グラフィクスとCAD 43－7
（1990．2．23）

グラフィックスディスプレイのベクトル描画の等速性

上田智章、長島健二、西口隆也、加井隆重、牧野　寛
ダイキン工業株式会社
電子技術研究所

あらまし

グラフィックスディスプレイの描画には、ベクトルの描画と多角形の塗り潰しがある。これらをハードウェアで高速に描画するには、異なったアーキテクチャが必要であり、この実現の為にピクセルバッファの設計が有効である。ピクセルバッファの形状が、描画性能に大きく関わることを示し、多角形塗り潰しだけでなくランダム方向にベクトル描画を高速化するため、通常と異なり、ピクセルバッファを垂直方向に拡張する方法を提案する。フレームメモリをダブルバッファ化し、多チャンネルのインタリーブを構成する事によりピクセルバッファを拡大し、専用ゲートアレイを２種類開発し、任意方向に300,000ﾍﾞｸﾀｰ/秒の性能を実現した。

Random vector drawing at a uniform rate in the graphic display.

Tomoaki Ueda,Kenji Nagashima,Takaya Nishiguchi,Takashige Kai,Hiroshi Makino

Electronic Engineering Laboratory
DAIKIN INDUSTRIES.LTD.

1000-2 Ohtani,Okamoto-Cho,Kusatsu Shiga 525,Japan

Abstract

The graphic display provides with vector drawings and polygon fillings. We show that design of pixel buffering is an effective means for high speed vector drawings and polygon fillings. We propose a new construction method of pixel buffers. The pixel buffers are double buffered and interleavingly controlled. We developed two Gate-Arrays for the pixel buffers and the memory address controller. The vector drawing speed of 300,000 vectors/sec was realized in all directions.

1．まえがき

ラスターグラフィックスディスプレイは、高速に図形データを表示するための専用ハードウェアを持っており、そのハードウェアによって表示速度が大きく変化する。グラフィックスディスプレイ内での主な処理は、通常以下に挙げる処理が行なわれ、どの部分をハードウェア化するかでマシンの性格付けが行なわれる。

①複雑なプリミティブの発生
②座標変換
③光源計算*
④座標クリップ
⑤スキャンコンバージョン*
⑥勾配演算
⑦累積加算
⑧Z値比較（隠面処理）
⑨フレームデータ更新
（*：多角形描画時のみ必要）

これらの処理は、図1に示すグラフィックスパイプラインと言われるハードウェアで構成されるのが一般的である。以下では、グラフィックパイプラインを通じて処理が行なわれるものとして話を進める。

図形データは、直線、多角形、円、球等のプリミティブで定義され、拡大縮小、回転等の座標変換が行なわれ、各頂点での色を決定するための光源計算と輝度計算が行なわれる。更に、ＣＲＴに表示する部分だけを切りとる座標クリップが行なわれ、多角形であれば、スキャンラインに分解するスキャンコンバージョン処理がなされる。その後、カラー値、座標値の補間のための勾配演算が行なわれ、ピクセル単位にデータが生成され、各ピクセルに対応するＺ値が比較されて必要に応じて更新される[1]。

描画速度については、各パイプラインステージでの処理時間のバランスが重要であり、描画速度を決定する要因として大きなものは、

- グラフィックスパイプラインでの座標変換処理速度
- 微分解析器（Digital Differential Analyzer 以後、ＤＤＡと略す）のピクセルデータの生成速度

が挙げられる。

グラフィックスディスプレイの描画には、ベクトルの描画と多角形の塗り潰しがある。これらを高速に描画するハードウェアとして、ピクセルバッファを用いるが、ベクトルは、任意方向に描画しなければならないし、角形はスキャンコンバージョン後、水平方向のみの描画となるため、その形状が描画速度に影響する。

更に、ベクトル描画においては、カタログスペックと使用した際の印象から受ける実用スペックとの違いといった問題がある。これは、一般にはベクトル描画において、描画するデータにより描画速度が変化する事が原因として知られている[2]。この点につき、ハードウェアの制約から描画速度に影響を与えることを示し、描画データに依存せずランダム方向ベクトル描画において、等速性を保証できるハードウェアの方式につき提案する。図1に、各パイプライ

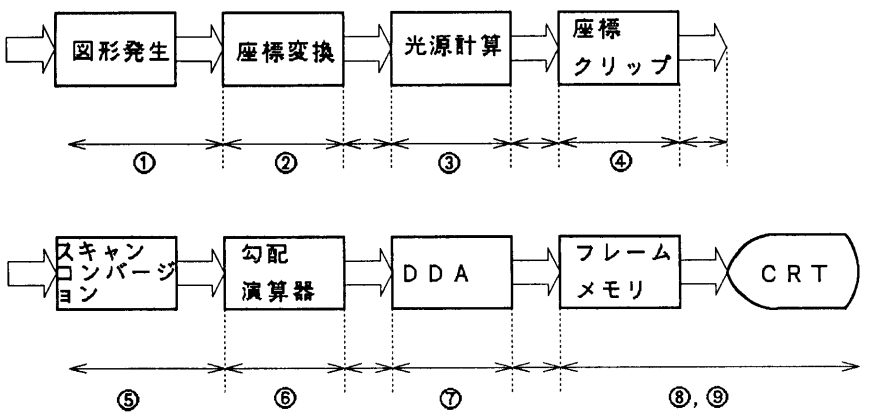


図1．グラフィックパイプライン処理分散

ンステージでの処理を上記番号に対応した番号で示す。本論文では、図中⑥～⑨で示される描画処理すなわちＤＤＡで発生したピクセルデータをフレームメモリ上への展開するまでのハードウェアについて主に述べる。

２．データローディング処理

現在、ベクター描画性能は300,000ﾍﾞｸﾀｰ／秒程度に高速化されているため、１ベクター当たりその処理時間は3.3μsec程度となる。図１に示すように描画部分の各処理段は、パイプライン化されており、勾配演算速度がＤＤＡに影響する事はほとんど無い。従来の様に、100,000ﾍﾞｸﾀｰ／秒程度の描画速度では問題にならなかったが、描画性能が向上するとデータの転送時間が問題となる。各段間のデータ入力バッファがパイプライン化されていないと、データ転送量の多さから各パイプラインステージの処理時間に加算され、ベクター描画時間に影響する。

例えば１２８０×１０２４ピクセルの４０６９色表示で、１６ビットＺバッファを持つグラフィックスディスプレイについて、描画部への図形データのローディングを考える。ローディングは、除算器へのロードとＤＤＡへのロードに分けられる。勾配除算器への入力データ例は、表１に示すものである。同様に、勾配演算器からＤＤＡへ引き渡すデータは、表２のようになる。勾配除算段は高速化のため並列化しＤＤＡに並列に接続して、

表１．除算器への入力データ例

データ	成　分	ビット幅
始点アドレス	X_S, Y_S	各　１１
	Z_S	１６
始点輝度値	I_S	１２
	or R_S, G_S, B_S	各　８
終点アドレス	X_E, Y_E	各　１１
	Z_E	１６
終点輝度値	I_E	１２
	or R_E, G_E, B_E	各　８

上記のローディング時間を最小にする。しかし、スキャンコンバージョン部の除算段では、上位プロセッサからのデータを各成分毎に順に転送するため、シリアル形式のバッファで十分であり、パラレル形式でパイプラインレジスタを構成する必要は無い（図２参照）。

ＤＤＡでの処理時間は次のものである。

①データローディング

②累積加算

③ハードウェア制御オーバーヘッド

データローディング時間は図形データに依存せず一定であるが、累積加算時間はベクトル長に依存し、平均的なベクトル長を４０ピクセルとすると、４０Ｔ時間（Ｔ:ＤＤＡが１ピクセルのデータを生成する時間）必要である。又、ハードウェアの制御オーバーヘッドは、フレームメモリへのデータ転送回数と時間に依存する。従って、同じベクトル長であってもデータ転送回数を小さくすれば描画速度を高速化できる。

一方、勾配除算はシフト減算形式で行なった場合、その必要回数もＸ、Ｙ、Ｚの内最もレンジの

表２．ＤＤＡへの入力データ例

データ	成　分	ビット幅
始点アドレス	X_S, Y_S	各　１１
	Z_S	１６
始点輝度値	I	１２
	or R_S, G_S, B_S	各　８
Ｘ－Ｙ勾配	ｄＸ／ｄＭ	２２
	orｄＹ／ｄＭ	２２
Ｚ勾配	ｄＺ／ｄＭ	２７
輝度勾配	dI/dM	２３
	or dR/dM,	１９
	dG/dM,	１９
	dB/dM	１９
ベクトル長	ｄＭ	１１
Ｍａｊｏｒ軸のフラグ	XorY	１

但し、ｄＭ:絶対値(Xs-Xe)or絶対値(Ys-Ye)の何れか大きい値

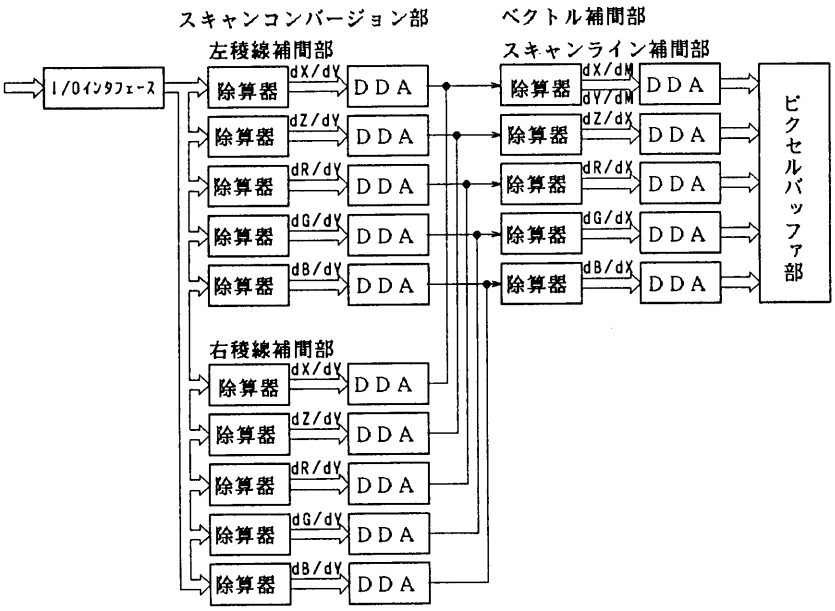


図２．勾配除算器、ＤＤＡの並列化

広い精度のＺバッファに関して行った１６回の為、除算クロックと累積加算クロックが同程度であれば、累積加算に比べ処理が短くボトルネックにはならない。

３．ピクセルバッファ

ＤＤＡが描画データを発生した後フレームメモリへ書き込む場合、ＤＤＡはピクセル当たり６０nsecでデータを発生するが、フレームメモリは通常１００nsec程度アクセスタイムを要し、２００nsec程度のサイクルタイムを必要とする。従って、ＤＤＡからフレームメモリへの書き込みに、数ピクセル分のバッファを用意する事によりこの速度差を相殺する。これを、ピクセルバッファと称する。ピクセルバッファは緩衝バッファであるため、サイズは大きいほどその機能は有効になる。しかし、このピクセルバッファのデータをフレームメモリに一度で転送できるピクセルサイズを選択しなければ転送サイクルが複数回必要となるため有効に働かない。

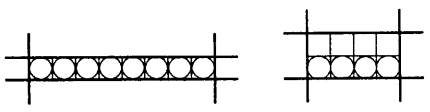
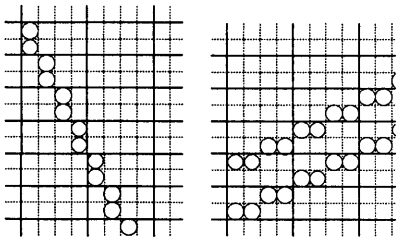


図３．ポリゴン描画時のピクセルバッファ

３．１．ピクセルバッファ構造

ピクセルバッファは、その構成形状により描画速度が変化する。多角形の塗りつぶしでは、スキャンコンバージョン後は水平方向の描画に限定されるため、ピクセルバッファは水平方向に長い方が効率よくフレームメモリに書き込める（図３参照）。しかし、ランダム描画においては描画ベクトルの方向性が限定できないため、任意方向にできるだけ多くのピクセルをバッファリングしなければならない。

ランダムベクトル描画に於いては、ピクセルバッファ領域をはみ出したときフレームメモリへのデータ転送要求が発生する。この時、ＤＤＡを停止させないためピクセルバッファをダブルバッファ構造にする。しかし、描画データによりバッファリングできるピクセル数が変化し、ダブルバッファであってもＤＤＡを強制的に停止させる必要



(a)縦方向描画　　(b)横方向描画

図４．ランダムベクトル描画(4×2ﾋﾟｸｾﾙﾊﾞｯﾌｧ)

が生じる。

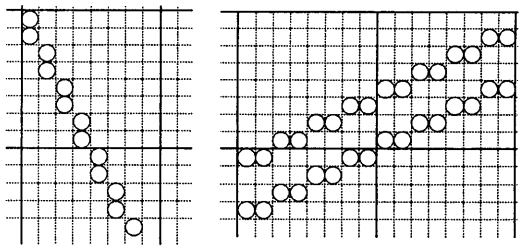
４×２のピクセルバッファを構成した場合を例にピクセルバッファの動作を説明する。図４で太線は、ピクセルバッファの境界を示しＤＤＡがこれを横切って描画する時、ピクセルバッファの内容をフレームメモリに転送する。例えば図４(a)の場合ＤＤＡが６０nsecで動作すると、２ピクセル描画(120nsec) 毎に、データ転送要求が発生する。従って、ピクセルバッファからフレームメモリへの転送が、１２０nsec以内に終了しなければＤＤＡは停止する。しかし、通常メモリサイクルタイムは２００nsec程度必要なためＤＤＡは停止する。[3]

図４(b) の場合２本のライン描画で勾配が同一のものを示すが、下のものは４ピクセル毎(240 nsec) にピクセルバッファが切り替わるため、ＤＤＡは停止しない。しかし、上のラインは２ピクセル毎(120 nsec)にバッファが切り替わるために、ＤＤＡは２ピクセル描画毎に停止する事になる。従って、描画データによりその描画の際のピクセルバッファの転送回数即ちオーバーヘッドが変化するため、描画速度がデータに依存し、必ずしも最大性能が得られない事になる。

３．２．ピクセルバッファの拡大手法

スクリーンサイズよりも大きなフレームメモリを実装する事により、ピクセルバッファサイズを大きく採ることが可能となる。現在グラフィックスディスプレイで使用されているＣＲＴの解像度は、通常１２８０×１０２４ピクセルであるが、フレームメモリとして、２０４８×１０２４ピクセル用意する。このように、２Ｍピクセルを２５６KVideo-RAM(64K×4)（以後V-RAMと略す）で構成すると８個で１プレーンあるいは３２個で４プレーンが構成でき、３２がピクセルバッファの構成ピクセル数になる。

更に、近年グラフィックスシステムは、描画プレーンをダブルバッファで装備することが標準化している。従ってピクセルバッファを大きく取るためにこのダブルバッファを有効に利用する。この場合２０４８×２０４８ピクセルのフレームメ



(a)縦方向描画　　(b)横方向描画

図５．ランダムベクトル描画(8×8ﾋﾟｸｾﾙﾊﾞｯﾌｧ)

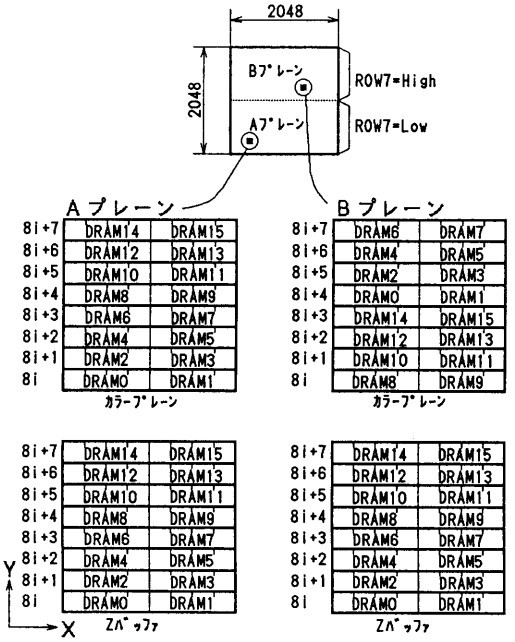


図６．フレームメモリＤ－ＲＡＭ割付

モリを想定して、フレームメモリを１６個で１プレーンあるいは、６４個で４プレーンを構成し、ピクセルバッファを８×８に構成する事ができる。

図５の場合、太線がピクセルバッファの境界を示し、ＤＤＡがこの境界を横切るときにフレームメモリへのデータ書き込みが行われる。ピクセルバッファが、ダブルバッファであり１回のフレームメモリへの転送でピクセルバッファの書き込みが終了するならば、ＤＤＡは停止しない。この場合に、ＤＤＡが停止するのは、ベクトルの始点あるいは終点部分で描画ピクセルが４ピクセルに満たない場合だけである。

グラフィックスは、４０９６色同時表示モードでのダブルバッファを、ＲＧＢ各８Bit に対応させて１６７０万色同時表示モードを実現している。従って、スキャナが表示の際フレームメモリを読み出す時、前者では１２プレーンを後者では２４プレーンを同時にアクセスする必要がある。従って、カラープレーンのV-RAM をＡプレーンとＢプレーンとでは、図６の様に、メモリマッピングを変える事により２４プレーン同時読みだしを実現する。

また、３次元グラフィックスでは、Ｚバッファ法による隠面消去が一般的であり、ピクセルバッファを拡大するためには、Ｚバッファも同様に広げなければならない。しかし、Ｚバッファはカラープレーンの書き込みアクセスプレーン（ＡorＢ）に関わらず、シングルプレーンで構わない。従って、Ｚバッファとカラープレーンでは、メモリ配置を図６に示すように変える必要がある。これにより、４０９６色ダブルバッファでの書き込みでは高速性が確保され、フルカラーモードでの表示も可能となる。但し、フルカラーモードでの書き込みは１２プレーン単位でしか出来ないため２度のアクセスになる。しかし、この欠点もフルカラーでのフレームメモリをダブルバッファにするならば解消される。

４．メモリインタリーブ方式

ＤＤＡの描画時間は、６０nsec/pixel程度まで高速化されているが更に高速化が望まれている。従って、これを実現するためにＤＤＡを複数装備する。ＤＤＡを並列化する事により実効的なＤＤＡ速度を向上させるためには、ピクセルバッファへの書き込みが独立に出来、更にそのデータをフレームメモリに独立に書き込めなければならない。これを実現するために、フレームメモリをスキャンライン単位でインタリーブし、ピクセル

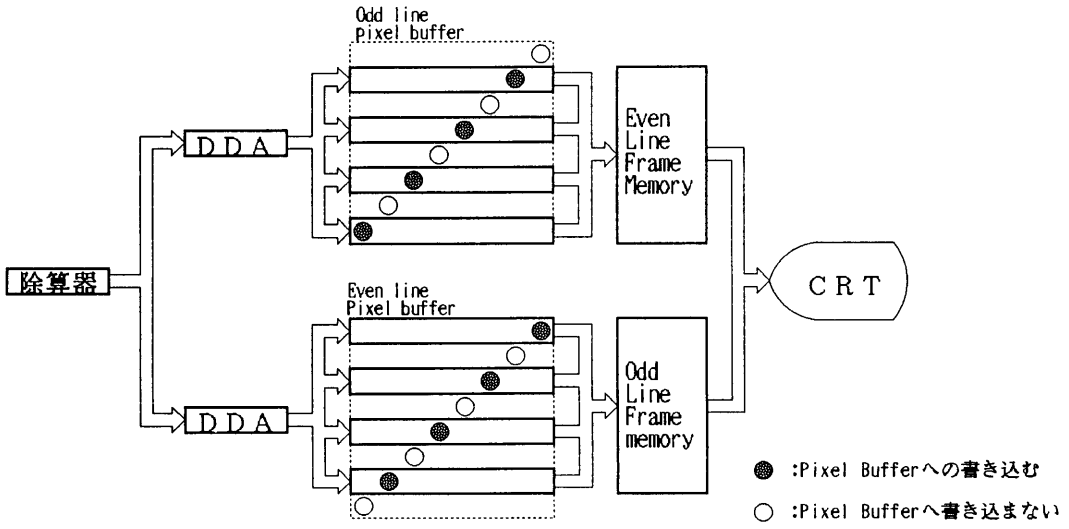


図７．２chＤＤＡとピクセルバッファ構成

バッファの内容をスキャンライン単位で描画終了後、待ち時間無しでフレームメモリへ転送する。

ＤＤＡの並列化として、同一ベクトルを複数のＤＤＡでデータ展開しメモリを分けて描画する方式を提案する。ピクセルバッファとフレームメモリを対応させ、ＤＤＡから展開されるデータをピクセルバッファへ展開する際に、フレームに対応したデータのみに限定し、発生データを分散化する。

図７に、２chＤＤＡで８way にインタリーブした場合を示す。８×８ピクセルバッファの構成を、８×１ピクセルバッファをＹ方向に８個配置し、各ピクセルバッファに対応したフレームメモリを構成する。偶数スキャンラインと奇数スキャンラインで描画ＤＤＡを分け、それぞれを独立に動作させる。これによりＹ方向への描画には、一切のオーバーヘッドがパイプライン化され見えなくなる。この方法では、同一スキャンライン上に、双方向に連続的にライン描画しない限り、ピクセルバッファからフレームメモリへのデータ転送のオーバーヘッドは解消できる。

５．ゲートアレイについて

ピクセルバッファを大きくする、あるいは多チャンネルのメモリインタリーブを実現するために、我々は、２種類のゲートアレイを用いた。グラフィックスディスプレイでは、同時表示色の増加、Ｚバッファの高精度化、ワーキングプレーンの増加により、４４プレーン/pixel（２４：カラー、１６：Ｚ、４：オーバレイ）になっており、今後も増大傾向を示している。従って、８×８のピクセルバッファを実現する為には、カラープレーンを１２フレームとしてもピクセルバッファとの間に、ＤＤＡ側に４４本、フレーム側に１９２０本のデータ信号線を設ける必要がある。従って、ゲートアレイ化する際の問題点として、入出力ピン数がネックになった。そこで、我々はピクセルバッファを自由に構成出来るように、４ピクセル単位での構成を基本としゲートアレイ化した。また、メモリインタリーブ回路も同様に、４ライン分を１チップ化した。

以下に各ゲートアレイの機能を挙げる。

①．ピクセルバッファ（ＰＩＢ:PIxel Buffer）

- ＣＭＯＳ　５０００ゲート
- １６０ピン　ＱＦＰパッケージ
- ４ピクセル×１６Bit
  ダブルピクセルバッファ
- ４ピクセル同時Ｚ比較（隠面消去機能）
- 各種ラインパターン発生器
- セクショニングバッファモード
- 半透明パターン発生器
- ピクセルＲ／Ｗ機能
- 高速消去モード
- Ｚソーティング機能
- ３Ｄヒット機能

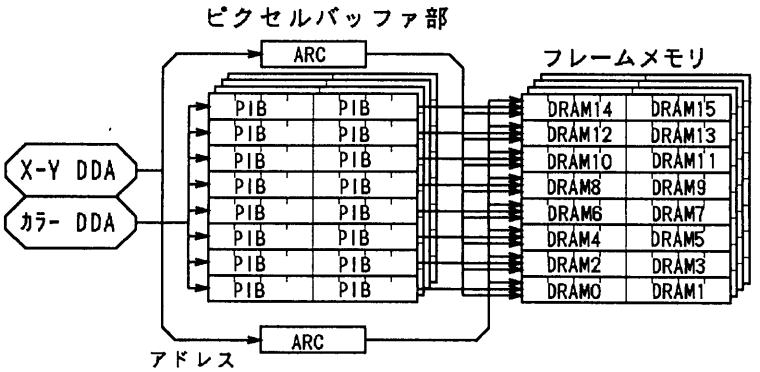


図８．ピクセルバッファとフレームメモリ配置図

②．アドレスコントローラ（ＡＲＣ:AddRess Controller）

- ＣＭＯＳ　５０００ゲート
- １６０ピン　ＱＦＰパッケージ
- ４スキャンライン独立アドレス制御機能
- 塗り潰しＤＤＡ回路
- ＣＲＴコントローラ
- ２５６Ｋ V-RAMアドレス制御
- ピクセルＲ／Ｗ機能
- ２Ｄヒット機能・高速消去モード

８×８のピクセルバッファを構成するためにＰＩＢをカラープレーンに１６個、Ｚバッファに１６個を使用する。ＡＲＣは、４スキャンライン分のアドレス制御を独立に行なうため、カラープレーンに２個、Ｚバッファに２個使用する。これらの配置を図８に示す。各ゲートアレイは、制御プロセッサのデータバスに１６ビット巾で接続し、初期セットや各種パターンを内部レジスタに、ダウンロード可能とした。また、内部にプログラム可能なアドレスを持たせる事により、制御プロセッサからアドレスをダイナミックに変更している。これにより、３２個全てのゲートアレイに同一データを設定する（ハッチングパターンの書換、ラインパターンの書換、高速画面消去等）際の設定時間が短縮される。

６．むすび

本論文では、グラフィックスディスプレイでランダムベクトル描画する際の描画速度を低下させる問題点として、ピクセルバッファの形状とフレームメモリへのデータ転送時のオーバーヘッドを示し、それを回避するための実現手段を示した。本方式を採用したグラフィックスディスプレイ（ＣＯＭＴＥＣ　３５５５）での描画速度は、ベクトルの方向や、始点アドレスに関わらず３００,０００ﾍﾞｸﾀｰ／秒の描画性能を示している。

今後、描画速度を更に高速化するには、グラフィックパイプラインでの処理速度を上げることが急務である。しかしデバイスの進化は、既に4MV-RAM を実現しており、これはピクセルバッファサイズを４分の１にしてしまう。従って、今後増大する高速化の要望にたいしてここに挙げた方式ではなく新たな提案をしなければならないだろう。

グラフィックスに求められる対話性は、今後ますます増大するだろうし、それをサポートするためには、描画性能を更に高速化しなければならない。更に、システムとしての使い勝手を良くする要求に答えるため、より高速なハードウェアを開発して行く必要がある。

最後に、本研究の機会を与えてくださいました弊社伏見常務、電子技術研究所吉田所長、電子機器事業部下茂事業部長に感謝いたします。

参　考　文　献

(1)山口：図形処理工学,日刊工業新聞社,(1981)
(2)ﾌﾞﾗｲｱﾝ・ｸﾛｰﾙ:"グラフィックス・システムを評価するチェック・ポイント",日経ＣＧ４月号, pp.109-113(1988)
(3)日立ＩＣメモリデータブック,日立製作所, (1988)