回数に対応して、これが１５回行われることがわかる。それ故、この例の場合には、線形アレー２２０が必要とする追加の待ち時間は、サブステップ１５回分であり、これは３回のステップに等しい。
【０１１０】
テーブル１０は、従来の二次元三角アレーの利点の数と、本発明のパイプライン処理の線形アレー２２０の利点の数との比較を示す。パイプライン処理線形アレー２２０は、三角アレーが必要とするプロセッサ素子の数の１１％のプロセッサ素子しか必要としないが、待ち時間は６％しか増大していないことがわかる。しかし、回路設計にパイプライン段を

導入した場合、パイプライン処理線形アレーの「ステップ」時間が、三角アレーのそれよりすこし長くなる恐れがある。さらに、線形アレーの各プロセッサ素子は、通常、二次元三角アレー全体にわたって、例えば、分散した形で存在している一時的記憶装置よりも容量が少ない、レジスタ・ファイルの形の一時的記憶装置を含んでいなければならない。しかし、本発明を使用すれば、待ち時間をほんの少し損するだけで、ＱＲ分解を行うために必要な回路の数をかなり低減することができる。

【表１０】

	三角アレー	パイプライン 処理線形アレー
プロセッサ素子の数	１５２	１７
最後の命令ステップ	４６／１	４９／１

表１０－１７×１６ＱＲ分解の利点の数

【０１１１】

＜逆置換＞

図１９は、本発明の５素子線形アレー３２０である。この線形アレー３２０は、上記式（１３）を解くための二方向相互接続を含む。ＱＲ分解プロセスの終了時に計算を行う、アレー３２０のヘッドプロセッサ素子３２２は、ｒ行列である、r_{11}、r_{22}、r_{33}およびr_{44}のすべての主要な対角線方向の素子を保持する。その右に隣接するプロセッサ素子は、右に隣接する対角線方向の素子、すなわち、r_{12}、r_{23}、r_{34}およびy_4を保持する。

【０１１２】

テーブル１１は、図１９の線形アレー３２０が、式（１３）を解く方法の全体的な概要を示す。

【表１１】

	rnd1	sqr2	sqr3	sqr4	sqr5
Step 1	$1/r_{44}$				
Step 2	x_4				
Step 3	$1/r_{33}$	$y_3 - r_{34}x_4$			
Step 4	x_3		$y_2 - r_{24}x_4$		
Step 5	$1/r_{22}$	$y_2 - r_{24}x_4 - r_{23}x_3$		$y_1 - r_{14}x_4$	
Step 6	x_2		$y_1 - r_{14}x_4 - r_{13}x_3$		
Step 7	$1/r_{11}$	$y_1 - r_{14}x_4 - r_{13}x_3 - r_{12}x_2$			
Step 8	x_1				

表11−逆置換による解

【０１１３】

ステップ１においては、ヘッド・プロセッサ素子３２２で、r_{44}の逆数が計算される。この逆数もｒｎｄ１で示す。ステップ２においては、ヘッド・プロセッサ素子は、r_{44}の逆数に、ｓｑｒ２で示す上記プロセッサ素子からその右に送られた、y_4の数値を掛けることにより、x_4の解を計算する。ステップ３においては、ヘッド・プロセッサ素子は、r_{33}の逆数を計算する。その右（ｓｑｒ２）の通常のプロセッサ素子は、その左からそこに送られてきたx_4と、そのレジスタ・ファイルに記憶しているr_{34}とを掛ける。得られる

積が、右からそこへ送られてきたy_3の数値から引かれる。ステップ４においては、ヘッド・プロセッサ素子は、r_{33}の逆数に、右からそこへ送られてきた$y_3 - r_{34}x_4$を、掛けることによりx_3を計算する。ｓｑｒ３で示す第三のプロセッサ素子は、同様に、$y_2 - r_{24}x_4$を計算する。図に示すように、このプロセスは、x_1の解が得らるステップ８まで継続して行われる。乗算が行われるすべての場所は太字で示す。ステップ５において、プロセッサ素子、ｓｑｒ２が行う計算の中で、必要な乗算は$r_{23}x_3$だけであることに留意されたい。残りは右からそこへ送られる。これらステップの待ち時間は、図に示していない。実際には、ある種の演算は前もって行うことができる。例えば、r_{33}の逆数の計算は、逆数r_{44}の計算がスタートした後、一回のサブステップ時間が経過した後で、スタートすることができる。この特定のアルゴリズム用のパイプラインの充填効率は低いことに留意されたい。

【０１１４】

＜マトリックス反転＞

すでに説明したように、ＱＲ分解を行った後で、共通のＲを共有する多重逆置換を行うことにより、行列を反転することができる。この節においては、ただ一回の逆置換を行う間の、全待ち時間を少し長くするだけで、パイプライン処理素子の線形アレー上で、複数の逆置換を可能にする方法について説明する。

【０１１５】

式（１８）の複数の逆置換問題は、共通の行列Ｒを共有しているので、Ｒの主な対角線方向の要素だけの逆数をｊのすべての数値に対して一回計算しさえすればよい。

【表１２】

表12－反転前のレジスタの記憶

rnd1	sqr2	sqr3	sqr4	sqr5	sqr6	sqr7	sqr8
r_{11}	r_{12}	r_{13}	r_{14}	y_{11}	y_{12}	y_{13}	y_{14}
r_{22}	r_{23}	r_{24}	y_{21}	y_{22}	y_{23}	y_{24}	
r_{33}	r_{34}	y_{31}	y_{32}	y_{33}	y_{34}		
r_{44}	y_{41}	y_{42}	y_{43}	y_{44}			

【０１１６】

テーブル１２は、ＱＲ分解の終了時に、プロセッサ素子の８素子線形アレーのレジスタ・アレーに記憶しているものを示す。レジスタ・ファイルの内容は、式（１８）の素子である。

【表１３】

表１３－マトリックス反転の例

	rnd1	sqr2	sqr3	sqr4	sqr5	sqr6	sqr7	sqr8
Step 1/1	$1/r_{44}$							
Step 1/2	$1/r_{33}$							
Step	$1/r_{22}$							

1/3								
Step 1/4	$1/r_{11}$							
Step 1/5								
Step 2/1	x_{41}							
Step 2/2		x_{42}						
Step 2/3			x_{43}					
Step 2/4				x_{44}				
Step 2/5								
Step 3/1		y_{31} - $r_{34}x_{41}$						
Step			y_{32} -					

3/2			$r_{34}x_{42}$					
Step 3/3				y_{33} - $r_{34}x_{43}$				
Step 3/4					y_{34} - $r_{34}x_{44}$			
Step 3/5								
Step 4/1	x_{31}		y_{21} - $r_{24}x_{41}$					
Step 4/2		x_{32}		y_{22} - $r_{24}x_{42}$				
Step 4/3			x_{33}		y_{23} - $r_{24}x_{43}$			
Step 4/4				x_{34}		y_{24} - $r_{24}x_{44}$		
Step 4/5								
Step		y_{21} -		y_{11} -				

5/1		$r_{24}x_{41}$ - $r_{23}x_{31}$		$r_{14}x_{41}$				
Step 5/2			y_{22} - $r_{24}x_{42}$ - $r_{23}x_{32}$		y_{12} - $r_{14}x_{42}$			
Step 5/3				y_{23} - $r_{24}x_{43}$ - $r_{23}x_{33}$		y_{13} - $r_{14}x_{43}$		
Step 5/4					y_{24} - $r_{24}x_{44}$ - $r_{23}x_{34}$		y_{14} - $r_{14}x_{44}$	
Step 5/5								
Step 6/1	x_{21}		y_{11} - $r_{14}x_{41}$ - $r_{13}x_{31}$					
Step 6/2		x_{22}		y_{12} - $r_{14}x_{42}$ - $r_{13}x_{32}$				
Step			x_{23}		y_{13} -			

6/3					$r_{14}x_{43}$ - $r_{13}x_{33}$			
Step 6/4				x_{24}		y_{14} - $r_{14}x_{44}$ - $r_{13}x_{34}$		
Step 6/5								
Step 7/1		y_{11} - $r_{14}x_{41}$ - $r_{13}x_{31}$ - $r_{12}x_{21}$						
Step 7/2			y_{12} - $r_{14}x_{42}$ - $r_{13}x_{32}$ - $r_{12}x_{22}$					
Step 7/3				y_{13} - $r_{14}x_{43}$ - $r_{13}x_{33}$ - $r_{12}x_{23}$				
Step 7/4					y_{14} - $r_{14}x_{44}$ -			

					$r_{13}x_{34}$ - $r_{12}x_{24}$			
Step 7/5								
Step 8/1	x_{11}							
Step 8/2		x_{12}						
Step 8/3			x_{13}					
Step 8/4				x_{14}				

【０１１７】
テーブル１３は、本発明の８素子線形アレーが、行列を反転する方法を示す。このテーブルは、基本的には、アレーの未使用のパイプライン段にインターリーブした、テーブル１１の例の四つのコピーを示す。図７の例示としての実施形態の場合には、１回のサイクルの間に、二つまでの複素数を右から左に移動させることができ、他の二つまでの複素数を左から右に移動させることができる。テーブル１３の反転例は、１サイクル当りの一つのプロセッサ素子当り、せいぜい一回の複素数乗算しか行わないが、一方、ハードウェアは、一つのプロセッサ素子当りの１サイクル当り、一回以上の乗算を行うことができる。さらに、多くの利用可能な多くの追加の未使用のサブステップが、パイプライン内に存在する。当業者であれば、実際に、マトリックス反転の解を見つけだすために、その内部で、Ｒの逆数および対角線方向に位置しない素子が右に移動し、部分的な結果が左に移動する、多くの可能な命令の組およびプログラムの解が存在することを理解することができるだろう。四つの逆置換を行うための全待ち時間が、一回だけ逆置換を行うための待ち時間より、８％しか長くないこと、およびアレーのパイプラインは、多くの残りのアイドルなプロセッサ素子について、まだ少ししか使用されていないことに留意されたい。

【０１１８】
＜マトリックス－マトリックス乗算＞
この節においては、図１５の例示としての線形アレー２３０を使用して、第三の複素数行列、ＡＢ≡Ｃを形成するために、８×８複素数行列を掛け算する方法を説明する。使用する技術の場合、計算は、ブロックまたは部分行列に分割される。図２０は、１６の２×２の積の部分行列に分割された、８×８複素数積行列Ｃを示す。この図においては、c_{35}、c_{36}、c_{45}およびc_{46}を示す、一つの部分行列を強調してある。以下に示す例の場合には、図１５の四つの素子アレー２３０の各プロセッサ素子は、図２０の四つの列のうちの一つを計算するために専用に使用される。この場合、左から三番目のプロセッサ素子により計算される部分行列を強調してある。図２０の四つの行のうちの一つの行全体が、アレー

の四つのプロセッサ素子により同時に計算される。

【０１１９】

図２０の一つの行を処理している間に、アレー２３０は、アレーを通してＡの行の数値を循環させる。一方、Ｂの行は指定のプロセッサ素子に留まる。図２１は、マトリックスーマトリックス乗算の一つのステップを示す。Ａの行１および行２上の複素数の数値のペアは、各プロセッサ素子の、ｉｌｃおよびｉｌｓ入力（図７参照）上で、入手することができる。これらのペアの数値には、各プロセッサ素子のＢの二つの列からの一組の複素数が掛けられる。図２２は、複数の組の数値が、次のプロセッサ素子の方向へ時計方向にシフトされ、次の乗算がスタートする次のステップを示す。上記四つのステップが行われた後で、Ａの同じ二つの行上の数値の他の組、すなわち、a_{12}、a_{22}、a_{14}、a_{24}、a_{16}、a_{26}、a_{18}、a_{28}がロードされ、さらに四つのステップに対してプロセスが続行される。この時点において、Ｃの行１および２のすべての結果を生成するために、必要な、すべての乗算がスタートする。その後で、同じプロセスが、行３および４に対して再びスタートする。

【表１４】

命令ニーモニック	説　明
	ｒ１，ｒ２からｇレジスタおよび左のｃ、ｓをロードする。
	ｒ１からｇを乗算し、累算し、ロードする。
	ｒ１、ｒ２からｇおよびｈを乗算し、累算し、ロードする。
	左のｃ，ｓにｈを乗算し、累算し、ｒ２、ｒ１から 左のｃ、ｓおよびｇをロードする。
	ｒ２、ｒ１から、左およびｇを乗算し、累算し、 ロードする。
	ｒ２、ｒ１から、左およびｇを乗算し、累算し、 ロードする。
	ｃ、ｓおよびｇを乗算し、ｒ１からｇをロードする。
	ｒ１、ｒ２を使用して結果を書き込む。

表１４－マトリックスーマトリックス乗算に対する命令

【０１２０】

テーブル１４は、マトリックスーマトリックス乗算を行うために、テーブル８にすでに示した命令の組に追加できる、いくつかの可能な命令を示す。テーブル１５内は、第三の複素数行列、ＡＢ≡Ｃを形成する目的で、二つの８×８複素数行列を乗算するために、図１５の線形アレー２３０を制御するためのプログラムを示す。このプログラムの冒頭のところに、ＡおよびＢの最初の二つの列が、左側において、第一のプロセッサ素子に記憶されている。第三および第四の列は、第二のプロセッサ素子等にすでに記憶されている。すでに形成済みの積Ｃの最初の二つの列は、第一のプロセッサ素子に保持され、第三および第四の列は第二のプロセッサ素子等の内部に保持去れる。テーブル１５のプログラム・リストにおいては、第一の四つの列は、使用する命令ニーモニックを含む。四つの連続しているラインは、四つのプロセッサ素子を、それぞれ、制御するための個々の命令を示す。ニーモニックの右側の列の内部には、レジスタ・ファイルに対する二つの各ポートの制御を示すフィールドに対する数値を示す。フィールド名、ｄｒｏｗ、ｘおよびｃは、テーブル

６の制御に対応する。

【表１５】

表15- マトリックスーマトリックス乗算用のプログラム

```
# Program for multiplication of two complex 8x8 matrices on a four element
# array.  The two matrices to be multiplied are initialized in the register
# files before the beginning of this program. The results are left in the
# register file.  Text below to the right of column 18 are comments.
#
#    [ r1 ] [ r2 ]
#    dd x c dd x c
#    rr     rr
#    oo     oo
#    ww     ww    <- Everything between the arrows are comments              ->
lgl1,04,0,0,00,1,0 g=b11,b12 il=a11,a21
lgl1,05,0,0,00,1,0 g=b33,b34 il=a13,a23
lgl1,06,0,0,00,1,0 g=b55,b56 il=a15,a25
lgl1,07,0,0,00,1,0 g=b77,b78 il=a17,a27
1/1---------------
mcg1,07,0,0,00,0,0 g=b71,b72                              c11 =a11b11 c12,c21,c22
mcg1,04,0,0,00,0,0 g=b13,b14                              c13 =a13b33 c14,c23,c24
mcg1,05,0,0,00,0,0 g=b35,b36                              c15 =a15b55 c16,c25,c26
mcg1,06,0,0,00,0,0 g=b57,b58                              c17 =a17b77 c18,c27,c28
1/2---------------
mag1,06,0,0,00,0,0 g=b51,b52                              c11+=a17b71 c12,c21,c22
mag1,07,0,0,00,0,0                                        c13+=a11b13 c14,c23,c24
mag1,04,0,0,00,0,0                                        c15+=a13b35 c16,c25,c26
mag1,05,0,0,00,0,0                                        c17+=a15b57 c18,c27,c28
1/3---------------
mag1,05,0,0,00,0,0 g=b31,b32                              c11+=a15b51 c12,c21,c22
mag1,06,0,0,00,0,0
mag1,07,0,0,00,0,0
mag1,04,0,0,00,0,0
1/4---------------
mal1,04,0,1,00,1,1 g=b21,b22 il=a12,a22                   c11+=a13b31 c12,c21,c22
mal1,05,0,1,00,1,1
mal1,06,0,1,00,1,1
mal1,07,0,1,00,1,1
1/5---------------
mag1,07,0,1,00,0,0 g=b81,b82                              c11+=a12b21 c12,c21,c22
```

```
mag1,04,0,1,00,0,0
mag1,05,0,1,00,0,0
mag1,06,0,1,00,0,0
2/1---------------
mag1,06,0,1,00,0,0 g=b61,b62                    c11+=a18b81 c12,c21,c22
mag1,07,0,1,00,0,0
mag1,04,0,1,00,0,0
mag1,05,0,1,00,0,0
2/2---------------
mag1,05,0,1,00,0,0 g=b41,b42                    c11+=a16b61 c12,c21,c22
mag1,06,0,1,00,0,0
mag1,07,0,1,00,0,0
mag1,04,0,1,00,0,0
2/3---------------
mal1,04,0,0,01,1,0 g=b11,b12 il=a31,a41         c11+=a14b41 c12,c21,c22
mal1,05,0,0,01,1,0
mal1,06,0,0,01,1,0
mal1,07,0,0,01,1,0
2/4---------------
mcg1,07,0,0,00,0,0 g=b71,b72                    c31 =a31b11 c32,c41,c42
mcg1,04,0,0,00,0,0
mcg1,05,0,0,00,0,0
mcg1,06,0,0,00,0,0
2/5---------------
mag1,06,0,0,00,0,0 g=b51,b52                    c31+=a37b71 c32,c41,c42
mag1,07,0,0,00,0,0
mag1,04,0,0,00,0,0
mag1,05,0,0,00,0,0
3/1---------------
magh,05,0,0,07,0,1 g=b31,b32 ga=b81,b82         c31+=a35b51 c32,c41,c42
magh,06,0,0,04,0,1
magh,07,0,0,05,0,1
magh,04,0,0,06,0,1
3/2---------------
mal1,04,0,1,01,1,1 g=b21,b22 il=a32,a42         c31+=a33b31 c32,c41,c42
mal1,05,0,1,01,1,1
mal1,06,0,1,01,1,1
mal1,07,0,1,01,1,1
3/3---------------
maw1,08,0,0,08,0,1 r1<=c11,12 r2<=c21,22        c31+=a32b21 c32,c41,c42
maw1,08,0,0,08,0,1 r1<=c13,14 r2<=c23,24
```

```
mawl,08,0,0,08,0,1 r1<=c15,16 r2<=c25,26
mawl,08,0,0,08,0,1 r1<=c17,18 r2<=c27,28
3/4---------------
mahl,06,0,1,00,0,0 g=b61,b62                    c31+=a38b81 c32,c41,c42
mahl,07,0,1,00,0,0
mahl,04,0,1,00,0,0
mahl,05,0,1,00,0,0
3/5---------------
magl,05,0,1,00,0,0 g=b41,b42                    c31+=a36b61 c32,c41,c42
magl,06,0,1,00,0,0
magl,07,0,1,00,0,0
magl,04,0,1,00,0,0
4/1---------------
mall,04,0,0,02,1,0 g=b11,b12 il=a51,a61         c31+=a34b41 c32,c41,c42
mall,05,0,0,02,1,0
mall,06,0,0,02,1,0
mall,07,0,0,02,1,0
4/2---------------
mcgl,07,0,0,00,0,0 g=b71,b72                    c51 =a51b11 c52,c61,c62
mcgl,04,0,0,00,0,0
mcgl,05,0,0,00,0,0
mcgl,06,0,0,00,0,0
4/3---------------
magl,06,0,0,00,0,0 g=b51,b52                    c51+=a57b71 c52,c61,c62
magl,07,0,0,00,0,0
magl,04,0,0,00,0,0
magl,05,0,0,00,0,0
4/4---------------
magl,05,0,0,00,0,0 g=b31,b32                    c51+=a55b51 c52,c61,c62
magl,06,0,0,00,0,0
magl,07,0,0,00,0,0
magl,04,0,0,00,0,0
4/5---------------
mall,04,0,1,02,1,1 g=b21,b22 il=a52,a62         c51+=a53b31 c52,c61,c62
mall,05,0,1,02,1,1
mall,06,0,1,02,1,1
mall,07,0,1,02,1,1
5/1---------------
mawl,09,0,0,09,0,1 r1<=c31,32 r2<=c41,42        c51+=a52b21 c52,c61,c62
mawl,09,0,0,09,0,1 r1<=c33,34 r2<=c43,44
mawl,09,0,0,09,0,1 r1<=c35,36 r2<=c45,46
```

```
maw1,09,0,0,09,0,1 r1<=c37,38 r2<=c47,48
5/2---------------
mah1,06,0,1,00,0,0 g=b61,b62                    c51+=a58b81 c52,c61,c62
mah1,07,0,1,00,0,0
mah1,04,0,1,00,0,0
mah1,05,0,1,00,0,0
5/3---------------
mag1,05,0,1,00,0,0 g=b41,b42                    c51+=a56b61 c52,c61,c62
mag1,06,0,1,00,0,0
mag1,07,0,1,00,0,0
mag1,04,0,1,00,0,0
5/4---------------
mal1,04,0,0,03,1,0 g=b11,b12 il=a71,a81         c51+=a54b41 c52,c61,c62
mal1,05,0,0,03,1,0
mal1,06,0,0,03,1,0
mal1,07,0,0,03,1,0
5/5---------------
mcg1,07,0,0,00,0,0 g=b71,b72                    c71 =a71b11 c72,c81,c82
mcg1,04,0,0,00,0,0
mcg1,05,0,0,00,0,0
mcg1,06,0,0,00,0,0
6/1---------------
mag1,06,0,0,00,0,0 g=b51,b52                    c71+=a77b71 c72,c81,c82
mag1,07,0,0,00,0,0
mag1,04,0,0,00,0,0
mag1,05,0,0,00,0,0
6/2---------------
mag1,05,0,0,00,0,0 g=b31,b32                    c71+=a75b51 c72,c81,c82
mag1,06,0,0,00,0,0
mag1,07,0,0,00,0,0
mag1,04,0,0,00,0,0
6/3---------------
mal1,04,0,1,03,1,1 g=b21,b22 il=a72,a82         c71+=a73b31 c72,c81,c82
mal1,05,0,1,03,1,1
mal1,06,0,1,03,1,1
mal1,07,0,1,03,1,1
6/4---------------
maw1,10,0,0,10,0,1 r1<=c51,52 r2<=c61,62        c71+=a72b21 c72,c81,c82
maw1,10,0,0,10,0,1 r1<=c53,54 r2<=c63,64
maw1,10,0,0,10,0,1 r1<=c55,56 r2<=c65,66
maw1,10,0,0,10,0,1 r1<=c57,58 r2<=c67,68
```

```
6/5---------------
mah1,06,0,1,00,0,0 g=b61,b62                    c71+=a78b81 c72,c81,c82
mah1,07,0,1,00,0,0
mah1,04,0,1,00,0,0
mah1,05,0,1,00,0,0
7/1---------------
mag1,05,0,1,00,0,0 g=b41,b42                    c71+=a76b61 c72,c81,c82
mag1,06,0,1,00,0,0
mag1,07,0,1,00,0,0
mag1,04,0,1,00,0,0
7/2---------------
mal1,04,0,0,00,0,0 g=b11,b12                    c71+=a74b41 c72,c81,c82
mal1,05,0,0,00,0,0
mal1,06,0,0,00,0,0
mal1,07,0,0,00,0,0
7/3---------------
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
7/4---------------
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
7/5---------------
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
8/1---------------
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
8/2---------------
zsw1,11,0,0,11,0,1 r1<=c71,72 r2<=c81,82
zsw1,11,0,0,11,0,1 r1<=c73,74 r2<=c83,84
zsw1,11,0,0,11,0,1 r1<=c75,76 r2<=c85,96
zsw1,11,0,0,11,0,1 r1<=c77,78 r2<=c87,88
8/3---------------
```

```
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
8/4--------------
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
noop,00,0,0,00,0,0
8/5--------------
```

【０１２１】
テーブル１５のプログラムは、初期値を含む各プロセッサ素子の二つのｇレジスタ、および二つの入力左レジスタをロードすることによりスタートする。次のサブステップは、四つのクロス乗積の乗算をスタートし、Ａ値をアレーの周囲での時計方向のシフトをスタートする。次の命令は、累算Ｃクロス乗積に加算するために、乗算－累算命令をスタートする。サブステップ１／４においては、Ａの新しい数値をレジスタ・ファイルから読み出す。サブステップ２／３においては、各プロセッサ素子のＣ内の四つの素子、すなわち、第一プロセッサ素子のｃ１１、ｃ１２、ｃ２１およびｃ２２、第二のプロセッサ素子のｃ１３、ｃ１４、ｃ２３、ｃ２４等の計算を終了するために、最後の乗算－累算がスタートする。パイプラインの待ち時間のために、これらの結果は、サブステップ３／３まで、レジスタ・ファイルに書き戻されない。一方、アレーは、サブステップ２／４において、Ｃの次の組の数値の計算をスタートする。最後の乗算はサブステップ７／２でスタートし、パイプラインの待ち時間経過後、サブステップ８／２において、最終的な結果がレジスタ・ファイルに書き込まれる。それ故、８×８複素数マトリックス－マトリックス乗算は、アレー２３０上で終了するまで、４０サブステップ以下の時間しか掛からない。
【０１２２】
複素数行列に（例えば、７×７のような）奇数の大きさを乗算するために、何時でも次の偶数の大きさに行列をゼロで充填することができ、その後で、８×８乗算のために上記手順を行うことができる。しかし、そうすると、不必要な余分な乗算を行わなければならないために、効率が低下する。当業者であれば、別の方法として、奇数の大きさの行列を処理するために、適当な命令を、テーブル１４の行列に追加することができることをすぐに理解することができるだろう。
【０１２３】
各レジスタ・ファイルにもっと多くの列を記憶することによって、同じ大きさのアレー上で、もっと大きなアレーの乗算を行うことができる。例えば、第一のプロセッサ素子に第一の四つの列を記憶し、第二のプロセッサ素子等の上に列５－８を記憶することによって、図１５の線形アレー２３０上で、１６×１６の複素数マトリックス－マトリックス乗算を行うことができる。アレーのレジスタ・ファイルに入りきらない大きさの、行列の場合には、そのような行列を入れることができる大きさのブロックに分解して、周知のブロック行列乗算方法を使用することができる。この場合には、ホスト・プロセッサ、または第三のレジスタ・ファイル・ポートを使用する専用ロジックを、ブロックの緊密に接続しているアップローディンおよびダウンローディングと協力させなければならない。
【０１２４】
＜共分散および相互相関＞
この節においては、共分散および相互相関を同時に計算するための、図１６の線形アレー２４０の使用方法を説明する。式（２３）および（２４）は、共分散および相互相関を計算するためには、フォームの項、xx^Hおよびxy_dを計算しなければならないことを示す。

【０１２５】
上記のマトリックス－マトリックス乗算と比較すると、共分散の計算はもっと簡単である。二つのＮ×Ｎ行列を乗算する場合には、N^3の乗算を行わなければならない。共分散だけの計算なら、N^2の乗算を行いさえすればよい。下記の例の場合には、入力ベクトルｘ（例えば、アンテナのアレーからのデータ）、および期待値y_dは、図２３に示すように、頂部からアレー２４０内に入り、一番左の四つの各プロセッサ素子、および５番目のプロセッサ素子用のｇ１レジスタ用の、図７のｇ１、ｈ２、ｉｌｃ０（１）およびｉｌｓ０（１）レジスタにロードされる。次のサブステップにおいては、第一の四つの各プロセッサ素子は、前のステップにおいて、それ自身をすでに入力している二つの数値の四つのクロス乗積を計算する。例えば、左側の第一のプロセッサ素子は、四つの項、すなわち、$x_1x_1^*$、$x_1x_2^*$、$x_2x_1^*$および$x_2x_2^*$を計算する。同時に、ｉｌｃ０（１）およびｉｌｓ０（１）レジスタ内の数値が右に送られる。次のサブステップにおいては、第一の四つの各プロセッサ素子は、その左側にあった数値とｇ１、およびｈ２内に保持している数値との四つのクロス乗積を計算する。右側のプロセッサ素子は、二つの積、$x_7y_d^*$および$x_8y_d^*$を計算する。図２４は、第三のサブステップの時計方向の相互接続上のデータである。各乗算によるパイプライン遅延の終わりのところで、完了した項を、レジスタ・ファイルに書き込むことができる。別の方法としては、時間平均を更新するために、レジスタ・ファイルに記憶した前の数値により、完了した項を最初に加重し、平均することができる。その後で、図に示す例に対する共分散および相互相関を数ステップで完了することができる。当業者であれば、共分散および相互相関を計算するために、他の方法でアレーを構成し、プログラムすることができることを、理解することができるだろう。

【０１２６】
＜平方根正規化プロセス＞
図２５、図２６および図２７を参照しながら、図１０のところですでに説明した平方根正規化プロセスについて、さらに詳細に説明する。本発明のこの機能は、ある関数に対するアーギュメントの範囲縮小、および必要な機能を達成するための移行の結果の変換を助けるデジタル・ハードウェアを提供する。説明の都合上、上記関数は下記式で表わされるものと仮定する。
【数１３】

$$y(x)=\frac{1}{\sqrt{x}} \quad . \qquad (26)$$

【０１２７】
図２５は、この逆平方根機能のグラフである。このグラフから、ｘの数値が小さい場合には、逆平方根機能が非常に急勾配になることが分かる。このグラフの陰をつけた部分は、０．５＜ｙ（ｘ）≦１および１≦ｘ＜４の範囲に対応する。陰をつけた部分においては、関数の勾配は緩やかで、テーブル内の数値間に補間するというような周知の技術を使用して、もっと容易に達成することができる。

【０１２８】
図２５の陰をつけた部分に対応する、縮小した範囲内の数値を生成するための正規化プロセスは、識別を使用することができる。
【数１４】

$$\frac{1}{\sqrt{2^{2n}x}}=\frac{1}{2^n\sqrt{x}} \quad . \qquad (27)$$

ｎが整数である場合で、シフトが、ビット２ｎの偶数に制限されている場合には、「偶数」正規化を行うことができ、その場合には、二つの最上位のデータ・ビットは、ロジック１を含む。下記のテーブル１６に多数の例を示す。

【表１６】

正規化前のアーギュメント x	正規化された x	2n
$.11011_b = .84375$	$11.011_b = 3.375$	-2
$101.1_b = 5.5$	$01.011_b = 1.375$	2
$.0001101_b = .1015625$	$01.101_b = 1.625$	-4
$.000010111_b = .044921875$	$10.111_b = 2.875$	-6

表16- xの正規化の例

【０１２９】

テーブル１６の最初の二つの列の「＝」の左側の数値は、固定小数点２進表示であり、「＝」の右側の数値は十進法表示である。

【０１３０】

元のアーギュメントが、０．０４－６の範囲内の合ったことに留意されたい。正規化されたｘの範囲は、１．０≦ｘ＜４である。正規化されたｘの２進小数点の位置は、任意の点である。重要なことは、最小のｘに対する最大のｘの比率であり、正規化後は４に等しい。しかし、１－４の範囲はｘを選択するのに適した領域である。何故なら、この領域においては、関数の勾配が急でないからである。そのため、補間または他の周知の技術により、高い精度を達成することができる。

【０１３１】

正規化したｘが決定されると、整級数展開、テーブル駆動補間法、およびニュートン－ラプソン回帰のような、当業者にとっては周知の技術を使用して、正規化した数値ｘの逆平方根の計算を進めることができる。

【０１３２】

図２６は、上記の「偶数」正規化プロセスの可能なハードウェアによる実行である。正規化を行う前のアーギュメントｘの隣接するビットの各ペアは、図に示すように、１である場合には、他方または両方は、ロジック１の数値を持つことを示すように、一連のＯＲゲート３２５内の、対応する二つの入力ＯＲゲートに入力することができる。図２６の例の場合には、Ｗビットは、２進小数点の左側のすべてのビットの数を示し、この例の場合には偶数で１２である。Ｆビットは、ｘの小数部分のビットの全部の数を示すが、この例の場合には８である。一連のＯＲゲート３２５は、ｘのビットの隣接するペアの方を向いている、１０のＯＲゲートを含む。

【０１３３】

一連のＯＲゲート３２５の出力は、現在のハードウェア正規化装置の周知のタイプの優先順位エンコーダ３２７の入力になる。エンコーダ３２７は、アーギュメントの最上位ビットの場所に近い、ロジック１の出力と一緒に、特定のＯＲゲートに対応する２ｎの数値を出力する。エンコーダ３２７の出力のところの数値は、正規化されたｘを形成するために

、アーギュメントを２ｎだけシフトするたる形シフタ３３０を制御する。
【０１３４】
より詳細に説明すると、優先順位エンコーダ３２７は、左から右の方を向いている、第一のロジック１入力の位置に対応する一つの数を出力する。この例の場合には、最初の１は左から数えて第二のＯＲゲートからのものである。ＯＲゲートからの入力の次に示す０－１８の数字は、優先順位エンコーダ３２７が出力する２ｎの数値である。それ故、この例の場合、２ｎ＝２が出力である。すでに説明したように、数値２ｎは一番左の位置、または一番左の位置の次の位置に、何時でもロジック１を持つ正規化した仮数を形成するために、ｘをシフトするたる形シフタ３３０を制御するために使用される。
【０１３５】
図２６の正規化した仮数のビット部分は、「オフセット」と呼ばれるが、これについては、図１０のところですでに説明した。これらのビットは、例えば、読出し専用メモリ（ＲＯＭ）、または他のタイプのメモリに記憶しているテーブルのような、参照用テーブルから、関数の近似値を発見するための、アドレスとして使用することができる。この例の場合には、オフセットは、３２語を含むテーブルを示す五つのビットを持つ。しかし、０≦ｘ＜１に対応する、これら語の最も下の部分は、決して参照されることはないので、参照用テーブルは２４語を含んでいればそれで十分である。図２６の場合も、オフセットの右側のいくつかのビットには「ｄｘ」がつけられている。これらビットは、ｘがオフセットの数値より大きい量を示す。線形補間アルゴリズムを使用する場合には、参照用テーブルは、ｙ（オフセット）の数値と、オフセットのところのｙ（正規化したｘ）の勾配の両方を含む。正規化したｘのところのｙ（正規化したｘ）の数値を発見するために、ｄｘの数値にはテーブルから発見した勾配が掛けられ、オフセットのところの数値に加算される。
【０１３６】
正規化したｘの逆平方根を計算した後で、結果を、ビットの数の半分だけ、正規化シフトの方向とは反対方向にシフトすることによって、元のアーギュメントの必要な逆平方根を入手することができる。すなわち、正規化の際に、右側へ２ｎ＝６ビットだけシフトさせた場合には、非正規化により、結果は定数＋（ｎ＝３）ビットだけ、左側にシフトされる。それ故、上記識別を行うハードウェアを構成することによって、アーギュメントの範囲を容易に狭くすることができ、また結果を容易に調整することができる。
【０１３７】
再び図２６の例について説明するが、ｘの２進小数点は、左から数えてＷビットであることに留意されたい。参照用テーブルの数値は、テーブル１６に示す左から数えて２ビット目の位置にある２進小数点に基づいている。それ故、非正規化を行うためには、たる形シフタ３３０によるｎビットの可変シフトに加えて、（Ｗビット－２）／２に等しいシフトを何時でも行わなければならない。それ故、上記のｙ（正規化したｘ）の数値は、逆の方向に（ｎ－１－（Ｗビット／２））ビット分だけシフトさせ、その結果、必要なｙ（ｘ）とした、たる形シフタにより非正規化することができる。
【０１３８】
図２７は、ビットの全数が奇数である場合の、正規化プロセスにおける違いを示す。この例の場合には、Ｗビットの数は１１である。全数の数を偶数にするために、余分なゼロがｘの前に追加される。この例の場合には、一組のＯＲゲート３２５’は、図２６の一組のＯＲゲート３２５に対応するが、第一のＯＲゲートは除去されていて、前に追加したビットの右のビットは、優先順位エンコーダ３２７’に直接入力される。
【０１３９】
逆平方根の計算に基づいて、上記正規化技術を説明してきたが、この技術は、平方根の計算にも適用することができる。この技術は、簡単な方法で、立法根等のような他の分数ベキにも適用を広げることができる。例えば、立法根の場合には、３ビットの倍数分等だけシフトするように正規化装置を構成することができる。
【０１４０】
上記の本発明の実施形態は例示としてのものに過ぎない。他の実施形態は、プロセッサ素

子およびその線形アレーに対して他の構成を使用することができ、上記以外の計算も行うことができる。当業者であれば、添付の特許請求の範囲に含まれる、上記および多数の他の実施形態を容易に思いつくことができるだろう。

【図面の簡単な説明】

【図１】ＱＲ分解の例のための一組のギブンズ回転の各段である。

【図２】図１の例のギブンズ回転をプロセッサ素子の従来の二次元三角アレーで実行する方法である。

【図３】図１の例のギブンズ回転をプロセッサ素子の従来の二次元三角アレーで実行する方法である。

【図４】図１の例のギブンズ回転をプロセッサ素子の従来の二次元三角アレーで実行する方法である。

【図５】図１の例のギブンズ回転をプロセッサ素子の従来の二次元三角アレーで実行する方法である。

【図６】プロセッサ素子の二次元三角アレー上で実行される、従来の並列および収縮処理アプローチの比較である。

【図７】本発明の例示としての実施形態のプロセッサ素子である。

【図８】図７のプロセッサ素子の乗算装置の詳細図でである。

【図９】図７のプロセッサ素子で使用するのに適している、非線形関数ゼネレータである。

【図１０】図９の非線形関数ゼネレータで使用するのに適している逆平方根ゼネレータの一実施形態である。

【図１１】図９の非線形関数ゼネレータのパイプライン処理の実行である。

【図１２】図７のプロセッサ素子のレジスタ・ファイルの詳細図である。

【図１３】図１２のレジスタ・ファイルのポートに関連するクロスバー・スイッチである。

【図１４】本発明のプロセッサ素子の線形アレーのいくつかの例である。

【図１５】本発明のプロセッサ素子の線形アレーのいくつかの例である。

【図１６】本発明のプロセッサ素子の線形アレーのいくつかの例である。

【図１７】本発明のプロセッサ素子の線形アレーのいくつかの例である。

【図１８】ホスト・プロセッサと通信するコプロセッサ内のプロセッサ素子の線形アレーの一実施形態である。

【図１９】例示としての逆置換操作を実行するための本発明の線形アレーの一例である。

【図２０】本発明の線形アレー上で、マトリックス－マトリックス乗算を容易に行うために、その内部で、マトリックスを部分行列に分割するのを可能にする方法である。

【図２１】図１５の線形アレーを使用する、マトリックス－マトリックス乗算の性能である。

【図２２】図１５の線形アレーを使用する、マトリックス－マトリックス乗算の性能である。

【図２３】図１６の線形アレーを使用する共分散および相互相関の計算である。

【図２４】図１６の線形アレーを使用する共分散および相互相関の計算である。

【図２５】本発明の、平方根正規化プロセスを使用して、実行することができる逆平方根範囲縮小のグラフである。

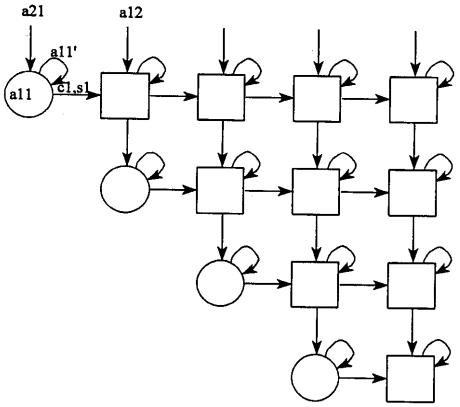
【図２６】本発明の正規化ハードウェアの一例である。

【図２７】本発明の正規化ハードウェアの一例である。

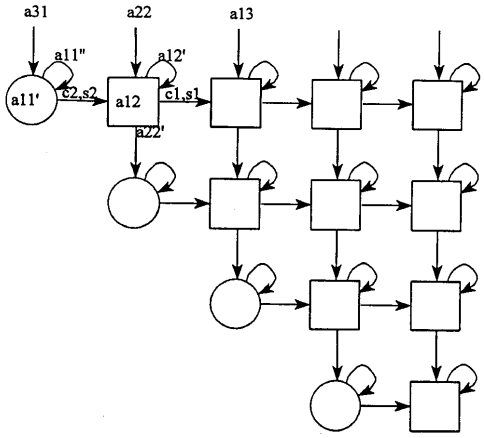
【図１】

1	→ 2	→ 3	→ 4	→ 5	→ 6
aaaay	aaaay	aaaay	aaaay	aaaay	aaaay
xxxxx	xxxxx	xxxxx	xxxxx	xxxxx	xxxxx
0xxxx	0xxxx	0xxxx	0xxxx	0xxxx	0xxxx
xxxxx	0xxxx	0xxxx	00xxx	00xxx	00xxx
xxxxx	xxxxx	0xxxx	0xxxx	00xxx	000xx

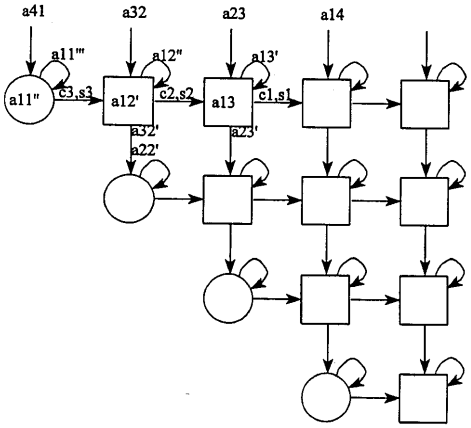
【図２】



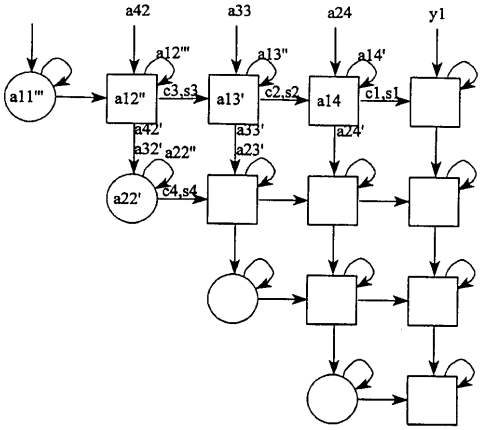
【図３】



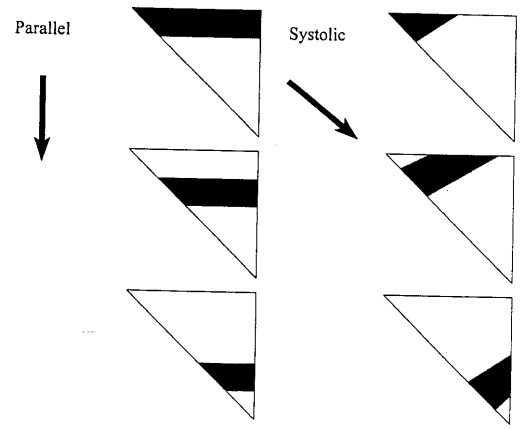
【図４】



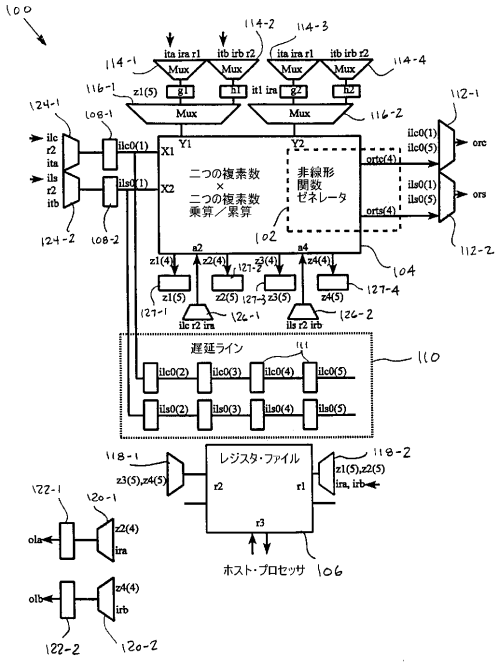
【図５】



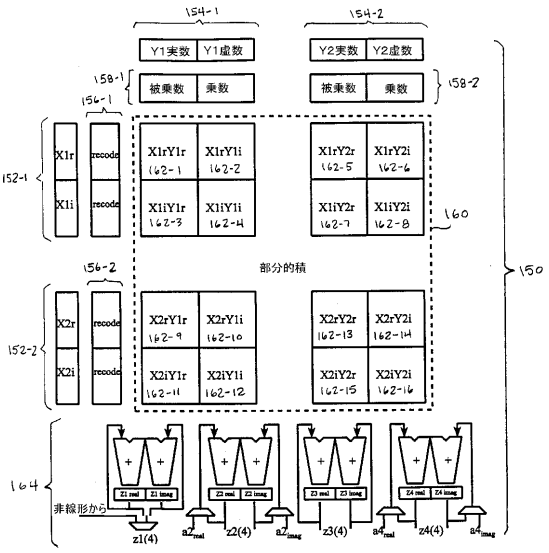
【図６】


Parallel
Systolic

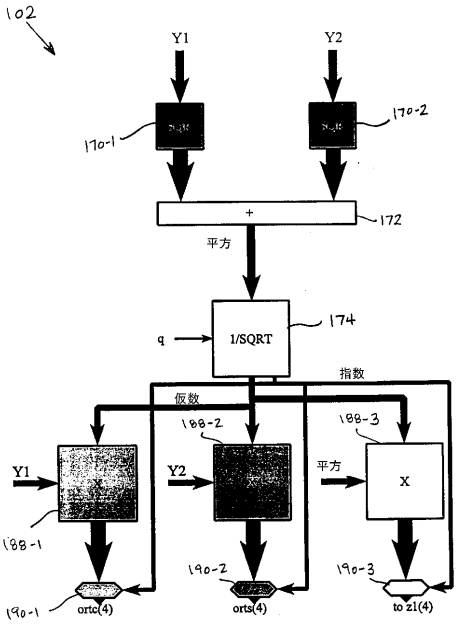
【図７】


二つの複素数 × 二つの複素数 乗算／累算
非線形関数ゼネレータ
遅延ライン
レジスタ・ファイル
ホスト・プロセッサ

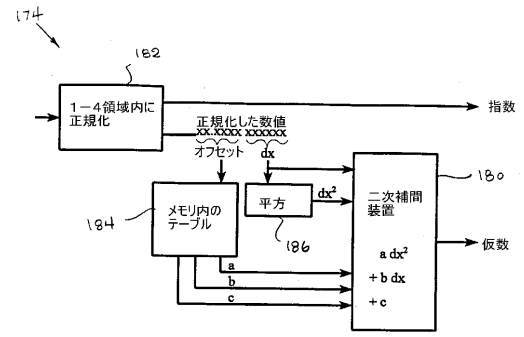
【図８】


Y1実数
Y1虚数
Y2実数
Y2虚数
被乗数
乗数
部分的積
非線形から

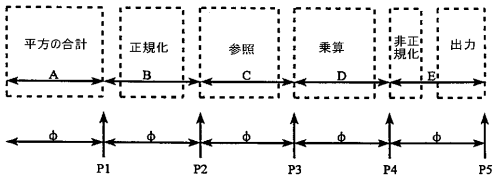
【図９】


Y1
Y2
平方
1/SQRT
指数
仮数
X
to z1(4)

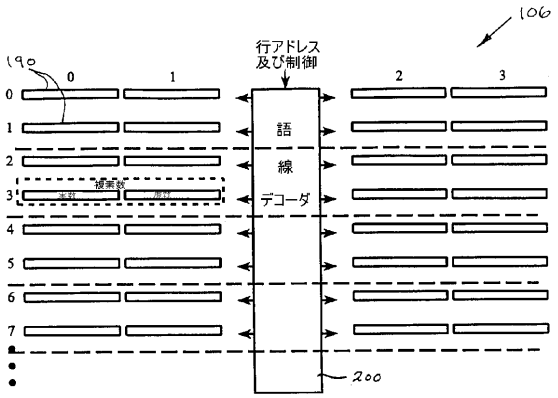
【図１０】



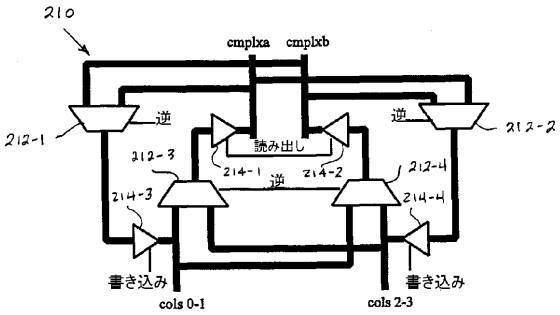
【図１１】



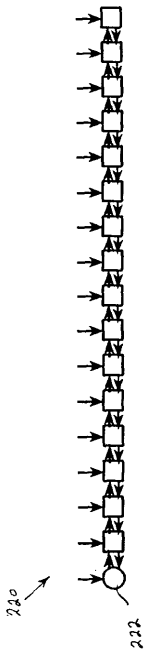
【図１２】



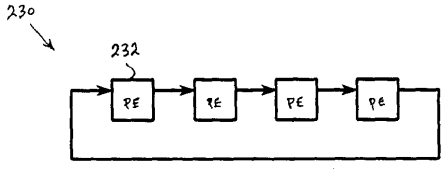
【図１３】



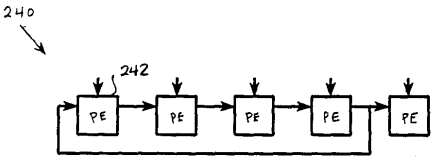
【図１４】



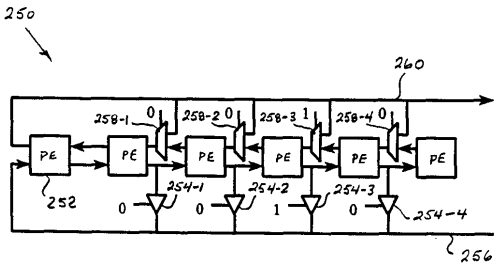
【図１５】



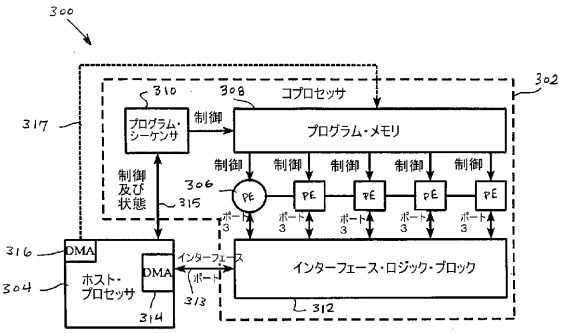
【図１６】



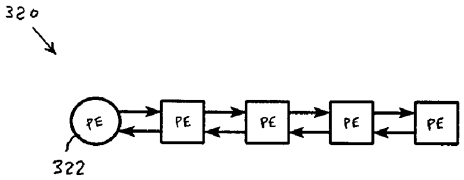
【図１７】



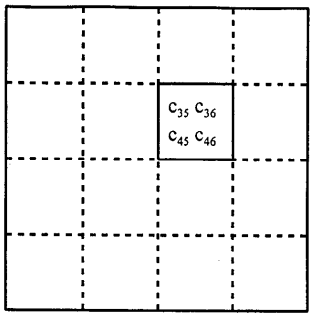
【図１８】



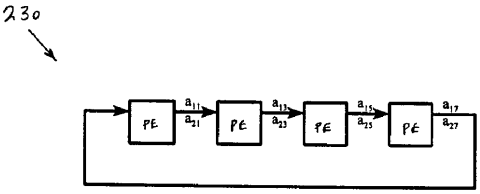
【図１９】



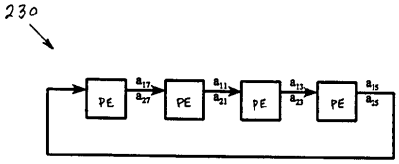
【図２０】



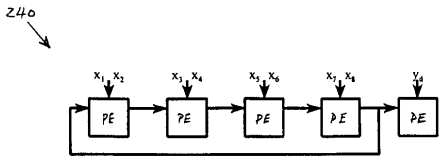
【図２１】



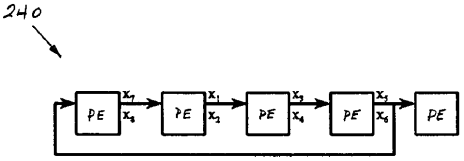
【図２２】



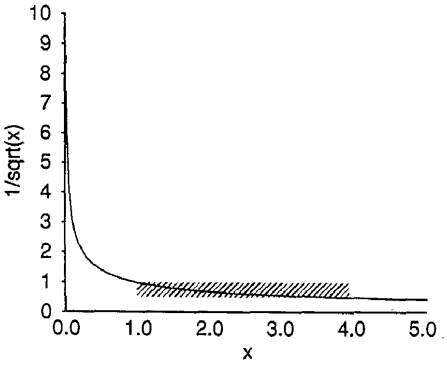
【図２３】



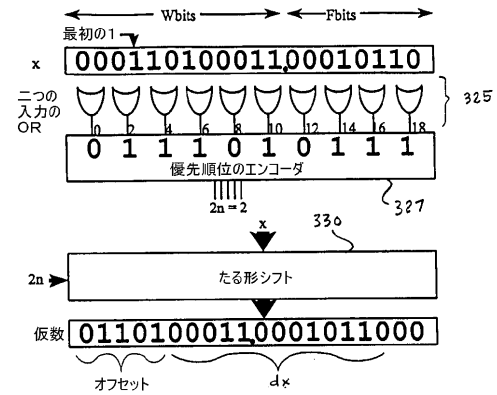
【図２４】



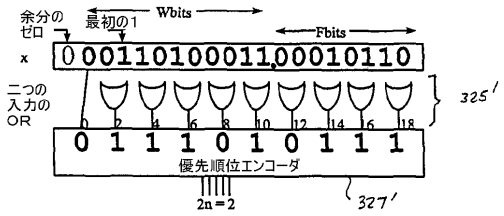
【図２５】



【図２６】



【図２７】



フロントページの続き

(74)代理人 100104352
弁理士 朝日 伸光
(72)発明者 アラン ジョエル グリーンバーガー
アメリカ合衆国 １８１０４ ペンシルヴァニア，サウス ホワイトホール タウンシップ，エヌ ３３ ストリート １１３３

合議体
審判長 赤川 誠一
審判官 久保 光宏
審判官 鈴木 匡明

(56)参考文献 国際公開第９９／００７４６（WO，Ａ１）
特開平３－７１３３１（ＪＰ，Ａ）
特開平５－１５１２４９（ＪＰ，Ａ）
特開平１１－９７９８２（ＪＰ，Ａ）
特開昭６０－１２７５９８（ＪＰ，Ａ）
特開平８－３２２２９９（ＪＰ，Ａ）
北原直紀・他、「ＣＯＲＤＩＣプロセッサを用いた超複素数乗算器」、１９９８年電子情報通信学会総合大会講演論文集、分冊：基礎・境界、社団法人電子情報通信学会、１９９８年3月6日、第１３５頁
渋沢進、「時空間適応処理のための並列ＱＲ分解の方法について」、電子情報通信学会技術研究報告、第９７巻、第３７５号（ＣＯＭＰ９７－６０～７２）、社団法人電子情報通信学会、１９９７年１１月１４日、第１７～２４頁
酒井英昭・他、「適応フィルタとＶＬＳＩ実現」、システム／制御／情報（システム制御情報学会誌）、第３８巻、第8号、システム制御情報学会、１９９４年8月１５日、第1～6頁，ＩＳＳＮ：０９１６－１６００
電子情報通信学会編、「ディジタル信号処理ハンドブック」、初版、株式会社オーム社、１９９３年1月３１日、第２８０～２８４頁、ＩＳＢＮ：４－２７４－０３４２３－２

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F17/10-17/18
G06F7/00-7/78
G06F1/02
G11C11/34-11/56
JSTPlus(JDream2)
CSDB(日本国特許庁)