

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7343473号
(P7343473)

(45)発行日 令和5年9月12日(2023.9.12)

(24)登録日 令和5年9月4日(2023.9.4)

(51)国際特許分類

G 0 6 F 17/16 (2006.01)

F I

G 0 6 F 17/16

N

請求項の数 24 (全28頁)

(21)出願番号	特願2020-501122(P2020-501122)	(73)特許権者	500395107
(86)(22)出願日	平成30年7月2日(2018.7.2)		アーム・リミテッド
(65)公表番号	特表2020-527795(P2020-527795 A)		グレート・ブリテン及び北部アイルランド連合王国、イングランド、シー・ビー1、9エヌ・ジェイ、ケンブリッジ、フルボーン・ロード 110
(43)公表日	令和2年9月10日(2020.9.10)	(74)代理人	110000855
(86)国際出願番号	PCT/GB2018/051851		弁理士法人浅村特許事務所
(87)国際公開番号	WO2019/016507	(72)発明者	ビスコンディ、エリク
(87)国際公開日	平成31年1月24日(2019.1.24)		フランス共和国、ソフィア アンティボリス セデックス、ペーパー 70124
審査請求日	令和3年6月25日(2021.6.25)		、ル パロス、アレ ピエール ツィラー 25、エイアールエム フランス エスア
(31)優先権主張番号	1711700.3		エス 気付
(32)優先日	平成29年7月20日(2017.7.20)	(72)発明者	エヨール、ムボウ
(33)優先権主張国・地域又は機関	英国(GB)		最終頁に続く

(54)【発明の名称】 レジスタベースの複素数処理

(57)【特許請求の範囲】

【請求項1】

命令をデコードするための命令デコーダ回路と、
入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む前記入力データ・ベクトルへの前記命令によって指定されるベクトル処理演算を選択的に適用するためのデータ処理回路と
を備え、
前記命令デコーダ回路は複素数演算、第1のソース・レジスタ、第2のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に応答して、前記データ処理回路が、
前記第1のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、前記第1のソース・レジスタ内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、
前記第2のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、前記第2のソース・レジスタ内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、
前記データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、前記複素数成分の結果セットは、前記複素数の第1のセット及び前記複素数の第2のセットに適用された前記複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、

前記複素数成分の結果セットを前記デスティネーション・レジスタに適用することとを行うように制御するための制御信号を生成し、

前記複素数処理命令は、前記第1のソース・レジスタ及び前記第2のソース・レジスタのうちの少なくとも1つについての複製インデックスをさらに指定し、前記データ処理回路はさらに前記制御信号に応答して、前記第1のソース・レジスタ及び前記第2のソース・レジスタのうちの前記少なくとも1つにおける前記複製インデックスによって示される複素数のソース・セットを複製する、装置。

【請求項2】

前記複素数成分の結果セットは前記複素数結果の実数成分のセットである、請求項1に記載の装置。

【請求項3】

前記複素数成分の結果セットは前記複素数結果の虚数成分のセットである、請求項1に記載の装置。

【請求項4】

前記データ処理回路は前記制御信号に応答して前記複素数成分の結果セットを前記デスティネーション・レジスタ内の連続位置に適用する、請求項1から3までのいずれかに記載の装置。

【請求項5】

前記複素数成分の結果セットのそれぞれの複素数成分は、前記データ項目の第1及び第2のセットのそれぞれのデータ項目よりビット幅が拡大している、請求項4に記載の装置。

【請求項6】

前記データ処理回路は前記制御信号に応答して、前記複素数成分の結果セットを、複素数成分のさらなる結果セットでインターリーブされる前記デスティネーション・レジスタ内の交互する位置に適用し、

前記複素数成分の結果セットが前記複素数結果の実数成分のセットである場合、前記複素数成分のさらなる結果セットは前記複素数結果の虚数成分のセットであり、

前記複素数成分の結果セットが前記複素数結果の虚数成分のセットである場合、前記複素数成分のさらなる結果セットは前記複素数結果の実数成分のセットである、請求項1から3までのいずれかに記載の装置。

【請求項7】

前記複素数処理命令は、前記データ項目の第1のセットが前記第1のソース・レジスタの奇数番号位置又は偶数番号位置から抽出されるかどうかをさらに指定し、前記複素数処理命令は、前記データ項目の第2のセットが前記第2のソース・レジスタの奇数番号位置又は偶数番号位置から抽出されるかどうかをさらに指定する、請求項1から6までのいずれかに記載の装置。

【請求項8】

前記複素数のソース・セットの複製は、前記複素数のソース・セットの複数回の複製を含む、請求項7に記載の装置。

【請求項9】

前記第1のソース・レジスタから抽出される前記データ項目の少なくとも1つの第1のセットは、前記複素数の第1のセットの実数および虚数成分を含み、

前記データ処理回路は前記制御信号にさらに応答して、前記実数および虚数成分をオペランドとして使用して、前記複素数成分の結果セットを生成する、請求項1から8までのいずれかに記載の装置。

【請求項10】

前記第2のソース・レジスタから抽出される前記データ項目の少なくとも1つの第2のセットは、前記複素数の第2のセットの実数および虚数成分を含み、

前記データ処理回路は前記制御信号にさらに応答して、前記複素数の第2のセットの実数および虚数成分をオペランドとして使用して、前記複素数成分の結果セットを生成する、請求項1から9までのいずれかに記載の装置。

10

20

30

40

50

【請求項 1 1】

前記複素数演算は、複素数ドット積演算であり、前記データ処理回路は前記制御信号に
応答して、前記複素数の第 1 のセットの実数成分－虚数成分対の対のセット、並びに前記
複素数の第 2 のセットの実数成分－虚数成分対の対のセット、の部分積から前記複素数成
分の結果セットを累算する累算回路を備える、請求項 9 又は 1 0 に記載の装置。

【請求項 1 2】

前記複素数演算は複素数乗算演算を含む、請求項 1 から 1 0 までのいずれかに記載の装
置。

【請求項 1 3】

前記複素数演算は複素数加算演算、又は複素数減算演算である、請求項 1 から 1 0 まで
のいずれかに記載の装置。

10

【請求項 1 4】

前記複素数処理命令がさらに転置パラメータを指定し、前記データ処理回路が前記制御
信号に応答して、前記複素数の第 1 及び第 2 のセットへの前記複素数演算の適用に先立っ
て前記複素数の第 1 及び第 2 のセットのうちの少なくとも 1 つに対して転置演算を適用す
る、請求項 1 から 1 3 までのいずれかに記載の装置。

【請求項 1 5】

前記転置演算は、

- 少なくとも 1 つの前記複素数の第 1 のセット及び前記複素数の第 2 のセットにおいて、
実数成分の虚数成分での置換、
- 虚数成分の実数成分での置換、
- 実数成分と虚数成分の交換、
- 実数成分の符号を反転させること、及び
- 虚数成分の符号を反転させること

20

のうちの少なくとも 1 つを含む、請求項 1 4 に記載の装置。

【請求項 1 6】

前記複素数処理命令は即値として前記転置パラメータを指定する、請求項 1 4 又は 1 5
に記載の装置。

【請求項 1 7】

前記複素数成分の結果セットを前記デスティネーション・レジスタに適用することは、
前記複素数成分の結果セットを前記デスティネーション・レジスタに格納することを含む
、請求項 1 から 1 6 までのいずれかに記載の装置。

30

【請求項 1 8】

前記複素数成分の結果セットを前記デスティネーション・レジスタに適用することは、
前記複素数成分の結果セットを前記デスティネーション・レジスタの内容で累算すること
を含む、請求項 1 から 1 7 までのいずれかに記載の装置。

【請求項 1 9】

前記デスティネーション・レジスタは前記第 1 のソース・レジスタ及び前記第 2 のソー
ス・レジスタのうちの 1 つである、請求項 1 から 1 8 までのいずれかに記載の装置。

【請求項 2 0】

前記複素数演算は飽和演算である、請求項 1 から 1 9 までのいずれかに記載の装置。

40

【請求項 2 1】

データ処理装置を動作させる方法であって、

命令をデコードするステップと、

入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む前記入力データ・
ベクトルへの前記命令によって指定されるベクトル処理演算を選択的に適用するステップ
と、

複素数演算、第 1 のソース・レジスタ、第 2 のソース・レジスタ、及びデスティネーシ
ョン・レジスタを指定する複素数処理命令に応答して、前記データ処理装置のデータ処理
回路が、

50

前記第 1 のソース・レジスタ内の交互する位置からデータ項目の少なくとも 1 つの第 1 のセットを抽出することであって、前記第 1 のソース・レジスタ内の連続データ項目は複素数の第 1 のセットの交互する実数成分及び虚数成分を含む、抽出することと、

前記第 2 のソース・レジスタ内の交互する位置からデータ項目の少なくとも 1 つの第 2 のセットを抽出することであって、前記第 2 のソース・レジスタ内の連続データ項目は複素数の第 2 のセットの交互する実数成分及び虚数成分を含む、抽出することと、

前記データ項目の第 1 及び第 2 のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、前記複素数成分の結果セットは、前記複素数の第 1 のセット及び前記複素数の第 2 のセットに適用された前記複素数演算の複素数結果の実数部及び虚数部のうちの 1 つである、生成することと、

前記複素数成分の結果セットを前記デスティネーション・レジスタに適用することとを行うように制御するための制御信号を生成するステップとを含む、

前記複素数処理命令は、前記第 1 のソース・レジスタ及び前記第 2 のソース・レジスタのうちの少なくとも 1 つについての複製インデックスをさらに指定し、前記データ処理回路はさらに前記制御信号に応答して、前記第 1 のソース・レジスタ及び前記第 2 のソース・レジスタのうちの前記少なくとも 1 つにおける前記複製インデックスによって示される複素数のソース・セットを複製する、方法。

【請求項 2 2】

命令をデコードする手段と、

入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む前記入力データ・ベクトルへの前記命令によって指定されるベクトル処理演算を選択的に適用する手段と、

複素数演算、第 1 のソース・レジスタ、第 2 のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に応答して、データ処理装置内のデータ処理を行うための手段が、

前記第 1 のソース・レジスタ内の交互する位置からデータ項目の少なくとも 1 つの第 1 のセットを抽出することであって、前記第 1 のソース・レジスタ内の連続データ項目は複素数の第 1 のセットの交互する実数成分及び虚数成分を含む、抽出することと、

前記第 2 のソース・レジスタ内の交互する位置からデータ項目の少なくとも 1 つの第 2 のセットを抽出することであって、前記第 2 のソース・レジスタ内の連続データ項目は複素数の第 2 のセットの交互する実数成分及び虚数成分を含む、抽出することと、

前記データ項目の第 1 及び第 2 のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、前記複素数成分の結果セットは、前記複素数の第 1 のセット及び前記複素数の第 2 のセットに適用された前記複素数演算の複素数結果の実数部及び虚数部のうちの 1 つである、生成することと、

前記複素数成分の結果セットを前記デスティネーション・レジスタに適用することとを行うように制御するための制御信号を生成する手段とを備え、

前記複素数処理命令は、前記第 1 のソース・レジスタ及び前記第 2 のソース・レジスタのうちの少なくとも 1 つについての複製インデックスをさらに指定し、前記データ処理を行うための手段はさらに前記制御信号に応答して、前記第 1 のソース・レジスタ及び前記第 2 のソース・レジスタのうちの前記少なくとも 1 つにおける前記複製インデックスによって示される複素数のソース・セットを複製する、装置。

【請求項 2 3】

命令実行環境を提供するためのホスト・データ処理装置を制御するためのコンピュータ・プログラムであって、

命令をデコードするための命令デコード・プログラム・ロジックと、

入力データ・ベクトル構造体内の個々の位置に複数の入力データ項目を含む前記入力データ・ベクトル構造体への前記命令によって指定されるベクトル処理演算を選択的に適用するためのデータ処理プログラム・ロジックと

10

20

30

40

50

を含み、

前記命令デコード・プログラム・ロジックは複素数演算、第1のソース・データ構造体、第2のソース・データ構造体、及びデスティネーション・データ構造体を指定する複素数処理命令に応答して、前記データ処理プログラム・ロジックが、

前記第1のソース・データ構造体内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、前記第1のソース・データ構造体内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、

前記第2のソース・データ構造体内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、前記第2のソース・データ構造体内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、

前記データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、前記複素数成分の結果セットは、前記複素数の第1のセット及び前記複素数の第2のセットに適用された前記複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、

前記複素数成分の結果セットを前記デスティネーション・データ構造体に適用することとを行うように制御するための制御信号を生成し、

前記複素数処理命令は、前記第1のソース・データ構造体及び前記第2のソース・データ構造体のうちの少なくとも1つについての複製インデックスをさらに指定し、前記データ処理プログラム・ロジックはさらに前記制御信号に応答して、前記第1のソース・データ構造体及び前記第2のソース・データ構造体のうちの前記少なくとも1つにおける前記複製インデックスによって示される複素数のソース・セットを複製する、コンピュータ・プログラム。

【請求項24】

請求項23に記載の前記コンピュータ・プログラムを非一時的に記憶するコンピュータ可読記憶媒体。

【発明の詳細な説明】

【技術分野】

【0001】

本開示はデータ処理装置に関する。より詳細には、データ処理装置で複素数演算を実施することに関する。

【背景技術】

【0002】

データ処理装置は様々なコンテキストにおいて複素数計算を実施することが要求され得る。いくつかのデータ処理システムは、ソース・オペランド又はベクトル処理命令によって指定される結果値が指定されるデータ処理演算に並列にかけられる複数のデータ項目を含むベクトルであるベクトル命令の処理をサポートする。複素数の2部構造（すなわち、その実数部、及び虚数部）は、ベクトル化されるやり方で複素数演算を実施することができるデータ処理システムを提供することが単純ではないことを意味している。

【先行技術文献】

【非特許文献】

【0003】

【文献】Robert Bedichek、「Some Efficient Architecture Simulation Techniques」、1990年冬季USENIX Conference、53～63頁

【発明の概要】

【課題を解決するための手段】

【0004】

一実施例の実施例において、命令をデコードするための命令デコーダ回路と、入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む入力データ・ベクトルへの命令によって指定されるベクトル処理演算を選択的に適用するためのデータ処理回路とを備え

10

20

30

40

50

る装置があり、命令デコーダ回路は複素数演算、第1のソース・レジスタ、第2のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に应答して、データ処理回路が、第1のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・レジスタ内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、第2のソース・レジスタ内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・レジスタに適用することとを行うように制御するための制御信号を生成する。

10

【0005】

別の実例の実施例において、データ処理装置を動作させる方法があり、方法は、命令をデコードするステップと、入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む入力データ・ベクトルへの命令によって指定されるベクトル処理演算を選択的に適用するステップと、複素数演算、第1のソース・レジスタ、第2のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に应答して、データ処理装置のデータ処理回路が第1のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・レジスタ内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、第2のソース・レジスタ内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・レジスタに適用することと、を行うように制御するための制御信号を生成するステップとを含む。

20

30

【0006】

別の実例の実施例において、命令をデコードする手段と、入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む入力データ・ベクトルへの命令によって指定されるベクトル処理演算を選択的に適用する手段と、複素数演算、第1のソース・レジスタ、第2のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に应答して、データ処理装置内のデータ処理が第1のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・レジスタ内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、第2のソース・レジスタ内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・レジスタに適用することと、を行うための手段を制御するための制御信号を生成する手段とを含む装置がある。

40

【0007】

別の実例の実施例において、命令実行環境を提供するためのホスト・データ処理装置を制御するためのコンピュータ・プログラムがあり、コンピュータ・プログラムは、命令を

50

デコードするための命令デコード・プログラム・ロジックと、入力データ・ベクトル構造体内の個々の位置に複数の入力データ項目を含む入力データ・ベクトル構造体への命令によって指定されるベクトル処理演算を選択的に適用するためのデータ処理プログラム・ロジックとを含み、命令デコード・プログラム・ロジックは複素数演算、第1のソース・データ構造体、第2のソース・データ構造体、及びデスティネーション・データ構造体を指定する複素数処理命令に応答して、データ処理プログラム・ロジックが、第1のソース・データ構造体内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・データ構造体内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・データ構造体内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、第2のソース・データ構造体内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・データ構造体に適用することと、を行うように制御するための制御信号を生成する。

【0008】

本技法は、その実施例を参照して添付の図面に図示されるように、単に実例としてさらに説明される。

【図面の簡単な説明】

【0009】

【図1】一実施例における、データ処理装置を概略的に示す図である。

【図2】一実施例における、非拡大(non-widening)複素数演算を実施するデータ処理回路を概略的に示す図である。

【図3】一実施例における、拡大(widening)複素数演算を実施するデータ処理回路を概略的に示す図である。

【図4】一実施例における、複製演算から複素数演算を始めるデータ処理回路を概略的に示す図である。

【図5】符号付き乗算複素数命令の4つの種類に応じた実施例における、データ処理回路の動作の図である。

【図6】ベクトル化された複素数の最終的なセットを生成するための、図5で示されるものなどの複素数命令の使用に続くナロー・シフト(narrowing shift)演算の使用の図である。

【図7】図7A～7Dは、一実施例における、4つの異なるタイプの複素数ドット積命令に応じたデータ処理回路を概略的に示す図である。

【図8】初期のデータ精度に戻るために、図7A～図7Dで示されるものなどの複素数ドット積命令の使用に続いて一実施例で実行される一連の演算を示す図である。

【図9】図9A～9Dは、一実施例における、4つの異なるタイプの非拡大複素数ドット積命令に応じたデータ処理回路を概略的に示す図である。

【図10】初期のデータ精度に戻るために、図9A～9Dで示されるものなどの命令の実行に続いて一実施例で実行される一連の演算を示す図である。

【図11】図11A～11Bは、一実施例における、さらに転置パラメータを含む、複素数加算命令をサポートするためのデータ処理回路を概略的に示す図である。

【図12】一実施例における4つの個々の転置パラメータに応じて実行される4つの転置のセットを示す図である。

【図13】一実施例の方法を実行する際に行われる一連のステップを示す図である。

【図14】一実施例において、シミュレータ実装形態を提供するシステムのコンポーネントを概略的に示す図である。

【発明を実施するための形態】

10

20

30

40

50

【0010】

少なくともいくつかの実施例は、命令をデコードするための命令デコーダ回路と、入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む入力データ・ベクトルへの命令によって指定されるベクトル処理演算を選択的に適用するためのデータ処理回路とを備える装置であり、命令デコーダ回路は複素数演算、第1のソース・レジスタ、第2のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に応答して、データ処理回路が、第1のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・レジスタ内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、第2のソース・レジスタ内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・レジスタに適用することとを行うように制御するための制御信号を生成する、装置を提供する。

10

【0011】

現代のデータ処理システムは、様々なコンテキスト、例えば数例を挙げると、デジタル通信、レーダ・システム、及び光学システムにおいて、複素数の算術的計算を実施するように要求されることがある。さらには、ベクトル化されたやり方でデータを処理することができる（すなわち、単一の命令によって指定されるデータ処理演算をデータ・ベクトル内の複数のデータ項目に対して並列に適用することができる）データ処理システムは、スループット及び処理効率の点から有利性をもたらすことができる。しかしながら、複素数データにベクトル処理技法を適用することは、いくつかの理由で単純ではない。まず、複素数の固有の構造（実数成分及び虚数成分）は、データ処理システムが、ベクトル内のデータ項目の大きなセット内のそれぞれの別個の複素数のデータ項目のこれらの個々の部分を正確に扱えるように構成されなければならないこと、さらには、入力データがベクトル化されている性質により、複素数の2つのセットの個々の成分を所与の複素数演算でまとめることができるやり方に対してさらなる制約が加えられることを意味している。本技法は、データ処理回路がソース・レジスタのそれぞれにおいて交互する位置、つまりソース・レジスタが連続データ項目のセットを保持しており、ここでこれらのデータ項目は複素数のセットの実数成分と虚数成分との交互になっており、且つ実数成分と虚数成分とが別個の複素数を形成するために対（または「ペア」ともいう）となっていると認識される場所、からデータ項目のセットを抽出する手法を提供する。それによって、データ処理回路は例えば、ソース・レジスタから実数成分を引き出すことができる（他の実例では代替又は追加で、同様に虚数成分を抽出できることに留意すべきであるが）。よって、データ処理回路は次いで必要とされる複素数演算を、個々のソース・レジスタから抽出した成分に適用して、結果セットの複素数成分を作ることができ、ここで複素数成分の結果セットは、複素数演算の複素数結果の実数成分のセット、又は虚数部のいずれかである。これは、複素数成分のさらなるセットがやはり生成されることを排除するものではないこと、すなわち、複素数のセットの実数部及び虚数部の両方が生成され得るが、いくつかの実施例においてはどちらか一方しか生成されないことに留意されたい。次いで複素数成分の結果セットは複素数処理命令で指定されるデスティネーション・レジスタに適用されるが、この適用は様々な形態を取ることができる。例えば、結果セットは単にデスティネーション・レジスタに格納することができるか、又はデスティネーション・レジスタの既存の内容に累算する（数学的方法の一種）ことができる。

20

30

40

【0012】

上で言及したように、複素数成分の結果セットは、定義された特定の複素数処理命令に依存して変化することができるが、いくつかの実施例において、複素数成分の結果セット

50

は複素数結果の実数成分のセットである。同様に、他の実施例では、複素数成分の結果セットは複素数結果の虚数成分のセットである。

【0013】

複素数処理命令に応答してデータ処理回路によって生成された複素数成分の結果セットは、様々な方法でデスティネーション・レジスタに適用することができるが、いくつかの実施例において、データ処理回路は制御信号に応答して複素数成分の結果セットをデスティネーション・レジスタ内の連続位置に適用する。それにより、複素数成分の結果セットが複素数演算の複素数結果の実数部及び虚数部のうちの1つである場合、これはデスティネーション・レジスタの内容が第1及び第2のソース・レジスタの内容とは異なるフォーマットであること、すなわち、第1及び第2のソース・レジスタが複素数のセットの実数成分及び虚数成分の連続する対を保持している場合、デスティネーション・レジスタは連続する実数成分又は連続する虚数成分のいずれかのみを保持していることを意味している。これによって、多様な複素数演算を定義することが可能となるが、結果については、ベクトル処理構成内で扱われることができるだけである。

10

【0014】

実際一部のそのような実施例において、複素数成分の結果セットのそれぞれの複素数成分は、データ項目の第1及び第2のセットのそれぞれのデータ項目より拡大している。それにより、複素数成分の結果セットの成分のこの拡大は、複素数成分の結果セットの精度が、ベクトル処理環境内で演算するにもかかわらずより良好に保存され、そうでなければ結果セットの複素数成分は、それらをベクトル処理の自身の個々のレーンにフィットさせるために、著しい切り捨てを必要としていたかも知れないことを意味している。したがって、本技法は例えば、複素数の結果セットの実数成分及び虚数成分を2つの別個の命令によって決定することができ、その後これらの成分を実数及び虚数の複素数対の連続するセットに再結合する機会を伴う手法を提供するものであることを理解すべきである。

20

【0015】

それにもかかわらず、上で言及したいくつかの実施例においてデータ処理回路は複素数成分の結果セットをデスティネーション・レジスタの連続位置に適用することができるが、いくつかの実施例においては、データ処理回路は複素数実数成分の結果セット、及び対応する複素数虚数成分の結果セットの両方を生成し、これらは連続する実数成分－虚数成分対が結果として複素数の連続するセットを形成するよう、デスティネーション・レジスタへの適用においてインターリーブすることができる。それにより、いくつかの実施例において、データ処理回路は制御信号に応答して、複素数成分の結果セットを、複素数成分のさらなる結果セットでインターリーブされるデスティネーション・レジスタ内の交互する位置に適用し、複素数成分の結果セットが複素数結果の実数成分のセットである場合、複素数成分のさらなる結果セットは複素数結果の虚数成分のセットであり、複素数成分の結果セットが複素数結果の虚数成分のセットである場合、複素数成分のさらなる結果セットは複素数結果の実数成分のセットである。

30

【0016】

データ処理回路がデータ項目の第1のセット及びデータ項目の第2のセットを第1及び第2のソース・レジスタ内のそれらの個々の交互する位置から抽出するやり方は様々な定義することができ、実際いくつかの実施例において様々な制御することができ、例えば、いくつかの実施例において複素数処理命令は、データ項目の第1のセットが第1のソース・レジスタの奇数番号位置又は偶数番号位置から抽出されるかどうかをさらに指定し、複素数処理命令は、データ項目の第2のセットが第2のソース・レジスタの奇数番号位置又は偶数番号位置から抽出されるかどうかをさらに指定する。「奇数」及び「偶数」位置は、本明細書で「下位(又はボトム(bottom))」及び「上位(又はトップ(top))」位置と称されることもある。

40

【0017】

また、複素数演算を実施するために、データ項目の第1及び第2のセットが抽出される前に、第1のソース・レジスタ及び第2のソース・レジスタのうちの少なくとも1つに適

50

用されるさらなるデータ処理があってもよく、いくつかの実施例において、複素数処理命令は、第1のソース・レジスタ及び第2のソース・レジスタのうちの少なくとも1つについての複製インデックスをさらに指定し、データ処理回路はさらに制御信号に応答して、第1のソース・レジスタ及び第2のソース・レジスタのうちの少なくとも1つにおける複製インデックスによって示される複素数のソース・セットを複製する。したがって、第1及び第2のソース・レジスタのうちの少なくとも1つの「インデックス付け」によって、プログラマは個々のレジスタに保持されているデータのサブセットを識別することができ、このサブセットは再作成することができ、それにより第1及び第2のデータ・セットの抽出が次いでこの再作成されたサブセットを使用することができる。例えば、定義されるインデックスの性質に依存して、複製インデックスはベクトル・レジスタの所定の部分の2つの2分の1のうちの1つを指定することができ、次いで複製によってその指定された2分の1を、それぞれ所定の部分内の他の2分の1にコピーする。具体的な一実例を挙げると、複製インデックスが適用されるベクトル・レジスタの単位が128ビット長であり、複製インデックスがこの単位の上半分を指定し、本技法がその128ビットの空間を利用して4つの連続する32ビットの複素数成分（すなわち、2つの64ビットの複素数）を格納してある場合、複製によって、識別された64ビットの複素数を、その128ビット単位中で隣接する64ビット空間に複製する。この動作は、ベクトル・レジスタの全幅にわたって繰り返される。

【0018】

上記実例は一回実施される複製を示しているが、いくつかの実施例において、複素数のソース・セットの複製は、複素数のソース・セットの複数回の複製を含む。例えば、複製インデックスがベクトルの所与の単位の4分の1を指定する場合、この4分の1は単位すべてを満たすために3回複製することができる。上記実例の128ビット単位をそのままにしておくが、複素数成分はそれぞれの16ビットである一実例においては、これによって識別された32ビットの複素数（16ビットの実数成分及び16ビットの虚数成分）が、この128ビット単位上にある他の3つの32ビットの部分に複製される。このような「インデックス付け」を本明細書で説明される命令のあらゆるものに適用することができる。

【0019】

いくつかの実施例においては、単一のデータ項目のセットのみが第1のソース・レジスタから抽出されており、第1のソース・レジスタ内の複素数のセットの実数成分のセット又は虚数成分のセットのいずれかを表現しているが、いくつかの実施例では実数成分及び虚数成分の両方が抽出される。それにより、これらの実施例では、第1のソース・レジスタから抽出されるデータ項目の少なくとも1つの第1のセットは、データ項目の第1のセット及びデータ項目の第3のセットを含み、データ項目の第1のセットが複素数の第1のセットの実数成分である場合、データ項目の第3のセットは複素数の第1のセットの虚数成分であり、データ項目の第1のセットが複素数の第1のセットの虚数成分である場合、データ項目の第3のセットは複素数の第1のセットの実数成分であり、またデータ処理回路は制御信号にさらに応答して、さらにデータ項目の第3のセットをオペランドとして使用して、複素数成分の結果セットを生成する。したがって、データ処理回路は複素数演算における改善のために複素数の第1のセットの実数成分及び虚数成分の両方を有しており、定義された特定の複素数処理命令（したがって、複素数演算）に相当である。

【0020】

同様に、いくつかの実施例において、第2のソース・レジスタから抽出されるデータ項目の少なくとも1つの第2のセットは、データ項目の第2のセット及びデータ項目の第4のセットを含み、データ項目の第2のセットが複素数の第2のセットの実数成分である場合、データ項目の第4のセットは複素数の第2のセットの虚数成分であり、データ項目の第2のセットが複素数の第2のセットの虚数成分である場合、データ項目の第4のセットは複素数の第2のセットの実数成分であり、またデータ処理回路は制御信号にさらに応答して、さらにデータ項目の第4のセットをオペランドとして使用して、複素数成分の結果

セットを生成する。

【0021】

複素数演算は様々な形態を取ることができるが、いくつかの実施例において、複素数演算は、複素数ドット積演算であり、データ処理回路は制御信号に応答して、複素数の第1のセットの実数成分－虚数成分対の対のセット、並びに複素数の第2のセットの実数成分－虚数成分対の対のセット、の部分積から複素数成分の結果セットを累算する累算回路を備える。いくつかの実施例において、複素数演算は複素数乗算演算を含む。いくつかの実施例において、複素数演算は複素数加算演算である。いくつかの実施例において、複素数演算は複素数減算演算である。

【0022】

本技法は複素数の第1及び第2のセットの1つ又は両方の成分の追加的な操作が複素数演算に先立って実行され得ることをさらに提供するが、これは特に必要とされる複素数結果の効率的な決定を可能とすると見出されているためであり、例えば、そうでなければ第1及び第2のソース・レジスタのうちの少なくとも1つの内容を操作するために追加的な命令が予め実行されることを必要としていたであろう。ひいては、いくつかの実施例は複素数処理命令がさらに転置パラメータを指定すること、またデータ処理回路が制御信号に応答して、複素数の第1及び第2のセットへの複素数演算の適用に先立って複素数の第1及び第2のセットのうちの少なくとも1つに対して転置演算を適用することを提供する。

【0023】

複素数の第1及び第2のセットのうちの少なくとも1つに適用されるこの転置演算は、様々な定義することができる。これは「回転」と称されることがあり、実際、いくつかの事例においては、原点を中心とする複素数平面において表現される複素数の回転に対応することができるが、本技法はそのような「真」の回転に対応する転置に限定されず、いくつかの実施例においては、転置演算は以下のうちの少なくとも1つを含む：少なくとも1つの複素数の第1のセット及び複素数の第2のセットにおいて、実数成分の虚数成分での置換、虚数成分の実数成分での置換、実数成分と虚数成分の交換、実数成分を負の値にすること（negation）、及び虚数成分を負の値にすること。

【0024】

転置パラメータは様々な方法で与えられ得るが、いくつかの実施例において、複素数処理命令は即値として転置パラメータを指定する。上で言及したように、転置は常に複素数平面内の真の回転に直接対応しているわけではないが、それにもかかわらず、転置パラメータは本明細書において所々「回転値」と称され、いくつかの実施例において、4つの値のセットのうちの1つを取ることができ、複素数平面の回転の実例に類似して、複素数平面内の4分の1回転の回転角を指して0、90、180、及び270と記されていることがある。

【0025】

上で言及したように、複素数成分の結果セットは様々な方法でデスティネーション・レジスタに適用することができるが、いくつかの実施例において、複素数成分の結果セットをデスティネーション・レジスタに適用することは、複素数成分の結果セットをデスティネーション・レジスタに格納することを含む。他の実施例において、複素数成分の結果セットをデスティネーション・レジスタに適用することは、複素数成分の結果セットをデスティネーション・レジスタの内容で累算することを含む。

【0026】

デスティネーション・レジスタは、第1及び第2のソース・レジスタ両方とは別個のレジスタとして指定することができるが、いくつかの実施例において、デスティネーション・レジスタが第1のソース・レジスタ及び第2のソース・レジスタのうちの1つである「相殺的」演算を定義することができる。

【0027】

データ処理回路が、結果値がその値を保持するよう定義された記憶装置には大きくなり過ぎてしまう状況を扱うやり方は、様々な形態を取ることができる。例えば、値はラップ

10

20

30

40

50

・アラウンドする（すなわち、最大値を超えると0に戻る）ことができる、一方でいくつかの実施例において、複素数演算は飽和演算（すなわち、換言すると、最大値を超えると結果はその最大値のまま保持される）である。

【0028】

少なくともいくつかの実施例は、データ処理装置を動作させる方法を提供し、方法は、命令をデコードするステップと、入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む入力データ・ベクトルへの命令によって指定されるベクトル処理演算を選択的に適用するステップと、複素数演算、第1のソース・レジスタ、第2のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に応答して、データ処理装置のデータ処理回路が、第1のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・レジスタ内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、第2のソース・レジスタ内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・レジスタに適用することとを行うように制御するための制御信号を生成するステップとを含む。

【0029】

少なくともいくつかの実施例は、装置を提供し、装置は、命令をデコードする手段と、入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む入力データ・ベクトルへの命令によって指定されるベクトル処理演算を選択的に適用する手段と、複素数演算、第1のソース・レジスタ、第2のソース・レジスタ、及びデスティネーション・レジスタを指定する複素数処理命令に応答して、データ処理装置内のデータ処理が、第1のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・レジスタ内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・レジスタ内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出することであって、第2のソース・レジスタ内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・レジスタに適用することとを行うための手段を制御するための制御信号を生成する手段とを備える。

【0030】

少なくともいくつかの実施例は、命令実行環境を提供するためのホスト・データ処理装置を制御するためのコンピュータ・プログラムを提供し、コンピュータ・プログラムは、命令をデコードするための命令デコード・プログラム・ロジックと、入力データ・ベクトル構造体内の個々の位置に複数の入力データ項目を含む入力データ・ベクトル構造体への命令によって指定されるベクトル処理演算を選択的に適用するためのデータ処理プログラム・ロジックとを含み、命令デコード・プログラム・ロジックは複素数演算、第1のソース・データ構造体、第2のソース・データ構造体、及びデスティネーション・データ構造体を指定する複素数処理命令に応答して、データ処理プログラム・ロジックが、第1のソース・データ構造体内の交互する位置からデータ項目の少なくとも1つの第1のセットを抽出することであって、第1のソース・データ構造体内の連続データ項目は複素数の第1のセットの交互する実数成分及び虚数成分を含む、抽出することと、第2のソース・データ構造体内の交互する位置からデータ項目の少なくとも1つの第2のセットを抽出するこ

とであって、第2のソース・データ構造体内の連続データ項目は複素数の第2のセットの交互する実数成分及び虚数成分を含む、抽出することと、データ項目の第1及び第2のセットをオペランドとして使用して複素数成分の結果セットを生成することであって、複素数成分の結果セットは、複素数の第1のセット及び複素数の第2のセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの1つである、生成することと、複素数成分の結果セットをデスティネーション・データ構造体に適用することとを行うように制御するための制御信号を生成する。

【0031】

少なくともいくつかの実施例は、上で言及したコンピュータ・プログラムの実施例にしたがってコンピュータ・プログラムを非一時的に記憶するコンピュータ可読記憶媒体を提供する。

【0032】

いくつかの特定の実施例を次に図面を参照して説明する。

【0033】

図1は本技法の様々な実例を具現化し得るデータ処理装置10を概略的に図示している。装置は、それが実行する一連の命令に応答してデータ項目に対してデータ処理演算を実施するデータ処理回路12を備える。これらの命令は、データ処理装置がアクセス権を有するメモリ14から取り出され、当業者に馴染みのあるやり方でこの目的のためにフェッチ回路16が設けられる。さらには、フェッチ回路16によって取り出された命令は命令デコーダ回路18を通過し、命令デコーダ回路18は、処理回路12並びにレジスタ20のセット、及びロード／ストア・ユニット22の、設定及び動作の様々な態様を制御するように構成される制御信号を生成する。一般的に、データ処理回路12はパイプライン様式に構成することができるが、その詳細は本技法に関連性がない。当業者であれば図1が表す全体的な構成には馴染みがあるが、そのさらなる詳細な説明は、単に簡略化のためここでは省略する。図1から分かるように、レジスタ20はそれぞれ複数のデータ要素についての記憶装置を備え、それによって処理回路はデータ処理演算を指定されたレジスタ内の指定されたデータ要素に適用することができるか、又はデータ処理演算を指定されたレジスタ内の指定されたデータ要素の群（「ベクトル」）に適用することができる。特に、図示されるデータ処理装置はベクトル化されたデータ処理演算のパフォーマンス、具体的にはレジスタ20に保持されるデータ要素の観点から、複素数処理命令の実行に関連しており、そのさらなる説明をいくつかの具体的な実施例を参照して以下でより詳細に与える。命令の実行においてデータ処理回路12によって必要とされるデータ値、及びそれらのデータ処理命令の結果として生成されるデータ値は、ロード／ストア・ユニット22によって、メモリ14に書き込まれ、メモリ14から読み出される。図1のメモリ14は一般的に、本技法の命令を記憶することができるコンピュータ可読記憶媒体の実例として、典型的には処理回路がその後に行う所定の命令のシーケンスの一部（「プログラム」）として、見なされ得ることにも留意されたい。しかしながら処理回路は、RAM内の、ROM内の、ネットワーク・インターフェースを介して、などの、様々な異なるソースからそのようなプログラムにアクセスすることができる。本開示は、処理回路12が実行することができる様々な新規な命令を説明しており、これらの命令の実行などをサポートするために以下の図面はこれらの命令の性質、データ処理回路における変形例のさらなる説明を与える。

【0034】

図2は一実施例における、複素数乗算命令をサポートするための、データ処理回路を概略的に図示している。図面の下部に見られるように、複素数乗算命令CMULは第1のソース・レジスタS1、第2のソース・レジスタS2、及びデスティネーション・レジスタZを指定している。さらに、この実施例において、複素数乗算命令は関与する複素数成分のデータ・サイズをさらに指定しており、すなわちこれらが32ビット値であることを示す「.s」の追加的な指定を伴う。この命令に応答して（又は、図1の上述から理解されるように厳密に言うと、命令に応答してデコーダ回路18によって生成された制御信号に

10

20

30

40

50

応答して) データ処理回路は第1のソース・レジスタS1及び第2のソース・レジスタS2の両方から実数の複素数成分のセット及び虚数の複素数成分のセットを抽出し、これらの成分のそれぞれは32ビット値である。図2においてソース・レジスタS1及びS2の内容を記している実例の成分から分かるように、それぞれのソース・レジスタの内容は複素数成分対、すなわち、実数成分及び虚数成分のシーケンスである。ソース・レジスタ内のデータのこの構造は、複素数が共にレジスタ内のこの場所、及びそれらをロードしてきたメモリ場所の両方に格納されるという点で有用であり、これは別個の実数成分及び虚数成分をメモリからロードするための演算をロードすること及びデインタリーブすることに関連する複雑さが、必要ではないことを意味している。さらには、図示される命令及びそれをサポートするためのデータ処理回路の提供は、それぞれのソース・レジスタ内のインターリーブされた実数成分及び虚数成分を直接処理することができ、それによって連続的なロード及びストア命令の使用がこれらの値のセットをレジスタに入れ、またメモリに戻すことを可能にしており、さらに実装を容易にし、パフォーマンスの改善をサポートしていることを意味している。図2に図示される複素数乗算命令の具体的な実例において、第1及び第2のソース・レジスタから入力個々の対を受信する乗算ユニット30のセットが示されている。図面にはそれぞれのソース・レジスタ及びデスティネーション・レジスタ内には4つの成分のセットのみが示されて明示的に記されているが、本技法はこの処理がベクトル・レジスタの全幅にわたって(図面の左右への破線によって示されるように)複製することができることを提供することに留意すべきである。乗算ユニット30によって実施される乗法演算の結果は、対になった入力を加算ユニット32に与え、2つの実体

10

20

【0035】

図2のCMUL命令が属する、非拡大命令である、本技法によって提供される実例の命令のセットは次の通りである：

? CMUL : complex multiply with rotation;
 ? CMLA : complex multiply accumulate with rotation;
 ? SQRDCMULH : signed saturating rounding doubling complex multiply with rotation, returning high half; and
 ? SQRDCMLAH : signed saturating rounding doubling complex multiply-accumulate with rotation, returning high half.

30

(上記命令セットの訳は以下の通り)

- ・CMUL：回転ありの複素数乗算
- ・CMLA：回転ありの複素数乗累算
- ・SQRDCMULH：上位半分を返す、回転ありの符号付き飽和丸め付きダブル複素数乗算
- ・SQRDCMLAH：上位半分を返す、回転ありの符号付き飽和丸め付きダブル複素数乗累算

40

【0036】

図3は一実施例における別の複素数処理命令を概略的に図示しており、図2の実例とのいくつかの顕著な差異を示している。この実例において、命令の命名規則SQDMLALBTは「符号付き ダブル 乗算 加算 Long 下位 上位」(“signed doubling multiply add long bottom top”)に対応している。図3から分かるように、この命令をサポートするために与えられるデータ処理回路の観点から、4つの乗算ユニット36のセットは第1及び第2のソース・レジスタから入力を受信し、命令はやはり「シングルの」32ビット値のセットがそれぞれから抽出されるべきであるよう指定「s」している。この命令の「下位 上位」(“bottom top”)の特徴は、成分対の下半分は第1のソース・レジスタzs1

50

から抽出され、成分対の上半分は第2のソース・レジスタ zs_2 から抽出されることを示していることに留意されたい。これらの対同士の乗法演算の結果、加算ユニット38のセットへの1つの入力を抽出し、加算ユニット38はこの値のセットをデスティネーション・レジスタ zd から抽出される値へ加算する、すなわち、これは累算演算である。さらに、命令はデスティネーション・レジスタ内の成分のサイズを「d」すなわち64ビット値として指定することに留意されたい。換言すると、この演算は拡大演算である。図示される命令の結果として、ソース・レジスタ zs_1 内の複素数のセットとソース・レジスタ zs_2 内の複素数のセットとの乗算の結果の虚数成分のセットが生成され、デスティネーション・レジスタ zd に累算され、結果成分が入力成分に対して拡大することを可能にしており、それによって精度が保存される。図2の事例のように、図面では4つの成分のセットのみがそれぞれのソース・レジスタから抽出されてデスティネーション・レジスタに累算されるように明示的に示されているが、本技法はこの処理がベクトル・レジスタの全幅にわたって複製することができる（図面の左右への破線及び省略によって示されるように）ことを提供する。

10

【0037】

図3のSQDMLALBT命令が属する、拡大命令であり、加算／減算下位／上位下位／上位（ここで下位及び上位はアクセスされる複素数の下半分及び上半分を指す、すなわちそれぞれの複素数の実数部及び虚数部）を指定する、本技法によって提供される実例の命令のセットは次の通りである：

```
?    SQDMLALB : signed multiply add long bottom bottom;
?    SQDMLSLB : signed multiply subtract long bottom bottom;
?    SQDMLALT : signed multiply add long top top;
?    SQDMLSLT : signed multiply subtract long top top;
?    SQDMLALBT : signed multiply add long bottom top; and
?    SQDMLSLBT : signed multiply subtract long bottom top.
```

20

（上記命令セットの訳は以下の通り）

- ・SQDMLALB：符号付き 乗算 加算 Long 下位 下位
- ・SQDMLSLB：符号付き 乗算 減算 Long 下位 下位
- ・SQDMLALT：符号付き 乗算 加算 Long 上位 上位
- ・SQDMLSLT：符号付き 乗算 減算 Long 上位 上位
- ・SQDMLALBT：符号付き 乗算 加算 Long 下位 上位
- ・SQDMLSLBT：符号付き 乗算 減算 Long 下位 上位

30

【0038】

これらはすべて「MLA」（乗算加算（multiply add）－乗累算としても知られる）命令であり、「MUL」バージョン（すなわち純粋に乗算のバージョン）が、例えばデスティネーション・レジスタのあらゆる既存の内容が累算されるべきではない累算処理の初回の繰り返しでの使用のために、やはり与えられることに留意されたい。これらのためには、上記列挙の「MLA」をちょうど「MUL」に置き換える必要がある。

【0039】

図4は、本技法にしたがって、別の実例の命令、すなわち複素数乗累算命令をサポートするために一実施例において提供されるデータ処理回路を概略的に図示している。図4において特に留意すべきこととして、第2のソース・レジスタ zs_2 についての複製インデックス（[1]）の追加的な指定が挙げられる。図面で示されるように、サポートされる処理のベクトル化された性質が128ビットの粒度で適用される場合、すなわち図示される処理がベクトル・レジスタの幅にわたってこれらの128ビットの粒度で繰り返される場合、且つ命令が、入力成分サイズがシングルすなわち32ビットであると指定する場合、複製インデックスを「1」と指定することは、128ビットの粒度の上半分にある32ビット値の対によって表される複素数が、その下半分に複製されるべきであることを示す。他の実例において、例えば、命令が同じ128ビット粒度内で「ハーフ」（すなわち16ビット）の値を指定する場合、粒度の内容は2倍の数の複素数を表すことになり、すな

40

50

わち 8 個の 16 ビット値があることになり、この複製インデックスによってそのセット内の (4 つのうち) 第 2 の複素数が他の 3 つに複製されることになることに留意されたい。しかしながら、図 4 に示される具体的な実例を考慮すると、複製に続いて第 1 及び第 2 のソース・レジスタから抽出された成分は、実際には図 2 に示されるユニット 30 と同じ乗算ユニットであってもよい乗算ユニット 40 への入力の対を与える。同様に、乗算ユニット 40 の対の結果は加算ユニット 42 への入力を与える。図 2 における事例のように、加算ユニット 42 のうち 2 つは負の値にされた入力のうちの 1 つを有しているが、図 2 の実例とは対照的に、図 4 の実例における命令は累算演算であり、そのためデスティネーション・レジスタで保持される個々の成分の既存の内容は、加算ユニット 42 へ追加的な入力を与えることに留意されたい。このように生成された結果値は、デスティネーション・レジスタ zds の個々の成分の新しい内容を形成する。これは拡大命令ではない、すなわち結果成分が入力成分と同じサイズであり、したがって、この命令のアクションに続くデスティネーション・レジスタの内容は、複素数のセットのインターリーブされた実数成分及び虚数成分のセットであること、すなわち 2 つのソース・レジスタが有するのと同じフォーマットであることに留意されたい。図 2 及び図 3 の事例のように、図面では 4 つの成分のセットのみがそれぞれのソース・レジスタから抽出されてデスティネーション・レジスタに累算されるように明示的に示されているが、本技法はこの処理がベクトル・レジスタの全幅にわたって複製することができる (図面の左右への破線及び省略によって示されるように) ことを提供する。

【0040】

図 5 は、いくつかの実施例におけるさらなる命令のセットの動作を図示している。これらはすべていずれかの連続する実数の複素数成分のセット (第 1 の命令の実例において) 又は連続する虚数成分のセット (第 3 及び第 4 の命令の実例において) を生成する拡大命令であることが分かる。やはり、図 2 ~ 図 4 について上述したように、それぞれの命令は処理されるデータ要素のサイズ、及び生成される結果データ要素のサイズを指定するが、ここでは具体的なサイズは図面では汎用のままである、例えば $\langle Zda \rangle$ 、 $\langle Tb \rangle$ 及び $\langle Zn \rangle \langle Tb \rangle$ であることに留意されたい。図 5 の表記では、処理の終了において、作られた実数及び虚数の結果はデインターリーブされており、デスティネーション・レジスタ Zda はすべての実数結果を含み、且つ Zdb はすべての虚数結果を含むことが分かる。図 6 に目を移すと、次にデスティネーション・レジスタ Zc の複素数成分の図示される結果セットを作るために、ナロー・シフト演算を使用してレジスタ Za 及び Zb からのデータを並べ替えることができるが、この場合これらは元々のソース・レジスタ Zn 及び Zm (図 5) の連続する複素数のフォーマットであり、したがってデータのさらなる再配置は必要とされない。したがってこの複素数の結果のセットは連続的なストア動作でメモリに格納し戻すことができる。先の図面の事例のように、図 5 及び図 6 においては 4 つの成分のセットのみがそれぞれのソース・レジスタから抽出されてデスティネーション・レジスタに累算されるように明示的に示されているが、本技法はこの処理がベクトル・レジスタの全幅にわたって複製することができることを提供する。

【0041】

これらの乗算 加算/減算 下位/上位 下位/上位命令 (“multiply add/subtract bottom/top bottom/top instructions”) を利用する命令 (1) の実例のプログラム・シーケンスは、例えば次の通りであり得る：

実例のコード・シーケンス (1)：

...

. Loop :

...

LD1 $\langle Tb \rangle$ $\langle Zn \rangle$. $\langle Tb \rangle$, $\langle Pn \rangle / z$, [$\langle Xn \rangle$]

LD1 $\langle Tb \rangle$ $\langle Zm \rangle$. $\langle Tb \rangle$, p / z , [$\langle Xm \rangle$]

SQDMLALB $\langle Zda \rangle$. $\langle T \rangle$, $\langle Zn \rangle$. $\langle Tb \rangle$, $\langle Zm \rangle$. $\langle Tb \rangle$

SQDMLSLT $\langle Zda \rangle$. $\langle T \rangle$, $\langle Zn \rangle$. $\langle Tb \rangle$, $\langle Zm \rangle$. $\langle Tb \rangle$

10

20

30

40

50

```

S Q D M L A L T B < Z d b > . < T > , < Z n > . < T b > , < Z m > . < T b >
S Q D M L A L T B < Z d b > . < T > , < Z m > . < T b > , < Z n > . < T b >
. . .
. e n d L o o p :
S Q S H R N B < Z d a > . < T b > , < Z d a > . < T > , < i m m >
S Q S H R N T < Z d a > . < T b > , < Z d b > . < T > , < i m m >
S T 1 < Z d a > . < T b > , < P d > , [ X d ]
...

```

【0042】

上で言及した乗算命令と同じく、本技法はまた様々な複素数ドット積命令を提供し、その一部を、それらをサポートするために与えられるデータ処理回路と併せて図7A～図7Bの実例の実施例に示す。そのような複素数ドット積演算をサポートする第1の手法では、図7A～図7Dが図示するように結果値の実数部及び虚数部は別個に計算される。それにより、これは示される命令のセットが結果データに対して拡大する効果を生み出すことを可能にし、図7A～図7Dの実例から分かるように、データ精度を4倍高める、すなわち入力複素数成分（実数及び虚数）がそれぞれ16ビット値である場合、結果値は64ビット値である。ドット積ユニット50は、それぞれのソース・レジスタから得られる複素数の対の、個々の実数成分及び虚数成分からの4つの入力成分の個々のセットからそれぞれ部分積を算出する。「-1」と付されるいくつかの実体において、与えられる入力値の値にされることに留意されたい。このように生成される部分積が、加算回路52によって累算される。やはりこれらの図面で図示されるように、これらの演算は累算演算であり、デスティネーション・レジスタの個々の部分の既存の内容は、最終的な結果を生成する加算ユニットに対するさらなる入力を形成する。それにより、図7A～図7Dで示される演算の簡略な概要において、C D O T R Eは2つの複素数乗算の実数部を累算し、C D O T I Mは2つの複素数乗算の虚数部を累算し、C C D O T R Eは2つの複素共役乗算の実数部を累算し、C C D O T I Mは2つの複素共役乗算の虚数部を累算する。

【0043】

これらの命令のうちの2つの使用に続き（「通常の」複素数乗算が実施されているか、又は複素共役乗算かどうかに依存する）、初期のデータ精度に戻るため、図8に示されるような一連の演算を実行することができる。見て分かるように、レジスタZa、Zb、Zc、及びZd内の実数成分及び虚数成分は、個々の実数成分及び虚数成分の2つのセットをデスティネーション・レジスタZe及びZfにインターリーブするためにナロー・シフト演算にかけられ、次いで成分サイズをさらに削減し（そして元々のソース・レジスタの成分サイズに戻る）、1つのレジスタ内に最終的な複素数結果を与えるよう実数成分及び虚数成分をインターリーブするためにさらなるナロー・シフト演算が実施される。先の図面の事例のように、図7A～図7D、及び図8で明示的に示される回路及び処理は、ベクトル・レジスタの全幅にわたって複製することができる。

【0044】

本技法の利益は、次の2つの実例のコード・シーケンス（2）及び（3）を考慮することによって理解され得、それらは図7A～図7Dで図示されるようなものなどの命令を使用する本技法にしたがって取られる手法と比較した場合の、そのような複素数ドット積処理の先行技術の実装形態を表している。本技法のC D O T命令によって、データ処理回路の計算スループットの著しい改善（2倍）を可能にし、一方で累算ヘッドルームを増加させている（16ビット入力に対して64ビットの累算器、すなわち4倍）ことが分かる。

【0045】

実例の先行技術のコード・シーケンス（ARM（登録商標） NEON（商標）の、複素数ドット積の実装形態）（2）：

```

. . .
. L o o p :
. . .

```

10

20

30

40

50

```

// aとbから8つの複素数値をロード
ld2 {v0.8h, v1.8h}, [aPtr], #32
ld2 {v2.8h, v3.8h}, [bPtr], #32
smlal v4.4s, v0.4h, v2.4h // c.re+=a.re*b.re
smlal v5.4s, v1.4h, v2.4h // c.im+=a.im*b.re
smlsl v4.4s, v1.4h, v3.4h // c.re-=a.re*b.im
smlal v5.4s, v0.4h, v3.4h // c.im+=a.re*b.im
smlal2 v6.4s, v0.8h, v2.8h // c.re+=a.re*b.
re
smlal2 v7.4s, v1.8h, v2.8h // c.im+=a.im*b.
re
smlsl2 v6.4s, v1.8h, v3.8h // c.re-=a.re*b.
im
smlal2 v7.4s, v0.8h, v3.8h // c.im+=a.re*b.
im
...
.endLoop:
...
add v4.4s, v4.4s, v6.4s
add v5.4s, v5.4s, v7.4s
addv s4, v4.4s //実数に対して水平の累算
addv s5, v5.4s //虚数に対して水平の累算
sqshrn h4, s4, #16 //実数を16右シフト
sqshrn h5, s5, #16 //虚数を16右シフト
st2 {v4.h, v5.h} [0], [outPtr] //出力を格納
[0046]
本技法による実例のコード・シーケンス（CDOTを使用する複素数ドット積について
）（3）：
...
.Loop:
...
// aとbから8つの複素数値をロード
ld1h z0.h, <Pn>/z, [aPtr, #0]
ld1h z2.h, <Pn>/z, [aPtr, #16]
ld1h z1.h, <Pn>/z, [bPtr, #0]
ld1h z3.h, <Pn>/z, [bPtr, #16]
cdotre z4.d, z0.h, z1.h // c.re+=a.re*b.re
cdotim z5.d, z0.h, z1.h // c.im+=a.im*b.re
cdotre z6.d, z2.h, z3.h // c.re+=a.re*b.re
cdotim z7.d, z2.h, z3.h // c.im+=a.im*b.re
...
.endLoop:
...
sqrshrn b z8.s, z4.d, #16
sqrshrn b z8.s, z6.d, #16
sqrshrn b z9.s, z5.d, #16
sqrshrn b z9.s, z7.d, #16
sqrshrn b z10.h, z8.s, #16
sqrshrn b z10.h, z9.s, #16
rev z11.h, <Pm>/m, z10.h

```

```

u z i p 1 z 1 2 . s , < P m > / m , z 1 1 . s
s t r s 1 2 , [ c P t r ]
. . .

```

【 0 0 4 7 】

また本技法によって複素数ドット積演算をサポートする代替的な手法が提供され、図 9 A～図 9 D 及び図 1 0 を参照して次に議論する。ここで、上で議論した複素数ドット積命令とは対照的に、データの拡大は 2 倍だけであり、図 9 A～図 9 D の実例から分かるように、ドット積命令の結果はやはり結果レジスタ内でインターリーブされる（すなわち、交互する実数成分及び虚数成分）。この手法では、図 9 A～図 9 D から分かるように、個々のソース・レジスタから抽出された成分に対する演算の初期のセットは、乗算ユニット 5 4 によって実施される乗法演算だけである。図 7 A～図 7 D の実例のように、ある入力

負の値にされることに留意されたい。また、図 9 A～図 9 D の実例では命令は 1 つだけ定義され、すなわち S Q R D C D O T であるが、4 つの実例において示されるこの命令に対して 4 つの異なる応答が、命令内の追加的なパラメータ（すなわち、# 0、# 9 0、# 1 8 0、及び # 2 7 0）の使用から生じ、これらのパラメータは転置を表している、すなわちそれぞれのソース・レジスタの個々の実数成分及び虚数成分を結合する特定のやり方を示していることにも留意されたい。これらは複素数平面内の複素数の回転と類似していることから口語表現的には「回転」と称されるが、実施される転置は一般的には複素数平面内の回転に厳密に対応しないと理解することが重要である。これは、成分の対応する使用が「回転値」の 4 つの異なる値について第 1 及び第 2 のソース・レジスタから得られる図 1 2 をさらに参照して理解され得ることであるが、ここで第 2 のソース・レジスタについて使用される転置が実際に、回転パラメータ値によって与えられる回転角によって表される複素数の複素数平面内の回転に対応していることが分かるが、これは第 1 のソース・レジスタの場合に適用される転置についてはこの通りではなく、それゆえ一般的にこれらの転置が厳密には複素数平面内の回転ではないとコメントされる。乗算ユニット 5 4 の結果は、図 9 A～図 9 D に示される累算（加算）ユニット 5 6 に第 1 の入力の対を与え、この時デスティネーション・レジスタの個々の部分の既存の内容がさらなる入力（すなわち累算用）を与える。したがって、実数成分及び虚数成分の両方がこの手法にしたがって生成され、複素数ドット積結果を作るが、それにもかかわらず、これはなお生成された実数及び虚数成分が、実数及び虚数の入力成分のサイズの 2 倍となる、拡大演算である。上述の第 1 の手法のように、図 9 A～図 9 D の S Q R D C D O T 命令の、命令の使用にはナロー・シフト演算の対が続くことができ、データを初期のデータ精度に戻し、実数及び虚数結果を 1 つのデスティネーション・レジスタにインターリーブすることができ、この時別のデスティネーション・レジスタ内の実数及び虚数成分は複素数成分の単一のセットを作り、インターリーブされた実数成分及び虚数成分は図 1 0 から分かるように単一のデスティネーション・レジスタ内に複素数値を形成する。先の図面の事例のように、図 9 A～図 9 D、及び図 1 0 で明示的に示される回路及び処理は、ベクトル・レジスタの全幅にわたって複製することができる。

【 0 0 4 8 】

上で言及した「回転値」はまた本明細書で説明される他の複素数処理命令の多くに有用に適用することができ、次に説明する。例えば、複素数加算命令は、極表示で考えると、要素の対を相殺的なやり方で第 1 のソース・ベクトルの対応する要素に加算する前に、第 2 のソース・ベクトル内で複素数を 9 0 度又は 2 7 0 度回転させることができる。

【 0 0 4 9 】

複素数乗算加算命令は、2 つの命令を結合することにより複素数に対して乗算加算演算の作成を可能とするためにオペランドの変形を実施する。変形は次のように実施される：

（1）極形式で考えて、第 2 のソース・ベクトル内の複素数を、第 1 のソース・ベクトルの重複している実数成分を乗算する前に、0 度、又は 1 8 0 度回転させる。（2）極形式で考えて、第 2 のソース・ベクトル内の複素数を、第 1 のソース・ベクトルの重複している虚数成分を乗算する前に、9 0 度、又は 2 7 0 度回転させる。得られる積は次いでデス

ディネーション且つ加数ベクトルの対応する成分に加算される。これらの変形を使用して、2つの命令を有利に結合することができる。例えば、次のシーケンスを考える場合：

S Q R D C M L A H Z d a . S , Z n . S , Z m . S , # A

S Q R D C M L A H Z d a . S , Z n . S , Z m . S , # B

この時AとBとの意味のある結合は次の通りである：

A = 0 , B = 9 0。この場合、Z n及びZ m内の複素数の2つのベクトルは乗算され、その積がZ d a内の複素数に加算される。

A = 0 , B = 2 7 0。この場合、Z n内の複素数の共役はZ m内の複素数で乗算され、その積がZ d a内の複素数に加算される。

A = 1 8 0 , B = 2 7 0。この場合、Z n及びZ m内の複素数の2つのベクトルは乗算され、その積がZ d a内の複素数から減算される。

A = 1 8 0 , B = 9 0。この場合、Z n内の複素数の共役はZ m内の複素数で乗算され、その積がZ d a内の複素数から減算される。

【0050】

上述のように、拡大複素数固定小数点命令は、複素数値の実数成分及び虚数成分をデインタリーブするが、一方で入力値よりも数値精度のより高い複素数結果の成分を生成するためにそれら进行处理する（すなわち、それらが「拡大する」）。したがって、これらの命令は、複素数結果の両方の成分をデスティネーションにおいて隣接する要素に書き込むべくそれぞれの複素数の実数成分及び虚数成分と一緒に処理する他の複素数命令とは、殊に異なっている。以下の命令は複素数乗算加算の結果の拡大した成分を生成する際、有用である：

- ・ S Q D M L A L B T : 拡大した虚数
- ・ S Q D M L S L T : 拡大した実数
- ・ S Q D M L A L B : 拡大した共役実数
- ・ S Q D M L S L B T : 拡大した共役虚数

【0051】

固定小数点複素数ドット積命令は、ソース・ベクトルを例えば8ビット、又は16ビットの符号付き整数複素数の対にデリミットする。それぞれの対の中で、第1のソース・ベクトル内の複素数値は第2のソース・ベクトル内の対応する複素数値で乗算され、得られる拡大した積は合計され、第1のソース・ベクトル内の複素数の対をオーバーラップさせる累算器の例えば32ビット又は64ビット要素に加算される。

【0052】

これらの命令は、基本的なドット積の機能性を提供することに加え複素数値をデインタリーブする。したがって、ソース・ベクトルはインターリーブされた複素数の実数成分及び虚数成分を含むが、デスティネーション・ベクトルは拡大実数成分又は拡大虚数成分のいずれかの排他的セットを保持している。これらの命令についての回転値を使用すると次の通りである：

回転が0の場合、結果は複素数ドット積演算の実数成分を含むベクトルである。

回転が90の場合、結果は複素数ドット積演算の虚数成分を含むベクトルである。

回転が180の場合、結果は複素共役ドット積演算の実数成分を含むベクトルである。

回転が270の場合、結果は複素共役ドット積演算の虚数成分を含むベクトルである。

【0053】

符号付き／符号なし整数複素数ドット積命令は、ソース・ベクトルを例えば8ビット、又は16ビットの符号付き／符号なし整数複素数の対にデリミットする。それぞれの対の中で、第1のソース・ベクトル内の複素数値は第2のソース・ベクトル内の対応する複素数値で乗算され、得られる拡大した積は合計され、第1のソース・ベクトル内の複素数の対をアラインする累算器の、例えば32ビット又は64ビット要素（デスティネーション要素）に加算される。

【0054】

被乗数ソース・ベクトルから読み出された値は、乗算ステップが実施される前に回転パ

ラメータに基づいて変形される。イミディエイトな回転値に依存して、ソース複素数の部分のうちの1つ又は両方が並べ替えられる、及び／又は負の値にされるか、ソース複素数の部分のいずれも並べ替えられない、及び／又は負の値にされない。これらの命令についての回転値を使用すると次の通りである：

回転が0の場合、第2のソースからの複素数の奇数番号の成分が負の値にされ、デスティネーション・ベクトルは複素数ドット積演算の実数成分を含む。

回転が90の場合、第2のソースからの複素数の実数部と虚数部がスワップされ、デスティネーション・ベクトルは複素数ドット積演算の虚数成分を含む。

回転が180の場合、変形は適用されず、デスティネーション・ベクトルは複素共役ドット積演算の実数成分を含む。

回転が270の場合、第2のソースからの複素数の偶数番号の成分が負の値にされ、次に第2のソースから導出されて得られるベクトルの実数部と虚数部がスワップされ、デスティネーション・ベクトルは複素共役ドット積演算の虚数成分を含む。

【0055】

いくつかの実施例において、複素数演算は様々な複素数加算（すなわち、複素数減算を含む）であってもよく、これらの一部をサポートするためのデータ処理回路を図11A及び図11Bに概略的に図示している。繰り返しになるが、上で言及される「回転値」が利用され、図11Aの実例では#90の値が使用され、図11Bの実例では#270の値が使用される。図面で示されるように、これらの回転値（4つのセットである）は実際には2ビット値で表現されるよう提案され、これらの実例では00は#90に対応し、01は#270に対応していることに留意されたい。

【0056】

図11A及び図11Bで示される実例は、複素数加算命令をサポートするよう提供されるデータ処理回路の表現であり、一実施例において、命令セットはSQCAD又はUQADDのいずれかとして定義され（すなわち、飽和複素数加算命令の符号付き又は符号なしの変形例）、ここで飽和とは結果のそれぞれの成分内の結果値がラップ・アラウンドしないが、所定の最大値において飽和することを意味している。したがって、加算ユニット60は入力ソース・レジスタの個々の実数成分及び虚数成分から入力を得、（いくつかの入力は負の値にされることに留意されたい）これらを合計してデスティネーション・レジスタ内でインターリーブされる実数成分及び虚数成分を作る。回転パラメータの異なる値を使用して、第1及び第2のソース・レジスタ内の、すべての必要とされる複素数の実数成分及び虚数成分の結合の並べ替えを達成することができる。先の図面の事例のように、図11A及び図11Bで明示的に示される回路及び処理は、ベクトル・レジスタの全幅にわたって複製することができる。

【0057】

さらなる実施例において、SUBLTB及びADDLBT命令はインターリーブされた実数成分及び虚数成分に対して動作し、拡大命令である。これらの命令は、X及びYを複素数として複素数加算（ $X + jY$ ）又は（ $X - jY$ ）の結果の拡大した成分を生成する際、有用である：

ADDLBT：（ $X + jY$ ）を計算する際の拡大した虚数値

SUBLTB：（ $X + jY$ ）を計算する際の拡大した実数値

SUBLTB：（ $X - jY$ ）を計算する際の拡大した虚数値

ADDLBT：（ $X - jY$ ）を計算する際の拡大した実数値

【0058】

図13は、一実施例の方法を実施する際に行われる一連のステップを示している。フローは、ステップ100から始まると考えることができるが、ここで受信された命令のシーケンス内の次の命令がデコードされる。次いでステップ102で、これが本技法の複素数命令かどうか判断される。そうではない場合、フローはステップ104に進みここでこれらの命令が適当に処理されるが、これは本開示が考慮するところではない。次にフローはステップ100に戻り、次の命令をデコードする。しかしながら、これが複素数命令であ

る場合は、フローはステップ 106 に進み、ここで遭遇する複素数命令の特定のタイプに依存して、第 1 のソース・レジスタから実数成分及び／又は虚数成分が抽出され、ステップ 108 において第 2 のソース・レジスタから実数成分及び／又は虚数成分が抽出される。ステップ 106 及びステップ 108 の抽出は典型的には並列に実施され、図 13 では明瞭さのために、これを順次的な様式で説明するだけであることに留意すべきである。ステップ 110 において、これらの抽出された成分を使用して複素数演算が実施される。ステップ 112 はこれがインターリーブされる結果か、そうでないかを判断する。次にフローがステップ 114 に進む場合、実数及び虚数の結果成分が、デスティネーション・レジスタに適用される際にインターリーブされ、そうではなくこのフローがステップ 116 に進む場合は、この時実数の結果成分のセット又は虚数の結果成分のセットいずれかであるインターリーブされる結果が指定されたデスティネーション・レジスタに適用される。どのルートを取っても、次の命令をデコードするためにフローはステップ 100 に戻る。

【0059】

図 14 は使用され得るシミュレータ実装形態を図示している。先に説明された実施例は考慮される技法をサポートする特定の処理ハードウェアを動作させるための装置及び方法の点で本発明を実装している一方、本明細書で説明される実施例にしたがっており、コンピュータ・プログラムの使用を通じて実装される命令実行環境を提供することも可能である。そのようなコンピュータ・プログラムは、それらがハードウェア・アーキテクチャのソフトウェアベースの実装形態を提供する限りにおいて、しばしばシミュレータと称される。シミュレータ・コンピュータ・プログラムの種類としては、エミュレータ、仮想マシン、モデル、及びダイナミック・バイナリ変換器を含むバイナリ変換器が挙げられる。典型的には、シミュレータ実装形態は、任意選択でホスト・オペレーティングシステム 720 を実行させ、シミュレータ・プログラム 710 をサポートする、ホスト・プロセッサ 730 上で実行することができる。いくつかの構成において、ハードウェアと提供される命令実行環境との間に複数のシミュレーションのレイヤ、及び／又は同一のホスト・プロセッサ上に設けられる複数の別個の命令実行環境があってもよい。歴史的には、適度な速度で実行するシミュレータ実装形態を提供するためにはパワーのあるプロセッサが要求されてきたが、そのような手法は一定の状況下で正当化されてもよく、例えば互換性又は再使用目的のために別のプロセッサにとってネイティブなコードを実行する所望がある場合などである。例えば、シミュレータ実装形態は、ホスト・プロセッサ・ハードウェアによってサポートされていない追加的な機能性を伴う命令実行環境を提供すること、又は典型的には異なるハードウェア・アーキテクチャに関連付けられる命令実行環境を提供することができる。シミュレーションの概略は Robert Bedichek, 「Some Efficient Architecture Simulation Techniques」、1990 年冬季 USENIX Conference、53～63 頁に示されている。

【0060】

ある程度、その実施例は特定のハードウェア構築物又は特徴を参照して以前に説明されており、シミュレートされる実施例においては、等価な機能性が適切なソフトウェア構築物又は特徴によって提供され得る。例えば、特定の回路がコンピュータ・プログラム・ロジックとしてシミュレートされる実施例に実装されてもよい。同様に、レジスタ又はキャッシュなどのメモリ・ハードウェアが、ソフトウェア・データ構造体としてシミュレートされた実施例に実装されてもよい。以前に説明された実施例において参照されるハードウェア要素の 1 つ又は複数のホスト・ハードウェアに存在する構成（例えば、ホスト・プロセッサ 730）において、いくつかのシミュレートされる実施例は適切であればホスト・ハードウェアを利用することができる。

【0061】

シミュレータ・プログラム 710 はコンピュータ可読記憶媒体（非一時的媒体であってもよい）に記憶することができ、ターゲット・コード 700 にプログラム・インターフェース（命令実行環境）を提供するが、ターゲット・コード 700 はシミュレータ・プログ

ラム 710 によってモデリングされるハードウェア・アーキテクチャのアプリケーション・プログラム・インターフェースと同一のものである。したがって、上述の複素数処理命令を含むターゲット・コード 700 のプログラム命令は、命令実行環境内からシミュレータ・プログラム 710 を使用して実行することができ、それによって、上で議論した装置のハードウェア特徴を実際には有していないホスト・コンピュータ 730 がこれらの特徴をエミュレートすることができる。

【0062】

簡単で全体的な概要としては、入力データ・ベクトル内の個々の位置に複数の入力データ項目を含む入力データ・ベクトルに対してベクトル処理演算をサポートするための、装置、方法、プログラム、及び複素数処理命令が提供される。命令に応答して、データ項目の少なくとも 1 つの第 1 のセットが第 1 のソース・レジスタ内の交互する位置から抽出され、データ項目の少なくとも 1 つの第 2 のセットが第 2 のソース・レジスタ内の交互する位置から抽出され、ここで第 1 及び第 2 のソース・レジスタ内の連続データ項目は複素数の個々のセットの交互する実数成分及び虚数成分を含む。複素数成分の結果セットが、データ項目の 2 つのセットをオペランドとして使用して生成され、複素数成分の結果セットは、複素数の 2 つのセットに適用された複素数演算の複素数結果の実数部及び虚数部のうちの 1 つである。複素数成分の結果セットがデスティネーション・レジスタに適用される。

【0063】

本出願において、語句「するように構成される」は、装置の要素が定義される動作を実行することができる構成を有することを意味するために使用される。この文脈において、「構成」はハードウェア又はソフトウェアの内部接続の、配置又はやり方を意味する。例えば、装置は定義される動作を提供する専用のハードウェアを有してもよく、又はプロセッサ若しくは他の処理デバイスは機能を実施するためにプログラムされてもよい。「するように構成される」は、定義される動作を提供するために装置要素がいかなるやり方でも変更される必要があることを含意するものではない。

【0064】

図示的な実施例を本明細書において添付の図面を参照して詳細に説明してきたが、本発明はこれらの正確な実施例に限定されず、当業者により添付の特許請求の範囲によって定義されるような本発明の範囲及び趣旨を逸脱することなく、その様々な変形、付加、及び修正がそこになされ得ることを理解されたい。例えば、本発明の範囲を逸脱することなく、従属請求項の特徴と独立請求項の特徴との様々な組合せが可能である。

10

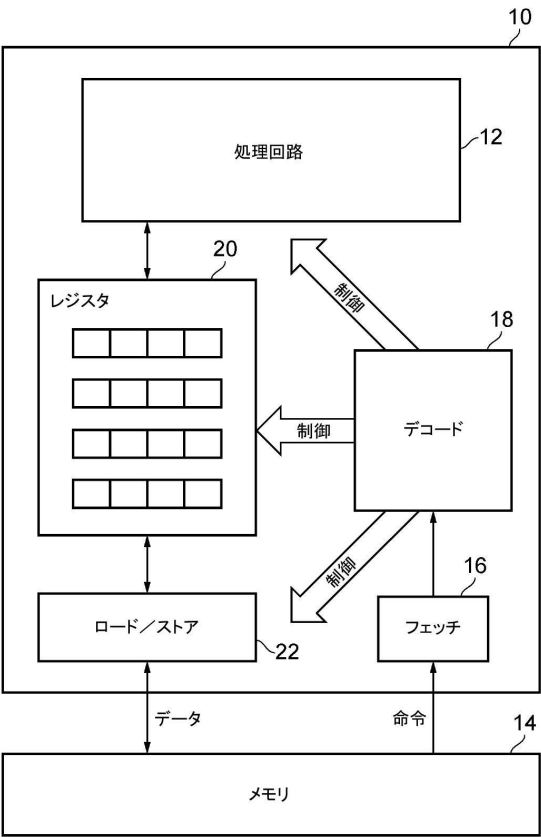
20

30

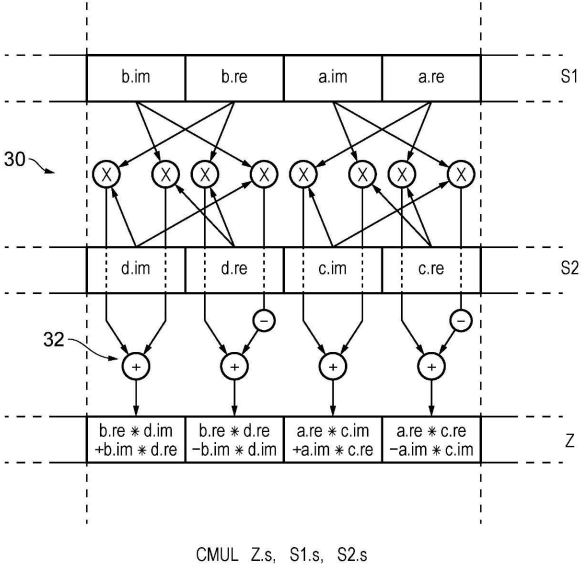
40

50

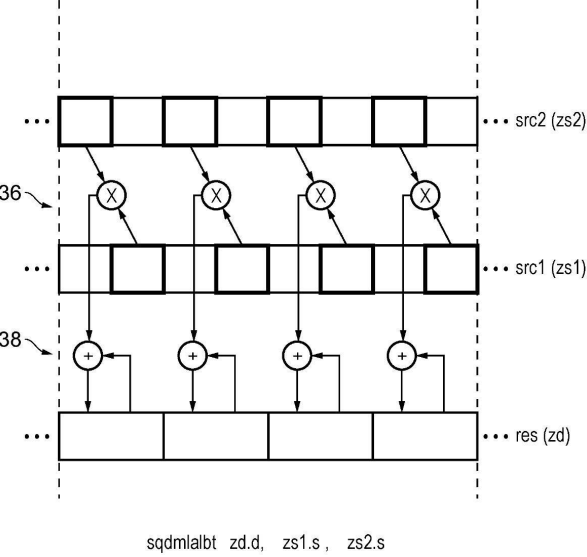
【図面】
【図 1】



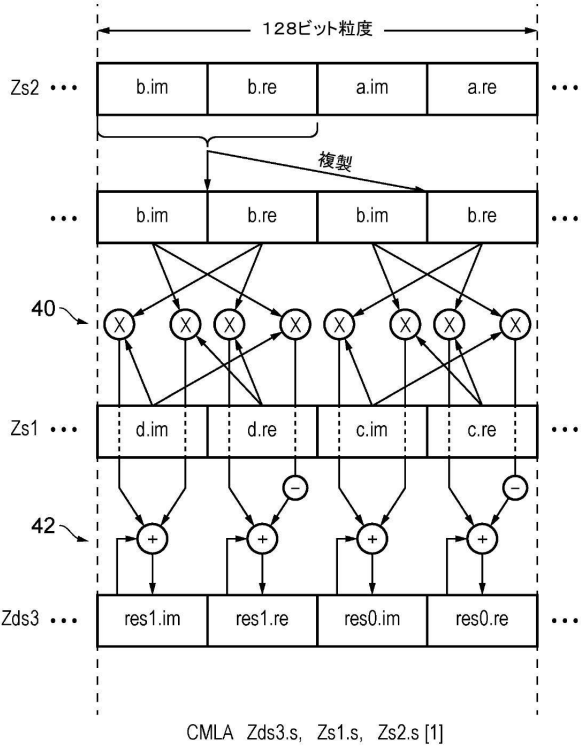
【図 2】



【図 3】



【図 4】



10

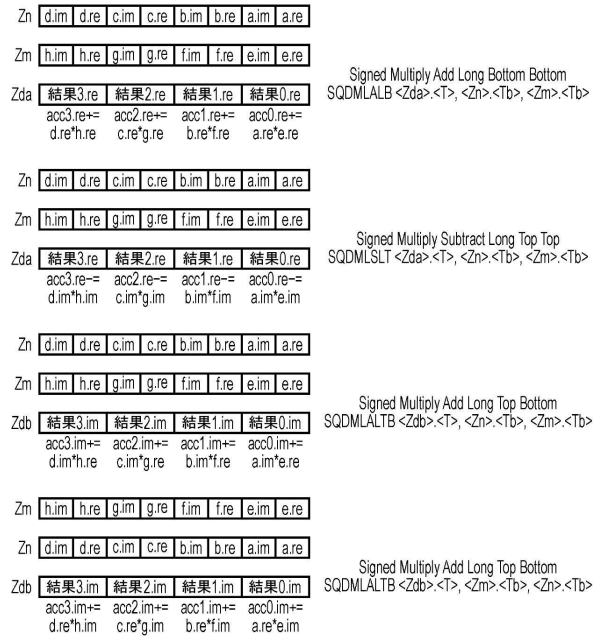
20

30

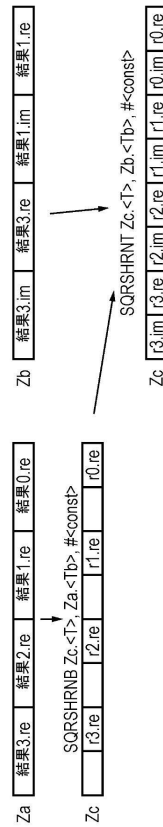
40

50

【圖 5】



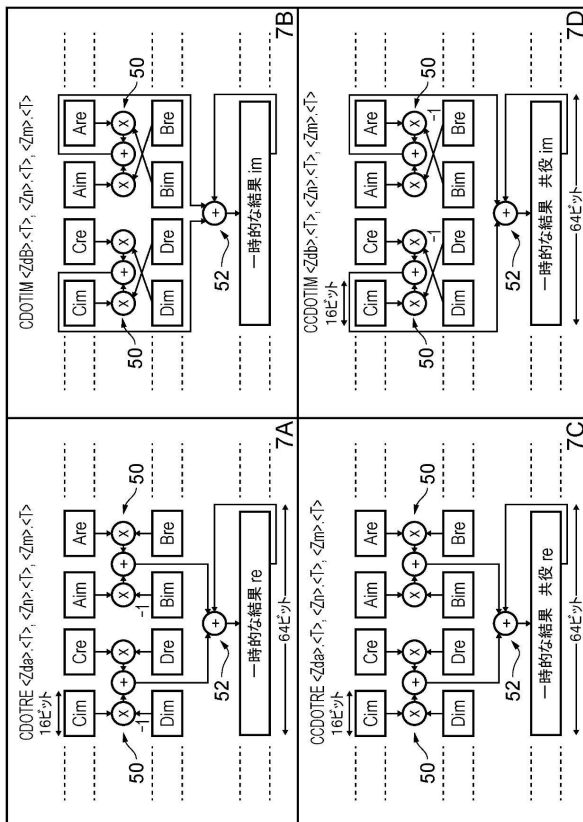
【図 6】



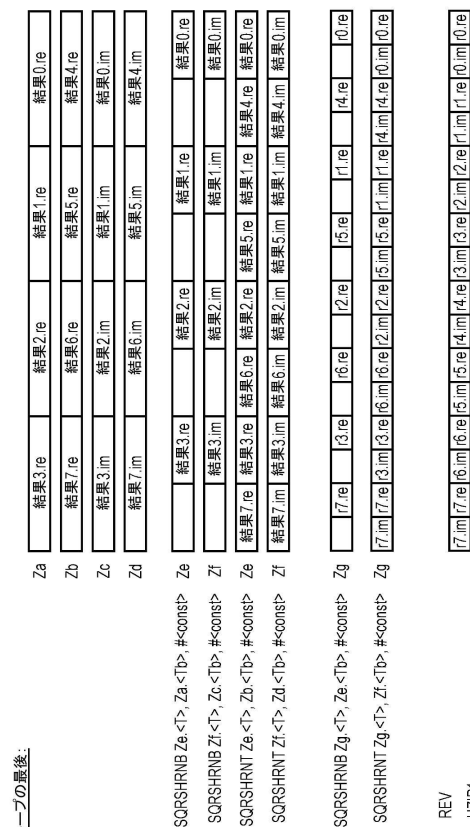
10

20

【圖 7】



【図 8】

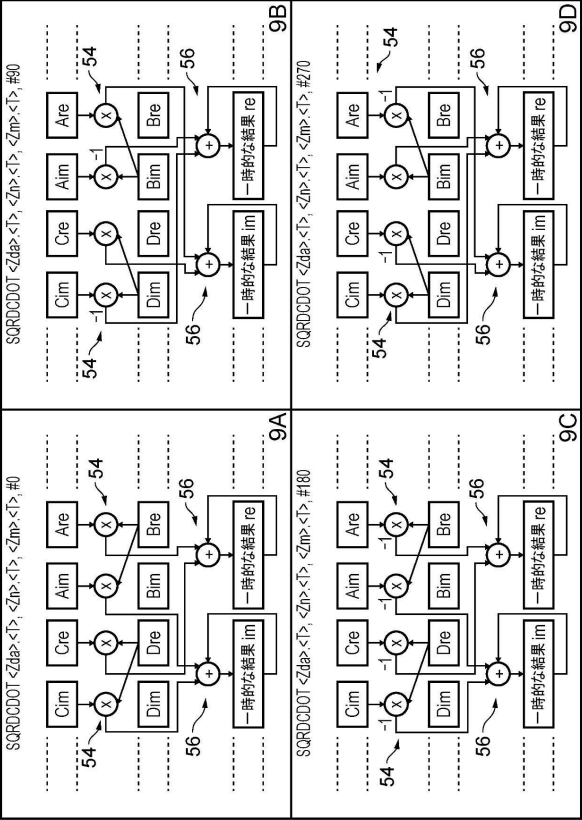


30

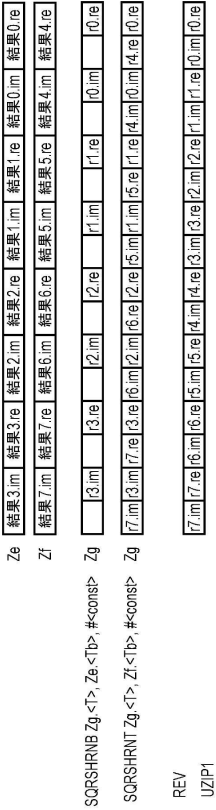
40

50

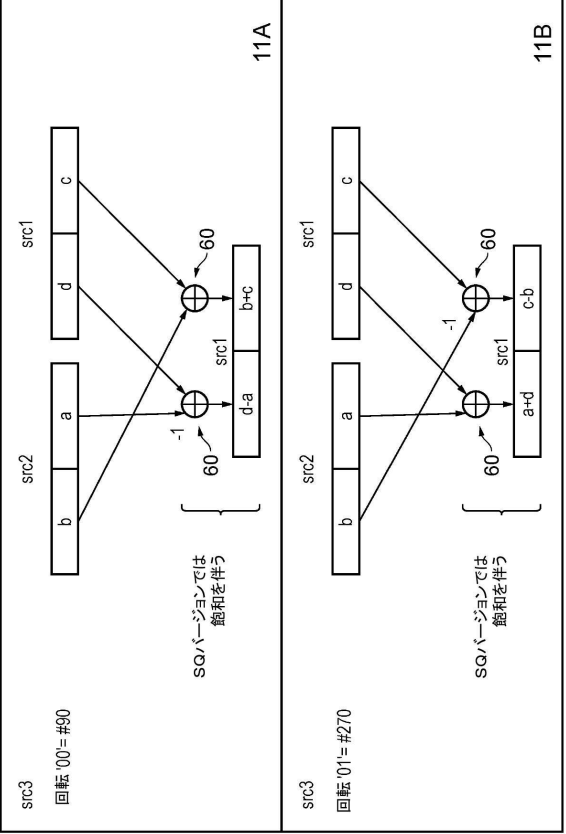
【図 9】



【図 10】



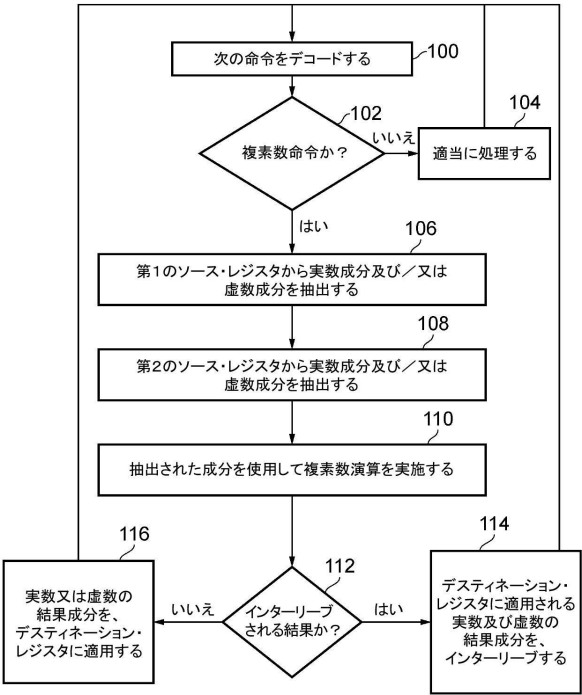
【図 11】



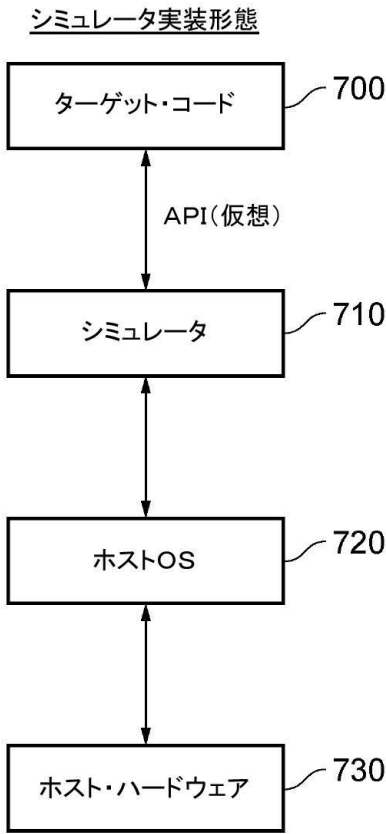
【図 12】

	出力	虚数	実数
#0	Src1	Re	Re
	Src2	Im	Re
#90	Src1	Im	Im
	Src2	Re	-Im
#180	Src1	Re	Re
	Src2	-Im	-Re
#270	Src1	Im	Im
	Src2	-Re	Im

【図 1 3】



【図 1 4】



10

20

30

40

50

フロントページの続き

英国、ケンブリッジシャー、ケンブリッジ、チェリー ヒントン、フルボーン ロード 110、エ
イアールエム リミテッド 気付

審査官 田中 幸雄

- (56)参考文献 特開2009-48532 (JP, A)
米国特許第5579253 (US, A)
米国特許出願公開第2005/0193185 (US, A1)
特表2009-505214 (JP, A)
特表2002-527808 (JP, A)
米国特許出願公開第2017/0177359 (US, A1)
- (58)調査した分野 (Int.Cl., DB名)
G06F 17/16