特別研究報告書

ボリュームレンダリング専用並列計算機 ReVolver/C40 のシェーディングステージ

指導教官　富田 眞治 教授

京都大学工学部情報工学科

荻野 友隆

平成 7 年 2 月 15 日

ボリュームレンダリング専用並列計算機 ReVolver/C40 のシェーディングステージ

荻野 友隆

内容梗概

ボリュームレンダリングによる画像生成法は、医療画像の生成手法として、また科学技術計算の結果の可視化手段として重要視されている。しかし、この手法は膨大な記憶容量と計算量を要するため、当研究室ではボリュームレンダリング専用並列計算機 ReVolver/C40 を開発中である。ReVolver/C40 は視線生成/ピクセル値計算/シェーディング処理を独立した機構に置き、パイプライン動作によって高速表示を可能としている。

本論文では、ReVolver/C40 においてシェーディング処理を行う、シェーディングステージの構成について述べる。

シェーディングとは、光源情報を用いて生成画像の輝度を補正することにより、画像に現実感を持たせたり画像情報の把握を容易にする手法であり、このステージでは depth gradient shading の手法を用いて計算している。この手法は、画像生成で得られた深さバッファ、すなわちスクリーン上の各ピクセルごとに視点と表面ボクセル間の距離 (深さ) を格納した 2 次元配列を用いて表面の勾配を計算し、ピクセルの輝度を求めるピクセルアプローチの一種である。また、勾配計算の際に X/Y 方向それぞれについて目的のピクセルを中央に含む数ピクセルの区間領域を取り出し、その両端の depth を用いて勾配を求めているため、処理内容が簡素で高速化に適している。このため、種々のボリュームレンダリング専用計算機において採用されている。

シェーディングステージの処理内容は主にピクセル値の受け取り・シェーディング・画像出力の 3 つに分けられ、シェーディング処理の部分に DSP を中核とする PE を 16 台配置している。PE がピクセル値の

受け取りや画像出力という繁雑な処理を気にせずシェーディング処理に専念できるよう、ピクセル値の受け取りは Pixel Transceiver(PT)、画像出力は Video Signal Generator (VSG) が担当している。シェーディングを行う PE は Shading Unit (SU) と呼ばれ、ピクセル値の受け取りバッファ (Receive Buffer, RB) と画像出力のためのフレームバッファ (Output Buffer, OB) を持ち、DSP が RB からピクセル値と深さ値を読み出してシェーディング処理を施した後、OB に書き込む。PT, SU, VSG が同一フレームのバッファにアクセスしないように、RB と OB はそれぞれデュアルポートメモリを用いたダブルバッファ構成になっている。

この構成により、ピクセル値の書き込み/シェーディング処理/画像表示は独立し、フレーム単位のパイプライン動作を行うことが可能となった。

プロトタイプでは、DSP に 50MHz 版の TMS320C40 を使用し、RB は高速 SRAM を用いて 2 バンク構成にし、OB には画像用マルチポートメモリ (VRAM) を使用している。

プロトタイプによる性能予測で、このシェーディングステージは最高秒間 20.2 フレームの速度でシェーディング処理を行うことができることがわかった。

Shading Stage of Parallel Volume Rendering Computer 'ReVolver/C40'

TOMOTAKA OGINO

Abstract

Image generating by volume rendering has been regarded important as both a generating method of medical images and a viewing method of results acquired from scientific computation. Because this method requires an enormous storage and large computation, we are developing a parallel machine specially for volume rendering 'ReVolver/C40.' This machine can display images fast with a pipeline processing of three independent structures, Ray Casting Stage, Pixel Calculation Stage, and Shading Stage.

This paper describes Shading Stage in ReVolver/C40, which shades to generated silhouette images.

Shading is a method to correct intensity at each pixel in a generated image with a light information, and gets the image more real and comprehensive. Shading stage computes that intensity with a *depth gradient shading* method. This is one of the *pixel approach*, which calculates the surface gradient and the intensity of a pixel with a *depth buffer*, a two dimensional array which has a *depth*, the distance from an eye point to a surface voxel at each pixel in a screen.

Depth gradient shading is simple and fit for high-speed processing because it calculates a gradient with only two depths at each end pixels of a short segment including a target pixel for both x-coordinate and y-coordinate. So, this method is applied for various machines specified for volume rendering.

Shading Stage includes three processes: receiving pixel value, shading, and image output. Sixteen processor elements which has a DSP as a kernel are placed for the shading process. Each processor element in charge of shading is called *Shading Unit.* The *Pixel Transceiver*, the Shading Units, and the *Video Signal Generator* take charge of pixel value receiving, shading process, and image output respectively. A Shading Unit has a *receive buffer*, a pixel value receive buffer and an *output buffer*, a frame buffer for image output.

Both receive buffer and output buffer have twice memory as much as required, so accesses of these three components(DSP, pixel transceiver, and video signal generator) never conflict.

This structure enabled shading stage to get the three processes independent from each other, and to work on pipeline at a frame unit.

In a prototype, TMS320C40(50MHz) is applied to DSP, 2 banks memory with high-speed SRAM are used as receive buffer, and multi-port memory specified for images (VRAM) is used as output buffer.

This shading stage turned out to be able to shade 20.2 frame images per second at its highest speed in a performance prediction with the prototype.

ボリュームレンダリング専用並列計算機
ReVolver/C40 のシェーディングステージ

目次

1　はじめに

ボリュームレンダリングによる画像生成法は、これまで医療画像の生成手法として重要な地位を占めてきた。さらに近年では、スーパーコンピュータなど高速な計算機を利用した構造解析や流体力学などの科学技術計算の結果が膨大に蓄積されつつあり、これらの計算結果の解析手法として、ボリュームレンダリングは注目を集めている。この手法は膨大な記憶容量と計算量を必要とするため、並列アルゴリズムの開発のみならず、専用計算機の開発も行われている。しかし、既存の専用計算機は平行投影が主流であり、物体の大きさを強調した表示ができなかった。

そこで当研究室では、遠近法を用いた高速表示が可能なボリュームレンダリング専用並列計算機 ReVolver/C40 を開発中である。ReVolver/C40 では、高速表示のために (1) 視線ベクトルの生成, (2) ピクセル値の計算, (3) シェーディング処理がパイプライン化された構成になっている。

本論文では、ReVolver/C40 において (3) のシェーディング処理と画像表示を行うシェーディングステージの構成について述べる。第 2 節においてボリュームレンダリングにおけるシェーディング手法について述べ、第 3 節において ReVolver/C40 の特徴・構成について述べる。第 4 節ではシェーディングステージで採用する depth gradient shading アルゴリズムとシェーディングステージの構成、そしてこのアーキテクチャにもとづくプロトタイプの実装について述べる。

2　シェーディング処理

2.1　用語の定義

ボリュームレンダリングとは、3 次元空間に配置されたデータを 2 次元のスクリーン上に投影することにより、3 次元空間を可視化する手法

である。この3次元データ空間をボリュームと呼ぶ。

ボリュームを立方格子状に分割し、離散的にデータを与えたものをボリュームデータと呼び、それぞれの格子点を中心とした単位立方体をボクセルと呼ぶ。各ボクセルの持つデータ値をボクセル値と呼ぶ。

ボクセル値としてボクセルの色や透明度などが挙げられるが、ReVolver/C40ではボリュームデータのメモリ容量を抑えるために、ボクセル値は1byteのインデックスとして与えられている。色や透明度の実体は属性テーブルに格納されており、このインデックスを用いて属性テーブルを参照後、色と透明度を得る。

2.2 既存のシェーディング処理

シェーディングとは、光源情報を用いて生成画像の輝度を補正することにより、画像に現実感を持たせたり画像情報の把握を容易にする手法である。

物体の表面上の1点における輝度を求めるには、その点における法線ベクトルと光線の入射ベクトルという二つの要素が必要となる。I_{in}を入射光の強さ、αを入射角、k_dを物体表面の拡散反射率とすると、反射光の強さI_dは次式で与えられる[5]。

$$I_d = I_{in} \cdot k_d \cdot \cos\alpha \tag{1}$$

なお、$\cos\alpha$は図1に示すように法線方向の単位ベクトル$\mathbf{N}$、光線の入射方向の単位ベクトル$\mathbf{L}$を用いて次のように表わされる。

$$\cos\alpha = \mathbf{N} \cdot (-\mathbf{L}) \tag{2}$$

ボリュームレンダリングにおいては、ボリュームデータそのものは表面の法線ベクトルに関する情報を持たないため、周りのボクセルの状態から法線情報を推定する必要がある。

この法線情報の求め方によって、シェーディング手法は以下のような二つのアプローチに分けられる [3]。

ピクセルアプローチ　スクリーン上の各ピクセルについて、視点から表面ボクセルまでの距離 (深さ, depth) を配置した深さバッファ(Depth Buffer) を作成し、それから表面の情報を獲得してシェーディングを行う方法である。以下の 4 種類の手法が提案されている。

- **深さシェーディング (depth-only shading):**　あるピクセルの輝度を求める際に、そのピクセルの深さのみを用いて求める方法である。周辺のピクセルを参照する必要がないため、並列処理は容易である [8]。

- **深さ勾配シェーディング (depth gradient shading):**　周辺のピクセルの深さを参照して勾配を近似する手法である。目的のピクセルを含む数ピクセルの区間領域を X/Y 方向それぞれについて定義し、両端の深さをのみを用いて勾配を近似する。この手法の詳細については、第 4.1 節で述べる [4, 11, 12]。

- **勾配補間法 (gradient interpolation):**　X/Y 方向について、深さ値が一定な区間ごとに分け、あるピクセルの勾配はそれが含まれる区間の中の位置と、隣接する 2 つの区間の深さの差によって定める [7]。

- **共勾配シェーディング (congradient shading):**　法線計算に 5 ピクセルの区間の depth 値を用いており、レンダリング前に depth 値の全ての組み合わせについて輝度を計算し、テーブルに格納しておく。レンダリング時にはテーブル参照のみで輝度が求まる [10]。

このアプローチの利点は、シェーディング処理が画像生成処理そのものと独立しているため、パイプライン化により高速化を図れる事にある。

逆に、ボリュームデータを一旦深さバッファに変換するため、求められた勾配の精度がボクセルアプローチの場合に比べ悪いという欠点を持つ。

ボクセルアプローチ　ボリュームデータから直接表面の勾配を求め、シェーディングを行う手法である。以下の 4 種類の手法が提案されている。

·ブロック/定数シェーディング **(Block/Constant Shading):**　それぞれのボクセルを一旦ポリゴンの立方体に変換し、既存のレンダリング技法を用いてシェーディングを行う [8]。

·文脈シェーディング **(contextual shading):**　隣接ボクセルの並びから局所的な法線を求め、目的のピクセルの周辺の法線を取って重みを付けて平均化することにより、滑らかなシェーディング画像を得る [8]。

·法線平均化 **(normal averaging):**　3^3 ボクセルの値から 法線を 81 通りのいずれかの値に近似し、あらかじめテーブルに持っておく [8, 13]。

·密度勾配シェーディング **(density gradient shading):**　ボクセル値が 2 値でなく、密度などの段階的な数値になっているボリューム空間において、近傍のボクセル値を比較することにより、表面となりうる面を求める [9]。

このアプローチの利点は、ボリューム空間を用いるため法線が正確に求められる事にある。法線の計算を前処理で済ませ、全ボクセルについて法線データをあらかじめ持っておくことにより、レンダリング中の処理速度の低下は起こらない。しかし法線データの分だけデータ量が増大するため、大きなボリューム空間をレンダリングする場合には実現困難である。

2.3 ReVolver/C40 におけるシェーディング処理

ReVolver/C40 では、512^3 という大きなボリューム空間を扱う。法線データをあらかじめ持っておく余裕はないため、第 2.2 節で述べたボクセルアプローチの手法を用いることはできない。

ピクセルアプローチの手法の場合、深さバッファはスクーンのピクセル数に依存し、ボリューム空間に依存しない。さらに、この深さバッファを設けることで、画像生成処理とシェーディング処理をパイプライン処理することが可能となり、高速化が図れる。

よって、ReVolver/C40 ではピクセルアプローチの手法を採用し、シェーディング処理を行う専用のシェーディングステージを設け、画像生成機構 (RCS, PCS) とのパイプライン動作を行う。なお、画像生成機構の処理速度に合わせて、シェーディングステージでは高画質と高速処理を合わせ持つ depth gradient shading を実装する。

3 ReVolver/C40

ReVolver/C40 は、512^3 ボクセルのボリューム空間を持ち、512×512 ピクセルの画像がレンダリングできるボリュームレンダリング専用並列計算機である。3 ステージのマクロパイプラインから構成され、各ステージについても並列処理により高速化を図っている。

特に、ピクセル値計算の並列化のために、ボリュームデータを分割し、多数のメモリバンクに格納している。視線上のボクセルを読み出す際には、図 2 (a) のように視線上で等間隔に読み出す場合は、コンフリクトのないバンク構成にするのは困難である。そこで、視線ベクトルの x,y,z 軸成分のうち、ある方向の視線について最も絶対値の大きい軸を主軸と定義し、図 2 (b) のように主軸上で等間隔に読み出す。この場合、主軸に垂直なボリューム平面でバンク分けすることで、コンフリクトのない

バンク構成になる。これを任意に視線に対してバンクコンフリクトをなくすように拡張するためには、ボリュームメモリを3重化し、x,y,z各主軸ごとに対応したバンクを用意すればよい。

以上より、任意の視線についてバンクコンフリクトのないメモリ構成を実現している[2]。

3.1 ReVolver/C40 の構成

ReVolver/C40は、3つのステージと、全体をコントロールするSystem Control Unitから構成される。これらは図3のように接続され、パイプラインで動作する。

ホストインターフェース(System Control Unit, SCU)　システム全体の制御を行い、ホストマシンから受け取ったコマンド・プログラム・ボリュームデータを適切なステージに伝達する。

視線生成ステージ(Ray Casting Stage, RCS)　スクリーン上の全てのピクセルについて、視点からピクセルを通過する視線ベクトルを生成し、ボリュームと交差する視線をピクセル値計算ステージへ送る。図4のように8台のPEで構成され、並列に動作する。

ピクセル値計算ステージ(Pixel Calculation Stage, PCS)　ボリュームデータを保持しているステージであり、RCSからの視線データに基づいてボクセル値を読み出す。ボクセル値は1byteのインデックス値なので、これを用いて属性テーブルを参照し、ピクセル値計算を行う。512台のPEがそれぞれ1平面分のボリュームデータを持ち、図5のように1次元に接続されている。

PE でピクセル値を計算する際、隣接するボクセル値を比較し、その差がある定値以上であればそこを表面を判定し、表面と視線との交点の座標を、ピクセル値とともに隣の PE へ送る。この座標値は、深さの情報としてシェーディングステージで用いられる。

シェーディングステージ(Shading Stage, SS) PCS からピクセル値と交点の座標を受け取り、交点の座標から depth 値を計算する。そして、これらの値をもとに depth gradient shading を行う。(第 4.3 節参照)

3.2 ReVolver/C40 での並列処理

ReVolver/C40 のパイプラインでは、さまざまな並列処理を用いている。最上位の並列処理はフレームを単位としたパイプライン処理であり、画像生成 ·シェーディング ·画像出力からなる。(図 6 参照) 画像生成は RCS, PCS が行い、シェーディングと画像出力は SS が行う。また下位の並列処理として、RCS のスクリーンのスキャンライン分割による視線生成処理の並列化、PCS の 1 次元に接続された PE によるピクセル値計算のパイプライン、そして SS におけるスクリーンの領域分割によるシェーディング処理の並列化が挙げられる。

4 シェーディングステージ

4.1 depth gradient Shading の概要

視線上のボクセル値を視点に近い側から調べ、ボクセル値が大きく変化する所をボリュームの表面とみなすことができる。スクリーンからこの表面までの距離を各ピクセルごとに求め、2 次元配列に格納したものを深さバッファ(depth buffer) と呼ぶ。depth gradient shading は、こ

の深さバッファから表面の法線を近似計算し、それをもとに各ピクセルの輝度を求める手法である。

ある点 P における法線ベクトルを求める場合、X 方向については図 7 の斜線区間の両端ピクセルについて depth 値を取り出し、depth 値の差と座標の差を使って勾配を近似する。Y 方向についても同様に勾配を近似し、2 つの勾配から法線ベクトルを求める。

座標 (x, y) のピクセルにおける depth を D(x,y) をおくと、このピクセルにおける X 方向の勾配は、以下のアルゴリズムで求まる。

4.1.1　X 方向の勾配を求めるアルゴリズム

勾配を求めるために使用するピクセルの範囲を w とする。たとえば、$x, x \pm 1, x \pm 2, x \pm 3$ の座標の 7 ピクセルを用いるなら、$w = 3$ となる。

図 8 のように、勾配の変化が単調増加/減少する場合は、$D(x-w, y)$ と $D(x+w, y)$ の差を取ればよい。しかし、図 9 の場合、同様に差を取ると勾配が平坦になり突起が再現されない。したがって、P における depth の凹凸をできるだけ正確に近似するためには、近似に用いる区間を選択しなければならない。図 9 の場合、$D(x+x_{min}, y)$ と $D(x+x_{max}, y)$ との差を取ることにより、突起の部分が再現される。

$$\begin{cases} \Delta Dx_{+}(i) = D(x+i, y) - D(x+i+1, y) \\ \Delta Dx_{-}(i) = D(x-i, y) - D(x-i-1, y) \end{cases} \tag{3}$$

とすると、x_{max} を求めるアルゴリズムは以下のようになる。

1. $j = 0$ とする。
2. $\Delta Dx_{+}(j) \neq 0$ ならば、$\Delta Dx_{+}(i) \neq 0$ かつ$\Delta Dx_{+}(i)$ の符号が変化しない i の最大値を求め、それを x_{max} とする。
3. $\Delta Dx_{+}(j) = 0$ ならば、j に 1 を加え、$j \leq w-1$ なら 2. に戻る。$j > w-1$ なら $x_{max} = w$ とする。

このフローチャートを図 10 にしめす。$x_{min}, y_{min}, y_{max}$ についても同様に求めることができる。

4.1.2　法線と輝度の計算

4.1.1 節で求められた値 $x_{min}, x_{max}, y_{min}, y_{max}$ について、

$$\begin{cases} d_x = D(x + x_{max}, y) - D(x + x_{min}, y) \\ d_y = D(x, y + y_{max}) - D(x, y + y_{min}) \\ p_x = x_{max} - x_{min} + 1 \\ p_y = y_{max} - y_{min} + 1 \end{cases} \tag{4}$$

と置いた時、P における法線ベクトルを$\overrightarrow{v_x} = (p_x, 0, d_x)$ と$\overrightarrow{v_y} = (0, p_y, d_x)$ の外積、すなわち$\overrightarrow{N_P} = \overrightarrow{v_x} \times \overrightarrow{v_y} = (-p_y \cdot d_x, -p_x \cdot d_y, p_x \cdot p_y)$ とする。

これを用いて、式 (1) ･式 (2) から点 P における輝度を求めることができる。なお、ReVolver/C40 では処理速度を優先させ、光源モデルは平行光線に限定し、拡散反射率はボクセル値にかかわらず一定としている。

4.1.1 節の w の値を大きくすると勾配は平均化されて滑らかになるが、大き過ぎると、本来離れているはずの物体がつながったように表示されてしまう。文献 [4] では、w の値は 3 が最も適当としている。

4.2　シェーディング処理の並列化

ReVolver/C40 の行う depth gradient shading は、図 7 にしめした 13 ピクセルの領域から法線ベクトルを求めて陰影を計算する。つまり、一つのピクセルをシェーディングをするためには、そのピクセルを含む 13 ピクセルの領域があればよく、メモリアクセスに局所性がある。

そこで画面を図 11 のように分割し、それぞれを 1 つの PE (以下、Shading Unit (SU) という) に割り当てることで、並列処理による性能向上が期待できる。

この時、各 SU 上には自分がシェーディング処理を施す領域に加え、図 12 のように上下左右各 3 ピクセル分の周辺領域の depth 情報を割り当てる。このように、depth 情報を一部重複して各 SU に割り当てることで、法線計算時の隣接 SU 間の通信を不要にしている。

4.3 シェーディングステージの処理

シェーディングステージの構成を 図 13 にしめす。中心部分に該当しシェーディング処理を行う Shading Unit(SU)、各ピクセルのデータを受け取り適切な SU に分配する Pixel Transceiver(PT)、ビデオ信号を生成する Video Signal Generator(VSG) の 3 つの部分で構成されている。

なお、シェーディングステージでは第 4.3.1 節で述べる Pixel Transceiver の内部構成の単純化のため、depth 値の周辺情報として各 SU が持つピクセル数は、3 ピクセル ではなく 4 ピクセルとなっている。

4.3.1 Pixel Transceiver の処理

PT は、画面上の各ピクセルの情報 (座標 ・色 ・深さ) を PCS から受け取り、次にその座標をもとにどの SU の担当であるかを計算し、その SU の持つメモリ (Receive Buffer, RB) へ書き込む処理を行う。

SS が PCS から受け取るデータの内容は、ピクセルの座標、ピクセル値、スクリーンからの深さ (depth) 情報 である。(表 1 参照) これらのデータは 1 ピクセルあたり合計 139bit になるが、PCS と SS は 16bit のデータ線で接続されているため、時分割されている。

そこで PT で行う処理内容は以下の通りになる。

1. データのバッファリング: PCS から送られてくるデータを一度で蓄え、16bit に分割されたデータからもとのピクセルデータを復元する。

2．データの型変換:　ピクセル値は PCS の内部精度である RGB それぞれ 16bit 浮動小数点形式で送られてくる。しかし、後述の VSG でアナログ信号に変換する際には RGB それぞれ 8bit で行うので、ここで 8bit 整数値に変換する。

3．SU の選択:　ピクセル座標をもとに、書き込むべき SU を選択する。ピクセル値は 1 ピクセルに対し対応する SU は 1 つであるが、depth 値は第 4.2 節で述べたように複数の SU が重複して持つ場合がある。図 14 の場合は 3 台の SU が重複して持つ。一般的に SU が 2^{2^n} 台、ピクセルの数が 2^{2^m} ピクセルの時、スクリーン上で座標 (X, Y) のピクセルが $\mathrm{SU}[n \cdot s_y + s_x]$ の座標 $(x+4, y+4)$ に対応するとした場合、s_x, s_y, x, y は以下にように求められる。

$$\begin{cases} s_x = 2^{n-m} \cdot X,\ s_y = 2^{n-m} \cdot Y \\ x = X \bmod 2^{m-n},\ y = Y \bmod 2^{m-n} \end{cases} \tag{5}$$

また、以下のいずれか一つの条件を満たすピクセルの場合、複数の SU に重複して配られる。

$$\begin{cases} x < 4 \text{ ならば、 左隣の } (x + 2^{m-n} + 4, y + 4) \\ y < 4 \text{ ならば、 上隣の } (x + 4, y + 2^{m-n} + 4) \\ x > 2^{m-n} - 4 \text{ ならば、 右隣の } (x - 2^{m-n} + 4, y + 4) \\ y > 2^{m-n} - 4 \text{ ならば、 下隣の } (x + 4, y - 2^{m-n} + 4) \end{cases} \tag{6}$$

4．Receiver Buffer のアドレス計算:　SU 内の Receive Buffer の書き込みアドレスを求める。RB のアドレス空間は図 15 にように表わされるので、座標 (x, y) のピクセルのアドレスは次の式で求められる。

$$\text{アドレス} = y \times (128 + 4 \times 2) + x = y \times (128 + 8) + x \tag{7}$$

5. Receiver Buffer への書き込み: 3. で選択した SU の 4. で計算したアドレスにピクセル値と depth 値を書き込む。複数の SU が重複して depth 値を持つ場合は、それぞれの SU に書き込む。つまり、同じ depth 値は最高 3 回書き込まれる。

4.3.2 Shading Unit の処理

Shading Unit(SU) は、ピクセルの色と深さの情報が格納された depth buffer をもとに、以下の処理を行う。

1. depth の計算: Recieve Buffer に書き込まれるデータは、第 4.3.1 節で述べたように入射距離、基本距離、交点座標の 3 種類のパラメータに分けられているので、これらのパラメータから depth 値を求める。入射距離を L_{in}、基本距離を l、交点座標を v とすると、depth 値 Dはこれらの値から次式を用いて計算できる。

$$D = L_{in} + l \cdot v \tag{8}$$

2. シェーディング処理: 1. で求めた depth 値を用いて第 4.1 節で述べた depth gradient shading を行う。

SU の構成を図 16 にしめす。シェーディング処理を行う DSP を中心に、PCS からピクセル値や depth 値などのピクセルデータを受け取るための Receive Buffer(RB) と、シェーディング処理を終えたピクセル値を格納する Output Buffer(OB) とで構成される。OB はディスプレイ画面に出力するためのフレームバッファの役割も兼ねている。これを 1 台の SU とし、複数台の SU で並列処理を行う。DSP の読み書きと PT からの書き込み、そして VSG への転送処理はそれぞれ並列に行うため、RB と OB はそれぞれ 2 バンク構成にし、フレーム単位のシェーディング処理が終わるたびにアドレス空間を切り換える。

このため、図 17 のように DSP による読み書きと PT から書き込みは異なるバンクに対して行なわれ、同様に DSP の読み書きと VSG への転送処理も異なるバンクに対して行われる。

以上の構成により、SU は PCS からのピクセル値の受け取り処理や VSG への転送処理から完全に独立し、シェーディング処理に最大性能を発揮することができる。

なお、depth gradient shading の他にもさまざまな並列画像処理を行うことができるように、各 SU は COM ポートを用いて画像処理に適したメッシュ状に接続されている。

4.3.3 Video Signal Generator の処理

Video Signal Generator(VSG) の構成を図 18 にしめす。CRT Controler はドットクロックや垂直/水平同期パルスを生成し、表示すべきピクセル値を持つ SU 内の OB に転送を要求する。要求を受けた SU では、該当するピクセル値を Video Data Bus(VDB) へ出力する。そして VSG が VDB を介して得たピクセル値を D/A Converter を通してアナログ信号に変換し、RGB コンポーネント信号としてディスプレイ画面に出力する。

ディスプレイのリフレッシュレートを 60Hz とし、ピクセル比 1:1 で 512×512 ピクセルを表示する場合、ドットクロックは 27.6MHz、1 ピクセルの表示期間は 36.3ns となる。また水平同期期間は 31.0μs、水平表示期間は 24.8μs、垂直同期期間は 16.7ms、垂直表示期間は 15.9ms となる。

4.4 ReVolver/C40 プロトタイプの実装設計

ReVolver/C40 の実用性を検証するため、規模を 1/4 としたプロトタイプの実装を行っている。SU のプロセッサとして Texas Instruments

社の DSP である TMS320C40 (50MHz) を採用する。

4.4.1 プロトタイプの規模

ReVolver/C40 は RCS, PCS, SS の 3 ステージによるパイプライン計算機であるので、各ステージでの処理速度のバランスがとれるような構成にしなければならない。ReVolver/C40 では PCS の処理が最も重いため、PCS の処理速度で全体の速度が決定される。

ReVolver/C40 プロトタイプでは PCS は 128 台構成であり、この場合に PCS が 1 フレーム分のピクセルを出力する時間は最速 68ms となる [1]。よって、SS では 68ms で 1 フレームのシェーディングを終えればよい。

一方、DSP のアセンブリ言語で depth gradient shading のプログラムを書くと、X·Y 方向それぞれ勾配の計算に 41 命令、ピクセルのロードストアを含めた輝度の計算に 69 命令、合計 151 命令必要である。一方、DSP の命令サイクルは 20ns であるので、1 ピクセルのシェーディングに必要な処理時間は 3020ns となる。

各ピクセルのシェーディング処理には依存関係がないので、512×512 ピクセル分のシェーディングは並列処理によって 68ms 以内に終えることができる。1 ピクセルのシェーディング時間から、具体的には 12 台以上の SU で並列処理すればよいことがわかる。

そこで、領域分割ならびに実装の容易さを考慮し、SS は 16 台の SU で構成することとする。

4.4.2 Pixel Transceiver(PT) の実装

図 19 にしめす通り、PT から $\overline{\text{SU\_Sel[0:15]}}$ の 16 本の信号線が各 Shading Unit に接続されている。PCS より受け取ったピクセル値の座

標に対応する Shading Unit に対し、この信号線をアクティブにすることで、ピクセル値の受け渡しを通知する。

PT の内部構成を以下にしめす。(図 20 参照)

アドレス計算回路　ピクセルの X·Y 座標から、式 (5) を用いてピクセルデータを渡す SU の選択とその RB 内アドレスの計算を行う。

SU の番号とピクセル値のアドレスを求めるには、式 (5) を用いるまでもなく、座標値 X·Y それぞれについて上位 2bit と下位 7bit を取り出すことで計算できる。しかし depth 値のアドレスは、SU 内の座標を (x, y) とすると、式 (7) より

$$\begin{aligned}\text{アドレス} &= y \times (128 + 8) + x \\ &= y \times 128 + y \times 8 + x\end{aligned}$$

で与えられるため、別途加算器が必要となる。このため、X·Y の座標と結果のアドレスを格納するためのラッチと、depth 値を計算するためのカウンタ ·加算器を用いて構成する。

float→int 変換回路　PCS から送られてくる 16bit 浮動小数点形式の値を 8bit 整数値へ変換する。浮動小数点形式の指数部で与えられる回数分だけ仮数部を右へシフトすることにより求められる。このため、浮動小数点形式の値と変換後の RGB の各値を格納するラッチと、シフタ、カウンタで構成される。

32bit 値復元回路　32bit のパラメータは 16bit ずつ 2 回に分けられているので、ここでは一旦ラッチで蓄えて元の 32bit の値を復元する。16bit のラッチを用いて構成している。

4.4.3 Shading Unit(SU) の実装

スクリーン画面の大きさは 512×512、SU は 16 台で構成するので、各 SU に均等に分割すると、1 台につき 128×128 ピクセルとなる。本

節では、SU の Receiver Buffer および Output Buffer の実装法について述べる。

Receive Buffer Receive Buffer は Pixel Transceiver と DSP の 2 箇所からアクセスされるので、2 ポートアクセスが実現なされていなければならない。しかし、Receive Buffer のアクセスの性質として、次に挙げる特徴がある。

- Pixel Transceiver 側のアクセスと DSP 側のアクセスは全く異なる領域に対して行われる。(第 4.3.2 節参照)
- アクセス頻度は、Pixel Transceiver 側/DSP 側共ほぼ同じである。

このため、Receive Buffer の 2 つのポートは対称でなければならない。この条件を満たす構成法として、

1. 2 ポートメモリを使う。
2. 2 つのバンクに分けてスイッチングを行う。
3. 1 つの高速なメモリを 1/2 のアクセス速度とみなして擬似 2 ポートにする。

が挙げられる。市販の 2 ポートメモリは小容量のものしかなく、Receive Buffer に要求される 512kbytes 弱の容量を満たすには多数必要なため、1. は実装上無理である。2. では 2 つのポートのアクセスが競合することは全くないが、DSP のデータバス幅に合わせてビット幅 32bit で実装した場合、バス制御のための 3 状態バスバッファが多数必要になり実装面積上困難となる。3. は、実装上は容易であるが、2 つのポートからのアクセスが競合した場合に、優先順位制御が必要となる。また、競合時に DSP 側のアクセス速度が低下する恐れがあり、シェーディング処理の

際にランダムアクセスが頻繁に起きる Receive Buffer には不適である。2. は 16bit 幅すれば実装可能である。32bit アクセスを行う時は 2 回アクセスする必要があるが、シェーディング処理の命令数による予測の結果、32bit アクセスによる処理時間の増加は全体において 2% 未満であるため、問題はない。

以上より、アクセスタイム 15ns の 1Mbit(128k×8bit) の SRAM を 2 個接続し、16bit 幅のノーウェイトメモリを構成する。

このメモリを図 21 (a) にしめすように 2 バンク配置し、それぞれ 3 状態バスバッファを通して DSP のデータバスの上位 16bit(D31-D16)、下位 16bit(D15-D0) に接続する。これより、16bit 幅のメモリ×2 バンク、あるいは 32bit 幅のメモリ×1 バンクとして使用することができる。また PT からのアドレスバスと DSP のアドレスバスは、図 21 (b) のように 2 つのセレクタを通して SRAM のアドレス線に入力し、制御信号によるスイッチングを実現している。

Output Buffer　Output Buffer も同様に、DSP と Video Signal Generator の 2 箇所からアクセスされるので、2 ポートアクセスが実現なされていなければならない。Output Buffer のアクセスの性質として、

- DSP 側のアクセスと VSG 側のアクセスは全く異なる領域に対して行われる。(第 4.3.2 節参照)
- アクセス頻度は、DSP 側は 3020ns ごとの散発アクセスであり、VSG 側は、単発のバーストアクセスであり、連続領域を 36ns のサイクルタイムで転送する。

このため、Output Buffer の 2 つのポートは非対称である。この特徴に適したメモリとして、マルチポートダイナミック RAM (VRAM) を使用する。VRAM は DRAM と同じ方法でアクセス可能な RAM ポート

と、高速なシーケンシャルアクセスが可能なSAMポートを持っている。Output Bufferの場合、DSPからの書き込みには高速性は要求されないためRAMポートを使い、VSGへの転送にはSAMポートを用いる。

以上より、OBには1Mbit(128k×8bit)のVRAMを4個接続し、図22にしめす通り32bit幅のメモリを構成する。

なお、このメモリ容量は必要最低限の容量の5.3倍となっているが、これはメモリ価格の低下により、大容量のメモリが現実的な価格で手に入るようになったからである。また、ビット幅がピクセル値のRGBに合わせた24bitでなく、DSPのデータバス幅に合わせた32bitを採用したのは、RBのように3状態バスバッファなどを付ける必要がないのでビット幅の拡張が容易であるためである。しかもこれらの拡張によってOBが画像表示用のフレームバッファとしての使い方の他に第4.4.4節で述べるような用途が考えられるためである。

また、VSGからのアドレスバスとDSPのアドレスバスは一旦ラッチに蓄え、VRAMのアドレス線へ入力している。VRAMのアドレス指定は、DRAMと同じくローアドレスとカラムアドレスに分けて行うため、上位9bitと下位8bitに分割し別々のラッチに蓄えている。

4.4.4 Video Signal Generator(VSG)の実装

CRTCでは外部から入力されるクロックをもとにドットクロックと垂直/水平同期パルスを発生させる。また、表示すべきピクセルの座標を計算し、そのピクセルを持つSUのOBに対し転送するピクセルの先頭アドレスを渡し、$\overline{\text{SU_Req[0:15]}}$をアクティブにして転送要求を通知する。

要求を受け取ったOBでは、要求されたアドレスを用いて内部のRAMポートから転送すべきピクセルの先頭アドレスをSAMポートに送り、シリアルクロックに従って順次Video Data Bus(VDB)に出力していく。このピクセル値は一旦CRTCを経由してD/A Converterへ流れて

いく。

第 4.4.3 節で述べたように OB のデータバス幅を 32bit とすることで、各ピクセルごとに RGB の他に α チャンネル (透明度) などの情報を持たせることができる。一方、OB は大容量の VRAM で構成するため、512×512 ピクセルのフレームバッファを最大 4 面持つことができる。これらの性質を利用して、VSG に以下の拡張機能を持たせることができる。

複数のフレームバッファの合成: 4 面のフレームバッファを、ピクセルごとに透過度を指定して合成する。ピクセル値の合成回路はモノクロの場合、次のとおりである。ピクセル値は 0(暗) ∼ 1(明) の範囲の値を持ち、α 値は 0(不透明) ∼ 1(透明) の範囲の値を持つとする。奥にあるピクセル値を p_b、手前にあるピクセル値とその透明度をそれぞれ p_f, α_f とすると、合成後のピクセル値 p_{out} は以下の式で与えられる。

$$p_{out} = p_b \cdot \alpha_f + p_f \cdot (1 - \alpha_f) \tag{9}$$

しかし、実装の際には次の問題点がある。

1. α 値を 8bit とした 256 段階の透過度合成では乗算器を用いた演算になり、処理時間がピクセル表示期間 (36.3ns) に間に合わない。
2. VRAM のシリアルサイクルが 30ns なので、ピクセル値を 1 本の VDB を介して受け取っていたのでは、合成のために 2 つ以上のピクセル値を受け取る時間がない。

これらの問題点は、以下の方法で解決できる。

1. 単純で高速化可能な演算方式を採用する。
   α 値を 2bit とし、25%、50%、75% の 3 種類の合成をサポートする。これより、合成回路が図 23 のように 3 個の加算器で実現できるため、CRTC の中に実装することが可能である。

2. 図24のように、Video Data Bus(VDB)を2本配置する。
   この場合SU0~7とSU8~15のピクセル値を同時に出力することができる。これらのピクセル値はCRTCに入力され、合成演算を行なったあと、D/A Converterへ出力する。

これより、ボリュームデータの上に文字やマウスポインタなどを重ねるなど、ボリューム空間の解析の助けとなるさまざまな情報を表示することができる。(図25参照)

表示面積の拡大: ディスプレイに同時に表示可能な面積を増やす。ただし、OB(VRAM)のシリアルサイクルが最短30nsなので、ドットクロックは最高33MHzとなる。この状態で表示可能な画面の最大サイズは、744×560ピクセルとなる。

フレームバッファの拡大: 4面のフレームバッファをそのまま2次元的に連結し、広い画面を構成する。この場合、1024×1024ピクセルの実画面のうち、実際に表示できるのは744×560ピクセルであるため、実画面の表示位置をずらしてスクロール表示を行う。

主記憶の拡大: バス幅がDSPのバス幅と同じ32bitなので、フレームバッファとしての用途の他に、表示しない部分を主記憶の一部として用いることができる。フレームバッファに512×512ピクセル領域を用い、さらに2バンク分使用した場合、使用領域は64kwords(256kbytes)となる。残りの256kbytesはプログラム領域や作業領域として用いることができる。

4.5 プロトタイプの基板設計

各 SU はサブ基板に実装し、メイン基板に搭載する。メイン基板には8枚までのサブ基板が搭載可能なので、メイン基板を2枚使用し、そのうち1枚に Pixel Transceiver と Video Signal Generator を実装する。

サブ基板とメイン基板の配線図を図26、図27にしめす。Shading Unit の Output Buffer と Video Data Bus 間の3状態バスバッファをメイン基板上に配置することで、サブ基板の部品点数の減少を図っている。

なお、サブ基板間はフラットケーブルで接続し、各 SU 間のメッシュ結合を行う。

5 おわりに

本論文において、ReVolver/C40 におけるシェーディングステージの構成と、そのプロトタイプの構成について述べた。

画像生成部 (RCS, PCS) とパイプラインピッチを合わせるために高速な depth gradient shading の手法を採用し、16台の Shadung Unit(SU) による並列処理を行っている。

また、depth gradient shading における法線計算では、周辺ピクセルの情報を必要とするため、あらかじめそれぞれの SU に重複して持たせておくことにより、SU 間の通信を不要にしている。各 SU が持つ Receiver Buffer(RB)/Output Buffer(OB) はそれぞれデュアルバッファ構成になっており、プロトタイプの実装上では RB は 高速 SRAM を用いた2バンク構成にし、OB は VRAM を使用している。

この構成により、ピクセル値の書き込み/シェーディング処理/画像表示は独立し、フレーム単位のパイプライン動作を行うことが可能となった。

光源モデルを平行光線に限定し、拡散反射率がボクセル値によらず一定とした場合、プロトタイプのシェーディング速度は1枚の画像の処理

に49.4ms、すなわち秒間20.2フレームである事がわかった。

今後は引き続きプロトタイプの実装を行い、完成後にその評価を行う。Shading Unitは汎用DSPを用いているため、DSPのプログラムを変更して平行光線の他に点光源やスポット光源を用いたシェーディングを実装し、処理速度や生成画像の評価を行う。またPCSから送られるパラメータを変更し、表面のボクセル値を受け取る事で物体ごとに拡散反射率を変えた画像のシェーディングを行い、同様の評価を行う。

なお、プロトタイプの実装状況は現在サブ基板(SU)とメイン基板(Pixel Transceiver, Video Signal Generator)の配線リストの作成が終わり、各部で用いるFPGA内部の設計を行っている。

謝辞

本研究の機会を与えて下さった、本教室教授富田眞治先生に深く感謝の意を表します。

数々の御指導、御助言を頂いた本教室中島浩先生、森眞一郎先生に心から感謝いたします。

本研究の共同研究者である、金喜都氏、對馬雄次氏、中山明則氏をはじめ、計算機システム講座の諸氏に感謝いたします。

また、この場をお借りして、筆者を支えて下さいました友人諸氏にお礼申し上げます。

参考文献

[1] 對馬 雄次: ボリュームレンダリング専用並列計算機 — *ReVolver/C40* —, 京都大学修士論文, 1995

[2] 明石 英也, 北須賀 輝明, 他: ボリューム·レンダリング向き並列計算機のアーキテクチャ, Technical Report 92-ARC-97-15, 情報処理学会計算機アーキテクチャ研究会, 1992

[3] Arie Kaufmann (訳: 藤代 一成), ボリューム·ビジュアライゼーション概論, PIXEL No. 116

[4] Toshiaki OHASHI, Tetsuya UCHIKI and Mario TOKORO: A Three-Dimensional Shaded Display Method for Voxel-Based Representation, EUROGRAPHICS 85, pp. 221-232, 1985.

[5] 鷺島 敬之, 西澤 貞次, 浅原 重夫: 並列図形処理, 並列処理シリーズ, コロナ社, 1991

[6] TMS320C4x User's Guide, Texas Instruments, 1993

[7] Bright, S. et al.: Shading of Solid Voxel Models, CGF, 5(2), 131-138, 1986

[8] Chen, L. et al.: Surface Shading in the Cuberille Environment, CG&A, 5(12), 33-43, 1985

[9] Chuang, K.S. et al.: Boundary Detection in Grey Level Scenes, Proc, NCGA '89 Conf., 112-117, 1989

[10] Cohen, L.T et al.: Real-Time Discrete Shading, TVC, 6(1), 16-27, 1990

[11] Goldwasser, S.M.: A Generalized Object Display Processor Architecture, CG&A, 4(10), 42-55, 1984

[12] Gordon, D. et al.: Image Space Shading of 3-Dimensional Objects, CGIP, 29(3), 361-376, 1985

[13] Tam, Y.W. et al: Display of 3D Medical Images, Proc. Graphics Interface '88, 78-86, 1988

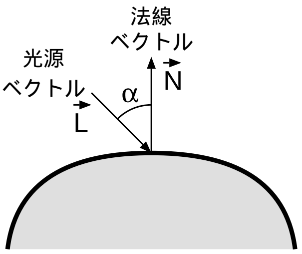


図 1: $\cos\alpha$ の計算方法

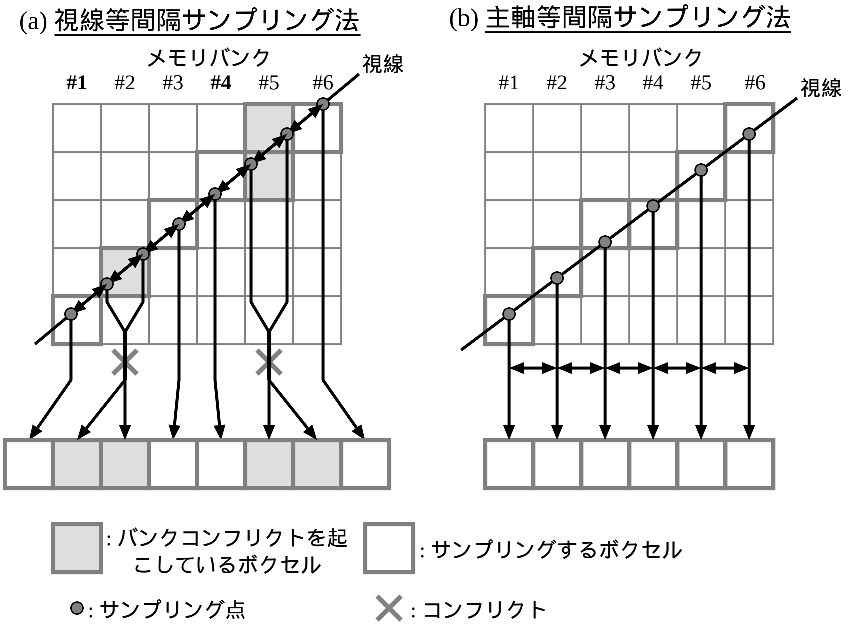


図 2: ボクセルのサンプリング

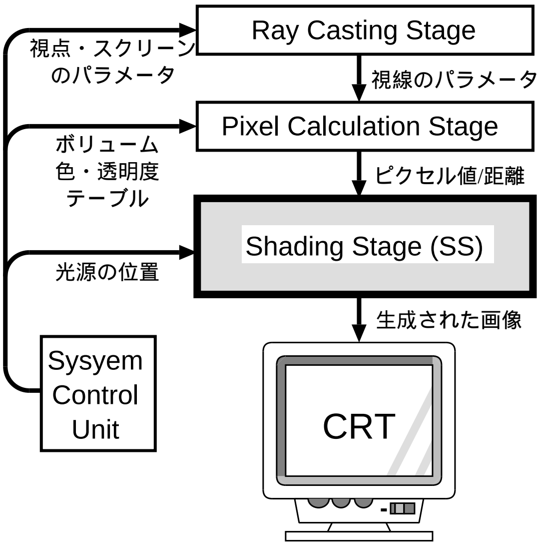


図 3: ReVolver/C40 の全体構成

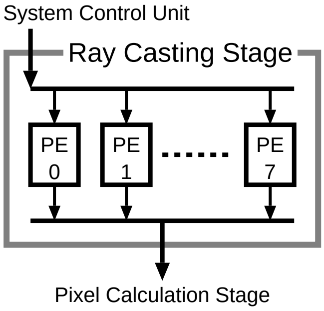


図 4: 視線生成ステージ (RCS) の構成図

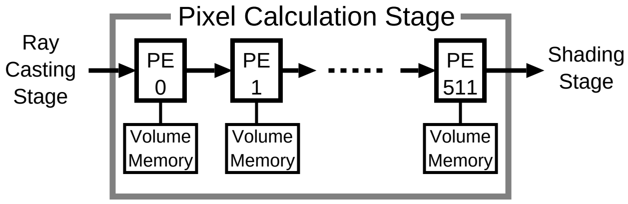


図 5: ピクセル値計算ステージ (PCS) の構成図

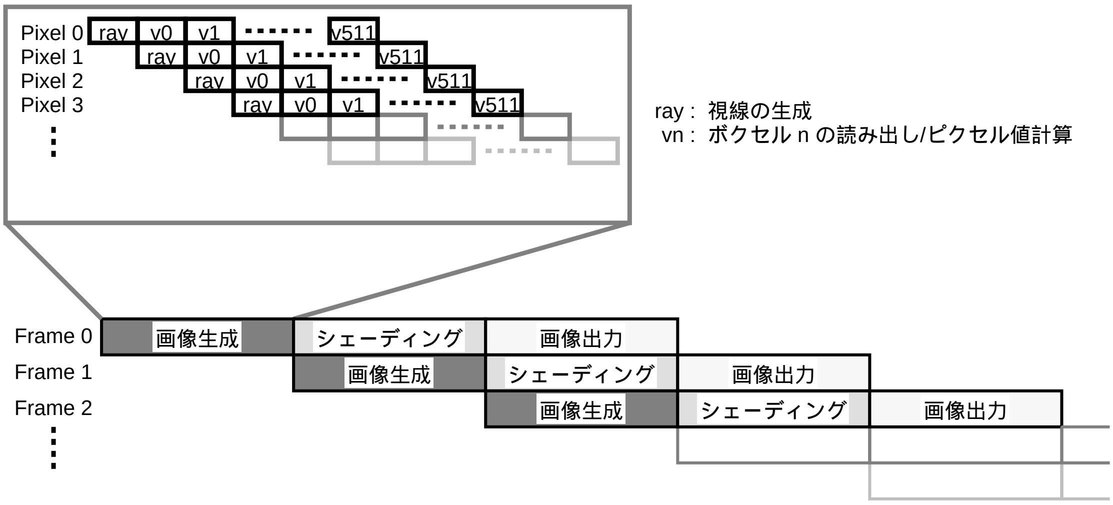


図 6: ReVolver/C40 のパイプライン処理

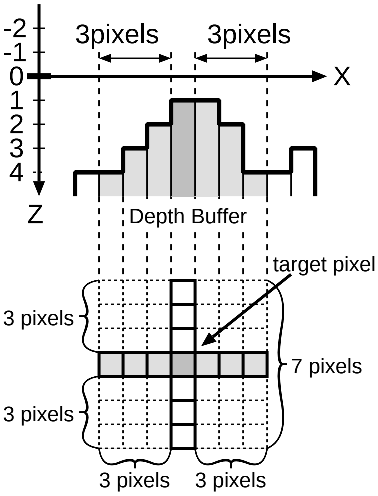


図 7: depth gradient shading の法線計算に必要なピクセル

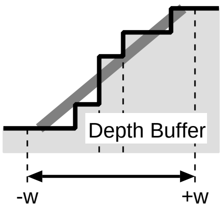


図 8: 勾配の求め方 (1) depth が単調増加/減少する場合

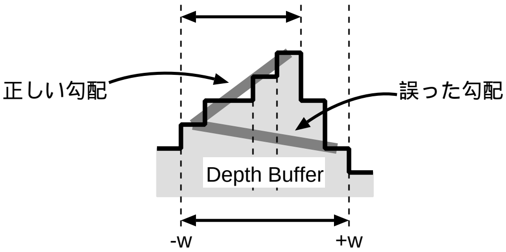
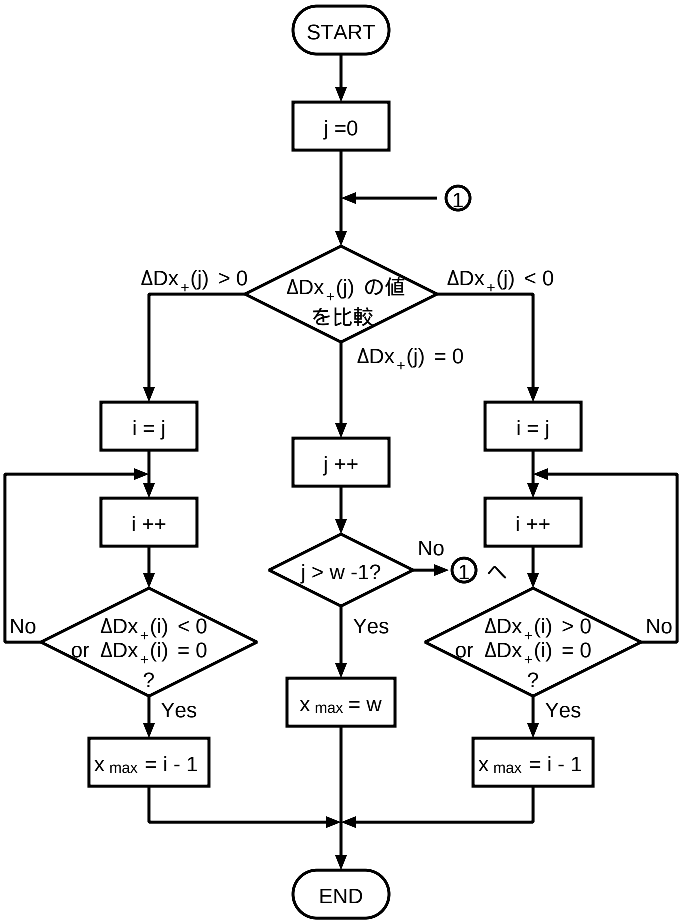


図 9: 勾配の求め方 (2) depth の増減が変化する場合



ただし、 $\Delta Dx_{+}(i) = D(x+i, y) - D(x+i+1, y)$

x_{min} を求める場合は、

$\Delta Dx_{-}(i) = D(x-i, y) - D(x-i-1, y)$ とする。

図 10: x_{max} を求めるアルゴリズムのフローチャート

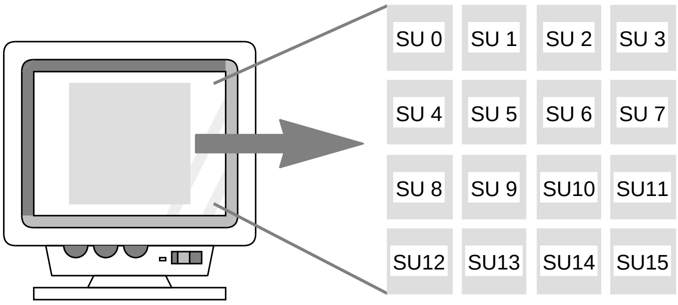


図 11: フレーム・バッファの分割

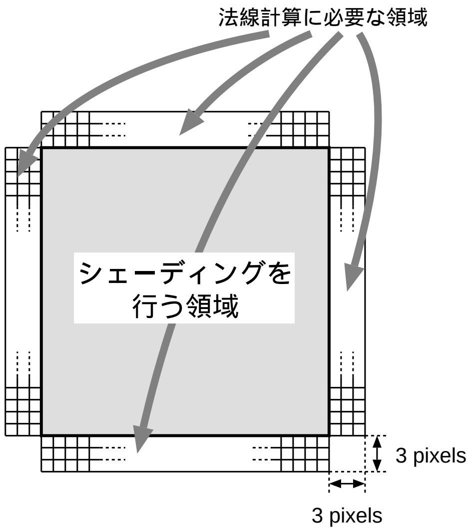


図 12: 法線計算に必要なピクセル値の範囲

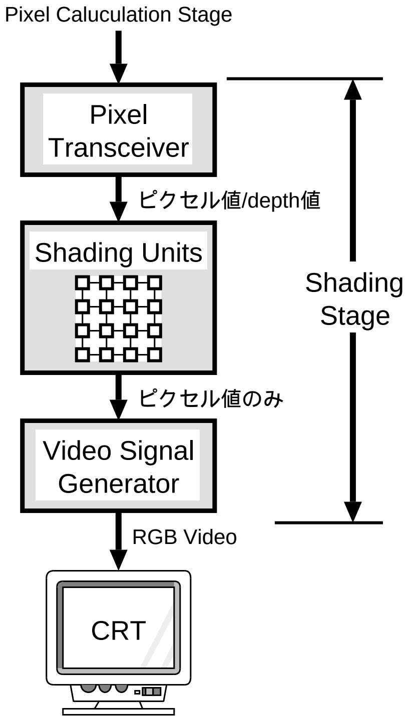


図 13: シェーディング ・ステージ (SS) の全体構成

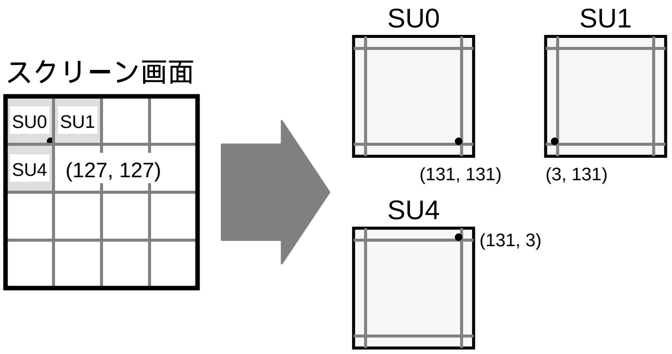


図 14: 複数の SU で共有されるピクセル

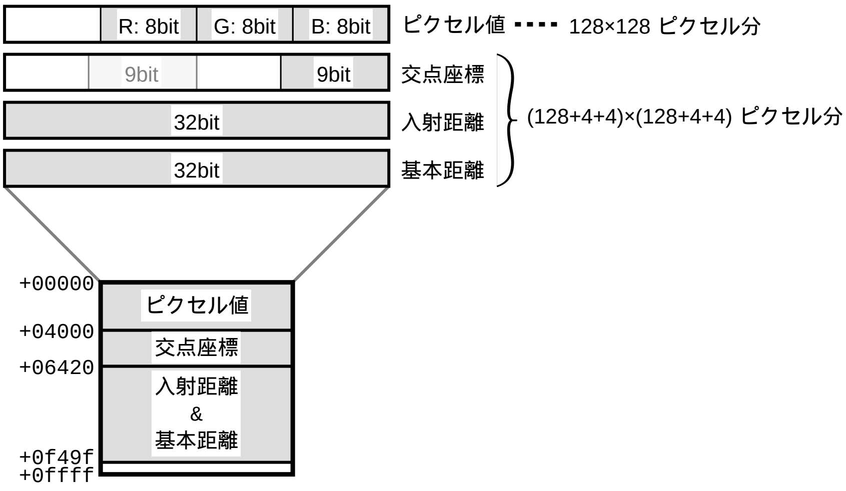


図 15: Receive Buffer のメモリマップ

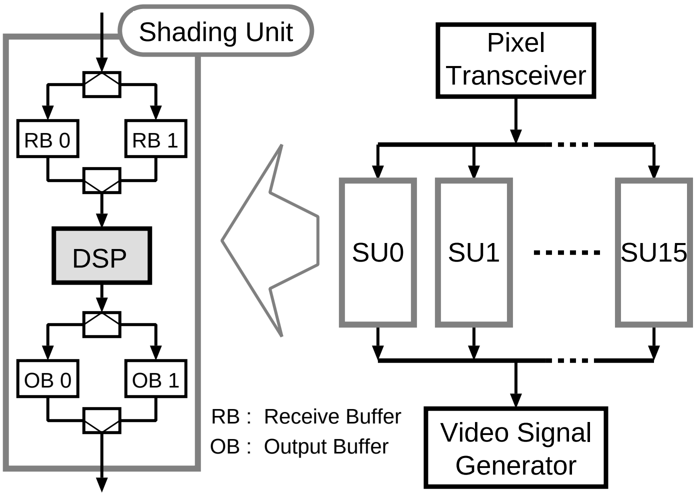


図 16: Shading Unit (SU) の構成

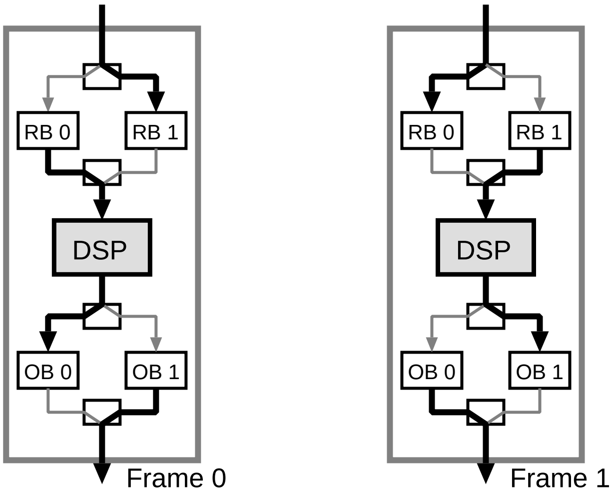


図 17: Shading Unit のデータの流れ

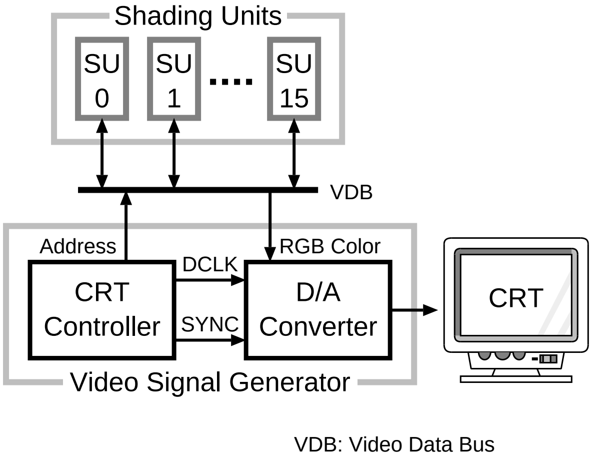


図 18: Video Signal Generator (VSG) の構成

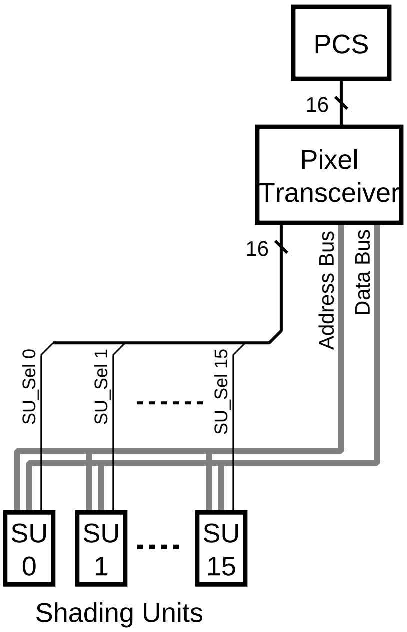


図 19: Pixel Transceiver (PT) の SU select 信号

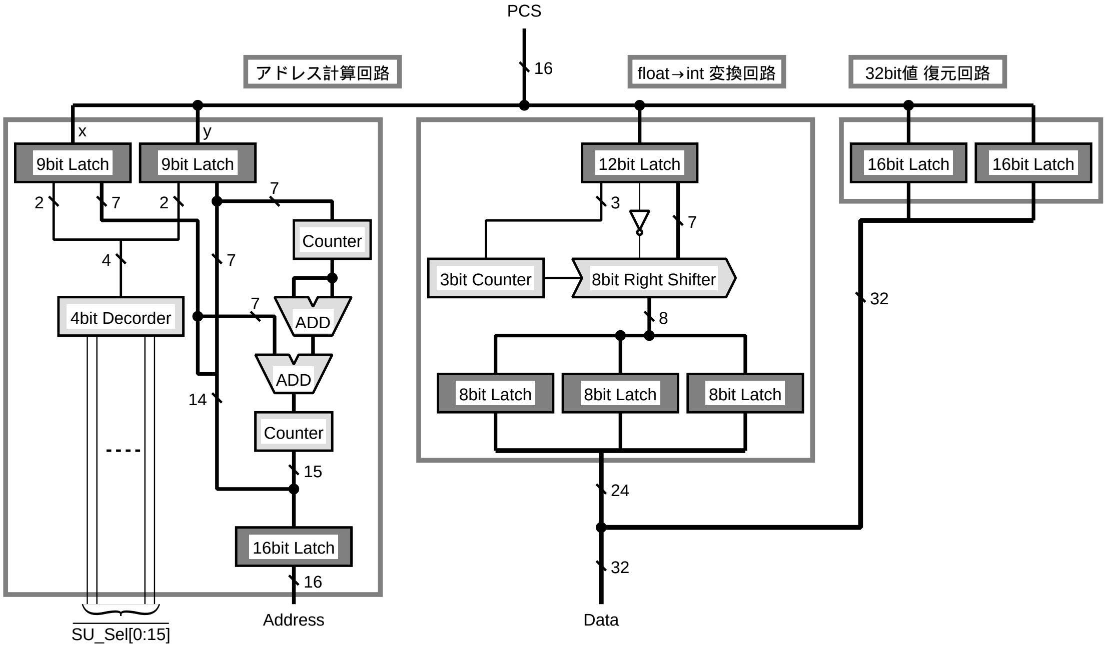
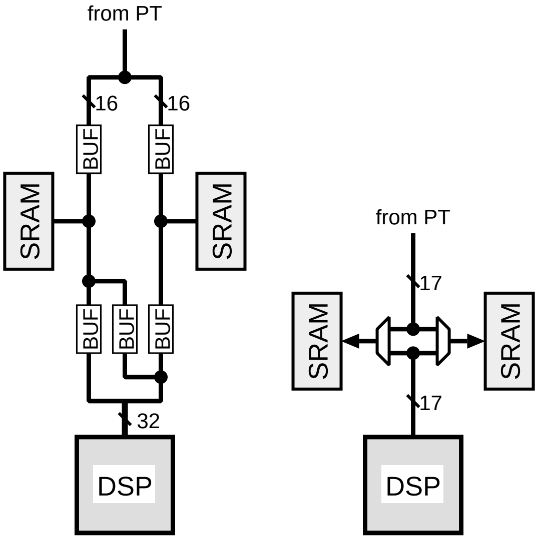


図 20: Pixel Transceiver (PT) の内部構成



(a) データバスの接続　(b) アドレスバスの接続

図 21: Shading Unit (Recieve Buffer 部) の接続

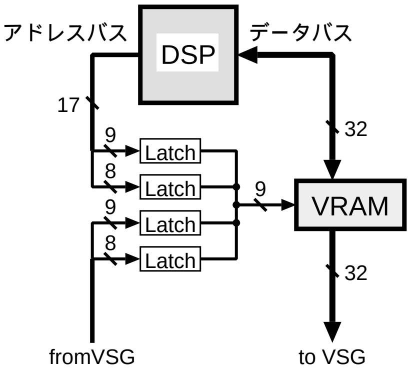


図 22: Shading Unit (Output Buffer 部) の接続

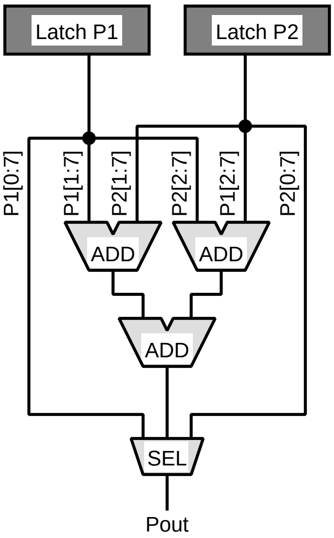


図 23: フレームバッファ合成回路

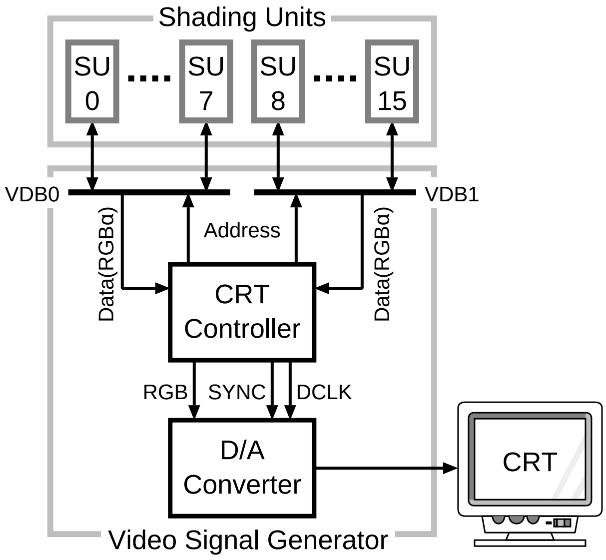


図 24: Video Signal Generator (VSG) のプロトタイプ実装時の構成

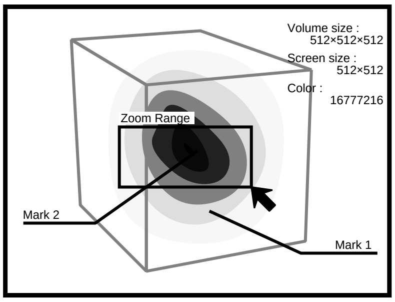


図 25: CRT 画像のイメージ

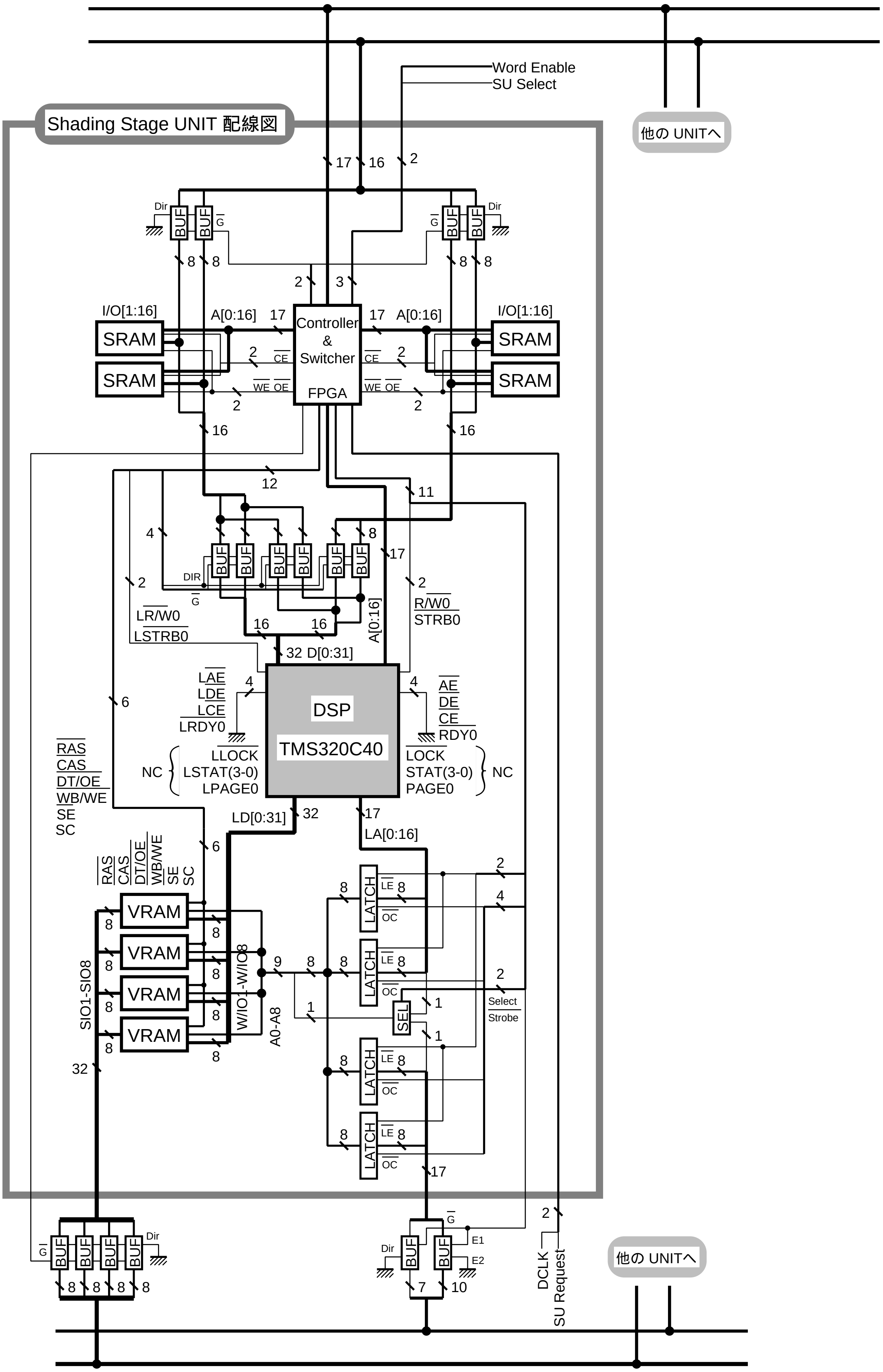
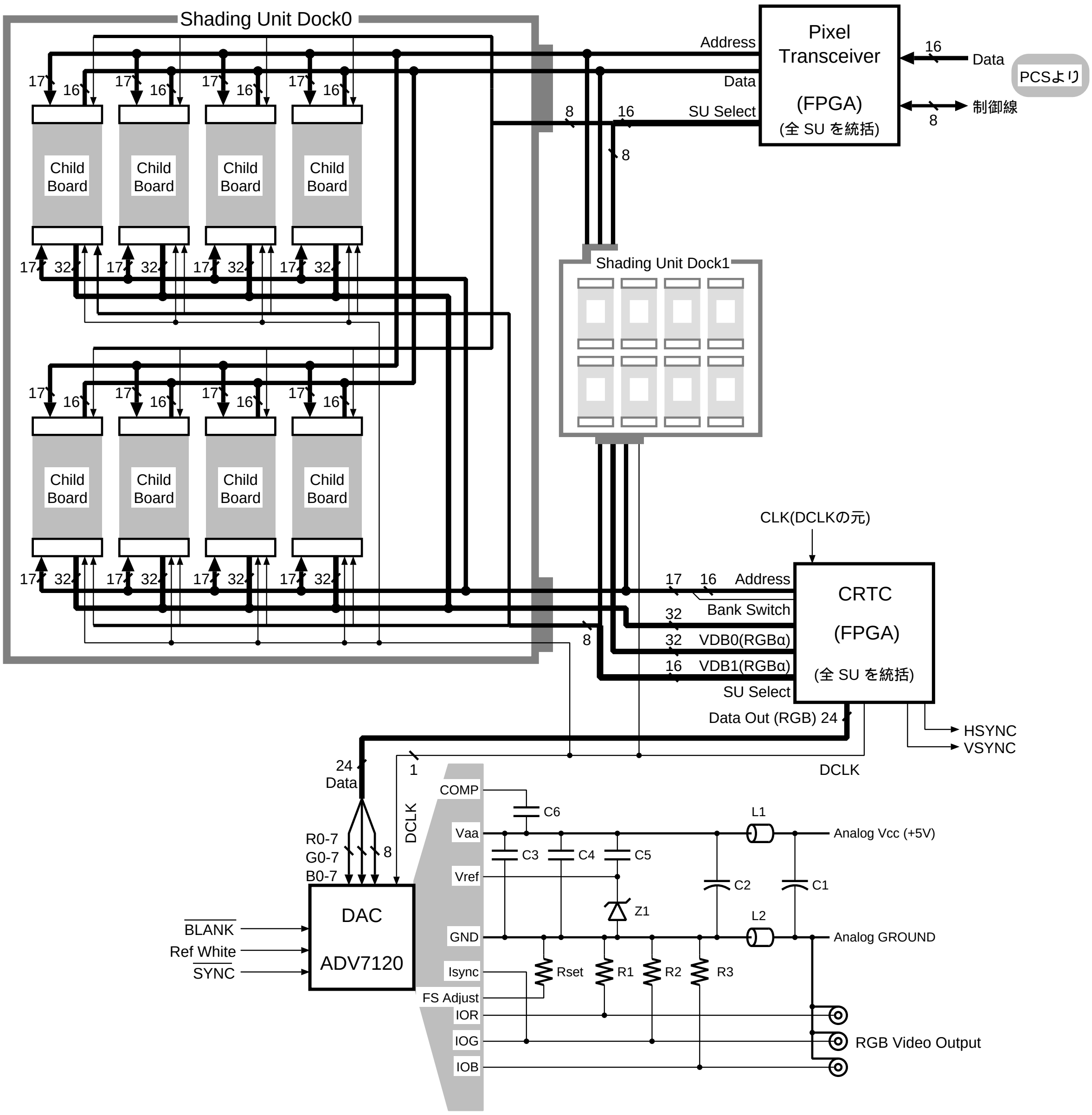


図 26: Shading Unit 基板の配線図



C1	33μF タンタルコンデンサー
C2	10μF タンタルコンデンサー
C3, C4, C5, C6	0.1μF セラミックコンデンサー
L1, L2	フェライトビーズ (Fair-Rite 274300111 or Murata BL01/02/03)
R1, R2, R3	75Ω 1% 金属フィルム抵抗 (Dale CMF-55C)
Rset	560Ω 1% 金属フィルム抵抗 (Dale CMF-55C)
Z1	1.235V Voltage Reference (参照電位) (Analog Devices AD589JH)

図 27: Shading Stage マザーボード基板の配線図

表 1: シェーディングステージ (SS) がピクセル値計算ステージ (PCS) から受け取るパラメータ

パラメータの種類	内訳	ビット数	数値の型
ピクセルの座標	x	9bit	int
	y	9bit	int
ピクセル値	R	16bit	float
	G	16bit	float
	B	16bit	float
深さ (depth) 値	入射距離	32bit	float
	基本距離	32bit	float
	交点座標	9bit	float

(int: 整数形式, float: 浮動小数点形式)