(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4724346号
(P4724346)
(45)発行日　平成23年7月13日(2011.7.13)　(24)登録日　平成23年4月15日(2011.4.15)

(51)Int.Cl.　　F I
G06T 15/00 (2011.01)　　G06T 15/00　１００Ａ
G06T 1/20 (2006.01)　　G06T 1/20　Ｃ

請求項の数 8　(全 49 頁)

(21)出願番号　特願2001-542049 (P2001-542049)
(86)(22)出願日　平成12年12月5日 (2000.12.5)
(65)公表番号　特表2003-515851 (P2003-515851A)
(43)公表日　平成15年5月7日 (2003.5.7)
(86)国際出願番号　PCT/US2000/033043
(87)国際公開番号　WO2001/041069
(87)国際公開日　平成13年6月7日 (2001.6.7)
審査請求日　平成19年11月7日 (2007.11.7)
(31)優先権主張番号　09/456,102
(32)優先日　平成11年12月6日 (1999.12.6)
(33)優先権主張国　米国(US)

(73)特許権者　501261300
エヌヴィディア　コーポレイション
アメリカ合衆国，カリフォルニア　９５０５０，サンタ　クララ，サン　トーマス　エクスプレスウェイ　２７０１
(74)代理人　100094318
弁理士　山田　行一
(74)代理人　100123995
弁理士　野田　雅一
(74)代理人　100107456
弁理士　池田　成人
(72)発明者　リンドホルム、ジョン
アメリカ合衆国、カリフォルニア州　９５０１４　クパーティノ、バージニア・スワン・プレイス　１０３２５
最終頁に続く

(54)【発明の名称】グラフィックプロセッサ中の変換モジュール用の方法、装置および製品

(57)【特許請求の範囲】
【請求項１】
（ａ）頂点データをベクトルの形態で受取り、ベクトル頂点データに関してベクトル演算を行うためのベクトル演算モジュールと、
（ｂ）ベクトル演算モジュールからのスカラー頂点データをベクトル頂点データに変換するための、ベクトル演算モジュールに結合された変換モジュールと、
（ｃ）ベクトル演算モジュールの出力を記憶して、その出力をベクトル演算モジュールにフィードバックするための、ベクトル演算モジュールに結合されたレジスタとを具備しているグラフィック処理中にスカラーおよびベクトル成分を処理するためのシステム。
【請求項２】
ベクトル演算モジュールは乗算器および加算器の少なくとも一方を含んでいる請求項１記載のシステム。
【請求項３】
ゼロレイテンシーはレジスタをバイパスすることによって達成される請求項１記載のシステム。
【請求項４】
レジスタは、ベクトル頂点データを発生するベクトル成分書込みマスクを含んでいる請求項３記載のシステム。
【請求項５】
さらに、ベクトル演算モジュールの出力に関してスカラー演算を実行し、それによって

頂点データをスカラーの形態でレンダリングするための適合されたスカラー演算モジュールを備えている請求項１記載のシステム。
【請求項６】
スカラー演算には、逆数または逆平方根演算が含まれる請求項５記載のシステム。
【請求項７】
（ａ）複数のマトリックス、複数の加重値および頂点データをバッファにおいて受取り、
（ｂ）複数の積の和を計算し、各積は頂点データと、マトリックスの１つと、および加重値の少なくとも１つとの乗算により計算されたものであり、
（ｃ）付加的な処理のために積の和を出力するステップを含んでおり、
（ｄ）ステップ（ａ）乃至（ｃ）は、バッファの出力に結合された第１の入力を有する乗算論理装置と、この乗算論理装置の出力に結合された第１の入力を有する演算論理装置と、この演算論理装置の出力に結合された入力を有するレジスタ装置と、演算論理装置またはレジスタ装置の出力に結合された入力を備えた反転論理装置と、この反転論理装置の出力と乗算論理装置の第２の入力との間に結合された変換モジュールと、乗算論理装置および演算論理装置に結合されたメモリと、ならびに演算論理装置の出力に結合された出力コンバータとを備えた単一の集積回路によって行われ、反転論理装置は逆数または逆平方根演算を行い、変換モジュールはスカラー頂点データをベクトル頂点データに変換するように構成されており、メモリには頂点データを処理するときに使用される複数の定数および変数が記憶されており、出力コンバータは処理された頂点データを出力するようにライティングモジュールに結合されている、ハードウェア構成グラフィックパイプラインにおけるグラフィック処理中にブレンディング演算を行う方法。
【請求項８】
（ａ）複数のマトリックスと、複数の加重値と、および頂点データとを受取るためのバッファと、
（ｂ）このバッファに結合された単一の集積回路とを備えており、この集積回路は複数の積の和を計算し、各積は頂点データと、マトリックスの１つと、および加重値の少なくとも１つとの乗算により計算されたものであり、
（ｃ）積の和は付加的な処理のために単一の集積回路から出力され、単一の集積回路は、バッファの出力に結合された第１の入力を有する乗算論理装置と、この乗算論理装置の出力に結合された第１の入力を有する演算論理装置と、この演算論理装置の出力に結合された入力を有するレジスタ装置と、演算論理装置またはレジスタ装置の出力に結合された入力を備えた反転論理装置と、この反転論理装置の出力と乗算論理装置の第２の入力との間に結合された変換モジュールと、乗算論理装置および演算論理装置に結合されたメモリと、ならびに演算論理装置の出力に結合された出力コンバータとを備えており、反転論理装置は逆数または逆平方根演算を行い、変換モジュールはスカラー頂点データをベクトル頂点データに変換するように構成されており、メモリには頂点データを処理するときに使用される複数の定数および変数が記憶されており、出力コンバータは処理された頂点データを出力するようにライティングモジュールに結合されている、グラフィックパイプラインにおけるグラフィック処理中にブレンディング演算を行うためのシステム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、一般に、グラフィックプロセッサに関し、とくに、グラフィックパイプラインシステムの変換モジュールに関する。
【０００２】
【従来の技術】
従来技術を表す図１は、パイプライン方式のグラフィック処理システムを構成する一般的な従来技術のシステムを示している。このシステムにおいて、データソース10はプリミティブを既定する拡張された頂点のストリームを発生する。これらの頂点は、記憶されるた

めに一時に１つづつ頂点メモリ13によってパイプライングラフィックシステム12に送られる。頂点メモリ13からパイプライングラフィックシステム12中に拡張された頂点が受取られると、頂点は変換モジュール14によって変換され、ライティングモジュール16によって照明され、さらに、ラスター化装置18によるレンダリングのためにクリップされて設定され、それによってレンダリングされたプリミティブを発生し、これが後で表示装置20上に表示される。

【０００３】

動作中、変換モジュール14は、頂点をモデル座標で受取ってこれらの３次元頂点をそれらのモデル座標から、それらが最終的に表示される２次元ウインドウに変換するために使用されることができる。変換を行うために、ビューポート、ビューマトリックス、ワールドマトリックス、投影マトリックス等の標準的な変換パラメータが使用されてもよい。

【０００４】

同時に、上述したパラメータによって、幾何学的変換により、あるオブジェクトに関する別のオブジェクトの位置が表現され、回転され、クリップされ、そのサイズが定められると共に、３次元情景において見ている位置、方向および遠近が変更されることを可能にする。３次元頂点をそれらのモデル座標から、それらが表示される２次元ウインドウに変換する座標変換には、典型的に、移行、回転およびスケーリングの１以上が必要である。

【０００５】

従来技術の変換システムは典型的に、変換プロセス中に別々に発生されたスカラーおよびベクトル値を処理する。たとえば、位置属性、すなわち（Ｘ，Ｙ，Ｚ，Ｗ）は乗算器および、または加算器のようなベクトル演算子により処理され、それによってスカラー値をレンダリングしてもよい。スカラー演算子がこのようなスカラー値を処理してもよいが、それは典型的にベクトル演算子によって再度処理されることはない。グラフィックパイプライン処理中に処理された頂点データのスカラーおよびベクトル形態の処理を統合しようとする試みは現在まで行われていない。

【０００６】

変換モジュール14によって処理されるさらに別のプロセスはブレンディング、すなわち“スキニング”である。スキニングとは、オブジェクト間の結合部をブレンディングすることによって分割された多角形オブジェクトに現実性を加えるプロセスのことである。従来技術の図１Ａは、スキニングが行なわれる前および後の１対のオブジェクト22を示している。

【０００７】

【発明が解決しようとする課題】

通常、スキニングプロセスはコンピュータプログラムおよび汎用プロセッサを使用して行なわれる。したがって、専用回路に関連した利益、すなわち、速度、効率等を得るためにハードウェア上でスキニングを実施しようと試みられたことがない。

【０００８】

【課題を解決するための手段】

変換システムがグラフィック処理するための方法、装置および製品が提供される。システムには入力バッファが含まれており、この入力バッファは頂点データを頂点属性バッファから受取るためにこれに結合されるように構成されている。乗算論理装置は、入力バッファの出力に結合された第１の入力を有している。演算論理装置もまた設けられており、それは乗算論理装置の出力に結合された第１の入力を有している。演算論理装置の出力にはレジスタ装置の入力が結合されている。

【０００９】

反転論理装置もまた設けられており、それは演算論理装置の出力に結合された入力を有し、逆数または逆平方根演算を行う。１実施形態において、変換モジュールの反転論理装置においてヌルW属性値を処理する方法が提供される。W属性がヌルである場合、ゼロによる除算は役に立たない無限大値を発生させるので、ラスター化装置の設定モジュールがスクリーンスペースにおいてエッジ方程式を発生できないため、ヌルW属性値の処理はとく

に重要なものとなる。使用において、頂点データを受取ると、変換モジュールの反転論理装置は頂点データのW属性の値を識別する。識別されたW属性値がヌルである場合、頂点データのW属性に関する除算演算は最小および最大指数にクランプされる。このクランプされた値をラスター化装置の設定モジュールがエッジ方程式を発生するために使用する。

【００１０】

さらに、反転論理装置の出力と乗算論理装置の第２の入力との間に結合された変換モジュールが含まれている。使用において、変換モジュールはスカラー頂点データをベクトル頂点データに変換するように作用する。

【００１１】

メモリは乗算論理装置および演算論理装置に結合される。メモリには、頂点データを処理するために入力バッファ、乗算論理装置、演算論理装置、レジスタ装置、反転論理装置および変換モジュールと共同して使用される複数の定数および変数が記憶されている。最後に、出力コンバータは演算論理装置の出力に結合され、処理された頂点データをライティングモジュールに出力するためにこれに結合されている。

【００１２】

本発明の１つの特徴において、変換システムはスカラーおよびベクトル成分の両方をグラフィック処理中に処理するように構成されることができる。これを行うために、頂点データはベクトルの形態で受取られ、これの後ベクトル演算がベクトル頂点データに関して行われる。演算論理装置および乗算論理装置または任意の別のタイプのベクトル演算モジュールがこのようなベクトル演算を行ってもよい。

【００１３】

次に、ベクトル演算の出力についてスカラー演算が実行され、それによって頂点データをスカラーの形態でレンダリングしてもよい。反転論理装置または任意の別のタイプのスカラー演算モジュールがスカラー演算を行ってもよい。このようなスカラー頂点データは、その後ベクトル演算を行うためにベクトル頂点データに変換されてもよい。ベクトル演算を行うためのレジスタはまたベクトル演算の出力を記憶する。オプションとして、レジスタは、ベクトル演算の出力に基づいてベクトル頂点データを発生させるマスキング機能を備えていてもよい。

【００１４】

本発明のさらに別の特徴において、グラフィックパイプラインにおけるグラフィック処理中にブレンディング、すなわち“スキニング”演算のハードウェア構成を設ける技術が使用されてもよい。パイプラインにおける処理中に、複数のマトリックスおよびそのマトリックスの１つにそれぞれ対応した複数の加重値が受取られる。処理されるべき頂点データもまた受取られる。

【００１５】

その後、頂点データとマトリックスの１つとそのマトリックスに対応した加重とを乗算することによって複数の積の和が計算されることができる。その後、このような積の和は付加的な処理のために出力される。

【００１６】

１実施形態において、マトリックスはモデルビューマトリックスを含んでいてもよく、付加的な処理はライティング動作を含んでいてもよい。この実施形態では、表示するために合成マトリックスもまた積の和と乗算されてもよい。さらに、そのマトリックスは逆マトリックスを含んでいてもよく、頂点データは正規ベクトルを含んでいてもよい。このような場合にもまた、ライティング動作が付加的な処理に含まれてもよい。

【００１７】

【発明の実施の形態】

本発明のこれらおよび別の利点は、以下の詳細な説明を読み、その図面の種々の図を検討することによって明らかになるであろう。

上記およびその他の特徴および利点は、添付図面を参照とする以下の本発明の好ましい実施形態の詳細な説明からさらによく理解されるであろう。

図１および１Ａは、従来技術を示している。図１Ｂ乃至３２Ｃは、本発明のグラフィックパイプラインシステムを示す。

【００１８】

図１Ｂは、本発明の１実施形態の種々のコンポーネントを示すフロー図である。示されているように、本発明は、頂点属性バッファ（ＶＡＢ）50、変換モジュール52、ライティングモジュール54、および設定モジュール57を備えたラスター化モジュール56を含む４つの主要なモジュールに分けられる。１実施形態において、上記の各モジュールは、以下においてさらに詳細に説明するように単一の半導体プラットフォーム上に配置されている。この説明において、単一の半導体プラットフォームとはただ１つの単一の半導体ベースの集積回路またはチップのことである。

【００１９】

ＶＡＢ50は位置、垂線、カラー、テクスチャ座標のような複数の頂点属性状態を収集し、保持するために含まれている。完成された頂点は変換モジュール52によって処理され、その後ライティングモジュール52に送られる。変換モジュール52は、照明を行うライティングモジュール54に対してベクトル発生する。ライティングモジュール54の出力は、プリミティブを設定する設定モジュールに適したスクリーンスペースデータである。その後、ラスター化モジュール56はプリミティブのラスター化を行う。変換およびライティングモジュール52および54は、コマンドが一度スタートされると常に終了されるように、コマンドレベルでのみ機能停止することを認識しなければならない。

【００２０】

１実施形態において、本発明は、オープン・グラフィック・ライブラリ（商標名オープンＧＬ）およびＤ３Ｄ（商標名）変換およびライティングパイプラインを少なくとも部分的に使用するハードウェア構造を含んでいる。オープンＧＬ（商標名）は２－Ｄおよび３－Ｄグラフィックイメージを既定するコンピュータ業界の標準アプリケーションプログラムインターフェース（ＡＰＩ）である。このオープンＧＬ（商標名）により、アプリケーションは任意のオペレーティングシステムにおいて任意のオープンＧＬ（商標名）に従うグラフィックアダプタを使用して同じ効果を生成する。オープンＧＬ（商標名）は、１組のコマンドまたは即時実行機能を特定する。各コマンドは描写アクションを指令するか、あるいは特別な効果を生じさせる。

【００２１】

図２は、本発明の１実施形態によるＶＡＢ50の概略図である。示されているように、ＶＡＢ50はコマンドビット200 を送り、一方において頂点の属性を表すデータビット204 とモードビット202 とを記憶する。使用において、ＶＡＢ50は頂点のデータビット204 を受取り、これを出力する。

【００２２】

ＶＡＢ50は、複数の可能な頂点属性状態をデータビット204 により受取り、記憶するように構成されている。使用において、このようなデータビット204 すなわち頂点データが受取られ、ＶＡＢ50中に記憶された後、頂点データはＶＡＢ50からグラフィック処理モジュール、すなわち変換モジュール52に出力される。さらに、図２Ａを参照としてさらに詳細に後述する他の処理に加えて、頂点データがＶＡＢ50に入力される方法を決定するコマンドビット200 がＶＡＢ50によって送られる。このようなコマンドビット200 はマイクロ制御装置、ＣＰＵ、データソース、またはコマンドビット200 を発生できる任意の他のタイプのソースのようなコマンドビットソースから受取られる。

【００２３】

さらに、モードビット202 が送られ、このモードビット202 は処理動作の複数のモードの状態を示す。したがって、モードビット202 は、後続するグラフィック処理モジュール中で頂点データが処理される方法を決定するように適応されている。このようなモードビット202 は、マイクロ制御装置、ＣＰＵ、データソース、またはモードビット202 を発生することのできる任意の他のタイプのソースのようなコマンドビットソースから受取られる。

【００２４】
ＶＡＢ50に関連した種々の機能が専用ハードウェア、ソフトウェアまたは任意の他のタイプの論理装置によって制御されてもよい。種々の実施形態において、モードビット202 の６４、１２８、２５６または任意の他の数が使用されてもよい。
【００２５】
ＶＡＢ50はまた１２８ビットフォーマットに変換される必要のある６４ビットデータのための収集地点として機能する。ＶＡＢ50の入力は６４ビット／サイクルであり、その出力は１２８ビット／サイクルである。別の実施形態において、ＶＡＢ50は１２８ビットビットデータのための収集地点として機能してもよく、また、ＶＡＢ50の入力は１２８ビット／サイクルまたは任意の他の組合せであってよい。さらにＶＡＢ50は複数の頂点属性のために確保されたスロットを有しており、それらは全てＩＥＥＥ３２ビットフロートである。このようなスロットの数はユーザの所望に応じて異なってよい。表１は、本発明によって使用される例示的な頂点属性を示している。
【００２６】
表１
位置：ｘ，ｙ，ｚ，ｗ
拡散カラー：ｒ，ｇ，ｂ，ａ
反射カラー：ｒ，ｇ，ｂ
フォグ：ｆ
テクスチャ０：ｓ，ｔ，ｒ，ｑ
テクスチャ１：ｓ，ｔ，ｒ，ｑ
垂線：ｎｘ，ｎｙ，ｎｚ
スキンウエイト：ｗ
【００２７】
動作中、ＶＡＢ50は、ｘ，ｙの書込み時にｚ，ｗ対を（０．０，１．０）にデフォルトすることが可能となるため、ｚ，ｗデータ対の前にｘ，ｙデータ対が書込まれるものとして動作する。これはオープンＧＬ（商標名）およびＤ３Ｄ（商標名）におけるデフォルト成分にとって重要である可能性がある。位置、テクスチャ０およびテクスチャ１のスロットは第３および第４の成分を（０．０，１．０）にデフォルトすることを認識しなければならない。さらに、拡散カラースロットは第４の成分を（１．０）にデフォルトし、テクスチャスロットは第２の成分を（０．０）にデフォルトする。
【００２８】
ＶＡＢ50は、データビット204 をアセンブルするために使用されるさらに別のスロット205 を含んでおり、このデータビット204 は変換モジュール54に送られることができ、あるいはライティングモジュール54から受取られることができる。スロット205 におけるデータビット204 は浮動小数点または整数フォーマットであることができる。上述したように、各頂点のデータビット204 は、データビット204 の処理に影響を与えるモードを表すモードビット202 の関連したセットを有している。これらのモードビット202 は、以下さらに詳細に説明する理由のためにデータビット204 により変換およびライティングモジュール52および54を通って送られる。
【００２９】
１実施形態において、ＶＡＢ50によって受取られる１８の有効なＶＡＢ、変換およびライティングコマンドが存在することができる。図２Ａは、本発明の１実施形態によるＶＡＢ50によって受取られることのできる種々のコマンドを示すチャートである。図２Ａのチャートに示されている全てのロードおよび読出し文脈コマンドならびにパススルーコマンドは、１２８ビットまでの、またはその他のサイズの１つのデータワードを転送することを認識しなければならない。
【００３０】
図２Ａの各コマンドは、データビット204 の各セットが１つのＶＡＢアドレスの高ダブルワードまたは低ダブルワード中に書込まれるか否かを指示する制御情報を含んでいる可能

性がある。さらに、ワードレベルの制御を行う２ビットの書込みマスクが使用されてもよい。さらに、実行されるべき現在のコマンドに対するデータビット204 の全てが存在していることを開始（ｌａｕｎｃｈ）ビットがＶＡＢ制御装置に通知してもよい。

【００３１】

各コマンドは関連した機能停止フィールドを有しており、このフィールドによってルックアップは、そのコマンドが文脈メモリを読出す読出しコマンドであるか、あるいは文脈メモリに書込む書込みコマンドであるかに関する情報を見出すことが可能になる。現在実行中のコマンドの機能停止フィールドを使用することによって、新しいコマンドは、矛盾の場合には待機させられ、あるいは進行することを可能にされることができる。

【００３２】

動作において、ＶＡＢ50はサイクル当り１２８ビット（または他の任意のサイズ）までの１つの入力データワードを受取り、サイクル当り１２８ビット（または他の任意のサイズ）までの１つのデータワードを出力することができる。ロードコマンドに対して、これは、データをＶＡＢ50中にロードして１２８ビットのクワド（ｑｕａｄ）ワードを生成するのに２サイクル必要であり、それをドレインするために１サイクル必要であることを意味する。ライティングモジュール54内のスカラーメモリに対して、全クワドワードを累算する必要はなく、これらは１サイクル／アドレスでロードされることができる。１つの頂点に対して、７つのＶＡＢスロットをロードするのに１４サイクルまで必要になる可能性があり、一方それらをドレインするには７サイクルあればよい。しかしながら、実行中の頂点コマンドを変更する頂点状態を更新するだけでよいことを認識すべきである。これは、ある場合には頂点位置が２サイクルで更新され、一方頂点データをドレインするのに７サイクルを要することを意味する。ｘ，ｙ位置の場合には、たった１サイクルあればよいことを認識しなければならない。

【００３３】

図２Ｂは、グラフィック処理中に頂点属性をＶＡＢ50にロードし、ＶＡＢ50からドレインする１つの方法を示すフローチャートである。最初に、動作210 で、ＶＡＢ50において頂点属性の少なくとも１つのセットが処理されるために受取られる。上述したように、頂点属性の各セットは特有で、単一の頂点に対応していることができる。

【００３４】

使用において、頂点属性は動作212 でＶＡＢ50が受取ったときにその中に記憶される。さらに、記憶された頂点属性の各セットは変換モジュール52の複数の入力バッファの対応した１つに転送される。受取られた頂点属性セットはまた、動作216 で示されているように、受取られた頂点属性が現在ＶＡＢ50中に記憶されている異なったセットの対応した頂点属性を有しているか否かを決定するために監視される。

【００３５】

決定ブロック217 において記憶されている頂点属性が受取られた頂点属性に対応していると決定されたとき、動作218 に示されているように、記憶されている頂点属性は変換モジュール52の対応した入力バッファにばらばらな順序で出力される。記憶されている頂点属性が出力されるとすぐに、対応した入ってきた頂点属性がＶＡＢ50中にその場所を占有してもよい。しかしながら、対応が全く見出されない場合、動作219 に示されているように、記憶されている頂点属性の各セットは規則的な予め定められたシーケンスにしたがって変換モジュール52の対応した入力バッファに転送されてもよい。

【００３６】

記憶されている頂点属性は、それが関連した開始コマンドを有している場合には、上述した方式で転送されない可能性があることを注意すべきである。さらに、上記の方法が適切に行われるためにＶＡＢ50の出力の帯域幅は少なくともＶＡＢ50の入力の帯域幅でなければならない。

【００３７】

図２Ｃは、図２Ｂの動作を実行するために使用される本発明のアーキテクチャを示す概略図である。示されているように、ＶＡＢ50は書込みデータ端子ＷＤと、読出しデータ端子

ＲＤと、書込みアドレス端子ＷＡと、および読出しアドレス端子ＲＡとを有している。読出しデータ端子は第１のクロック制御されたバッファ230 に結合され、データビット204をＶＢＡ50から出力する。

【００３８】

第１のマルチプレクサ232 もまた含まれており、これはＶＡＢ50の読出しアドレス端子と第２のクロック制御されたバッファ234 とに結合された出力を有している。第１のマルチプレクサ232 の第１の入力はＶＡＢ50の書込みアドレス端子に結合され、一方第１のマルチプレクサ232 の第２の入力は第２のマルチプレクサ236 の出力に結合されている。論理モジュール238 は第１および第２のマルチプレクサ232 および236 と、ＶＡＢ50の書込みアドレス端子と、第２のクロック制御されたバッファ234 の出力との間に結合されている。

【００３９】

使用において、論理モジュール238 は、入ってきた頂点属性が未決定でありＶＡＢ50においてドレインするか否かを決定する作用を行う。１実施形態において、この決定は、頂点属性が未決定であるか否かを示すビットレジスタを監視することより容易に行われることができる。入ってきた頂点属性がこの時点でＶＡＢ50中に一致したものを有していると決定された場合、論理モジュール238 は、入ってきた頂点属性がすぐにその場所に記憶されるようにその一致した頂点属性をドレインするために第１のマルチプレクサ232 を制御する。他方、入ってきた頂点属性がこの時点でＶＡＢ50中に一致したものを有しないと決定された場合、論理モジュール238 は、ＶＡＢ50がドレインされ、入ってきた頂点属性が、論理モジュール238 によって更新される第２のマルチプレクサ236 の入力によって順次または別のある予め定められた順序でロードされるように、第１のマルチプレクサ232 を制御する。

【００４０】

その結果、ＶＡＢ50は、新しい入ってきた頂点属性がロードされる前に、多数の頂点属性をドレインする必要がない。未決定の頂点属性は、可能ならば、対応したＶＡＢの対応したもの押し出し、それによってそれが進行することを可能にする。その結果、ＶＡＢ50は任意の順序でドレインすることができる。この能力がないと、ＶＡＢ50をドレインするのに７サイクルを要し、それをロードするのに、おそらく、さらに１４サイクル要する。ロードとドレインとを重複させることにより、さらに高い性能が得られる。これは、入力バッファが空であり、ＶＡＢ50が変換モジュール52の入力バッファ中にドレインできる場合にのみ可能であることを認識しなければならない。

【００４１】

図３は、本発明の１実施形態によるＶＡＢ50に関連したモードビットを示している。変換／ライティングモード情報は、モードビット202 によりレジスタ中に記憶される。モードビット202 は、以下において明らかになるように、変換モジュール52およびライティングモジュール54のシーケンサを駆動するために使用される。各頂点は特有であることが可能である関連したモードビット202 を有しており、したがって特有に構成されたプログラムシーケンスを実行することができる。モードビット202 は一般にグラフィックＡＰＩに直接マップすることができるが、それらのあるものは導出されてもよい。

【００４２】

１実施形態において、図３のアクティブな光ビット（ＬＩＳ）は隣接していてもよい。さらに、パススルービット（ＶＰＡＳ）は、オンにされたときに頂点データがスケールおよびバイアスと共に通過され、変換もライティングも行われないという点で特有である。ＶＰＡＳが真（ｔｒｕｅ）である場合に使用される可能なモードビット202 はテクスチャ分割ビット（ＴＤＶ０，１）およびフォグビット（商標名Ｄ３Ｄにおいてフォグ値を抽出するために使用される）である。したがって、ＶＰＡＳは予め変換されたデータに対して使用され、ＴＤＶ０，１は商標名Ｄ３Ｄの文脈では円筒状ラップモードを処理するために使用される。

【００４３】

図４は、本発明の１実施形態の変換モジュールを示している。示されているように、変換モジュール52は６つの入力バッファ400 によってＶＡＢ50に接続されている。１実施形態において、各入力バッファ400 はサイズが７＊１２８ビットである。６つの入力バッファ400 はそれぞれ７つのクワドワードを記憶することができる。そのような各入力バッファ400 は、パスデータが位置データと重複されていることを除いて、そのレイアウトがＶＡＢ50と同じである。

【００４４】

１実施形態において、入力バッファ400 がロードされた前の段階からデータが変化しているか否かを示すように各入力バッファ400 の各属性に対して１つのビットが指定されていてもよい。この設計によって、各入力バッファ400 は変化したデータに関してのみロードされてもよい。

【００４５】

変換モジュール52はさらに、ライティングモジュール54中の６つの出力頂点バッファ402に接続されている。出力バッファは第１のバッファ404 と、第２のバッファ406 と、および第３のバッファ408 とを含んでいる。以下から明らかになるように、第３のバッファ408 の内容、すなわち位置、テクスチャ座標データ等はライティングモジュール54では使用されない。しかしながら、第１のバッファ404 および第２のバッファ406 の両者は光線およびカラーデータをライティングモジュール54に入力するために使用される。ライティングモジュールは２つの読出し入力を処理するように構成されているため、２つのバッファが使用される。データは読出しコンフリクト等に関する問題を回避するように構成されていることを認識すべきである。

【００４６】

さらに、変換モジュール52には文脈メモリ410 およびマイクロコードＲＯＭメモリ412 が結合されている。変換モジュール52はオブジェクト空間頂点データをスクリーン空間に変換して、ライティングモジュール54が必要とするベクトルを生じさせる作用をする。変換モジュール52はまたスキニング（ｓｋｉｎｎｉｎｇ）およびテクスチャ座標を処理する。１実施形態において、変換モジュール52は平行して４つのフロートを処理する１２８ビット設計であってもよく、４項ドット積を行うために最適化されていてもよい。

【００４７】

図４Ａは、変換モジュール52中で多数のスレッドを実行する本発明の１実施形態による方法を示すフローチャートである。動作において、変換モジュール52はインターリーブすることにより３つの頂点を平行して処理することができる。したがって、書込みおよびそれに後いて文脈メモリ410 からの読出しのようなコマンドの間に機能停止状況が生じなければ、３つのコマンドが同時に平行して実行されることができる。３つの実行スレッドは互いに独立しており、任意コマンドであることができる。これは、全ての頂点が特有の対応したモードビット202 を含んでいるためである。

【００４８】

図４Ａに示されているように、多数のスレッドを実行する方法は、動作420 において実行されるべき現在のスレッドを決定することを含んでいる。この決定は、グラフィック処理モジュールがある動作の終了に必要とするサイクルの数を識別して、そのサイクルを追跡することにより行われることができる。サイクルを追跡することによって、各スレッドはあるサイクルに割当てられ、それによって現在のサイクルに基づいて現在のスレッドを決定することができる。しかしながら、このような決定は、効果的であると思われる任意の所望の方式で行われてもよいことを認識しなければならない。

【００４９】

次に動作422 において、現在のサイクル中に実行されるべきスレッドに関連した命令が、対応したプログラムカウンタ番号を使用して検索される。その後、この命令は動作424 においてグラフィック処理モジュールに関して実行される。

【００５０】

１使用例において、この方法は、最初に、第１のプログラムカウンタによって第１の命令

またはコードセグメントにアクセスすることを含む。上述のように、このようなプログラムカウンタは第１の実行スレッドと関連している。次に、第１のコードセグメントがグラフィック処理モジュールにおいて実行される。すぐに明らかになるように、このようなグラフィック処理モジュールは加算器、乗算器または任意の他の機能装置あるいはその組合せの形態をとることができる。

【００５１】

グラフィック処理モジュールは実行を完了するために２以上のクロックサイクルを必要とするため、第１のコードセグメントの実行後１クロックサイクル経過してすぐに第２のコードセグメントが第２のプログラムカウンタによってアクセスされてもよい。第２のプログラムカウンタは第２の実行スレッドと関連しており、各実行スレッドが特有の頂点を処理する。

【００５２】

その後、第２のコードセグメントは、グラフィック処理モジュール中での第１のコードセグメントの実行の終了前に、グラフィック処理モジュールにおいて実行を開始してもよい。使用において、グラフィック処理モジュールは出力を発生するために全てのスレッドのそれぞれに対して予め定められた数のサイクルを必要とする。したがって、全ての予め定められた数のサイクルのそれぞれに対してこの例の種々のステップが反復されてもよい。

【００５３】

この技術は、従来技術より優れた多くの利点を提供する。もちろん、本発明の機能装置はさらに効率的に使用される。さらに、多数スレッド方式が使用されると仮定された場合、統御コードがより効率的に書込まれることができる。

【００５４】

たとえば、答えを出力するのに３クロックサイクルを必要とする乗算器がグラフィック処理モジュールに含まれている場合、ａ＝ｂ＊ｃおよびｄ＝ｅ＊ａのような後続する動作の間に２つの無動作コマンドを含むことが必要となる。その理由は、３クロックサイクル後まで“ａ”が利用できないためである。しかしながら、この実施形態では、コードがａ＝ｂ＊ｃの直後にｄ＝ｅ＊ａを呼出すだけでもよい。それは、このようなコードは３クロックサイクルごとに１回呼出される３つの実行スレッドの１つとして実行されると仮定されることができるからである。

【００５５】

図４Ｂは、図４Ａの方法が行われる手順を示すフロー図である。示されているように、各実行スレッドは、命令メモリ452 において命令またはコードセグメントにアクセスするために使用される関連したプログラムカウンタ450 を有している。その後、このような命令は加算器456 、乗算器454 および、または反転論理装置またはレジスタ459 のようなグラフィック処理モジュールを動作するために使用されてもよい。

【００５６】

上記の処理モジュールの２以上のものがタンデム方式で使用される状況に適応させるために、グラフィック処理モジュール間において１以上のコードセグメント遅延素子457 が使用される。３スレッドフレームワークが使用される場合、３クロックサイクルコードセグメント遅延素子457 が使用される。１実施形態において、加算命令が乗算命令に後続した場合に、コードセグメント遅延素子457 が使用される。このような場合、乗算器456 が出力を発生するために十分な時間が確実に経過するように、乗算命令の実行後３クロックサイクル経過するまで加算命令は実行されない。

【００５７】

各命令の実行後、現在の実行スレッドのプログラムカウンタ450 が更新され、次の実行スレッドのプログラムカウンタが関連した命令にアクセスするためにラウンドロビンシーケンスでモジュール458 により呼出される。プログラムカウンタは、インクリメント、ジャンプ、呼出および復帰、テーブルジャンプ、および、またはディスパッチを含む任意の方式で使用されることができるが、それに限定されないことを認識しなければならない。ディスパッチとは、受取られたパラメータに基づいてコードセグメント実行の開始地点を決

定することである。さらに、この多数スレッド実行フレームワークに関連した原理は、本発明のグラフィック処理パイプラインのライティングモジュール54にも適用可能であることを認識することが重要である。

【００５８】

３スレッドフレームワークが使用される場合、任意のある時間に各スレッドが１つの入力バッファおよび１つの出力バッファに割当てられる。これによって、３つのコマンドを処理しながら、さらに３つのコマンドをデータと共にロードすることが可能になる。入力バッファおよび出力バッファは、以下において図２７および２８を参照として説明する方式によりラウンドロビンシーケンスで割当てられる。

【００５９】

したがって、実行スレッドは時間的および機能的にインターリーブされる。これは、各機能装置が３つのステージにパイプラインされ、各スレッドがいつでも１つのステージを占有していることを意味する。１実施形態において、３つのスレッドは常に同じシーケンスで実行するように、すなわち０、１、３に設定されてもよい。概念上、スレッドはｔ＝クロックモジュロ３において機能装置に入力される。機能装置が動作し始めると、結果を出力するのに３サイクルを要し（６サイクルを必要とするＩＬＵを除いて）、その時同じスレッドは再びアクティブである。

【００６０】

図５は、本発明の１実施形態による図４の変換モジュール52の機能装置を示している。示されているように、頂点データを受取るためにＶＡＢ50に結合するように構成された入力バッファ400 が含まれている。

【００６１】

メモリ論理装置（ＭＬＵ）500 は、入力バッファ400 の出力に結合された第１の入力を有している。オプションとして、ＭＬＵ500 の出力は、その第１の入力に結合されたフィードバックループ502 を有していてもよい。

【００６２】

演算論理装置（ＡＬＵ）504 もまた設けられており、このＡＬＵの第１の入力はＭＬＵ500 の出力に結合されている。ＡＬＵ504 の出力はさらに、その第２の入力に接続されたフィードバックループ506 を有している。このようなフィードバックループ502 はさらに、それに結合された遅延素子508 を有していてもよい。ＡＬＵ504 の出力には、レジスタ装置510 の入力が結合されている。レジスタ装置510 の出力は、ＭＬＵ500 の第１および第２の入力に結合されていることを認識しなければならない。

【００６３】

反転論理装置（ＩＬＵ）512 が設けられており、このＩＬＵ512 は、逆数または逆平方根演算を行うためにＡＬＵ504 の出力に結合された入力を含んでいる。別の実施形態において、ＩＬＵ512 はレジスタ装置510 の出力に結合された入力を含んでいてもよい。

【００６４】

さらに、変換またはスメアリング（ｓｍｅａｒｉｎｇ）モジュール514 が含まれており、このモジュール514 はＩＬＵ512 の出力とＭＬＵ500 の第２の入力との間に結合されている。使用において、この変換モジュール514 はスカラー頂点データをベクトル頂点データに変換するように機能する。これはスカラーデータをベクトルと乗算して、乗算器および、または加算器が処理するベクトル演算子にすることによって行なわれる。たとえば、スカラーＡは、変換後、ベクトル（Ａ，Ａ，Ａ，Ａ）になってもよい。別の実施形態では、スメアリングモジュール514 はＭＬＵ500 と関連したマルチプレクサまたは本発明の任意の他のコンポーネント中に含まれていてもよい。オプションとして、レジスタ516 はＩＬＵ512 の出力と変換装置514 の入力との間に結合されていてもよい。さらに、このようなレジスタ516 はスレッド（ｔｈｒｅａｄ）されてもよい。

【００６５】

メモリ410 は、ＭＬＵ500 の第２の入力とＡＬＵ504 の出力とに結合されている。とくに、メモリ410 はＭＬＵ500 の第２の入力に結合された読出し端子を有している。さらに、

メモリ410 はＡＬＵ504 の出力に結合された書込み端子を有している。

【００６６】

メモリ410 は、頂点データを処理するために入力バッファ400 、ＭＬＵ500 、ＡＬＵ504 、レジスタ装置510 、ＩＬＵ512 および変換モジュール514 と共に使用されるために複数の定数および変数が記憶されている。このような処理には、オブジェクト空間頂点データをスクリーン空間頂点データに変換し、ベクトルを発生すること等が含まれる。

【００６７】

最後に、出力コンバータ518 はＡＬＵ504 の出力に結合されている。出力コンバータ518 は、処理された頂点データがこれに出力されるように出力バッファを介してライティングモジュール54に結合されている。ＩＬＵを除く全てのデータ通路は１２８ビット幅であるように設計されてもよく、あるいは別のデータ通路幅が使用されてもよい。

【００６８】

図６は、本発明の１実施形態による図５の変換モジュール52のＭＬＵ500 の概略図である。示されているように、変換モジュール52のＭＬＵ500 は、並列に結合された４つの乗算器600 を含んでいる。

【００６９】

変換モジュール52のＭＬＵ500 は、３つの異なった方式で２つの４成分ベクトルを乗算するか、あるいは１つの４成分ベクトルをパス（ｐａｓｓ）することができる。ＭＬＵ500 は、多重演算を行うことができる。表２は、変換モジュール52のＭＬＵ500 に関連したこのような演算を示している。

【００７０】

表２

【数１】

```
CMLU_MULT  o[0] = a[0]*b[0],o[1] = a[1]*b[1],o[2] = a[2]*b[2],o[3] = a[3]*b[3]
CMLU_MULA  o[0] = a[0]*b[0],o[1] = a[1]*b[1],o[2] = a[2]*b[2],o[3] = a[3]
CMLU_MULB  o[0] = a[0]*b[0],o[1] = a[1]*b[1],o[2] = a[2]*b[2],o[3] = b[3]
CMLU_PASA  o[0] = a[0],o[1] = a[1],o[2] = a[2],o[3] = a[3]
CMLU_PASB  o[0] = b[0],o[1] = b[1],o[2] = b[2],o[3] = b[3]
```

【００７１】

表３には、可能なＡおよびＢ入力が示されている。

【００７２】

表３

ＭＡ＿Ｍ	ＭＬＵ
ＭＡ＿Ｖ	入力バッファ
ＭＡ＿Ｒ	ＲＬＵ（ＭＢ＿Ｒと共有された）
ＭＢ＿Ｉ	ＩＬＵ
ＭＢ＿Ｃ	文脈メモリ
ＭＢ＿Ｒ	ＲＬＵ（ＭＡ＿Ｒと共有された）

【００７３】

表４は、クロス乗積に対して使用されることのできる回転オプションを示している。

【００７４】

表４

ＭＲ＿ＮＯＮＥ	変更なし
ＭＲ＿ＡＬＢＲ	A[XYZ]ベクトルを左に、B[XYZ]ベクトルを右に回転する
ＭＲ＿ＡＲＢＬ	A[XYZ]ベクトルを右に、B[XYZ]ベクトルを左に回転する

【００７５】

図７は、本発明の１実施形態による図５の変換モジュール52のＡＬＵ504 の概略図である。示されているように、変換モジュール52のＡＬＵ504 は、並列／直列に結合された３つの加算器700 を含んでいる。使用において、変換モジュール52のＡＬＵ504 は２つの３成

分ベクトルを加算し、１つの４成分ベクトルをパスし、あるいはベクトル成分を出力を横切ってスメア（ｓｍｅａｒ）することができる。表５は、変換モジュール52のＡＬＵ504が行うことのできる種々の演算を示している。

【００７６】

表５

【数２】

CALU_ADDA o[0] = a[0]+b[0],o[1] = a[1]+b[1],o[2] = a[2]+b[2],o[3] = a[3]
CALU_ADDB o[0] = a[0]+b[0],o[1] = a[1]+b[1],o[2] = a[2]+b[2],o[3] = b[3]
CALU_SUM3B o[0123] = b[0] + b[1] + b[2]
CALU_SUM4B o[0123] = b[0] + b[1] + b[2] + b[3]
CALU_SMRB0 o[0123] = b[0]
CALU_SMRB1 o[0123] = b[1]
CALU_SMRB2 o[0123] = b[2]
CALU_SMRB3 o[0123] = b[3]
CALU_PASA o[0] = a[0],o[1] = a[1],o[2] = a[2],o[3] = a[3]
CALU_PASB o[0] = b[0],o[1] = b[1],o[2] = b[2],o[3] = b[3]

【００７７】

表６は、変換モジュール52のＡＬＵ504 のＡおよびＢ入力を示している。

【００７８】

表６

ＡＡ＿Ａ	ＡＬＵ（１つの命令遅延）
ＡＡ＿Ｃ	文脈メモリ
ＡＢ＿Ｍ	ＭＬＵ

無変更、Ｂの否定、Ａの否定を行なうことによりＡおよびＢ入力の符号ビットを修正することもまた可能であり、ここでＡ，Ｂは絶対値である。ＡＬＵ504 がスカラー頂点データを出力した場合、このスカラー頂点データは、各出力がスカラー頂点データを表しているという意味で出力を横切ってスメアされていることを認識しなければならない。ＭＬＵ500 およびＡＬＵ504 のパス制御信号のそれぞれが演算中全ての特殊値処理をディスエーブルすることができる。

【００７９】

図８は、本発明の１実施形態による図５の変換モジュール52のベクトルレジスタファイル510 の概略図である。示されているように、ベクトルレジスタファイル510 は４組のレジスタ800 を含んでおり、各レジスタ800 は対応したマルチプレクサ802 の第１の入力に結合された出力と、対応したマルチプレクサ802 の第２の入力に結合された入力とを有している。

【００８０】

本発明の１実施形態において、ベクトルレジスタファイル510 はスレッドされている。すなわち、ベクトルレジスタファイル510 の３つのコピーが存在し、各スレッドがそれ自身のコピーを有している。１実施形態では、各コピーは８つのレジスタを含んでおり、その各レジスタはサイズが１２８ビットであり、４つのフロートを記憶することができる。ベクトルレジスタファイル510 はＡＬＵ504 から書込まれ、その出力はＭＬＵ500 にフィードバックされる。ベクトルレジスタファイル510 はサイクル当り１回の書込みおよび１回の読出しを行なう。

【００８１】

動作において、各レジスタコンポーネントへの書込み動作を個々にマスクすることもできる。ベクトルレジスタファイル510 は、書込みアドレスが読出しアドレスと同じである場合、入力から出力へのバイパス路511 によってゼロレイテンシーを示す。この場合、マスクされていないコンポーネントはレジスタから取出され、マスクされたコンポーネントはバイパスされる。このように、ベクトルレジスタファイル510 はコンポーネント単位でベ

クトルを生成し、あるいはＡＬＵ　ＳＭＲ演算（表５参照）と共にベクトル成分の順序を変更することに対して非常に有用である。一時的な結果はまたベクトルレジスタファイル510 中に記憶されることができる。

【００８２】

図９は、本発明の１実施形態による図５の変換モジュール52のＩＬＵ512 の概略図である。示されているように、変換モジュール52のＩＬＵ512 は浮動小数点の逆数（１／Ｄ）および逆平方根（１／Ｄ＾（１／２））を発生することができる。このような演算を行なうために、２つの反復処理のいずれか一方が小数部に関して実行されてもよい。このような処理は任意の所望の専用ハードウェアにより実行されてもよく、以下に示されている：

	逆数（1／D）	逆平方根（1／D＾（1／2））
	$x_{n+1}=x_n(2-x_n*D)$	$x_{n+1}=(1/2)*x_n(3-x_n^2*D)$
(1)	x_n（速度）に対する表検索	x_n（速度）に対する表検索
	x_n	x_n*x_n
(2)	第１回目の反復：乗算－加算	第１回目の反復：乗算－加算
	$2-x_n*D$	$3-x_n^2*D$
(3)	第１回目の反復：乗算	第１回目の反復：乗算
	$x_n(2-x_n*D)$	$(1/2)*x_n(3-x_n^2*D)$
(4)	第２回目の反復：演算なし	第２回目の反復：２乗
	x_n ＋１をパス	x_{n+1}^2
(5)	第２回目の反復：乗算－加算	第２回目の反復：乗算－加算
	$2-x_{n+1}*D$	$3-x_{n+1}^2*D$
(6)	第２回目の反復：乗算	第２回目の反復：乗算
	$x_{n+1}(2-x_{n+1}*D)$	$(1/2)*x_{n+1}(3-x_{n+1}^2*D)$

示されているように、２つの処理は類似しており、簡単な設計を行なっても差しつかえない。この反復は、しきい値精度が満足されるまで繰り返されることを認識しなければならない。

【００８３】

動作において、ＩＬＵ512 は逆数演算および逆平方根演算を含む２つの基本的な演算を行なう。他の装置とは異なり、それは出力を発生するために６サイクルを必要とする。その入力はスカラーであり、したがって出力もそうである。前述したように、ＩＬＵ512 の出力におけるスレッド保持レジスタ516 は、有効な結果が発生される次の回まで結果をラッチするように当てにされている。さらに、スカラー出力は、ＭＬＵ500 に供給される前にベクトルにスメアされる。反転装置512 は、約２２小数部ビット範囲内までの正確なＩＥＥＥ（米国電気電子技術者協会）出力を発生するために検索表および２つのパスＮｅｗｔｏｎ－Ｒａｐｈｓｏｎ反復を使用する。表７は、変換モジュール52のＩＬＵ512 によって行なわれることのできる種々の演算を示している。

【００８４】

表７

ＣＩＬＵ＿ＩＮＶ	ｏ＝１．０／ａ
ＣＩＬＵ＿ＩＳＱ	ｏ＝１．０／ｓｑｒｔ（ａ）
ＣＩＬＵ＿ＣＩＮＶ	ｏ＝１．０／ａ（レンジクランプにより）
ＣＩＬＵ＿ＮＯＰ	出力なし

表７の上述したレンジクランプ反転演算は、クリッピング演算がラスター化モジュール56

により処理されることを可能にするために使用されてもよい。座標はスクリーン空間に直接変換され、これは、均質のクリップスペースがほぼ０．０である場合に問題を結果的に生じさせる可能性が高い。各除算において１．０／０．０による乗算を回避するために、１／ｗ計算が最小および最大ベキ指数にクランプされる。

【００８５】

使用において、図５に示されている文脈メモリ410 は、クワドワードだけを使用して読出しおよび書込みを行なう。このメモリはＭＬＵ500 またはＡＬＵ504 によって各サイクルごとに読出されることができ、ＡＬＵ504 によって書込まれることができる。メモリ読出しはサイクル当り１度だけ可能である。読出しが必要である場合には、それは命令の開始時に行なわれ、それから３サイクル後にＡＬＵ504 にパイプラインされる。文脈メモリ410 は必ずしもスレッドされなくてよい。

【００８６】

図１０は、本発明の１実施形態による図５の変換モジュール52の出力コンバータ518 の出力アドレスのチャートである。出力コンバータ518 は出力を適切な目的地に導き、データのビット精度を変更し、性能を増加させるためにあるデータ撹拌（ｓｗｉｚｚｌｉｎｇ）を行なうことができる。ライティングモジュール54に送られる予定である全てのデータは、Ｓ１Ｅ８Ｍ１３として編成された２２ビット浮動小数点フォーマット（１符号、８ベキ指数、１３小数部ビット）に丸められる。ライティングモジュール54における図４に示されているような目的地バッファ402 はスレッドされる。

【００８７】

データ撹拌は、ベクトルを発生しているときに有用である。このような技術により、ベクトルを生成する場合に損失を生じずに距離ベクトル（１，ｄ，ｄ＊ｄ）を発生することが可能となる。距離ベクトルはフォグ、地点パラメータおよび照明減衰に対して使用される。これは、アイベクトルおよび照明方向ベクトルにより行なわれる。表８は、このようなベクトルに関連した種々の演算を示している。以下の表において、ベクトルを２乗するとはｄ^2 ＝ｄｏｔ［（ｘ，ｙ，ｚ），（ｘ，ｙ，ｚ）］であるｄ^2 を（ｘ，ｙ，ｚ）のｗコンポーネント中に記憶することを指していることを認識しなければならない。

【００８８】

表８

（１）ベクトルを２乗する（ｘ，ｙ，ｚ，ｄ＊ｄ）（ｄ＊ｄをＶＢＵＦに出力し、１．０をＶＢＵＦに出力する）

（２）ｄ＊ｄの逆平方根を発生する（１／ｄ）

（３）ベクトルを正規化する（ｘ／ｄ，ｙ／ｄ，ｚ／ｄ，ｄ）（ｘ／ｄ，ｙ／ｄ，ｚ／ｄをＷＢＵＦに出力し、ｄをＶＢＵＦに出力する）

本発明において行なわれた数学的計算は常にＩＥＥＥ方式に従ったものである必要はないことを認識しなければならない。たとえば、任意の数により乗算された“０”は“０”をレンダリングすると仮定されることができる。これは、ｄ＝０であるｄ＝ｄ^2 ＊１／（ｄ^2 ）$^{1/2}$ のような式を処理する場合にとくに有用である。上記の仮定を行わないと、このような式はエラーを生じ、したがって関連した計算を行なうときに問題が発生する。

【００８９】

図１１は、本発明の１実施形態による図５の変換モジュール52のマイクロコード編成を示す図である。変換モジュールのマイクロコードは、４４ビットの総帯域幅を形成する１５のフィールドに構成されてもよい。フィールドは、装置のデータフローを一致させるために遅延されてもよい。ＭＬＵ500 の演算はゼロの遅延で実行される。ＡＬＵ演算は１の遅延で実行され、ＲＬＵの出力演算は２の遅延で実行される。各遅延は３サイクルと等価である。

【００９０】

図１２は、本発明の１実施形態による図５の変換モジュール52のシーケンサ1200の概略図である。図１２に示されているように、変換モジュール52のシーケンサ1200は、処理動作の複数のモードの状態を示すモードビットをＶＡＢ50から受取るように構成されたバッフ

ァ1202を含んでいる。

【００９１】

メモリ412 もまた含まれており、このメモリ412 は、モードの状態にしたがって処理動作を行なうようにそれぞれ構成されたコードセグメントを記憶することができる。シーケンシングモジュール1206はメモリ412 と制御ベクトルモジュール1205との間に結合されており、この制御ベクトルモジュール1205はバッファ1202に結合され、モードビット202 から得られた制御ベクトルに基づいてメモリ412 中の複数のアドレスを識別する。シーケンシングモジュール1206はさらに、データを出力バッファ1207に転送するように変換モジュール52を動作するために使用されることのできるコードセグメントを検索するためにメモリ412 中のアドレスにアクセスするように構成されている。

【００９２】

図１３は、図１２の変換モジュール52のシーケンサ1200の使用に関連した種々の動作を詳細に示すフローチャートである。示されているように、シーケンサ1200は、変換またはライティング動作におけるグラフィック処理をシーケンス化するように構成されている。動作1320において、処理動作の複数のモードの状態を示すモードビット202 が最初に受取られる。１実施形態において、モードビット202 はソフトウェア駆動装置から受取られてもよい。

【００９３】

その後、動作1322において、メモリ中の複数のアドレスがモードビット202 に基づいて識別される。その後、動作1324において、そのモードの状態にしたがって処理動作を行なうようにそれぞれ構成されたコードセグメントを検索するために、メモリ中のこのようなアドレスがアクセスされる。続いて、動作1326に示されているように、頂点データを処理するために変換またはライティングモジュールによりコードセグメントが実行される。

【００９４】

図１４は、図１２の変換モジュール52のシーケンサ1200のシーケンシングモジュール1206の動作を詳細に示すフロー図である。示されているように、複数のモードレジスタ1430はそれぞれ、単一の頂点に対応するモードビット202 の特有のセットを含んでいる。モードレジスタ1430は、図４Ａおよび４Ｂを参照として上述した方式での多数の実行スレッドの実行を可能にするためにラウンドロビンシーケンスでポールされることを認識すべきである。

【００９５】

現在の実行スレッドが選択されると、モードビット202 の対応したグループは動作1432でデコードされる。動作1432においてモードビット202 がデコードされると、対応した頂点データを処理する特定のコードセグメントがＲＯＭ1404においてアクセスされたか否かをそれぞれ示す複数のビットを含む制御ベクトルが供給される。

【００９６】

コードセグメントがＲＯＭ1404でアクセスされ実行されるべきであるか否かを決定するとき、ポインタ動作1436は現在のスレッドポインタをインクリメントして、次の実行スレッドを開始し、それによって類似の動作を継続するように第２のグループモードビット202 を獲得する。これはラウンドロビンシーケンスで各スレッドに対して継続される。

【００９７】

制御ベクトルが一度、モードビット202 の特定のグループに対して形成されると、優先度エンコーダ動作1438は次の“１”またはエネーブルされた制御ベクトルのビットを決定し、識別する。このようなビットが発見されると、優先度エンコーダ動作1438は実行のために、制御ベクトルのエネーブルビットに対応するアドレスをＲＯＭ1404中に生成する。

【００９８】

残りのスレッドを処理した後、およびモードビットがデコードされ、制御ベクトルが再度有効になった後、モードビット202 の最初のグループに戻るとき、マスキング動作1434は先の“１”または前に識別されたエネーブルされたビットをマスクするために使用される。これはマスク動作1434後に全ての残りのビットの解析を可能にする。

【００９９】
前述のプロセスは以下の表を使用して示されている。表９はサブジェクト頂点データについて実行される複数の式を示している。
表９
【数３】

```
R =   (a           )
R =   (a +     d*e  )
R =   (a + b*c +     f)
R =   (a + b*c + d*c   )
R = 1.0/(a           )
R = 1.0/(a +     d*e  )
R = 1.0/(a + b*c +     f)
R = 1.0/(a + b*c + d*c   )
```

【０１００】
示されているように、反転演算に加えて加算される積には４つの可能性が存在する（ａ，ｂ＊ｃ，ｄ＊ｅ，ｆおよび１／ｘ）。次に、モードフィールドが規定される。表１０はモードフィールドの対、ｍｏｄｅ．ｙとｍｏｄｅ．ｚを示し、それぞれ表９の演算の予め定められたセットに割当てられている。
【０１０１】
表１０
【数４】

```
mode.y[4] 0: R = a
          1: R = a + d*e
          2: R = a + b*c + f
          3: R = a + b*c + d*e

mode.z[2] 0: R = R
          1: R = 1.0/R
```

【０１０２】
その後、各演算は関連するアドレスと共にメモリに位置付けされる。表１１は関連する演算をそれぞれ有する複数のメモリアドレスを示している。また制御ベクトル定義のセットも示されている。
【０１０３】
表１１
【数５】

```
ROM[0]: R = a
ROM[1]: R = R + b*c
ROM[2]: R = R + d*e
ROM[3]: R = R + f
ROM[4]: R = 1.0/R

cv[0] = 1;
cv[1] = (mode.y==2 || mode.y==3) ? 1 : 0;
cv[2] = (mode.y==1 || mode.y==3) ? 1 : 0;
cv[3] = (mode.y==2) ? 1 : 0;
cv[4] = (mode.z==1) ? 1 : 0;
```

【０１０４】

表１２は１例の実行を示している。
【０１０５】
表１２
Ｒ＝ａ＋ｄ＊ｅは以下に対応する：
ｍｏｄｅ．ｙ＝１；
ｍｏｄｅ．ｚ＝０；
これは以下の制御ベクトルを与える：
ｃｖ［０］＝１；
ｃｖ［１］＝０；
ｃｖ［２］＝１；
ｃｖ［３］＝０；
ｃｖ［４］＝０；
実行
第１のサイクル：
ｃｖ［０］はＴＲＵＥであるので、ＲＯＭ［０］を実行
制御ベクトルにさらに多くのＴＲＵＥ値が存在するので、プログラムを終了
しない
第２のサイクル：
ｃｖ［１］はＦＡＬＳＥであるので、観察し続ける
ｃｖ［２］はＴＲＵＥであるので、ＲＯＭ［２］を実行
制御ベクトルにはＴＲＵＥ値がもはや存在しないので、プログラムを終了する。
【０１０６】
このようにして、変換モジュール52のシーケンサ1200はスレッドされたモードビット202から得られるスレッドされた制御ベクトルをステップし、対応する制御ベクトルビットが“ＴＲＵＥ”に設定されるあらゆるＲＯＭアドレスを実行する。制御ベクトルはＲＯＭと同一の長さを有する。シーケンサ1200は１つの“１”のレート、または予め定められたサイクル数毎にエネーブルされたビットで任意の制御ベクトルをステップできる。モードビット202 を使用しないコマンドはその簡潔性のためにオンザフライマイクロコードにより実行される。
【０１０７】
このような状態をモードビット202 の特有のストリングにより表示することによって、種々の動作の状態を決定するためにグラフィック処理ハードウェアの複数のイフ－ゼン（ｉｆ－ｔｈｅｎ）節を実行することは必要ではない。改良された性能はそれによって与えられる。概念的に、これはプログラム言語のイフ節がシーケンサ1200へ移動するかのようであり、シーケンサ1200はモードビット202 により示されるように“ＦＡＬＳＥ”状態で即時に命令をスキップする。
【０１０８】
前述したように、コードセグメントはＲＯＭに記憶され、これはモードビットにより識別される動作の種々の状態を処理することができる。１実施形態では、別々のコードセグメントはモードビットにより示される各動作を処理するために検索される。その代りとして、１つの包括性コードセグメントは可能であるそれぞれまたは幾つかの動作の組合わせを処理するために書込まれてもよい。しかしながら、各動作の組合わせでこのような大きいコードセグメントを生成することは付加的なコードスペースを必要とし、それ故、普通に使用される動作の組合わせだけでコードセグメントをモジュール化することが有効であることに注意する。
【０１０９】
モードビット202 は一度頂点が実行を開始すると変化しないので、制御ベクトルの生成はシーケンサに入る前に１つの頂点毎に１度実行されさえすればよい。しかしながら、これについての例外が動作が反復されるライティングのような幾つかのケースで生じる。最後の頂点命令が発見されるとき、シーケンス信号の終了（ＥＯＳ）が表明される。これは入

力および出力バッファの状態を変更し、図２８Ａと２８Ｂを参照して説明した方法で次のコマンドの開始を可能にするために使用される。ＥＯＳ信号は命令が処理される方法と類似の目的地バッファを解除するために遅延されるパイプラインであることに注意する。図４Ｂを参照する。

【０１１０】

図１４Ａはグラフィック処理中のスカラーおよびベクトル頂点データの管理を一体化するために使用される本発明の種々の機能コンポーネントを示したフロー図である。示されているように、１つの機能アスペクト1440はベクトル頂点データを処理モジュール、即ち加算器、乗算器等へ入力し、ベクトル頂点データを出力することを含んでいる。別の機能アスペクト1442では、ベクトル頂点データはベクトル処理モジュール、即ち加算器、乗算器等により処理され、これは再度ベクトル頂点データへ変換されるかスメアされるスカラー頂点データを出力する。

【０１１１】

さらに別の機能アスペクト1444では、ベクトル頂点データはマスクされ、それによってスカラー頂点データに変換され、その後、これはベクトル頂点データを生成する目的で、メモリ、即ちレジスタ論理装置中に記憶される。さらに別の機能アスペクト1446では、スカラー頂点データはベクトル処理モジュール、即ち加算器、乗算器等により抽出され、これはスカラー処理モジュール、即ち反転論理装置により処理され、スカラー頂点データをレンダリングする。このスカラー頂点データは再度ベクトル頂点データに変換される。

【０１１２】

図１４Ｂは図５の変換モジュール52に対応している図１４Ａに示されている本発明の機能コンポーネントの１つの可能な組合せ1451を示すフロー図である。機能アスペクト1444および1446は図４Ｂを参照して前述した方法と類似の方法で関連する遅延を有することに注意すべきである。図１４Ｃは図１４Ａに示されている本発明の機能コンポーネントの別の可能な組合せ1453を示すフロー図である。

【０１１３】

マルチプレクサは図１４Ａ－１４Ｃの機能モジュール中のベクトル頂点データからスカラー頂点データを抽出する。このようなマルチプレクサは種々の機能モジュールによる処理前に必要とされる任意のデータのスウィズリングに対しても応答可能である。１実施形態では、マルチプレクサはベクトルの頂点データを通過し回転することができ、他の処理用のＡＬＵ等の他のグラフィック処理モジュールに依存する。さらに別の実施形態では、マルチプレクサはペナルティなしで独立して属性を任意選択的に再配置することができる。

【０１１４】

図１４Ｄは特定用途向け集積回路（ＡＳＩＣ）のようなハードウェア構造によりグラフィックパイプラインにおけるグラフィック処理中に変換システムがブレンディングまたはスキン動作を行うように構成されている方法を示している。パイプラインでの処理中に、動作1470では、複数のマトリックス、それぞれ１つのマトリックスに対応する複数の加重値および頂点データが受信される。付加的なマトリックスのセットは正規の頂点データで必要とされる可能性があることに注意すべきである。

【０１１５】

続いて、動作1472では、複数の積の和がその後計算され、各積は頂点データと、１つのマトリックスと、そのマトリックスに対応する加重との乗算により計算される。このような積の和はその後、さらに処理を行うために動作1474で出力される。

【０１１６】

要約すると、以下の積の和が計算される。

式＃１

$i=1 \cdots x$ に対して $v' = \Sigma w_i * M_i * v$

ここでｖ＝入力された頂点データ、

ｗ＝加重値、

Ｍ＝マトリックス、

ｘ＝マトリックスの数、
ｖ'＝処理されるモジュールへ出力される頂点データ
式＃２
ｉ＝１…ｘに対してｎ'＝Σw_i＊I_i＊ｎ
ここでｎ＝入力された頂点データ（正規ベクトル）、
ｗ＝加重値、
Ｉ＝反転マトリックス（逆転置マトリックス）、
ｘ＝反転マトリックスの数、
ｎ'＝処理モジュールへ出力される頂点データ（正規ベクトル）
式＃３
V_s＝［O_x，O_y，O_z，φ］'＋
１／（ｖ"$_{we}$）＊［（ｖ"$_x$），（ｖ"$_y$）（ｖ"$_z$），１］'
ここでｖ"＝Ｃ＊ｖ'、
ｖ'＝式＃１からの積の和、
Ｃ＝［S_x，S_y，S_z，１］'＊Ｐ
Ｐ＝投影マトリックス、
v_s＝表示目的のスクリーンベクトル、
Ｏ＝ビューポートオフセット、
Ｓ＝ビューポートスケール

【０１１７】

前述した加重w_iを表す方法が多数存在することに注意すべきである。例えば式＃１と＃２では、ｉ＝１…（ｘ－１）ではw_x（w_i、ここではｉ＝ｘ）は式１－Σw_iにより計算されることが言われている。このようにして加重w_iを表すことにより、全ての加重ｗが１に合計されることが確実にされる。

【０１１８】

１実施形態では、マトリックスはモデルビューマトリックス（Ｍ）を含み、積の和（ｖ'）はライティング動作によりさらに処理されるために出力される（式１参照）。この積の和（ｖ'）はまた合成マトリックス（Ｃ）の使用によって表示目的で別の積の和（v_s）を生成するためにも使用される（式３参照）。マトリックスは反転マトリックス（Ｉ）を含み、頂点データは正規ベクトルデータ（ｎ）を含む。このようなケースでは、付加的な処理はライティング動作を含む（式＃２参照）。

【０１１９】

図１５は本発明の１実施形態によるライティングモジュール54の概略図である。示されているように、ライティングモジュール54は変換モジュール52が頂点データを出力するバッファ402 を含んでいる。示されているように、バッファ408 は通路1501によりライティングモジュール54をバイパスする。さらにライティングモジュール54には文脈メモリ1500とマイクロコードＲＯＭメモリ1502に結合されている。

【０１２０】

ライティングモジュール54はフォッグおよびポイントパラメータに加えてライティングを処理するように構成されている。使用において、ライティングモジュール54はバッファバイパス経路1501を制御し、拡散、ポイントサイズ、スペキュラー出力色およびフォッグ値を計算する。ライティングモジュール54は変換モジュール52と同一のモードビット202 を使用することに注意すべきである。

【０１２１】

ライティングモジュール54はさらに変換モジュール52に関してそれ程正確性を必要とせず、それ故、３ワードで組織される２２ビット浮動小数点値（1.8.13フォーマット）を処理する。第３のバッファ408 のデータは１２８ビットであるので、これはライティングモジュール54周辺のバイパス経路1501を使用する。ライティングモジュール54は事象駆動され、同時に図４Ａと４Ｂを参照して前述した変換モジュール52と類似の方法で３つのスレッドを実行する。ライティングモジュール54は外部ソースからコマンド発信許可を必要とす

ることに注意しなければならない。

【０１２２】

図１６は本発明の１実施形態による図１５のライティングモジュール54の機能装置を示す概略図である。示されているように、変換システムに結合されてそこから頂点データを受信するように構成されている入力バッファ402 が含まれている。前述したように、入力バッファ402 は第１の入力バッファ404 、第２の入力406 、第３の入力バッファ408 を含んでいる。第１のバッファの入力404 、第２の入力バッファ406 、第３の入力バッファ408 の入力は変換モジュール52の出力に結合されている。バイパスの目的で、第３の入力バッファ408 の出力は遅延素子1608によりライティングモジュール54の出力に結合されている。

【０１２３】

さらに、第１の入力バッファ404 の出力に結合されている第１の入力と、第２の入力バッファ406 の出力に結合されている第２の入力を有するＭＬＵ1610が含まれている。ＭＬＵ1610の出力はその第２の入力に結合されているフィードバックループ1612を有する。演算論理装置（ＡＬＵ）1614は第２の入力バッファ406 の出力に結合されている第１の入力を有する。ＡＬＵ1614はさらにＭＬＵ1610の出力に結合されている第２の入力を有する。ＡＬＵ1614の出力はライティングモジュール54の出力に結合されている。ＡＬＵ1614の出力と第３の入力バッファ408 の出力はマルチプレクサ1616によりライティングモジュール54の出力に結合されていることに注意すべきである。

【０１２４】

次に、ＡＬＵ1614の出力に結合されている入力と、ＡＬＵ1614の第１の入力に結合されている出力とを有する第１のレジスタ装置1618が設けられている。第２のレジスタ装置1620はＡＬＵ1614の出力に結合されている入力を有する。またこのような第２のレジスタ1620はＭＬＵ1610の第１の入力と第２の入力に結合されている出力を有する。

【０１２５】

ライティング論理装置（ＬＬＵ）1622もまた設けられ、ＡＬＵ1614の出力に結合されている第１の入力と、第１の入力バッファ404 の出力に結合されている第２の入力と、ＭＬＵ1610の第１の入力に結合されている出力とを有する。ＬＬＵ1622の第２の入力は遅延素子1624により第１の入力バッファ404 の出力に結合されていることに注意すべきである。さらに、ＬＬＵ1622の出力は先入れ先出しレジスタ装置1626を介してＭＬＵ1610の第１の入力に結合されている。図１６に示されているように、ＬＬＵ1622の出力はまた変換モジュール1628によりＭＬＵ1610の第１の入力にも結合されている。動作において、このような変換モジュール1628は変換モジュール52と類似の方法でスカラー頂点データをベクトル頂点データへ変換するように構成されている。

【０１２６】

最後に、メモリ1500はＭＬＵ1610の入力と演算論理装置1614の出力の少なくとも一方に結合されている。特に、メモリ1500はＭＬＵ1610の第１および第２の入力に結合されている読取り端子を有する。さらにメモリ1500はＡＬＵ1614の出力に結合されている書込み端子を有する。

【０１２７】

メモリは頂点データを処理するため、入力バッファ402 、ＭＬＵ1610、ＡＬＵ1614、第１のレジスタ装置1618、第２のレジスタ装置1620、ＬＬＵ1622と共に使用される複数の定数および変数を記憶している。

【０１２８】

図１７は本発明の１実施形態による図１６のライティングモジュール54のＭＬＵ1610の概略図である。示されているように、ライティングモジュール54のＭＬＵ1610は並列している３つの乗算器1700を含んでいる。動作において、本発明のＭＬＵ1610は２対３コンポーネントベクトルを乗算し、または１対３コンポーネントベクトルを通過するように構成されている。３コンポーネントベクトルの乗算はドット積または並列乗算により行われる。表１３はライティングモジュール54のＭＬＵ1610が実行できる動作を示している。

【０１２９】
表１３
【数６】

ZMLU_MULT	o[0] = a[0]*b[0], o[1] = a[1]*b[1], o[2] = a[2]*b[2]
ZMLU_PASA	o[0] = a[0], o[1] = a[1], o[2] = a[2]
ZMLU_PASB	o[0] = b[0], o[1] = b[1], o[2] = b[2]

【０１３０】
表１４はライティングモジュール54のＭＬＵ1610の可能なＡおよびＢ入力を示している。
表１４

ＭＡ＿Ｖ	ＶＢＵＦＦＥＲ
ＭＡ＿Ｌ	ＬＬＵ
ＭＡ＿Ｒ	ＲＬＵ［２，３］（ＭＢ＿Ｒと共有）
ＭＡ＿Ｃ	コンテキストメモリ（ＭＢ＿Ｃと共有）
ＭＢ＿Ｍ	ＭＬＵ
ＭＢ＿Ｗ	ＷＢＵＦＦＥＲ
ＭＢ＿Ｒ	ＲＬＵ［２，３］（ＭＡ＿Ｒと共有）
ＭＢ＿Ｃ	コンテキストメモリ（ＭＡ＿Ｃと共有）

【０１３１】
図１８は本発明の１実施形態による図１６のライティングモジュール54のＡＬＵ1614の概略図である。示されているように、ＡＬＵ1614は並列／直列の３つの加算器1800を含んでいる。使用において、ＡＬＵ1614は２対３コンポーネントベクトルを加算し、または１対３コンポーネントベクトルを通過するように構成されている。表１５はライティングモジュール54のＡＬＵ1614が実行できる種々の動作を示している。
【０１３２】
表１５
【数７】

ZALU_ADD	o[0] = a[0]+b[0], o[1] = a[1]+b[1], o[2] = a[2]+b[2]
ZALU_SUM3B	o[012] = b[0] + b[1] + b[2]
ZALU_PASA	o[0] = a[0], o[1] = a[1], o[2] = a[2]
ZALU_PASB	o[0] = b[0], o[1] = b[1], o[2] = b[2]

【０１３３】
表１６はライティングモジュール54のＡＬＵ1614の可能なＡおよびＢ入力を示している。
表１６

ＡＡ＿Ｗ	ＷＢＵＦＦＥＲ
ＡＡ＿Ｒ	ＲＬＵ［０，１］
ＡＢ＿Ｍ	ＭＬＵ

【０１３４】
図１９は本発明の１実施形態による図１６のライティングモジュール54のレジスタ装置1618と1620の概略図である。示されているように、レジスタ装置1618と1620はそれぞれ２セットのレジスタ1900を含んでおり、レジスタ1900はそれぞれ対応するマルチプレクサ1902の第１の入力に接続されている出力と、マルチプレクサ1902の第２の入力に結合されている入力とを有する。
【０１３５】
ライティングモジュール54のレジスタ装置1618と1620はＡＬＵ1614の２つのレジスタと、ＭＬＵ1610の２つのレジスタに分離される。１実施形態ではこれらのレジスタはスレッドされている。レジスタ装置1618と1620は書込みアドレスが読取りアドレスと同一であると

き、入力から出力へのバイパス通路のためにゼロの待ち時間を有する。

【０１３６】

図２０は本発明の１実施形態による図１６のライティングモジュール54のＬＬＵ1622の概略図である。ＬＬＵ1622はライティングモジュール54のライティング装置である。スカラーブロックは後に光＋マテリアルカラーを乗算するために使用されるライティング係数を計算する。ＬＬＵ1622は２つのＭＡＣと、インバータと、４つの小さいメモリとフラグレジスタを含んでいる。

【０１３７】

フラグレジスタはライティング方程式の条件付き部分を実行するために使用される。出力は環境、拡散、スペキュラー係数である。スカラーメモリはスペキュラー近似に使用される変数と定数を含んでいる。各メモリの第１の位置は（ctx0とctx2では）1.0 および（ctx1とctx3では）0.0 を含んでいる。１実施形態ではこれらはハードワイヤで結線され、ロードされる必要はない。

【０１３８】

使用において、ＬＬＵ1622は機能的に式（ｘ＋Ｌ）／（Ｍ＊ｘ＋Ｎ）を実行する。この式はスペキュラーライティング項を近似するために使用される。ＬＬＵ1622への入力はライティングモジュール54のＡＬＵ1614からであり、ライティング方程式で使用されるドット積である。図１６に関して前述したように、ＬＬＵ1622とＭＬＵ1610との間に出力ＦＩＦＯ1626が存在し、これはＭＬＵ1610が係数を必要とするまで、係数をバッファする。１実施形態ではこのようなＦＩＦＯ1626は遅延素子1608および1624、レジスタ1618および1620と共にスレッドされる。可能なカラーのマテリアル処理により、拡散およびスペキュラー出力がＭＬＵ1610により消費されるときはわからない。

【０１３９】

ライティングモジュール54はＲ，Ｇ，Ｂコンポーネントのみを処理するので、拡散出力アルファコンポーネントを処理するための特別に構成されたハードウェアが存在する。このような特別に構成されたハードウェアは２つのタイプのアルファコンポーネント、即ちｖｔｘカラーφ［Ｔｂｕｆｆｅｒ］および記憶されたｃｔｘ［Ｃｔｘ　ｓｔｏｒｅ］を出力できる。先のアルファコンポーネント間の選択はモードビット202 により支配される。

【０１４０】

動作において、ＬＬＵ1622はライティングの周囲（Ｃａ）、拡散（Ｃｄｅ）、スペキュラー（Ｃｓ）係数を計算する。これらの係数は頂点のカラーに対する光の影響を生成するため周囲、拡散、スペキュラーカラーと乗算される。表１６ＡはＬＬＵ1622により受信された入力のリストと、ライティングの環境（Ｃａ）、拡散（Ｃｄｅ）、スペキュラー（Ｃｓ）係数を生成するために実行される計算を含んでいる。任意の所望のハードウェア構成はＬＬＵ1622の構成に使用されることに注意する。１実施形態では、図２０で示されている特別な構成が使用される。

【０１４１】

表１６Ａ

入力規定：

ｎ＝正規ベクトル（変換エンジンから）

ｅ＝正規化されたアイベクトル（変換エンジンから）

ｌ＝正規化された光線ベクトル（変換エンジンから）

ｓ＝スポットライトベクトル＊光線ベクトル（変換エンジンから）

Ｄ＝距離ベクトル（１，ｄ，ｄ＊ｄ）（変換エンジンから）

ｈ＝半角ベクトル（変換エンジンから）

Ｋ＝減衰定数ベクトル（Ｋ０，Ｋ１，Ｋ２）（変換エンジンから）

ＬＬＵはその計算を実行するため以下のスカラーデータを受信する。

ｎ＊１　　（ＭＬＵ／ＡＬＵから）

ｎ＊ｈ　　（ＭＬＵ／ＡＬＵから）

Ｋ＊Ｄ　　（ＭＬＵ／ＡＬＵから）

ｓ　　（変換エンジンから）
パワー０　（ctx0-3メモリからのマテリアル指数）
パワー１　（ctx0-3メモリからのスポットライト指数）
距離　　（ctx0-3メモリから）
カットオフ　（ctx0-3メモリから）
無限大光
ＬＬＵ計算：
Ｃａ＝１．０
Ｃｄ＝ｎ＊ｌ
Ｃｓ＝（ｎ＊ｈ）＾power0
ローカル光
ＬＬＵ計算：
ａｔｔ＝１．０／（Ｋ＊Ｄ）
Ｃａ＝ａｔｔ
Ｃｄ＝ａｔｔ＊（ｎ＊ｌ）
Ｃｓ＝ａｔｔ＊（（ｎ＊ｈ）＾power0）
スポットライト
ＬＬＵ計算：
ａｔｔ＝（ｓ ＾power1）／（Ｋ＊Ｄ）
Ｃａ＝ａｔｔ
Ｃｄ＝ａｔｔ＊（ｎ＊ｌ）
Ｃｓ＝ａｔｔ＊（（ｎ＊ｈ）＾power0）

【０１４２】

前述したように、頂点シーケンサを制御するモードビットは頂点データ自体または頂点データから得られた結果により必ずしも変更されない。頂点データが頂点処理を変更することを可能にするため、ＬＬＵ1622は与えられたフラグレジスタ1623を使用する。ビットをこのフラグレジスタでＴＲＵＥに設定することにより、フラグが計算の出力制御で特定されるならば、計算結果の０．０にクランプすることが可能である。フラグレジスタ1623の別の使用はレジスタ書込みのための書込みマスクを設定することである。

【０１４３】

フラグレジスタ1623は性能のペナルティがなくライティング方程式で０．０へイフ／ゼン／エルスクランピングを行うためにＬＬＵ1622中に設けられる。種々のオペランドの符号ビットはフラグを設定する。表１６Ｂはフラグレジスタ1623のフラグが設定される方法と結果的なクランピングを示している。

【０１４４】

表１６Ｂ
無限光
ＬＬＵ計算：
Ｄｆｌａｇ＝（ｎ＊ｌ）のサインビット
Ｓｆｌａｇ＝（ｎ＊ｈ）のサインビット
クランプ：
Ｃａ＝（０　　）？０：Ｃａ；
Ｃｄ＝（Ｄｆｌａｇ　　）？０：Ｃｄ；
Ｃｘ＝（Ｄｆｌａｇ｜Ｓｆｌａｇ）？０：Ｃｓ；
局部光
ＬＬＵ計算：
Ｒｆｌａｇ＝（range-d ）のサインビット
Ｄｆｌａｇ＝（ｎ＊ｌ）のサインビット
Ｓｆｌａｇ＝（ｎ＊ｈ）のサインビット
クランプ：

Ｃａ＝（Ｒｆｌａｇ　　）？０：Ｃａ；
Ｃｄ＝（Ｒｆｌａｇ｜Ｄｆｌａｇ　　）？０：Ｃｄ；
Ｃｘ＝（Ｒｆｌａｇ｜Ｄｆｌａｇ｜Ｓｆｌａｇ）？０：Ｃｓ；
スポットライト
ＬＬＵ計算：
Ｃｆｌａｇ＝（s-cutoff）のサインビット
Ｒｆｌａｇ＝（range-d ）のサインビット
Ｄｆｌａｇ＝（ｎ＊ｌ）のサインビット
Ｓｆｌａｇ＝（ｎ＊ｈ）のサインビット
クランプ：
Ｃａ＝（Ｃｆｌａｇ｜Ｒｆｌａｇ　　）？０：Ｃａ；
Ｃｄ＝（Ｃｆｌａｇ｜Ｒｆｌａｇ｜Ｄｆｌａｇ　　）？０：Ｃｄ；
Ｃｘ＝（Ｃｆｌａｇ｜Ｒｆｌａｇ｜Ｄｆｌａｇ｜Ｓｆｌａｇ）？０：Ｃｓ；

【０１４５】

図２１は本発明の１実施形態による図１６のライティングモジュールに関連したフラグレジスタ1623の組織を示している。フラグレジスタ1623は８つの１ビットフラグを含み、ＡＬＵ（ＩＦＬＡＧ）またはＭＡＣ０（ＭＦＬＡＧ）出力の符号ビットにより設定される。

【０１４６】

ＬＬＵ1622が３ワードにスメアされる場合ＭＬＵ1610へスカラ値を出力するとき、フラグレジスタのマスクを特定する。レジスタとマスクが真であるならば、０．０は出力を置換える。表１７は出力された環境、拡散、スペキュラー属性で使用される図２１の種々のフラグを示している。

表１７

周囲マスク：　　Ｃ，Ｒ，　Ｕ
拡散マスク：　Ｄ，　Ｃ，Ｒ，　Ｕ
スペキュラーマスク：Ｄ，Ｓ，Ｃ，Ｒ，Ｔ，Ｕ

【０１４７】

スペキュラー項で使用される近似は実際のｃｏｓ（ｔｈｅｔａ）**が０．０になる場合、負になる。結果として、クランピング動作を実行する必要がある。このため、Ｔ，Ｕフラグが使用される。表１８はＬＬＵ1622の機能論理装置（ＦＬＵ）1621が行うことができる種々の動作を示している。図２０に注意する。

【０１４８】

表１８

ＺＦＬＵ＿ＩＮＶ	ｏ＝１／ａ　（仮数の正確度－１２ビット）
ＺＦＬＵ＿ＩＳＱ	ｏ＝１／ｓｑｒｔ（ａ）（仮数の正確度－６ビット）
ＺＦＬＵ＿ＰＡＳＳ	ｏ＝ａ
ＺＦＬＵ＿ＰＡＳＳ１	ｏ＝１．０
ＺＦＬＵ＿ＭＩＮ１	ｏ＝（ａ＜１．０）？ａ：１．０
ＺＦＬＵ＿ＮＯＰ	ｏ＝０．０

【０１４９】

図２２は本発明の１実施形態による図１６のライティングモジュール54に関連したマイクロコードフィールドを示す図である。示されているように、ライティングモジュール54のマイクロコードは全体幅が８５ビットである３３フィールドに配置されている。フィールドは装置のデータ流を整合するように遅延される。ＭＬＵ動作は遅延ゼロで行われ、ＡＬＵ動作は遅延１で行われ、ＲＬＵ、ＬＬＵ出力動作は遅延２で行われる。各遅延は３サイクルに等しい。

【０１５０】

図２３は本発明の１実施形態による図１６のライティングモジュール54に関連したシーケンサ2300の概略図である。示されているように、ライティングモジュール54のシーケンサ2300はプロセス動作の複数のモードの状態を示すモードビット202 を受信するように構成

されている入力バッファ2302を含んでいる。また、それぞれモードの状態にしたがってプロセス動作を実行するように構成されているコードセグメントを記憶できるメモリ1502も含まれている。

【０１５１】

シーケンスモジュール2306はモードビットから得られる制御ベクトル2305に基づいてメモリ1502中の複数のアドレスを識別するためメモリ1502とバッファ2302との間に結合されている。シーケンスモジュール2306はさらに、ライティングモジュール54を動作するために使用されるコードセグメントを検索するためにメモリ1502中のアドレスをアクセスするように構成されている。

【０１５２】

ライティングモジュール54のシーケンサ2300は変換モジュール52のシーケンサと類似している。動作において、ライティングモジュール54のシーケンサ2300はスレッドされたモードビット202 から得られるスレッドされた制御ベクトルによりステップし、それぞれのＲＯＭアドレスを実行し、その対応する制御ベクトルビットは“１”に設定される。制御ベクトルはＲＯＭが有するワードと同数のビットを有する。シーケンサ2300はスレッド毎に予め定められた数のサイクルで１つの“１”またはエネーブルビットのレートで任意の制御ベクトルをステップできる。モードビット202 を使用しないコマンドはオンザフライマイクロコード発生により実行される。ライティングモジュール54のシーケンサ2300と変換モジュール52のシーケンサ1200との主な違いは、ライティングモジュール54のシーケンサ2300はループバックし８回までライティングコードを実行できることである。

【０１５３】

ライティングモジュール54のシーケンサ2300はそれぞれ新しい頂点ではゼロで開始し、マイクロコードシーケンスの終了時では１だけインクリメントする光カウンタを有する。モードビット202 のＬＩＳフィールドが一致するビットフィールドで“１”を含んでいるならば、シーケンサ2300は戻り、ライティングマイクロコードブロックの開始時でスタートする。これはゼロがＬＩＳフィールドで発見されるか、８つの光が行われるまで継続する。カラーの累算は拡散およびスペキュラーカラーを記憶するＡＬＵレジスタを（１光線毎に）インクリメントすることによって行われる。自動メモリアドレスのインデックスは各光線で正確なパラメータをフェッチするために光カウンタを使用して実行される。

【０１５４】

図２４は本発明の１実施形態にしたがって変換モジュール52およびライティングモジュール54のシーケンサが関連したバッファの入力および出力を制御することができる方法について詳細に説明するフローチャートである。示されているように、頂点データは動作2420でバッファの第１のセットの１つのバッファで最初に受信される。頂点データが受信されるバッファはラウンドロビンシーケンスに基づいている。

【０１５５】

続いて、動作2422では、バッファの第２のセットのエンプティバッファもまたラウンドロビンシーケンスに基づいて識別される。変換モジュール52は第１のセットのバッファと、第２のセットのバッファとの間に結合されている。第２のセットのバッファのエンプティバッファが識別されるとき、頂点データは変換モジュールで処理され、変換モジュールから第２のセットのバッファの識別されたエンプティバッファへ出力される。動作ステップ2424および2426を参照。

【０１５６】

同様に、バッファの第３のセットのエンプティバッファまたはメモリ中のスロット或いはスペースは動作2428でラウンドロビンシーケンスに基づいて識別される。ライティングモジュール54はバッファの第２のセットと第３のセットの間に結合されている。バッファの第３のセットのエンプティバッファが識別されるとき、頂点データは動作2430で示されているようにライティングモジュールで処理される。頂点データはしたがってライティングモジュール52からバッファの第３のセットの識別されたエンプティバッファへ出力される。動作2432を参照。バッファまたはメモリ中のスロットの数はフレキシブルであり、変更

されてもよいことに注意すべきである。

【０１５７】

図２５は図２４の方法にしたがって変換モジュール52およびライティングモジュール54のシーケンサが関連したバッファの入力および出力を制御することができる方法の説明図である。示されているように、第１のセットのバッファまたは入力バッファ400 は変換モジュール52に出力を供給し、変換モジュール52は第２のセットのバッファまたは中間バッファ404 、406 に出力を与える。第２のセットのバッファ404 、406 はメモリ2550へ出力（ドレイン）するライティングモジュール54に出力を与える。

【０１５８】

図２５で説明されている方法を実行するため、メモリ2550のスロットと、第１および第２のセットのバッファはそれぞれ頂点データを最初に受信したときに特有の識別子をそれぞれ割当てられる。さらに、各バッファの現在の状態は追跡される。このような状態は割当てられた状態、有効な状態、アクチブ状態または行われた状態を含んでいる。

【０１５９】

割当てられた状態は、バッファ／スロットが先のグラフィック処理モジュール、即ち変換モジュールまたはライティングモジュールの出力を受信するように既に割当てられていることを示している。書込みポインタがラウンドロビンシーケンスでバッファ／スロットを走査しているとき、割当てられた状態のバッファ／スロットはこのような書込みポインタを次のバッファまたはスロットにインクリメントさせる。

【０１６０】

バッファ／スロットが有効な状態であるならば、そのバッファ／スロットは頂点データを受信するために使用される。他方で、アクチブ状態はバッファ／スロットが現在、実行状態であるかまたは頂点データを受信していることを示す。このアクチブ状態はスレッドが完了するまで維持され、その後読取りポインタをインクリメントし、したがってバッファ／スロットを有効状態に戻す。第１のセットのバッファ400 はそれらを割当てるグラフィック処理モジュールが先に存在しないので、単に有効状態であることだけができることに注意する。

【０１６１】

状態のシーケンスの１例を説明する。第１のセットのバッファ400 と新しいコマンドビットのセット200 の一方で頂点データを受信するとき、このようなバッファは有効状態に置かれ、その後バッファ402 、404 の第２のセットの１つが変換モジュール52の出力の予測において割当てられた状態に置かれる。

【０１６２】

バッファ404 、406 の第２のセットが割当に使用可能ではないならば、第１のセットのバッファ400 中の頂点データは処理されることができない。さらに実行されるコードセグメントが同時に行われる他のコードセグメントと干渉するか否かを決定するためのチェックが行われる。干渉するならば、第１のセットのバッファ400 の頂点データは処理されずストール（機能停止）状態が開始される。

【０１６３】

第２のセットのバッファ404 、406 の１つが割当状態に置かれた後、第１のセットのバッファ400 はアクチブ状態に置かれる。変換モジュール52が実行を終了したとき、第２のセットのバッファ404 、406 は読取られ、その後有効状態に置かれる。これらの状態の変化は第２のセット404 、406 とメモリ2550のスロット間の頂点データの転送中も同様に行われる。

【０１６４】

図２５Ｂは設定モジュール57とトラバーサルモジュール58とを含むラスター化モジュール56を示している。ラスター化モジュール56は代わりの方法でエリアベースのラスター化を実行するように構成されている。特に、複数の多角形を規定するセンスポイントがプリミティブに、またはその近くに位置され、その後一次方程式がプリミティブ中に存在する画素を決定するためにそのポイントにおいて評価される。動作中、この評価はポイントが効

率的な目的で代わりの方法で移動されるときに反復される。さらに、ラスター化モジュール56は何等クリッピングプロセスなしで動作するように構成される。

【０１６５】

図２６はラスター化モジュール56の設定モジュール57の概略図である。示されているように、設定モジュール57は所望の浮動小数点計算を実行するためにデータと制御信号をそれらの適切な機能装置へ導く処理をする制御セクション61を含んでいる。プリミティブシーケンサ62は頂点のシーケンスを三角形、直線または点に変える処理をする。さらに浮動小数点データパスセクション64は設定装置で必要とされる数学を実行するマルチプレクサおよび浮動小数点計算装置を含んでいる。

【０１６６】

図２６の参照を続けると、ラスター化装置は整数値でのみ動作するので、出力フォーマットセクション63はエッジスロープとエッジ値の内部浮動小数点フォーマットをラスター化装置に適している整数のフォーマットに変換する処理をする。勿論、別の実施形態では、ラスター化装置は浮動小数点を使用し、したがって出力フォーマットセクション63の必要性をなくすことができる。

【０１６７】

動作において、出力フォーマットセクション63はブロック浮動小数点変換を実行する。よく知られているように、所定の数、即ち２．３４e^{10}では、浮動小数点フォーマットは仮数（２．３４）とその指数（１０）を追跡する。ブロック浮動小数点変換は基本的に指数が同一であるように、入来するデータの仮数の小数点位置を操作する。このため、指数はラスター化モジュール56で処理される必要はない。

【０１６８】

図２６Ａは図２５Ｂのラスター化モジュール56の設定モジュール57によって計算される種々のパラメータを示している。このようなパラメータは関連する機能を実行するためにラスター化モジュール56に必要とされる。プリミティブ2600を受信するとき、設定モジュール57はプリミティブ2600のスロープ2601、スタート位置2602、スタート値2604を含む３つの値を計算する。

【０１６９】

スロープ2601はラスター化中に使用されるプリミティブ2600のエッジの一次方程式の係数を生成するために使用される。スロープ2601は例えば以下示す式＃４および＃５を使用することにより計算される。

式＃４および＃５

$スロープ_A = y_0 - y_1$

$スロープ_B = x_1 - x_0$

ここでy_0、y_1およびx_0、x_1は図２６Ａで示されている頂点の座標である。

【０１７０】

スロープはまた１つの回転動作等を使用することによって頂点の座標を使用して計算されることに注意する。

【０１７１】

スタート位置2602はさらに以下詳細に説明するようにエリアラスター化のスタート点を示している。スタート値2604は図２６Ａで示されている陰影を付けられた三角形の面積に等しく、またエリアベースのラスター化プロセス中にも使用される。このようなスタート値2604はスクリーンについてのラスター位置をステップするように選択され、各ステップでスロープを付加することはラスター位置がエッジにあるとき丁度ゼロに等しい。スタート値2604の計算は以下の式＃６を使用して実現される。

式＃６

$starting\_value = スロープ_A *（x_s - x_0）+ スロープ_B *（y_s - y_0）$

ここで、x_s，y_s＝スタート位置2602、

$スロープ_A$，$スロープ_B$＝図２６Ａで示されている座標に基づいた１つのエッジのスロープ、

x_0，y_0＝図２６Ａで示されているエッジの頂点の１つの座標

【０１７２】

前述の値はまた他のタイプのプリミティブに対して計算されることを理解すべきである。例えば、直線の場合、余分のスロープは４つの側面の境界のあるボックスで計算されなければならない。このようなスロープは境界のあるボックスの反対側のスロープの逆数を取ることにより容易に計算されることができる。余分のスロープの計算に加えて、別のスタート値が直線のプリミティブの場合に計算されることを必要とすることに注意すべきである。

【０１７３】

図２７はラスター化モジュール56が例えば三角形等の複数のプリミティブのうちの１つを処理する方法を示している。特に、最初の動作は最初にラスター化装置のモジュール56の設定モジュール57により実行される。プリミティブを受信するとき、一次方程式の一次方程式係数は当業者によく知られた方法で図２６Ａのスロープ2601を使用して動作2700でプリミティブを規定する直線で決定される。よく知られているように、３つの一次方程式が三角形を規定するのに必要とされる。他方で、直線のようなプリミティブは４つの側面と４つの一次方程式により長方形または平行四辺形として描かれる。

【０１７４】

その後、動作2702では、任意のプリミティブ頂点が負のＷ－座標を有するならば、その一次方程式係数は変更される。このプロセスに関する付加的な情報を図３２を参照してさらに詳細に説明する。

【０１７５】

ラスター化モジュール56の設定モジュール57もまたプリミティブの境界のあるボックスを計算することに注意しなければならない。ほとんどの三角形では、境界を有するボックスは３つの頂点の最小値および最大値を含んでいる。直線では、境界を有するボックスの４つの平行四辺形のコーナーが計算される。負のＷ－座標の頂点を有する三角形または直線では、描かれるエリアは頂点の凸閉の殻を超えて延在する。

【０１７６】

ＯｐｅｎＧＬ（商標名）のコマンドの１つは描かれない境界外を規定するシザー長方形である。ラスター化モジュール56の設定モジュール57は境界のあるボックスとシザー長方形との交差点を計算する。シザー長方形は長方形であるので、４つの付加的な一次方程式が与えられる。シザー長方形に関連する一次方程式は平凡な形状、即ち水平または垂直を有することに注意する。

【０１７７】

さらに、３－Ｄスペースでは、近距離の平面と遠距離の平面とは平行であり、視線に対して直角である。プリミティブが三角形である場合、３つの頂点が含まれ、任意の方位を有する平面を規定する。プリミティブの平面と、近距離および遠距離の平面との交差点は２つの関連する一次方程式を有する２つの直線を含んでいる。

【０１７８】

したがって、各プリミティブはそれが三角形または直線の形態を取るかに応じて全部で９または１０の一次方程式をそれぞれ有する。再び三角形の場合、このような一次方程式は三角形を規定する３つの一次方程式と、境界のあるボックスを規定する４つの一次方程式と、プリミティブが存在する平面と近距離の平面および遠距離の平面との交差点を規定する２つの一次方程式とを含んでいる。

【０１７９】

図２７を参照し続けると、プロセスは動作2704で進行し、プリミティブ上またはその近くの複数の点を位置付ける。スタート位置2602は図２６Ａで示されているように、このような位置付けを指示している。このような点は含まれる凸形領域を規定し、凸形領域のコーナーに位置している。図２７Ａは例えば長方形等の凸形領域2707を囲むこのようなセンスポイント2705を示している。１実施形態では、このような長方形はサイズが８×２画素である。さらに点はプリミティブの上部の頂点を囲むように最初に位置される。選択肢とし

て、これは切捨てを使用して実現されてもよい。

【０１８０】

プリミティブが一度位置付けられると、プロセスは以下説明する方法でプリミティブの行を処理することにより動作2706で開始するトラバーサルモジュール58により継続される。各行の処理後、ジャンプ位置が決定2708で発見されるか否かを決定する。ジャンプ位置は次の行を処理するためスタート位置にあり、以下詳細に説明する。決定2708でジャンプ位置が発見されたことが決定されるならば、凸面領域を規定するセンスポイントは動作2710に移動される。しかしながら、ジャンプ位置が発見されていないことが決定されたならば、プロセスは終了される。別の実施形態では、列、対角線または任意の他のタイプのストリングが行の代わりに動作2706で処理されることに注意すべきである。

【０１８１】

図２８は図２７の処理行動作2706に関連した本発明のプロセスを示すフローチャートである。示されているように、プロセスは多角形を規定するセンスポイントが決定2801で右に移動されるか否かを決定するため、動作2800でセンスポイントを計算することにより開始する。このような決定は最も右のセンスポイントの位置に基づいて行われる。最も右のセンスポイントがプリミティブの同一エッジ外に位置されないならば、右方向の移動は許容され、現在位置の右への位置（ＸおよびＹ座標）は動作2802でスナップ位置として記憶される。しかしながら、最も右のセンスポイントがプリミティブの１以上のエッジ外に位置されるならば、右方向の移動は許容されず、動作2802はスキップされる。

【０１８２】

次に、一次方程式は動作2804で凸形領域、例えば長方形の点で評価される。この評価は点がプリミティブ中に存在するか否かの決定を含んでいる。ポイントがプリミティブ中に存在するか否かについてのこのような決定は、各一次方程式の評価が各センスポイントで正の値または負の値を与えるか否かを決定することを含んでいる。

【０１８３】

一次方程式はプリミティブ内では正であり、その外では負であるように公式化されることができる。画素が丁度エッジ上に存在する包含的なエッジが描かれ、ゼロに評価され、正として扱われる。描かれるべきではない排他的なエッジは開始の一次方程式の値から１の値を最初に減算することにより負にされることができる。したがって、排他的エッジ上の画素は正のゼロの代わりに負値（－１）に評価される。これはセンスポイントの移行が包含的／排他的ポリシーを無視し、単に一次方程式の符号を試験することを許容する。

【０１８４】

一次方程式が点において評価された後、決定2806でセンスポイントの現在の位置がジャンプ位置を構成するか否かが決定される。２つの下部のセンスポイントが両者ともエッジ外でなければ、ジャンプ位置は記憶されることに注意すべきである。決定2806で、ジャンプ位置が発見されたことが決定されたならば、動作2808でジャンプ位置が計算され記憶される（または存在するならば先に記憶されたジャンプ位置で置換する）。しかしながらノーであるならば、動作2808はスキップされる。

【０１８５】

図２８の参照を続けると、決定2810で、最も左のセンスポイントが両者ともプリミティブのエッジ外であるか否かが決定される。このプロセスは再び両者の最も左のセンスポイントの一次方程式の評価が正または負値を与えるか否かを決定することを含んでいる。特に適切なセンスポイントで９または１０のエッジ式の係数を計算するとき、９または１０値が与えられ、それらは９または１０の符号ビットを有する。現在の側面が完全にエッジ外であるか否かを決定するために、例えば本発明は２つのセンスポイントからの１０の符号ビットを共に論理積（ＡＮＤ）処理する。任意のビットが残存するならば、両者のポイントはそのエッジ外である。

【０１８６】

最も左のセンスポイントが両者ともプリミティブエッジ外ではないことが決定されたならば、左方向にあると考えられるプリミティブの部分がさらに残留していることが結論付け

され。センスポイントは動作2812に左へ移動される。決定2810で、両者の最も左のセンスポイントがプリミティブのエッジ外であることが決定されたならば。左方向にあると考えられるプリミティブの部分がさらに残留していないことが結論付けされる。次に、決定2814で、動作2802から得られたスナップ位置が存在するか否かの決定が行われる。

【０１８７】

決定2814で、スナップ位置が存在しないことが決定されたならば、プロセスは行われる。しかしながら、スナップ位置が存在するならば、センスポイントは動作2816でスナップ位置に移動される。その後、2804－2812の動作に類似した動作はプリミティブの右側をマップするように行わされる。これは凸形領域の点で一次方程式を評価することにより動作2818で開始する。

【０１８８】

一次方程式が点で評価された後、センスポイントの現在の位置が動作2820でジャンプ位置を構成するか否かが決定される。決定2820で、ジャンプ位置が発見されたことが決定されたならば、動作2822でジャンプ位置が計算され記憶される。ノーであるならば、動作2822はスキップされる。

【０１８９】

図２８の参照を続けると、決定2824で、最も右のセンスポイントが両者ともプリミティブのエッジ外であるか否かが決定される。最も右のセンスポイントが両者ともプリミティブのエッジ外ではないことが決定されたならば、右方向にあると考えられるプリミティブ部分がさらに残留していることが結論付けされ、センスポイントは動作2826で右に移動される、決定2824で、最も右のセンスポイントが両者ともプリミティブのエッジ外にあることが決定されたならば、右方向にあると考えられるプリミティブの部分がさらに残留していないことが結論付けされ、瞬時のプロセスが実行される。

【０１９０】

図２８Ａと図２８Ｂは本発明のセンスポイントがプリミティブ2850に関して移動されるシーケンスを示している。種々の代りの方法が決定2800で点が左に移動することができるか否かを決定し、最初に右に進行することを含むことに注意する。一次方程式は点が任意の所望の方法でプリミティブ内または外であるかを示すために規定される。

【０１９１】

反復するループのステップ動作を防ぐため、本発明はしたがってラスター化中に全体的な移動方向を使用する。最初の構成はトップ－ダウンを行い、次へステップダウンする前に１つの行の１つ１つの凸領域に行く。行のトップ－ダウンを行い、右その後左へ、または左その後右へステップしないことによりループは阻止される。

【０１９２】

前述のプロセスの１例は図２７Ａの多角形を規定する点Ｐ１、Ｐ２、Ｐ３、Ｐ４を参照して示されている、動作において、隣接するセンスポイントの対はそれらの方向のステッピングが生産的（productive）であるか否かを決定するため検査されることができる。例えば図２７Ａ中のＰ３とＰ４の両者が多角形のエッジ外であるが、Ｐ１および／またはＰ２は多角形のエッジ外ではないならば、明白に描くことのできる内部エリアは右ではなく左に位置する。したがってセンスポイントは右へ移動すべきではない。反対に、Ｐ３とＰ４の両者が全てのエッジ内であるならば、描くことのできる内部エリアは丁度Ｐ３とＰ４を越えて存在し、右へのステップが適切である。Ｐ３とＰ４が同じエッジの外ではないならば、右へのステップが生産的である。この同じ論理はＰ１とＰ３により誘導される上方向へのステップまたは、Ｐ１とＰ２により誘導される左のステップ、またはＰ２とＰ４に基づいた下方向のステップにも適用される。

【０１９３】

前述のプロセスはしたがってガイドとしてセンスポイントを使用して、プリミティブの内部周辺の点により規定される凸形区域を移動またはステップする。点によって規定される凸形領域が大きいので、多数の画素は同時に試験される。使用中、全てのセンスポイントがプリミティブの全てのエッジ内であるならば、全ての囲まれた画素は描かれることが可

能でなければならない（凸形のプリミティブを想定する）。コーナー部を検査することにより多くの利点が与えられ、即ちプリミティブの任意のエリアを与える能力は内部、外部または分割である。後者のケースでのみ、点により規定される凸形領域の個々の画素が試験される必要がある。このような場合、点により規定される凸形領域の画素はこれらがプリミティブに存在するか否かを決定するために別の方法により１つづつ試験される。さらに、センスポイントはエリアを分割するエッジと分割しないエッジを規定することにより、必要な試験の量を減少する。

【０１９４】

図２９は図２７の処理行動作2706に関連した本発明のプロセスの別の犂耕体プロセスを示すフローチャートである。示されているように、最初に決定2900で、先の移動が第１または第２の方向であるかを決定する。実際に先の移動が存在しなかったならば、デフォルトの先の移動が仮定される。決定2900で、先の移動が第２の方向であることが決定されたならば、動作2902で図２８の動作2804と類似した方法で一次方程式が凸形領域、例えば長方形の点で評価される。

【０１９５】

図２９の参照を続けると、次に、決定2904で、長方形の第１の側面のセンスポイントが両者ともプリミティブのエッジ外であるか否かに関して決定が行われる。ノーであるならば、センスポイントは動作2906で第１の方向で移動またはステップされる。長方形の第１の側面のセンスポイントが両者ともプリミティブのエッジ外であるという決定が行われると、決定2905で、点が下方向に移動できるか否か、換言すると、現在位置がジャンプ位置を構成するか否かが決定される。イエスならば、動作2908でジャンプ位置が計算され、記憶され、その後プロセスが行われる。

【０１９６】

他方で、決定2900で、先の移動が第１の方向であることが決定されたならば、動作2902－2908と類似の動作が実行される、特に動作2910で、動作一次方程式は凸形領域、例えば長方形の点で評価される。決定2912で、長方形の第２の側面のセンスポイントが両者ともプリミティブのエッジ外であるか否かに関する決定が行われる。ノーであるならば、センスポイントは動作2914で第２の方向で移動またはステップされる。長方形の第２の側面のセンスポイントが両者ともプリミティブのエッジ外であるという決定が行われると、決定2913で、点が下方向に移動できるか否か、換言すると、現在位置がジャンプ位置を構成するか否かが決定される。イエスならば、動作2916でジャンプ位置が計算され、記憶され、その後プロセスが行われる。

【０１９７】

図２９Ａは図２９の犂耕体プロセスにしたがって本発明のセンスポイントがプリミティブに関して移動されるシーケンスを示している。前述の犂耕体ラスター化はハードウェアに対してより良好な性能を与えるあるルールに従うようにシーケンスを規制する。示されているように、犂耕体ラスター化は前後に曲がる蛇行パターンを与える。水平の犂耕体シーケンスは例えばプリミティブ三角形内に全ての画素を生成し、それらは左から右へ１つの行に存在し、その後、次の行で右から左へ画素を生成する。このような折曲がったパスは生成された画素から最近予め発生された画素までの平均距離が比較的小さいことを確実にする。

【０１９８】

ほぼ最近に予め発生された画素の発生は、画素および／またはそれらの対応するテキスチャ値が限定されたサイズのメモリ中に維持されるときに重要である。犂耕体シーケンスはこのようなメモリに既にロードされている画素またはテキスチャを頻繁に発見し、それ故メモリのロードの反復が行われる頻度が少なくなる。

【０１９９】

１つの選択肢として、ラスター化の前にプリミティブを複数の部分に分離する少なくとも１つの境界が使用される。動作において、点は各部分で別々に移動される。さらに、点は第２の部分で移動される前に第１の部分の全体を移動される。

【０２００】
図３０は境界を使用する別の犂耕体プロセスを示しているフローチャートである。１つの選択肢として、境界を使用するか否かの決定はプリミティブの大きさに基づく。図３０で示されているように、境界を処理する犂耕体プロセスは、少なくとも１つの境界が規定され、プリミティブを複数の部分またはスワス（swath ）に分割する付加的な動作3000を除いて図２７のプロセスと類似している。
【０２０１】
図３０の参照を続けると、付加的な決定3001はプリミティブのあらゆる部分の完了にしたがう。特に、決定3001で、隣接部分のスタート位置が動作3006で発見されるか否かが決定される。イエスであるならば、センスポイントにより規定される凸形領域は動作3002でプリミティブの隣接部分のスタート点に移動され、動作3004－3010はプリミティブの新しい部分に対して反復される。さらに動作3006のスタート位置の決定に関する情報を図３１を参照してさらに詳細に説明する。
【０２０２】
図３１Ａは図３０の境界ベースの犂耕体プロセスにしたがって本発明の凸形状領域がプリミティブに関して移動されるプロセスを示している。示されているように、処理される第１の部分はプリミティブの最上部の頂点を含む部分である。動作中、左の隣接部分が処理され、その後近接する左の隣接部分が処理され、以下同様に処理される。これは、左の隣接部分がなくなるまで継続される。次に第１の部分の右への隣接部分が処理され、その後、近接する右の隣接部分が処理され、全ての右の隣接部分が処理されるまで継続される。他のタイプの順序付け方式がユーザの要望にしたがって使用されてもよいことを認識すべきである。
【０２０３】
図３１は図３０のプロセス行の動作3006に関連したプロセスを示すフローチャートである。このようなプロセスは決定3118と3121を除いて図２９の犂耕体プロセスに類似している。決定3118と3120の両者は任意のセンスポイントが任意の境界を通過しているか否かを決定する。センスポイントが境界内であることが決定されさえすれば、それぞれのループが継続される。
【０２０４】
動作3119と3121では、プリミティブの隣接部分のスタート位置は捜索され、決定3118と3120で、凸形領域の任意のセンスポイントがそれぞれ任意の境界を通過していることを決定したときに記憶される。図３１Ａで示されているように、このようなスタート位置3126は境界を越えて存在するプリミティブ部分の最上部点としてそれぞれ規定される。この位置を記憶することにより、プロセスがプリミティブにおける隣接する境界の規定された部分で反復されるときにスタート点が与えられる。
【０２０５】
動作3119と3121は両者ともプリミティブの第１の部分を処理しながら実行されることに注意する。図３１で明白に示していないが、部分を第１の部分の左に処理するときこのような動作の第１の動作だけが行われ、部分を第１の部分の右に処理するとき、このような動作の第２の動作だけが行われる。換言すると、部分を第１の部分の左に処理するとき、スタート位置は現在処理された部分の最も左の境界が超過されたときだけ決定される。同様に、部分を第１の部分の右に処理するとき、スタート位置は現在処理された部分の最も右の境界が超過されたときだけ決定される。
【０２０６】
ラスター化中に境界を使用することは、パイプライン処理中の非常に臨界的な問題を解決する。プリミティブが非常に広いならば、１つの行の画素に関連する記憶媒体は限定されたサイズのメモリに適合しない。境界によるラスター化は三角形を限定された幅の行（または列）に分離し、次の部分へ移動する前に、このような部分内に全ての画素を生成する。
【０２０７】

例えば、三角形が１００画素幅であっても、限定されたサイズの画素またはテキスチャメモリは先の２０画素の情報だけを保持する。画素シーケンスを１０画素幅の垂直部分内に存在するように制限することによって、以前のおよび現在の行の全ての画素はメモリに適合することが可能である。これは、境界の規定された部分内の犂耕体シーケンスが常にメモリの（存在するならば）現在の行の以前の画素と、メモリの（存在するならば）上の行の画素とを有することを意味している。

【０２０８】

ほとんどの基礎的なメモリシステムはブロック単位のあるオーバーヘッドによりデータのブロックを転送する。メモリシステムに対する小さいアクセスはこのオーバーヘッドにより重いペナルティを課される。効率的であるように、大きいアクセスが使用され、ブロックの残りは次に使用される場合のために維持される。さらに、キャッシュメモリシステムは複数のこれらの最近のブロックを維持し、メモリアクセスが避けられることができる確率を増加させる。

【０２０９】

本発明の犂耕体シーケンスは、現在のラインの１端部のすぐ下の画素を反転し処理するときにシングル－リテイン－ブロックを使用する。さらに、犂耕体シーケンスはラスター化を特定サイズの部分に限定するときキャッシュを使用する。特に部分内の２つの走査線はキャッシュに適合され、第２の走査線を通じて第１の走査線のキャッシュ記憶から利点が得られる。

【０２１０】

シーケンスまたは境界の規定された部分の数には制限はない。本発明は垂直部分および水平の犂耕体パターンの例を使用したが、類似の原理が水平部分および垂直の犂耕体パターンまたは、対角線部分およびパターンまで拡張される。１実施形態では、ストリング（例えば、行、列、対角線等）の長さはストリングが存在するプリミティブの大きさよりも小さいようにそれぞれ限定される。

【０２１１】

図３２は図２７の動作2702に関連したプロセスに関連したプロセスを示すフローチャートである。瞬間的なプロセスは目の後に存在する部分でプリミティブを処理するように設計されている。これらの域外の部分はその次のラスター化動作で問題を生じる。これを実現するため、瞬間的なプロセスは変数Ｗを使用し、これは投影、即ち遠近法でオブジェクトを観察するために共通して使用される。変数Ｗは他の座標Ｘ、Ｙ、Ｚが近くのものを大きく、遠くのものを小さくするために割算される数字である。変数Ｗは投影の中心と、対応する頂点との間の距離を表す。

【０２１２】

図３２で示されているように、プリミティブは最初に受信され、複数の頂点により規定される。そのような各頂点はＷ値を含んでいる。プリミティブを受信するとき、設定モジュールは頂点に基づいてプリミティブを特徴付けするラインを規定する役目を行う。動作3200に注意。

【０２１３】

Ｗ値はその後、決定3202で解析される。示されているように、１つのＷ－値が負であるならば、負の値を有する頂点と反対のラインの一次方程式は動作3204でフリップされる。換言すると、一次方程式の係数は－１により乗算される。さらに、２つのＷ－値が負であるならば、正のＷ－値を有する頂点と、負のＷ－値を有するそれぞれの頂点とを接続するラインの一次方程式は動作3206でフリップされる。３つのＷ－値が負であるならば、不合格（カル）状態3207が生じ、本発明は三角形を不合格とする。負であるＷ－値がないならば、付加的な措置は取られない。

【０２１４】

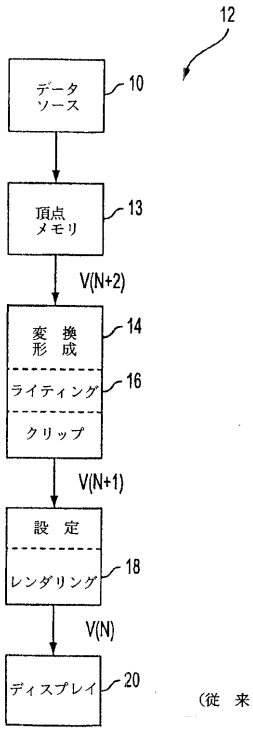
図３２Ａ－３２Ｃはフリップ一次方程式が、処理されるスクリーンの部分に影響を与える方法を示している。図３２ＡはＷ値が負のものではなく、一次方程式が変更されない状態の場合を示している。示されているように、プリミティブの内部部分はこのようなケース

で満たされている。
【０２１５】
図３２Ｂは１つのＷ－値が負であり、したがってその一次方程式がフリップされるケースを示している。示されているように、頂点と対向するプリミティブ部分は現在のケースで満たされている。特に、描かれるエリアは－Ｗ頂点を共有する２つの三角形の面と共直線性である２つのラインにより境界を与えられ、さらに、２つの＋Ｗ頂点を共有する三角形の面により境界を与えられる。
【０２１６】
図３２Ｃは２つのＷ－値が負であり、したがってその一次方程式がフリップされるケースを示している。示されているように、頂点と対向するプリミティブ部分は図２７－３２を参照して前述した方法および／またはプロセスを使用して満たされる。換言すると、描かれるエリアは＋Ｗ頂点を共有する２つの三角形の面と共直線性である２つのラインにより境界を与えられ、さらに、＋Ｗ頂点に近接する。
【０２１７】
本発明はしたがって全ての３つの前述のケースを処理することができる。三角形の部分が近距離および／遠距離の平面を越えているならば、これらの平面内にその部分だけを描く。三角形が１または２の負のＺ頂点を有するならば、正確な＋Ｚ部分だけが描かれる。
【０２１８】
全ての頂点がオフスクリーンであり、三角形が目の後方から遠距離の平面を越えて延在しても、画素は三角形内およびスクリーン上であり、近限界と遠限界との間にＺを有する。本発明は悪い画素を使用する時間の浪費を少なくすることを確実にする。スクリーンエッジまたは近距離平面と遠距離平面による全てのクリッピングが容易に使用されることができる凸形領域のオンスクリーンで常に行われるので、これは可能である。
【０２１９】
スタート点が満たされるエリア内ではないときに時によって問題が生じる。上部頂点がオフスクリーンであるか近距離の平面または遠距離の平面によりクリップされる場合にこれは生じる。この場合、トラバーサルステージは描かれる区域の上部点を検索しなければならず、上から開始する。これは三角形のエッジスロープとＺスロープの符号により誘導されることにより効率的に行われる。これは三角形の一次方程式が描かれる領域外であることとその理由を発見するために、三角形の一次方程式を試験できる。外部にあるエッジおよび／またはＺ限界を知ったとき、そのエッジまたは限界へ近付けるステップ方向を知る。（選択肢のあるとき）好みによって水平から垂直に移動することによって、描かれた領域の検索は上部に描くことのできる画素が存在するならば、それを発見する。オープンアップする外部（－Ｗ）三角形でもこの問題は生じる。この場合、描かれる区域は全ての３つの頂点よりも上方に延在する。
【０２２０】
本発明の１実施形態ではトラバーサルは三角形の上部から下部へ進行する。負のＷ－値をもたず頂点がシザー長方形であるならば、スタート点は三角形の上部の頂点である。トラバーサルは常にシザー長方形内で開始し、その外ではないので、エッジにより囲まれるエリアがシザー長方形を越えて延在しても、シザー長方形内の三角形部分だけが描かれる。このようにして、簡単なシザー長方形の長方形エッジクリッピングが行われる。
【０２２１】
種々の実施形態を前述したが、これらは技術的範囲の限定ではなく例示でのみ示されていることを理解すべきである。したがって、本発明の技術的範囲は前述の例示的な実施形態により限定されず、特許請求の範囲とそれらの均等物にしたがってのみ限定される。
【図面の簡単な説明】
【図１】　従来技術のグラフィック処理方法のフロー図。
【図１Ａ】　従来技術のスキニング方法を示す概略図。
【図１Ｂ】　単一の半導体プラットフォーム上に構成された本発明の１実施形態の種々のコンポーネントを示すフロー図。

【図２】　本発明の１実施形態による頂点属性バッファ（ＶＡＢ）の概略図。
【図２Ａ】　本発明の１実施形態によるＶＡＢによって受取られることのできる種々のコマンドを示すチャート。
【図２Ｂ】　本発明の１実施形態によるＶＡＢとの間で頂点属性をロードし、ドレインする方法を示すフローチャート。
【図２Ｃ】　図２Ｂの動作を実行するために使用される本発明のアーキテクチャの概略図。
【図３】　本発明の１実施形態によるＶＡＢに関連したモードビットの説明図。
【図４】　本発明の変換モジュールを示す概略図。
【図４Ａ】　本発明の１実施形態による多数の実行スレッドを実行する方法を示すフローチャート。
【図４Ｂ】　本発明の１実施形態にしたがって図４Ａの方法が行われる手順を示すフロー図。
【図５】　本発明の１実施形態による図４の変換モジュールの機能装置の概略図。
【図６】　図５の変換モジュールの乗算論理装置（ＭＬＵ）の概略図。
【図７】　図５の変換モジュールの演算論理装置（ＡＬＵ）の概略図。
【図８】　図５の変換モジュールのレジスタファイルの概略図。
【図９】　図５の変換モジュールの反転論理装置（ＩＬＵ）の概略図。
【図１０】　本発明の１実施形態による図５の変換モジュールの出力コンバータの出力アドレスのチャート。
【図１１】　本発明の１実施形態による図５の変換モジュールのマイクロコード編成図。
【図１２】　本発明の１実施形態による図５の変換モジュールのシーケンサの概略図。
【図１３】　図１２の変換モジュールのシーケンサの使用に関連した種々の動作を詳細に示すフローチャート。
【図１４】　図１２の変換モジュールのシーケンサのシーケンシングコンポーネントの動作を詳細に示すフロー図。
【図１４Ａ】　グラフィック処理中にスカラーおよびベクトル成分を処理するために使用される本発明のコンポーネントを示すフロー図。
【図１４Ｂ】　図５の変換モジュールに対応している図１４Ａに示されている本発明の機能コンポーネントの１つの可能な組合せ1451を示すフロー図。
【図１４Ｃ】　図１４Ａに示されている本発明の機能コンポーネントの別の可能な組合せ1453を示すフロー図。
【図１４Ｄ】　本発明の１実施形態にしたがって図１２の変換モジュールにより実施されるグラフィック処理中にブレンディング動作を行う方法を示すフロー図。
【図１５】　本発明の１実施形態のライティングモジュールの概略図。
【図１６】　本発明の１実施形態による図１５のライティングモジュールの機能装置を示す概略図。
【図１７】　本発明の１実施形態による図１６のライティングモジュールの乗算論理装置（ＭＬＵ）の概略図。
【図１８】　本発明の１実施形態による図１６のライティングモジュールの演算論理装置（ＡＬＵ）の概略図。
【図１９】　本発明の１実施形態による図１６のライティングモジュールのレジスタ装置の概略図。
【図２０】　本発明の１実施形態による図１６のライティングモジュールのライティング論理装置（ＬＬＵ）の概略図。
【図２１】　本発明の１実施形態による図１６のライティングモジュールに関連したフラッグレジスタの説明図。
【図２２】　本発明の１実施形態による図１６のライティングモジュールに関連したマイクロコードフィールドの説明図。
【図２３】　本発明の１実施形態による図１６のライティングモジュールに関連したシー

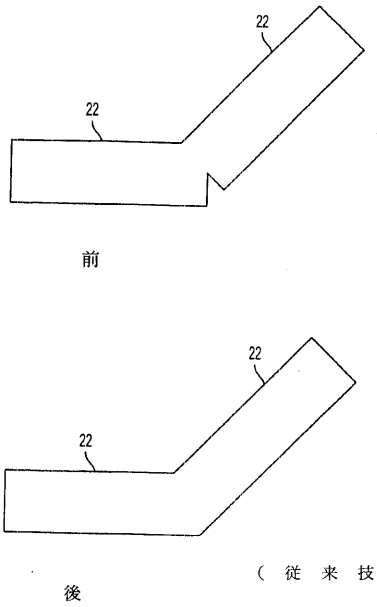
ケンサの概略図。
【図２４】　本発明の１実施形態にしたがって変換およびライティングモジュールのシーケンサが関連したバッファの入力および出力をどのように制御することができるかを詳細に説明するフローチャート。
【図２５】　図２４の方法にしたがって変換およびライティングモジュールのシーケンサが関連したバッファの入力および出力をどのように制御することができるかを示す概略図。
【図２５Ｂ】　図１Ｂのラスター化装置の種々のモジュールの概略図。
【図２６】　本発明のラスター化モジュールの設定モジュールの概略図。
【図２６Ａ】　図２６のラスター化装置の設定モジュールによって計算される種々のパラメータを示す説明図。
【図２７】　図２６に示されているラスター化装置コンポーネントの設定およびトラバーサルモジュールに関連した本発明の方法を示すフローチャート。
【図２７Ａ】　本発明の１実施形態にしたがってプリミティブにおけるエリアを識別するために移動される凸状領域を囲む方向ポイントを示す説明図。
【図２８】　図２７の処理行動作2706に関連している本発明のプロセスを示すフローチャート。
【図２８Ａ】　本発明の凸状領域がプリミティブに関して移動されるシーケンスを示す概略図。
【図２８Ｂ】　本発明の凸状領域がプリミティブに関して移動されるシーケンスの別の例を示す概略図。
【図２９】　図２７の処理行動作2706に関連した本発明のプロセスの別の犂耕体プロセスを示すフローチャート。
【図２９Ａ】　図２９の犂耕体プロセスにしたがって本発明の凸状領域がプリミティブに関して移動されるシーケンスを示す概略図。
【図３０】　境界を使用する別の犂耕体プロセスを示すフローチャート。
【図３１】　図３０の動作3006に関連したプロセスを示すフローチャート。
【図３１Ａ】　図３０および３１の境界ベースの犂耕体プロセスにしたがって本発明の凸状領域がプリミティブに関して移動されるシーケンスを示す概略図。
【図３２】　図２７の動作2702に関連したプロセスを示すフローチャート。
【図３２Ａ】　図３２のプロセスにおいて負のＷ値が１つも計算されないとき、どのようなエリアが描かれるかを示す説明図。
【図３２Ｂ】　図３２のプロセスにおいて負のＷ値が１つだけ計算されたとき、どのようなエリアが描かれるかを示す説明図。
【図３２Ｃ】　図３２のプロセスにおいて負のＷ値が２つだけ計算されたとき、どのようなエリアが描かれるかを示す説明図。

【図１】



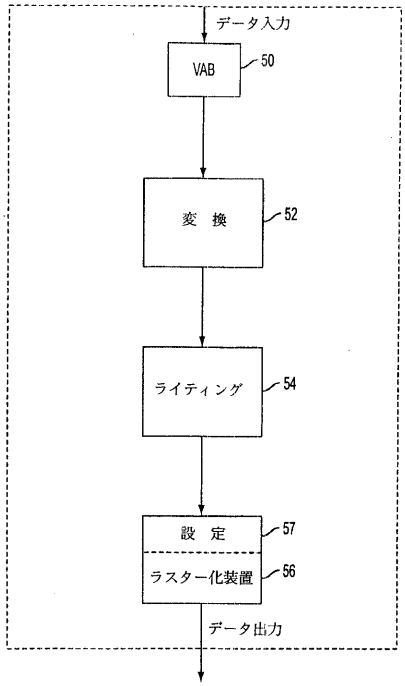
（従来技術）

【図１Ａ】

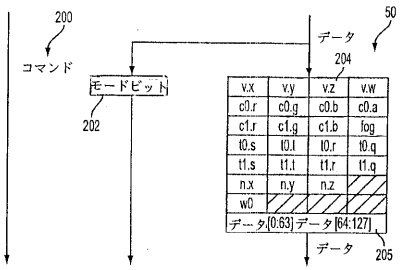


（従来技術）

【図１Ｂ】



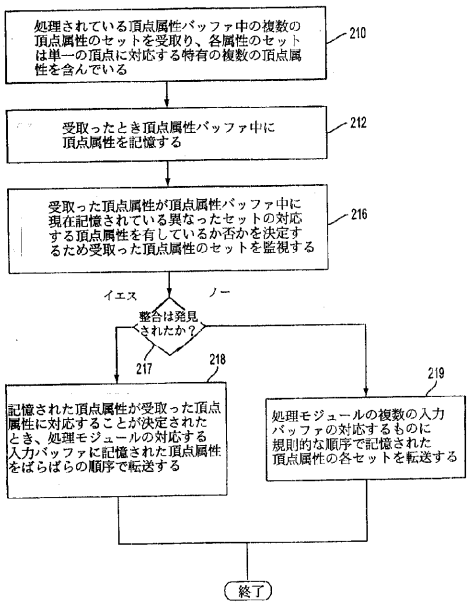
【図２】



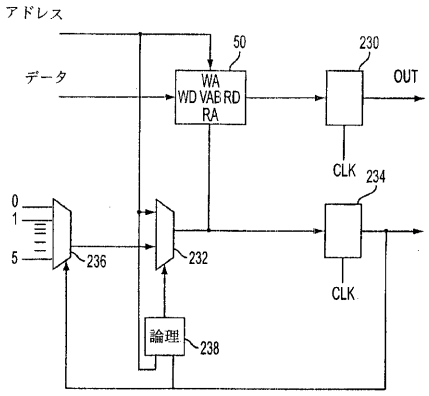
【図２Ａ】

コマンド	変換機能停止	ライティング機能停止	記述
FE2XF_CMD_NOP			NO OPERATION. CAN BE USED AS A SPACER BETWEEN COMMANDS.
FE2XF_CMD_VERTEX	READ	READ	VERTEX DATA.
FE2XF_CMD_PASSTHR			PASSTHROUGH. TRANSFORM AND LIGHTING PASS THE DATA THROUGH.
FE2XF_CMD_RDVAB			READ THE VAB CONTENTS WHEN CONTEXT SWITCHING.
FE2XF_CMD_LDMODE			LOAD NEW MODE BITS.
FE2XF_CMD_LDXFCTX	WRITE		LOAD TRANSFORM CONTEXT MEMORY DATA.
FE2XF_CMD_RDXFCTX	READ		READ TRANSFORM CONTEXT MEMORY DATA.
FE2XF_CMD_LDLTCTX		WRITE	LOAD LIGHTING CONTEXT MEMORY DATA.
FE2XF_CMD_RDLTCTX		READ	READ LIGHTING CONTEXT MEMORY DATA.
FE2XF_CMD_LDLTC0		WRITE	LOAD LIGHTING CONTEXT0 MEMORY DATA.
FE2XF_CMD_RDLTC0		READ	READ LIGHTING CONTEXT0 MEMORY DATA.
FE2XF_CMD_LDLTC1		WRITE	LOAD LIGHTING CONTEXT1 MEMORY DATA.
FE2XF_CMD_RDLTC1		READ	READ LIGHTING CONTEXT1 MEMORY DATA.
FE2XF_CMD_LDLTC2		WRITE	LOAD LIGHTING CONTEXT2 MEMORY DATA.
FE2XF_CMD_RDLTC2		READ	READ LIGHTING CONTEXT2 MEMORY DATA.
FE2XF_CMD_LTLTC3		WRITE	LOAD LIGHTING CONTEXT3 MEMORY DATA.
FE2XF_CMD_RDLTC3		READ	READ LIGHTING CONTEXT3 MEMORY DATA.
FE2XF_CMD_SYNC	READ+WRITE	READ+WRITE	SIMILAR TO NOP, BUT IS NOT ALLOWED TO BE PROCESSED IN PARALLEL.

【図２Ｂ】



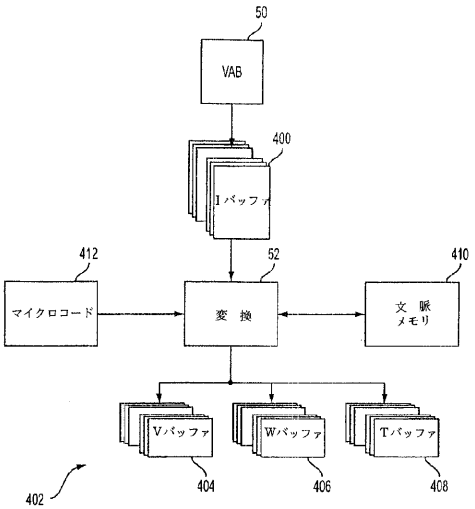
【図２Ｃ】



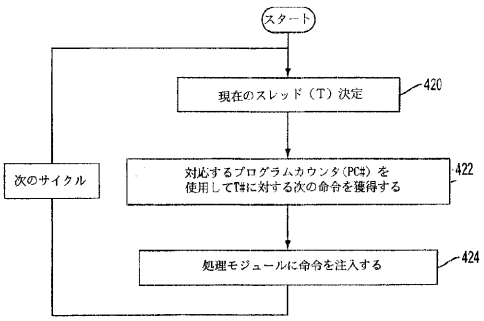
【図３】

モードビット	ビット	記述
T0	1	TEXTURE 0 ENABLE
TXF0	1	TEXTURE 0 MATRIX TRANSFORM ENABLE
TDV0	1	TEXTURE 0 w DIVIDE ENABLE
T0S	3	TEXTURE 0 TEXGEN s CONTROL
T0T	3	TEXTURE 0 TEXGEN t CONTROL
T0U	3	TEXTURE 0 TEXGEN r CONTROL
T0Q	2	TEXTURE 0 TEXGEN q CONTROL
T1	1	TEXTURE 1 ENABLE
TXF1	1	TEXTURE 1 MATRIX TRANSFORM ENABLE
TDV1	1	TEXTURE 1 w DIVIDE ENABLE
T1S	3	TEXTURE 1 TEXGEN s CONTROL
T1T	3	TEXTURE 1 TEXGEN t CONTROL
T1U	3	TEXTURE 1 TEXGEN r CONTROL
T1Q	2	TEXTURE 1 TEXGEN q CONTROL
ETY	1	EYE TYPE INFINITE(0) OR LOCAL(1)
LIT	1	LIGHTING ENABLE
NRM	1	NORMAL NORMALIZE ENABLE
FOG	1	FOG ENABLE
LIS	16	LIGHT STATUS (8 LIGHTS BY 2 BITS EACH, 0:OFF,1:INFINITE,2:LOCAL,3:SPOTLIGHT)
FG	2	FOGGEN CONTROL(0: OFF, 1:RADIAL, 2: PLANE)
LAT	1	LIGHT ATTENUATION CONTROL (0: INVERT, 1: NO INVERT)
C1I	1	SPECULAR COLOR INPUT ENABLE
C1O	1	SPECULAR COLOR OUTPUT ENABLE
CM	4	COLOR MATERIAL CONTROL (1: EMISSIVE, 2: AMBIENT, 4: DIFFUSE, 8:SPECULAR)
PP	1	POINT PARAMETER ENABLE
SKIN	1	SKINNING ENABLE
VPAS	1	VERTEX PASS ENABLE

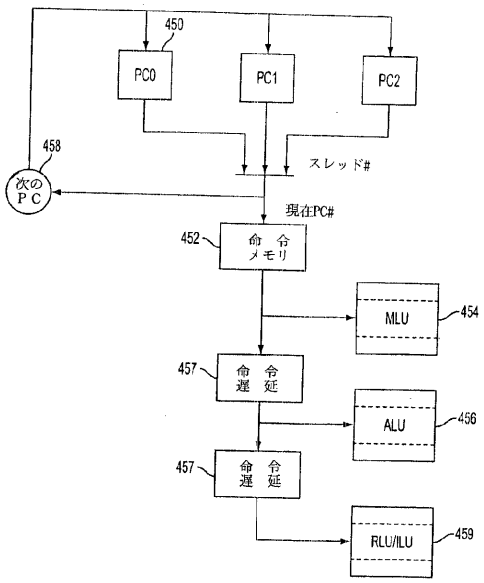
【図４】



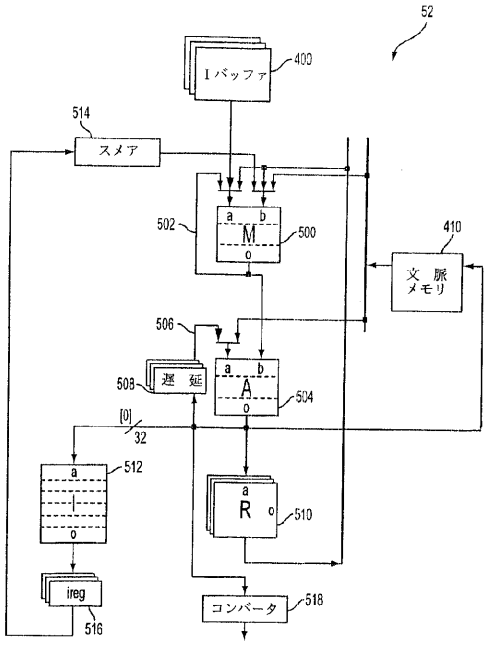
【図４Ａ】



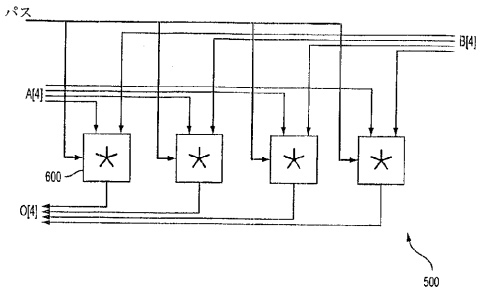
【図４Ｂ】



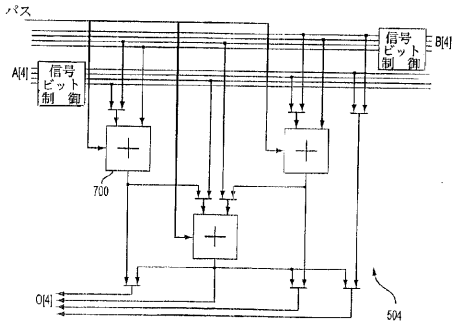
【図５】



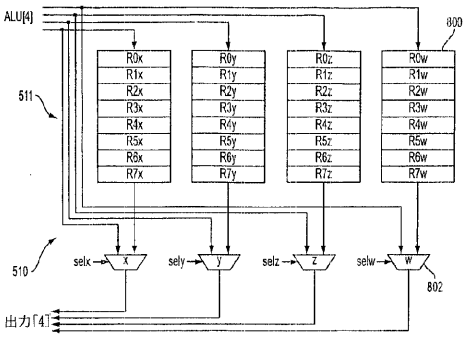
【図６】



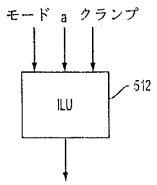
【図７】



【図８】



【図９】



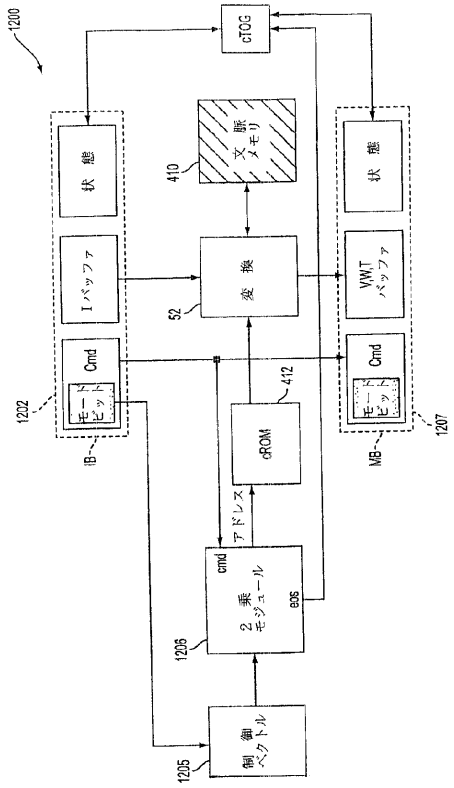
【図１０】

アドレス	ターゲット	作用	記述
TPOS	TBUFFER	T[0] = ALU	POSITION
TT0	TBUFFER	T[3] = ALU	TEXTURE0
TT1	TBUFFER	T[4] = ALU	TEXTURE1
WEV	WBUFFER,VBUFFER	W[0] = ALU, V[0].y = ALU.w	EYE VECTOR
WLV0	WBUFFER,VBUFFER	W[1] = ALU, V[1].y = ALU.w	LIGHT0 DIRECTION VECTOR
WLV1	WBUFFER,VBUFFER	W[2] = ALU, V[2].y = ALU.w	LIGHT1 DIRECTION VECTOR
WLV2	WBUFFER,VBUFFER	W[3] = ALU, V[3].y = ALU.w	LIGHT2 DIRECTION VECTOR
WLV3	WBUFFER,VBUFFER	W[4] = ALU, V[4].y = ALU.w	LIGHT3 DIRECTION VECTOR
WLV4	WBUFFER,VBUFFER	W[5] = ALU, V[5].y = ALU.w	LIGHT4 DIRECTION VECTOR
WLV5	WBUFFER,VBUFFER	W[6] = ALU, V[6].y = ALU.w	LIGHT5 DIRECTION VECTOR
WLV6	WBUFFER,VBUFFER	W[7] = ALU, V[7].y = ALU.w	LIGHT6 DIRECTION VECTOR
WLV7	WBUFFER,VBUFFER	W[8] = ALU, V[8].y = ALU.w	LIGHT7 DIRECTION VECTOR
WSL0	WBUFFER	W[9] = ALU	SPOTLIGHT0 cos
WSL1	WBUFFER	W[10] = ALU	SPOTLIGHT1 cos
WSL2	WBUFFER	W[11] = ALU	SPOTLIGHT2 cos
WSL3	WBUFFER	W[12] = ALU	SPOTLIGHT3 cos
WSL4	WBUFFER	W[13] = ALU	SPOTLIGHT4 cos
WSL5	WBUFFER	W[14] = ALU	SPOTLIGHT5 cos
WSL6	WBUFFER	W[15] = ALU	SPOTLIGHT6 cos
WSL7	WBUFFER	W[16] = ALU	SPOTLIGHT7 cos
VED	VBUFFER	V[0].x = 1.0, V[0].z = ALU.w	EYE RADIAL DISTANCE VECTOR
VLD0	VBUFFER	V[1].x = 1.0, V[1].z = ALU.w	LIGHT0 DISTANCE VECTOR
VLD1	VBUFFER	V[2].x = 1.0, V[2].z = ALU.w	LIGHT1 DISTANCE VECTOR
VLD2	VBUFFER	V[3].x = 1.0, V[3].z = ALU.w	LIGHT2 DISTANCE VECTOR
VLD3	VBUFFER	V[4].x = 1.0, V[4].z = ALU.w	LIGHT3 DISTANCE VECTOR
VLD4	VBUFFER	V[5].x = 1.0, V[5].z = ALU.w	LIGHT4 DISTANCE VECTOR
VLD5	VBUFFER	V[6].x = 1.0, V[6].z = ALU.w	LIGHT5 DISTANCE VECTOR
VLD6	VBUFFER	V[7].x = 1.0, V[7].z = ALU.w	LIGHT6 DISTANCE VECTOR
VLD7	VBUFFER	V[8].x = 1.0, V[8].z = ALU.w	LIGHT7 DISTANCE VECTOR
VC0	VBUFFER,TBUFFER	V[9] = ALU, T[1] = ALU	DIFFUSE COLOR
VC1	VBUFFER,TBUFFER	V[10] = ALU, T[2] = ALU	SPECULAR COLOR
VNRM	VBUFFER	V[11] = ALU	NORMAL VECTOR
VED2	VBUFFER	V[12] = ALU	EYE PLANAR DISTANCE VECTOR
TVW_NOP			NO VALID OUTPUT.

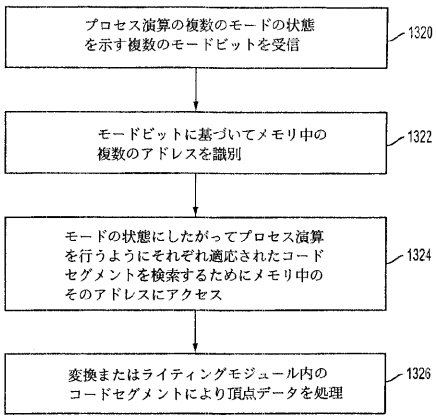
【図１１】

マイクロコードフィールド	ビット位置	遅延	記述
oa	6	0: 5	2	OUTPUT BUFFER WRITE ADDRESS
rra	3	6: 8	0	RLU READ ADDRESS
rwm	4	9:12	2	RLU WRITE MASK
rwa	3	13:15	2	RLU WRITE ADDRESS
ilu	2	16:17	2	ILU OPERATION
alu	4	18:21	1	ALU OPERATION
ais	2	22:23	1	ALU SIGN CONTROL
aia	1	24	1	ALU INPUT A MUX
mlu	3	25:27	0	MLU OPERATION
mib	2	28:29	0	MLU INPUT B MUX
mia	2	30:31	0	MLU INPUT A MUX
va	3	32:34	0	INPUT BUFFER READ ADDRESS
ce	1	35	0,2	CONTEXT MEMORY READ/WRITE
ca	6	36:41	0,2	CONTEXT MEMORY ADDRESS
mr	2	42:43	0	MLU INPUT VECTOR ROTATE

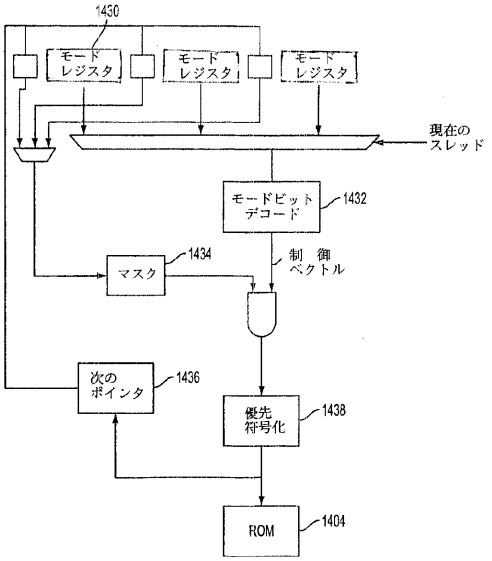
【図１２】



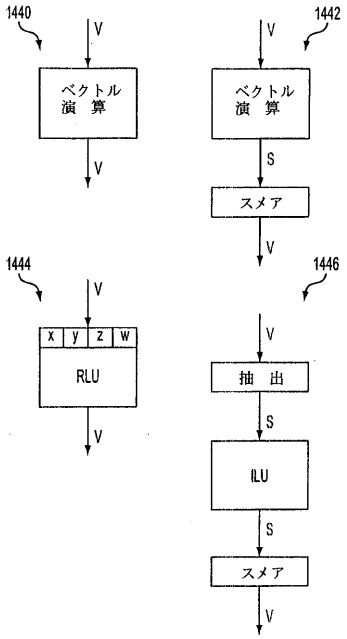
【図１３】



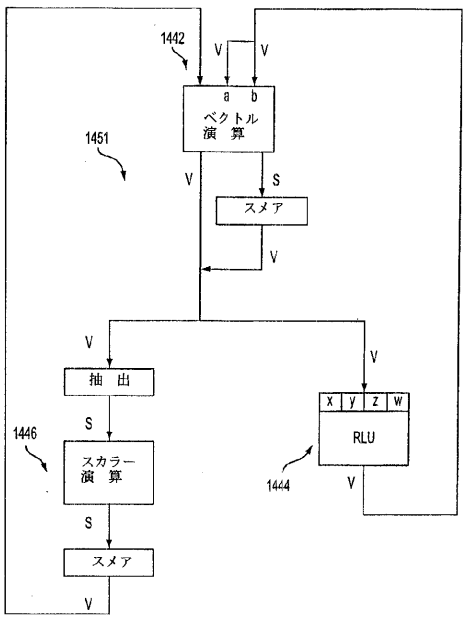
【図１４】



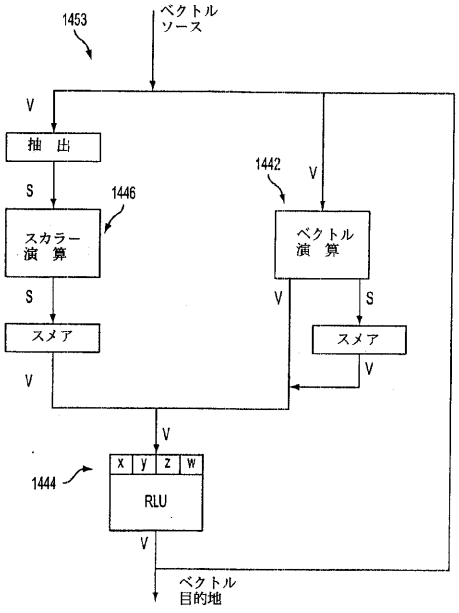
【図１４Ａ】



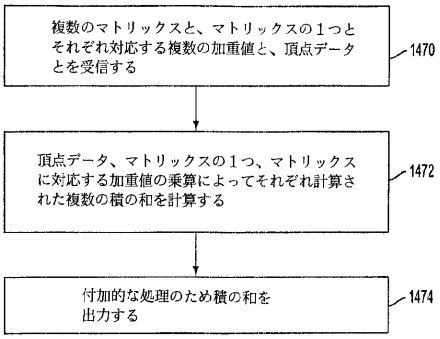
【図１４Ｂ】



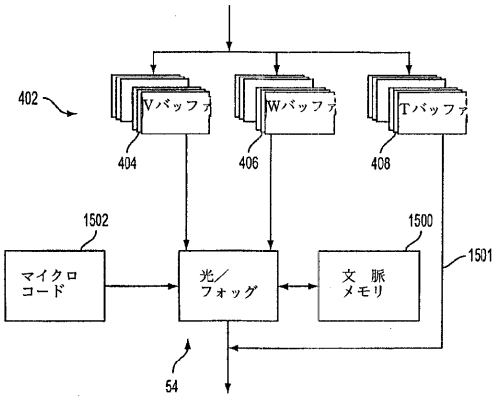
【図１４Ｃ】



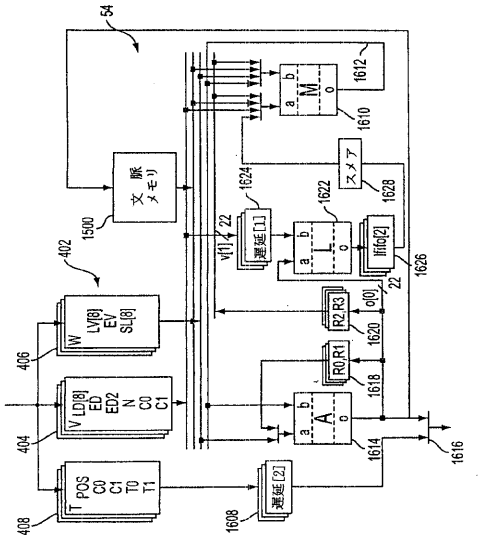
【図１４Ｄ】



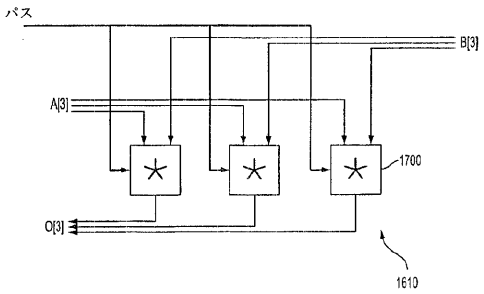
【図１５】



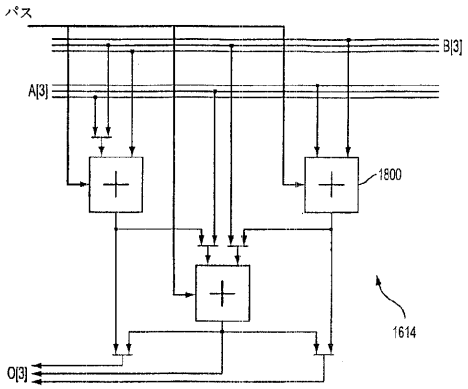
【図１６】



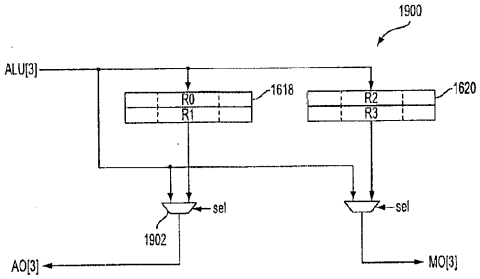
【図１７】



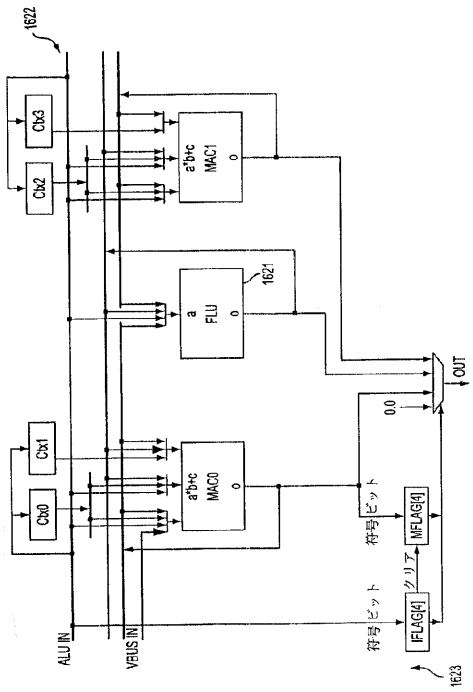
【図１８】



【図１９】



【図２０】



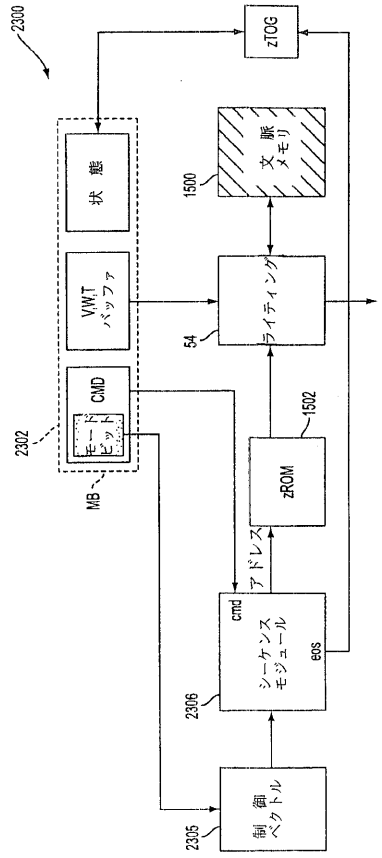
【図２１】

名称	レジスタ	記述
Z	IFLAG	CLEAR FLAG. SETS IFLAG AND MFLAG TO 0.
C	IFLAG	SPOTLIGHT CONE FLAG. SET IF VERTEX IS OUTSIDE SPOTLIGHT CONE.
S	IFLAG	SPECULAR2 FLAG. SET IF SPECULAR CONTRIBUTION IS NEGATIVE.
D	IFLAG	DIFFUSE FLAG. SET IF DIFFUSE TERM IS NEGATIVE.
	MFLAG	
U	MFLAG	SPOTLIGHT CONE ATTENUATION FLAG. SET IF SPOTLIGHT CONE ATTENUATION CONTRIBUTION IS NEGATIVE.
T	MFLAG	SPECULAR FLAG. SET IF SPECULAR CONTRIBUTION IS NEGATIVE.
R	MFLAG	RANGE FLAG. SET IF VERTEX IS TOO FAR AWAY FROM THE LIGHT.

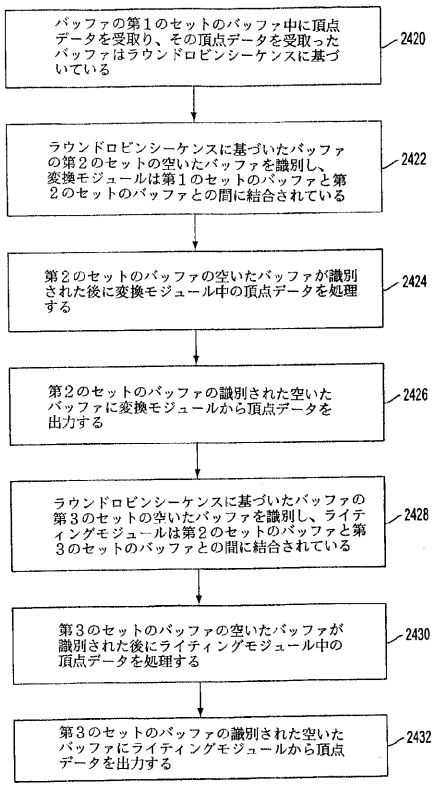
【図２２】

マイクロコードフィールド	ビット	位置	遅延	記述
oa	3	0:2	2	OUTPUT ADDRESS
rwe	1	3	2	RLU WRITE ENABLE
rwa	2	4:5	2	RLU WRITE ADDRESS
R23	1	6	0	RLU (MLU) READ ADDRESS
R01	1	7	1	RLU (ALU) READ ADDRESS
aia	1	8	1	ALU INPUT A MUX
alu	2	9:10	1	ALU OPERATION
mib	2	11:12	0	MLU INPUT B MUX
mia	2	13:14	0	MLU INPUT A MUX
mlu	2	15:16	0	MLU OPERATION
mwa	5	17:21	0	MLU WBUFFER READ ADDRESS
awa	5	22:26	1	ALU WBUFFER READ ADDRESS
va	4	27:30	0	VBUFFER READ ADDRESS
os	2	31:32	2	LLU OUTPUT ADDRESS
fm	6	33:38	2	FLAG REGISTER MASK
mfe	1	39	2	MFLAG WRITE ENABLE
mfa	2	40:41	2	MFLAG WRITE ADDRESS
ife	1	42	2	IFLAG WRITE ENABLE
ifa	2	43:44	2	IFLAG WRITE ADDRESS
fia	2	45:46	2	FLU INPUT A MUX
flu	3	47:49	2	FLU OPERATION
M1c	1	50	2	MAC1 INPUT C MUX
M1b	2	51:52	2	MAC1 INPUT B MUX
M1a	2	53:54	2	MAC1 INPUT A MUX
M0c	2	55:56	2	MAC0 INPUT C MUX
M0b	2	57:58	2	MAC0 INPUT B MUX
M0a	2	59:60	2	MAC0 INPUT A MUX
ce	3	61:63	0,2	CONTEXT MEMORY READ/WRITE ENABLE
ca	6	64:69	0,2	CONTEXT MEMORY ADDRESS
C3a	4	70:73	2	CONTEXT3 MEMORY ADDRESS
C2a	4	74:77	2	CONTEXT2 MEMORY ADDRESS
C1a	5	78:82	2	CONTEXT1 MEMORY ADDRESS
C0a	2	83:84	2	CONTEXT0 MEMORY ADDRESS

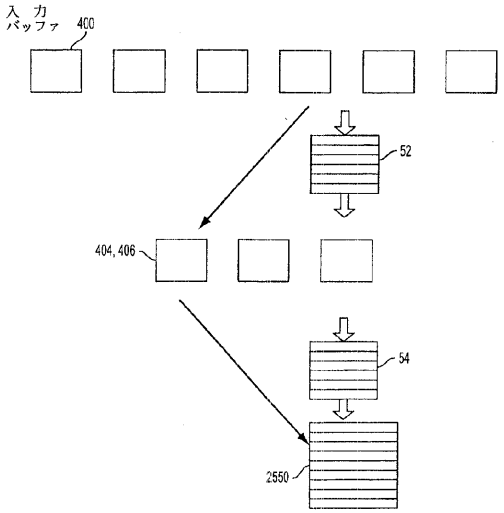
【図２３】



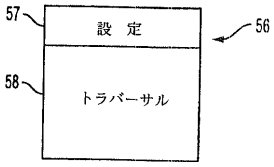
【図２４】



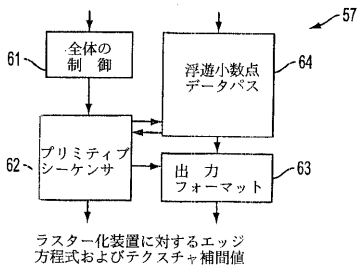
【図２５】



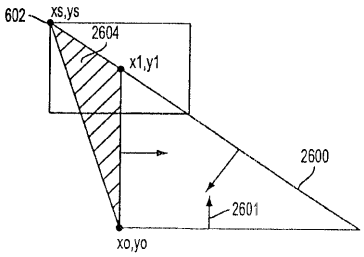
【図２５Ｂ】



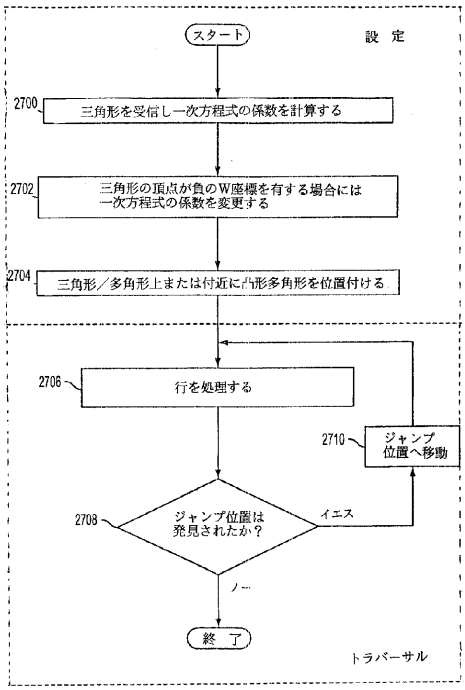
【図２６】



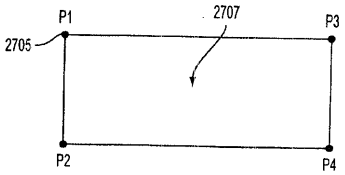
【図２６Ａ】



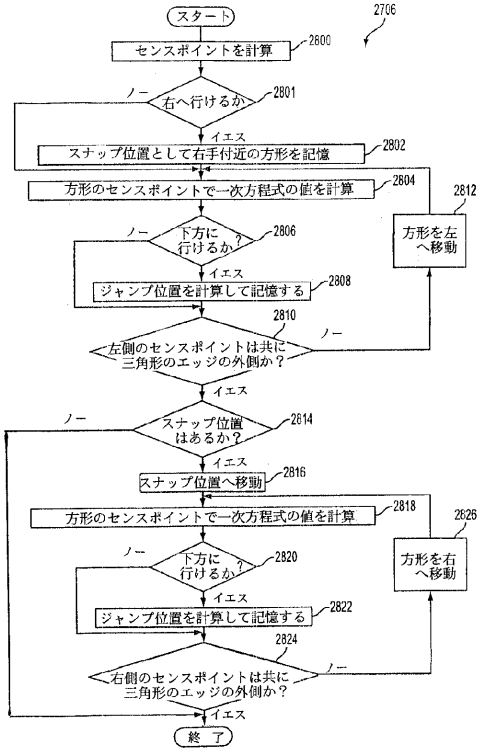
【図２７】



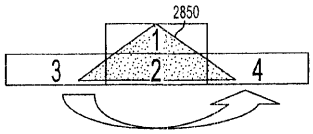
【図２７Ａ】



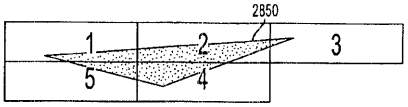
【図２８】



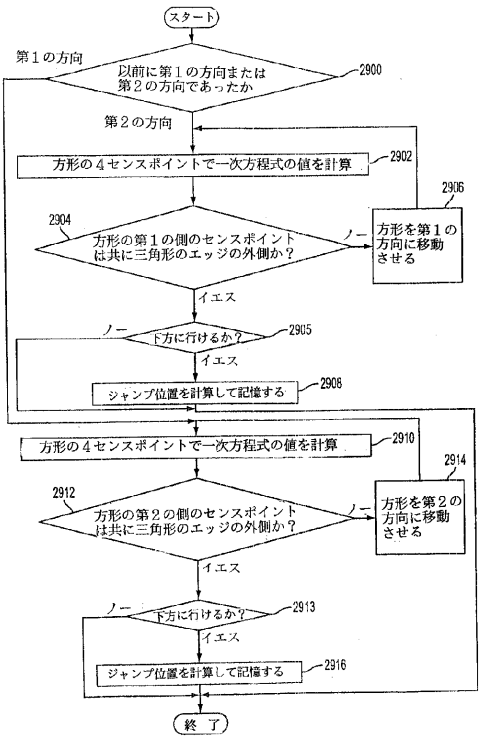
【図２８Ａ】



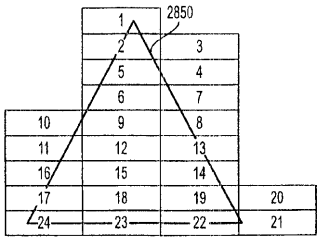
【図２８Ｂ】



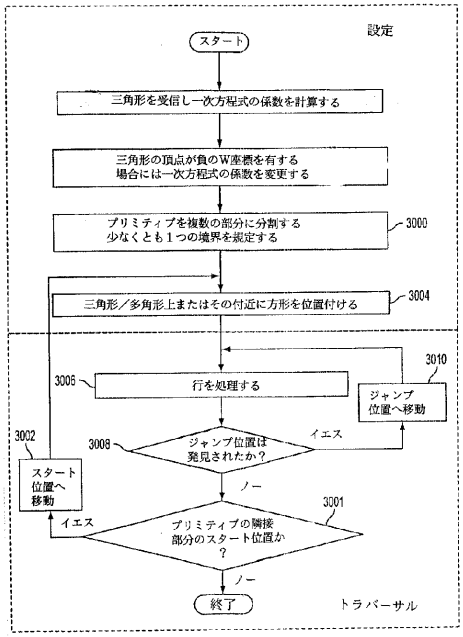
【図２９】



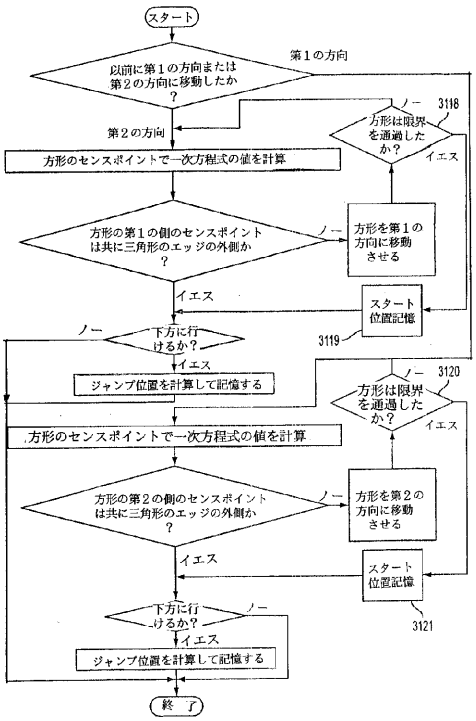
【図２９Ａ】



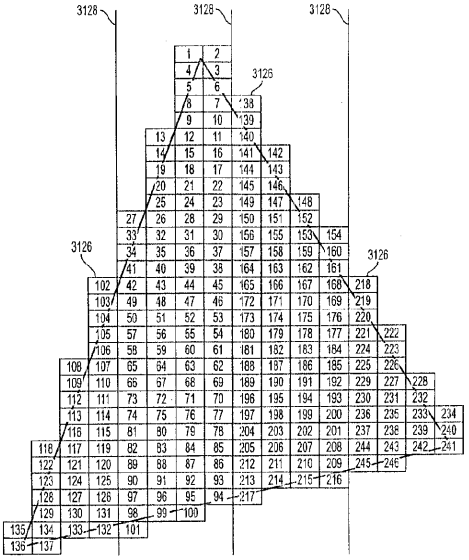
【図３０】



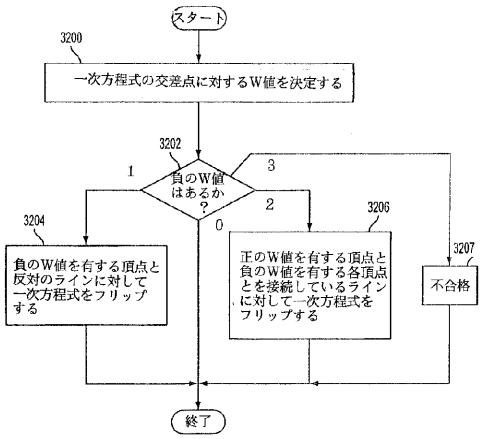
【図３１】



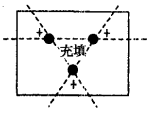
【図３１Ａ】



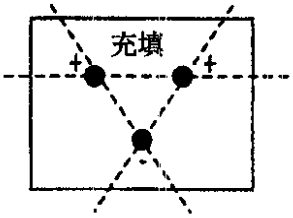
【図３２】



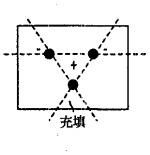
【図３２Ａ】



【図３２Ｂ】



【図３２Ｃ】



フロントページの続き

(72)発明者　モイ、サイモン
アメリカ合衆国、カリフォルニア州　９４０４０　マウンテン・ビュー、フリーダム・レーン　１００
(72)発明者　カーク、デビッド
アメリカ合衆国、カリフォルニア州　９４１２３　サン・フランシスコ、ブロデリック・ストリート　２９６５
(72)発明者　サベラ、パオロ
アメリカ合衆国、カリフォルニア州　９４５８８　プレザントン、カンバーランド・ギャップ・コート　３４５７

審査官　伊知地　和之

(56)参考文献　特開平０７－０２１１５５（ＪＰ，Ａ）
特開平０３－１２７１８８（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06T 11/00 - 19/20
G06T 1/20