ビットマップ・グラフィックスプロセッサEDPU

(4) WCSを用いたコマンドの高速化手法

6L－5

峰久 次郎　西岡 由利子　東 幸哉　神山 祐史

松下電器産業(株)　情報通信関西研究所

1. はじめに

EDPUではコマンド体系をCGIに準拠させ、ユーザーとのインターフェースの向上を図っている。このコマンド体系はマイクロプログラム制御方式により実現しているが、数多くのコマンドを限られたマイクロプログラムメモリ容量の中で実現するために、コマンドに付随するオプションやデータの組合せに左右されない汎用的なアルゴリズムを採用している。特に専用ハードウェアで最適化・高速化を図った直線描画・矩形領域転送コマンド等の基本機能以外では、実行速度とともに汎用性を考慮したアルゴリズムを採用している。

一方グラフィックスプロセッサの用途を考慮した場合、特定条件下でできるだけ高速の処理を行いたいとの要求も存在している。このような特定用途に対する要求にマイクロプログラムレベルで対応するためにEDPUではWCS(Writable Control Strage: 書換え可能制御メモリ)を128ワード×32ビット備えた。このWCSに特定用途向けのマイクロプログラムを格納することで高速化した専用コマンドを構築できる[1]。本稿ではこのWCSの効果について簡単に述べた後、WCSを用いた高速化手法についてのシミュレーション結果によりその有効性を示す。

2. WCSの効果

WCSにより以下の効果を得ることができる。

(1) 高速化

機能を限定した専用マイクロプログラムをWCSに格納することでEDPUのROMに組み込まれたコマンドより高速なコマンドを実現する。

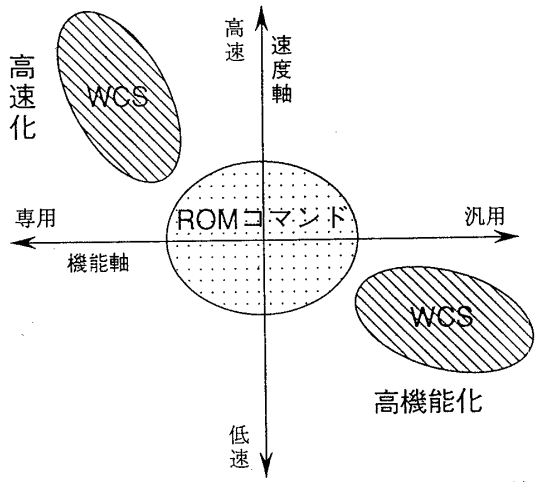
(2) 機能拡張

組み込みコマンドには含まれていないグラフィックス機能、たとえば画像データ圧縮伸張機能や円弧フィル(扇形塗りつぶし)等を実現する。このような機能を実現するためにはWCSよりも大きなサイズのマイクロプログラムが必要とされる場合もあるが、EDPUではWCSにオーバーレイ機能(必要なプログラムを順次WCSへロードする機能)を備えることでWCSの容量に制限を受けることなく機能拡張を可能にしている。

以上の2点を模式的に示したのが第1図である。図中のROMコマンドはEDPUの組み込みコマンドで実現された機能を示し、高速性を重視した図中左上の領域と高機能性を重視した右下領域とをWCSによりカバーできることを示している。

(3) 検査

マイクロプログラムでは内部ハードウェアを特定して動作させることが可能で、効率的にチップ検査ができる。EDPUでは約200本のテストプログラムを作成しチップのスクリーニング、評価に利用している。



第1図　機能と実行速度からみたWCSの効果

3. 高速化手法

ここではWCSを用いた高速化手法について述べる。高速化の手法として以下の3つについて検討した。

a. 条件判定の簡略化

組み込みコマンドではオプションの組合せ、データの組合せに関係なく機能を実現するために、条件分岐、ループの終了判定等に余分な処理時間を要している。限定された機能・条件を生かし、条件分岐・終了判定を簡素化して高速化を図る。

b. ハードウェアの活用

EDPUはグラフィックスプロセッサとして様々な専用ハードウェアを備えている。オプション、データの特定の組合せでは組み込みコマンドで使用されていないハードウェアを利用して高速化を図ることができる場合がある。

c. アルゴリズムの変更

組み込みコマンドでは汎用性を重視したアルゴリズムを採用している。従ってオプション、データの特定の組合せに特化したアルゴリズムに変更することで高速化を図ることができる。

4. 高速化例

上記に述べた方針に従って高速化を図った例を以下に示す。ここではソース空間およびデスティネーション空間がフレームメモリに設定され、バス効率が100%であると仮定してシミュレーションを行い、実行ステップ数を比較した。

a. 条件判定の簡略化

フォント(キャラクタ)転送:　EDPUの組み込みコマンドで用意されているフォント転送コ

A Bitmap Graphics Processor EDPU (4) Boosting execution speed by utilizing WCS.
Jiro MINEHISA, Yuriko NISHIOKA, Yukiya HIGASHI, Hiroshi KAMIYAMA
Kansai Information and Communications Research Laboratory
Matsushita Electric Industrial Co.,Ltd.

マンド(CHR、POLCHR)では傾き(スラント)、拡大、回転の3つのオプション以外にも、フォントサイズや転送先の1ピクセルを構成するビット数(DBPP:Destination Bit Per Pixel)等を選択できるようになっている。このためフォントサイズやDBPPが固定されている場合でも、その条件を生かせずにパフォーマンスを向上させることができない。そこで、フォントサイズをそれぞれ16×16、32×32、48×48に、DBPPは1に固定した専用プログラムを開発し、組み込みコマンドとの速度比較を行った。この例では条件を固定することにより、汎用のフォント転送コマンドに比べ、転送1ワード当りに条件判断を2箇所削除できたのに加えアルゴリズムが簡略化できた。シミュレーション結果を第2図に示す。実行ステップ数を約1/2以下に低減できることがわかる。

b. ハードウェアの活用

回転縮小矩形領域転送: 矩形領域転送コマンド(MV)のオプションではm/n(m,n=1～16)の任意倍率の拡大／縮小および回転つきデータ転送を行うことができる。しかし、倍率を$1/2^k$倍(K=1～4)、1ピクセルを構成するビット数を1に固定した場合にはハードウェアに組み込まれた機能を利用することで高速化が図れる。即ち、iビット(i=2,4,8,16)の多値データを1ビットの2値データに変換する色比較部の機能[2]を利用して32ビットデータを1度に32/iビットに縮小する。このように32ビット単位で処理ができるので、マイクロプログラムで1ピクセル単位に制御しているMVコマンドに比べ高速に処理できる。倍率をそれぞれ1/4、1/8に設定した場合のシミュレーション結果を第3図に示す。条件によっては20倍近く高速化できる場合がある。

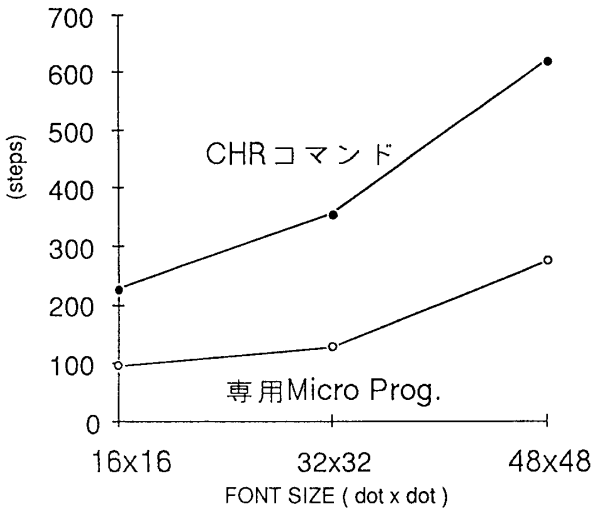
c. アルゴリズムの変更

太線描画: EDPUでは太線を描画する場合、16×16の矩形領域を描画点へ貼付けていく処理を行う。従って矩形領域のほとんどが利用されない比較的小さなパターンの場合には無駄なIO動作を繰り返すことになる。比較的線幅の小さい直線(太線)を描画する場合を考えると、LINコマンドの太線オプションを利用するよりも、細線を繰り返し描画した方が処理速度が早くなる場合がある。線幅3の専用マイクロプログラムを作成し、LINコマンドとの速度比較を行った。そのシミュレーション結果を第4図に示す。

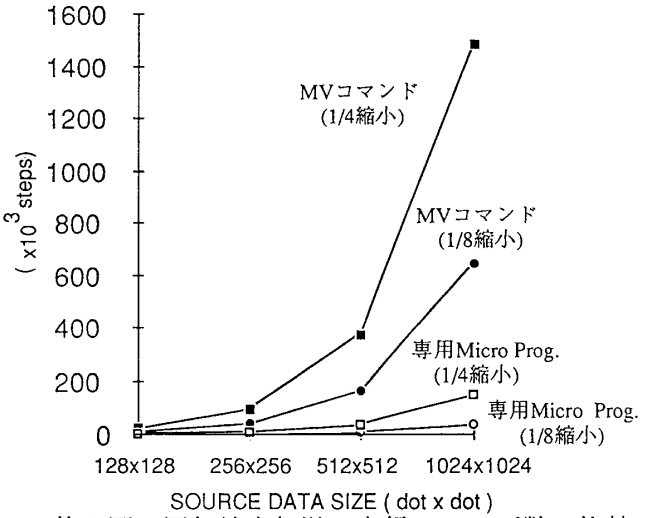
5. おわりに

以上、本稿ではEDPUに組み込まれているWCSを用いた高速化手法についてまとめ、組み込みコマンドとの比較を行った。コマンドの専用化による高速化では1桁以上のパフォーマンスの向上が得られる事例があり、WCSが有効であることが確認できた。今後はここで述べた手法に従い、アプリケーションに特化した高速化コマンドの開発を進めていきたい。

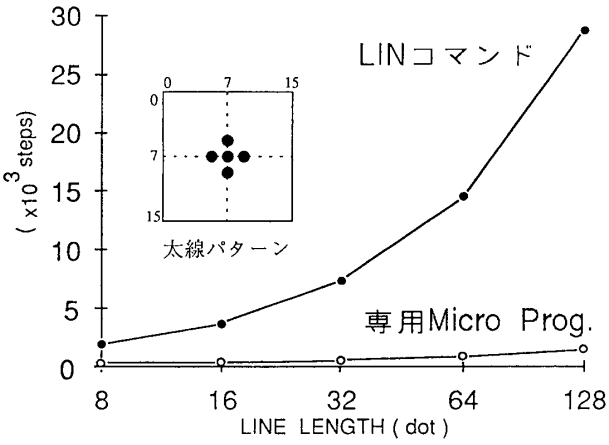
最後に、マイクロプログラムの開発を進めるにあたって非常な御助力をいただいた松下電器技術開発股份有限公司台北技術研究所各位、ならびにマイクロプログラム開発に協力頂いた松下電器産業半導体サポートセンター三浦和男氏に感謝します。



第2図 フォント転送の実行ステップ数の比較



第3図 回転縮小転送の実行ステップ数の比較



第4図 直線発生実行ステップ数の比較
(線幅3の場合)

参考文献

[1]「高速画像データ転送機能を有するCRTコントローラ」、出口他、情報処理学会計算機アーキテクチャ研究会技報、VOL.85、No.53 (1985.12)

[2]「グラフィックプロセッサの2値化、多値化変換方式に関する一考察」、東他、昭和63年電子情報通信学会春季全国大会、No.C-272 (1988.03)