(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2009-140491
(P2009-140491A)
(43)公開日　平成21年6月25日(2009.6.25)

(51)Int.Cl.		F I		テーマコード（参考）
G06T 1/20	(2006.01)	G06T 1/20	C	5B056
G06F 15/80	(2006.01)	G06T 1/20	B	5B057
G06F 17/10	(2006.01)	G06F 15/80		
G06F 9/46	(2006.01)	G06F 17/10	S	
		G06F 9/46	410	

審査請求 有　請求項の数 23　ＯＬ　外国語出願　（全 44 頁）

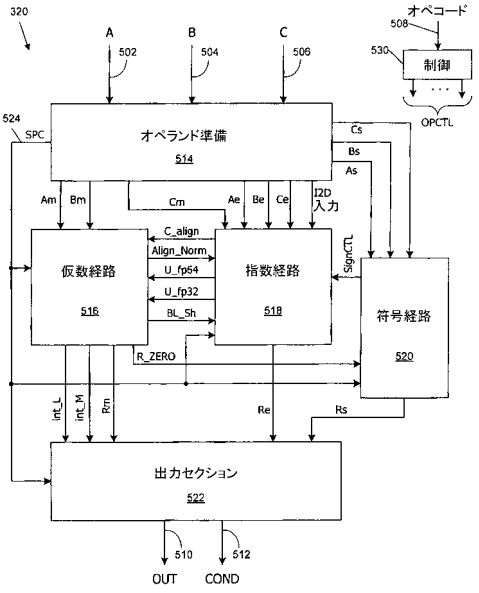
(21)出願番号　特願2008-302713 (P2008-302713)
(22)出願日　平成20年11月27日 (2008.11.27)
(31)優先権主張番号　11/952,858
(32)優先日　平成19年12月7日 (2007.12.7)
(33)優先権主張国　米国 (US)

(71)出願人　501261300
エヌヴィディア　コーポレイション
アメリカ合衆国，カリフォルニア　９５０５０，サンタ　クララ，サン　トーマス　エクスプレスウェイ　２７０１
(74)代理人　100094318
弁理士　山田　行一
(74)代理人　100123995
弁理士　野田　雅一
(74)代理人　100107456
弁理士　池田　成人
(72)発明者　スチュアート　エフ.　オベルマン
アメリカ合衆国，　カリフォルニア州，サンタ　クララ，　サン　トマス　エクスプレスウェイ　２７０１

最終頁に続く

(54)【発明の名称】融合型積和演算機能ユニット

(57)【要約】　（修正有）
【課題】倍精度のサポートを加えることによる、チップ面積に対する影響が低減する融合型積和演算機能ユニットを提供する。
【解決手段】グラフィックスプロセッサに、レンダリングに使用される単精度の機能ユニットに加えて、倍精度の算術演算の直接的なサポートを提供する機能ユニットが追加される。倍精度の機能ユニットは、融合型積和演算を含む複数の異なる演算を、少なくとも倍精度の幅であるデータ経路及び／又は論理回路を使用して、実行することができる。倍精度の機能ユニット及び単精度の機能ユニットは、共通の命令発行回路によって制御することができ、コアに含まれている倍精度の機能ユニットの複製の数を、単精度の機能ユニットの複製の数よりも少なくすることができる。
【選択図】図５



【特許請求の範囲】
【請求項１】
画像データを生成するようになっており、複数の同時スレッドを実行するようになっている処理コアを含んでおり、単精度オペランドに対して動作するレンダリングパイプラインを備えており、
前記処理コアが、倍精度の入力オペランドのセットに対して複数の倍精度演算のうちの一つを選択的に実行するようになっている多目的の倍精度機能ユニットを更に含んでおり、該多目的の倍精度機能ユニットが、少なくとも一つの算術演算論理回路を含んでおり、
前記倍精度機能ユニットの前記算術演算論理回路の全てが、倍精度において動作するよう十分な広さになっている、
グラフィックスプロセッサ。
【請求項２】
前記倍精度機能ユニットは、前記複数の倍精度演算のそれぞれが同数のクロックサイクルで完了するように、なっている、請求項１に記載のグラフィックスプロセッサ。
【請求項３】
前記倍精度機能ユニットは、前記複数の倍精度演算のそれぞれが、オーバーフロー条件又はアンダーフロー条件が発生するかにかかわらず、同数のクロックサイクルで完了するように、なっている、請求項２に記載のグラフィックスプロセッサ。
【請求項４】
前記倍精度機能ユニットは、オーバーフロー条件又はアンダーフロー条件が発生した場合に、浮動小数点算術演算の規格に準拠するオーバーフロー結果又はアンダーフロー結果を生成し、前記オーバーフロー条件又は前記アンダーフロー条件が発生したか否かを示すための出力ステータスフラグをセットするように、なっている、請求項３に記載のグラフィックスプロセッサ。
【請求項５】
前記倍精度機能ユニットは、前記複数の倍精度演算のうちの任意の一つを完了するために要する時間が、浮動小数点例外によって影響されないように、なっている、請求項１に記載のグラフィックスプロセッサ。
【請求項６】
前記複数の倍精度演算が
二つの倍精度オペランドを加算する加算演算と、
二つの倍精度オペランドを乗算する乗算演算と、
第１の倍精度オペランドと第２の倍精度オペランドとの積を計算し、次いで、前記積に第３の倍精度オペランドを加算する融合型積和演算と、
を含んでいる、請求項１に記載のグラフィックスプロセッサ。
【請求項７】
前記複数の倍精度演算が、第１のオペランドと第２のオペランドとに対して比較テストを実行し、前記比較テストが満たされているか否かを示すブール結果を生成する倍精度比較（ＤＳＥＴ）演算を更に含んでいる、請求項６に記載のグラフィックスプロセッサ。
【請求項８】
前記複数の倍精度演算が、
二つの倍精度入力オペランドのうちの大きい方のオペランドを返す倍精度最大値（ＤＭＡＸ）演算と、
二つの倍精度入力オペランドのうちの小さい方のオペランドを返す倍精度最小値（ＤＭＩＮ）演算と、
を更に含んでいる、請求項６に記載のグラフィックスプロセッサ。
【請求項９】
前記複数の倍精度演算が、倍精度形式から倍精度以外の形式にオペランドを変換する少なくとも一つの形式変換演算を更に含んでいる、請求項６に記載のグラフィックスプロセッサ。

【請求項１０】
前記複数の倍精度演算が、
倍精度以外の形式から倍精度形式にオペランドを変換する少なくとも一つの形式変換演算、
をさらに含んでいる、請求項６に記載のグラフィックスプロセッサ。
【請求項１１】
画像データを生成するようになっており、複数の同時スレッドを実行するようになっている処理コアを含んでいるレンダリングパイプラインを備えており、
前記処理コアが、一以上の単精度オペランドに対して算術演算を実行するようになっている単精度機能ユニットを含んでおり、
前記処理コアが、倍精度入力オペランドのセットに対して融合型積和演算を実行して倍精度の結果を提供するようになっている倍精度の積和演算（ＤＦＭＡ）機能ユニット、を更に含んでおり、
前記ＤＦＭＡ機能ユニットがＤＦＭＡパイプラインを含んでおり、前記ＤＦＭＡパイプラインは、当該ＤＦＭＡパイプラインを通る単一の経路において前記融合型積和演算が実行されるよう十分に広いデータ経路を有する、
グラフィックスプロセッサ。
【請求項１２】
前記ＤＦＭＡ機能ユニットが、
二つの倍精度仮数の積を１回の反復において計算するようになっている乗算器と、
二つの倍精度仮数の和を１回の反復において計算するようになっている加算器と、
を含んでいる、請求項１１に記載のグラフィックスプロセッサ。
【請求項１３】
前記ＤＦＭＡ機能ユニットが、一対の倍精度入力オペランドに対して乗算演算を実行して倍精度の結果を提供するように、更に構成されている、請求項１１に記載のグラフィックスプロセッサ。
【請求項１４】
前記乗算演算と前記融合型積和演算のそれぞれが、同数のクロックサイクルで完了する、請求項１３に記載のグラフィックスプロセッサ。
【請求項１５】
前記ＤＦＭＡ機能ユニットが、一対の倍精度入力オペランドに対して加算演算を実行して倍精度の結果を提供するように、更に構成されている、請求項１１に記載のグラフィックスプロセッサ。
【請求項１６】
前記加算演算と前記融合型積和演算のそれぞれが、同数のクロックサイクルで完了する、請求項１５に記載のグラフィックスプロセッサ。
【請求項１７】
前記ＤＦＭＡ機能ユニットが、一対の倍精度入力オペランドに対して乗算演算を実行して倍精度の結果を提供するように、更に構成されており、
前記融合型積和演算、前記加算演算、及び前記乗算演算のそれぞれが、オーバーフロー条件又はアンダーフロー条件が発生するか否かにかかわらず同数のクロックサイクルで完了する、
請求項１６に記載のグラフィックスプロセッサ。
【請求項１８】
前記ＤＦＭＡ機能ユニットが、オーバーフロー条件又はアンダーフロー条件が発生した場合に、浮動小数点算術演算の規格に準拠するオーバーフロー結果又はアンダーフロー結果を生成し、前記オーバーフロー条件又は前記アンダーフロー条件が発生したか否かをを示すための出力ステータスフラグをセットするように、更に構成されている、請求項１７に記載のグラフィックスプロセッサ。
【請求項１９】

前記処理コアが、並列に動作するようになっている前記第１の機能ユニットの複数（Ｐ個）の複製と、前記ＤＦＭＡ機能ユニットの複数（Ｎ個）の複製と、を含んでいる、請求項１１に記載のグラフィックスプロセッサ。

【請求項２０】

前記数Ｐが前記数Ｎよりも大きい、請求項１９に記載のグラフィックスプロセッサ。

【請求項２１】

前記数Ｎが１である、請求項２０に記載のグラフィックスプロセッサ。

【請求項２２】

前記処理コアが、前記ＤＦＭＡ機能ユニットを対象とするＰ個のセットの倍精度入力オペランドを集めて、前記Ｐ個のセットの倍精度オペランドのうち異なるセットの倍精度オペランドを、異なるクロックサイクルにおいて、前記ＤＦＭＡ機能ユニットに提供するようになっている入力マネージャ回路を更に含んでいる、請求項２１に記載のグラフィックスプロセッサ。

【請求項２３】

前記入力マネージャ回路が、前記第１の機能ユニットを対象とするＰ個のセットの単精度入力オペランドを集めて、前記第１の機能ユニットの前記Ｐ個の複製のそれぞれに、前記Ｐ個のセットの単精度オペランドのうち異なるセットの単精度オペランドを、並列に提供するように、なっている、請求項２２に記載のグラフィックスプロセッサ。

【発明の詳細な説明】

【技術分野】

【０００１】

[0001]本発明は、グラフィックスプロセッサに関するものであり、より詳細には、グラフィックスプロセッサにおける倍精度の融合型積和演算（Fused Multiply-add）機能ユニットに関するものである

【発明の背景】

【０００２】

[0002]グラフィックスプロセッサは、コンピュータシステムにおいて、２次元のジオメトリデータ又は３次元のジオメトリデータから画像をレンダリングするステップを高速化する目的で一般に使用されている。このようなプロセッサは、一般には、高い並列性及び高いスループットを持つように設計されており、数千個のプリミティブを並列に処理して、複雑でリアルなアニメーション画像をリアルタイムでレンダリングすることができる。高性能のグラフィックスプロセッサは、一般的な中央処理装置（ＣＰＵ）よりも高い計算能力を提供する。

【０００３】

[0003]最近では、グラフィックスプロセッサの処理能力を利用して、画像のレンダリングには無関係の様々な計算を高速化することが注目されている。「汎用」グラフィックスプロセッサを使用して、科学分野、金融分野、ビジネス分野、及びその他の分野における計算を実行することができる。

【０００４】

[0004]汎用計算用にグラフィックスプロセッサを適合させる上での一つの問題は、グラフィックスプロセッサは、通常、比較的低い数値精度用に設計されていることである。高品質の画像は、３２ビット（「単精度」）浮動小数点値、場合によっては１６ビット（「半精度」）浮動小数点値を使用してレンダリングされ得るものであり、機能ユニット及び内部パイプラインは、これらのデータ幅がサポートされるように構成されている。それに対して、多くの汎用計算では、より高い数値精度、例えば、６４ビット（「倍精度」）が要求される。

【０００５】

[0005]より高い数値精度をサポートするために、グラフィックスプロセッサによっては、一連の機械語命令と３２ビット又は１６ビットの機能ユニットとを用いて倍精度の計算を実行するためのソフトウェア手法を使用している。この方法ではスループットが低下し

、例えば、１回の６４ビット乗算演算を完了するのに１００個以上の機械語命令が必要となることがある。このような長い命令列により、グラフィックスプロセッサの倍精度のスループットが大幅に低下することがある。一つの代表的なケースにおいては、グラフィックスプロセッサが倍精度の計算を実行するときのスループットは、高性能のデュアルコアＣＰＵチップによって可能であるスループットの約１／５であると推定される（これと比較して、同じグラフィックスプロセッサは、単精度の計算をデュアルコアＣＰＵのスループットの約１５～２０倍で実行することができる）。ソフトウェアベースのソリューションは非常に遅いため、従来のグラフィックスプロセッサを倍精度の計算に使用することはめったにない。

【０００６】

[0006]別のソリューションは、単純に、グラフィックスプロセッサの算術演算回路の全てを、倍精度のオペランドを扱うよう十分な広さにすることである。これにより、倍精度演算におけるグラフィックスプロセッサのスループットが増大して、単精度におけるスループットに匹敵するものとなる。しかしながら、グラフィックスプロセッサは、一般に、並列演算をサポートするために、各算術演算回路の数十個の複製を備えており、このような各回路のサイズを増大させることによって、チップ面積、コスト、及び電力消費量が相当に増大する。

【０００７】

[0007]更に別のソリューションは、所有者が同一の同時係属中の米国特許出願第１１／３５９，３５３号明細書（出願日：２００６年２月２１日）に記載されているように、単精度の算術演算回路を利用して倍精度の演算を実行することである。この方法においては、単精度の機能ユニットに含まれている特殊なハードウェアを使用して、倍精度の演算を反復的に実行する。この方法は、ソフトウェアベースのソリューションよりも相当に高速であるが（スループットは、単精度のスループットの１／１００以下ではなく、例えば１／４となり得る）、チップの設計が大幅に複雑となり得る。さらに、単精度の演算と倍精度の演算との間で同じ機能ユニットを共有する結果として、同じ機能ユニットを必要とする命令が多すぎる場合に、そのユニットがパイプラインにおけるボトルネックとなることがある。

【発明の概要】

【０００８】

[0008]本発明の実施形態は、グラフィックスプロセッサにおける倍精度の算術演算を直接的にサポートする。レンダリングに使用される単精度の機能ユニットに加えて、多目的の倍精度の機能ユニットが提供される。倍精度の機能ユニットは、倍精度の入力に対する融合型積和演算を含む複数の異なる演算を、少なくとも倍精度の幅であるデータ経路及び／又は論理回路を使用して、実行することができる。倍精度の機能ユニット及び単精度の機能ユニットは、共通の命令発行回路によって制御することができ、コアに含まれている倍精度の機能ユニットの複製の数を、単精度の機能ユニットの複製の数よりも少なくすることができ、これによって、倍精度のサポートを加えることによるチップ面積に対する影響が低減する。

【０００９】

[0009]本発明の一態様によると、グラフィックスプロセッサは、画像データを生成するようになっているレンダリングパイプラインを有する。このレンダリングパイプラインは、単精度オペランド用に動作するものであり、複数の同時スレッドを実行するようになっている処理コアを含んでいる。処理コアは、倍精度の入力オペランドのセットに対して複数の倍精度演算のうちの一つを選択的に実行するようになっている多目的の倍精度機能ユニットを含んでいる。この多目的の倍精度機能ユニットは、少なくとも一つの算術演算論理回路を含んでおり、倍精度機能ユニットの算術演算論理回路の全ては、倍精度において動作するための十分な広さになっている。いくつかの実施形態においては、倍精度機能ユニットは、倍精度演算のそれぞれが同数のクロックサイクルで完了するように構成されている。また、倍精度機能ユニットは、倍精度演算のうちの任意の一つを完了するために要

求される時間（例えば、クロックサイクルの数）がアンダーフロー条件又はオーバーフロー条件によって影響されないように、構成されていてもよい。

【００１０】

[0010]様々な倍精度演算及び倍精度演算の組合せをサポートすることができる。一実施形態においては、倍精度演算は、二つの倍精度オペランドを加算する加算演算と、二つの倍精度オペランドを乗算する乗算演算と、第１の倍精度オペランドと第２の倍精度オペランドとの積を計算し、次いで、この積に第３の倍精度オペランドを加算する、融合型積和演算と、を含んでいる。サポートすることのできる別の倍精度演算としては、第１のオペランドと第２のオペランドとに対して比較テストを実行し、この比較テストが満たされているかを示すブール結果を生成する倍精度比較（ＤＳＥＴ）演算、二つの倍精度入力オペランドのうち大きい方のオペランドを返す倍精度最大値（ＤＭＡＸ）演算、二つの倍精度入力オペランドのうち小さい方のオペランドを返す倍精度最小値（ＤＭＩＮ）演算、が挙げられる。さらには、倍精度形式から倍精度以外の形式に（又はその逆に）オペランドを変換する形式変換演算もサポートすることができる。

【００１１】

[0011]本発明の別の態様によると、グラフィックスプロセッサは、画像データを生成するようになっているレンダリングパイプラインを含んでいる。このレンダリングパイプラインは、複数の同時スレッドを実行するようになっている処理コアを含んでいる。処理コアは、一以上の単精度オペランドに対して算術演算を実行するようになっている単精度の機能ユニットと、倍精度入力オペランドのセットに対して積和演算を実行して倍精度の結果を提供するようになっている倍精度の融合型積和演算（ＤＦＭＡ）機能ユニットと、を含んでいる。このＤＦＭＡ機能ユニットは、ＤＦＭＡパイプラインを含んでおり、ＤＦＭＡパイプラインは、当該ＤＦＭＡパイプラインを通る単一のパスで積和演算を実行するのに十分に広いデータ経路を有することが好適である。例えば、ＤＦＭＡ機能ユニットは、倍精度の二つの仮数の積を１回の反復で計算するようになっている乗算器と、倍精度の二つの仮数の和を１回の反復で計算するようになっている加算器と、を含んでいてもよい。

【００１２】

[0012]さらに、ＤＦＭＡ機能ユニットを、別の演算を実行するように構成することもできる。例えば、いくつかの実施形態においては、ＤＦＭＡは、一組の倍精度入力オペランドに対して乗算演算を実行して倍精度の結果を提供するように構成されている。いくつかの実施形態においては、乗算演算と融合型積和演算は、それぞれ、同数のクロックサイクルで完了する。同様に、ＤＦＭＡ機能ユニットを、一組の倍精度入力オペランドに対して加算演算を実行して倍精度の結果を提供するように構成することができる。一実施形態においては、加算演算と融合型積和演算は、それぞれ、同数のクロックサイクルで完了する。

【００１３】

[0013]いくつかの実施形態においては、処理コアは、並列に動作するようになっている第１の機能ユニットの複数（Ｐ個）の複製と、ＤＦＭＡ機能ユニットの複数（Ｎ個）の複製と、を含んでいる。ここで、数Ｐは数Ｎよりも大きい。一実施形態においては、数Ｎは１である。

【００１４】

[0014]処理コアは、入力マネージャ回路を含んでいてもよく、当該入力マネージャ回路は、ＤＦＭＡ機能ユニット用のＰ個のセットの倍精度入力オペランドを集めて、Ｐ個のセットの倍精度オペランドのうち異なるセットの倍精度オペランドを、異なる（例えば、連続する）クロックサイクルで、ＤＦＭＡ機能ユニットに提供するようになっている。さらに、入力マネージャ回路は、第１の機能ユニット用のＰ個のセットの単精度入力オペランドを集めて、Ｐ個のセットの単精度オペランドのうちの異なるセットの単精度オペランドを、第１の機能ユニットのＰ個の複製の各々に並列に提供してもよい。

【００１５】

[0015]本発明の概念及び利点は、以下の詳細な説明と添付の図面から深く理解されるで

あろう。
【図面の簡単な説明】
【００１６】
【図１】本発明の実施形態によるコンピュータシステムのブロック図である。
【図２】本発明の実施形態による、グラフィックス処理ユニットにおいて実施することのできるレンダリングパイプラインのブロック図である。
【図３】本発明の実施形態による実行コアのブロック図である。
【図４】本発明の実施形態による倍精度機能ユニットによって実行することのできる倍精度算術演算と、倍精度比較演算と、形式変換演算とを一覧表示した図である。
【図５】本発明の実施形態による倍精度機能ユニットの単純化したブロック図である。
【図６】図５の倍精度機能ユニットにおけるオペランド準備ブロックのブロック図である。
【図７】図５の倍精度機能ユニットにおける指数経路のブロック図である。
【図８】図５の倍精度機能ユニットにおける仮数経路のブロック図である。
【図９】図５の倍精度機能ユニットにおける符号経路のブロック図である。
【図１０】図５の倍精度機能ユニットにおける出力セクションのブロック図である。
【図１１】本発明の実施形態による実行コアのブロック図である。
【図１２】本発明の実施形態による、倍精度機能ユニットのためのオペランドの順序付けを示すブロック図である。
【詳細な説明】
【００１７】
[0028]本発明の実施形態は、専用の倍精度（例えば、６４ビット）機能ユニットを含むグラフィックスプロセッサを提供する。一実施形態においては、倍精度機能ユニットは、加算演算と、乗算演算と、融合型積和演算と、更には、倍精度比較と、倍精度形式とそれ以外の形式との間の形式変換とを実行することができる。
【００１８】
Ｉ．システムの概要
Ａ．コンピュータシステムの概要
[0029]図１は、本発明の実施形態によるコンピュータシステム１００のブロック図である。コンピュータシステム１００は、メモリブリッジ１０５を含むバス経路を介して通信する中央処理装置（ＣＰＵ）１０２及びシステムメモリ１０４を含んでいる。メモリブリッジ１０５は、例えば、従来のノースブリッジチップであってもよく、バス又はその他の通信経路１０６（例：ＨｙｐｅｒＴｒａｎｓｐｏｒｔリンク）を介してＩ／Ｏ（入力／出力）ブリッジ１０７に接続されている。Ｉ／Ｏブリッジ１０７は、例えば、従来のサウスブリッジチップであってもよく、一以上のユーザ入力装置１０８（例：キーボード、マウス）からユーザ入力を受け取り、当該入力をバス１０６及びメモリブリッジ１０５を介してＣＰＵ　１０２に転送する。視覚的出力は、ピクセルベースの表示装置１１０（例：従来のＣＲＴ又はＬＣＤベースのモニター）によって提供され、この表示装置１１０は、バス又はその他の通信経路１１３（例：ＰＣＩ　Ｅｘｐｒｅｓｓ（ＰＣＩ－Ｅ）、アクセラレーテッドグラフィックスポート（ＡＧＰ）リンク）を介してメモリブリッジ１０５に接続されたグラフィックスサブシステム１１２の制御下で動作する。Ｉ／Ｏブリッジ１０７には、システムディスク１１４も接続されている。スイッチ１１６は、Ｉ／Ｏブリッジ１０７と、その他のコンポーネント、例えば、ネットワークアダプタ１１８、様々なアドインカード１２０，１２１との間の接続を提供している。Ｉ／Ｏブリッジ１０７には、それ以外のコンポーネント（図示していない）として、ＵＳＢ接続装置又はその他のポート接続装置、ＣＤドライブ、ＤＶＤドライブなどを接続することもできる。様々なコンポーネントの間のバス接続は、バスプロトコル（例えば、ＰＣＩ（ペリフェラルコンポーネントインターコネクト）、ＰＣＩ－Ｅ、ＡＧＰ、ＨｙｐｅｒＴｒａｎｓｐｏｒｔ）、又はその他の任意のバス通信プロトコル或いはポイントツーポイント通信プロトコルを使用して実施することができ、異なるデバイスの間の接続には、この技術分野において公知であるよ

うに、様々なプロトコルを使用することができる。

【００１９】

[0030]グラフィックス処理サブシステム１１２は、グラフィックス処理ユニット（ＧＰＵ）１２２とグラフィックスメモリ１２４とを含んでおり、これらは、例えば、一以上の集積回路デバイス、例えば、プログラマブルプロセッサ、特定用途向け集積回路（ＡＳＩＣ）、メモリデバイスを使用して実施することができる。ＧＰＵ　１２２は、様々なタスクとして、ＣＰＵ　１０２及び／又はシステムメモリ１０４によってメモリブリッジ１０５及びバス１１３を介して供給されるグラフィックスデータからピクセルデータを生成すること、グラフィックスメモリ１２４と対話してピクセルデータを格納及び更新すること等に関連するタスクを実行するように、構成することができる。例えば、ＧＰＵ　１２２は、ＣＰＵ　１０２上で実行される様々なプログラムによって提供される２次元又は３次元のシーンデータから、ピクセルデータを生成することができる。さらに、ＧＰＵ　１２２は、メモリブリッジ１０５を介して受け取るピクセルデータを、さらなる処理を実行して、又はそのまま、グラフィックスメモリ１２４に格納することができる。ＧＰＵ　１２２は、グラフィックスメモリ１２４からのピクセルデータを表示装置１１０に提供するように構成されたスキャンアウトモジュールも含んでいる。

【００２０】

[0031]さらに、ＧＰＵ　１２２は、データ処理タスクのために、汎用計算を実行するように構成されている。当該タスクには、グラフィックスアプリケーションに関連するタスク（例えば、ビデオゲーム等における物理モデリング）と、グラフィックスアプリケーションには関連しないタスクとが含まれる。汎用計算の場合には、ＧＰＵ　１２２は、システムメモリ１０４又はグラフィックスメモリ１２４から入力データを読み取り、一以上のプログラムを実行して当該データを処理し、出力データをシステムメモリ１０４又はグラフィックスメモリ１２４に書き込むことが好適である。ＧＰＵ　１２２は、汎用計算において使用するための一以上の倍精度の融合型積和演算ユニット（図１には示していない）と、レンダリング演算時に使用する別の単精度の機能ユニットとを含むことが好適である。

【００２１】

[0032]ＣＰＵ　１０２は、システム１００のマスタープロセッサとして動作し、他のシステムコンポーネントの動作を制御及び調整する。具体的には、ＣＰＵ　１０２は、ＧＰＵ　１２２の動作を制御するコマンドを発行する。いくつかの実施形態においては、ＣＰＵ　１０２は、ＧＰＵ　１２２用のコマンドストリームをコマンドバッファに書き込む。このコマンドバッファは、システムメモリ１０４、グラフィックスメモリ１２４、又は、ＣＰＵ　１０２及びＧＰＵ　１２２の双方からアクセス可能な別の記憶域中に存在し得る。ＧＰＵ　１２２は、コマンドバッファからコマンドストリームを読み取り、ＣＰＵ　１０２の動作とは非同期にコマンドを実行する。これらのコマンドは、画像を生成するための従来のレンダリングコマンドを含み得るものであり、また、ＣＰＵ　１０２上で実行されるアプリケーションが画像の生成には関連しないデータ処理をＧＰＵ　１２２の計算能力を利用して行うことを可能にする汎用計算コマンドを含み得る。

【００２２】

[0033]図１に示したシステムは例示を目的としており、変形及び変更が可能であることが理解されるであろう。バスのトポロジー（例えば、ブリッジの数、ブリッジの配置編成）は、必要に応じて修正することができる。例えば、いくつかの実施形態においては、システムメモリ１０４が、ブリッジを通じてではなく直接的にＣＰＵ　１０２に接続されており、他のデバイスが、メモリブリッジ１０５及びＣＰＵ　１０２を介してシステムメモリ１０４と通信する。別のトポロジーにおいては、グラフィックスサブシステム１１２が、メモリブリッジ１０５ではなくＩ／Ｏブリッジ１０７に接続される。さらに別の実施形態においては、Ｉ／Ｏブリッジ１０７とメモリブリッジ１０５とを一つのチップに統合することができる。図１に示した特定のコンポーネントはオプションであり、例えば、任意の数のアドインカード或いは周辺装置をサポートすることができる。いくつかの実施形態

においては、スイッチ１１６が省かれ、ネットワークアダプタ１１８及びアドインカード１２０，１２１がＩ／Ｏブリッジ１０７に直接接続される。

【００２３】

[0034]さらに、ＧＰＵ　１２２と、システム１００の他のコンポーネントとの間の接続を変更することも可能である。いくつかの実施形態においては、グラフィックスサブシステム１１２が、システム１００の拡張スロットに挿入することのできるアドインカードとして実施される。別の実施形態においては、ＧＰＵが、バスブリッジ（例えば、メモリブリッジ１０５、Ｉ／Ｏブリッジ１０７）と共に一つのチップに統合される。更に別の実施形態においては、ＧＰＵ　１２２の要素のいくつか又は全てをＣＰＵ　１０２と統合することができる。

【００２４】

[0035]ＧＰＵは、任意の量のローカルグラフィックスメモリを備えていてもよく（ローカルメモリを設けなくてもよい）、ローカルメモリ及びシステムメモリを任意の組合せで使用することができる。例えば、ユニファイドメモリアーキテクチャ（ＵＭＡ）の実施形態においては、専用のグラフィックスメモリデバイスを設けず、ＧＰＵは、システムメモリを独占的又は略独占的に使用する。ＵＭＡの実施形態においては、ＧＰＵをバスブリッジチップに組み込むことができ、又は、ＧＰＵをブリッジチップ及びシステムメモリに接続する高速バス（例：ＰＣＩ－Ｅ）を備えた個別のチップとして、ＧＰＵを提供することが可能である。

【００２５】

[0036]さらに、例えば、１枚のグラフィックスカードに複数のＧＰＵを含めることによって、或いは、複数のグラフィックスカードをバス１１３に接続することによって、任意の数のＧＰＵをシステムに含めることができることも理解されたい。複数のＧＰＵを、同じ表示装置又は複数の異なる表示装置への画像を生成するように並列に動作させてもよく、或いは、一つのＧＰＵが画像を生成するように動作する一方で、別のＧＰＵが汎用計算（後から説明する倍精度の計算を含む）を実行してもよい。

【００２６】

[0037]さらには、本発明の態様を具現化するＧＰＵは、様々な装置、例えば、汎用コンピュータシステム、ビデオゲームコンソール及びその他の特殊用途のコンピュータシステム、ＤＶＤプレーヤー、携帯機器（携帯電話、携帯情報端末など）に組み込むことができる。

【００２７】

Ｂ．レンダリングパイプラインの概要

[0038]図２は、本発明の実施形態による、図１のＧＰＵ　１２２において実施することのできるレンダリングパイプライン２００のブロック図である。この実施形態においては、レンダリングパイプライン２００は、適用可能なグラフィックス関連のプログラム（例えば、頂点シェーダー、ジオメトリシェーダー、ピクセルシェーダーのうちの少なくとも一つ）と汎用計算プログラムとが、同じ並列処理ハードウェア（本明細書においては「マルチスレッドコアアレイ（ｍｕｌｔｉｔｈｒｅａｄｅｄ　ｃｏｒｅ　ａｒｒａｙ）」２０２と称する）を使用して実行されるアーキテクチャを使用して、実施されている。

【００２８】

[0039]レンダリングパイプライン２００は、マルチスレッドコアアレイ２０２に加えて、フロントエンド２０４及びデータアセンブラ２０６と、セットアップモジュール２０８と、ラスタライザ２１０と、カラーアセンブリモジュール２１２と、ラスターオペレーションモジュール（ＲＯＰ）２１４と、を含んでいる。これらのコンポーネントのそれぞれは、従来の集積回路技術又はその他の技術を使用して実施することができる。

【００２９】

[0040]レンダリング演算においては、フロントエンド２０４が、状態情報（ＳＴＡＴＥ）と、コマンド（ＣＭＤ）と、ジオメトリデータ（ＧＤＡＴＡ）とを、例えば図１のＣＰＵ　１０２から受け取る。いくつかの実施形態においては、ＣＰＵ　１０２は、ジオメト

リデータを直接提供するのではなく、ジオメトリデータが格納されているシステムメモリ１０４中の位置への参照情報を提供し、データアセンブラ２０６が、システムメモリ１０４からデータを取得する。レンダリング演算においては、状態情報、コマンド、及びジオメトリデータは、基本的に従来の性質のものとすることができ、これらを使用することにより、レンダリング後の（一以上の）必要な画像（例えば、シーンのジオメトリ、ライティング、シェーディング、テクスチャ、モーション、カメラパラメータのうちの一つ以上）を定義することができる。

【００３０】

[0041]状態情報及びレンダリングコマンドは、レンダリングパイプライン２００の様々なステージにおける処理パラメータ及びアクションを定義する。フロントエンド２０４は、状態情報及びレンダリングコマンドを、制御経路（図示していない）を介して、レンダリングパイプライン２００の別のコンポーネントに導く。この技術分野において公知であるように、これらのコンポーネントは、処理時にアクセスされる様々な制御レジスタに値を格納し、又は制御レジスタ内の値を更新することによって、受け取った状態情報に応答することができ、パイプライン内で受け取ったデータを処理することによって、レンダリングコマンドに応答することができる。

【００３１】

[0042]フロントエンド２０４は、ジオメトリデータをデータアセンブラ２０６に導く。データアセンブラ２０６は、ジオメトリデータをフォーマッティングし、それをマルチスレッドコアアレイ２０２におけるジオメトリモジュール２１８への配送用に準備する。

【００３２】

[0043]ジオメトリモジュール２１８は、頂点データに対して頂点シェーダープログラム及び／又はジオメトリシェーダープログラムを実行するよう、マルチスレッドコアアレイ２０２におけるプログラマブル処理エンジン（図示していない）に命令する。これらのプログラムは、フロントエンド２０４によって提供される状態情報に応答して選択される。頂点シェーダープログラム及び／又はジオメトリシェーダープログラムは、この技術分野において公知であるようにレンダリングアプリケーションによって指定することができ、異なる頂点及び／又はプリミティブに、異なるシェーダープログラムを適用することができる。いくつかの実施形態においては、頂点シェーダープログラム及びジオメトリシェーダープログラムは、マルチスレッドコアアレイ２０２における同じプログラマブル処理コアを使用して実行される。従って、一つの処理コアは、ある時点においては頂点シェーダーとして動作し、頂点プログラムの命令を受け取って実行することができ、別の時点においては、同じ処理コアがジオメトリシェーダーとして動作し、ジオメトリプログラムの命令を受け取って実行することができる。処理コアはマルチスレッド化することができ、異なるタイプのシェーダープログラムを実行する異なるスレッドを、マルチスレッドコアアレイ２０２において同時に進行させることができる。

【００３３】

[0044]頂点シェーダープログラム及び／又はジオメトリシェーダープログラムが実行された後に、ジオメトリモジュール２１８は、処理されたジオメトリデータ（ＧＤＡＴＡ’）をセットアップモジュール２０８に渡す。セットアップモジュール２０８は、一般的に従来の設計のモジュールとすることができるものであり、各プリミティブのクリップ空間座標又はスクリーン空間座標からエッジ方程式（ｅｄｇｅ　ｅｑｕａｔｉｏｎｓ）を生成する。エッジ方程式は、スクリーン空間内の点がプリミティブの内側であるか外側であるかを判定する目的に好適に使用可能である。

【００３４】

[0045]セットアップモジュール２０８は、プリミティブ（ＰＲＩＭ）のそれぞれをラスタライザ２１０に提供する。ラスタライザ２１０（一般的に従来の設計とすることができる）は、どのピクセル（存在時）がプリミティブによってカバーされているかを、例えば従来のスキャン変換アルゴリズム（ｓｃａｎ－ｃｏｎｖｅｒｓｉｏｎ　ａｌｇｏｒｉｔｈｍｓ）を使用して判定する。本明細書において使用する「ピクセル」（又は「フラグメン

ト」）は、一般には、一つのカラー値が決定される、２次元スクリーン空間内の領域を意味する。ピクセルの数及び配置は、レンダリングパイプライン２００の設定可能なパラメータとすることができ、特定の表示装置の画面解像度に相関させてもよく、又は相関させなくてもよい。

【００３５】

[0046]ラスタライザ２１０は、どのピクセルがプリミティブによってカバーされているかを判定した後、プリミティブ（ＰＲＩＭ）と、そのプリミティブによってカバーされているピクセルのスクリーン座標（Ｘ，Ｙ）のリストとを、カラーアセンブリモジュール２１２に提供する。カラーアセンブリモジュール２１２は、ラスタライザ２１０から受け取ったプリミティブとカバレッジ情報とを、プリミティブの頂点の属性（例：色成分、テクスチャ座標、面法線）に関連付けて、属性のいくつか又は全てをスクリーン座標空間内の位置の関数として定義する平面方程式（又はその他の適切な方程式）を生成する。

【００３６】

[0047]これらの属性方程式は、ピクセルシェーダープログラムにおいて、プリミティブの中の任意の位置における属性の値を計算するために好適に使用可能である。当該方程式は、従来の手法を使用して生成することができる。例えば、一実施形態においては、カラーアセンブリモジュール２１２は、属性Ｕそれぞれについて$U = Ax + By + C$という形式の平面方程式の係数Ａ、Ｂ、及びＣを生成する。

【００３７】

[0048]カラーアセンブリモジュール２１２は、ピクセルの少なくとも一つのサンプリング位置をカバーしているプリミティブそれぞれの属性方程式（ＥＱＳ、例えば、平面方程式の係数Ａ、Ｂ、及びＣを含んでいることができる）と、カバーされているピクセルのスクリーン座標（Ｘ，Ｙ）のリストとを、マルチスレッドコアアレイ２０２におけるピクセルモジュール２２４に提供する。ピクセルモジュール２２４は、マルチスレッドコアアレイ２０２におけるプログラマブル処理エンジン（図示していない）に命令して、プリミティブによってカバーされているピクセルそれぞれに対して一以上のピクセルシェーダープログラムを実行させる。これらのプログラムは、フロントエンド２０４によって提供される状態情報に応答して選択される。あらゆる所与のセットのピクセルに対して使用するピクセルシェーダープログラムは、頂点シェーダープログラム及びジオメトリシェーダープログラムと同様に、レンダリングアプリケーションが指定することができる。

【００３８】

[0049]ピクセルシェーダープログラムは、頂点シェーダープログラム及び／又はジオメトリシェーダープログラムを実行する同じプログラマブル処理エンジンを使用して、マルチスレッドコアアレイ２０２において好適に実行される。従って、一つの処理エンジンは、ある時点においては頂点シェーダーとして動作し、頂点プログラムの命令を受け取って実行することができ、別の時点においては、同じ処理エンジンがジオメトリシェーダーとして動作し、ジオメトリプログラムの命令を受け取って実行することができ、更に別の時点においては、同じ処理エンジンがピクセルシェーダーとして動作し、ピクセルシェーダープログラムの命令を受け取って実行することができる。

【００３９】

[0050]ピクセル又はピクセルのグループの処理が完了した時点で、ピクセルモジュール２２４は、処理されたピクセル（ＰＤＡＴＡ）をＲＯＰ　２１４に提供する。ＲＯＰ　２１４は、一般的に従来の設計のものであってもよく、ピクセルモジュール２２４から受け取るピクセル値と、フレームバッファ２２６内の構築中の画像のピクセルとを統合する。このフレームバッファ２２６は、例えば、グラフィックスメモリ１２４内に位置し得る。いくつかの実施形態においては、ＲＯＰ　２１４は、ピクセルをマスクすることができ、又は、新しいピクセルと、レンダリングされている画像に以前に書き込まれたピクセルとをブレンドすることができる。デプスバッファ、アルファバッファ、及びステンシルバッファを使用して、レンダリングされている画像に対する、入力されるピクセルそれぞれの寄与（存在時）を決定することもできる。入力されるピクセル値それぞれと、以前に格納

されているピクセル値との適切な組合せに対応するピクセルデータＰＤＡＴＡ’が、再びフレームバッファ２２６に書き込まれる。画像が完成した時点で、フレームバッファ２２６を表示装置にスキャンアウトし、及び／又は、更なる処理を行うことができる。

【００４０】

[0051]汎用計算においては、マルチスレッドコアアレイをピクセルモジュール２２４によって（又はジオメトリモジュール２１８によって）制御することができる。フロントエンド２０４は、例えば図１のＣＰＵ　１０２から状態情報（ＳＴＡＴＥ）及び処理コマンド（ＣＭＤ）を受け取り、これらの状態情報及びコマンドを、制御経路（図示していない）を介して作業配分ユニットに提供する。この作業配分ユニットは、例えばカラーアセンブリモジュール２１２又はピクセルモジュール２２４に組み込むことができる。作業配分ユニットは、マルチスレッドコアアレイ２０２を構成している複数の処理コアの間で処理タスクを分配する。様々な作業配分アルゴリズムを使用することができる。

【００４１】

[0052]処理タスクのそれぞれは、複数の処理スレッドを実行することを含むことが好適であり、この場合、スレッドのそれぞれは同じプログラムを実行する。プログラムは、「グローバルメモリ」（例：システムメモリ１０４、グラフィックスメモリ１２４、又は、ＧＰＵ　１２２及びＣＰＵ　１０２の両方からアクセス可能な任意の別のメモリ）から入力データを読み取るための命令と、少なくともいくつかの倍精度演算を含む様々な演算を入力データに対して実行して出力データを生成するための命令と、出力データをグローバルメモリに書き込むための命令と、を含むことが好適である。具体的な処理タスクについては、本発明において重要ではない。

【００４２】

[0053]図２に記載したレンダリングパイプラインは例示を目的としており、変形及び変更が可能であることが理解されるであろう。このパイプラインは、図示したユニットとは異なるユニットを含むことができ、処理イベントの順序は、本明細書に説明した順序とは異なっていてもよい。さらに、本明細書に説明したモジュールのいくつか又は全ての複数のインスタンスを並列に動作させることができる。このような一実施形態においては、マルチスレッドコアアレイ２０２は、二つ以上のジオメトリモジュール２１８と、それと同じ数の並列に動作するピクセルモジュール２２４とを含む。ジオメトリモジュール及びピクセルモジュールのそれぞれは、マルチスレッドコアアレイ２０２における処理エンジンの異なるサブセットを協働して制御する。

【００４３】

Ｃ．コアの概要

[0054]マルチスレッドコアアレイ２０２は、多数の処理スレッドを並列に実行するようになっている一以上の処理コアを含むことが好適である。ここで、「スレッド」との用語は、特定のセットの入力データに対して実行される特定のプログラムのインスタンスを意味する。例えば、スレッドは、一つの頂点の属性に対して実行される頂点シェーダープログラムのインスタンス、或いは、与えられたプリミティブ及びピクセルに対して実行されるピクセルシェーダープログラムのインスタンス、又は、汎用計算プログラムのインスタンスとすることができる。

【００４４】

[0055]図３は、本発明の実施形態による実行コア３００のブロック図である。実行コア３００は、例えば、上述したマルチスレッドコアアレイ２０２において実施することができ、様々な計算を実行するための任意の一連の命令を実行するように構成されている。いくつかの実施形態においては、同じ実行コア３００を使用して、グラフィックスレンダリングの全ての段階におけるシェーダープログラム（例えば、頂点シェーダープログラム、ジオメトリシェーダープログラム、ピクセルシェーダープログラムのうちの一つ以上）と、汎用計算プログラムとを、実行することができる。

【００４５】

[0056]実行コア３００は、フェッチ・ディスパッチユニット３０２と、発行ユニット３

０４と、倍精度の融合型積和演算（ＤＦＭＡ）ユニット３２０と、ＤＦＭＡ以外の複数（Ｎ個）の機能ユニット（ＦＵ）３２２と、レジスタファイル３２４と、を含んでいる。機能ユニット３２０，３２２のそれぞれは、指定された演算を実行するように構成されている。一実施形態においては、ＤＦＭＡユニット３２０は、後述するように、倍精度の融合型積和演算と、それ以外の倍精度演算とを好適に実施する。なお、コア３００には任意の数のＤＦＭＡユニット３２０を含めることができることを理解されたい。

【００４６】

[0057]ＤＦＭＡ以外の機能ユニット３２２は、基本的に従来の設計のものであってもよく、様々な演算（例えば、単精度の加算演算、乗算演算、ビットごとの論理演算、比較演算、形式変換演算、テクスチャフィルタリング、メモリアクセス（例：ロード動作及び格納動作）、超越関数の近似、補間）をサポートすることができる。機能ユニット３２０，３２２はパイプライン化されていてもよく、これにより、この技術分野において公知であるように、前の命令が終了する前に新しい命令を発行することができる。また、任意の組合せの機能ユニットが提供されていてもよい。

【００４７】

[0058]実行コア３００の動作時、フェッチ・ディスパッチユニット３０２は、命令の格納域（図示していない）から命令を取得し、当該命令をデコードし、そして、当該命令を、関連するオペランド参照又はオペランドデータと一緒に、オペコードとして発行ユニット３０４にディスパッチする。発行ユニット３０４は、命令のそれぞれについて、参照先のオペランドを、例えばレジスタファイル３２４から取得する。命令のオペランドの全てが用意されると、発行ユニット３０４は、オペコード及びオペランドをＤＦＭＡユニット３２０又はＤＦＭＡ以外の機能ユニット３２２に送ることによって、その命令を発行する。発行ユニット３０４は、与えられた命令を実行するための適切な機能ユニットを、オペコードを使用して選択することが好適である。フェッチ・ディスパッチユニット３０２及び発行ユニット３０４は、従来のマイクロプロセッサのアーキテクチャ及び技術を使用して実施することができ、その詳細な説明については、本発明を理解する上で重要ではないため省略する。

【００４８】

[0059]ＤＦＭＡユニット３２０及びＤＦＭＡ以外の機能ユニット３２２は、オペコード及び関連するオペランドを受け取り、それらのオペランドに対して、指定された演算を実行する。結果のデータは、レジスタファイル３２４（又は別の転送先）にデータ転送経路３２６を介して転送することのできる結果値の形式で提供される。レジスタファイル３２４は、いくつかの実施形態においては、特定のスレッドに割り当てられる区域を有するローカルレジスタファイルと、複数のスレッド間でのデータの共有を可能にするグローバルレジスタファイル、とを含んでいる。レジスタファイル３２４は、プログラムの実行時に、入力データ、中間結果、及びその他のデータを格納する目的に使用することができる。レジスタファイル３２４の具体的な実施形態については、本発明において重要ではない。

【００４９】

[0060]一実施形態においては、コア３００はマルチスレッド化されており、例えば、スレッドのそれぞれに関連付けられる現在の状態情報を維持することによって、最大数（例えば、３８４個、７６８個）までのスレッドを同時に実行することができる。コア３００は、例えば、あるクロックサイクルにおいて、頂点スレッドからのプログラム命令を発行した後、別の頂点スレッドからの、又は別のタイプのスレッド（例えば、ジオメトリスレッド、ピクセルスレッドなど）からのプログラム命令を発行することができるように、一つのスレッドから別のスレッドに迅速に切り替えるように設計されていることが好適である。

【００５０】

[0061]図３の実行コアは例示を目的としており、変形及び変更が可能であることが理解されるであろう。プロセッサには任意の数のコアを含めることができ、コアには任意の数の機能ユニットを含めることができる。フェッチ・ディスパッチユニット３０２及び発行

ユニット３０４は、任意の望ましいマイクロアーキテクチャとして、例えば、スケーラアーキテクチャ、スーパースケーラアーキテクチャ、又はベクトルアーキテクチャを実施することができ、インオーダー又はアウトオブオーダーの命令発行方式、投機的実行モード、単一命令複数データ（ＳＩＭＤ）命令発行方式などを必要に応じて採用する。いくつかのアーキテクチャにおいては、発行ユニットは、複数の機能ユニットを対象とする複数のオペコード及びオペランドを含むロング命令ワード、又は、一つの機能ユニットを対象とする複数のオペコード及び／又はオペランドを含むロング命令ワードを受け取る、又は発行する、又は受け取って発行することができる。いくつかのアーキテクチャにおいては、実行コアは、例えばＳＩＭＤ命令を実行するための並列に動作可能な、機能ユニットそれぞれの複数のインスタンスを含むことができる。さらに、実行コアは、パイプライン化された一連の機能ユニットを含むことができ、この場合、一つのステージにおける機能ユニットからの結果は、レジスタファイルに直接転送されるのではなく、後のステージにおける機能ユニットに転送される。このような構成の機能ユニットは、一つのロング命令ワード又は複数の個別の命令によって制御することができる。

【００５１】

[0062]さらに、本発明の教示内容にアクセスするこの技術分野における通常の技能を有する者には、ＤＦＭＡユニット３２０を任意のマイクロプロセッサの中の機能ユニットとして実施することができ、グラフィックスプロセッサ又は何らかの特定のプロセッサ、或いは実行コアのアーキテクチャには限定されないことが認識されるであろう。例えば、ＤＦＭＡユニット３２０を、汎用並列処理ユニット（ｇｅｎｅｒａｌ－ｐｕｒｐｏｓｅ ｐａｒａｌｌｅｌ ｐｒｏｃｅｓｓｉｎｇ ｕｎｉｔ）又は汎用ＣＰＵにおいて実施することができる。

【００５２】

Ｃ．ＤＦＭＡユニットの概要

[0063]本発明の一実施形態によると、実行コア３００は、ＤＦＭＡユニット３２０を含んでおり、当該ＤＦＭＡユニット３２０は、３種類の演算、すなわち、倍精度算術演算、比較演算、倍精度形式とそれ以外の形式との間の形式変換、を実行する。

【００５３】

[0064]ＤＦＭＡユニット３２０は、倍精度浮動小数点形式における入力及び出力を扱い、変換演算においては、倍精度以外の浮動小数点形式又は固定小数点形式における入力及び出力を扱うことが好適である。演算によってオペランドの形式が異なっていてもよい。ＤＦＭＡユニット３２０の実施形態について説明する前に、代表的な形式について定義しておく。

【００５４】

[0065]本明細書において使用する「ｆｐ３２」は、ＩＥＥＥ７５４規格の単精度浮動小数点形式を意味し、この形式においては、正規の浮動小数点数が、符号ビットと、８個の指数ビットと、２３個の仮数部ビットとによって表される。2^{-126}～2^{127}の範囲内の指数が１～２５４の整数を使用して表されるように、指数は１２７だけプラス方向にバイアスされる。「正規」数の場合、２３個の仮数部ビットは、２４ビットの仮数のうちの小数部分として解釈され、整数部分として１が暗黙的に含まれている（本明細書においては、用語「仮数部」は、先頭の１が暗黙的に含まれているときに使用しているのに対し、「仮数」は、先頭の１が明示的に含まれている（該当時）ことを表す目的で使用している）。

【００５５】

[0066]本明細書において使用する「ｆｐ６４」は、ＩＥＥＥ７５４規格の倍精度浮動小数点形式を意味し、この形式においては、正規の浮動小数点数が、符号ビットと、１１個の指数ビットと、５２個の仮数部ビットとによって表される。2^{-1022}～2^{1023}の範囲内の指数が１～２０４６の整数を使用して表されるように、指数は１０２３だけプラス方向にバイアスされる。「正規」数の場合、５２個の仮数部ビットは、５３ビットの仮数のうちの小数部分として解釈され、整数部分として１が暗黙的に含まれている。

【００５６】
[0067]本明細書において使用する「ｆｐ１６」は、グラフィックスにおいて一般的に使用される「半精度」浮動小数点形式を意味し、この形式においては、正規の浮動小数点数が、符号ビットと、５個の指数ビットと、１０個の仮数部ビットとによって表される。2^{-14}～2^{15}の範囲内の指数が１～３０の整数を使用して表されるように、指数は１５だけプラス方向にバイアスされる。「正規」数の場合、１０個の仮数部ビットは、１１ビットの仮数のうちの小数部分として解釈され、整数部分として１が暗黙的に含まれている。

【００５７】
[0068]ｆｐ１６形式、ｆｐ３２形式、及びｆｐ６４形式においては、指数ビット全てが０である数を非正規数（又は「デノーマル」）と称し、仮数における暗黙的な先頭の１を持たないものとして解釈される。このような数は、例えば、計算におけるアンダーフローを表すことができる。指数ビット全てが１であり、仮数部ビット全てが０である（正又は負の）数は、（正又は負の）無限大（ＩＮＦ）と称する。この数は、例えば、計算におけるオーバーフローを表すことができる。指数ビット全てが１であり、仮数部ビットが０以外である数は、非数（ＮａＮ）と称し、例えば、定義されていない値を表す目的に使用することができる。ゼロ（０）も特殊な数とみなされ、指数ビット及び仮数部ビットの全てが０にセットされていることによって表される。ゼロはどちらの符号を持つこともでき、従って、正のゼロ及び負のゼロが許可される。

【００５８】
[0069]固定小数点形式は、本明細書においては、形式が符号付きであるか符号なしであるかを示す先頭の「ｓ」又は「ｕ」と、ビットの総数を表す数（例：１６、３２、６４）とによって指定する。従って、ｓ３２は符号付き３２ビット形式を意味し、ｕ６４は符号なし６４ビット形式を意味し、他も同様である。符号付き形式の場合、２の補数表現を使用することが有利である。本明細書において使用している全ての形式において、最上位ビット（ＭＳＢ）はビットフィールドの左端であり、最下位ビット（ＬＳＢ）は右端である。

【００５９】
[0070]なお、これらの形式は、本明細書においては例示を目的として定義及び使用しており、ＤＦＭＡユニットは、本発明の範囲から逸脱することなく、これらの形式又はそれ以外の形式の任意の組合せをサポートすることができることを理解されたい。具体的には、「単精度」及び「倍精度」は、現在定義されている標準形式に限定されず、任意の二つの異なる小数点形式を意味することができることを理解されたい。倍精度形式（例：ｆｐ６４）は、より広い範囲の浮動小数点数を表し、及び／又は、より高い精度の浮動小数点数を表す関連する単精度形式（例：ｆｐ３２）よりも多くの数のビットを使用する任意の形式を意味する。同様に、「半精度」は、一般的には、より狭い範囲の浮動小数点数を表し、及び／又は、より低い精度の浮動小数点数を表す、関連する単精度形式よりも少ないビットを使用する形式を意味することができる。

【００６０】
[0071]次に、本発明によるＤＦＭＡユニット３２０の実施形態について説明する。図４は、ＤＦＭＡユニット３２０のこの実施形態によって実行することのできる、倍精度の算術演算、比較演算、及び形式変換演算を一覧表示する表４００である。

【００６１】
[0072]セクション４０２は、算術演算を一覧表示している。加算（ＤＡＤＤ）は、ｆｐ６４の二つの入力Ａ及びＣを加算し、ｆｐ６４の和Ａ＋Ｃを返す。乗算（ＤＭＵＬ）は、ｆｐ６４の二つの入力Ａ及びＢを乗算し、ｆｐ６４の積Ａ＊Ｂを返す。融合型積和演算（ＤＦＭＡ）は、ｆｐ６４の三つの入力Ａ，Ｂ，及びＣを受け取り、Ａ＊Ｂ＋Ｃを計算する。この演算は、積Ａ＊Ｂが、Ｃに加算される前に丸められないことにおいて「融合型（ｆｕｓｅｄ）」である。正確な値Ａ＊Ｂを使用することにより、精度が向上し、浮動小数点算術演算の近い将来の規格であるＩＥＥＥ７５４Ｒに準拠する。

【００６２】
[0073]セクション４０４は、比較演算を一覧表示している。最大値演算（ＤＭＡＸ）は、ｆｐ６４のオペランドＡ及びオペランドＢのうちの大きい方を返し、最小値演算（ＤＭＩＮ）は、二つのオペランドのうちの小さい方を返す。二項テスト演算（ＤＳＥＴ）は、倍精度のオペランドＡ及びオペランドＢに対して、複数の二項関係テスト（ｂｉｎａｒｙ　ｒｅｌａｔｉｏｎｓｈｉｐ　ｔｅｓｔｓ）のうちの一つを実行し、そのテストが満たされているかを示すブール値を返す。この実施形態においては、テストすることのできる二項関係としては、より大きい（Ａ＞Ｂ）、より小さい（Ａ＜Ｂ）、等しい（Ａ＝Ｂ）、順序付けできない（Ａ？Ｂ、これは、Ａ又はＢのいずれかがＮａＮである場合に真である）、さらには、否定（例：Ａ≠Ｂ）、及び様々な組合せテスト（A?B、Ａ＜＞Ｂ、Ａ？＝Ｂなど）が挙げられる。
【００６３】
[0074]セクション４０６は、形式変換演算及び丸め演算を一覧している。この実施形態においては、ＤＦＭＡユニット３２０は、ｆｐ６４形式の数を、別の６４ビット形式又は３２ビット形式の数に変換する、又はその逆に変換することができる。Ｄ２Ｆ演算は、オペランドＡをｆｐ６４からｆｐ３２に変換し、Ｆ２Ｄは、オペランドＡをｆｐ３２からｆｐ６４に変換する。Ｄ２Ｉ演算は、オペランドＡを、ｆｐ６４から、ｓ６４、ｕ６４、ｓ３２、及びｕ３２のうちのいずれかの形式に変換する。変換後の形式を特定するのに個別のオペコードを使用することができることを理解されたい。Ｉ２Ｄ演算は、整数オペランドＣを、ｓ６４、ｕ６４、ｓ３２、及びｕ３２のうちのいずれかの形式からｆｐ６４形式に変換する。この場合も、変換前の形式を識別するのに個別のオペコードを使用することができることを理解されたい。この実施形態においては、倍精度形式への変換及び倍精度形式からの変換の全てがＤＦＭＡユニット３２０によってサポートされる。ＤＦＭＡではない機能ユニットは、他の形式変換（例えば、ｆｐ３２形式とｆｐ１６形式との間、ｆｐ３２形式と整数形式との間）を実行することができる。
【００６４】
[0075]Ｄ２Ｄ演算は、ｆｐ６４のオペランドに丸め演算（例えば、ＩＥＥＥ丸めモード）を適用する目的に使用される。これらの演算は、ｆｐ６４のオペランドを、ｆｐ６４形式で表される整数値に丸める。一実施形態においては、サポートされるＤ２Ｄ演算として、切捨て（０の方向に丸める）、シーリング（＋無限大の方向に丸める）、フロア（－無限大の方向に丸める）、最近接（最も近い整数に切り上げる又は切り捨てる）が挙げられる。
【００６５】
[0076]この実施形態においては、ＤＦＭＡユニット３２０は、より高度な数学関数（例えば、除算、剰余、平方根）については、直接的なハードウェアサポートを提供しない。しかしながら、ＤＦＭＡユニット３２０を使用して、ソフトウェアベースでのこれらの演算の実施を高速化することができる。例えば、除算するための一つの一般的な方法では、商ｑ＝ａ／ｂが推定され、次いで、ｔ＝ｑ＊ｂ－ａを使用してその推定がテストされる。ｔが０であれば、商ｑが正しく求められている。０でない場合、ｔの大きさを用いて推定した商ｑが修正され、ｔが０になるまでテストが繰り返される。各反復のテスト結果ｔは、１回のＤＦＭＡ演算（Ａ＝ｑ、Ｂ＝ｂ、Ｃ＝－ａ）を使用して正確に計算することができる。同様に、平方根の場合、一つの一般的な方法では、$r = a^{1/2}$が推定され、次いで、ｔ＝ｒ＊ｒ－ａが計算され、その推定がテストされ、ｔが０でなければｒが修正される。この場合も、各反復のテスト結果ｔは、１回のＤＦＭＡ演算（Ａ＝Ｂ＝ｒ、Ｃ＝－ａ）を使用して正確に計算することができる。
【００６６】
[0077]第ＩＩ節及び第ＩＩＩ節では、図４に示した演算の全てを実行することのできるＤＦＭＡユニット３２０について説明する。第ＩＩ節では、ＤＦＭＡユニット３２０の回路構造について説明し、第ＩＩＩ節では、その回路構造を使用して、図４に一覧した演算を実行する方法について説明する。なお、本明細書に説明するＤＦＭＡユニット３２０は

例示を目的としており、異なる機能の組合せを、回路ブロックの適切な組合せを使用してサポートすることができることを理解されたい。

【００６７】

ＩＩ．ＤＦＭＡユニットの構造

[0078]図５は、図４に示した演算の全てをサポートする本発明の実施形態によるＤＦＭＡユニット３２０の単純化したブロック図である。この実施形態においては、ＤＦＭＡユニット３２０は、全ての演算に使用されるマルチステージパイプラインを実施している。ＤＦＭＡユニット３２０は、プロセッササイクルのそれぞれにおいて、（例えば、図３の発行ユニット３０４から）三つの新しいオペランド（A_0，B_0，C_0）を、オペランド入力経路５０２，５０４，５０６を介して受け取り、実行すべき演算を示すオペコードを、オペコード経路５０８を介して受け取ることができる。この実施形態においては、演算は、図４に示した任意の演算とすることができる。オペコードは、演算のみならず、オペランドの入力形式と、結果に使用する出力形式（入力形式と同じ形式又は異なる形式）とを示すことが好適である。なお、図４に示した演算は、当該演算に関連付けられた複数のオペコードを有することができることに留意されたい。例えば、出力がｓ６４であるＤ２Ｉのための一つのオペコードと、出力がｓ３２であるＤ２Ｉのための別のオペコードとが存在し得る。

【００６８】

[0079]ＤＦＭＡユニット３２０は、演算のそれぞれを全てのパイプラインステージを通じて処理し、６４ビット（特定の形式変換演算の場合には３２ビット）の結果値（ＯＵＴ）を信号経路５１０上に生成し、対応する条件コード（ＣＯＮＤ）を信号経路５１２上に生成する。これらの信号は、アーキテクチャに応じて、例えば、図３に示したようにレジスタファイル３２４、発行ユニット３０４、又はプロセッサコアの別の要素に転送され得る。一実施形態においては、パイプラインステージは、プロセッササイクルに対応する。別の実施形態においては、一つのステージが複数のプロセッササイクルを含み得る。さらに、パイプライン内の複数の異なる経路は、並列に動作することが有利である。

【００６９】

[0080]第ＩＩ．Ａ節では、ＤＦＭＡパイプラインの概要について説明し、第ＩＩ．Ｂ－Ｉ節では、各セクションの回路ブロックについて詳しく説明する。

【００７０】

Ａ．ＤＦＭＡパイプライン

[0081]最初に、パイプラインについて、ＤＦＭＡ演算時に回路ブロックがどのように使用されるかに関連して説明する。オペランド準備ブロック５１４は、（まだｆｐ６４形式ではないオペランドについて）オペランドのフォーマッティングと、特殊数の検出とを実行する。さらに、オペランド準備ブロック５１４は、入力されるｆｐ６４のオペランドから、仮数ビット（Ａｍ，Ｂｍ，Ｃｍ）と、指数ビット（Ａｅ，Ｂｅ，Ｃｅ）と、符号ビット（Ａｓ，Ｂｓ，Ｃｓ）とを取り出す。一実施形態においては、オペランドの無効な組合せは存在しない。演算において使用されないオペランドは、単に無視することができる。

【００７１】

[0082]仮数経路５１６は、仮数Ａｍと仮数Ｂｍの積を計算する。これと並列に、指数経路５１８は、指数Ａｅ及び指数Ｂｅを使用して、積Ａ＊ＢとオペランドＣとの間の相対的な位置合わせ量を求め、オペランドＣ用の位置合わせ後の仮数（Ｃ＿ａｌｉｇｎ）を仮数経路５１６に供給する。仮数経路５１６は、Ｃ＿ａｌｉｇｎを積Ａｍ＊Ｂｍに加算し、次いで、結果を正規化する。仮数経路５１６は、この正規化に基づいて、位置合わせ信号（ＡＬＩＧＮ＿ＮＯＲＭ）を指数経路５１８に戻し、指数経路５１８は、このＡＬＩＧＮ＿ＮＯＲＭ信号と、指数Ａｅ，Ｂｅ，及びＣｅとを一緒に使用して、最終結果の指数を求める。

【００７２】

[0083]符号経路５２０は、オペランド準備ブロック５１４から符号ビットＡｓ，Ｂｓ，及びＣｓを受け取り、結果の符号を求める。仮数経路５１６は、結果が０である場合を検

出し、結果ゼロ（Ｒ＿ＺＥＲＯ）を符号経路５２０に提供する。

【００７３】

[0084]出力セクション５２２は、仮数経路５１６からの結果の仮数Ｒｍと、指数経路５１８からの結果の指数Ｒｅと、符号経路５２０からの結果の符号Ｒｓとを受け取る。さらに、出力セクション５２２は、オペランド準備ブロック５１４から特殊数信号（ＳＰＣ）を受け取る。出力セクション５２２は、これらの情報に基づいて、出力経路５１０に提供することができるように最終結果（ＯＵＴ）をフォーマッティングし、条件コード（ＣＯＮＤ）を出力経路５１２上に生成する。条件コード（結果よりも少ない数のビットを含んでいることが好適である）は、結果の特性に関する一般情報を伝える。例えば、条件コードは、結果が正である、負である、０である、ＮａＮである、無限大である、或いは非正規であるかを示すビットを含み得る。この技術分野において公知であるように、結果と一緒に条件コードが提供される場合、その結果を使用する下流の要素は、場合によっては、その処理において結果自体ではなく条件コードを使用することができる。いくつかの実施形態においては、演算の実行時に例外又はその他のイベントが発生したことを、条件コードを使用して示すことができる。別の実施形態においては、条件コードを完全に省くことができる。

【００７４】

[0085]なお、「仮数経路」、「指数経路」などの名称は、特定の演算（例：ＤＦＭＡ）時に経路それぞれの様々な回路ブロックによって実行される機能を示唆するものであるが、任意の内部データ経路上の回路ブロックを、演算に依存する方式で様々な用途に利用することができることを理解されたい。例については後述する。

【００７５】

[0086]データ経路に加えて、ＤＦＭＡユニット３２０は、制御経路（図５には制御ブロック５３０として表してある）も提供する。制御ブロック５３０は、オペコードを受け取り、オペコードに依存する様々な制御信号（この図ではまとめて「ＯＰＣＴＬ」として表してある）を生成する。この制御信号は、パイプラインを通じたデータ伝搬と同期して回路ブロックのそれぞれに伝搬され得る（様々な回路ブロックまでのＯＰＣＴＬ信号の接続は図５には示していない）。後述するように、ＯＰＣＴＬ信号を使用して、ＤＦＭＡユニット３２２０の様々な回路ブロックの動作をオペコードに応答して有効化する、無効化する、及びその他の制御を行うことにより、複数の異なる演算を同じパイプライン要素を使用して実行することを可能とする。本明細書において言及する様々なＯＰＣＴＬ信号は、オペコード自体を含み得るものであり、又は、例えば、制御ブロック５３０において実施されている組合せ論理回路によってオペコードから導かれる何らかの別の信号を含み得るものである。いくつかの実施形態においては、制御ブロック５３０は、いくつかのパイプラインステージにおける複数の回路ブロックを使用して実施することができる。なお、一つの演算時に複数の異なるブロックに提供されるＯＰＣＴＬ信号は、同じ信号、又は異なる信号とすることができることを理解されたい。この技術分野における通常の技能を有する者には、本開示に基づいて、適切なＯＰＣＴＬ信号を構築することができるであろう。

【００７６】

[0087]一つのステージの複数の回路ブロックに要求される処理時間は、回路ブロックごとに異なることがあり、一つのステージに要求される時間は、演算によって異なり得ることに留意されたい。従って、ＤＦＭＡユニット３２０は、一つのパイプラインステージから次のステージまでの複数の異なる経路上のデータ伝搬を制御する目的で、様々なタイミング・同期回路（図５には示していない）を含むことも可能である。また、任意の適切なタイミング回路（例：ラッチ、送信ゲート）が使用されてもよい。

【００７７】

Ａ．オペランドの準備

[0088]図６は、本発明の実施形態によるオペランド準備ブロック５１４のブロック図である。オペランド準備ブロック５１４は、入力オペランドＡ，Ｂ，及びＣを受け取り、仮数部分（Ａｍ，Ｂｍ，Ｃｍ）を仮数経路５１６に提供し、指数部分（Ａｅ，Ｂｅ，Ｃｅ）

を指数経路５１８に提供し、符号ビット（Ａｓ，Ｂｓ，Ｃｓ）を符号経路５２０に提供する。

【００７８】

[0089]オペランドＡ，Ｂ，及びＣは、それぞれのＮａＮ検出ブロック６１２，６１４，６１６と、それぞれの絶対値／符号反転ブロック６１８，６２０，６２２において受け取られる。ＮａＮ検出ブロック６１２，６１４，６１６のそれぞれは、受け取ったオペランドがＮａＮ（指数ビット全てが１であり、仮数部ビットが０以外である）であるかを判定し、対応する制御信号を生成する。

【００７９】

[0090]絶対値／符号反転ブロック６１８，６２０，６２２は、ＯＰＣＴＬ信号（図示していない）に応答してオペランドの符号ビットを反転させる目的に使用することができる。例えば、図４に一覧表示した演算において、オペランドの負数又はオペランドの絶対値を使用することが指定されることがある。ブロック６１８，６２０，６２２は、符号ビットを反転させてオペランドの符号を反転する、又は符号ビットを負でない状態（ＩＥＥＥ７５４形式の場合には０）に強制することができる。入力オペランドがＮａＮである場合、対応する絶対値／符号反転ブロック６１８，６２０，６２２は、そのＮａＮを更に「クワイエット型にし」（例えば、仮数部の先頭ビットを１にセットすることによる）、符号ビットをそのままにする。絶対値／符号反転ブロック６１８，６２０，６２２は、それぞれの出力をオペランド選択マルチプレクサ６３２，６３４，６３６に提供する。

【００８０】

[0091]倍精度算術演算の場合、絶対値／符号反転ブロック６１８によって生成されるオペランドＡ，Ｂ，及びＣを直接的に使用することができる。比較演算の場合、Ａ／Ｂ比較回路６２４がオペランドＡとオペランドＢとを比較する。一実施形態においては、絶対値／符号反転ブロック６２０がオペランドＢの符号を反転し、Ａ／Ｂ比較回路６２４が、Ａと－Ｂとの和を、これらがあたかも固定小数点数であるかのように計算する。結果が正であればＡがＢよりも大きく、結果が負であればＡがＢよりも小さく、結果が０であればＡはＢに等しい。さらに、Ａ／Ｂ比較回路６２４は、ＮａＮ検出回路６１２及び６１４からのＮａＮ情報を受け取ることができる（これらの経路は図６には示していない）。Ａ又はＢのいずれか（又は双方）がＮａＮであるならば、Ａ及びＢは「順序付けできない」。結果の情報は制御論理回路６３０に提供される。制御論理回路６３０は、結果情報を信号Ｒ＿ＴＥＳＴとして出力セクション５２２に提供し、更に、制御信号をオペランド選択マルチプレクサ６３２，６３４，６３６に提供する。

【００８１】

[0092]形式変換オペランドの場合、入力はｆｐ６４形式ではないことがある。ｆｐ３２抽出回路６２６は、Ｆ２Ｄ演算時にアクティブである。ｆｐ３２抽出回路６２６は、オペランドＡを受け取って、入力がｆｐ３２の非正規数であるかのテスト全てを実行する。さらに、ｆｐ３２抽出回路６２６は、受け取ったオペランドの仮数部フィールドを２３ビットから５２ビットに拡張する（例えば、末尾の０を追加することによる）。ｆｐ３２抽出回路６２６は、ｆｐ３２の８ビットの指数を１１ビットに拡張し、指数のバイアスを１２７から１０２３に増大させる（例えば、ｆｐ３２の指数に８９６を加えることよる）。

【００８２】

[0093]符号なし／符号付き（Ｕ／Ｓ）抽出回路６２８は、Ｉ２Ｄ演算時にアクティブである。Ｕ／Ｓ抽出回路６２８は、ｕ３２、ｓ３２、ｕ６４、ｓ６４のいずれかの形式の固定小数点オペランドＣを受け取り、ｆｐ６４に変換することができるようにそれを準備する。Ｕ／Ｓ抽出回路６２８は、固定小数点オペランドを、１の補数（又は２の補数）の形式から符号・絶対値形式（ｓｉｇｎ－ｍａｇｎｉｔｕｄｅ　ｆｏｒｍ）に変換し、オペランドが仮数部フィールドにおいて桁が合うように先頭又は最後に０を付加する。Ｕ／Ｓ抽出回路６２８は、自身の出力をオペランド選択マルチプレクサ６３６に提供し、さらに、Ｉ２Ｄ入力信号として指数経路５１８に提供する。

【００８３】

[0094]オペランド選択マルチプレクサ６３２，６３４，６３６は、制御論理回路６３０からの信号に応答して、オペランドＡ，Ｂ，Ｃを選択する。オペランド選択マルチプレクサ６３２は、絶対値／符号反転回路６１８からのオペランドＡと、（ｆｐ６４形式で表現されている）一定値０．０及び１．０との間での選択を行う。ＤＭＵＬ演算及びＤＦＭＡ演算の場合、オペランドＡが選択される。ＤＭＩＮ（ＤＭＡＸ）演算の場合、Ａ＜Ｂ（Ａ＞Ｂ）であればオペランドＡが選択され、そうでなければ１．０が選択される。ＤＡＤＤ演算及びＩ２Ｄ演算の場合、０．０が選択される。

【００８４】

[0095]オペランド選択マルチプレクサ６３４は、絶対値／符号反転回路６２０からのオペランドＢと、（ｆｐ６４形式で表現されている）一定値０．０及び１．０との間での選択を行う。ＤＭＵＬ演算及びＤＦＭＡ演算の場合、オペランドＢが選択される。ＤＭＩＮ（ＤＭＡＸ）演算の場合、Ｂ＜Ａ（Ｂ＞Ａ）であればオペランドＢが選択され、そうでなければ１．０が選択される。ＤＡＤＤ演算及びＩ２Ｄ演算の場合、０．０が選択される。

【００８５】

[0096]オペランド選択マルチプレクサ６３６は、絶対値／符号反転回路６２２からのオペランドＣと、ｆｐ３２抽出回路６２６からの抽出されたｆｐ３２値と、Ｕ／Ｓ抽出回路６２８からの抽出された符号なし又は符号付き整数値と、（ｆｐ６４形式で表現されている）一定値０．０との間での選択を行う。ＤＡＤＤ演算及びＤＦＭＡ演算の場合、オペランドＣが選択される。ＤＭＵＬ演算及び比較演算の場合、一定値０．０が選択される。Ｆ２Ｄ演算の場合、ｆｐ３２抽出回路６２６からの抽出されたｆｐ３２値が選択され、Ｉ２Ｄ演算の場合、Ｕ／Ｓ抽出回路６２８からの抽出されたｕ／ｓ値が選択される。

【００８６】

[0097]選択マルチプレクサ６３２，６３４，６３６によって選択されたオペランドＡ，Ｂ，及びＣは、特殊数検出回路６３８，６４０，６４２に提供される。ｆｐ６４のオペランドの場合、特殊数検出回路６３８，６４０，６４２は、全ての特殊数条件（非正規、ＮａＮ、無限大、０を含む）を検出する。Ｆ２Ｄ演算の場合、特殊数検出回路６４２は、ｆｐ３２抽出回路６２６から経路６４４を介してｆｐ３２特殊数情報を受け取る。特殊数検出回路６３８，６４０，６４２のそれぞれは、オペランドが特殊数であるか否かと、特殊数であるならばそのタイプとを示す特殊数信号（ＳＰＣ）を生成する。特殊数信号ＳＰＣは、図５に示したように信号経路５２４を通じて出力セクション５２２に提供される。特殊数検出回路には、一般的に従来の設計のものを使用することができる。代替の一実施形態においては、（回路６１２，６１４，及び６１６によって実行される）ＮａＮ検出は、回路６３８，６４０，６４２では繰り返されない。代わりに、特殊数検出回路６３８，６４０，６４２のそれぞれは、ＮａＮ検出回路６１２，６１４，及び６１６のうちの対応する回路からＮａＮ信号を受け取り、その信号を使用して、オペランドがＮａＮであるかを判定する。

【００８７】

[0098]特殊数検出回路６３８，６４０，及び６４２は、いずれかの特殊数が検出されるか否かにかかわらず、オペランドを、仮数ビットと、指数ビットと、符号ビットとに分ける。特殊数検出回路６３８は、オペランドＡの仮数の部分（Ａｍ）を仮数経路５１６（図５）に提供し、オペランドＡの指数の部分（Ａｅ）を指数経路５１８に提供し、符号ビット（Ａｓ）を符号経路５２０に提供する。特殊数検出回路６４０は、オペランドＢの仮数の部分（Ｂｍ）を仮数経路５１６に提供し、オペランドＢの指数の部分（Ｂｅ）を指数経路５１８に提供し、符号ビット（Ｂｓ）を符号経路５２０に提供する。特殊数検出回路６４２は、オペランドＣの仮数の部分（Ｃｍ）及び指数の部分（Ｃｅ）を指数経路５１８に提供し、符号ビット（Ｃｓ）を符号経路５２０に提供する。いくつかの実施形態においては、特殊数検出回路６３８，６４０，６４２は、仮数Ａｍ，Ｂｍ，Ｃｍに先頭の１を付加する（数が非正規であるときを除く）。

【００８８】

Ｂ．指数経路

[0099]図７は、本発明の実施形態による指数経路５１８のブロック図である。

【００８９】

[0100]指数計算回路７０２は、オペランド準備ブロック５１４（図５）から指数ビットＡｅ，Ｂｅ，及びＣｅを受け取り、ＤＦＭＡ結果のＡ＊Ｂ＋Ｃのブロック指数を計算する。従来の指数計算回路が使用されてもよい。一実施形態においては、全てのオペランドが正規数である場合、指数計算回路は、ＡｅとＢｅとを加算し、ｆｐ６４の指数のバイアス（１０２３）を減算して積Ａ＊Ｂの指数を求め、次いで、この積の指数と指数Ｃｅのうちの大きい方を、ＤＦＭＡ結果のブロック指数（ＢＬＥ）として選択する。このブロック指数ＢＬＥは、下流の最終指数計算回路７０４に提供される。一以上のオペランドが非正規数（特殊数信号ＳＰＣによって示される）である場合、適切な論理回路を使用してブロック指数ＢＬＥを求めることができる。別の実施形態においては、特殊数が含まれる演算における指数の決定は、後述するように出力セクション５２２において扱われる。

【００９０】

[0101]さらに、指数計算ブロック７０２は、Ｃｍと積Ａｍ＊Ｂｍの小数点位置が合うようにオペランドＣの仮数を実質的に左又は右にシフトさせる量を求める。この量は、制御信号Ｓｈ＿Ｃとしてシフト回路７０６に提供される。この制御信号は、Ｃｍを右シフトすることによって、実質的な左シフト又は右シフトを必ず達成することができるように、Ｃｍの余分なパディングを考慮することが好適である。

【００９１】

[0102]仮数Ｃｍは符号反転回路７０８に提供され、符号反転回路７０８は、条件付きで、すなわち、Ｃと積Ａ＊Ｂとの間に相対的マイナス符号（ｒｅｌａｔｉｖｅ　ｍｉｎｕｓ　ｓｉｇｎ）が存在している場合に、（例えば、１の補数による符号反転を使用して）Ｃｍの符号を反転する。相対的マイナス符号は、後述するように符号経路５２０において検出され、符号制御信号ＳｉｇｎＣＴＬは、相対的マイナス符号が存在しているかを示す。符号反転回路７０８の出力（Ｃｍ又は～Ｃｍのいずれか）は、シフト回路７０６に提供される。

【００９２】

[0103]一実施形態においては、シフト回路７０６は２１７ビットのバレルシフターであり、５４ビットの仮数Ｃｍを最大１５７ビットだけ右シフトすることができる。Ｃｍを右シフトする量は、Ｓｈ＿Ｃ信号によって決まる。仮数Ｃｍは、必要な距離だけ右シフトすることができるように位置合わせされた状態でシフターに入力されることが好適である。２１７ビットというサイズは、５３ビットの仮数Ｃｍ（及びガードビットとラウンドビット）の全体を、１０６ビットの積Ａ＊Ｂ（及び積のガードビットとラウンドビット）の左又は右に合わせる（１０６ビットの積Ａ＊Ｂは、２１７ビットフィールドのＭＳＢから右に５５ビットの位置に合わせられる）のに十分な空間が確保されるように選択されている。右シフトによってバレルシフターからはみ出したビットは破棄することができる。別の実施形態においては、フラグビットを使用して、右シフトによってバレルシフターからはみ出したビットの全てが「１」であるかを追跡し、この情報を、後述する丸め演算において使用することができる。

【００９３】

[0104]代替実施形態においては、従来のスワップマルチプレクサ（ｓｗａｐ　ｍｕｘ）を使用して、積Ａｍ＊ＢｍとＣｍとの間で大きい方のオペランドを選択することができ、次いで、小さい方のオペランドを右シフトすることができる。

【００９４】

[0105]Ｄ２Ｄ演算の場合、仮数ＣｍはＤ２Ｄ論理回路７１０にも提供される。Ｄ２Ｄ論理回路７１０は、仮数Ｃｍと、指数Ｃｅと、符号Ｃｓとを受け取り、整数丸め規則を適用する。一実施形態においては、Ｄ２Ｄ論理回路７１０は、２進小数点の位置を指数Ｃｅに基づいて決定し、次いで、ＯＰＣＴＬ信号（図示していない）に基づいて選択される丸め規則を適用する。丸めモードは、従来の論理回路を使用して実施することができる。また、任意の組合せの丸めモード（例えば、切捨てモード、シーリングモード、フロアモード

、最近接モード、ただしこれらに限定されない）をサポートしてもよい。

【００９５】

[0106]選択マルチプレクサ７１２は、シフトされた仮数Ｃ＿Ｓｈｉｆｔと、Ｄ２Ｄ論理回路の出力と、Ｕ／Ｓ抽出回路６２８（図６）からのＩ２Ｄ入力とを受け取り、これらの入力のうちの一つを、ＯＰＣＴＬ信号に基づいて、仮数経路５１６に供給される位置合わせ後の仮数Ｃ＿ａｌｉｇｎとして選択する。倍精度算術演算及び比較演算の場合、オペランドＣ＿Ｓｈｉｆｔが選択される。形式変換Ｄ２Ｄ又はＩ２Ｄの場合、Ｃ＿Ｓｈｉｆｔ以外の該当する入力が選択される。

【００９６】

[0107]アンダーフロー論理回路７１３は、ｆｐ６４及びｆｐ３２の結果における潜在的なアンダーフローを検出するように構成されている。Ｄ２Ｆ演算以外の演算の場合、アンダーフロー論理回路７１３は、１１ビットのｆｐ６４ブロック指数ＢＬＥが０であるか、又は、非正規結果が生じうるほど十分に０に近いかを判定する。アンダーフロー論理回路７１３は、指数が０に達しない範囲で仮数を左シフトすることのできる最大ビット数を、ブロック指数に基づいて求める。この数は、８ビットのアンダーフロー信号Ｕ＿ｆｐ６４として仮数経路５１６（図８を参照）に提供される。Ｄ２Ｆ演算の場合、指数は８ビットのｆｐ３２の指数として扱われ、アンダーフロー論理回路７１３は、許容される最大左シフトを求める。この数は、８ビットのアンダーフロー信号Ｕ＿ｆｐ３２として仮数経路５１６に提供される。

【００９７】

[0108]指数経路５１８は、最終指数計算論理回路７０４を更に含んでいる。減算回路７２０には、ブロック指数ＢＬＥが提供される。さらに、減算回路７２０には、仮数経路５１６からのブロックシフト（ＢＬ＿Ｓｈ）信号が提供される。このＢＬ＿Ｓｈ信号は、後述するように、積Ａｍ＊ＢｍをオペランドＣ＿ａｌｉｇｎに加算したときのＭＳＢの相殺効果を反映している。減算回路７２０は、ＢＬＥからＢＬ＿Ｓｈを減算して差ＥＤＩＦを求める。アンダーフロー／オーバーフロー回路７２２は、減算結果ＥＤＩＦにおけるアンダーフロー又はオーバーフローを検出する。プラス１回路７２４は、結果のＥＤＩＦに１を加算し、マルチプレクサ７２０は、アンダーフロー／オーバーフロー条件に基づいて、ＥＤＩＦ信号とＥＤＩＦ＋１信号のいずれかを結果の指数Ｒｅとして選択する。結果Ｒｅとアンダーフロー／オーバーフロー信号（Ｕ／Ｏ）とが、出力セクション５２２に提供される。

【００９８】

Ｃ．仮数経路

[0109]図８は、本発明の実施形態による仮数経路５１６のブロック図である。仮数経路５１６は、オペランドＡ，Ｂ，及びＣの仮数に対して積演算及び和演算を実行する。

【００９９】

[0110]５３×５３乗算器８０２は、（上述した）オペランド準備ブロック５１４から仮数Ａｍ及びＢｍを受け取り、１０６ビットの積Ａｍ＊Ｂｍを計算する。この積は１６８ビット加算器８０４に提供され、加算器８０４は、位置合わせ後の仮数Ｃ＿ａｌｉｇｎを更に受け取る。バレルシフター７０６によって使用される２１７ビットフィールドの末尾部分のビットは破棄することができ、或いは、末尾部分のビットが０でないか、又は全て１であるかを示すフラグビットを維持することができる。加算器８０４は、出力Ｓｕｍ及び～Ｓｕｍ（和の２の補数）を生成する。マルチプレクサ８０６は、Ｓｕｍ及び～Ｓｕｍのいずれかを、和のＭＳＢ（符号ビット）に基づいて選択する。選択された和（Ｓ）は、ゼロ検出回路８１４と左シフト回路８１６とに提供される。ゼロ検出回路８１４は、選択された和Ｓが０であるか否かを判定し、対応するＲ＿ＺＥＲＯ信号を符号経路５２０に提供する。

【０１００】

[0111]さらに、仮数経路５１６は、和Ｓを正規化する。ＬＺＤ回路８０８，８１０を使用して、Ｓｕｍ及び～Ｓｕｍの双方について先頭のゼロの検出（ＬＺＤ）が並列に実行さ

れる。ＬＺＤ回路８０８，８１０のそれぞれは、自身の入力における先頭のゼロの数を示すＬＺＤ信号（Ｚ１，Ｚ２）を生成する。ＬＺＤマルチプレクサ８１２は、和のＭＳＢ（符号ビット）に基づいて、該当するＬＺＤ信号（Ｚ１又はＺ２）を選択する。マルチプレクサ８０６によってＳｕｍが選択されるならばＺ２を選択し、マルチプレクサ８０６によって～Ｓｕｍが選択されるならばＺ１を選択する。選択されたＬＺＤ信号は、ブロックシフト信号ＢＬ＿Ｓｈとして指数経路５１８に提供され、指数経路５１８において、当該ＬＺＤ信号が使用されて、上述したように結果の指数が調整される。

【０１０１】

[0112]正規化論理回路８１８は、和Ｓの正規化シフトを決める左シフト量Ｌｓｈｉｆｔを選択する。正規数の結果の場合、左シフト量は、５２ビットの仮数部（及びガードビットとラウンドビット）を残して先頭の１が仮数フィールドの外にシフトされるだけ十分に大きいことが好適である。しかしながら、場合によっては、結果は、ｆｐ６４非正規数又はｆｐ３２非正規数として表現すべきアンダーフローである。一実施形態においては、Ｄ２Ｆ以外の演算の場合、正規化論理回路８１８は、ＢＬ＿Ｓｈがアンダーフロー信号Ｕ＿ｆｐ６４よりも大きくない限りは、ＬＺＤマルチプレクサ８１２からの出力ＢＬ＿Ｓｈを選択し、ＢＬ＿Ｓｈがアンダーフロー信号Ｕ＿ｆｐ６４よりも大きい場合、正規化論理回路８１８は、左シフト量としてＵ＿ｆｐ６４を選択する。Ｄ２Ｆ演算の場合、正規化論理回路８１８は、ｆｐ３２アンダーフロー信号Ｕ＿ｆｐ３２を使用して左シフト量Ｌｓｈｉｆｔを制限する。

【０１０２】

[0113]左シフト回路８１６は、和Ｓを量Ｌｓｈｉｆｔだけ左シフトする。結果のＳｎは、丸め論理回路８２０と、プラス１加算器８２２と、仮数選択マルチプレクサ８２４とに提供される。丸め論理回路８２０は、ＩＥＥＥ規格の算術演算に対して定義されている四つの丸めモード（最近接、フロア、シーリング、切捨て）を実施することが好適であり、異なるモードにおいては異なる結果が選択され得る。ＯＰＣＴＬ信号又は別の制御信号（図示していない）を使用して、丸めモードの一つを指定することができる。丸め論理回路８２０は、丸めモードと、正規化された和Ｓｎとに基づいて、結果のＳｎを選択するのか、又はプラス１加算器８２２によって計算されるＳｎ＋１を選択するのかを決定する。選択マルチプレクサ８２４は、適切な結果（Ｓｎ又はＳｎ＋１）を選択することによって、丸め論理回路８２０からの制御信号に応答する。

【０１０３】

[0114]マルチプレクサ８２４によって選択された結果は、フォーマッティングブロック８２６に渡される。浮動小数点出力を有する演算の場合、ブロック８２６は、仮数Ｒｍを出力セクション５２２に提供する。和Ｓは、（整数演算をサポートするため）少なくとも６４ビット幅であることが好適であり、余分なビットはフォーマッティングブロック８２６によって削除することもできる。Ｄ２Ｉ演算（整数出力を有する）の場合、フォーマッティングブロック８２６は、結果を、ＬＳＢを含む５２ビットのｉｎｔ＿Ｌフィールドと、ＭＳＢを含む１１ビットのｉｎｔ＿Ｍフィールドとに分ける。Ｒｍ、ｉｎｔ＿Ｌ、及びｉｎｔ＿Ｍは、出力セクション５２２に提供される。

【０１０４】

Ｄ．符号経路

[0115]図９は、本発明の実施形態による符号経路５２０のブロック図である。符号経路５２０は、オペランド準備ブロック５１４（図５）からオペランドの符号Ａｓ，Ｂｓ，及びＣｓを受け取る。さらに、符号経路５２０は、仮数経路５１６からの結果ゼロ信号Ｒ＿Ｚｅｒｏと、進行中の演算のタイプを示すＯＰＣＴＬ信号と、オペランド準備ブロック５１４からの特殊数信号ＳＰＣとを受け取る。符号経路５２０は、これらの情報に基づいて、結果の符号を求めて符号ビットＲｓを生成する。

【０１０５】

[0116]より具体的には、符号経路５２０は、積／和回路９０２と最終符号回路９０４とを含んでいる。積／和回路９０２は、オペランド準備ブロック５１４から、オペランドＡ

，Ｂ，及びＣの符号ビットＡｓ，Ｂｓ，及びＣｓを受け取る。積／和回路９０２は、符号ビットＡｓ及びＢｓと従来の符号論理規則（ｓｉｇｎ　ｌｏｇｉｃ　ｒｕｌｅｓ）とを使用して、積Ａ＊Ｂの符号（Ｓｐ）を求め、次いで、この積の符号と符号ビットＣｓとを比較し、積とオペランドＣとが同符号を有するか又は異符号を有するかを判定する。積／和回路９０４は、この情報に基づいて、ＳｉｇｎＣＴＬ信号をアサート又はデアサートし、ＳｉｇｎＣＴＬ信号が、最終符号回路９０４と、指数経路５１８における符号反転ブロック７０８（図７）とに提供される。さらには、積とオペランドＣとが同符号を有する場合、最終結果もその符号を有し、積とオペランドＣとが異符号を有する場合、結果は、どちらが大きいかによって決まる。

【０１０６】

[0117]最終符号回路９０４は、最終的な符号を決定する上で必要な情報の全てを受け取る。具体的には、最終符号回路９０４は、符号情報（積の符号Ｓｐを含む）と、積／和回路９０２からのＳｉｇｎＣＴＬ信号と、符号ビットＡｓ，Ｂｓ，及びＣｓとを受け取る。さらに、最終符号回路９０４は、仮数経路５１６からのゼロ検出信号Ｒ＿ＺＥＲＯと、オペランド準備ブロック５１４からの特殊数信号ＳＰＣとを受け取る。さらに、最終符号回路９０４は、仮数経路５１６における加算器８０４から和のＭＳＢ（和が正であるか負であるかを示す）を受け取る。

【０１０７】

[0118]最終符号回路９０４は、これらの情報に基づいて、従来の符号論理回路を使用して結果の符号ビットＲｓを決定することができる。例えば、ＤＦＭＡ演算の場合、符号ビットＳｐとＣｓとが同じであるならば、結果もその符号を有する。ＳｐとＣｓとが異符号であるならば、仮数経路５１６における加算器８０４が、（Ａｍ＊Ｂｍ）－Ｃ＿ａｌｉｇｎを計算する。Ａｍ＊ＢｍがＣ＿ａｌｉｇｎより大きい場合、加算器８０４は正の結果Ｓｕｍを計算し、積の符号Ｓｐを選択すべきである。Ａｍ＊ＢｍがＣ＿ａｌｉｇｎより小さい場合、加算器８０４は負の結果Ｓｕｍを計算し、符号Ｃｓを選択すべきである。加算器８０４の出力ＳｕｍのＭＳＢは、結果の符号を示しており、この選択を駆動する目的に使用することができる。結果Ｓｕｍが０である場合、Ｒ＿ＺＥＲＯ信号がアサートされ、最終符号回路９０４は、いずれか適切な符号を選択することができる（ｆｐ６４形式では０は正又は負のどちらでもよい）。ＤＦＭＡ以外の演算の場合、最終符号回路９０４は、最終的な符号としていずれかのオペランドの符号を通過させることができる。

【０１０８】

Ｅ．出力セクション

[0119]図１０は、本発明の実施形態によるＤＦＭＡユニット３２０の出力セクション５２２のブロック図である。

【０１０９】

[0120]出力マルチプレクサ制御論理回路１００２は、指数経路５１８（図７）からのアンダーフロー／オーバーフロー（Ｕ／Ｏ）信号と、オペランド準備ブロック５１４（図６）からのＲ＿ｔｅｓｔ信号及びＳＰＣ信号と、進行中の演算のタイプを示すＯＰＣＴＬ信号とを受け取る。出力マルチプレクサ制御論理回路１００２は、これらの情報に基づいて、仮数部選択マルチプレクサ１００４及び指数選択マルチプレクサ１００６のための選択制御信号を生成する。さらに、出力マルチプレクサ制御論理回路１００２は、条件コード信号ＣＯＮＤ（例えば、オーバーフロー又はアンダーフロー条件、ＮａＮ条件、又はその他の条件を示すことができる）を生成する。いくつかの実施形態においては、条件コードは、ＤＳＥＴ演算時にブール結果を伝える目的にも使用される。

【０１１０】

[0121]仮数部選択マルチプレクサ１００４は、仮数経路５１６からの結果の仮数部Ｒｍと、最大５２ビットの整数出力（Ｄ２Ｉ演算時に使用される）と、複数の特殊値とを受け取る。特殊値としては、一実施形態においては、１の５２ビットフィールド（Ｄ２Ｉ演算において６４ビットの最大整数を表現するために使用される）、０の５２ビットフィールド（０．０又は１．０が結果である場合に使用される）、５２ビットフィールド０ｘ０＿

００００＿８０００＿００００（Ｄ２Ｉ演算において３２ビットの最小整数を表現するために使用される）、先頭が１である５２ビットフィールド（内部で生成されるクワイエット型ＮａＮを表すために使用される）、ｍａｘ＿ｉｎｔ３２値（例：０ｘ７ｆｆｆ＿ｆｆｆｆ＿ｆｆｆｆ＿ｆｆｆｆ）（Ｄ２Ｉ演算において３２ビットの最大整数を表現するために使用される）、クワイエット型ＮａＮ値（オペランド準備ブロック５１４からの、ＮａＮである入力オペランドを通過させるために使用される）、ｍｉｎ＿ｄｅｎｏｒｍ値（例：最後のビット位置が１）（アンダーフローの場合に使用される）が挙げられる。演算に応じて、及びオペランドのいずれか又は結果が特殊数であるかに応じて、入力のいずれかを選択することができる。

【０１１１】

[0122]指数選択マルチプレクサ１００６は、指数経路５１８からの結果の指数Ｒｅと、最大１１個の整数ビット（整数形式の出力の場合のＭＳＢ）と、複数の特殊値とを受け取る。特殊値としては、一実施形態においては、０ｘ３ｆｆ（ｆｐ６４における１．０の指数）、０ｘ０００（非正規及び０．０の場合の指数）、０ｘ７ｆｅ（正規数のｆｐ６４最大指数）、０ｘ７ｆｆ（ｆｐ６４　ＮａＮ又はｆｐ６４無限大の結果の場合）が挙げられる。演算に応じて、及びオペランドのいずれか又は結果が特殊数であるかに応じて、入力のいずれかを選択することができる。

【０１１２】

[0123]連結ブロック１００８は、符号ビットＲｓと、マルチプレクサ１００４によって選択される仮数部ビットと、マルチプレクサ１００６によって選択される指数ビットとを受け取る。連結ブロック１００８は、結果を、（例えば、ＩＥＥＥ７５４規格に従って符号、指数、仮数部の順序に）フォーマッティングし、６４ビットの出力信号ＯＵＴを提供する。

【０１１３】

Ｆ．オペランドのバイパス経路又は通過経路

[0124]ＤＦＭＡユニット３２０は、いくつかの実施形態においては、オペランドを修正せずに様々な回路ブロックを伝搬させることのできるバイパス経路又は通過経路を提供する。例えば、いくつかの演算時、乗算器８０２は、入力（例：Ａｍ）に１．０を乗算して、入力Ａｍを実質的に通過させる。Ａｍに１．０を乗算するのではなく、乗算器８０２の周囲に入力Ａｍのバイパス経路を提供することができる。このバイパス経路は、Ａｍが加算器８０４への入力に正しいタイミングで到着するように、乗算器８０２と同じ数のクロックサイクルを消費することが有利である。しかしながら、乗算器８０２がバイパスされるとき、乗算器８０２を電力遮断状態又は低電力状態に設定することができ、これによって、回路面積が少し増大することと引き換えに電力消費量が低減する。同様に、いくつかの演算時、加算器８０４を使用して入力（例：Ｃ＿ａｌｉｇｎ）に0を加算し、入力Ｃ＿ａｌｉｇｎを実質的に通過させる。特に、加算器８０４の出力Ｓｕｍ及び～Ｓｕｍのどちらをマルチプレクサ８０６によって選択すべきかが事前に既知である演算の場合、Ｃ＿ａｌｉｇｎに0を加算するのではなく、加算器８０４の周囲に入力Ｃ＿ａｌｉｇｎのためのバイパス経路を提供することができる。入力Ｃ＿ａｌｉｇｎを、Ｓｕｍの経路及び～Ｓｕｍの経路のうちの正しい経路にバイパスさせることができる。この場合も、バイパス経路は、タイミングが影響を受けないように、加算器８０４と同じ数のクロックサイクルを消費することが有利である。しかしながら、加算器８０４をバイパスする演算においては、加算器８０４を電力遮断状態又は低電力状態に設定することができるので、電力消費量を低減することができる。

【０１１４】

[0125]従って、（後の）第ＩＩＩ節における演算の説明では、様々なオペランドを特定の回路ブロックにバイパス又は通過させることを言及している。このことは、オペランドに影響を与えない演算（例：０を加算する、１．０を乗算する）を実行するよう介在する回路ブロックを制御して、当該回路ブロックへの入力がそのまま出力として通過させることによって、又は、バイパス経路を使用することによって、達成することができることを

理解されたい。さらには、ある回路ブロックの周囲のバイパス経路又は通過経路をたどるとき、以降の回路ブロックにおいては、引き続きそのバイパス経路を必ずしもたどらなくてよい。さらには、一つの回路ブロックにおいて修正された値は、以降の回路ブロックの周囲のバイパス経路をたどることができる。演算時に特定の回路ブロックがバイパスされる場合、その回路ブロックを電力遮断状態に設定して電力消費量を低減する、又は、通常に動作させて、例えば選択マルチプレクサ又はその他の回路を使用することによってその出力を無視させることができる。

【０１１５】

[0126]本明細書に説明したＤＦＭＡユニットは例示を目的としており、変形及び変更が可能であることが理解されるであろう。本明細書に説明した回路ブロックの多くは、従来の機能を提供し、この技術分野において公知である技術を使用して実施することができる。従って、これらのブロックの詳しい説明は省略した。演算回路をブロックに分割する方式は変更することができ、ブロックを組み合わせる、或いはブロックを変更することができる。さらに、パイプラインステージの数と、特定の回路ブロック又は演算を特定のパイプラインステージに割り当てる方式も、変更する、又は別の形態をとることができる。特定の実施形態における回路ブロックの選択及び編成は、サポートする一連の演算に依存し、演算の可能な組合せのそれぞれにおいて、本明細書に説明したブロックの全てが必ずしも要求されるわけではないことが、当業者には認識されるであろう。

【０１１６】

ＩＩＩ．ＤＦＭＡユニットの演算

[0127]ＤＦＭＡユニット３２０は、図４に一覧した演算の全てを、上述した回路ブロックを利用して、回路面積の面で効率的にサポートすることが好適である。従って、ＤＦＭＡユニット３２０の動作は、少なくともいくつかの側面において、実行される演算に依存する。以下の節では、ＤＦＭＡユニット３２０を使用して、図４に一覧した演算のそれぞれを実行する方法について説明する。

【０１１７】

[0128]なお、浮動小数点例外（例えば、オーバーフロー条件、アンダーフロー条件を含む）は、余分な処理サイクルを必要とすることなくＤＦＭＡユニット３２０の中で扱われることに留意されたい。例えば、入力オペランドがＮａＮ又はその他の特殊数である演算は、図５のオペランド準備ブロック５１４において検出され、出力セクション５２２において適切な特殊数出力が選択される。ＮａＮ、アンダーフロー、オーバーフロー、又はその他の特殊数が演算の過程において生じた場合、その条件が検出され、出力セクション５２２において適切な特殊数出力が選択される。

【０１１８】

Ａ．融合型積和演算（ＤＦＭＡ）

[0129]ＤＦＭＡ演算の場合、ＤＦＭＡユニット３２０は、ｆｐ６４形式のオペランドＡ０，Ｂ０，及びＣ０と、ＤＦＭＡ演算を実行することを示すオペコードと、を受け取る。ＮａＮ回路６１２，６１４，６１６は、選択されるオペランドのいずれか一つ以上がＮａＮであるかを判定する。絶対値／符号反転回路６１８，６２０，６２２は、オペランドのそれぞれについて、必要な場合に符号ビットを反転する（又は反転しない）。オペランド選択マルチプレクサ６３２，６３４，及び６３６は、それぞれの絶対値／符号反転回路６１８，６２０，及び６２２の出力を選択し、これらの出力を特殊数検出回路６３８，６４０，６４２に提供する。特殊数検出回路６３８，６４０，及び６４２は、オペランドのそれぞれが特殊数であるかを判定し、該当する特殊数ＳＰＣ信号を経路５２４上に生成する。特殊数検出回路６３８，６４０，及び６４２は、仮数Ａｍ，Ｂｍ，及びＣｍ（正規数の場合は先頭に１が付加されており、非正規数の場合は先頭に０が付加されている）を仮数経路５１６に提供し、指数Ａｅ，Ｂｅ，及びＣｅを指数経路５１８に提供し、符号ビットＡｓ，Ｂｓ，及びＣｓを符号経路５２０に提供する。

【０１１９】

[0130]Ａ／Ｂ比較回路６２４、ｆｐ３２抽出回路６２６、及び、Ｕ／Ｓ整数抽出回路６

２８はＤＦＭＡ演算においては使用されず、これらの回路は、必要に応じて、電力遮断状態又は低電力状態に設定され得る。

【０１２０】

[0131]符号経路５２０においては、積／和回路９０２が、積Ａ＊Ｂが正であるか負であるかを符号ビットＡｓ及びＢｓから判定し、積の符号Ｓｐと符号ビットＣｓとを比較する。積とＣｓとが異符号を有する場合、異符号を示すためにＳｉｇｎＣＴＬ信号がアサートされ、積とＣｓとが同符号を有する場合、ＳｉｇｎＣＴＬ信号がデアサートされる。

【０１２１】

[0132]指数経路５１８（図７）においては、指数計算ブロック７０２が、指数Ａｅ，Ｂｅ，及びＣｅを受け取る。指数計算ブロック７０２は、指数Ａｅと指数Ｂｅとを加算し、積Ａ＊Ｂのブロック指数を求め、次いで、この積のブロック指数と指数Ｃｅのうちの大きい方を、結果のブロック指数ＢＬＥとして選択する。さらに、指数計算ブロック７０２は、積のブロック指数と指数Ｃｅのうちの小さい方を、二つのうちの大きい方から減算し、対応するシフト制御信号Ｓｈ＿Ｃを生成する。アンダーフロー論理回路７１３は、ブロック指数ＢＬＥがアンダーフロー又は潜在的なアンダーフローに対応しているかを検出し、アンダーフロー信号Ｕ＿ｆｐ６４を生成する（ＤＦＭＡ演算時にはＵ＿ｆｐ３２信号は使用されない。）。

【０１２２】

[0133]符号反転ブロック７０８は、オペランド準備ブロック５１４からの仮数Ｃｍと、符号経路５２０からのＳｉｇｎＣＴＬ信号とを受け取る。ＳｉｇｎＣＴＬ信号がアサートされている場合、符号反転ブロック７０８は、相対的マイナス符号を考慮するために仮数Ｃｍを反転し、反転されたＣｍをシフト回路７０６に提供する。アサートされていない場合、符号反転ブロック７０８は、Ｃｍを修正せずにシフト回路７０６に提供する。

【０１２３】

[0134]シフト回路７０６は、符号反転ブロック７０８によって提供された仮数Ｃｍを、シフト制御信号Ｓｈ＿Ｃに対応する量だけ右シフトし、シフトされた仮数Ｃ＿Ｓｈｉｆｔを選択マルチプレクサ７１２に提供する。選択マルチプレクサ７１２は、シフトされた仮数Ｃ＿Ｓｈｉｆｔを選択し、このシフトされた仮数をオペランドＣ＿ａｌｉｇｎとして仮数経路５１６に提供する。

【０１２４】

[0135]仮数経路５１６（図８）においては、乗算器８０２が、１０６ビットの積Ａｍ＊Ｂｍを計算し、この積を１６８ビットの加算器８０４に提供する。乗算器８０２の動作は、指数計算ブロック７０２の動作と並列に行うことができる。

【０１２５】

[0136]加算器８０４は、指数経路５１８の選択マルチプレクサ７１２からオペランドＣ＿ａｌｉｇｎを受け取り、入力Ａｍ＊ＢｍとＣ＿ａｌｉｇｎとを加算してＳｕｍ及び～Ｓｕｍを求める。ＳｕｍのＭＳＢに基づいて、マルチプレクサ８０６は、これら出力の一方を最終的な和として選択する。Ｓｕｍが正である（ＭＳＢが０）ならばＳｕｍが選択され、Ｓｕｍが負である（ＭＳＢが１）ならば～Ｓｕｍが選択される。ＬＺＤ回路８０８及び８１０は、それぞれ、～Ｓｕｍ及びＳｕｍにおける先頭の０の数を求める。マルチプレクサ８１２は、ＬＺＤ出力の一方を先頭の０の数として選択し、先頭ゼロ信号ＢＬ＿Ｓｈを指数経路５１８と正規化論理回路８１８とに提供する。

【０１２６】

[0137]マルチプレクサ８０６によって選択される最終的な和Ｓは、ゼロ検出回路８１４にも提供される。ゼロ検出回路８１４は、最終的な和が０である場合、符号経路５２０へのＲ＿ＺＥＲＯ信号をアサートし、そうでない場合、Ｒ＿ＺＥＲＯ信号をアサートしない。

【０１２７】

[0138]正規化論理回路８１８は、Ｕ＿ｆｐ６４信号がアンダーフローを示していない限りは、先頭ゼロ信号を正規化信号Ｌｓｈｉｆｔとして選択し、Ｕ＿ｆｐ６４信号がアンダ

ーフローを示している場合、仮数は指数１に対応する位置までシフトされるのみであり、従って、結果は非正規形式において表現される。シフト回路８１６は、選択された和ＳをＬｓｈｉｆｔ信号に応答して左シフトし、正規化された和Ｓｎを生成する。プラス１加算器８２２は、正規化された和Ｓｎに１を加算する。丸め論理回路８２０は、（ＯＰＣＴＬ信号によって指定される）丸めモードと、（経路８２１上の）正規化された和ＳｎのＬＳＢとを使用して、正規化された和を切り上げるべきであるかを判定する。切り上げるべきである場合、丸め論理回路８２０は、加算器８２２からの出力Ｓｎ＋１が選択されるように、選択マルチプレクサ８２４を制御する。そうでない場合、選択マルチプレクサ８２４は、正規化された和Ｓｎを選択する。選択マルチプレクサ８２４は、選択された結果Ｒｍを出力セクション５２２に提供する。いくつかの実施形態においては、選択マルチプレクサ８２４は、結果の仮数から先頭ビット（正規数の場合は１）をドロップする。

【０１２８】

[0139]丸め演算と並列に、指数経路５１８（図７）は、結果の指数Ｒｅを計算する。具体的には、減算ブロック７２０が、指数計算ブロック７０２からのブロック指数ＢＬＥと、仮数経路５１６からのブロックシフト信号ＢＬ＿Ｓｈとを受け取る。減算ブロック７２０は、これら二つの入力を減算し、その結果ＥＤＩＦを、アンダーフロー／オーバーフロー論理回路７２２と、プラス１加算器７２４と、選択マルチプレクサ７２６とに提供する。アンダーフロー／オーバーフロー論理回路７２２は、結果のＭＳＢを使用して、アンダーフロー又はオーバーフローが発生したかを判定し、アンダーフロー又はオーバーフローの存在の有無を反映するＵ／Ｏ信号を生成する。選択マルチプレクサ７２６は、このＵ／Ｏ信号に基づいて、減算結果ＥＤＩＦと、プラス１加算器７２４の出力との間での選択を行う。選択された値は、結果の指数Ｒｅとして、Ｕ／Ｏ信号と共に出力セクション５２２に提供される。

【０１２９】

[0140]丸め演算と並列に、符号経路５２０（図９）における最終符号回路９０４は、積／和回路９０２によって判定された符号と、仮数経路５１６から受け取るＲ＿ＺＥＲＯ信号及び和のＭＳＢと、オペランド準備ブロック５１４から受け取る特殊数ＳＰＣ信号とに基づいて、最終的な符号Ｒｓを決定する。

【０１３０】

[0141]出力セクション５２２（図１０）は、仮数経路５１６からの結果の仮数Ｒｍと、指数経路５１８からの結果の指数Ｒｅと、符号経路５２０からの結果の符号Ｒｓと、オペランド準備ブロック５１４からの特殊数ＳＰＣ信号と、指数経路５１８からのＵ／Ｏ信号とを受け取る。ＳＰＣ信号及びＵ／Ｏ信号に基づいて、出力マルチプレクサ制御論理回路１００２は、仮数部マルチプレクサ１００４のための制御信号と、指数マルチプレクサ１００６のための制御信号とを生成する。さらに、出力マルチプレクサ制御論理回路１００２は、様々な条件コードＣＯＮＤ（例えば、結果がオーバーフロー、アンダーフロー、又はＮａＮであるかを示す）を生成する。

【０１３１】

[0142]仮数部マルチプレクサ１００４は、正規数及び非正規数の場合、仮数部Ｒｍを選択する。アンダーフローの場合、丸めモードに応じて、０又は仮数部ｍｉｎ＿ｄｅｎｏｒｍが選択される。オーバーフロー（無限大）の場合、仮数部０ｘ０＿０００＿０００＿０００が選択される。いずれかの入力オペランドがＮａＮである場合、クワイエット型ＮａＮの仮数部が選択される。演算中にＮａＮが発生した場合、内部（クワイエット型）ＮａＮの仮数０ｘ８＿０００＿０００が選択される。

【０１３２】

[0143]指数マルチプレクサ１００６は、正規数の場合、結果の指数Ｒｅを選択する。非正規数及びアンダーフローの場合、指数０ｘ０００が選択される。無限大又はＮａＮの場合、最大指数０ｘ７ｆｆが選択される。

【０１３３】

[0144]連結ブロック１００８は、選択された仮数部と、選択された指数と、符号Ｒｓと

を受け取り、ｆｐ６４の最終的な結果ＯＵＴを生成する。条件コードは必要に応じて設定され得る。

【０１３４】

[0145]なお、ＤＦＭＡユニット３２０では、全てのＤＦＭＡ演算が、オーバーフロー又はアンダーフローにかかわらず同数のサイクルで完了することに留意されたい。さらに、ＤＦＭＡユニット３２０は、ＩＥＥＥ７５４規格に従って、浮動小数点算術演算におけるオーバーフロー／アンダーフロー時の予測デフォルト動作（ｅｘｐｅｃｔｅｄ　ｄｅｆａｕｌｔ　ｏｖｅｒｆｌｏｗ／ｕｎｄｅｒｆｌｏｗ　ｂｅｈａｖｉｏｒ）を実施する。即ち、適切な結果ＯＵＴが返され、オーバーフロー／アンダーフロー条件を示すように、（条件コードＣＯＮＤにおける）ステータスフラグがセットされる。いくつかの実施形態においては、これらの条件を扱うためのユーザ定義のトラップが実施されてもよく、トラップを発生させるべきであるかを判定するよう、条件コードＣＯＮＤを使用することが可能である。

【０１３５】

Ｂ．乗算

[0146]乗算（ＤＭＵＬ）は、オペランドＣを０にセットして、ＤＦＭＡユニット３２０が、Ａ＊Ｂ＋０．０を計算することで、上述したＤＦＭＡ演算と同じように実施することが可能である。一実施形態においては、オペコードがＤＭＵＬ演算を示している場合に、選択マルチプレクサ６３６（図６）を使用して、入力オペランドＣをｆｐ６４の値０に置き換えることができる。

【０１３６】

Ｃ．加算

[0147]加算（ＤＡＤＤ）は、オペランドＢを１．０にセットして、ＤＦＭＡユニット３２０がＡ＊１．０＋Ｃを計算することで、上述したＤＦＭＡ演算と同じように実施することが可能である。一実施形態においては、オペコードがＤＡＤＤ演算を示している場合に、選択マルチプレクサ６３４（図６）を使用して、入力オペランドＢをｆｐ６４の値１．０に置き換えることができる。

【０１３７】

Ｄ．ＤＭＡＸ及びＤＭＩＮ

[0148]ＤＭＡＸ演算又はＤＭＩＮ演算の場合、オペランド準備ブロック５１４（図６）が、オペランドＡ及びオペランドＢを受け取る。ＮａＮ回路６１２及び６１４が、選択されたオペランドのいずれか一方又は双方がＮａＮであるかを判定する。絶対値／符号反転回路６１８，６２０が、必要に応じて符号ビットを反転する（又は反転しない）。

【０１３８】

[0149]Ａ／Ｂ比較回路６２４は、絶対値／符号反転回路６１８，６２０からオペランドＡ及びオペランドＢを受け取り、例えば、これらのオペランドがあたかも整数であるかのようにＡからＢを減算することによって、比較を実行する。この減算に基づいて、Ａ／Ｂ比較回路６２４は、ＡがＢよりも大きい、ＡがＢよりも小さい、又はＡがＢに等しいかを示すＣＯＭＰ信号を生成する。ＣＯＭＰ信号は制御論理回路６３０に提供される。制御論理回路６３０は、対応するＲ＿Ｔｅｓｔ信号を生成し、さらに、選択マルチプレクサ６３２，６３４，及び６３６用の選択信号を生成する。

【０１３９】

[0150]具体的には、ＤＭＡＸ演算の場合、オペランドＡのマルチプレクサ６３２は、ＡがＢより大きければオペランドＡを選択し、ＡがＢより小さければオペランド１．０を選択し、一方で、オペランドＢのマルチプレクサ６３４は、ＢがＡより大きければオペランドＢを選択し、ＢがＡより小さければオペランド１．０を選択する。ＤＭＩＮ演算の場合、オペランドＡのマルチプレクサ６３２は、ＡがＢより小さければオペランドＡを選択し、ＡがＢより大きければオペランド１．０を選択し、一方で、オペランドＢのマルチプレクサ６３４は、ＢがＡより小さければオペランドＢを選択し、ＢがＡより大きければオペランド１．０を選択する。ＤＭＡＸ及びＤＭＩＮのいずれの場合も、Ａ＝Ｂである特殊な

場合は、マルチプレクサ６３２がオペランドＡを選択する一方でマルチプレクサ６３４がオペランド１．０を選択するように制御することによって、或いは、マルチプレクサ６３２がオペランド１．０を選択する一方でマルチプレクサ６３４がオペランドＢを選択するように制御することによって、扱うことができる。いずれの場合にも、オペランドＣのマルチプレクサ６３６は、オペランド０．０を選択するように動作することが好適である。

【０１４０】

[0151]特殊数検出回路６３８，６４０，及び６４２は、オペランドが特殊数であるかを判定し、該当する特殊数ＳＰＣ信号を経路５２４上に生成する。特殊数検出回路６３８，６４０，及び６４２は、仮数Ａｍ，Ｂｍ，及びＣｍ（正規数の場合は先頭に１が付加されており、非正規数の場合は先頭に０が付加されている）を仮数経路５１６に提供し、指数Ａｅ，Ｂｅ，及びＣｅを指数経路５１８に提供し、符号ビットＡｓ，Ｂｓ，及びＣｓを符号経路５２０に提供する。

【０１４１】

[0152]ｆｐ３２抽出回路６２６及び符号なし／符号付き整数抽出回路６２８は、ＤＭＡＸ演算又はＤＭＩＮ演算の場合には使用されず、これらの回路は、必要に応じて電力遮断状態又は低電力状態に設定され得る。

【０１４２】

[0153]仮数経路５１６、指数経路５１８、及び符号経路５２０は、ＤＦＭＡ演算について上述したように動作する。ＤＭＡＸ演算の場合、仮数経路５１６、指数経路５１８、及び符号経路５２０は、ｍａｘ（Ａ，Ｂ）＊１．０＋０．０を計算する。ＤＭＩＮ演算の場合、仮数経路５１６、指数経路５１８、及び符号経路５２０は、ｍｉｎ（Ａ，Ｂ）＊１．０＋０．０を計算する。従って、正規数の場合、Ｒｍ，Ｒｅ，及びＲｓは、所望の結果の仮数、指数、及び符号に対応する。

【０１４３】

[0154]出力セクション５２２（図１０）は、特殊数を扱う。具体的には、ＤＭＡＸ演算及びＤＭＩＮ演算の結果は、ＮａＮオペランドに対して定義されておらず、結果はＮａＮ値にセットされ得る。出力マルチプレクサ制御論理回路１００２は、結果をＮａＮとするべきかを、特殊数ＳＰＣ信号を使用して判定する。ＮａＮとするべきである場合、仮数部マルチプレクサ１００４がクワイエット型ＮａＮ入力を選択し、指数マルチプレクサが０ｘ７ｆｆを選択する。そうでない場合、結果Ｒｍ及び結果Ｒｅが選択される。条件コードは必要に応じて設定され得る。

【０１４４】

[0155]代替実施形態においては、仮数経路５１６、指数経路５１８、及び符号経路５２０のコンポーネントのいくつか又は全てをバイパスすることができる。バイパスされるコンポーネントは低電力状態に設定され得る。バイパス経路が、仮数経路５１６、指数経路５１８、及び符号経路５２０のうちの最も長い経路と同数のパイプラインステージを占めるように、バイパス経路に様々な遅延回路（ラッチなど）を含めてもよい。これにより、ＤＦＭＡユニット３２０における全ての演算について、完了に要するサイクルが同数となり、これにより、命令発行論理回路が単純化される。

【０１４５】

Ｅ．ＤＳＥＴ

[0156]ＤＳＥＴ演算では、ＤＭＡＸ及びＤＭＩＮと同様に、オペランド準備ブロック５１４におけるＡ／Ｂ比較回路６２４（図６）が使用される。ＤＳＥＴでは、ＤＭＡＸ及びＤＭＩＮとは異なり、入力オペランドの一方が返されるのではなく、テストされた条件が満たされているかを示すブール値が返される。

【０１４６】

[0157]ＤＳＥＴ演算の場合、オペランド準備ブロック５１４（図６）が、オペランドＡ及びオペランドＢを受け取る。ＮａＮ回路６１２及び６１４は、選択されたオペランドのいずれか一方又は双方がＮａＮであるかを判定する。絶対値／符号反転回路６１８，６２０は、必要な場合に符号ビットを反転する。

【０１４７】
[0158]Ａ／Ｂ比較回路６２４は、絶対値／符号反転回路６１８，６２０からオペランドＡ及びオペランドＢを受け取り、例えば、これらのオペランドがあたかも整数であるかのようにＡからＢを減算することによって、比較を実行し、それぞれの符号ビットを考慮する。Ａ／Ｂ比較回路６２４は、この減算に基づいて、ＡがＢよりも大きい、ＡがＢよりも小さい、又はＡがＢに等しいかを示すＣＯＭＰ信号を生成する。ＣＯＭＰ信号は制御論理回路６３０に提供される。制御論理回路６３０は、対応するＲ＿Ｔｅｓｔ信号を生成し、さらに、Ａマルチプレクサ６３２、Ｂマルチプレクサ６３４、及びＣマルチプレクサ６３６用の選択信号を生成する。ＤＳＥＴ演算の結果はブール値であるため、一実施形態では、三つのマルチプレクサ６３２，６３４，６３６の全てがゼロオペランドを選択する。別の実施形態においては、マルチプレクサ６３２及び６３４がオペランドＡ及びオペランドＢを選択する。特殊数検出回路６３８及び６４０は、これらのオペランドが特殊数であるかを判定し、該当する特殊数ＳＰＣ信号を経路５２４上に生成する。

【０１４８】
[0159]ｆｐ３２抽出回路６２６及び符号なし／符号付き整数抽出回路６２８は、ＤＳＥＴ演算の場合には使用されず、これらの回路は、必要に応じて電力遮断状態又は低電力状態に設定され得る。

【０１４９】
[0160]仮数経路５１６、指数経路５１８、及び符号経路５２０は、ＤＦＭＡ演算について上述したように動作するか、又は、これらの一部又は全体をバイパスすることができる。バイパスされるコンポーネントは低電力状態に設定され得る。上述したように、バイパス経路が、仮数経路５１６、指数経路５１８、及び符号経路５２０のうちの最も長い経路と同じ数のパイプラインステージを占めるように、バイパス経路に様々な遅延回路（ラッチなど）を含めることができる。これにより、ＤＦＭＡユニット３２０における全ての演算について、完了に要するサイクルが同数となり、これにより、命令発行論理回路が単純化される。

【０１５０】
[0161]出力セクション５２２（図１０）は、特殊数を扱う。具体的には、ＩＥＥＥ７５４規格においては、Ａ又はＢ（又は双方）がＮａＮであるならば、Ａ及びＢは順序付けできない。出力マルチプレクサ制御論理回路１００２は、ＡがＢよりも大きい、ＡがＢよりも小さい、又はＡがＢに等しいかを示すＲ＿Ｔｅｓｔ信号と、Ａ又はＢがＮａＮであるかを示す特殊数ＳＰＣ信号と、要求されている特定のテスト演算を示すＯＰＣＴＬ信号とを受け取る。出力マルチプレクサ制御論理回路１００２は、Ｒ＿Ｔｅｓｔ信号及びＳＰＣ信号を使用して、要求されたテストが満たされているかを判定する。一実施形態においては、ＤＳＥＴ演算の結果を条件コードとして提供し、結果ＯＵＴを無視する。その場合、出力マルチプレクサ制御論理回路１００２は、結果を示すように条件コードＣＯＮＤを設定し、オプションとして、出力ＯＵＴの仮数部及び指数を選択することができる。別の実施形態においては、テスト結果を反映するように出力ＯＵＴを設定することができ、この場合、出力マルチプレクサ制御論理回路１００２は、テストが満たされているならば論理真に対応する６４ビット値が選択され、テストが満たされていないならば論理偽に対応する６４ビット値が選択されるように、仮数部マルチプレクサ１００４及び指数マルチプレクサ１００６を動作させる。

【０１５１】
Ｆ．形式変換
[0162]いくつかの実施形態においては、ＤＦＭＡユニット３２０は、倍精度形式とそれ以外の形式との間での形式変換演算もサポートする。以下に例を説明する。

【０１５２】
１．ｆｐ３２からｆｐ６４（Ｆ２Ｄ）
[0163]Ｆ２Ｄ演算の場合、ｆｐ３２入力オペランドＡが、対応するｆｐ６４数に変換される。特殊数入力は適切に扱われる。例えば、ｆｐ３２無限大又はｆｐ３２　ＮａＮは、

ｆｐ６４無限大又はｆｐ６４　ＮａＮに変換される。全てのｆｐ３２非正規数は、ｆｐ６４の正規数に変換することができる。

【０１５３】

[0164]オペランド準備ブロック５１４（図６）は、ｆｐ３２のオペランドＡを受け取る。絶対値／符号反転回路６１８は、オペランドＡを修正することなくｆｐ３２抽出ブロック６２６に通過させる。ｆｐ３２抽出ブロック６２６は、オペランドＡに対してｆｐ６４形式への最初の上位変換を実行する。具体的には、ｆｐ３２抽出ブロック６２６は、８ビットの指数を取り出し、１０２３－１２７＝８９６を加算して、ｆｐ６４形式の正しいバイアスを有する１１ビットの指数を生成する。２３ビットの仮数には、末尾の０がパディングされる。さらに、ｆｐ３２抽出ブロック６２６は、オペランドＡがｆｐ３２の特殊数（例：無限大、ＮａＮ、０、又は非正規）であるかを判定し、その情報を、経路６４４を介して特殊数検出回路６４２に提供する。さらに、ｆｐ３２抽出ブロック６２６は、オペランドの符号を反転する、或いはオペランドに絶対値を適用することもできる。

【０１５４】

[0165]オペランドＣのマルチプレクサ６３６は、ｆｐ３２抽出ブロック６２６によって提供される上位変換されたオペランドを選択し、オペランドＡのマルチプレクサ６３２及びオペランドＢのマルチプレクサ６３４は、ゼロオペランドを選択する。特殊数検出回路６４２は、オペランドがｆｐ３２非正規数でない限りは、仮数に先頭の１を付加する。さらに、特殊数検出回路６４２は、ｆｐ３２非正規数が正規数として特定されている場合を除き（全てのｆｐ３２非正規数はｆｐ６４において正規数として表現することができるため）、ｆｐ３２抽出ブロック６２６によって提供される特殊数情報を、自身の特殊数ＳＰＣ信号として使用する。

【０１５５】

[0166]仮数経路５１６及び指数経路５１８は、ＤＦＭＡ演算について上述したように動作して、ｆｐ６４形式において０．０＊０．０＋Ｃを計算する。仮数経路５１６及び指数経路５１８における正規化要素は、上位変換されたｆｐ６４オペランドを正規化する。代替実施形態においては、図８を参照し、指数経路５１８からの位置合わせ後の仮数Ｃ＿ａｌｉｇｎを、仮数経路５１６における加算器８０４の周囲をマルチプレクサ８０６のＳｕｍ入力にバイパスさせることができる。ここで、乗算器８０２及び加算器８０４を低電力状態に設定することができる。符号経路５２０は符号ビットＣｓを通過させることが好適である。

【０１５６】

[0167]出力セクション５２２（図１０）においては、入力オペランドがｆｐ３２無限大、ｆｐ３２　ＮａＮ、又はｆｐ３２　０であったことを特殊数ＳＰＣ信号が示していない限りは、正規化されたｆｐ６４の結果（Ｒｍ，Ｒｓ，Ｒｅ）が選択される。入力オペランドがｆｐ３２無限大であった場合、出力マルチプレクサ制御論理回路１００２は、ｆｐ６４無限大の仮数部（０ｘ０＿０００＿０００＿０００）が選択されるように仮数部マルチプレクサ１００４を動作させ、ｆｐ６４無限大の指数（０ｘ７ｆｆ）が選択されるように指数マルチプレクサ１００６を動作させる。入力オペランドがｆｐ３２　ＮａＮであった場合、出力マルチプレクサ制御論理回路１００２は、ｆｐ６４クワイエット型ＮａＮの仮数部が選択されるように仮数部マルチプレクサ１００４を動作させ、ｆｐ６４　ＮａＮの指数（０ｘ７ｆｆ）が選択されるように指数マルチプレクサ１００６を動作させる。入力オペランドがｆｐ３２　０であった場合、出力マルチプレクサ制御論理回路１００２は、ｆｐ６４　０の仮数部（０ｘ０＿０００＿０００＿０００）が選択されるように仮数部マルチプレクサ１００４を動作させ、ｆｐ６４　０の指数（０ｘ０００）が選択されるように指数マルチプレクサ１００６を動作させる。条件コードは、必要に応じて設定され得る。

【０１５７】

２．整数からｆｐ６４（Ｉ２Ｄ）

[0168]Ｉ２Ｄ演算の場合、整数（ｕ６４，ｓ６４，ｕ３２，又はｓ３２形式）がｆｐ６

４形式に変換される。オペランド準備ブロック５１４（図６）は、６４ビットの整数オペランドＣを受け取る。３２ビット整数形式の場合、３２個の先頭の０を付加することができる。絶対値／符号反転回路６２２は、オペランドＣを修正することなくＵ／Ｓ抽出ブロック６２８に通過させる。Ｕ／Ｓ抽出ブロック６２８は、オペランドＣに対してｆｐ６４形式への最初の上位変換を実行する。具体的には、抽出ブロック６２８は、オペランドＣにおける先頭の１の位置を（例えばプライオリティエンコーダを使用して）判定する。１１ビットの指数は、指数フィールドを１０８６（2^{63}に対応する）に初期化することによって求められる。３２ビット形式の入力の場合、先頭の１がドロップされ、仮数に末尾の０がパディングされて、５２ビットの仮数部が生成される。６４ビット形式の入力の場合、必要に応じて仮数が５３ビットに切り捨てられ、先頭の１がドロップされる。また、ガードビット及びラウンドビットが必要に応じて維持されてもよい。

【０１５８】

[0169]さらに、Ｕ／Ｓ抽出ブロック６２８は、入力オペランドが０であるかを判定し、特殊数検出回路６４２用の対応の制御信号を生成する。０以外の特殊数（非正規、無限大、及びＮａＮ）は、Ｉ２Ｄ演算時には発生せず、検出する必要がない。

【０１５９】

[0170]オペランドＣのマルチプレクサ６３６は、Ｕ／Ｓ抽出ブロック６２８によって提供される上位変換されたオペランドを選択する。オペランドＡのマルチプレクサ６３２とオペランドＢのマルチプレクサ６３４のそれぞれは、ゼロオペランドを選択する。特殊数検出回路６４２は、Ｕ／Ｓ抽出ブロック６２８によって提供されるゼロ情報を使用して、入力オペランドが０であるかを示す特殊数ＳＰＣ信号を生成する。

【０１６０】

[0171]仮数経路５１６及び指数経路５１８は、ＤＦＭＡ演算について上述したように動作し、０．０＊０．０＋Ｃを計算する。仮数経路５１６及び指数経路５１８における正規化要素は、上位変換されたｆｐ６４オペランドを正規化する。代替実施形態においては、図８を参照し、指数経路５１８からの位置合わせ後の仮数Ｃ＿ａｌｉｇｎを、仮数経路５１６における加算器８０４の周囲をマルチプレクサ８０６のＳｕｍ入力にバイパスさせることができる。乗算器８０２及び加算器８０４は低電力状態に設定され得る。符号経路５２０は符号ビットＣｓを通過させることが好適である。

【０１６１】

[0172]出力セクション５２２（図１０）においては、入力オペランドが整数０であったことを特殊数ＳＰＣ信号が示していない限りは、正規化されたｆｐ６４の結果（Ｒｍ，Ｒｓ，Ｒｅ）が選択される。入力オペランドが整数０であった場合、出力マルチプレクサ制御論理回路１００２は、ｆｐ６４　０の仮数部（０ｘ０＿０００＿０００＿０００）が選択されるように仮数部マルチプレクサ１００４を動作させ、ｆｐ６４　０の指数（０ｘ０００）が選択されるように指数マルチプレクサ１００６を動作させる。必要に応じて条件コードが設定され得る。

【０１６２】

３．ｆｐ６４からｆｐ３２（Ｄ２Ｆ）

[0173]ｆｐ６４は、ｆｐ３２よりも広い範囲の浮動小数点数をカバーしているので、ｆｐ６４からｆｐ３２への（Ｄ２Ｆ）変換では、ｆｐ３２値におけるオーバーフロー及びアンダーフローを検出することが要求される。

【０１６３】

[0174]Ｄ２Ｆ演算の場合、オペランド準備ブロック５１４（図６）にオペランドＣが提供される。絶対値／符号反転回路６２２は、必要に応じて絶対値又はオペランドの符号反転を実行し、オペランドＣをオペランドＣのマルチプレクサ６３６に渡す。オペランドＣのマルチプレクサ６３６は、オペランドＣを選択し、特殊数検出回路６４２に提供する。特殊数検出回路６４２は、ｆｐ６４非正規数、ｆｐ６４　０、ｆｐ６４無限大、又はｆｐ６４　ＮａＮを検出し、対応するＳＰＣ信号を出力セクション５２２に提供する。選択マルチプレクサ６３２及び６３４は、オペランド０．０を選択する。

【０１６４】
[0175]指数経路５１８（図７）においては、指数計算ブロック７０２は、ｆｐ６４指数を８９７だけマイナス方向にバイアスし、対応するｆｐ３２の指数を求める。ｆｐ３２の指数がアンダーフローする場合、指数計算ブロック７０２は、アンダーフローが排除されるようにＣの仮数を右シフトするＳｈ＿Ｃ信号を生成する（２１８ビット以上のシフトが必要である場合、Ｃの仮数は０になる）。シフト回路７０６は、Ｓｈ＿Ｃ信号に従ってＣの仮数を右シフトする。結果はマルチプレクサ７１２によって選択され、位置合わせ後の仮数Ｃ＿ａｌｉｇｎとして仮数経路５１６に提供される。アンダーフロー論理回路７１３は、ｆｐ３２アンダーフローを検出し、Ｕ＿ｆｐ３２信号を生成する。
【０１６５】
[0176]仮数経路５１６（図８）においては、乗算器８０２が、積０．０＊０．０を計算する（又はバイパスされる）。この積（０）は、加算器８０４によって仮数Ｃ＿ａｌｉｇｎに加算される。マルチプレクサ８０６は、結果Ｓｕｍを選択する（入力が符号・絶対値形式であるため）。０の結果は、回路８１４によって検出される。０以外の結果は、ＤＦＭＡ演算に関して上述したように正規化される。丸め論理回路８２０を使用して、切り上げるかを判定することができる。なお、結果は２３ビットのｆｐ３２仮数であるため、プラス１加算器８２２は、（５３番目ではなく）２４番目のビット位置に１を加算する必要があることに留意されたい。
【０１６６】
[0177]出力セクション５２２（図１０）は、結果を組み立てる。ｆｐ３２の２３ビットの仮数部は、５２ビットフィールドのＲｍにおいて提供される。出力マルチプレクサ制御論理回路１００２は、結果がｆｐ３２の正規数ではない場合を除き、Ｒｍが選択されるように仮数部マルチプレクサ１００４を制御する。ｆｐ３２　０又はｆｐ３２無限大である場合、０の仮数０ｘ００００００＿００００＿００００が選択され、ｆｐ３２　ＮａＮである場合、クワイエット型ｆｐ３２　ＮａＮの仮数が選択される。ｆｐ３２非正規数の場合、Ｒｍを使用することができる。
【０１６７】
[0178]８ビットのｆｐ３２の指数は、１１ビットの指数フィールドにおいて提供される。出力マルチプレクサ制御論理回路１００２は、結果がｆｐ３２の正規数ではない場合を除き、Ｒｅが選択されるように指数マルチプレクサ１００４を制御する。ｆｐ３２非正規数又はｆｐ３２　０である場合、０の指数０ｘ０００が選択される。ｆｐ３２無限大又はｆｐ３２　ＮａＮである場合、ｆｐ３２の最大指数０ｘ７ｆｆが選択される。
【０１６８】
[0179]連結ブロック１００８は、Ｒｍ及びＲｅを、６４ビットの出力フィールドのうちの３１ビットにパックし、符号ビットＲｓを先頭に付加する。１１ビットの指数における３個のＭＳＢがドロップされ、５２ビットの仮数部における２９個のＬＳＢがドロップされる。ｆｐ３２の結果は、例えば、必要に応じて６４ビットフィールドのＭＳＢ又はＬＳＢにおいて位置合わせすることができる。必要に応じて条件コードを設定することができる。
【０１６９】
４．ｆｐ６４から整数（Ｄ２Ｄ）
[0180]Ｄ２Ｉ演算の場合、オーバーフロー及びアンダーフローが検出される。オーバーフローは最大整数値にセットされ、アンダーフローは０にセットされる。
【０１７０】
[0181]変換するオペランドは、ｆｐ６４形式におけるオペランドＣとして提供される。絶対値／符号反転回路６２２は、必要に応じて絶対値又はオペランドの符号反転を実行し、オペランドＣをオペランドＣのマルチプレクサ６３６に渡す。オペランドＣのマルチプレクサ６３６は、オペランドＣを選択し、特殊数検出回路６４２に提供する。特殊数検出回路６４２は、ｆｐ６４非正規数、ｆｐ６４　０、ｆｐ６４無限大、又はｆｐ６４　ＮａＮを検出し、対応するＳＰＣ信号を出力セクション５２２に提供する。選択マルチプレク

サ６３２及び６３４は、オペランド０．０を選択する。

【０１７１】

[0182]指数経路５１８（図７）においては、指数計算ブロック７０２は、２進小数点を整数位置に合わせるためにＣｍをシフトすべき量を、指数Ｃｅを使用して求め、対応するＳｈ＿Ｃ信号を生成する。一実施形態においては、指数計算ブロック７０２は、指数のバイアスを取り除き、仮数部の幅と、使用される整数形式と、３２ビット形式の結果を６４ビットフィールドにおいて表現する方式（例：３２個のＭＳＢ又は３２個のＬＳＢを使用する）とを考慮する。さらに、変換後の整数形式において結果がオーバーフローするか、又はアンダーフローするかを、指数Ｃｅを使用して判定する。オーバーフロー又はアンダーフローする場合、対応するオーバーフロー信号又はアンダーフロー信号（図示していない）が、出力セクション５２２における出力マルチプレクサ制御論理回路１００２（図１０）に送られることが好適である。

【０１７２】

[0183]シフト回路７０６は、Ｃｍを量Ｃ＿Ｓｈｉｆｔだけシフトさせ、このＣ＿Ｓｈｉｆｔ信号がマルチプレクサ７１２によってＣ＿ａｌｉｇｎ信号として選択される。

【０１７３】

[0184]仮数経路５１６（図８）においては、乗算器８０２が、結果０．０を加算器８０４に提供する。加算器８０４は、０．０をＣ＿ａｌｉｇｎに加算し、Ｃが正であるか負であるかに応じて、Ｓｕｍ又は～Ｓｕｍが選択される。シフター８１６は、この結果をシフトしないことが好適である。整数フォーマッティングブロック８２６は、結果を、１１ビットのＭＳＢフィールドｉｎｔ＿Ｍと、５３ビットのＬＳＢフィールドｉｎｔ＿Ｌとに分ける。

【０１７４】

[0185]出力セクション５２２（図１０）においては、出力マルチプレクサ制御論理回路１００２が、オーバーフロー、アンダーフロー、又は特殊数オペランドである場合を除いて、結果のｉｎｔ＿Ｌ及び結果のｉｎｔ＿Ｍがそれぞれ選択されるように仮数部マルチプレクサ１００４及び指数マルチプレクサ１００６を制御する。オーバーフローの場合、出力形式（ｕ３２，ｓ３２，ｕ６４，又はｓ６４）における最大整数が選択され、アンダーフローの場合、０が選択される。条件コードは、必要に応じて設定され得る。

【０１７５】

ＩＶ．更なる実施形態

[0186]本発明について、特定の実施形態に関連して説明してきたが、膨大な修正・変更が可能であることが当業者には認識されるであろう。例えば、ＤＦＭＡユニットは、より多くの機能、より少ない機能、又は異なる機能が、組合せとしてサポートされるように実施されてもよく、或いは、任意の形式又は任意の形式の組合せにおけるオペランド及び結果がサポートされるように実施されてもよい。

【０１７６】

[0187]本明細書において説明した様々なバイパス経路及び通過経路は、異なる形態をとることもできる。一般的に、回路ブロックの周囲のバイパス経路が説明してある箇所では、その経路を、そのブロックにおける恒等演算（すなわち、オペランドに影響を与えない演算（例：０を加算する））に置き換えることができる。ある演算時にバイパスされる回路ブロックは、アイドル状態（例えば、低電力状態）に設定することができ、或いは、通常に動作させ、（例えば、選択マルチプレクサ又はその他の回路の動作によって）下流のブロックが結果を無視するようにすることができる。

【０１７７】

[0188]ＤＦＭＡパイプラインは、任意の数のステージに分割することができ、ステージそれぞれにおけるコンポーネントの組合せは、必要に応じて異なる組合せとすることができる。さらに、本明細書において特定の回路ブロックによって提供される機能を、複数のパイプラインステージにまたがるように分割することができる。例えば、乗算器のツリーが複数のステージを占めることができる。さらに、様々なブロックの機能を修正・変更す

ることができる。いくつかの実施形態においては、例えば、異なる加算器回路又は異なる乗算器回路を使用することができる。

【０１７８】

[0189]さらに、ＤＦＭＡユニットについて、理解を促進する目的で、回路ブロックに基づいて説明してきた。これらの回路ブロックは、様々な回路コンポーネント及び様々なレイアウトを使用して実施することができ、本明細書に説明してあるブロックが、特定の一連のコンポーネント又は特定の物理的レイアウトに限定されないことが、当業者には認識されるであろう。ブロックは、必要に応じて、物理的に結合する、又は分けることができる。

【０１７９】

[0190]プロセッサは、実行コアの中に一以上のＤＦＭＡユニットを含んでいてもよい。例えば、スーパースケーラ命令発行方式（すなわち、１サイクルあたり二つ以上の命令を発行する）又はＳＩＭＤ命令発行方式が望ましい場合、複数のＤＦＭＡユニットを実施することができ、異なるＤＦＭＡユニットが、異なる機能の組合せをサポートすることができる。さらに、プロセッサは、複数の実行コアを含んでいてもよく、コアのそれぞれが自身の（一つ以上の）ＤＦＭＡユニットを有することができる。

【０１８０】

[0191]いくつかの実施形態においては、実行コアがＳＩＭＤ命令発行をサポートし、一つのＤＦＭＡパイプラインにおいて複数のデータセットを連続的に処理することができるように、一つのＤＦＭＡユニットを、入力の順序付け及び出力収集のための適切な論理回路と組み合わせて使用することができる。

【０１８１】

[0192]図１１は、本発明の実施形態による、ＤＦＭＡ機能ユニット１１０２を含む実行コア１１００のブロック図である。ＤＦＭＡユニット１１０２は、上述したＤＦＭＡユニット３２０に類似するユニット、又は同じユニットとすることができる。コア１１００はＳＩＭＤ命令を発行する、すなわち、Ｐ組の異なるセットの単精度オペランドを有する同じ命令を、Ｐ個のセットの単精度ＳＩＭＤユニット１１０４に並列に発行することができる。ＳＩＭＤユニット１１０４のそれぞれは、同じオペコードと、異なるセットのオペランドとを受け取る。Ｐ個のＳＩＭＤユニット１１０４は、並列に動作してＰ個の結果を生成する。ＤＦＭＡユニット１１０２には、Ｐ－ｗａｙ　ＳＩＭＤ命令が、Ｐ個の一連のＳＩＳＤ（単一命令単一データ）命令として発行される。

【０１８２】

[0193]入力マネージャ１１０６（命令発行ユニットの一部とすることができる）は、ＳＩＭＤ命令のオペランドを集め、ＳＩＭＤ命令のＰ個のセットのオペランド全てが集まったとき、それらオペランド及び適用可能なオペコードを、Ｐ個のＳＩＭＤユニット１１０４又はＤＦＭＡユニット１１０２のいずれかに提供する。出力収集器１００８は、ＳＩＭＤユニット１１０４又はＤＦＭＡユニット１１０２からの結果を集め、それらの結果を、結果バス１１１０を介してレジスタファイル（図１１には示していない）に提供する。いくつかの実施形態においては、結果バス１１１０は、入力マネージャ１１０６へのバイパス経路も提供し、従って、結果を次の命令において使用することができるように、結果をレジスタファイルに提供するのと並列に入力マネージャ１１０６に提供することができる。一つのＤＦＭＡユニット１１０２を使用して、見かけ上のＳＩＭＤ動作を提供する目的で、入力マネージャ１１０６は、例えば、Ｐ個の連続するクロックサイクルのそれぞれにおいて、異なるセットのオペランドを有する同じオペコードを発行することによって、ＤＦＭＡユニット１１０２への命令の発行をシリアル化することが有利である。

【０１８３】

[0194]図１２は、本発明の実施形態による、ＤＦＭＡユニット１１０２のためのシリアル化された命令発行方式を示しているブロック図である。入力オペランド収集ユニット１２０２（図１１の入力マネージャ１１０６に含めることができる）は、二つの収集器１２０４，１２０６を含んでいる。収集器１２０４，１２０６のそれぞれは、３２ビットレジ

スタの配列であり、Ｐ組の単精度オペランドトリプレットＡ，Ｂ，及びＣのための十分な空間を提供する。言い換えれば、収集器１２０４，１２０６のそれぞれは、一つのＳＩＭＤ命令の全てのオペランドを格納することができる。入力オペランド収集ユニット１２０２は、例えば図３のレジスタファイル３２４及び／又は図１１の結果バス１１１０から、オペランドを取得する。与えられた命令に対して、どのオペランドを集めるかを判定するために、タグ、又はその他の従来の手法を使用することができる。一つの命令のオペランドを、その命令が発行されるよりも数クロックサイクル前に集めることができるための十分な収集器１２０６が提供される。

【０１８４】

[0195]単精度命令の場合、Ｐ個のＳＩＭＤユニット１１０４が一つの命令を実行するために必要なオペランドの全てが、一方の収集器（例：収集器１２０４）にロードされる。Ｐ個のＳＩＭＤユニット１１０４に命令が発行されるとき、収集器１２０４全体が並列に読み取られて、ＳＩＭＤユニット１１０４のそれぞれに異なるオペランドトリプレットＡ，Ｂ，Ｃが提供されることが好適である。

【０１８５】

[0196]ＤＦＭＡユニット１１０２への命令の場合、オペランドは倍精度（例：６４ビット）である。オペランドのそれぞれは、双方の収集器１２０４，１２０６の中の対応するレジスタを使用して格納することができる。例えば、収集器１２０４の中のレジスタ１２０８は、オペランドＡの一つのインスタンスの３２個のＭＳＢ（例：符号ビット、１１個の指数ビット、及び仮数部の２０個のＭＳＢ）を格納することができ、その一方で、収集器１２０６の中のレジスタ１２１０は、同じオペランドの３２個のＬＳＢ（例：仮数部の残りの３２ビット）を格納する。このように、倍精度のＰ－ｗａｙ　ＳＩＭＤ命令に必要なオペランドトリプレットＡ，Ｂ，Ｃの全てを、二つの単精度収集器１２０４，１２０６を使用して集めることができる。

【０１８６】

[0197]コア１１００は、一つのＤＦＭＡユニット１１０２のみを備えており、Ｐ個のセットのオペランドは、出力マルチプレクサ（ＭＵＸ）１２１２，１２１４（いずれもカウンタ１２１６によって制御される）を使用して連続的に提供されることが好適である。マルチプレクサ１２１２及び１２１４は、カウンタ１２１６に応答して、それぞれの収集器１２０４及び１２０６からのオペランドトリプレットのＭＳＢ及びＬＳＢを選択する。例えば、図示したデータ経路においては、マルチプレクサ１２１２は、収集器１２０４の中のレジスタ１２０８からのオペランドＡの３２個のＭＳＢを選択することができ、その一方で、マルチプレクサ１２１４は、収集器１２０６の中のレジスタ１２１０からの同じオペランドＡの３２個のＬＳＢを選択することができる。これら６４ビットが、倍精度幅の経路を通じてＤＦＭＡユニット１１０２に提供される。同様に、（レジスタ１２２０及びレジスタ１２２２からの）オペランドＢと（レジスタ１２２４及びレジスタ１２２６からの）オペランドＣとを、同じカウンタ１２１６によって制御される対応するマルチプレクサ（図示していない）を使用して、ＤＦＭＡユニット１１０２に提供することができる。次のクロックサイクルにおいては、収集器１２０４及び１２０６の中の次のセットのレジスタからのオペランドＡ，Ｂ，及びＣをＤＦＭＡユニット１１０２に提供することができ、Ｐ個のセットのオペランドの全てが提供されるまで、以下同様に繰り返す。

【０１８７】

[0198]マルチプレクサ１２１２及び１２１４は、収集器１２０４及び１２０６とともに、ＤＦＭＡユニット１１０２における見かけ上のＳＩＭＤ実行を提供する（ただしスループットは低下する）。従って、コア１１００のプログラミングモデルでは、全ての命令（倍精度命令を含む）についてＰ－ｗａｙ　ＳＩＭＤの実行が利用可能であることを想定することができる。

【０１８８】

[0199]本明細書に説明したオペランドの収集及び順序付けの論理回路は、例示を目的としており、変形及び変更が可能であることが理解されるであろう。ＳＩＭＤ対応型のコア

には、任意の数のＤＦＭＡユニットを提供することができ、任意の数のＤＦＭＡユニットに並列に命令を発行することができる。いくつかの実施形態においては、単精度演算に対する倍精度演算におけるスループットは、ＤＦＭＡユニットの数に対応して増減する。例えば、Ｐ個のＳＩＭＤユニットとＮ個のＤＦＭＡユニットとが存在している場合、倍精度におけるスループットは、単精度におけるスループットのＮ／Ｐである。いくつかの実施形態においては、ＮはＰに等しいことが最適である。別の実施形態においては、別の要因（例：レジスタファイルと機能ユニットとの間の内部データ経路の幅）によって、倍精度におけるスループットが、存在するＤＦＭＡユニットの数には関係なく、単精度におけるスループットより低い値に制限されることがある。その場合、Ｎは、その別の制限要因下において可能な値よりも大きくないことが最適である。

【０１８９】

[0200]さらに、ＤＦＭＡユニットが単精度の機能ユニットとは個別であるため、ＤＦＭＡユニットが使用されないとき、例えば、グラフィックスプロセッサ又はコアが、レンダリングプロセス、或いは倍精度を必要としないその他の計算のみに使用されているとき、ＤＦＭＡユニットの電力を落とすことができることに留意されたい。さらには、ＤＦＭＡユニットを、それ以外の回路コンポーネントの動作に影響を与えることなく、集積回路の設計から省くことができる。これにより、異なるチップによって倍精度演算の異なるサポートレベルが提供される製品ファミリーの設計が容易になる。例えば、ＧＰＵファミリーは、それぞれが少なくとも一つのＤＦＭＡユニットを含む多数のコアを有する高性能のＧＰＵと、倍精度がハードウェアベースでサポートされず、ＤＦＭＡユニットが存在しない低性能のＧＰＵとを含んでいてもよい。

【０１９０】

[0201]さらに、本発明をグラフィックスプロセッサに関連して説明してきたが、別のプロセッサ（例えば、数学コプロセッサ、ベクトルプロセッサ、又は汎用プロセッサ）においても本発明の態様を採用することができることが、当業者には理解されるであろう。

【０１９１】

[0202]このように、本発明を特定の実施形態に関連して説明してきたが、本発明は、請求項の範囲内のあらゆる変形形態及び均等の形態を包含することを理解されたい。

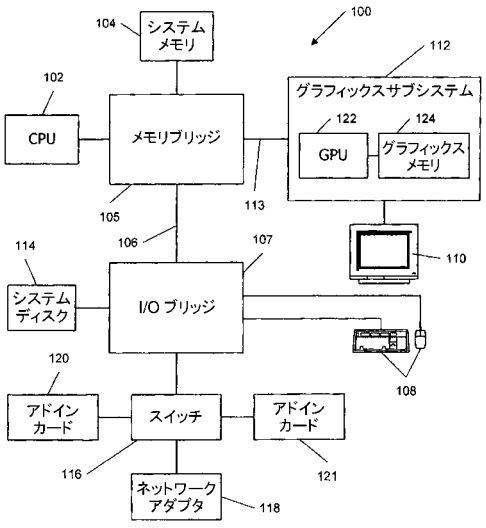
【符号の説明】

【０１９２】

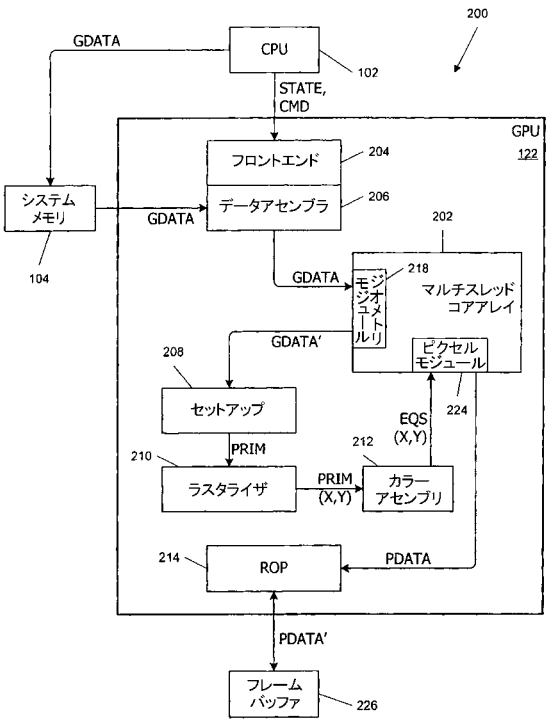
１００…コンピュータシステム、１０２…ＣＰＵ、１０４…システムメモリ、１０５…メモリブリッジ、１０６…通信経路、１０７…Ｉ／Ｏブリッジ、１０８…ユーザ入力装置、１１０…表示装置、１１２…グラフィックスサブシステム、１１３…通信経路、１１４…システムディスク、１１６…スイッチ、１１８…ネットワークアダプタ、１２０，１２１…アドインカード、１２２…ＧＰＵ、１２４…グラフィックスメモリ、２００…レンダリングパイプライン、２０２…マルチスレッドコアアレイ、２０４…フロントエンド、２０６…データアセンブラ、２０８…セットアッププモジュール、２１０…ラスタライザ、２１２…カラーアセンブリモジュール、２１４…ラスターオペレーションモジュール（ＲＯＰ）、２１８…ジオメトリモジュール、２２４…ピクセルモジュール、２２６…フレームバッファ、３００…実行コア、３０２…フェッチ・ディスパッチユニット、３０４…発行ユニット、３２０…ＤＦＭＡユニット、３２２…機能ユニット、３２４…レジスタファイル、３２６…データ転送経路、５０２，５０４，５０６…オペランド入力経路、５０８…オペコード経路、５１０，５１２，５２４…信号経路、５１４…オペランド準備、５１６…仮数経路、５１８…指数経路、５２０…符号経路、５２２…出力セクション、５３０…制御ブロック、６１２，６１４，６１６…ＮａＮ検出ブロック、６１８，６２０，６２２…絶対値／符号反転ブロック、６２４…Ａ／Ｂ比較回路、６２６…ＦＰ３２抽出回路、６２８…Ｕ／Ｓ抽出回路、６３０…制御論理回路、６３２…Ａマルチプレクサ、６３４…Ｂマルチプレクサ、６３６…Ｃマルチプレクサ、６３８，６４０，６４２…特殊数検出回路、７０２…指数計算回路、７０４…最終指数計算回路、７０６…シフト回路、７０８…符号反転回路、７１０…Ｄ２Ｄ論理回路、７１２…マルチプレクサ、７１３…アンダーフロ

ー論理回路、７２０…減算回路、７２２…アンダーフロー／オーバーフロー回路、７２４…プラス１回路、７２６…マルチプレクサ、８０２…５３×５３乗算器、８０４…１６８ビット加算器、８０６…マルチプレクサ、８０８，８１０…ＬＺＤ、８１２…マルチプレクサ、８１４…ゼロ検出回路、８１６…シフト回路、８１８…正規化論理回路、８２０…丸め論理回路、８２２…プラス１加算器、８２４…マルチプレクサ、８２６…フォーマッティング、９０２…積／和回路、９０４…最終符号回路、１００２…出力マルチプレクサ制御論理回路、１００４…仮数部マルチプレクサ、１００６…指数マルチプレクサ、１００８…連結ブロック、１１０２…ＤＦＭＡユニット、１１０４…ＳＩＭＤユニット、１１０６…入力マネージャ、１１０８…出力収集器、１１１０…結果バス、１２０２…オペランド収集、１２０８（０）～１２０８（Ｐ－１），１２１０（０）～１２１０（Ｐ－１），１２２０，１２２４，１２２６…レジスタ、１２１２，１２１４…マルチプレクサ、１２１６…カウント。

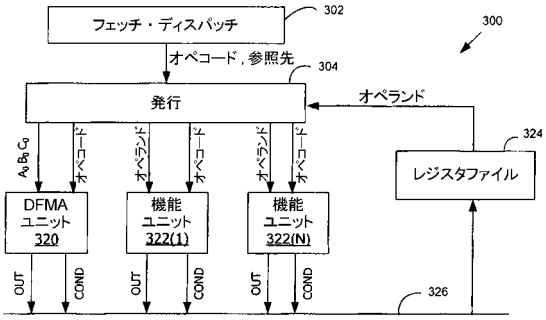
【図１】



【図２】



【図３】



【図４】

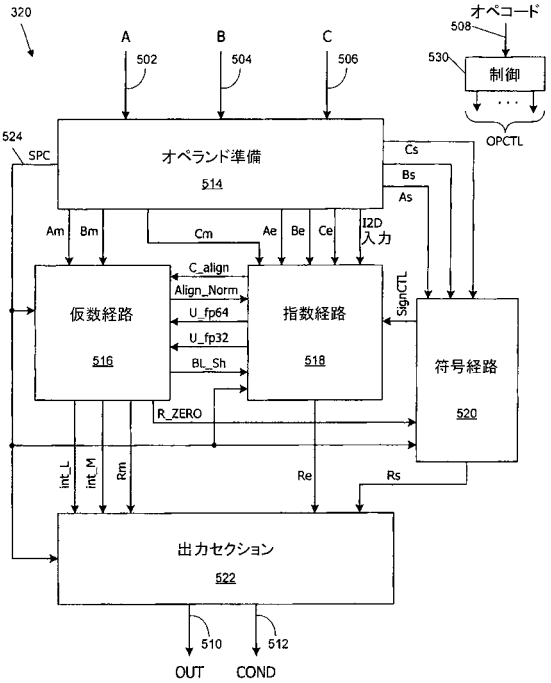
400

倍精度算術演算 402		
名称	入力	結果
DADD	A,C: fp64	A+C
DMUL	A,B: fp64	A*B
DFMA	A,B,C: fp64	A*B+C: fp64

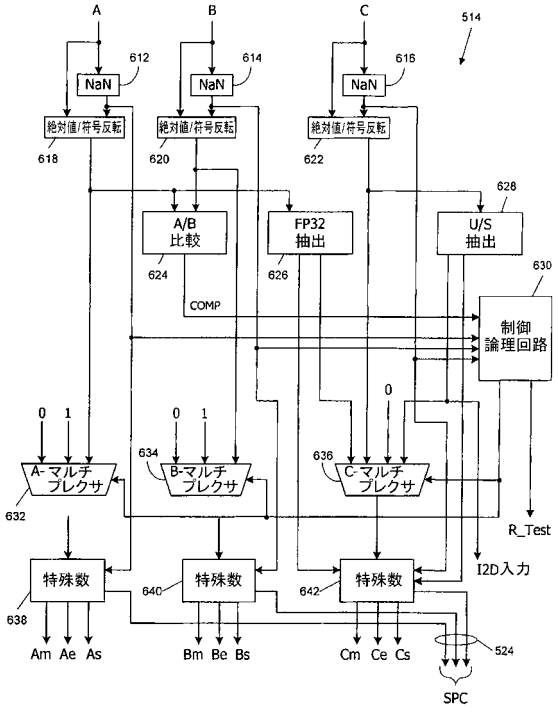
倍精度比較演算 404		
名称	入力	結果
DMAX	A,B: fp64	max(A,B)
DMIN	A,B: fp64	min(A,B)
DSET	A,B: fp64	R:ブール値

形式変換及び丸め 406		
名称	入力	結果
D2F	A: fp64	A': fp32
F2D	A: fp32	A': fp64
D2I	A: fp64	A': u/s64 or u/s32
I2D	C: u/s64 or u/s32	C': fp64
D2D	A: fp64	A': fp64

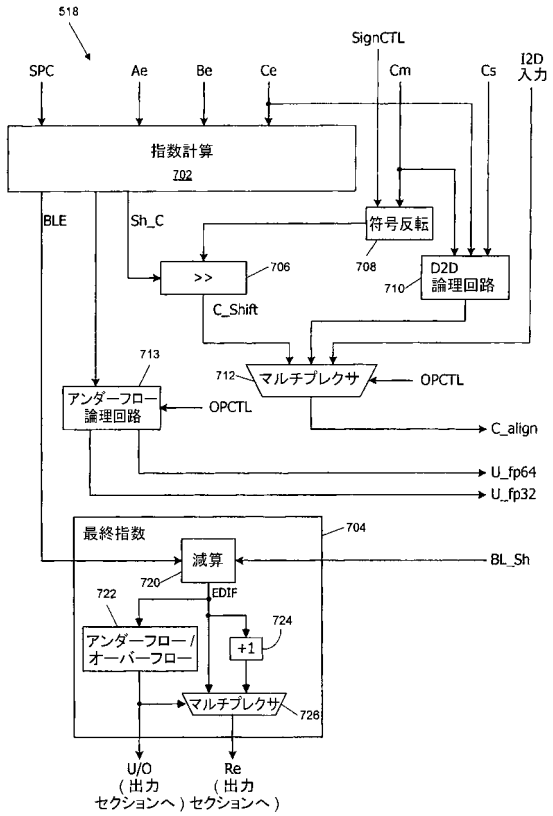
【図５】



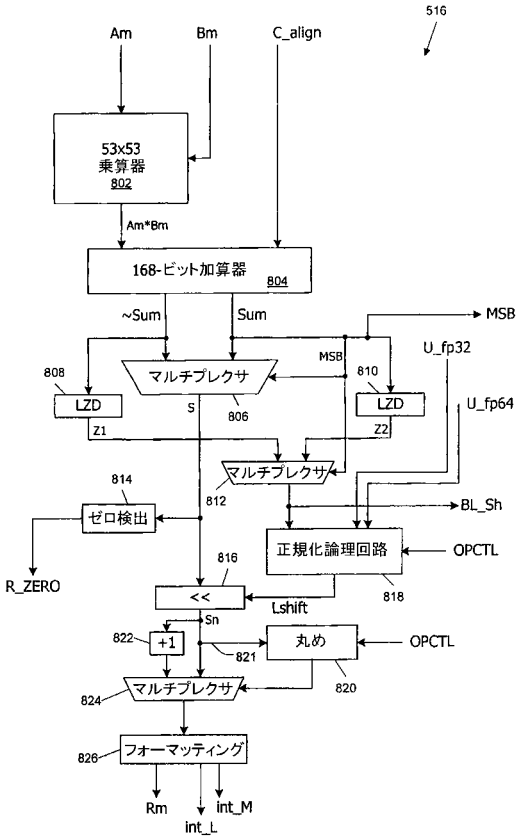
【図６】



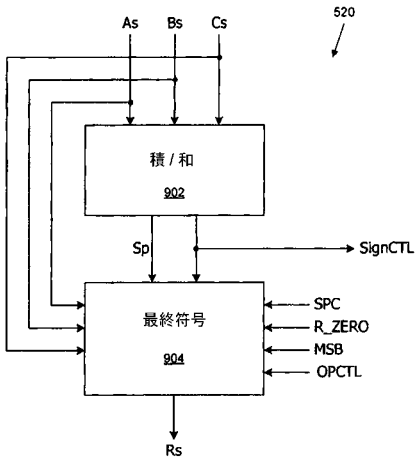
【図７】


518
SPC
Ae
Be
Ce
SignCTL
Cm
Cs
I2D 入力
指数計算 702
BLE
Sh_C
符号反転
708
>>
706
C_Shift
D2D 論理回路
710
713
アンダーフロー論理回路
OPCTL
712
マルチプレクサ
OPCTL
C_align
U_fp64
U_fp32
最終指数
704
減算
BL_Sh
720
EDIF
722
アンダーフロー / オーバーフロー
+1
724
マルチプレクサ
726
U/O （出力セクションへ）
Re （出力セクションへ）

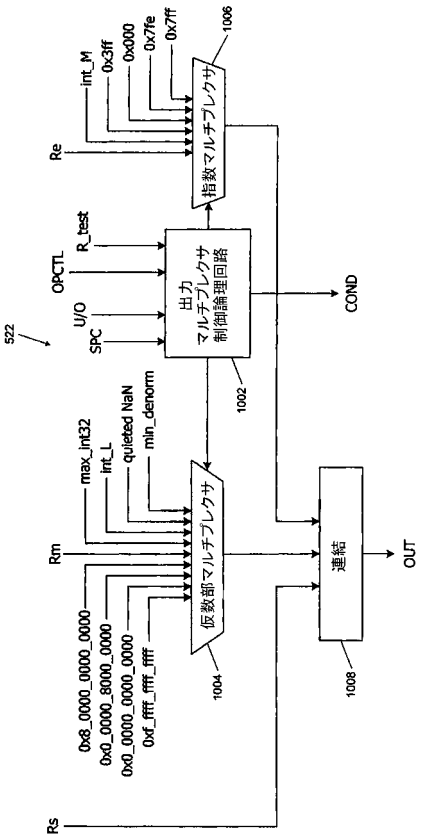
【図８】


516
Am
Bm
C_align
53x53 乗算器 802
Am*Bm
168-ビット加算器 804
~Sum
Sum
MSB
U_fp32
808
LZD
Z1
マルチプレクサ
S
806
MSB
810
LZD
Z2
U_fp64
マルチプレクサ
814
ゼロ検出
812
BL_Sh
正規化論理回路
OPCTL
R_ZERO
816
<<
Lshift
818
Sn
822
+1
821
丸め
OPCTL
820
824
マルチプレクサ
826
フォーマッティング
Rm
int_M
int_L

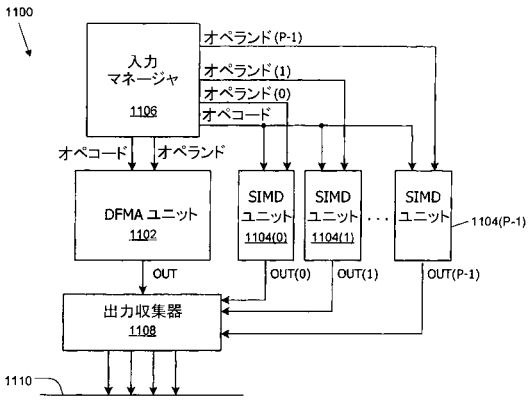
【図９】


As
Bs
Cs
520
積 / 和 902
Sp
SignCTL
最終符号 904
SPC
R_ZERO
MSB
OPCTL
Rs

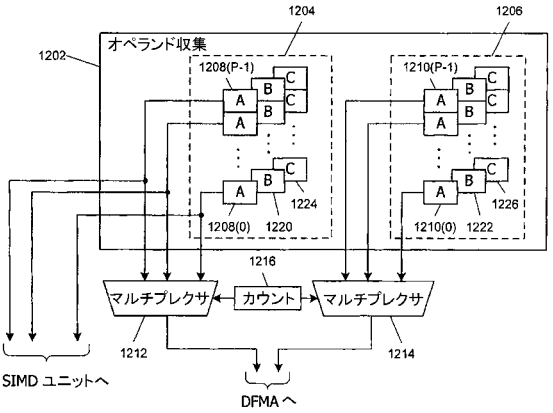
【図１０】


1006
int_M
0x3ff
0x000
0x7fe
0x7ff
Re
指数マルチプレクサ
R_test
OPCTL
出力マルチプレクサ制御論理回路
COND
U/O
SPC
522
1002
max_int32
int_L
quieted NaN
min_denorm
Rm
仮数部マルチプレクサ
連結
OUT
0x8_0000_0000_0000
0x0_0000_8000_0000
0x0_0000_0000_0000
0xf_ffff_ffff_ffff
1004
1008
Rs

【図１１】


1100
入力
マネージャ
1106
オペランド(P-1)
オペランド(1)
オペランド(0)
オペコード
オペコード
オペランド
DFMA ユニット
1102
SIMD
ユニット
1104(0)
SIMD
ユニット
1104(1)
SIMD
ユニット
1104(P-1)
OUT
OUT(0)
OUT(1)
OUT(P-1)
出力収集器
1108
1110

【図１２】


1204
1206
1202
オペランド収集
1208(P-1)
1210(P-1)
A
B
C
1224
1226
1208(0)
1220
1210(0)
1222
1216
マルチプレクサ
カウント
マルチプレクサ
1212
1214
SIMD ユニットへ
DFMA へ

フロントページの続き

(72)発明者　ミン　ワイ.　シウ
アメリカ合衆国,　カリフォルニア州,　サンタ　クララ,　サン　トマス　エクスプレスウェイ　２７０１
(72)発明者　デイヴィッド　シー.　タネンバウム
アメリカ合衆国,　カリフォルニア州,　サンタ　クララ,　サン　トマス　エクスプレスウェイ　２７０１

Ｆターム(参考) 5B056 AA05 BB71 CC01 FF01 FF02
5B057 AA20 CH04 CH05 CH09

【外国語明細書】
2009140491000001.pdf