バースト・アクセスの特性を生かしたグラフィックス表示システムを構築

Appendix 4

3D グラフィックス表示と SDRAM の利用効率を高めるアクセス手法

長船 俊

筆者は以前，本誌 2005 年 6 月号 特集「やってみよう！ FPGA システム設計入門」において，FPGA による 3D グラフィックス表示システムの設計事例について解説しました．また，このシステムを CQ 出版社発売の評価キット「組み込みシステム開発評価キット」にも移植し，2006 年秋にパシフィコ横浜で開催された Embedded Technology 2006 で展示することもできました．

しかし，2005 年 6 月号掲載時はページ数などの都合から，3D グラフィックス表示の基礎的な解説に終始し，SDRAM コントローラ周辺についての詳しい解説はできませんでした．

筆者の設計したこの 3D グラフィックス表示システムは，“ PROCYON ”と命名しています．PROCYON は，RAS/CAS 制御やバースト転送長など，SDRAM のバースト・アクセスに最適なアルゴリズムを採用しているため，SDRAM コントローラとも密接に関係して動作しています．

ここでは，一般的な使い方とは少し違った SDRAM の特徴的な活用事例として，筆者の開発した 3D グラフィックス表示システムの SDRAM 制御部分に焦点を当てて解説します．

● SDRAM の標準的なアクセス・パターンとその問題点

図 1 は SDRAM の標準的なアクセス・パターンです．CPU のキャッシュ・フィルやビデオ表示など，アドレスがほぼリニアに変化するもの（アドレス連続であるもの）については，SDRAM はバースト・アクセスを使って大幅に性能を引き上げることができます．反面，SDRAM はバースト・アクセスに適したメモリで，ランダム・アクセスをする場合，バースト・アクセス時の 1/5 ～ 1/8 程度の速度になってしまいます（**図 2**）．

3D グラフィックス表示においてランダム・アクセスが必要

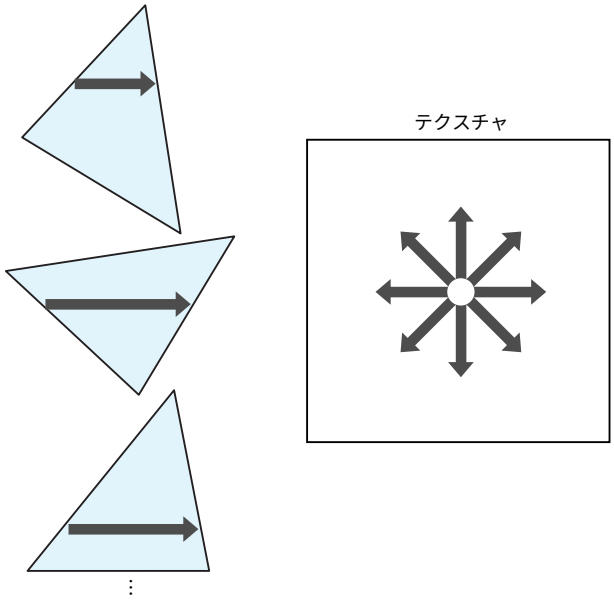

図 3　テクスチャ・マップド・ポリゴン
ポリゴンの表示状態によって，あらゆる方向へテクスチャ・ピクセル・スキャンが発生する

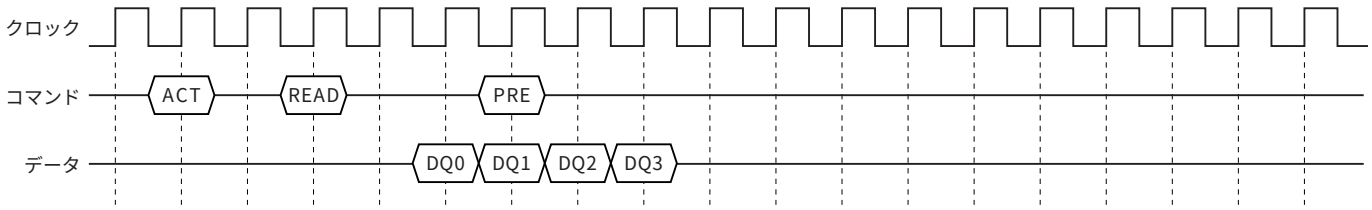

図 1　標準的な SDRAM のバースト・アクセス例（バースト長 4，CL=2）

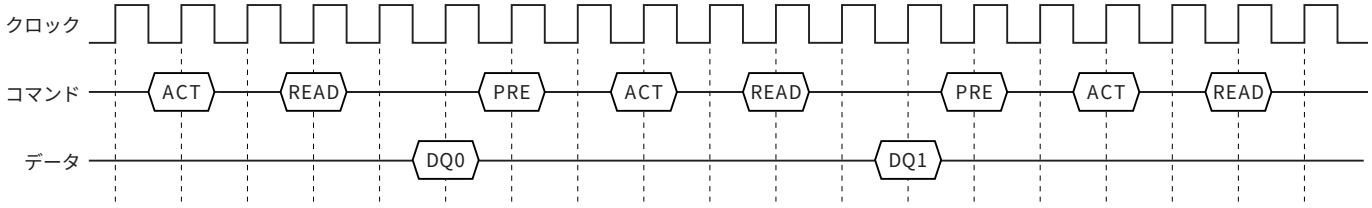

図 2　標準的な SDRAM のランダム・アクセス例（バースト長 1，CL=2）

KEYWORD ―SDRAM，RAS，CAS，ロウ・アドレス，カラム・アドレス，バースト長

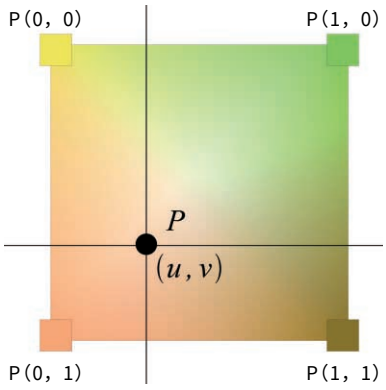

図4　バイリニア補間

点の周囲の格子点4点のテクスチャ・ピクセル・データを線形補間して描画ピクセル・データを得る

なものの筆頭といえば，スプライトやポリゴンのテクスチャ・ピクセル・アクセスです．テクスチャ・ピクセル・アクセスでは，フレーム・バッファ上に描画するポリゴンの状態によって，あらゆる方向へのスキャンが発生することになります(**図3**)．

この場合のスキャン・アドレスには小数部をもつ実数が使われますが，テクスチャ・ピクセルの画素座標(μν座標)はほとんど整数の離散値です．そのため，スキャン・アドレスからテクスチャ・ピクセル・データを得るためには，何らかの内挿処理を行わなければなりません．

もっとも簡単な処理は，スキャン・アドレスの小数部をすべて切り捨ててテクスチャ・ピクセル座標を読み出す方法ですが，普通は何らかのフィルタリング処理を行います．

PROCYONではリソース対性能の兼ね合いから，バイリニア・フィルタリングを採用しました．バイリニア・フィルタリングとは，等倍以上の拡大率の場合にテクスチャ・ピクセルの隙間を補間する手法の一つで，ある点Pの近傍4ピクセルから色を線形補間によって内挿するというものです(**図4**)．

この方法は計算量が少なくてすむ反面，拡大率が大きい場合に画像が正しく再現されないという欠点もあります．しかし，少ない計算量(4回の積和演算)でそこそこの画質が得られることから，速度を重視する描画エンジンでは好んで使用されています(実際のポリゴン描画エンジンでは，これに奥行き方向の線形補間を追加したトライリニア・フィルタリングが使用されている)．

さて，いくらバイリニア・フィルタリングが少ない計算量で済むとはいえ，1ピクセルの描画には4テクスチャ・ピクセルを読み込む必要があります．単純に考えても4倍のメモリ帯域が必要になりそうです．仮に，バイリニア・フィルタリング処理に入力するデータ列をP(0，0)，P(1，0)，P(0，1)，P(1，1)とします．このときP(0，0)とP(1，0)，またP(0，1)とP(1，1)は連続アドレスであり，SDRAMのバースト・アクセスの効果が期待できます．

ところが，P(0，1)からP(1，0)はアドレスが飛ぶため，いったんロウ・アドレスをプリチャージして，再び該当ロウ・アドレスをアクティブにしなければなりません．SDRAMのプリチャージ→ロウ・アクティブ・ディレイ，ロウ・アクティブ→コマンド・ディレイ，さらにCASレイテンシを考えると，バースト・アクセスの倍以上の待ち時間が必要なことが分かります(**図5**)．

しかし，SDRAMを教科書的に使うこのアクセス方法では，これ以上速くすることはできません．速度アップの正攻法としては，クロックを上げる，バス幅を増やす，テクスチャ・キャッシュを導入するなどが考えられるでしょう．しかし，既に100MHz～133MHzで動作しているSDRAMのクロックを上げるのは容易ではありません．また，上げられたとしてもせいぜい166MHzがいいところでしょう．

バス幅を増やすというのも常とう手段です．しかし，バイリニア・フィルタリングでは2テクスチャ・ピクセルぶん以上のビット幅は，テクスチャ・キャッシュを導入しない限り効果的に使うことができません．一般的な手法なら，ここでテクスチャ・キャッシュの導入ということになるわけですが，これは高速化と引き換えに，リソースもある程度の量を要求されます(右掲のコラムを参照)．

● データの並びを最適化する

もともと小規模FPGAに実装することを目的としたPROCYONでは，バス幅を増やしたりテクスチャ・キャッシュを導入するといった正攻法の手段を採ることができません．そのため，データの並びやSDRAMのアクセス・パターンを最適化することで性能を得るような設計が必要になってきます．

バイリニア・フィルタリング時のアクセスの欠点はy方向(アドレスが不連続な方向)のアクセスが必ず挟まることだと既に説明しました．

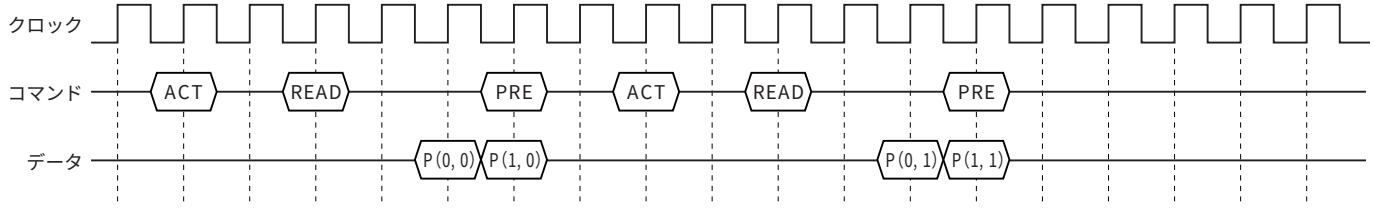

図5　バイリニア・フィルタリング時のアクセス(バースト長2，CL=2)