

マルチプロセッサ型汎用画像処理システム

- その2 制御法とソフトウェア -

Multi-purpose Image Processing System Using Multi-processors
-Part 2 System Control Sequence and Software -三ツ矢 英司
Eiji MITSUYA末永 康仁
Yasuhito SUENAGA奥平 雅士
Masashi OKUDAIRA河田 悦生
Etsuo KAWADA日本電信電話公社 横須賀電気通信研究所
YOKOSUKA Electrical Communication Laboratory N.T.T.

ABSTRACT

This paper describes an experimental image processing system having four processing units which has been developed to realize the Multi-processor layer of a Hierarchical Picture Understanding System (M-HIPUS). Various image processing programs developed on general purpose computers are easily transplanted onto M-HIPUS. Some high speed image processing techniques are employed in M-HIPUS, which include (1) look up table processing, (2) two dimensional memory access, and (3) functional data transferring.

1. まえがき

従来、各種の画像処理アルゴリズムが大型計算機もしくはミニコン等の汎用計算機で開発されてきている[1]。一方、画像処理を実用に供するためには、多くの場合、経済性、小型化、高速化の要求から所望の処理機能を有する専用の処理装置が必要になっている。最近では低機能のものは専用LSI、高機能の処理はソフトウェア処理により実現されることが多い。

この時、問題となるのは汎用計算機で開発された画像処理ソフトウェアをいかに効率よく専用装置上に移植するかである。従来の画像処理専用装置の多くは高速化を第一義として、ソフトウェアの移植性についてはあまり議論されていないように思われる。しかし、今後画像処理の高機能化は必至であり、画像処理装置の開発の効率化、短期間化をはかるためにはソフトウェアの移植性についても十分な考慮をしていく必要があると考える。

本論文ではこのような観点に立ち、現在検討を進めている階層型画像理解システム(HIPUS:Hierarchical Picture Understanding System)の概念[2]とソフトウェア移植に対する考え方、そしてこの考え方に基づいて試作検討を進め

ているマルチプロセッサ型画像処理システム[3][4]の高速化手法について述べる。

HIPUSの基本的な考え方は複数の高速化技術(並列化処理パイプライン処理、マルチプロセッサ処理)を階層的に配置し、前処理から認識・理解に至るまでの一環した画像処理プロセスを高速かつ効率的に実現しようとするものである。

また、今回試作した装置で実現している高速化手法についてはソフトウェアの移植性を阻害しないと言う点に最大の注意を払っている。具体的には演算の高速化を主目標とするLook Up Table処理(LUT処理)、画像処理向きメモリアクセス機能としての2次元メモリアクセス機能、データの形式を変換して処理アルゴリズムに最適なデータ形式を用意するための転送処理等である。

2. ソフトウェアの移植性と高速性

各種の画像処理を通常の汎用計算機(フォンノイマン型計算機)で行うことは、必ずしも効率的ではない。1次元メモリアドレス空間しか持たない汎用計算機で2次元データである画像を扱うことによる不整合、並列処理向きのアルゴリズムと逐次処理形式の計算機との整合性の問題等数

々の指摘がなされてきている[5]。しかし現実には従来から各種の画像処理プログラムが汎用計算機上で作成されてきている。

汎用計算機を用いた画像処理システムの最大の利点はソフトウェアを開発するための環境、すなわちプログラミング言語、エディタ、デバッガ等の機能が充実していることである。また、我々が最も身近に利用できるツールであることも重要である。そして、今後もアルゴリズムの開発・検討には汎用計算機を用いた画像処理システムが重要なツールとして位置付けられていくことは間違いないことであろう。

一方、画像処理技術を実用に供する場合、汎用計算機を用いたのでは、速度的にも、経済的にも引き合わないことが多い。そこで、高速化・経済化を目的として画像処理向きアーキテクチャを有する画像処理装置を開発し利用することになる[6]。

この時間問題となるのは、ソフトウェアの移植性である。図1に処理アルゴリズムの開発から実用機への投入に至るまでの流れの概略を示す。汎用計算機上で開発されてきた多くの資産である画像処理ソフトウェアをいかに効率良く専用機に移植できるかで専用システムの開発期間が大きく変る。このため専用機のアーキテクチャを処理の高速化のために特殊なものにすればする程、汎用計算機との間でのアーキテクチャ上の整合性がとりにくくなり移植の作業は困難になる。

ところで、画像処理技術のすべてがソフトウェアの移植を必要とするものではない。例えば、微分処理や平滑化、2値化で代表されるような画像の前処理技術と呼ばれるものの多くはアルゴリズムが確定したのも多く、これらの処理については、その機能を部品化して使えるような環境が半導体技術の進歩により現れつつある。各種の画像処理LSIがその例である[7][8][9]。

ソフトウェアの移植が必要となる部分は画像処理技術の中でも対象とする画像の種類や内容によりそのアルゴリズム、処理パラメータ等が変わるような処理に限られる。こ

れらの処理を実現するのに特殊なハードロジックや専用のアーキテクチャを利用していたのでは画像の内容が変わる度に、大きな変更が必要になる。ソフトウェア処理で実現した場合変更は極めて容易になる。また、これらのソフトウェアは先にも示したように汎用計算機で開発することが多い。このため、ソフトウェアの移植性に十分な注意を払う必要がある。

画像処理技術とそれに必要となるハードウェアの資質との関係を整理して、階層型画像理解システム(HIPUS: Hierarchical Picture Understanding System)を提案した[12]。HIPUSの基本は画像処理技術を3つの領域に分け、それぞれの領域に含まれる処理アルゴリズムに最も適した高速化技術を階層化して持つことによりシステム全体のスループットを向上させようとするものである。

表1にHIPUSにおける階層性とハードウェアアーキテク

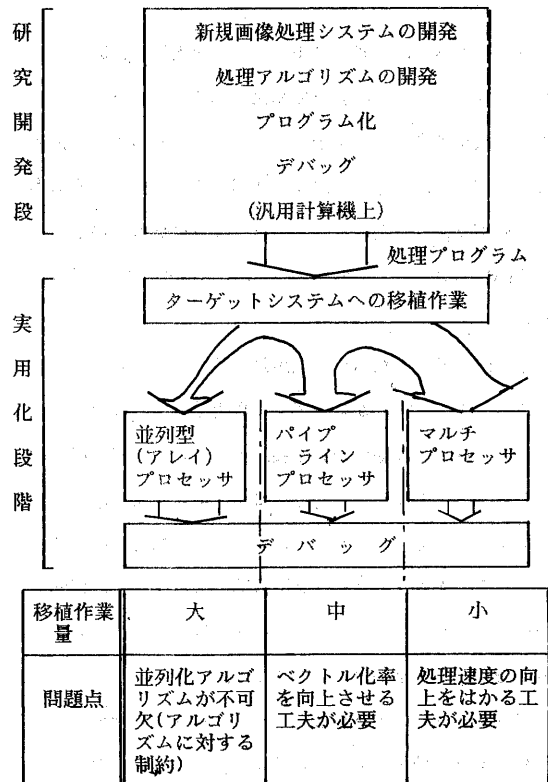


図1 ソフトウェアの移植過程と問題点

表1 HIPUSとその階層性および特徴

HIPUSの階層	階層の概要	処理の内容	特質
単一プロセッサ層	高性能の単一プロセッサで構成される階層	画像の内容から推論過程等も含め所望の結論を得て出力する	自然言語処理
マルチプロセッサ層	マイクロプロセッサ等を用いたマルチプロセッサ形式の階層	各種の前処理で得られたデータから画像の様々な特徴を得る領域分割、形状の検出、統計処理特徴点抽出	高級言語によるソフトウェア処理
アレイ・パイプラインプロセッサ層	アレイ構造もしくはパイプライン構造を持ち最も高速化される階層	画像データそのものから特徴抽出に必要なデータを求める平滑化、微分処理、2値化、ヒストグラム処理、細線化処理	LSI化による処理の部品化ファームウェア処理

チャの関係を示す。

最初に述べたソフトウェアの移植性に関する議論はマルチプロセッサ層以上の階層について着目したものであり、汎用計算機と同一のアーキテクチャを持つプロセッサユニットで構成するによりソフトウェアの移植性を高めようとするものである。

しかし、ソフトウェアの移植性に対しては有効なマルチプロセッサ構成のシステムには次のような問題点がある。

- (1)構成要素であるプロセッサユニットは基本的に汎用計算機と同一のアーキテクチャであるフォンノイマン型計算機であるため画像処理に対してフォンノイマン型計算機の持つ欠点をそのまま引き継いでいる。
- (2)プロセッサユニットとして現状のマイクロプロセッサを用いた場合 数値演算能力が低い。
- (3)画像データに必要となるビット処理機能が必ずしも充分ではない。
- (4)処理を高速化するためには基本的にプロセッサユニットの数を増やさなくてはならず、処理装置規模が大きくなる。
- (5)各プロセッサ間でメモリ共有するようなシステムではバス競合が生じる。

以上の内、(5)については本装置の主たる処理対象がリモートセンシングデータや設計図面等に代表される大型の画像データであり各プロセッサは画面を分割した形で処理を行うため、各プロセッサに対して十分なローカルメモリを持たせることでそれぞれ独立した動作が可能である。た

だし共有メモリーローカルメモリ間のデータ転送を効率良く行う手段は必要である。また、(1),(2),(3),(4)についてはソフトウェアの移植性を阻害しない範囲で補助手段を付加する必要がある。具体的には次章で述べるような方法が考えられる。

3. 高速化手法

ここでは、本装置で利用している画像処理の高速化、効率化のために適用した手法の内、特徴的なものについて述べる。

これらは、基本的に以下の3手法である。

- (1) 種々の演算処理を高速化するためのルックアップテーブル処理 (以下LUT処理と呼ぶ)
- (2) 2次元の画像データを効率良くアクセスするための2次元アクセス機構
- (3) 対象とする画像データの形式を処理アルゴリズムに適合させるためのインテリジェントデータ転送

3. 1 LUT処理

本装置においては処理に含まれる多くの演算処理、とりわけ数値演算処理をLUT処理により実現している。

LUT処理の概念は、図2に示すように与えられた入力変数に対応する出力値をテーブルの形式で保持しておき、入力変数の指し示すテーブルの特定の領域の内容を出力値とするものであり、近年、半導体技術の進歩によるメモリ価格の低下により種々の領域で利用されるようになってきている。

現在、LUTの最も一般的な例として知られているのは、画像表示のためのルックアップテーブルメモリであろう。表示用画像データを蓄積しているフレームメモリと画像表示系(D/A変換器)の間に挿入し、フレームメモリ中のデータを変更することなしに表示画像の輝度変換、色変換、さらには擬似動画表現を行うことができる。これらはLUTメモリの内容を変更することにより任意の入出力関係を実現できる特徴を利用したものである。

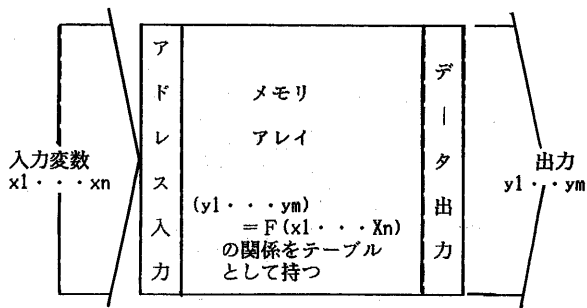


図2 LUTの概念

また、LUTによる演算処理の利点を以下に示す。

- (1) 処理速度が基本的に1回のメモリアクセス時間で済むため極めて高速に処理系を構成できる
- (2) LUTメモリとしてRAMを用いれば、演算系の修正、論理の変更がハードロジックで構成される系に比較して容易にできる。処理系の修正は基本的にメモリの内容を書き替えるだけで良い
- (3) 新しい処理系を構築する際にもハードウェアとしては標準化されており、効率が高い

一方、欠点としては入力される変数の増加によりLUT用メモリの容量が急激に増大することである。

LUTメモリの容量については以下のような関係が成り立つ。

今、入力変数の集合を $X=(x_i, i=1, 2, \dots, n)$

出力値の集合を $Y=(y_j, j=1, 2, \dots, m)$

として、 x_i, y_j 自身のとりうる状態数を $\langle x_i \rangle$ もしくは $\langle y_j \rangle$ とし各変数および出力値が他の変数および出力値に対して従属しないとすれば、LUTの入力線数 L_x および出力線数 L_y はそれぞれ

$$L_x = \sum_{i=1}^n \lceil \log \langle x_i \rangle \rceil \quad (1)$$

$$L_y = \sum_{j=1}^m \lceil \log \langle y_j \rangle \rceil$$

となる。ただし $\lceil \cdot \rceil$ は $\lceil \cdot \rceil$ 内の値以上の整数値の内最も小さい値を意味する。またLUTに必要な総ビット数 B は(2)式で与えられる。

$$B = 2^{L_x} \times L_y \quad (2)$$

もし入力変数間もしくは出力変数間に従属関係があればLUTのメモリ量は(2)式より小さくなる。

LUTの規模は L_x と L_y が与えられれば(2)式を用いて求めることができる。式からもわかるようにLUTの大きさは L_x により大きく左右される。今、 L_y として8、 L_x として16をとればLUTのメモリ容量は64Kbyteとなる。画像の場合、1画素当たりのデータ精度が自然画像の場合で8ビット程度であるとすれば、64Kbyte~128Kbyte程度のLUTであらゆる2変数関数が表現できることになる。また、2値画像に対しても近傍画素の状態に応じて処理内容が決められるような細線化等が有効に用いることができる。

さらにLUTの規模の上限については、現在のメモリ価格で考えても256Kbyte程度までのLUTであるなら、専用のハードロジックや演算回路に比べて経済的に処理系を実現できるものと考えられる。さらに今後のメモリ価格の低下を考慮に入れば、より大規模なLUTを使うことも可能である。

本装置においては1画像処理プロセッサ当たり1MbyteのLUT用メモリを配置して種々の演算処理に利用している。処理で行っている。

3. 2 次元アクセス機能

画像のような2次元データを通常の計算機で扱おうとする場合問題となるのは、計算機の1次元メモリ空間に格納された画像データを如何に効率良く2次元データとして扱うかにある。そのため2次元から1次元へのアドレス変換

系の効率化は重要な課題である。

図3に示すように画像データを $P(X, Y)$ としたとき、この画像に含まれる画素データ $p(x, y)$ をアクセスする場合、2次元座標 (x, y) は次式を経て1次元メモリ空間上の対応するアドレス A に変換される。

$$A = y \cdot X + x \quad (3)$$

通常の計算機ではこの変換をソフトウェアにより実現しているが、2次元空間上でランダムなデータアクセスを行った場合、1アクセス毎にこの変換処理を行う必要があり、アドレス変換によるオーバーヘッドが極めて大きくなる。また、アセンブラのような低水準言語で、この変換機構を記述しようとするとは煩雑であり、またプログラムも読みにくいものとなる。

このため本装置においては上記のアドレス変換機構をハードウェアとして持たせ、変換速度の向上とプログラムの記述性を高めている。

図4は本装置で用いたアドレス変換機構の概要であり、2次元座標のX軸、Y軸に対応するインデックスレジスタ FX, FY を設置し、次の2つの要素で修飾して1次元のメモリアドレスを生成している。

- (1) アクセス単位修飾
- (2) Xサイズ修飾

アクセス単位修飾はメモリをアクセスする際、画像データすなわち各画素の持つデータ長がバイト長のものである

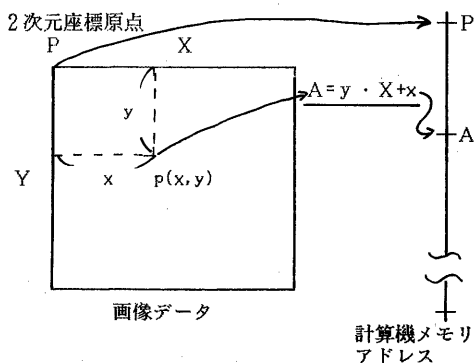


図3 2次元座標系の計算機メモリ空間への変換

か、ワード長のものであるか、さらにはビット長のものであるかを指定するものである。また、Xサイズ修飾は2次元データの一方の辺の画素数を規定するものである。理想的には任意の値にできることが望ましいがハードウェアの簡素化と通常扱う画像のサイズが2のべき乗であることから、本装置では256, 512, 1024, 2048の4種のサイズを用意しプログラムにより任意に設定できるようにしている。

本機能により2次元アクセス機能の効率化を図っている。具体例としてリスト1、2に本装置のアセンブラおよび通常の計算機のアセンブラにより記述した場合の画素配列KA(512, 512)の中の画素データKA(IX, IY)を読み出すためのプログラムの概要を示す。リストからもわかるように本装置では通常計算機で4ステップ程度必要となる処理を1ステップで実行することが可能であり、また記述性も高いものとなっている。特に、画素を任意方向に追跡しながら処理を行うような形式の処理プログラムはリスト1に示した2次元アクセス表現により極めて記述性が良好になる。

本装置によるプログラミング

<前提>

2次元インデックスレジスタ FX, FY
1次元レジスタ RO

配列 KA(512, 512)のメモリ上 P
での先頭アドレス

画素アドレスIX, IYはそれぞれFX1, FY1に格納されているものとする。

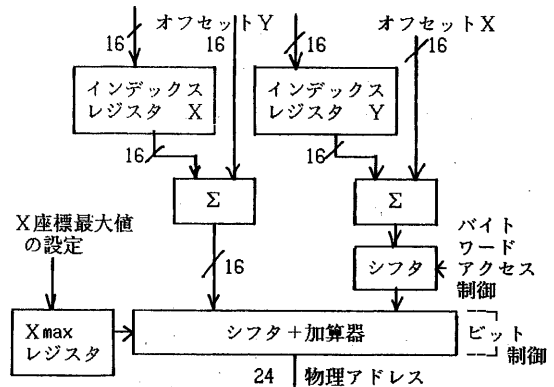


図4 2次元アドレス変換機構

<リスト1>

```
MOVW "0(FX1)"P/512(FY1), R0 ;read pixel data into R0
```

Xアドレス
 Xオフセット
 Yアドレス
 Yオフセット

ただしP/512は Pを512で除した値 (定数)

汎用計算機の場合(PDP11に準拠)

<前提>

汎用レジスタ R0, R1, R2

配列 KA(512,512)のメモリ上 P
での先頭アドレス

画素アドレスIX, IYはそれぞれR1, R2に格納されているものとする。

<リスト2>

```
ASH #9, R2, R2 ;shift left 9 times
ADD R2, R1 ;calculate pixel address
ADD P, R1 ;plus offset P
MOV (R1), R0 ;read pixel data into R0
```

3. 4 インテリジェントデータ転送

画像データは通常の数値データと異なり画素当たりのデータ長が2値画像では1ビット、白黒の自然画像では8ビット、カラー画像の場合は24ビットと画像の種類により様々であり、ビット処理、バイト処理等様々な形式の処理を行わなくてはならない。

通常の計算機においてはバイト長データが扱いやすいデータの最少単位であり、ビット処理は極めて不得意である。このため2値画像に対しても画素データをバイト長もしくはワード長として処理とすることが多い。一方、装置に入力される画像データの形式や表示装置に出力すべきデータ形式は処理する際に用いるデータ形式とは必ずしも一致しない。また画像データをメモリやディスク等に蓄積しておく場合には必要最小限の大きさにデータを変換しておくことが望ましい。

このため汎用計算機で作成されたソフトウェアを効率よく移植するためには、与えられる画像のデータ形式処理プログラムが前提としているデータ形式、出力するためのデータ形式、蓄積するための形式等の間の整合性を容易にか

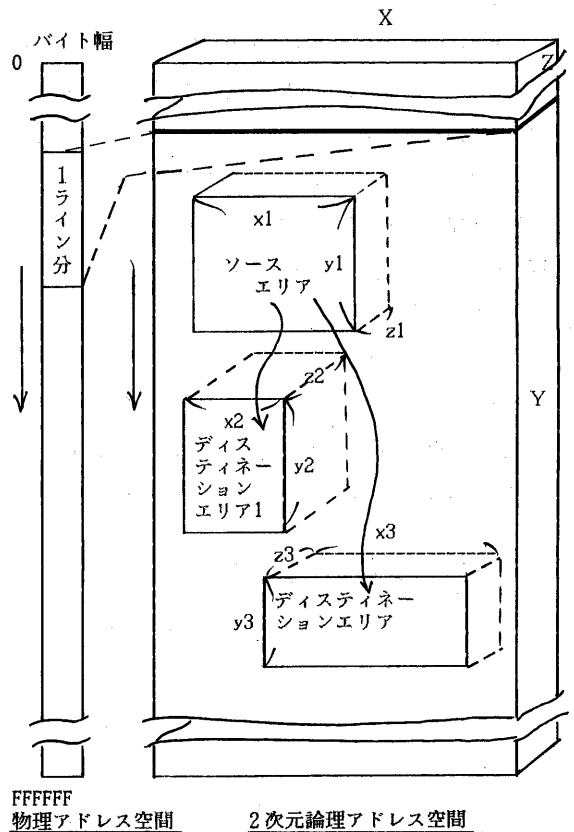
つ効率よくするためのデータフォーマット変換機能が不可欠である。

また画像の2次元性を保証した形式でデータを転送するためには画像の1部を切り出した矩形エリア間でのデータ転送も容易に実現できることが望ましい。

従来の装置ではこれらの処理をプログラムで実行する必要があった。これらは一見何の変哲もない処理であるため、装置設計上見逃されがちであるが、汎用計算機のプログラムで実現しようとすると意外に時間のかかる処理である。

本装置はこれらの処理を効率的に行うため、以下に示すデータ転送処理機能をハードウェアとして備えている。

- (1) 任意の2次元矩形エリア間でのデータ転送
- (2) データのフォーマット変換



データ転送条件 ; $x1 \cdot y1 = x2 \cdot y2 = x3 \cdot y3$
 $z1, z2, z3$; 1, 8, 16のいずれか
 X ; 256, 512, 1024, 2048のいずれかに設定可能

図5 2次元矩形エリア間でのデータ転送

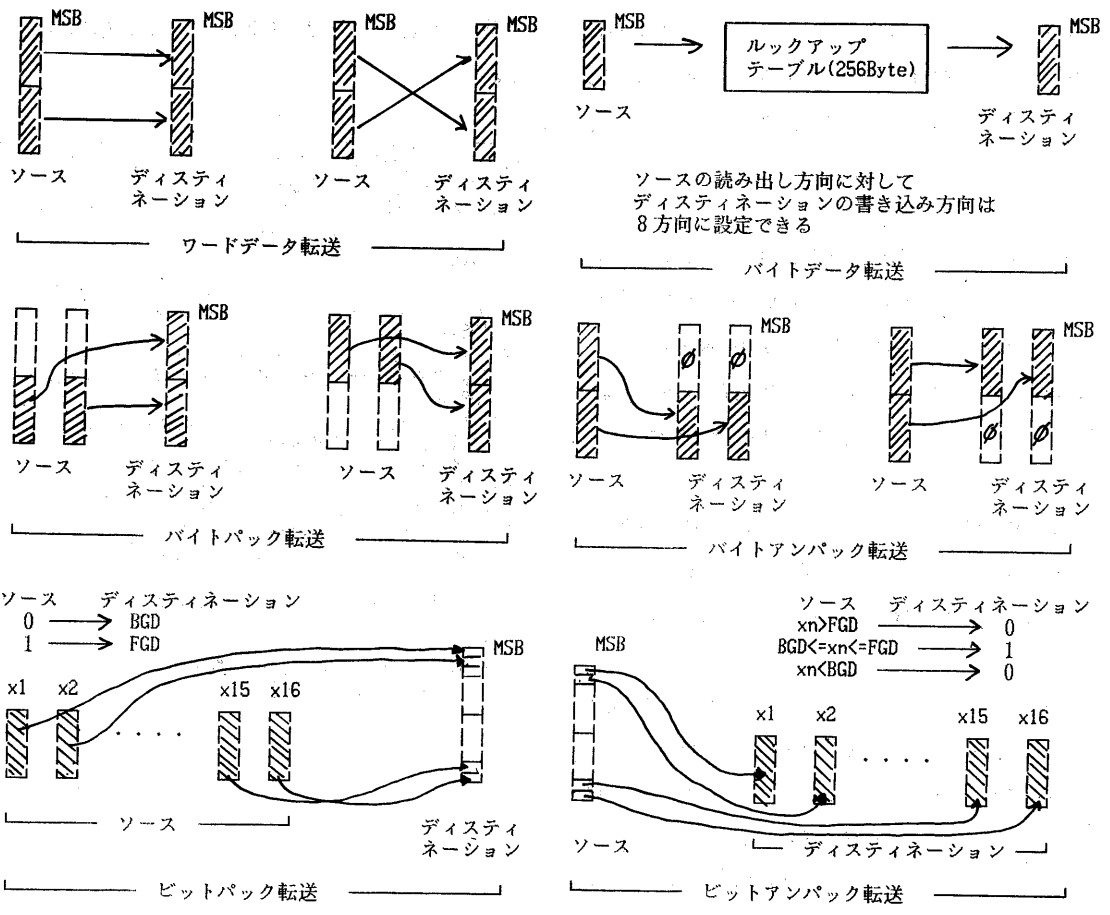


図6 転送部の転送モードの一覧

(3)濃度変換、2 値化、ウインドコンプレータ等データ変換機能

図5、6にこれらの機能の概要を示す。

4. 試作システムにおけるソフトウェアの作成

試作したマルチプロセッサ型画像処理装置の現時点における処理プログラムの作成過程は以下に行っている。

- (1) 汎用計算機で作成、デバッグの完了したソフトウェアについてLUT処理の可能な部分を抽出しLUT化を計る。ただし、元のプログラムがLUT処理を前提に作成されていればこの過程は省略できる。
- (2) 作成されたプログラムをコンパイルし、アセンブラ

コードを生成する。現在はコンパイラとして制御プロセッサ用のCコンパイラを用いているため、他の言語で記述されている場合、一旦、Cで記述する必要がある。

- (3) 生成されたアセンブラコードの内、2次元アクセスを行っている部分について、修正を加える。また、画像データの領域、各種変数領域の割当てを行う。
- (4) 本画像処理プロセッサ用のクロスアセンブラにより修正済みのコードをアセンブルする。
- (5) 生成されたマシンコードをプロセッサユニットにダウンロードし実行、デバッグを行う。

以上が基本的な手続きとなる。このうち(3)のパスだけ

は現在は人手に依っているが、比較的定型的な処理となるため、これも今後自動化が可能である。このため本画像処理プロセッサ用Cコンパイラを近く作成する予定である。

5. あとがき

現在、汎用画像処理システム研究の一環として試作検討を進めているマルチプロセッサ型画像処理システムについて、ソフトウェアの移植性に関する検討を踏まえて導入した高速化手法、(1) LUT処理、(2) 2次元アクセス機能、(3) インテリジェントデータ転送についてその概要を述べた。

LUT処理についてはマルチプロセッサの構成要素であるマイクロプロセッサの演算処理能力の向上を主目標としたものであり、極めて複雑な関数関係も容易に与えることができる。今後も、メモリの低価格化傾向は続いていくことが予想され、従来では考えられなかったような大規模なLUTも十分な経済性で実現できるものと思われる。

2次元アクセス機能については2次元データである画像を効率良く扱うためには重要な機能になると考えられる。今回の試作では通常の計算機における2次元アクセスの煩雑化を解消するレベルであったが、今後さらに高機能化する必要がある。

今後、画像処理技術は増々高度化していくものと予想される、このため画像処理の処理速度の向上に対する関心と同様の関心をソフトウェアの移植性に対して向けていく必要があると考える。

また、画像処理技術とこれを実行するためのハードウェア環境さらにはソフトウェア環境についても言語の問題も含めて整理していくことが重要になると考えられる。このためにHIPUSについて、さらに検討を深めるとともに、具体的課題として今回試作した装置のソフトウェア環境を整備していくなかでソフトウェア移植に関するより詳細な検討も行っていく予定である。

最後に、日頃、ご指導頂く安田 浩 画像応用研究室長にお礼を申し上げるとともに、本研究に当たり熱心な討論、

有益なご助言を頂いた画像応用研究室各位に感謝いたします。

(参考文献)

- [1] 「画像処理サブルーチン・パッケージ SPIDER」
電子技術総合研究所彙報 Vol.44 No.7,8 1980年
- [2] 三ツ矢, 末永 「マルチプロセッサ型汎用画像処理システム—その1 基本理念とハードウェア—」 本研究会
予稿
- [3] 三ツ矢 他; 「マルチ・プロセッサによる画像処理
装置の検討」 59年度信学会総合全国大会予稿 No.1432
1984年
- [4] 三ツ矢 他; 「マルチ・プロセッサによる画像処理
装置の構成」 59年度信学会総合全国大会予稿 No.1433
1984年
- [5] 木戸出, 坂上: 「パイプライン方式と完全並列処理方式
が増えた最近の画像処理装置」
日経エレクトロニクス 1982年 7.19号
- [6] 坂上, 木戸出: 「イメージプロセッサの最近の動向」
信学誌 Vol.67 No.1 1984年 1月
- [7] 福島, 小林他: 「画像処理用LSI-ISPのアーキテクチャ」
情報処理学会コンピュータビジョン研究会 26-6 1983年
9月
- [8] 川上, 島崎, 広上, 鈴木: 「事務処理分野をねらった2値
画像プロセッサ」日経エレクトロニクス 1983年 12.19
号
- [9] 松本, 黒川, 他: 「画像処理分野をねらったデータフロ
ー型プロセッサLSI」日経エレクトロニクス 1984年
4.9号