2U－2

# グラフィックスワークステーションにおける PEXの高速化方式*

中村 昭次†　木村 信二†　古賀 和義‡　藤井 秀樹§
†(株)日立製作所システム開発研究所　‡同 日立研究所　§同 大みか工場

## 1 はじめに

ワークステーションのオープン化、標準化が進む中、PEX(PHIGS/PHIGS PLUS Extension for X)[1]が3次元グラフィックス標準の有力候補として注目されている。また、ハードウェア界面においては、高性能なRISCチップの出現により、特別な専用ハードウェアのない単純な構成のシステムでも、実用可能な性能で3次元グラフィックスを実現できるようになってきた。

そこで、低価格、高性能なオープン化グラフィックス環境を構築するため、グラフィックスワークステーション向けにPEX高速化方式を開発した。本稿では、高速化方式の特徴と効果について報告する。

## 2 PEX 高速化方式

PEXは、Xウィンドウ上にPHIGSを実現する3次元グラフィックスのX拡張であり、PHIGS機能を幾つかのPEXリソースに機能分割する形で、Xに搭載されている。それゆえPEXでは、ストラクチャモードに基づくPHIGSだけでなく、PEXlibという低位層ライブラリによってイミディエートモード・レンダリングが可能になっている。多様化するAPニーズに対応するには、イミディエートモードが重要と考え、PEXlibインタフェースで、線分表示1 [Mv/s]以上の性能を達成することを目標とした。

### 2.1 協調型クライアント DHA 方式

PEXの高速化には、描画処理以外に、ウィンドウアーキテクチャに依存した課題がある。PEXはクライアントーサーバ方式のため、PEXプロトコルの転送が必要になるが、Xlibに比べてデータ量が多く、転送オーバヘッドが深刻になる。そこで、クライアントがサーバと同じ計算機上で動作する場合には、サーバを介さずクライアント側で直接的に描画処理するDHA(Direct Hardware Access)方式を採用した。図1にDHAによるPEXアーキテクチャを示す。ウィンドウ枠管理を始め、PEXリソースの制御は全てサーバ側に任せ、属性変更や図形表示といった描画系機能だけをクライアント側で実行する協調型クライアントDHA方式とした。ウィンドウや、レンダラのデータをDHAContextと呼ぶ共用メモリに置き、描画処理をDHAlibと呼ぶ共用ライブラリに集めることで、サーバとクライアントでの描画の一貫性を保てるようにした。

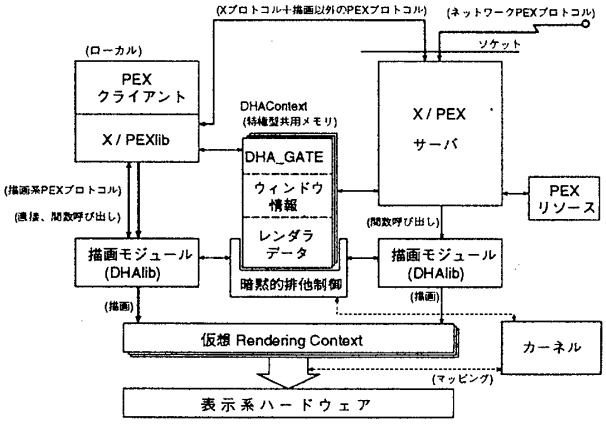


図 1: DHA によるPEX アーキテクチャ

---

* High performance implementation of PEX on graphics workstation

† Shouji NAKAMURA,Shinji KIMURA
Systems Development Laboratory,Hitachi,Ltd.

‡ Kazuyoshi KOGA　Hitachi Research Laboratory,Hitachi,Ltd.

§ Hideki FUJII　Omika Works,Hitachi,Ltd.

### 2.2 ライブラリベースのプロテクション方式

クライアント側でのDHA描画では、グラフィックス資源のプロテクションが問題になる。つまり、グラフィックス資源をアクセスするDHAlibは、APのライブラリ・プログラムであり、グラフィックス資源をアクセス可能としている。このため、APが暴走して、表示系ハードウェアを不当アクセスすると、表示画面が壊れ、複数プロセス間で共用しているDHAContextに対するメモリ破壊は、サーバや他のクライアントにまで、影響を及ぼす。

この解決方法としては、グラフィックス資源をアクセスする処理をカーネルに組み込む方法が従来行なわれているが、描画処理の細かい単位でシステムコールを発行しなければならず、オーバヘッドが極端に増大する欠点がある。最低でも数$\mu s$から数十$\mu s$は必要となる。そこで、APとカーネルの中間モードとして、DHA特権レベル描画を採用した。つまり、グラフィックス資源をDHA特権レベル以上ではアクセス可能とし、DHAlibの出入口で特権レベルを設定する。この方式により、ライブラリベースのプロテクションが可能になった。

### 2.3 暗黙的排他制御方式

DHA描画におけるもう一つの課題は、同期・排他制御の効率良い実現法にある。サーバによるウィンドウ配置状態変更処理とDHA描画処理の同期制御、また複数のDHAクライアント間でのDHA-Contextの排他制御の効率を高める必要がある。一つの実現方法に、DHAContextにセマフォを置く方法が考えられるが、2頂点線分のような軽い図形描画処理では、排他制御処理の負荷が無視できない。排他制御は必要不可欠な処理であるが、実処理形態に着目すると実際に効果があるのは稀で、高額な保険に過ぎないと考えられる。そこで、1クライアントが1レンダラを利用する通常の形態において、排他制御オーバヘッドを最小化する暗黙的排他制御方式を提案する。

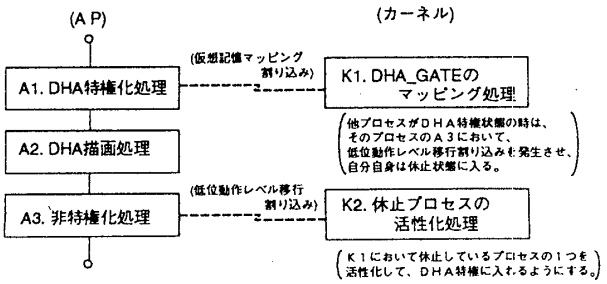


図 2: 暗黙的排他制御方式

図2に概念フローを示す。DHA特権で動作するDHAlibの1関数をクリティカル領域とし、特権モードの出入口で割り込みを発生させて、排他制御の実処理をカーネルに実行させる。割り込みの発生は、DHAContextの参照状態に基づき、必要な時だけに抑止する。DHA特権状態設定処理がそのまま排他制御の出入口処理に対応し、クライアントは排他制御を明示的に実行する訳ではないので、通常状態での排他制御オーバヘッドを低減できる。

## 3 性能

PEXlibの描画性能について報告する。

| 描画経路 | 線分 [kv/s] | 連続三角形 [kp/s] |
|---|---|---|
| サーバ描画 | 353 | 102 |
| DHA描画 | 1100 | 130 |

表の通り、DHA描画は、特に線分のような描画処理の軽い図形に対して効果的である。

## 4 おわりに

開発したPEX高速化方式は、クライアントが直接描画処理するDHAを採用し、ライブラリベースのプロテクション方式、暗黙的排他制御方式により、安全性と高速性を両立した点に特徴がある。既に中規模ワークステーション上で効果を確認しているが、描画処理そのものをより高速化できる高位機種において、さらに効果を発揮すると考えている。

## 参考文献

[1] Paula womack et.al : PEX Protocol Specification, MIT (1992)