信号処理ＬＳＩ高速化に関する研究

山内寛紀

信号処理ＬＳＩ高速化に関する研究

山内　寛紀

目次

第１章　序論

ディジタル信号処理技術の理論的基礎は１９６０年代に確立され、１９７０年代後半より実際にハードウエア化が試みられるようになった。その当時リアルタイム化は、制御分野や電話帯域分野等、サンプルレートが数ＫＨｚ以下の領域に限られていたが、現在では、ＨＤＴＶ（High Definition TV）のテレビ信号までの処理が可能になってきている。そしてディジタル信号処理技術は、エレクトロニクスの全分野に普及してきている。この急速な発展は以下の３つの技術により進められてきた。第１はディジタル信号処理のアルゴリズム。第２はアルゴリズムを具現化するＬＳＩの高集積化技術。そして第３は両技術の橋渡し的役割を演じるディジタル信号処理ＬＳＩ（以下信号処理ＬＳＩ）の高速化技術である。

本論文は、信号処理ＬＳＩの高速化技術に焦点を置いている。各種の応用分野に応じて種々提案されるディジタル信号処理アルゴリズムの特長と、時代と共に急速に拡大しているＬＳＩ集積技術の特長とを、適合させて設計していく手法について研究したものである。以下、信号処理ＬＳＩの発展の経緯、本研究の目的、そして本研究の概要について述べる。

1.1　信号処理ＬＳＩの発展の経緯

信号処理ＬＳＩは、信号処理向けに演算能力を強化したＬＳＩの総称であり、多種多様なＬＳＩを含んでいる。これを機能の汎用性の点から見ると、ＮＴＳＣ（National Television Standard Committee）方式の多重化されたテレビ信号を、輝度信号と色信号を分離するＬＳＩや、８×８サイズの２次元ＤＣＴ（Discrete Cosine Transform）を行うＬＳＩの様な、布線論理の専用ＬＳＩから、汎用μＰの持つほとんどの機能を搭載し、自立的に動作できるプログラム制御の汎用ＤＳＰ（Digital Signal Processor）までの広がりを持っている。

これらを大まかに分類すると、（Ａ）布線論理の専用ＬＳＩ、（Ｂ）プログラム制御の専用ＤＳＰ、（Ｃ）プログラム制御の汎用ＤＳＰの３つに分けることができる。それぞれの特長と、本研究で開発したＬＳＩの例を表１.１.１に示す。（Ａ）から（Ｃ）へと汎用性を

表１.１.１　信号処理ＬＳＩの分類

信号処理ＬＳＩ	特長			構成	本研究での開発例
	高速性	汎用性	回路規模		
専用ＬＳＩ	○	×	小	・布線論理 ・単一機能またはその組み合わせ	・並列乗算器ＬＳＩ ・２次元フーリエ変換ＬＳＩ ・電子ビーム描画装置用ＬＳＩ
専用ＤＳＰ	△	△	中	・マイクロプログラム制御 ・特定の限定された領域で使用可	・ビデオ符号化ＤＳＰ ・幾何学変換ＬＳＩ
汎用ＤＳＰ	×	○	大	・マイクロプログラム制御 ・信号処理の広い領域で使用可	・音声信号処理ＤＳＰ

高めるに従い、ハードウエア資源を本来の演算機能以外に振り分けることになり、ＬＳＩ技術が同一ならば当然処理能力が低下する。このためＬＳＩ技術が未熟な段階では、性能を最優先して専用ＬＳＩを開発する。そしてＬＳＩ技術が進むに従い、類似装置に再利用できるマイクロプログラム制御の専用ＤＳＰが現われ、最後に種々の機能を盛り込んだ汎用ＤＳＰがその領域での標準品として開発され、広く一般に使用されるようになる。この状況を、図１.１.１に示す[1]。以下、音声信号処理用ＤＳＰと、ビデオ符号化処理用ＤＳＰに分けて、述べることとする。

音声信号処理では、１９７６年に開発された電話音声信号の簡単なフィルタリングや波形変換等を行う布線論理の専用ディジタル信号処理ＬＳＩが始まりである。その後７９年から８０年にかけて種々のフィルタ処理をマイクロプログラム制御で行う初期ＤＳＰが、４社から発表された[2]。そしてこのアーキテクチャを原形として、８２年には、３２K-ADPCM（Adaptive Differential PCM）符号化を始めとする、種々の専用ＤＳＰが開発されている。その後８５年には、ＬＳＩ技術が1.5μmとなり、２００～３００K素子以上を集積できるようになったので、この集積度を活かして、広く音声信号処理全体に使うことのできる処理能力の高い汎用ＤＳＰが種々開発されている。すなわち、音声信号処理で必要な、数十ＭＯＰＳ（Mega Operation Per Second）を上限として、ＬＳＩ技術の進歩と共に、専用ＬＳＩから専用ＤＳＰ、そして汎用ＤＳＰへと移ってきている。

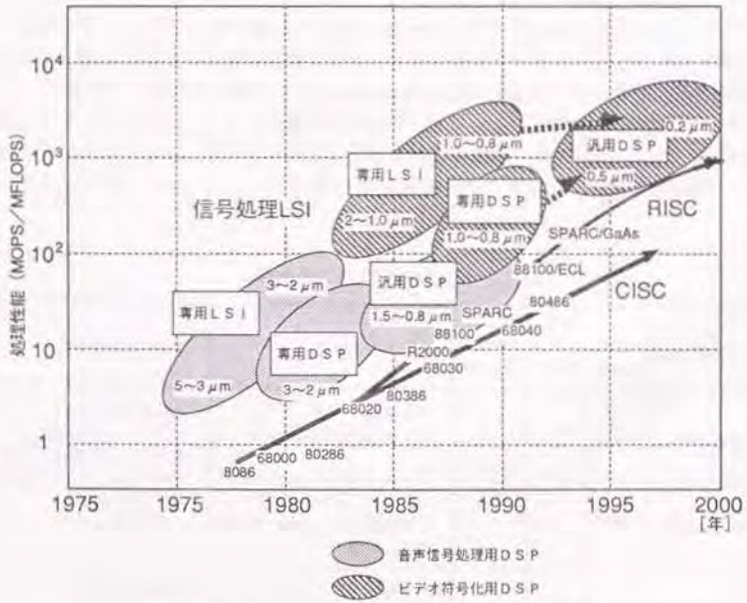


図１.１.１　信号処理ＬＳＩ処理性能の年次推移

一方ビデオ符号化処理では、８５年に発表されたテレビ信号（ＮＴＳＣ信号）を対象とした空間フィルタと色変換の専用ＬＳＩが始まりである。引き続いて８６年には、ＮＴＳＣ信号をフレーム内符号化により３２Ｍｂ／ｓに帯域圧縮する４品種の専用ＬＳＩファミリが、２μmＣＭＯＳ技術で開発されている[3]。さらに９０年には、ＣＣＩＴＴのH.２６１勧告に準拠した、テレビ電話・会議用の、フレーム内およびフレーム間符号化用の１３品種からなる専用ＬＳＩのチップセットが発表されている[4]。そして、ＬＳＩ技術がサブミクロン領域に近づくにつれ、H.２６１-ＣＯＤＥＣをプログラム制御の専用ＤＳＰで構成することが可能となり、８９年に、初期ビデオＤＳＰが２社より[5][6]、９１年から９２年にかけて、本格的なビデオＤＳＰが４社より発表されてきている[7][8][9][10]。後者はほとんどが0.８μｍＢｉＣＭＯＳ／ＣＭＯＳ技術で製造され、１ＧＯＰＳ（Giga Operation Per-Second）前後の性能を実現している。そして今後のディープサブミクロン技術の時代には、動画像処理用の汎用ＤＳＰが開発されてくる状況である。ビデオ符号化ＤＳＰの場合も、数ＧＯＰＳを上限として、ＬＳＩ技術の進歩に伴い、専用ＬＳＩから専用ＤＳＰ、そして汎用ＤＳＰへと移ってきていると言える。

この信号処理ＬＳＩ発展の歴史は、ＬＳＩ技術制約の基で、リアルタイム性と高機能性のトレードオフを適切に選択してきた結果であるといえる。単純な繰り返し処理が多くかつリアルタイム性が要求されるものは専用ＬＳＩ、種々の適応信号処理のようにプログラム制御でなければ実現できないものは専用ＤＳＰ、ディジタル移動機の高圧縮音声ＣＯＤＥＣのように多様で複雑なアルゴリズムに対処しなければならないものは汎用ＤＳＰと言うように、リアルタイム性，高機能性，ＬＳＩ技術制約の３者が最適に選択されてきている。そして信号処理ＬＳＩは、１０年でほぼ１００倍の性能向上を達成すると共に、汎用μＰと比較して、汎用ＤＳＰで１桁、専用ＤＳＰで1.５桁、専用ＬＳＩで２桁の高速性を維持してきている。

１.２　本研究の目的

前節で述べたように、信号処理ＬＳＩはＬＳＩ技術制約を解決して、リアルタイム性と高機能性を実現するのがポイントである。このＬＳＩ技術制約は、ＬＳＩ技術の発展と共に、集積規模とクロックスピードの両面で開放されてきている。しかし一方では、信号処理ＬＳＩの用途が、初期の簡単なフィルタ処理から、音声処理，静止画処理，動画像処理へと拡大してきているため、リアルタイム性と高機能性の両面、特にリアルタイム性への要求条件が、ＬＳＩ技術の発達以上に厳しくなってきている。すなわち、ＬＳＩ技術の実現力とアプリケーションからの要求条件との追いかけっこであるが、年々アプリケーションからの要求が、ＬＳＩ技術の能力を上回るようになって来ている。このため、ＬＳＩ技術制約を乗り越えるアーキテクチャへの期待は、いっそう強くなって来ている。

このような状況の中で、信号処理ＬＳＩアーキテクチャの技術課題は、リアルタイム性，高機能性，ＬＳＩ技術制約の３点であるといえる。著者は１９７０年末よりこの課題に取り組み、リアルタイム制御，音声符号化処理，画像処理，ビデオ符号化処理の各分野で信号処理ＬＳＩのアーキテクチャを研究し、実際にＬＳＩを開発してきた。そして一貫して上

記３点の研究課題を検討し解決してきた。各研究課題の中で取り組んだ研究項目を、表1.2.1に示す。

第１の研究課題である「リアルタイム性」については、高速処理の２つの方向から研究した。ひとつは演算器そのものを高速化して演算サイクルを上げていく方向であり、そのキー技術である「高速演算回路構成技術」について検討する。もうひとつはＤＳＰアーキテクチャの面から処理能力を上げていく方向であり、特に信号処理演算の特長を活かした「並列処理技術」について検討した。

第２の「高機能性」では、信号処理ＬＳＩの最大の特長である、「データメモリのアドレス生成技術」について研究した。理想的なアドレス生成は、信号処理のベクトル演算に対して、パイプライン演算を途切れなく実行させることである。このため、専用と汎用、または音声処理と画像処理等、適用領域毎に信号処理の特長を整理して、用途に合わせた最適な構成法を検討した。

第３の課題である「ＬＳＩ技術制約」は、演算サイクルと、集積可能な回路規模の両面があるが、前者は第１の課題で検討したので、ここでは後者の集積規模限界を乗り越える技術について研究した。具体的には、回路技術の点から「演算器の回路規模を削減する技術」を、アーキテクチャの点から「オンチップのメモリ容量を削減する技術」を検討した。

表1.2.1　研究の課題

課題	研究項目	
	大項目	小項目
リアルタイム性	高速演算回路構成技術（３章）	並列乗算器構成技術 浮動小数点演算器構成技術
	並列処理技術（５章）	パイプライン制御技術 ＳＩＭＤ並列処理技術 マルチＤＳＰ構成技術
高機能性	データメモリのアドレス生成技術（４章）	音声符号化ＤＳＰのアドレス生成技術 ２次元フーリエ変換のアドレス生成技術 ビデオ符号化ＤＳＰのアドレス生成技術
ＬＳＩ技術制約（６章）	演算器の回路規模削減技術	冗長２進演算器の回路規模削減 並列乗算器の回路規模削減
	オンチョプメモリ容量削減技術	プログラムメモリ容量の削減 データメモリ容量の削減

本研究は、上記３課題を解決し、信号処理ＬＳＩの高速化技術を確立することを目的とした。そして本技術を用いて実際に種々のＬＳＩを開発し、技術の有効性を実証した。

1.3　本論文の構成

本論文は、１９７９年から１９９０年までの１２年間の研究成果をまとめたものであり、以下の７章から構成されている。以下に各章の概要を述べる。第１章（本章）では、ＬＳＩ技術と信号処理ＬＳＩ発展の経緯を説明し、本研究の背景を明確にした。さらに本研究の目的について述べ、本論文の構成を示す。

第２章では、1.2節で述べた信号処理ＬＳＩ高速化の３課題をより詳細に論じ、その解決の方針について述べ、本研究の思想的背景を明確にする。そして、高速演算回路構成技術，データメモリアドレッシング技術，並列処理技術，ＬＳＩ技術制約の解決技術の４点を研究項目として明確化する。

第３章では、演算回路の高速化の観点から、信号処理演算の速度を律則する代表的な２種の演算器を取り上げ、それぞれ高速化手法を提案する。また具体的に試作してその効果を実証する。

（3-A）並列乗算器の高速化

乗加算は信号処理の最も基本的な演算であり、特に「２の補数」による固定小数点並列乗算器の高速化は、信号処理ＬＳＩ全体に共通する普遍的な課題である。
まず乗算器構成技術の点から、高速化の手法と拡張乗算の手法について検討する。後者は、小規模の並列乗算器を単位として、アレイ状に並べるか、または繰り返し使用することにより、任意語長の並列乗算器を実現する手法である。これは、チップの回路規模制限のもとで高速化するひとつの方向である。具体的に８×８ビットの乗算器チップを設計し、ｍ・ｎ個アレイ状に並べて８ｍ×８ｎビットの並列乗算器を構成できることを示す。
次にＬＳＩのインプリメント技術として、高速回路技術と高精度デバイスパラメータ評価技術について検討し、実際に設計・試作して評価する。高速回路技術では、バイポーラ論理を取り上げ、最も高速な２つの回路である、ＮＴＬ（Non Threshold Logic）とＬＣＭＬ（Low-level Current Mode Logic）の２種についてそれぞれ検討する。ＮＴＬについては、ゲートの過渡特性を決定するスピードアップコンデンサ容量の最適値を計算する手法を提案する。そして回路シミュレーションにより本手法の精度を定量的に評価し、ＮＴＬゲートの設計法として一般化する。ＬＣＭＬについては、レファレンス側のトランジスタに安定化容量を挿入し、電位の過渡変動を抑制することによって高速化する方法を検討する。回路シミュレーションと試作との対応を取り、最適設計手法として一般化する。高精度デバイスパラメータ評価技術では、エバース・モルモデルに基づき、ＴＥＧ（Test Element Group）チップ上のバイポーラ素子から特性を測定し、最小２乗法によりデバイスパラメータを自動的に抽出する手法を提案する。そして、ＳＳＴ（Super Self-aligned Technology）技術による微細

トランジスタに適用して、本手法の高精度性を実証する。

最後に、上記の高速化技術とＳＳＴ技術とを融合して開発した、８×８ビットの並列乗算器の試作例とその乗算時間の高速測定法を示す。高速性を実証することにより、それぞれの高速化技術の有効性を明らかにする。

（3-B）浮動少数点演算器の高速化

ダイナミックレンジの広い浮動小数点演算は、ユーザインタフェースの点からは理想的な処理系であるが、高速性と回路規模の点で固定小数点に大きく劣っている。特に高速化は最重要課題であり、回路規模をそれほど増大させないで演算速度を改良することが強く求められている。ここでは、高速フーリエ変換ＬＳＩを始め多様な用途のある、パイプライン乗算器とパイプライン加減算器を取り上げ、構成法の点から検討する。

乗算器では、異常処理を含めてパイプラインの切れ目を最適化して高速化する手法を提案する。また最終段の２入力加算器の構成法として、ＣＬＡ（Carry Look-ahead Adder）とＣＳＡ（Carry Select Adder）とを、速度，回路規模，検証ベクタ数の点から比較検討し、一般にＣＳＡが有利であることを定量的に示す。

加減算器では、２つの新しい高速化技術を提案する。第一は、指数部の減算と仮数部の桁合わせシフトとを並列に実行する手法であり、パイプライン演算器の場合に特に効果的である。第二は、仮数部のシフト減算を、桁上がり伝搬のない１の補数演算で近似する手法であり、高精度な近似不偏丸めとなることを示す。

第４章では、ベクタ演算を効率よく行うための、データメモリのアドレス生成技術について述べる。特に、音声符号化処理，２次元フーリエ変換，ビデオ符号化処理の３分野を取り上げ、それぞれのアルゴリズムを効率よく実行できるアドレス生成回路の構成法を明らかにする。また、アドレス生成回路を核としたＤＳＰ構成法を示す。

（4-A-1）音声ＤＳＰのアドレス生成技術

音声ＤＳＰでは、テーブル参照，フィルタリング，相関演算など様々な処理を平均して効率良く実行することが重要である。この要求に答えるために、多くの音声符号化アルゴリズムを取り上げて、アドレス生成の基本パタンを抽出し、それぞれを要素回路化し、それらの共通部分を核にして、ビルディングブロック状に構成する手法を提案する。

（4-A-2）音声ＤＳＰの構成法

前節でのアドレッシス生成技術を核として、高速音声ＤＳＰを設計する。そのため、アドレス生成以外の種々の技術について検討し、アーキテクチャを提案する。まず、種々の音声符号化アルゴリズムの分析から、アーキテクチャ設計の必要条件を抽出する手法を検討する。例えば、演算語長の策定には、有限語長シミュレーションでセグメンタルＳ／Ｎを求め、必要な精度を評価する手法を、データメモリ容量の策定には、仮想アーキテクチャ上で

のメモリマップを作成する手法を提案する。

次に、上記手法により定めたＤＳＰアーキテクチャの特長を示す。アドレス生成での特長に加えて、命令レベルでの特長について述べる。オーバヘッドのない繰り返し処理を可能とするリピート命令、マイクロフィールドを有効に活用するレジィデュアル制御命令、プリデコードによるオーバヘッドの少ないブランチ処理等による高速化の効果を示す。

さらに、インプリメント上の高速化技術について検討する。高速の浮動小数点ＡＬＵの構成法、演算のデータパス全体をマクロ化してレイアウトすることにより、最長遅延パスを短くするＬＳＩ-ＣＡＤの手法等について提案する。そして上記技術の有効性を確認するため、実際にＤＳＰを開発し、演算サイクルのスピードと種々のベンチマークの結果を示す。複素演算やＦＦＴ処理まで含めて、音声信号処理の広い領域に渡って高速処理できることを実証し、音声ＤＳＰの高速化技術の方向を明らかにする。

最後に本ＤＳＰを使った信号処理装置の設計法を示す。一例として、電磁波を用いた地下埋設物探知装置を取り上げる。この装置では、地中からの反射波をフーリエ変換し、周波数スペクトラム上に現われた特徴パラメータを高速に解析して、目標物を検知する信号処理を行う。装置の中枢である信号処理部に、本ＤＳＰを適用したときの、装置の構成と評価結果を示し、本ＤＳＰの有効性を明らかにする。

（4-B-1）２次元フーリエ変換ＬＳＩのアドレス生成技術

１マシンサイクル毎に１バタフライ演算を実行できる、強力なバタフライ演算器のデータメモリ構造とそのアドレス生成技術について検討する。その際、任意の２の冪乗の２次元フーリエ変換ができること、１次元ＦＦＴを処理単位として、インタリーブ処理が行えることを前提として、最適構成を考案した。データメモリのアーキテクチャとしては、４バンクの４ポートＲＡＭ構造を提案する。また、そのアドレス生成法としては、任意の２の冪乗ポイントのバタフライアドレスを生成する回路と、ビットリバースアドレスを生成する回路、それぞれの構成法を提案する。

（4-B-2）２次元フーリエ変換ＬＳＩの構成法

前節でのアドレス生成技術を核とした、２次元フーリエ変換ＬＳＩの構成技術について検討する。例えば演算回路では、複素バタフライを１マシンサイクルで処理するため、４個の乗算器と４個の３入力加減算器から構成した、６段パイプラインのバタフライ演算回路を提案する。またクロック系統の配置法等、高速性を損なわないＬＳＩインプリメンテーション上の手法について検討する。そして、このＬＳＩを実際にＢｉＣＭＯＳで試作し、性能を評価して、アドレス生成技術の有効性を示す。

（4-C）ビデオ符号化ＤＳＰのアドレス生成技術

ビデオ符号化処理では、２つのアドレス生成技術が重要である。一つは、任意の２次元のブロック領域を切り出してスキャンできるアドレス生成技術。もうひとつは、画像処理

のような、大量データ処理で必須となる並列処理に適合した、アドレス生成技術である。これらの特長を整理体系化する。

第５章では、高速化の中枢である並列処理技術について述べる。時間領域での並列処理であるパイプライン制御については、可変パイプライン技術と適応パイプライン技術を取り上げる。空間領域上の並列処理では、ＳＩＭＤ（Single Instruction Multi-Data Stream）制御を改良して、適応処理に適用できる手法を検討する。

（5-A）可変パイプライン制御技術

パイプラインデータパスをマイクロ命令でパイプライン制御する、ＤＳＰの基本手法について述べる。そしてこの基本技術を、音声ＤＳＰとビデオ符号化ＤＳＰに適用する。特にベクタ演算では、演算の種類によってデータパスが異なるので、これらのデータパスを組み合わせて一つの演算器を構成し、種々のパイプライン演算を淀みなく実行できるようにすることが重要である。これを、可変パイプライン制御法と名付けて、検討する。

（5-B）適応パイプラインを用いた専用ＬＳＩの構成法

電子ビーム直描装置の描画制御用に開発した、３品種からなるＬＳＩファミリを取り上げる。これらのＬＳＩは、適応パイプライン制御技術を駆使して、電子ビーム直描装置の描画速度を向上させている。この中で特に代表的な２種類の適応パイプライン制御技術について述べる。

いずれも、計算ステージと、計算結果に基づく描画ステージという、２つのステージをパイプライ処理する。一つは計算ステージと描画ステージの処理時間が共にデータ依存性を持っている場合であり、計算用と描画用の２つのカウンタを備えて、データの受渡しをすることで、効率良いパイプラインを構成できることを示す。もう一つは、データ依存性は描画ステージのみでがあるが、計算終了と同時に描画ステージに入ることが要求される処理である。すなわち、計算から描画に到るタイムラグをゼロとすることで、描画の正確度が最高となる処理である。これには、計算ステージに先行して、適応的に時間遅延を挿入することで実現できることを示す。

そして、これらの技術を実際に、バイポーラとＢｉＣＭＯＳとによりＬＳＩ化して、描画装置に実装し、予測どうりの性能が得られることを実証すると共に、適応パイプライン制御技術の効果を明らかにする。

（5-C）モディファイドＳＩＭＤ技術

複数のプロセッサエレメントが、それぞれの内部状態に因って異なる分岐をする場合、同一の分岐をするエレメント郡をひとまとまりにして並列処理することで、高速化を図る手法を提案し検討する。この例としては、複数の画素ブロックにそれぞれのプロセッサを割り振り、ブロックの状態に応じて、各プロセッサに異なる量子化処理をさせる場合等があ

り、符号化効率の高い適応アルゴリズム一般できる重要な技術である。また検討結果を、次章のビデオ符号化ＤＳＰに実際に適用し、その有効性を実証する。

第６章では、ＬＳＩ技術制約の解決法として、ＬＳＩ化にとって最も重要な回路規模の削減法について検討する。演算器については、冗長２進演算器と並列乗算器とを取り上げる。またオンチップメモリについては、プログラムメモリとデータメモリとを取り上げる。

（6-A-1）冗長２進演算器の回路規模削減技術

冗長２進加算器は、桁上がり伝搬が生じないため、語長に拠らない一定の高速演算ができる反面、回路規模が、通常２進系の３倍程度になるため、このままでは実用に供すのが難しい。冗長２進演算器を信号処理ＬＳＩに導入できるようにするポイントは、加算器の回路規模削減であるといって良い。

削減のために、１桁が３値で構成されている冗長２進数のコード化を検討する。加算器回路の論理の共通化を図れる適切なコード化を提案し、回路規模を２倍程度にまで削減できることを示す。

（6-A-2）冗長２進を活かした幾何学変換ＬＳＩの構成法

前節での成果を応用し、７０ＭＨｚのサンプルレートで動作する、高精細動画像（ＨＤＴＶ）用の専用ＤＳＰの構成法を検討する。特に、７０ＭＨｚレートの動画像の幾何学変換ができるようにするため、パイプライン演算サイクルを、冗長２進加算器１段分の遅延まで向上できるようにする構成を提案する。

幾何学変換では、アドレス計算が処理スループットを律則するので、任意のアドレス関数を高速に生成するアドレス計算ＤＳＰがキーデバイスとなる。そこで、任意の初等関数をシフトと加算の繰り返し演算で生成できるＣＯＲＤＩＣ（COodinate Rotation DIgital Computation）アルゴリズムに着目し、これをパイプラインアレイ（ＣＯＲＤＩＣアレイ）で実現する方法を提案する。

さらに、上記アーキテクチャを一層有効に機能させるためのインプリメント上の高速化技術を提案する。これは、マイクロコードをデータと並行に流すことにより、パイプラインレジスタ間のクロックスキューを低減させる制御法であり、コントロールストリームと名付ける。そして、実際に１００ＫＧ規模のＬＳＩに集積して、高速化技術の効果を実証する。

（6-B）並列乗算器の回路規模削減技術

ここでは、並列乗算器回路の大部分を占める部分積加算部での、回路規模削減手法を示す。２の補数乗算においては、部分積加算は、各部分積の符号ビットを最上位まで拡張して加算するのであるが、その替わりに、適当な複数個の桁に１を挿入する手法である。この手法が、任意桁の２の補数乗算に適用できることを示す。

（6-C-1）オンチップメモリの容量削減技術

ここでは、プログラムメモリとデータメモリそれぞれの容量削減技術を示す。まずプログラムメモリでは、マイクロフィールドが長くてスタティックステップの短いベクトル演算を、メインプログラムプログラム（メインメモリ上）から切り出して、ローカルメモリに移すことにより、メインメモリ容量を効果的に削減し、メインとローカルの総和としてのメモリ容量を減らせることを定量的に示す。そしてこの手法を、２階層マイクロプログラム制御技術として提案する。

またデータメモリでは、M個のプロセッサが共用する、Ｎワード×Mバンクのデータメモリを、ある特定のベクトル演算（動きベクトル検出のためのマッチング演算）の場合に、Ｍ・Ｎワード×Mポートメモリとして使用できる手法を提案する。そしてこの手法が、複数のプロセッサをオンチップ化して並列動作させる場合に、メモリ容量の大幅改善に結ぶつくことを示す。

（6-C-2）オンチップメモリ容量を削減したビデオＤＳＰの構成法

上記（6-C-1）と（6-C-2）の技術を応用して、動画像符号化を実時間で処理できる、非常に処理能力の高い、並列アーキテクチャのビデオＤＳＰの構成法を検討する。プログラム制御による汎用性の利点と、ビデオ処理に特化した高速性の利点を両立させる立場から検討する。

具体的には、0.8μm技術で３００ＭＯＰＳの性能を必要とする要求実現を取り上げ、ＬＳＩ技術制約を解決していく手法を論じる。特に高性能化の代表的手法である、高クロックサイクルの手法と、並列処理の手法について比較検討し、低消費電力の点で並列処理の方が有利であることを示す。そして具体的に、並列ビデオＤＳＰを設計し、その基本アーキテクチャを示す。また、試作による実証結果を示す。さらに、並列ＤＳＰでマルチプロセッサを構成して、ビデオＣＯＤＥＣをインプリメントし、そのときの、プログラムメモリの使用効率，データメモリの使用効率，負荷分散システムでのＤＳＰ稼働効率を論じ、本アーキテクチャの有効性を示す。

第７章では、本研究で得られた成果をまとめる。これらの成果は、信号処理ＬＳＩを部品の視点から高速化した成果の集大成である。一方これからのディープサブミクロン時代では、信号処理システム全体をＬＳＩ化することが可能になるため、ＬＳＩの高速化をシステムとしての視点から考えて行かなければならない。この点について、今後の展望と著者の見解を述べる。

1.4 参考文献

[1] 吉村、山内、”画像処理ＬＳＩの動向”、テレビジョン学会誌, vol.46, no.3, pp233-238, (Apr.,1992).

[2] 山内、”ＤＳＰの現状と将来展望”、電子情報通信学会技術研究報告, ICD91-98, pp.1-8, (Sep.,1991).

[3] M.Nagatani, H.Yoshimura, et al., "Digital Signal Processors for Decoding/Encoding Color TV Siganals", IEEE Journal of Solid-State Circuits, Vol. SC-21, No.6, (Dec.1986).

[4] 藤原、「小型カラー動画／テレビ電話を支えるＶＬＳＩチップセット」、pp.4-9, ９２年６月号、エレクトロニクス

[5] S.Nakagawa, H.Terane et al., "A 24-bit 50ns Digital Image Signal Processor," IEEE Journal of Solid-state Circuit," vol.25, no.6, pp.1484-1493, (Dec., 1990).

[6] K.Kikuchi, N.Yasuaki et al., "A Single-Chip 16-bit 25ns Realtime Video/Image Signal Processor," in Proc IEEE-ISSCC'89, pp.170-171, (Feb., 1989).

[7] T.Mimami, H.Yamauchi et al., "A 300-MOPS Video Signal Processor with a Parallel Architecture," in Proc. IEEE-ISSCC'91, pp.252-253, (Feb., 1991).

[8] J.Goto et al. "A 250MHz 16b 1.13M-Transistor 0.8μm BiCMOS Super-High-Speed Video Signal Processor," ISSCC Digest of Technical Paper, pp.254-255, (Feb.,1991).

[9] ITT . , " Vision Processor, " in Proc. IEEE-CICC'91, pp.252-253, (May, 1991).

[10] M.Toyokura et al., " A Video DSP with a Vector-Pipeline Architecture", ISSCC Digest of Technical Paper, pp.72-73, (Feb.,1992).

第２章　信号処理ＬＳＩ高速化の課題

2.1　高速化に向けてのアーキテクチャの発展[1]

前章では、信号処理ＬＳＩの発展を、要求性能とＬＳＩ技術との綱引きとして捉えた。ＬＳＩ技術が要求性能より低い時には、専用化することで高速化して性能を上げる。一方、ＬＳＩ技術にゆとりがある時には、汎用性と高機能性の方へとハードウエア資源を割り振る。そして専用化は、単一用途での需要が大きい場合には、経済的にも有効な手段であるが、市場が未成熟の場合には、小量・多品種の場合が多く、この場合には、汎用化して多用途への適用を図らなければならない。すなわち、信号処理ＬＳＩでは、ＬＳＩ技術制約のもとで、高速性と高機能性とを、バランス良く追及することになる。

ところが、高機能性は、汎用性ばかりではなく、広い意味での並列処理を実現する技術であり、重要な高速化技術でもある。例えば、データメモリのアドレス生成技術は、演算部とメモリ部とをパイプライン処理するための、アドレスとデータの並列処理技術であり、高速化技術の一つといえる。そこで、高機能性も信号処理ＬＳＩの高速化技術に含まれるとする。そして、信号処理ＬＳＩが、ＬＳＩ技術制約をいかに克服して高速性を実現してきたかを外観する。特に、高機能性では汎用ＤＳＰを、高速性ではビデオＤＳＰを取り上げる。

2.1.1　汎用ＤＳＰ

ＤＳＰの発展は大きく２つの段階に分けることができる。第一段階は、第１世代ＤＳＰの発明から第２世代ＤＳＰの出現までと、第１世代ＤＳＰを基本に開発された、モデムやオーディオ等の専用ＤＳＰの開発までである。第２段階はそれ以降現在までであり、第２世代ＤＳＰから第３世代ＤＳＰ（３２ｂ浮動小数点ＤＳＰ）および超並列エンジンへの発展と、ハイエンドな専用ＤＳＰの出現とからなる。この発展の概念を図２.１.１に示し、以下内容を概説する。

（１）第１段階での発展

第１世代ＤＳＰは、主に音声帯域でのディジタルフィルタのリアルタイム処理を目的に、１９７９年から８０年にかけて発表されている。この主要性能を表２.１.１に示す[2][3][4][5]。品種によって多少の違いは或るが、いずれも、ディジタル信号処理に合わせた高効率な特別のアーキテクチャが採用されている。良く知られているようにプロセッサの性能は、

（A）　処理に必要な命令ステップ数
（B）　命令に必要な実効クロック数
（C）　クロックのサイクル時間

の３つを小さくすることで向上する。第１世代ＤＳＰは、この３点それぞれに、理想的な解

答を提供した。まず（A）については、処理の対象をディジタル信号処理、特に乗加算のベクトル演算に絞って、この実行回路（乗加算器）をハードウエアで搭載した。これにより、１乗加算の実行に必要な命令を、１ステップにまで小さくした。（B）については、全命令を、原則として１マシンサイクル（１クロック）まで小さくできるように設計した。この実現のために、以下の２つのアーキテクチャが採用された。

・ハーバードアーキテクチャ
・マイクロプログラム制御のパイプラインアーキテクチャ

（C）では、すべての制御系の遅延を、演算器の最長演算時間、またはデータメモリのサイクルタイム以下となるようにし、演算器の性能いっぱいまでクロック速度を向上できるようにしている。以上の相乗効果により、第１世代ＤＳＰは、信号処理をおこなわせれば、当時の汎用マイクロプロセッサに比して、２桁程度高速であった。以上の高速化手法を採用したＤＳＰのモデル的なアーキテクチャを図２.１.２に示す。処理の基本は、「１マシンサイクルの間に、データメモリから演算器に常時２データを取り込み、乗加算を実行する」ことである。

その後ＬＳＩ技術の進歩に伴って、ＤＳＰは高速性と汎用性を高めていった。そして、１.５μｍ程度の技術による、いわゆる第２世代ＤＳＰが、８６年頃までに種々発表された。高速性の点では、上記（C）のクロックの高速化によるところが大きく、そしてその大

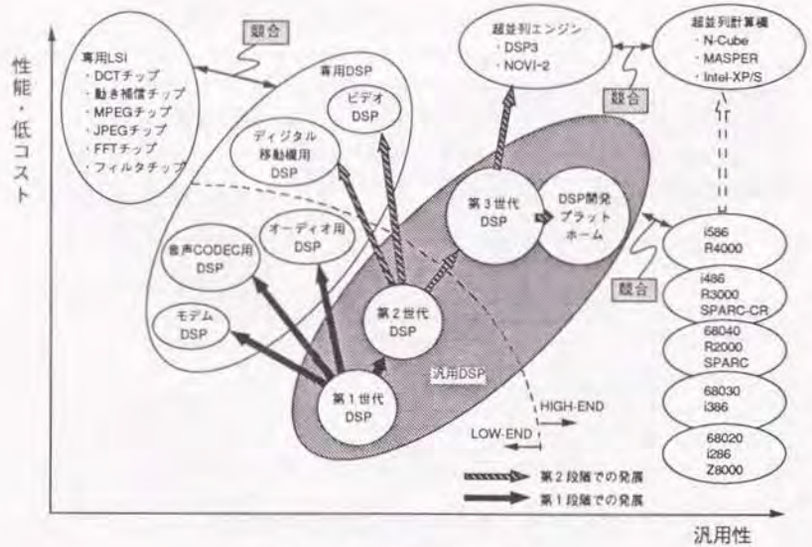


図２.１.１　汎用DSP発展の概念

表2.1.1　第1世代ＤＳＰ（初期ＤＳＰ）の概要

項　目	2920-10	S2811	D7720	DSP-20
演算サイクル	400 ns	300 ns	250 ns	800 ns
データ語長	25 b	16 b	16 b	20 b
乗算器	プログラム	並列 (12×12→16)	並列 (16×16→31)	並列 (20×16→36)
データRAM	40W	128 W	128 W	128 W
データROM	) 192W	128 W	512 W	) 1024 W
プログラムROM		256 W	512 W	
入出力	アナログ入力　4 アナログ出力　8	シリアルポート　1 パラレルポート　1	シリアルポート　1 パラレルポート　1	シリアルポート　1 パラレルポート　1
チップサイズ（mm^2）	30.38	26.07	28.44	68.50
消費電力（W）	0.8	1.0	0.9	1.5
パッケージ	28-DIP	28-DIP	28-DIP	40-DIP
特長	A-D,D-A内蔵		最も高速	最も高精度

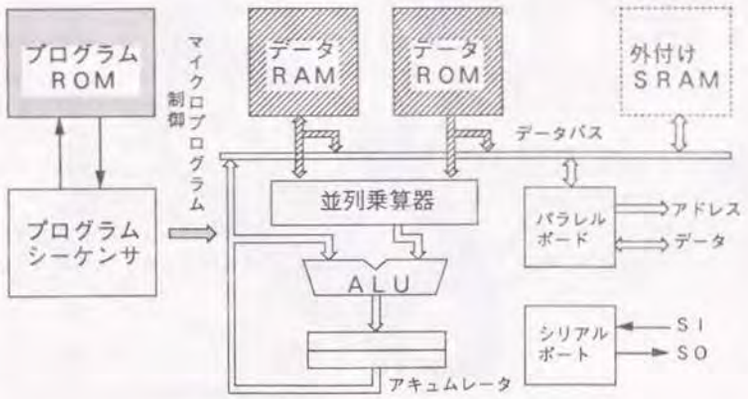


図2.1.2　第1世代ＤＳＰの基本構成

部分は、微細加工の進歩に拠っている。アーキテクチャの点からは、登載可能回路規模の増大を、高機能化に活かしている。高機能化に伴って、水平マイクロ命令語長が伸び、より多くの機能モジュールを並列に動作させることで、複雑な処理が、高速に実行できる様になった。これによりＤＳＰは、ディジタルフィルタ処理だけでなく、リアルタイム制御，音声分析・合成，高機能ＭＯＤＥＭ等の広い領域に渡って、項目（A）が改良され、高処理能力となった。そしてこれが、ＤＳＰ市場を広げることとなったのである。このとき採用された代表的な技術を以下に示す。また、第1世代から第2世代への改良を、表2.1.2に具体的に示す。

・演算語長の拡大（３２ｂ固定または、１８ｂ，２４ｂ等の変則的浮動小数点）
・チップ搭載のデータメモリとプログラムメモリの容量の拡大
・データメモリとプログラムメモリのアドレス空間の拡大
・周辺機能の強化（シリアルポート用ＤＭＡ，割り込み処理，マルチＤＳＰ処理等）

また第2世代ＤＳＰの、データパス部（アドレス生成部，データメモリ部，演算器部）をコアにして、必要に応じて専用回路を付与した各種専用ＤＳＰも開発された。代表例として、各種のモデム用ＤＳＰや３２ｋｂ／ｓの音声ＣＯＤＥＣ用ＤＳＰ等がある。この例の様に、最初は通信分野でのＡＳＳＰ（Application Specific Standard Product）化が進展したが、次第に、テレビの映像処理や高級オーディオ等の民生分野へも、導入されるようになった。

表2.1.2　第二世代ＤＳＰの代表的な改良点

項　目	内　容
演算語長	16ｂ固定 → 浮動小数点 12E6，16E6，16E8等
データメモリ（内部）	128 W → 512 W
データメモリ（外部）	－ → 4 K～128 KW
命令メモリ（内部）	512 W → 4 KW
命令メモリ（外部）	－ → 4 K～128 KW
演算サイクル	800～250ns → 100～50ns
処理能力	1～4 MOPS → 10～40 MFLOPS (MOPS)
素子数	40 K → 100～300 K
パッケージ	DIP → PGA
プロセス技術	3 μmNMOS → 1.2～2.0 μm NMOS/CMOS

（2） 第2段階での発展

第2段階では、第2世代ＤＳＰが基本となり、専用化と汎用化の両面で発展した。まず専用ＤＳＰについて少し触れると、第1段階で始まった専用ＤＳＰへの品種展開が、通信・民生分野を中心に一層活発化した。この方向は、丁度８７年頃から始まったＡＳＩＣ化の流れにも適合したため、大きな流れとなって、パソコン，家電，制御，画像処理等様々な分野で専用のものが開発された。またこれとは別に、８９年頃より、ディジタル移動機用ＤＳＰとビデオＤＳＰという、パーソナル通信とビジュアル通信の２つの分野を代表するＤＳＰの開発が、活発に進められた。ディジタル移動機では、情報量圧縮のための音声符号化と、通信路での誤り訂正用のための伝送路符号化とを行うＤＳＰが、携帯通信を実現するキーデバイスとなり、現在も活発に開発が続けられている。そしてここでは、低電力化と低コスト化が最大の課題である。またビデオＤＳＰは、動画の帯域圧縮を実時間で行う超高性能なＤＳＰであり、テレビ電話・テレビ会議等のマルチメディア通信と、動画蓄積を行うマルチメディアパソコンへのキーデバイスとされている。これについては、次節で詳述する。

一方汎用ＤＳＰでは、１μｍ前後のＬＳＩ技術を使って、第2世代ＤＳＰの汎用性をさらに高めた３２ｂ浮動小数点ＤＳＰ（以下第3世代ＤＳＰ）が、８８年前後に種々開発された。現在、第1世代、第2世代、そして第3世代ＤＳＰが共存し、第2世代以前のローエンドと、第3世代のハイエンドに分れている。注目すべきはハイエンドの第3世代ＤＳＰであり、現在、汎用化指向と高性能指向の２面への発展が進行中である。

（3） 第3世代ＤＳＰ

第2世代ＤＳＰでは、マシンサイクルタイム，メモリ容量，ＤＭＡ等、ハードウエアからの性能向上が特長として、ＤＳＰの適用領域を広げたことを述べた。これに対して、第3世代ＤＳＰと呼ばれる３２ｂ浮動小数点ＤＳＰの最大の特長は、データ形式として汎用コンピュータの標準単精度語長と同じ、３２ｂ浮動小数点演算語長を採用したことである。これにより、ＤＳＰは、一般の数値演算分野でも使用できるようになった。すなわちＤＳＰは、第3世代ＤＳＰになって、スタンドアロンリアルタイムシステムだけでなく、効率的な数値演算エンジンとしての用途も開けてきた。

現在の代表的な、第3世代ＤＳＰを表2.1.3に示す。浮動小数点演算能力は、２２ＭＦＬＯＰＳから５０ＭＦＬＯＰＳであり、これは現在の先端マイクロプロセッサの２～３倍程度上回っているにすぎない。並列処理を積極的に取り入れたビデオＤＳＰが、ギガＭＯＰＳ近くの性能を有するのと比較すると、その差は歴然としている。その第1の理由は、ビデオＤＳＰが、専用化によってハードウエア資源を高速化に活かしているのに対して、第3世代ＤＳＰは、メモリ空間や浮動小数点演算等の、汎用性に費やしているにからである。第2の理由は、高性能マイクロプロセッサが、スーパーパイプラインやスーパースケーラを採用し、ベクトル演算時には、ＤＳＰと同じように、演算器の最大性能を出せる構造になってきたためである。

このように、ＤＳＰとマイクロプロセッサとの演算スピードの差は、縮まってきているが、それでも、２～３倍の性能差を有している。この大きな理由の一つは、ＤＳＰが採用している３２ｂ浮動小数点データ形式が、高速演算に向いていることである。この形式は、仮数部，指数部共に２の補数の浮動小数点形式である（一部、モトローラ社の”ＤＳＰ９６Ｋ”は例外）。それに対して、マイクロプロセッサが採用している形式は、ＩＥＥＥ７５４標準規格と呼ばれており、仮数部がサインマグニチュード形式になっていることと、非数，オーバフロー，アンダフローの異常処理や非正規化数の処理が面倒なことから、演算回路の最長遅延パスが長くなり、高速演算に不利である。

しかし一方、ＤＳＰが一般の数値演算に使用されるためには、ＩＥＥＥ７５４規格でインタフェースを合わせる必要がある。このため、内部の演算形式として、２の補数形式を採用し、入出力でＩＥＥＥ７５４に変換することで、一般計算機とのデータ互換性を取っている。変換をソフトウェアで行い、回路規模を少なくしているもの（ＴＭＳ３２０-Ｃ３０）もあるが、これは例外であって、その後継版（ＴＭＳ-Ｃ４０）および、その他のＤＳＰでは、専用回路を搭載して１マシンサイクルで変換できるようになっている。

2.1.2に示す。これらはいずれも、大量の画像データ処理を行うために、汎用ＤＳＰと比較したとき、以下の２点で、それぞれアーキテクチャ上の工夫がなされている。

次に、第3世代ＤＳＰに共通するアーキテクチャ上の特長を述べる。その第1は、データメモリ空間が、１６ＭＷ以上（Ｍ９６ＫとＣ４０は２ＧＷ）となったことである。これは、グラフィックス，ビデオ処理，大規模数値演算，ニューラルネット等、従来のスーパコンピュータの領域への拡大を意識していると言える。

表2.1.3 主な第3世代ＤＳＰ

品名 （開発機関）	DSP32-C (ATT)	D77240 (NEC)	TMS320C30 (TI)	DSP96K (ﾓﾄﾛｰﾗ)
演算サイクル	80 ns	90 ns	60 ns	30 ns
浮動小数点演算 (FLOPS)	25 M	22 M	33 M	50 M
語長 IEEE754規格 入出力コンパチ データ形式 完全準拠	24E8 ○ - -	24E8 ○ - -	24E8 ソフトで対処 - -	24E8 ○ ○ ○
データメモリ内部 外部	512 W×2 16 MW	512 W×2 16 MW	1 kW×2 16 MW	1 kW×2 4 GW×2
命令メモリ 内部 外部	512 W 16 MW	2 kW 62 kW	4 kW(ROM) データと共用	1 kW 4 GW
パッケージ 素子数 消費電力 チップサイズ(mm2)	100-PGA 450 K 0.75W 88.2	132-PGA 370 K 1.5 W 9.55×9.72	180-PGA 700 K 1.0 W 12.7×12.7	123-PGA 750 K 1.0 W 15.0×15.0
開発環境 Ｃコンパイラ ＩＣＥ	 ○ ○	 開発中 ○	 ○ ○	 ○ ○

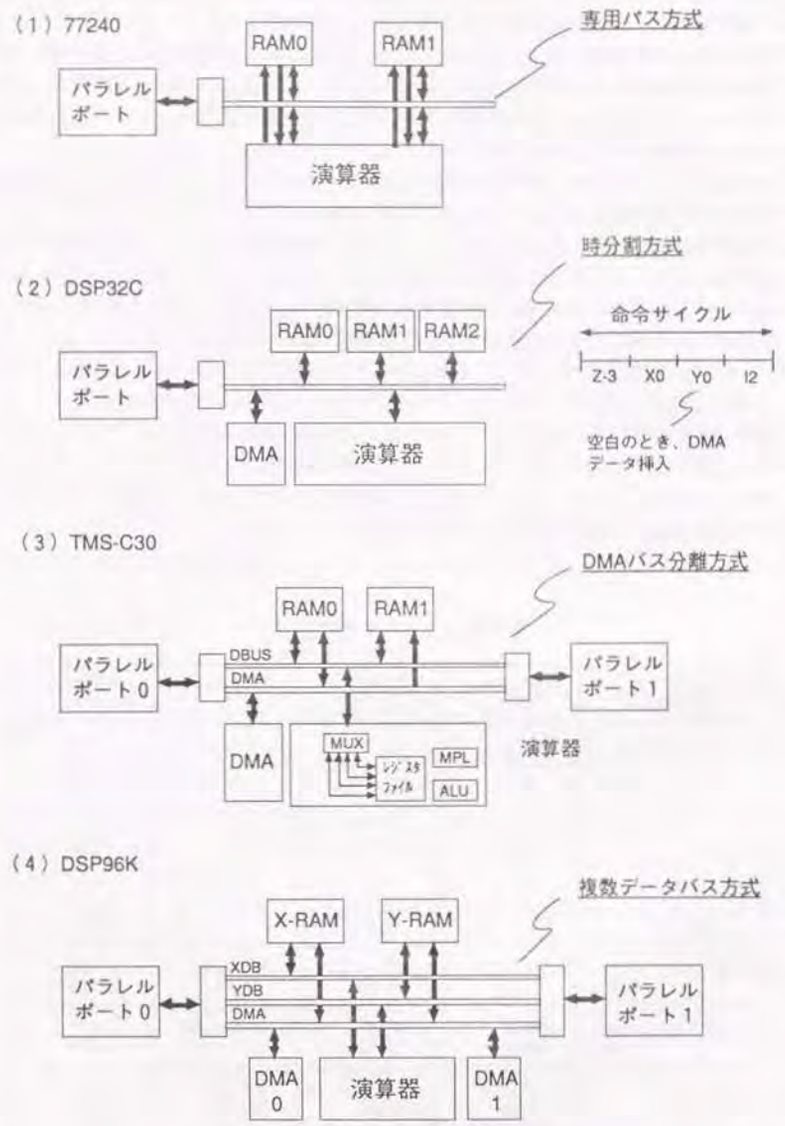


図２.１.３　第３世代ＤＳＰのデータバスの進展

第２は、バスが高性能化し、プログラムの転送，データの転送，ＤＭＡ転送の３処理が、常時並列実行されるようになったことである。このうちデータバスについて、各ＤＳＰの特長を図２.１.３に整理して示す。

７７２４０とＤＳＰ３２Ｃとは、開発が早かったため、一部、第２世代ＤＳＰのアーキテクチャを引きずっている。例えば、２面のデータメモリを使った２項ベクタ演算を取り上げてみる。この時、２面メモリから同時にデータを読みだして、その２データを演算回路へ転送しなければならない。７７２４０は、この転送を、１本のデータバスと、メモリと演算器とを直結する専用バスの２つを使って行っている。

一方ＤＳＰ３２Ｃでは、１本のバスにメモリと演算器をすべてぶら下げ、１演算サイクルを４相に分けて、Ｘデータ、Ｙデータ、Ｚデータの転送ステートを割り振る時分割方式を取っている。さらに、命令とＤＭＡを並列実行させる時には、命令種を早めにデコードしてバスの空くステートを先行検出し、空いたステートを使って転送させるという、高度な制御を行っている。しかしながらこの方式は、デバイス技術が進歩して、演算サイクルが短くなっていったときに、不利になると思われる。

アーキテクチャの新しいＴＭＳＣ３０では、ＤＭＡバスを専用に設け、ＤＭＡ転送効率を上げている。ただしデータバスは１本であるため、命令により２データを同時に演算器に転送することはできない。そこでこの弱点を補うため、演算器内に多ポートのレジスタファイルと複数バスとを設け、レジスタファイル容量の範囲内で、多変数ベクトル演算を連続実行できるようにしている。

ＤＳＰ９６Ｋは最も新しいアーキテクチャのため、大量データのベクタ演算に向いた構造になっている。データバスを、ＸＤＢとＹＤＢの２本としたので、２データの同時転送が可能である。さらにＤＭＡ転送専用のバスを設け、命令と並列に、外部パラレルポートからのＤＭＡ転送を可能としている。このように、バスが複数化・専用化する構造は、ブロック単位の大量データ処理が必要なビデオＤＳＰのアーキテクチャの発展方向と同じである。

以上をまとめると、第世代ＤＳＰアーキテクチャの特長は、以下のようになる。

・２の補数型３２ｂ浮動小数点演算
・データメモリ空間が１６MW以上へ
・データバスの複数化

２.１.２　ビデオＤＳＰ

９０年に近くなってサブミクロンＬＳＩ技術を向かえた頃より、動画像の蓄積・通信サービスを実用化できる環境が整ってきた。ＬＳＩ技術の発展に合わせて、動画像の符号化アルゴリズムの標準化も活発に進められ、９０年には、テレビ電話・テレビ会議用の、H.２６１標準化勧告が、そして９２年には、動画蓄積用のＭＰＥＧ１勧告がなされた。そして９３年には、ＮＴＳＣレベルの高画質動画を対象とした、Ｈ２６２とＭＰＥＧ２標準のＣＤ（Committee Draft）が作成されている。

この分野で、現在最もハイエンドなビデオＤＳＰは、H.２６１からＭＰＥＧ１をターゲットとしたところで開発が展開されている。この中で、代表的な５種のＤＳＰの特長を表

2.1.2に示す。これらはいずれも、大量の画像データ処理に向いたアーキテクチャになっている。アーキテクチャ改良の目的は、以下の2点である。

・画像符号化処理の効率的な実行
・GOPS（ギガオペレーション／秒）処理への対処

第1点は、画像符号化処理への専用化で実現している。データメモリに画像データを配置するための2次元アドレス生成機能や、距離計算を効率良く行う専用演算器の搭載等である。この特長は、1989年に発表された、前2種のDSPから備わっており、まさに、ビデオDSPの原形の機能と見なすことができる。

第2点については、前2種と、91年に発表された3種とでは、取り組み方が全く異なっている。前2種は、汎用DSPに近い構成を取っているため、チップ単体の処理能力はそれほど高くない。このため、ビデオCODECに必要なGOPSの処理能力は、多数のチップを使って、画面分割の負荷分散型マルチプロセッシングを前提としている。これに対して後者では、最初からH.261標準化アルゴリズムCODECを、できるだけ少ないチップで構成することを目的として開発されたため、それぞれ専用的なアーキテクチャを導入して、DSPの処理能力を上げている。IDSPでは、同一演算器を4個並列に並べ、演算単位で並列処理できる様にしている。S-VSPでは、超高速のパイプライン演算器を搭載して高速化している。このとき問題となるデータ入出力オーバヘッドは、チップ内に専用

表2.1.4　代表的なビデオDSPの特長

品名（機関）	DISP（三菱）	VISP（日電）	IDSP（NTT）	S-VSP（日電）	VP（IIT）
発表時期	89年	89年	91年	91年	91年
クロック	20MHz	40MHz	25MHz	250MHz	33MHz
最大処理能力[*1]	60MOPS	120MOPS	300MOPS	750MOPS	1419MOPS
共通アーキテクチャ	[1] 画像用2次元アドレッシング（ブロックアドレス）機能 [2] 距離計算のパイプライン処理：動き補償やベクトル量子化への対処 [3] その他：16〜24b演算精度，画素ブロック処理に必要十分な内臓メモリ,転送と処理の並列向きバス構造				
GOPS処理への対処	多数のチップによるマルチプロセッシング		・複数の演算器とメモリを活用して演算単位で並列処理 ・(複数メモリ)+(複数演算器)構成	・超高速パイプライン演算器の搭載 ・(専用メモリ)+(共通演算器)構成	・機能単位の並列パイプライン処理 ・(専用メモリ)+(専用演算器)構成
マルチプロセッサ構成[*2]	機能分割+画面分割	画面分割	機能分割	機能分割	機能分割

（＊1）著者の推定値　（＊2）代表的な構成

データメモリを適宜配置して、バッファリングすることで防止している。VP（VisionProcessor）では、機能単位に最適設計された専用演算器を配置し、全体を機能単位にパイプライン処理することで高速化している。以上3種の高速化の概念を図2.1.4に示す。以上の技術により後者のDSPは、いずれも、H.261のビデオCODECを、4ないし1チップで構成できる処理能力を有し、CODECの1ボード化を可能としている。

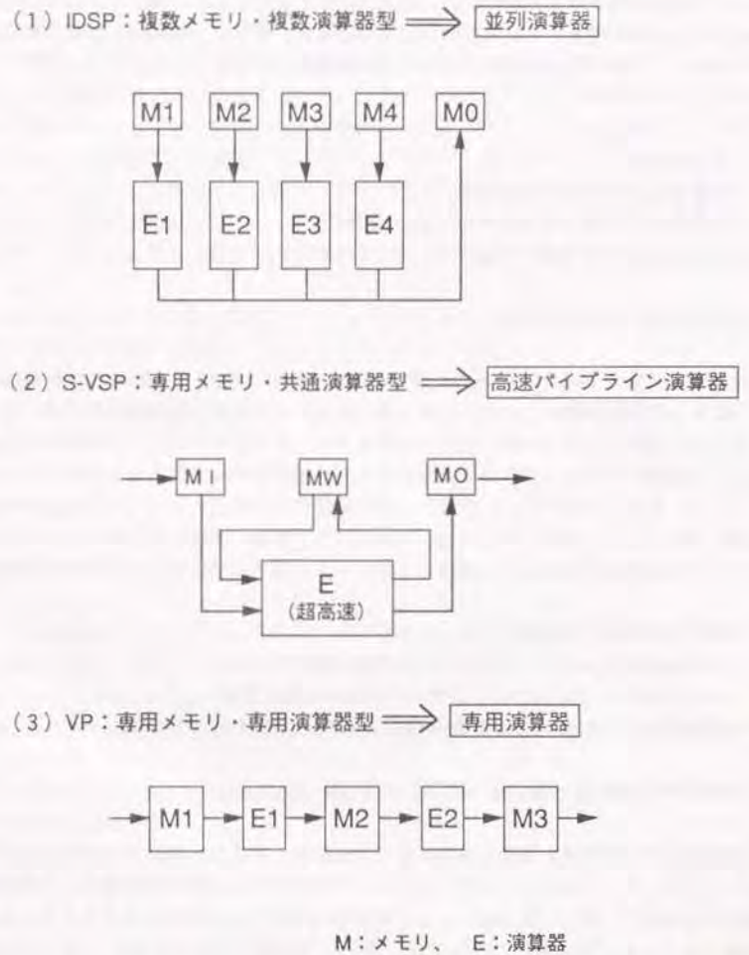


図2.1.4　各ビデオDSPの高速化手法

2.2　高速化技術の体系化

前節では、ＤＳＰアーキテクチャの発展は、高速化の歴史であることを述べた。この結果に基づいて、高速化技術を体系化すると、以下のように分類できる。

（A）演算サイクルを短くする技術

最長遅延パスを短くする技術であり、演算回路やデータメモリ等の高速化，クロックスキューの低減や単相クロック設計法等、論理設計技術からなる。この中で、以下の４演算器の高速化が、信号処理ＬＳＩにとって特に重要である。そして一般的に、ＬＳＩの集積度が上がるに従い、浮動小数点演算器や冗長２進演算器等、より多くのハードウェア要する演算器が使われてきている。

・並列乗算器
・語長制限した浮動小数点演算
・２の補数型３２ｂ単精度浮動小数点演算
・冗長２進による高速加算器

（B）時間方向の並列処理技術：パイプライン

ベクトル化率を高めて、いろんな演算をパイプライン処理できるようにする技術である。そのための技術課題は３点ある。第一は、命令を途切れなく取りだしでデコードする技術。第二は、処理アルゴリズムに従って必要なデータをデータメモリから演算器へ途切れなく転送する技術、第三は、演算器へ入力したデータを途切れなく加工する技術である。第一は、ハーバードアーキテクチャと呼ばれ、早い時期から当然のこととして取り入れられている。第二と第三については、それぞれ、アドレス生成技術，専用演算器のインプリ技術として、ＬＳＩの集積度向上と共に、発達してきている。以下に技術項目を具体的に列挙する。

・ハーバードアーキテキチャ
・信号処理用データメモリのアドレス生成技術
・ビデオ処理用２次元データメモリのアドレス生成技術
・信号処理・ビデオ処理用の複合パイプライン演算器インプリ技術

（C）空間方向の並列処理技術

第一に、データ転送と演算を並列に実行する技術、第二に、演算器を複数化して並列に実行する技術、そして第三に、マルチプロセッサ技術がある。第一の技術は、比較的早くから導入されてきた。第二と第三は、ＬＳＩの集積規模が１００ＫＧを越えるようになってから実現されている。特に第二の技術では、適応信号処理を並列に処理することが課題である。また信号処理におけるマルチプロセッサには、種々の単一機能のＤＳＰを複数個オンチップ化して、システム制御プロセッサの管理下で、ＤＳＰ単位の複雑なパイプライン制御を行うものと、ホストとなるＤＳＰのデータバスに、専用のコプロセッサを複数個接続して、並列処理するものとがある。以下に技術項目を列挙する。

・データバスの複数化
・ＤＭＡ機能の搭載
・ＳＩＭＤ型複数演算制御
・マルチプロセッサ制御

（D）データ転送のオーバヘッドをなくす技術

ＤＳＰの内部演算能力が高まるにつれて、ＤＳＰチップのデータ入出力スピードが、処理能力を制限するようになってきている。解決策の一つは、十分なメモリ容量をチップに搭載ることであり、回路技術で高密度メモリをつくる技術が重要である。もう一つは、同じデータの入出力をなくし、一度取り込んだデータでできるだけ多くの処理をする方法である。後者の方法は、演算器が複数個あってそれぞれがデータを共有して並列処理する場合に、メモリの多ポート化で、大幅な改善が期待できる。これらの技術は、以下の効果をもたらす。

・プログラムメモリ容量の拡大
・データメモリ容量の拡大

以上述べたの高速化技術（A）～（D）は、１章で述べた４つの研究課題を、それぞれ高速化の観点から整理したものである。これにより、各研究課題が、直接・間接に、すべて信号処理ＬＳＩの高速化に密接に関連していることを示した。そしてその改善への課題を明確にした。

2.3　参考文献

[1] 山内、”ＤＳＰの技術動向と応用事例”、１９９２年電子情報通信学会春季大会併催事業講習会予稿　「ディジタル信号処理プロセッサ」pp.2-21,（Mar.,1992).
[2] M.Townsend,"A Single Chip NMOS Signal Processor", IEEE ICASSP'80, p.390, 1980.
[3] G.P.Edwards,"A Speech/Speaker Recongnition and Response System", IEEE ICASSP'80, p.394, 1980.
[4] T.Nishitani et al.,"LSI Signal Processor Development for Communications Equipment", IEEE ICASSP'80, p.386,1980.
[5] J.S.Thompson et al.,"A Digital Signal Processor for Telecommunications", IEEE ICASSP'80, p.383, 1980.

第３章　高速演算回路の構成技術

３.１　並列乗算器の構成法

３.１.１　まえがき

乗算は、ディジタル信号処理で最も良く使われる演算であり、しかも固定小数点演算では、ほとんどの場合、演算回路の最長遅延パスとなっている。このため乗算器の高速化は、ディジタル信号処理の永遠のテーマであり、ＬＳＩ技術の黎明と共に、研究が開始されている。最初の乗算器ＬＳＩは、バイポーラ技術によるもので、１９７４年に２社から発表されている[1][2]が、信号処理を行う上では、以下の点で、まだまだ不満足なものであった。

その第１点は、乗算速度の不足である。ディジタル信号処理と共に急速に発展してきた音声符号化の分野では、ＡＴＣ（Adaptive Transform Coding）[3]、ＡＰＣ-ＡＢ[4]、さらに最近では、ＶＳＥＬＰ（Vector Sum Excited Linear Prediction）[5]等の処理量の多い高能率符号化が優勢になり、要求乗算回数が急激に増えている。また音声認識では、２乗距離計算の非常に多いＤＰ（Dynamic Pattern matching）演算[6]への適用が求められているし、さらに今日では、動画像符号化がへの適用が必須となってきている[7]。これらは、１６ビット精度で４０ＭＨｚ程度の乗算サイクルを必要としている。この実現には、ＬＳＩ技術の発展はもちろんであるが、乗算器アーキテクチャと回路技術の両面からの改善も必要である。

第２点は、語長の長い乗算器を構成できる拡張乗算機能がないため、応用領域が限定されることである。すなわち、乗算器ＬＳＩの演算語長以上の高精度の並列乗算器を必要としたとき、乗算器チップをアレ状に並べて拡張乗算器を構成するのであるが、上記のＬＳＩに拡張機能がないため、拡張した語長で「２の補数演算」を行うための付加回路と、各乗算器ＬＳＩからの部分乗算値を加算するための加算器アレイとが、外付けに必要になる。このため拡張乗算器の加算ステップを見ると、各ＬＳＩ内部での部分加算と外部加算器アレイでの加算とが、直列に実行される。このため、乗算器ＬＳＩの高速性を十分に活かした拡張乗算器を構成することができない。この改良には、乗算器ＬＳＩ内部に完全拡張機能（外付けの付加回路なしで最適な拡張乗算器アレイを構成できる機能）を搭載しなければならない。

第３点は、ディジタル信号処理の基本演算は、乗算単独ではなくて、乗加算であることに着目していない。信号処理用に改良するためには、累算入力を付与して、乗加算器としなければならない。

この節では、上記３点を改善して、ディジタル信号処理に広く使われる高速並列乗算器ＬＳＩを実現する手法について述べる。特に、乗算器の構成技術とインプリメント技術について述べる。構成技術では、デバイスに依存しない普遍的なアーキテクチャ技術についての成果を示す。インプリメント技術では、高速なデバイスとしてバイポーラ技術を取り上げる。そして、バイポーラの高速性をさらに伸ばす回路技術と、バイポーラデバイスの回路パラメータを高精度に抽出する技術について述べる。そして乗算器の試作例を示し、乗算器の乗算時間の高速測定法とその結果を示す。

３.１.２　乗算器構成技術

以下の条件を設ける。まず、データ形式は「２の補数」とする。基本機能は、８ビットデータＸ、Ｙの乗算と１５ビットデータＭの加算とを実行し、１５ビットデータＺ＝（ＸＹ＋Ｍ）を出力する。さらにこのＬＳＩをｍ×ｎ個アレイ状に並べることにより、任意の８ｍ×８ｎビットの乗算と（１６ｎ-１）ビットの加算を実行する、高速乗加算器を構成できる機能（完全拡張機能）を有することとする。

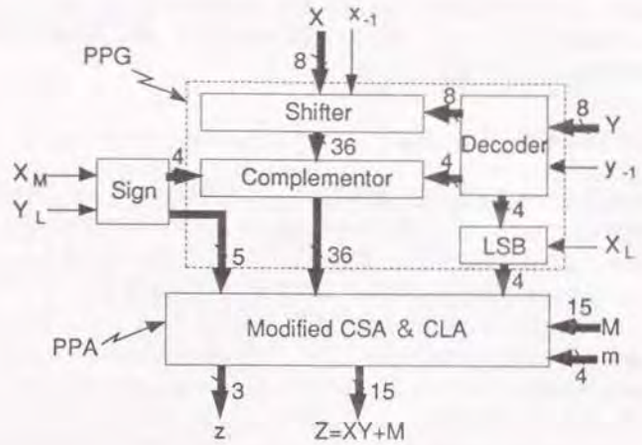
（１）　アーキテクチャ

一般に乗算器は、部分積生成器，加算器トリー，最終段のキャリー伝搬加算器の３ブロックから構成されている。そして各ブロックには、それぞれ、高速化手法が提案されている。それぞれの手法の長短を、高速性，回路規模，規則性の点から整理すると、表３.１.１のようになる。設計者は、求める乗算器が、速度優先か、サイズ優先か、またはその中間かによって、これらの中から適宜適切な手法を選択して構成する。ここでは入出力を２の補数とし、その範囲で最も高速な乗算器とするため、部分積生成器には、２次 Booth のアルゴリズムを[8][9]、加算器トリーには、新しく考案したＭＣＳＡ（Modified Carry Save Adder array）方式を、最終段加算器には、ＣＬＡ（Carry Look-head Adder）を採用した。

このＬＳＩのブロック図を図３.１.１に示す[10]。大きく分けて、部分積生成部（ＰＰＧ）と、部分積加算部（ＰＰＡ）とから成っている。部分積生成部は、主として、デコーダ，シフタ，コンプリメンタとから構成されている。デコーダでは、乗数Ｙを２次Boothのアルゴリズムに従って２ビット単位でデコードして、Ｘ選択，２Ｘ選択，および２の補数生成の３種の制御信号を４組づつ生成し、Ｘ選択信号と２Ｘ選択信号（８本）をシフタに、２の補数生成信号（４本）をコンプリメンタに入力する。そこで被乗数Ｘは、上記シフタとコ

表３.１.１　乗算器構成の種々の手法と特長

機　能	構　成	高速性	回路規模	規則性
部分積生成	ＡＮＤ-ＥＸＯＲのアレイ	△	◎	◎
	Ｂｏｏｔｈ（ビット列リコード）	◎	△	○
	テーブルルックアップ	△	×	◎
部分積加算	キャリーセーブ加算	△	○	◎
	飛び越しキャリーセーブ加算	○	○	○
	Wallace Tree	◎	○	×
	冗長２進	◎	○	○
最終加算	リップル加算	×	◎	◎
	キャリールックアヘッド加算	◎	△	×
	キャリーセレクト加算	◎	△	○



PPG: Partial Product Generator Block
PPA: Partial Product Addition Block

図３.１.１　乗算器ＬＳＩのブロック図

18 17 16	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1	digit
	1 1 1 1 1 1 1 1 1	from four Partial Products
	1 1 1 1 1 1 1 1 1	
	1 1 1 1 1 1 1 1 1	
	1 1 1 1	from LSB
1	1 1 1	from Sign
	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	from (M)
1 1	2	from (m)
1 2	2 3 3 4 4 7 6 5 6 4 5 3 4 2 3	total

図３.１.２　部分積加算部へ導入されるデータの内分け

ンプリメンタを通過することにより、２次 Booth のアルゴリズムに従った９ビット長の４組の部分積となり、それぞれ、後段の部分積加算部へ導入される。

部分積加算部へ導入されるデータの内分けを図３.１.２に示す。上記部分積からの３６本に加えて、ＬＳＢ（Least Significant Bit）から４本、Ｓｉｇｎから５本、加算入力Ｍから１５本、拡張加算入力ｍから４本の計２８本が、別途入力される。ＬＳＢ出力はデコーダでコンプリメンタの活性化を指示したときに、活性化すべき部分積の最下位ビットに１を加算する４組の信号であり、２ビットづつシフトして生成される部分積のそれぞれの最下位ビット（１，３，５，７桁）に入力される。

Ｓｉｇｎ出力は、最下位の部分積の符号ビット（９桁）と、各部分積の符号ビットの間（１０，１２，１４桁）、および拡張乗算時に２ビット上位に生成される部分積の符号ビットとの間（１６桁）に１を加算する５組の信号である。この信号は、４組の部分積を加算するときに、各部分積の符号ビットを拡張して桁揃えを行う替わりに、上記５箇所への１の挿入と各部分積の符号ビットの反転とで、全く等価な演算を行わせる手法（１加算法と命名）に基づいて作られた出力である。１加算法が、符号ビット拡張による従来手法と等価なことの証明は、乗算器の回路規模を削減する技術として、６.４節で述べる。

上記すべての入力を一括して加算する部分積加算部の構成を図３.１.３に示す。加算器アレイにおいては、高速性と回路の規則性の両立を図り、ＭＣＳＡ（Modified Carry Save

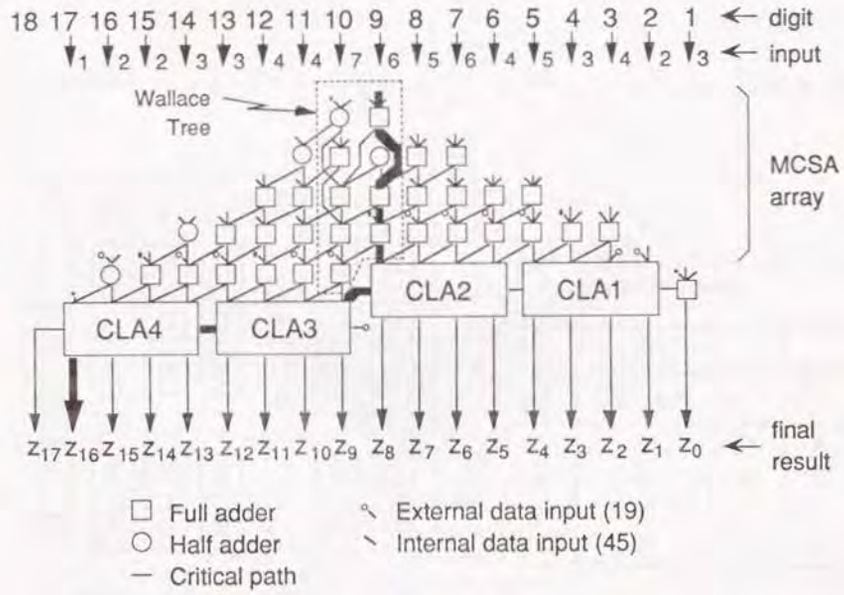


図３.１.３　部分積加算部の構成

Adder array）方式を考案した。これは、規則的なＣＳＡ（Carry Save Adder）アレイの最長遅延部に、部分的にWallace tree[12]を導入したもので、規則性を維持しながら高速化を図っているのが特長である。最終段の加算器には、４ビットのＣＬＡを４個直列に配置した。このときの加算部の最長遅延パスは、図の太線で示した経路となる。

（２）完全拡張機能[13]

（2-1）チップセレクト端子（X_L，X_M，Y_L）

拡張乗算器を構成するには、４種の乗算器ＬＳＩが必要である。それは、拡張した乗数および被乗数のどの部分を処理する乗算器ＬＳＩであるかによって、ＬＳＢ処理（コンプリメントのとき、最下位ビットに１を加える処理）、およびＳｉｇｎ処理（加算のとき、符号ビットを拡張して桁合わせを行う処理）を、活性化するかしないかを、選ばなければならないからである。この方法として、３本のチップセレクト端子、X_L，X_M，Y_Lを用いて上記４種（A，B，C，D）を識別する手法を考案した。３本のチップセレクト端子と４種のＬＳＩとの対応を表3.1.2に、４種のＬＳＩを用いた拡張乗算器の構成法を図3.1.4に示す。以下に４種のＬＳＩの特長と機能を示すことにより、本選択法を説明する。

（A）被乗数Ｘの最下位ビットを含むチップである。ＬＳＢ処理を活性化することにより、コンプリメンタが活性化された部分積の最下位ビットに、１を加算する。$X_L=1$と指定する。

（B）被乗数Ｘの最下位ビットと符号ビットを共に含まないチップである。ＬＳＢ処理、Ｓｉｇｎ処理共に不活性である。$X_L=0$，$X_M=0$と指定する。

表3.1.2　チップセレクト端子による４種のＬＳＩの選択

種類	チップセレクト		
	X_M	$X_M \cdot Y_M$	X_L
A	0	0	1
B	0	0	0
C	1	1	0
D	1	0	0

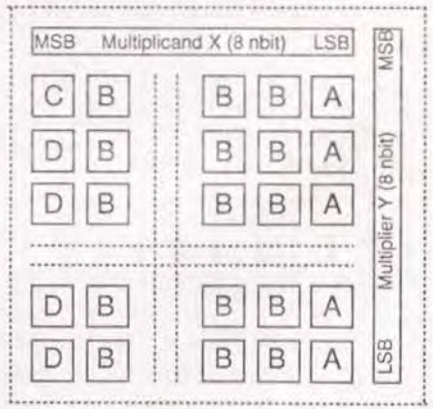


図3.1.4　４種のＬＳＩを用いて構成した拡張乗算器

（C）最下位に位置する部分積の符号ビットを含む（被乗数Ｘの符号ビットと乗数Ｙの最下位ビットを共に含む）チップである。当然Ｓｉｇｎ処理を活性化して、符号ビットの拡張を行う。ただし具体的には、前頁で述べた等価な演算手法（１加算法）に基づき、すべての部分積の符号ビットの反転と、９，１０，１２，１４，１６桁の５箇所への「１の加算」とを行う。$X_M=1$，$Y_L=1$と指定する。

（D）被乗数Ｘの符号ビットを含むが、最下位の部分積を含まないチップである。この場合は、最下位のＳｉｇｎビットを有しないので、Ｓｉｇｎの一部のみ活性化する。具体的には、すべての部分積の符号ビットの反転と、１０，１２，１４，１６桁の４箇所への「１の加算」とを行う。$X_M=1$，$Y_L=0$と指定する。

（2-2）拡張入出力（Ｍ（１５），ｍ（４），Ｚ（３））

８×８ビットの乗算器ＬＳＩを２５個用いて構成する４０×４０ビットの拡張乗算器を図3.1.5に示す。それぞれのＬＳＩは、内部で８×８ビットの乗算を実行して、拡張乗算

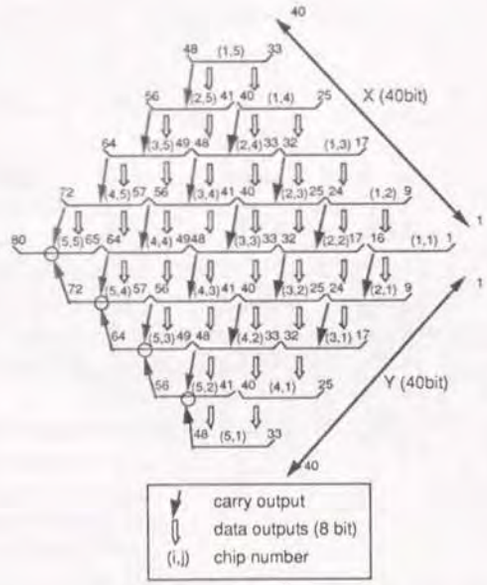


図3.1.5　拡張乗算器の構成法

器の部分積を生成し、隣接する前段ＬＳＩ出力との和をとり、結果を隣接する後段ＬＳＩへ出力する。このとき各ＬＳＩは、単体で使用するときの１～１５桁出力以外に、１６桁からの出力Ｚ15を生じる。

１～１５桁出力と１６桁出力については、図3.1.5から明らかなように、隣接する同位桁のＬＳＩで吸収するので、１～１５桁に亘る、データ入力M（１５）：M14～M0と、１６桁目のデータ入力：m15を設ければ良い。

また１７桁からのキャリー出力は、８ビット上位のＬＳＩの９桁目（１７桁-８桁）に設けた２本のデータ入力、ｍｓ，ｍｓ'で吸収する。一般に、９桁目に設ける入力端子数は、２本で十分なことが証明されている。この証明を3.1節の付録で示す。

しかしながら上記基本構成だけでは、１加算法に基づいて１６桁目に「１の加算」を行うときに不都合が生じる。それは、１６桁目への入力として、１加算法に基づく内部Ｓｉｇｎ回路からの１本が加わるため、基本構成での３本（隣接する同位桁ＬＳＩの１６桁出力からの１本と、図3.1.3から明らかなように、内部の１５桁目から１６桁目へのキャリー出力の２本）を加えて計４本となるため、１８桁からキャリー出力が生じるからである。

これを解決するため、本ＬＳＩでは、１８桁からのキャリー出力Ｚ17を拡張出力に加えた。そして最終的に、拡張出力を、１６，１７，１８桁の３本（Ｚ（３）：Ｚ17～Ｚ15）とした。さらにこれに合わせて、１７桁目にデータ入力m16を設け、１６，１７桁目のデータ

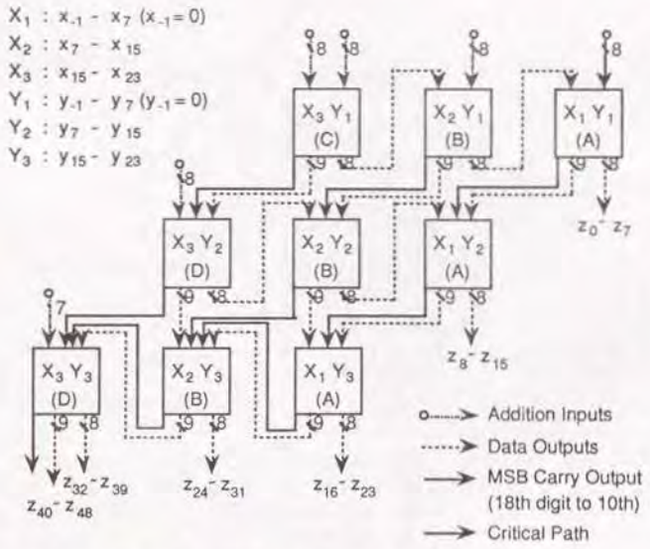


図3.1.6　本乗算器ＬＳＩを用いて構成した（２４×２４＋４７）ビットの乗加算器

入力m15，m16により同位桁ＬＳＩからのＺ15，Ｚ16出力を吸収し、さらにキャリー入力m8，m8'を１０桁目へ移し、８ビット下位に位置するＬＳＩからのＺ17出力を吸収する構成とした。すなわち、拡張入力端子を１０桁目に２本、１６，１７桁目に各１本の計４本（m（４）：m16，m15，m9，m9'）とした。

（2-3）拡張乗算器の乗算時間

本乗算器ＬＳＩを用いて構成した（２４×２４＋４７）ビットの乗加算器の構成を図3.1.6に示す。４７ビットの加算入力をオフにすれば、単なる２４×２４ビットの拡張乗算器である。この乗加算器および拡張乗算器の最長遅延パスは、ほぼ図3.1.3の太線で示した経路である。より正確に言えば、乗算器ＬＳＩの最長遅延パスが、被乗数入力からＺ16出力になっていることから、図3.1.6で示した太線より１ビット下位を通る経路であるが、近似的にはほとんど同じである。すなわち、最下位チップX1Y1のみ最長遅延パスを通り、残り４個の上位チップは、それぞれ下位のＬＳＩの１７桁目出力を９桁目入力で受けて、１７桁目から出力するまでの経路である。このことを、８ｎ×８ｎビットの場合に拡張すると、乗算時間Ｔは一般に次式で与えられる。

$$T = T0 + 2(n-1)T1$$

ここに、T0は乗算器ＬＳＩの乗算時間、T1は乗算器ＬＳＩの１０桁目のデータ入力m9（正確には、９桁目のデータデータ入力m8）から１７桁目のデータ出力Ｚ16までの遅延時間である。

3.1.3　ＬＳＩのインプリメント技術

（１）　高速回路技術

上記乗算器ＬＳＩを設計するに当たり、バイポーラ論理回路の中で最も高速な２つの回路である、ＮＴＬ回路とＬＣＭＬ回路の高速化手法を検討した。本節では、まず、それぞれの回路について提案した高速化手法を述べる。次に、回路シミュレーションまたはＴＥＧ試作により、技術の有効性を確認する。

（1-1）　ＮＴＬゲートの高速化[13]

従来、ＮＴＬゲートの高速性を左右するスピードアップコンデンサの最適設計値

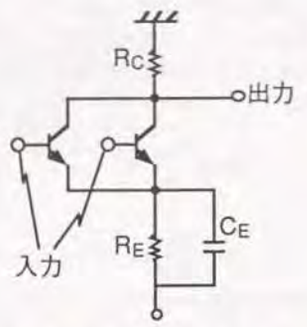


図3.1.7　直結型ＮＴＬゲート

は、回路シミュレーションに頼って決定している状況であり、設計効率が極めて悪かった。そこでデバイスパラメータから解析的に求める手法を提案し、設計の効率を向上させることとした。

直結型ＮＴＬゲートを図３.１.７に示す。この回路のスイッチングが高速なのは、２つの理由による。１つは、エミッタ負荷抵抗ＲＥを使って論理利得を下げ、明確なしきい値をなくすことで、入力電圧と共にゲートが連続的に応答する特性を持たせて、しきい値に達するまでの無駄時間を除去していること。２つは、論理振幅いっぱいを使ってオーバドライブし、急速にＯＮ状態に追い込んでいることである。この２番目の特長は、スピードアップコンデンサＣＥを付加することにより、一層大きくなる。その理由は、ＯＮ状態へ移行するときにコレクタ側から流れ込む電荷をＲＥのバイパスとなって、より急速に引き込めるからであるの[14]。

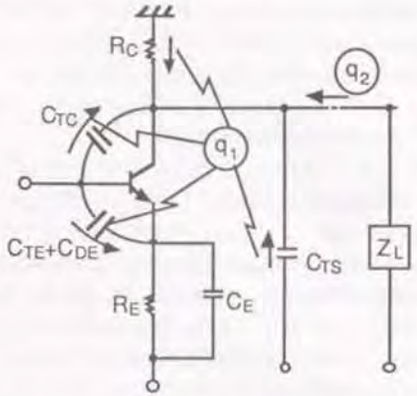


図３.１.８　ＯＮへ移行する時の過渡状態

しかしＣＥが大きすぎると一時的にコレクタ電流が過大となり、オーバシュートが現われるので、却って遅くなる。すなわち速度的に、最適なＣＥ値が存在する。ＯＦＦ状態へ移行するときにはＣＥの放電電流がエミッタを一時的にクランプし移行を速めるが、この時のＣＥ値はＯＮへ移行するときの値より小さくて十分である。

そこでＯＮへ移行する時について考える。Fan-in = 1 としたときの電荷の流れを図３.１.８に示す。電荷量として収支が調和するスピードアップコンデンサの値をＣＥとすると、ＣＥは、ゲートトランジスタの拡散容量ＣＤＥ，接合容量ＣＴＥ，ＣＴＣ，ＣＴＳを充放電する電化ｑ1と、後段の付加ＺＬ（後段トタンジスタのＣＴＥ，ＣＴＣ，ＣＴＳから成る）から流入する電荷ｑ2の合計量を、ＯＮの過渡時に吸収できる量である。すると、次式で表わされ。

$$q_1+q_2\approx\frac{R_E}{R_C}\cdot V_L\cdot C_{E0} \qquad (3.1.1)$$

$$q_1\approx\left(1-\frac{R_E}{R_C}\right)\cdot V_L\cdot\overline{C_{TE}}+\frac{3}{2}\tau_F\cdot I_C+\left(2V_L\cdot\overline{C_{TC}}+V_L\cdot\overline{C_S}\right)\cdot\left(\frac{V_L}{V_{EE}-R_E\cdot I_E}\right) \qquad (3.1.2)$$

$$q_2\approx\left[\left(1-\frac{R_E}{R_C}\right)\cdot V_L\cdot\overline{C_{TE}}+\tau_F\cdot I_C+2V_L\cdot\overline{C_{TC}}\right]\cdot\left(\frac{V_L}{V_{EE}-R_E\cdot I_E}\right) \qquad (3.1.3)$$

ここに、ＶＬは論理振幅、ＣＴＥ，ＣＴＣ，ＣＴＳは過渡時間の間のそれぞれの平均容量である。ＲＣ／ＲＥ＝1.55、ＶＬ＝0.45ｖ、ＶＥＥ＝1.1ｖ、エミッタとベースの接合容量の勾配係数を1/2とすると、式（３.１.１）～（３.１.３）よりＣＥは、

$$C_E=\left(1.52C_{TE0}+3.67C_{TC0}+0.86\overline{C_{TS}}\right)+6.86\tau_F\cdot I_C \qquad (3.1.4)$$

となる。ＣＴＥ０，ＣＴＣ０はゼロバイアスでの接合容量である。ここに第１項は接合容量に蓄積される電荷に基づくものであり、第２項はベース内蓄積電荷に基づくものである。遮断周波数ｆＴが高く、ＩＣが小さいほどＣＥは小さくなる。

式（３.１.４）と回路シミュレーションとの対応を図３.１.９に示す。回路シミュレーションでゲート遅延が最小となることを確認した最適容量値ＣＥＳと、計算から導出した最適値ＣＥを２倍した値（２ＣＥ）とは、広い電流域に渡って一致しており、本解析によるＣＥ値が、良い指標となっていることを裏付けている。ＣＥ値に掛かるファクター２は、オーバードライブの程度を示している。これより少ないときは、過渡電流が少ないため充電に時間がかかって遅くなるし、多すぎてもオーバシュートが現われ遅くなる。静的にバランスの取れる量の２倍が、スイッチング速度を最適にする値になっている。以上により、ＮＴＬゲートの最適設計を、計算式で簡単に行えるようになり、設計効率が向上した。

（1-2）　ＬＣＭＬゲートの高速化

ＬＣＭＬゲートは、電流切り替え型回路の論理振幅を小さくして、充放電荷を減らすことにより高速化したスイッチング回路であり、ＮＴＬについで高速である。ＬＳＩ化することで、100ps 程度のスイッチング時間が得られが、スピードが向上するにつれ、様々な要因による遅延が無視できなくなる。その中でも特に、スイッチング時に基準電位 Vref が変動することにより生じる遅延が大きい。以下、この解決法について述べる。

（1-2-1）　安定化容量の効果

２段縦積みのＬＣＭＬゲートを図３.１.１０に示す。この回路のスイッチング時の上段基準電位 Vref1 または下段基準電位 Vref2 （以下まとめて Vref とする）と、基準電位側のトランジスタのベース電流の変動の一例を図３.１.１１に示す。スイ

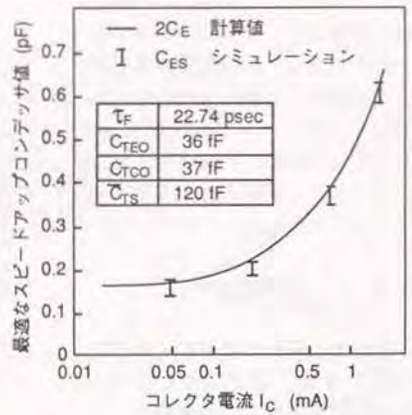


図３.１.９　本解析と回路シミュレーションとの対応

ッチング時にベースに流れる大きな過渡電流のために、基準電位が入力電圧と同位相に変動し、スイッチング時間を遅らせている[15]。

このベース過渡電流の生じる原因は、基準トランジスタのベース・エミッタ接合容量ＣＴＥ、ベース・コレクタ接合容量ＣＴＣ、そして拡散容量への充放電電荷が、基準電圧生成回路に流入するためである。これを解決するため、基準電圧生成回路に、この流入電流をバイパスする安定化容量（Ｃ１，Ｃ２）を設けることとした。以下Ｃ１，Ｃ２をまとめて、ＣＳとする。ＣＳの大きさは以下のようにして決定する。

基準トランジスタに繋がるゲート１個あたりの変動電荷量ｑは次式で近似される。

$$q=\int_{v_{e1}}^{v_{e2}} \frac{C_{TE}}{\sqrt{1-v/\phi}}dv + \tau_F \cdot i + \int_{v_{c1}}^{v_{c2}} \frac{C_{TC}}{\sqrt{1-v/\phi}}dv \qquad (3.1.5)$$

ここに、Ve1（Vc1）：スイッチング前のベース・エミッタ（コレクタ）間電圧、
Ve2（Vc2）：スイッチング後のベース・エミッタ（コレクタ）間電圧。

この電荷を安定化容量ＣＳで吸収するのであるから、ＣＳと基準電位 Vref の過渡変動量Δ Vref の間には次の関係がある。

$$q=C_S \cdot \Delta V_{ref} \qquad (3.1.6)$$

上式に基づき、高速のバイポーラプロセスであるＳＳＴ-２（Super Self-aligned process Technology - 2）を取り上げ、そのデバイスパラメータを使って[16]、安定化容量値を計算した。その結果を図３.１.１２に示す。ΔVref のＣＳ依存性が、計算値とシミュレーション値とで良く一致し、計算仮定の妥当性を良く裏付けている。またＣＳ＝０.５PFとすれば、ゲート電流０.２mAのとき Vref 変動は３０mV以下となり、スイッチング時の遅延を抑えられることがわかる。

（１-２-２）実験結果[17]

前節での解析に基づき、ＳＳＴ-２プロセスで、５１段のＬＣＭＬゲートのリング発振器を試作し、１ゲート当たり０.５PFの安定化容量

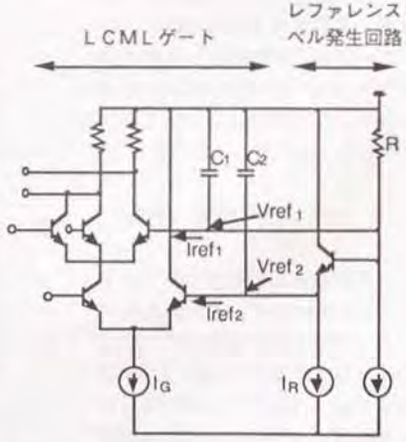


図３.１.１０　２段縦積みＬＣＭＬゲート

を、挿入した場合としない場合とについて性能を比較した。安定化容量は、高周波特性が良くて、かつ占有面積が小さいものとするため、１層目と２層目のアルミ電極間に厚さ1000Å の薄いプロズマ窒化膜を鋏む形で形成した。この場合、占有面積は９００μｍ２であった。

この実験結果を、基準電圧発生回路の電流ＩＲと、ゲート電流（エミッタ電流）ＩＧとの比Ｋ（＝ＩＲ／ＩＧ）に対する伝搬遅延時間ｔｐｄの依存性として、図３.１.１３に示す。パラメータは、ゲート電流である。Ｋおよびゲート電流の広い領域に渡って、ほぼ１５～２０％の高速化が達成されていることがわかる。例えばＩＧ＝３７０μＡのとき、ｔｐｄは２８０PSから２３０PS へ、５０PS 改善されている。

以上により、ＬＳＩ技術の進歩と共に顕著になる、基準電圧変動による速度劣化の問題を解決した。

（２）高精度デバイスパラメータ評価技術[18]

本節では、デバイスパラメータを、高精度抽出する方法について述べる。デバイスパラメータの精度は、回路設計の精度を決定するため、ＬＳＩが高性能化するに従い、より高精度に求めることが、要求されてきている。特に計算機処理により自動化して、再現性よく高精度に求めることが重要である。ここでは、エバース・モルモデルに基づき、ＴＥＧ（Test Element of Group）チップ上のバイポーラ素子から特性を自動的に測定し、さらに最小２乗法により、デバイスパラメータを自動的に抽出する手法を提案する。そして本手法により、高速バイ

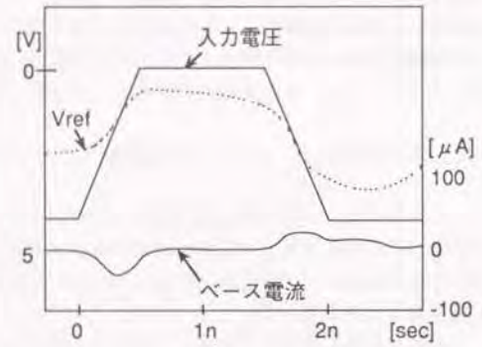


図３.１.１１　スイッチング時のVref および過渡電流の変動

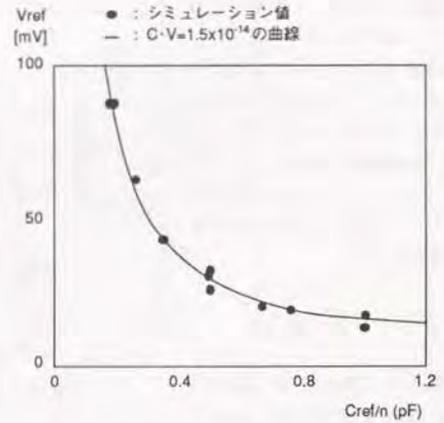


図３.１.１２　安定化容量による基準電圧変動抑制効果

ポーラ素子ＳＳＴ（Super Self-aligned Transistor）を評価して、その結果に基づいて、リング発信器モデルでシミュレーション値と実験値とを比較し、測定の精度の高さを実証する。

（2-1）パラメータ種別

バイポーラトランジスタのパラメータは、その測定手段により、（A）飽和電流やシリーズ抵抗のように、直流測定で求められるもの、（B）接合容量のように、低・中周波の交流測定で求められるもの、（C）拡散容量のように、超高周波または超高速パルス測定を必要とするものの、3種に分類される。この内、（A）と（B）のパラメータを、自動的に測定して、計算するシステムを開発した。（C）については、寄生容量等を除去した高精度の超高周波測定の条件を明確にした。以下（A）～（C）の各パラメータについて、それぞれ（2-2）～（2-4）節で述べる。

（2-2）直流測定とパラメータの分離法

トランジスタの等価回路モデルを図３.１.１４に示す。Ebers-Moll モデルを基本にして、より正確なモデルとするための２つの改良を行った。第１は、外部ベース抵抗rB と内部ベース抵抗rbb' を分離した。第２は、コレクタ拡がり抵抗rC1 とその他のコレクタシリーズ抵抗 rC2 とを分離した。

分離した抵抗をそれぞれ分けて求めるために、rB と rC2 とは電流依存性がないものとした。そしてあらかじめ別の手段で求めておく。次に後述する測定法により、（ rbb'(IC)+ rB）と（rC1(IC)+ rC2）とをそれぞれ求め、あらかじめ求めておいた、rB および rC2 との差分を取るって、rbb' と rC1を求めた。

測定プログラムの主要部分のフローチャートを図３.１.１５に示す。プログラムは大きく分けて、自動測定の部分と、デバイスパラメータを求める部分とからなっている。以下それぞれの概略を示す。

（2-2-1）自動測定

以下の５種の測定（mes1～mes5）からなっている。mes1とmes2 は、エミッタとコレクタの

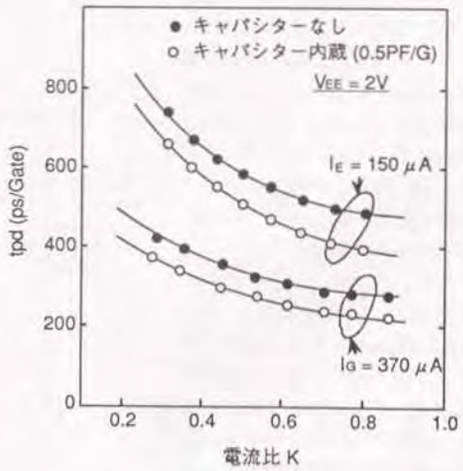


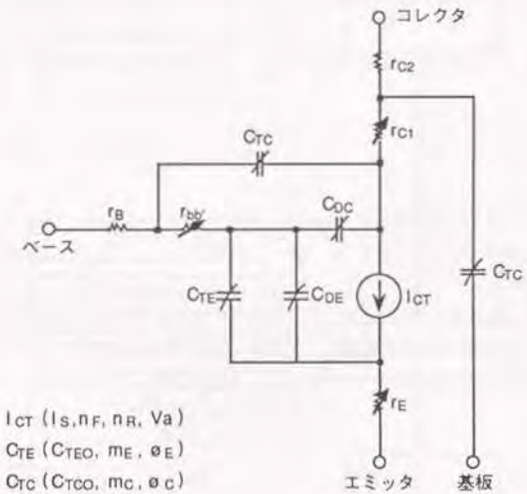
図３.１.１３　安定化容量の効果を示す実験結果

シリーズ抵抗を求める測定で、ベースから電流を流し込み、その電流の関数としてコレクタ・エミッタ間電圧 VCE を測定する手法[19]を用いている。mes3 と mes4 は、順方向および逆方向の VBE-Ic, VBE-IB 特性を求める測定で、この結果から、順方向電流利得 hFE(Ic) と逆方向電流利得 hFER(Ic)および内部ベース抵抗 rbb' (Ic) を求めるので、最も重要な測定である。mes5 は、IB をパラメータにしてVCE-Ic 特性を求める測定で、アーリー電圧の計算に用いる。

上記５種の自動測定のために、GPIB制御のマトリクススイッチを試作した。スイッチの特性は、接点抵抗260mΩ、絶縁抵抗50GΩ、スイッチングタイム2msecであり、100pA から100mA までの測定が可能である。各端子は２本のペア線で構成され、各接点が対になって同時にON-OFFするので、測定系の寄生抵抗は完全に除去できる。x 側の入出力は、電圧源，電流源，グランド各１端子と電圧計，電流計各２端子の計７端子があり、y 側の方はは、トランジスタの３端子とショート用２端子の計５端子がある。

（2-2-2）パラメータを求める処理

処理は OP1～OP5 の５段階からからなっている。OP1 と OP2 では、mes1, mes2 で求めた IB-VCE 特性から rE(Ic), rc1(Ic)+rc2 を求める。このとき、以下の３点に着目して、最も妥当と思われる手段をとった。



ICT (IS, nF, nR, Va)
CTE (CTEO, mE, øE)
CTC (CTCO, mC, øC)
CTS (CTSO, mS, øS)
CDE, COC ：別途専用の測定で求める

図３.１.１４　トランジスタの等価回路モデル

（A）低電流領域での α Rの大幅な変化による影響。
（B）測定電流 IBと、実動作状態でのコレクタ電流 Icとの関係。
（C）コレクタ拡がり抵抗 rc1(Ic) は、測定時と実動作時とで異なる（動作ベース領域が異なるため）。

（A）については、中電流領域でのデータから低電流領域での値を推定する方法で、影響を除去した。具体的には、I-r特性の実測定値をＣＲＴ上に描画し、測定者が適当な中電流領域を指定する。次にこの領域での値から、２次関数近似で低電流領域を推定し、推定値と中電流領域での実測値とが滑らかに繋がることにより、α Rの影響が除去されていると判断する。なるべく低電流領域に近く、かつ上記グラフが滑らかに繋がるように領域指定を行うことにより、良い結果が得られる。

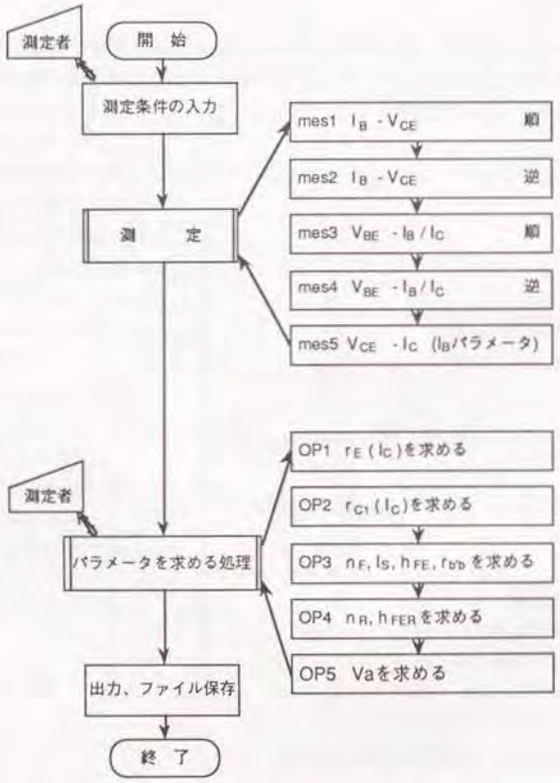


図3.1.15　直流測定とパラメータ分離のフローチャート

（B）については、mes3, mes4 で測定したhFE とhFER を用いて次式で求める。

$$I_B = I_E \ , \ I_C = \frac{I_E}{h_{FE}+1} \qquad (3.1.7)$$

（C）については、エミッタ直下の面積と全ベース領域の面積比 ζ と、あらかじめ求めておいた拡がり抵抗以外の抵抗値 rC2 を入力して補正する。エミッタ幅に比して、コレクタの低濃度領域の厚さが薄い場合には、

$$r_{C1(Ic)} = \frac{r_{Cm(Ic)} - r_{C2}}{\zeta - 1} + r_{C2} \qquad (3.1.8)$$

で近似する。ここに、rcm(Ic) は、mes2 での実測値である。低濃度領域が厚い場合には、厚さ d を考慮した補正式を用いる。

OP3 では、「vBE - log(Ic) 特性」の低電流領域の直線領域から nF と Is を計算する。同時に、計算で求めた理想直線と実測値との差（Δ vBE）からベース抵抗を求める。このベース抵抗を求める方法は、自動測定に向いている反面、２つの大きな数値の差（Δ vBE-rEIE）を利用するので、正確にデータ処理を行わなければならない。そこで以下の３ステップから成る手法により、低電流寄生領域での寄生抵抗の影響まで除去した「理想 vBE - log(Ic) 特性」を求めるようにした。

（A）選択した低電流領域でのデータから、最小２乗法で nF とIS の初期値（nF0, IS0）を求める。このとき、寄生抵抗は考慮しない。
（B）選択した低電流領域では、寄生抵抗に電流依存性がないと仮定し、コレクタ電流に換算した一定の抵抗値を次式から求める。

$$\delta = r_E \cdot \left(1+\frac{1}{h_{FE}}\right) + \frac{r_B + r'_{bb}}{h_{FE}} \qquad (3.1.9)$$

ここに rE は mes1 と OP1 とから、hFE は mes3 で得た測定値から自動的に取り入れる。rB と rbb' は、トランジスタのパターンレイアウト図から計算する。

（C）上で求めた換算抵抗値を δ の初期値 $\delta 0$ とし、さらにnF とISの初期値、nF0, IS0を加えて（nF0, IS0, $\delta 0$）を初期値とし、選択した低電流領域の特性を最も良く説明するパラメータ（nF, IS, δ）を計算する。この時最小２乗誤差を評価関数とした。すなわち、

$$\sum_{i=d_1}^{d_2} \left(\frac{V_{BE}}{V_{BEi}} - 1\right)^2 \quad \rightarrow \quad \min \qquad (3.1.10)$$

$$V_{BE} = \frac{kT}{q}\left\{n_F \cdot (\log I_C - \log I_S) + \delta I_C\right\}$$

収束計算には、共役勾配法である、Davison-Fletcher-Powellのアルゴリズム[20][21]と、Davidon探索法[21]を用いた。平均して4～6回のループで収束した。このようにして求めた正確なΔVBEと、OP1の処理で求めたREIEとからベース抵抗が求まる。

OP4では、OP3と同じ手法でnRを求めた。このときIsは、順方向特性の値を用いた。

OP5では、アーリー電圧Vaを計算する。このパラメータは、ベースの形成状態を知る上で、重要である。

（2-3）交流測定とパラメータ分離法

全体の処理フローを図3.1.16に示す。測定時には寄生容量を除去するため、実デバイスを搭載したパッケージでオープンキャリブレーションを行う。ただし完全なオープン状態を得ることは困難なため、理想状態との差による僅かな寄生容量が生じる。通常、0.1～0.3PF以下である。

次に、測定した接合容量を

$$C = C_S + \frac{C_{T0}}{(1+V/\Phi)^m} \qquad (3.1.11)$$

で表現し、このモデルから、寄生容量Csと、求める容量の3個のパラメータ（CT0, Φ, m）を分離する。このとき2つの問題点がある。

第1点は、測定値が式（3.1.11）からはずれる場合である。この例を図3.1.17に示す。（A）はベース・コレクタ接合容量に現われるパターンで、埋め込み層により空乏層

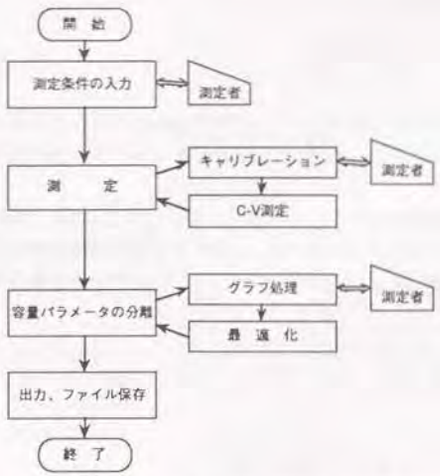


図3.1.16 交流測定のフローチャート

の伸びが押さえられているる場合である。（B）は主にベース・エミッタ接合容量に現われるパターンである。高バイアス時に、ブレークダウンが起きる直前でのコンダクタンス成分の急増による部分と、低バイアス時に、測定の交流信号の影響が現われてくる部分の、両方の影響が生じている。これらはいずれも、「（V+Φ）-（C-Cs）」特性曲線上で明確に識別できるので、図に示すように、有用な直線部分を指定することにより、解決できる。

第2点は、10fF程度と非常に微少な測定容量（CTE0, CTC0）を、数十倍の寄生容量を含んだ測定値から、如何に分離するかである。これに対しては、以下の2段階の手法を用いた。まずパラメータ（例えばCs）を振って「（V+Φ）-（C-Cs）」特性を描画し、その中から最も適切な曲線を選択する。次に上記選択値を初期値として、前節の直流パラメータ分離で用いたDavison-Fletcher-Powellのアルゴリズムと、Davidon探索法による最適化処理を行う。この時の評価関数も次式の最小2乗誤差を用いた。

$$\sum_{i=d_1}^{d_2} \left(\frac{C}{C_i} - 1\right)^2 \rightarrow \min \qquad (3.1.12)$$

上記処理の例として、エミッタ面積1.5×2.5μm2のSST-2トランジスタのベース・コレクタ接合容量での、初期値と収束結果を表3.1.3に示す。この時、収束後のモデルと実測値との誤差は、全バイアス領域に渡って上記トランジスタのベース・コレクタ接合容量以下（0.1fF以下）になり、極めて安定して収束することが確認できた。この理由は、マクロ的な視野で、物理的に妥当な

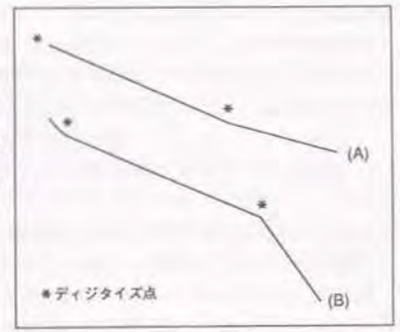


図3.1.17 ディジタイズによるデータの選択

表3.1.3 最適化ルーチンによる容量の収束例

	初期値	収束値
Cs	180 fF	181.3 fF
Co	16 fF	14.65 fF
ø	0.6 V	0.6238 V
m	0.4	0.3753

表3.1.4 SST-2トランジスタの容量パラメータ

CTEO	6.8 fF
øE	0.85 fF
mE	0.49
CTCO	14.2 fF
øC	0.65 fF
mC	0.38
CTSO	60.0 fF
øS	0.50 fF
mS	0.33

初期値を選択しているためと思われる。

上記手法で求めた、エミッタ面積 1.5×2.5μm2 のＳＳＴ-2トランジスタの容量パラメータを表3.1.4に示す。この値は、１０個のトランジスタについて測定し、その平均を取ったものである。

（2-4）高周波パラメータの測定法

超高速ＩＣ用の微少トランジスタの高周波特性を精度良く評価することは、容易でない。その理由の第１は、測定系とトランジスタとのインピーダン不整合により誤差が生じること。第２は、パッケージやパッド等の寄生素子の値の方が測定トランジスタの値より大きいことである。ここでは、上記問題点を解決する方法として、微少トランジスタを多数並列接続して、そのＳパラメータを測定し、その結果から正確に評価する手法を提案する。また実験結果を述べる。

トランジスタは、エミッタ面積1.5×5.5μm2のＳＳＴ-２である。このトランジスタの並列個数を、１，４，１２，３６個と変化させて測定し、その特性の変化を調べた。また寄生容量の効果を見るために、それぞれのワイヤボンディング用パッドの面積を、1000, 4000, 16000 μm2の３水準振らせて測定した。パッドは、厚さ1μm以上のロコス（SiO2）膜上に形成し、できるだけ容量を削減した。またＳパラメータの測定にあたっては、パッケージを使って、オープン，ショート，50Ω整合のキャリブレーションを行い、パッケージ容量の影響をできるだけ少なくした。測定範囲は、200MHzから3GHzとし、hパラメータに変換して、fT と入力インピーダンス Re(h11)を計算した。

トランジスタの並列個数と fT の関係を図３.１.１８に示す。コレクタ電流 Ic は、トランジスタ１個あたり0.5mA とした。トランジスタ個数の増加に伴い fTの実測値が高くなり、トランジスタ個数が１２個以上になって、6.4GHz (VCE=1v) の一定値になっている。

次にパッド面積と fT との関係を

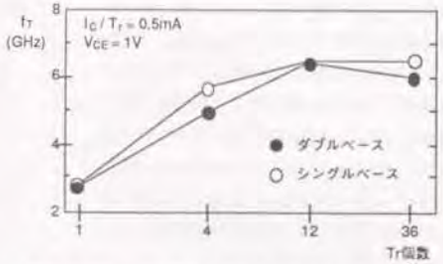


図３.１.１８　ｆＴ測定値のトランジスタ個数依存性

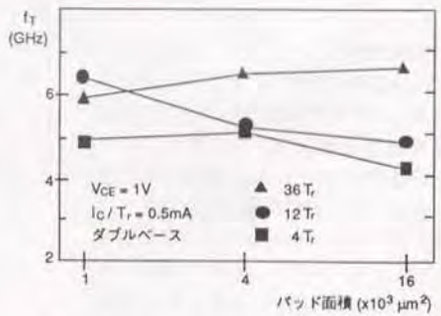


図３.１.１９　ｆＴ測定値のパッド面積依存性

図３.１.１９に示す。トランジスタ個数が１２個の場合には、パッド面積の増加に伴い低下するが、３６個になると、パッド容量に影響されない。

また入力インピーダンスは、トランジスタ個数の増加に伴い減少し、１２個で40～50Ωとなる。hFE-Ic特性は、トランジスタ個数に比例して高電流側に伸びており、３６並列でも、各トランジスタが均一動作していることが確認できた。

以上の結果より、インピーダンス不整合やパッドの寄生容量の影響をほぼ除去して、正確なfT（順方向遷移時間τF）を求める条件は、以下のようになることがわかった。

・トランジスタを１２個以上（エミッタ面積100μm2以上）並列にして測定する。
・厚さ１μm程度のロコス膜上パッドを形成する。面積は、4000μm2程度以下にする。
・パッケージでのキャリブレーションを行い、3GHzまで実測して外挿する。

（2-5）論理ゲートでの実測値との対応

以上の手法で求めた各デバイスパラメータの妥当性と精度を評価するため、ＳＳＴ-２プロセスでＬＣＭＬゲートのインバータを試作し伝搬遅延時間を測定した。そして、図3.1.14に示したトランジスタモデルのミュレーションと比較した。結果を図３.１.２０に示す。（Ａ）はエミッタ面積1.5×2.5μm2の場合、（Ｂ）はエミッタ面積1.5×6.5μm2の場合である。実験値とシミュレーションとは、極めて良く一致していることがわかる。この理由は、シリーズ抵抗 rE, rC1, rbb'、電流利得 hFE、順方向遷移時間τＦに電流依存性を取り入れていることと、接合容量等各パラメータの絶対値を正確に求めているからである。

なお順方向遷移時間τＦについては、トランジスタ（Ａ）を５０個並列（187μm2）に

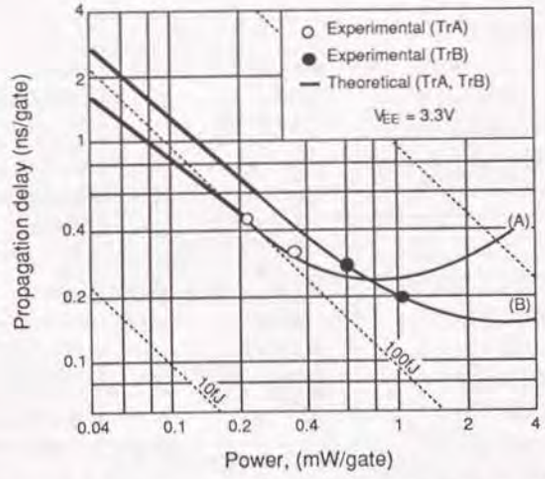


図３.１.２０　本測定システムで求めたトランジスタパラメータによるＬＣＭＬゲートのシミュレーションと実験値との対応

動作させてfTを測定し、「1/Ic-1/fT」特性から求めた。このときVCEは１ｖとし、実際のＬＣＭＬゲートの動作に近い状態で測定した。測定値は、コレクタ電流が、0.1mA, 0.2mA, 0.4mA のとき、それぞれ、18.2ps, 19.5ps, 26.0ps であった。

（3） ＬＳＩの設計試作と評価

以上述べてきた高速化技術を確認するため、実際に乗算器ＬＳＩを開発した。以下その具体的な設計と結果とを述べる。

（3-1）回路設計と試作

内部論理ゲートには、論理振幅0.45Vの2段縦積みＬＣＭＬゲートを採用した。高速性と共に低消費電力とするため、5.2V使用時の消費電力を１W以下と定め、全トリー数が８００であることから、基本ゲート電流を0.2mAとした。ファンアウトについては、4以下，５～７，８以上に分け、それぞれのゲート電流を0.2，0.4，0.6mAとした。

試作算器ＬＳＩを図3.1.21に示す。全回路をマニュアルでレイアウトしており、上から、部分積生成、加算器アレイ、最終加算器の３ブロック構成になっている。チップの４箇所に黒くみえるのは、３０PFの発振止め容量を内蔵した基準電圧発生回路であり、それぞれＬＳＩの1/4に当たる約２００トリーの電流源へＶＣＳレベル（電流源基準電位）を供給し

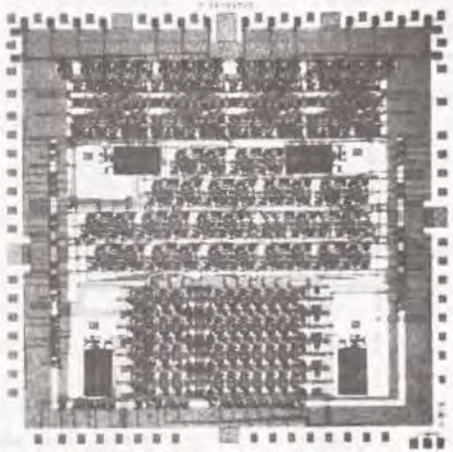


図3.1.21 試作乗算器ＬＳＩのチップ写真

表3.1.5 乗算器ＬＳＩの諸言

項　　目	特　　性
乗算時間	10nsec
消費電力	660mW (-3.3V)
入出力端子数	58
X　入　力	8+(1)*
Y　入　力	8+(1)*
M　入　力	15+(4)*
Z　出　力	15+(3)*
チップセレクタ	3
ゲート数	784
トランジスタ数	2664
素子数	4039
チップ面積	4.5 x 4.4 mm^2
入出カインタフェース	ECLコンパチ
内部ゲート回路	2レベルLCML
電　　源	単一(-3～-5.5V)

*拡張乗算器を構成する時

ている。配線は２層アルミで、１層目が4μm幅3μm間隔、２層目が6μm幅4μm間隔である。ＬＳＩ諸元を表3.1.5に示す。

（3-2）乗算時間の評価[22]

最長遅延経路を活性化するテストパターンにより行った。測定は、チップを７２ピンのＱＩＰ（Quadrature In-line Package）パッケージに封入し、それを、５０Ωのストリップラインを７２対引いた専用測定治具にマウントし、入出力波形の立ち上がり時間を1ns未満に押さえて測定した。

最長遅延パスの論理段数は、入出力レベル変換回路（ＬＣＭＬ-ＥＣＬ）を含めて、２１段である。最長遅延パスに基づいて搭載した各部分回路ＴＥＧのそれぞれの伝搬遅延時間の実測値と、スピードシミュレーションとの対応を表3.1.6に示す。各部分回路および乗算器全体について、両者は良く一致している。これにより、設計技術と評価技術の精度が十分高いことを実証した。

表3.1.6 各部分回路の伝搬遅延時間

		実測値 (nsec)	計測値 (nsec)
1	ECL-LCML変換	0.4	0.25
2	PPG	1.7	1.60
3	MCSA		2.40
	FA	1.0	0.80
4	CLA		4.32
	CLA4	1.8	1.75
5	LCML-ECL変換	1.2	1.04
乗　算　時　間		10.0	9.61

（3-3）拡張乗算器の乗算時間

（2-3）節で述べた拡張乗算器の乗算時間の計算式と、表3.1.6の乗算器ＬＳＩの実測結果とにより、乗算器ＬＳＩをアレイ状に並べた拡張乗算器の乗算時間を推定した。ＬＳＩの実測により、Ｔ０＝１０ns、Ｔ１＝7.5nsを得たので、各語長の拡張乗算器の乗算時間は、表３１.７に示すようになる。１６×１６ビット乗算器が２５ｎｓであり、これは、種々の複雑な音声処理で要求される性能を、満足している。また６４×６４ビット乗算器が１１５ｎｓとなり、本ＬＳＩ開発当時には、従来の８×８ビット乗算器と同程度という、革新的な高速性を達成した。

表3.1.7 拡張乗算器の乗算時間

n	ビット	乗算時間 (nsec)
1	8 x 8	10
2	16 x 16	25
3	24 x 24	40
4	32 x 32	55
8	64 x 64	115
16	128 x 128	235

（付録）拡張乗算入力本数の最大値が２であることの証明

一般に８ｎ×８ｎビットの乗算器を構成した場合の、各ＬＳＩからのすべてのキャリー出力は、図3.1.6から明らかなように、１７桁目から始まって、８桁おきの８ｉ＋９（ｉ＝１、２、３、…　、２ｎ-１）桁に出力され、上記（２ｎ-１）箇所の各桁からの出力数は、それぞれ、（１、２、３、…　、ｎ-１、ｎ、ｎ-１、…　、３、２、１）個である。

また上記キャリーのうち、最上位桁を除く（２ｎ-２）箇所からの出力を、８ビット上位のＬＳＩで吸収するとしたときの、対応する桁に位置するＬＳＩの数は、（２、３、４、…　、ｎ、ｎ-１、ｎ-２、…　、３、２、１）個である。

これより、キャリー入力端子をＬＳＩの９桁目にＣ個設けるとすると、Ｃは、

$$C\times(k-1)\geqq k \qquad k=2、3、4、\cdots、n$$

を満足しなければ成らない。上式を満足する最小の整数値は、Ｃ＝２である。

3.2　浮動小数点演算器の構成法

3.2.1　まえがき

ダイナミックレンジの広い浮動小数点演算は、ユーザインタフェースの点からは理想的な処理系であるが、高速性と回路規模の点で固定小数点に大きく劣っており、信号処理ＬＳＩへの導入は比較的遅れている。特に乗算器と加減算器の高速化は最重要課題である。従来この分野では多くの高速手法が提案されてきた。最近では、仮数部乗算に基数８のブースのアルゴリズムを用いた乗算器[23][24]、仮数部加算に冗長２進を用いた加算器[25]、５段パイプラインの加算器[24]等がある。これらの技術と最新のサブミクロンＬＳＩ技術とにより、演算サイクルは向上し、現在３３ＭＨｚから１００ＭＨｚになってきている。しかしながら、これらの技術を単純にＤＳＰに適用するには問題がある。その理由は、ハードウェア規模が大きいため、複数の乗算器や加算器をオンチップ化して演算回路を構成するときに、全体の面積が大きくなりすぎるからである。すなわち、固定小数点回路に比べて、今だ導入しにくい状況である。

そのため、回路規模をそれほど増大させないで演算速度を改良することが。、強く求められている。そこで、データ形式が指数部、仮数部共「２の補数」のパイプライン乗算器と加減算器について、いくつかの高速化手法を提案し、ＬＳＩの試作によってその効果を確認した。本節では、その高速化手法と結果について述べる。

3.2.2　浮動小数点乗算器の構成法

浮動小数点乗算器は、大きく以下の２つの処理ステップからなっている。

（Ａ）仮数部乗算と指数部加算
（Ｂ）正規化処理と丸め処理

これを処理時間からみると、（Ａ）の仮数部乗算が支配的であり、通常の構成の場合、６５%位を占めている。このため、３段パイプライで構成する場合には、仮数部乗算にパイプラインを２段割当て、（Ｂ）の正規化処理に１段割り当てるとバランスがとれる。しかしＤＳＰの場合、データメモリのサイクルタイムが相対的に長いため、３段構成にして演算サイクルを上げても、ＤＳＰ全体としてのサイクルタイムの向上には、繋がらない状況である。またハードウエア規模を下げる上からも、全体で２段パイプラインにするのが望ましい。

この考えに基づいて、遅延時間を最適に割り振った、２段パイプライン構造の２の補数形式の浮動小数点乗算器を検討した。この構成を図3.2.1に示す。パイプラインの初段では、８ビットの指数部加算と１６ビット仮数部のキャリーセーブ加算、そしてそれに続くキャリー先見伝搬加算の下位１４ビットの加算とを実行する。一般にＤＳＰでは、浮動小数点乗算器が最長遅延になっていないことから、高密度レイアウトの方を優先させ、２次ブースとキャリーセーブ加算とを採用している。キャリー伝搬加算は、２つのパイプラインステージにまたがらせ、初段と次段の遅延時間が等しくなるようなビット配分とした。このた

め、３１ビット乗算結果の残り１７ビットを後段のステージに配分した。

キャリー伝搬加算器では、２種の高速加算器である、ＣＬＡ加算器（キャリー先見加算器）とＣＳＡ加算器（キャリーセレクト加算器）[26]とを比較した。比較の条件を合わせるため、共に語長を３２ビットとし、４ビット単位の先見または選択回路を適用し、0.8μm-ＣＭＯＳで構成した。このときの両加算器の比較を、表3.2.1に示す。加算時間と回路規模では両者ほとんど差がない。しかし、ＣＳＡの方が、４ビット単位で構成したラックの中が規則的になるので、レイアウト面積と、論理検証でのテストベクトルの数が少なくできる点で優れている[27]。さらにより高速化を図るために、ラックの大きさを４ビット固定から、最適ビット配分構成にした場合に、規則性の差は一層大きくなる[28]。ＣＬＡでは、ラック内のビット長に応じて最適な構成を必要とするが、ＣＳＡでは、同じ１ビットのモジュールを再構成するだけでよい。このことは、ＣＡＤを使って加算器を自動合成する場合にも、モジュールの種類が少なくてすみ有利となる[29]。このため、乗算器に限らず、浮動小数点演算器で現われるすべてのキャリー伝搬加算器にＣＳＡを使用した。本浮動小数点乗算器で使用した最適ビット配分の３２ｂキャリーセレクト加算器の構成を図3.2.2に示す。

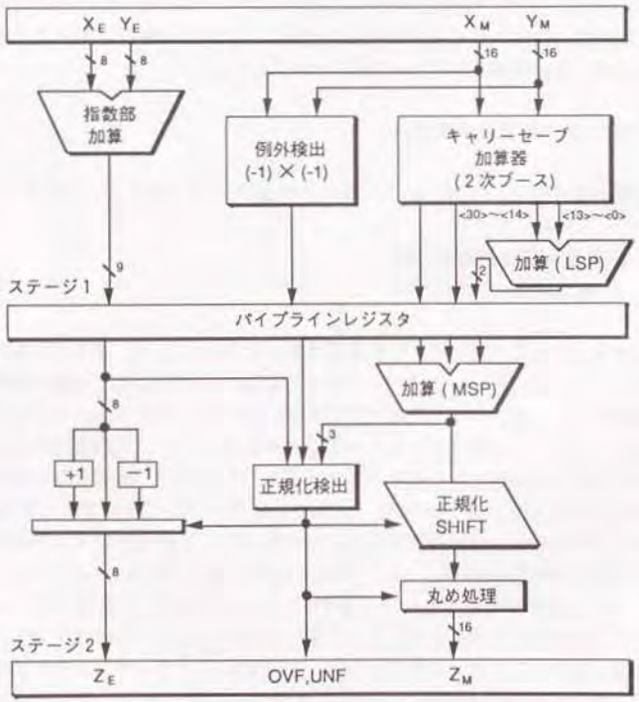
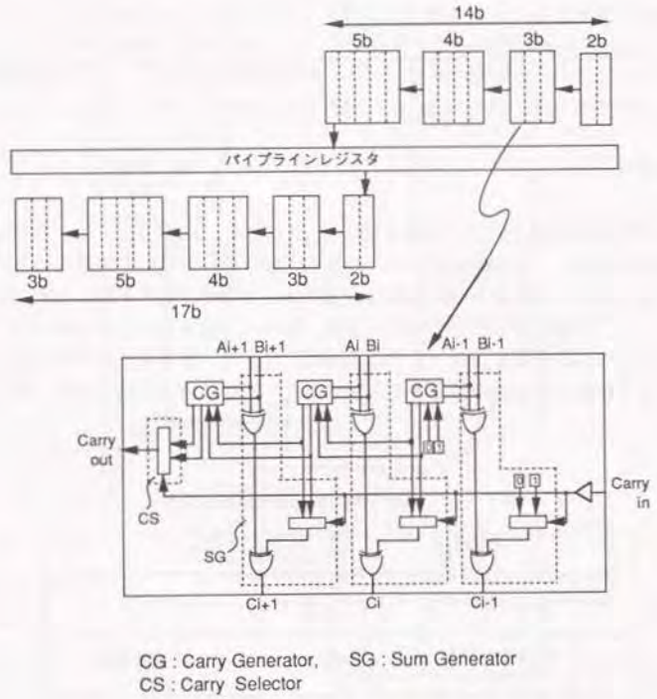


図3.2.1　浮動小数点乗算器のブロック図

表3.2.1　キャリーセレクト加算器 (CSA) と
キャリー先見加算器 (CSA) の比較

項　目 (1)(2)	ＣＳＡ	ＣＬＡ
加算時間	7.2 ns	7.3 ns
回路規模	200 G	196 G
テストベクタ数	16	50
回路の規則性	○	△
総合評価	○	△

(1) ８個の４ｂラック構成の３２ｂ加算器
(2) 0.8μm CMOS プロセスを想定



CG : Carry Generator,　SG : Sum Generator
CS : Carry Selector

図3.2.2　最適ビット配分キャリーセレクト加算器
(16b×16b 乗算器)

3.2.3　浮動小数点加減算器の高速化技術[30]

浮動小数点加減算器の処理は、以下の３つに分けられる。

（α）桁合わせ処理
（β）仮数部の加減算
（γ）正規化処理

この中で、一般に（α）の桁合わせ処理が最も論理段数が多く、最長遅延となっている。その理由の第１は、桁合わせ処理では、桁合わせのシフト数を検出するための指数部減算と、シフト数検出後の桁合わせ処理とが、シリアルに行われるためである。第２は、特に浮動小数点減算の場合での仮数部処理で、被減数の２の補数処理と桁合わせシフトとが、シリアルに実行されるからである。次に論理段数が多いのは、（γ）の正規化処理である。ここでは、正規化シフト数検出と、検出に続く正規化シフトの２つの処理がシリアルに行われるからである。このため、上記３処理をそれぞれパイプラインで区切って、３段のパイプライン加算器を構成する場合、桁合わせ処理と正規化処理の２つの高速化が、重要となる。

この問題を解決するため、本節では、表３.２.２に示す３つの高速化技術を提案する。この中で、（A）の並列桁合わせシフト技術と（B）の桁合わせ減算技術は、（α）の桁合わせ処理の高速化のための技術であり、（C）高速正規化回路は、（γ）の正規化処理の高速化のための技術である。以下（１）～（３）節で各技術の説明をおこなう。

（１）並列桁合わせシフト

従来の桁合わせシフト回路の構成を図３.２.３に示す。この方法では、指数部ＸＥとＹＥの大小と差を検出し、その結果に基づいて小さい方のデータの仮数部を差の分だけ右シフトするのであるが、そのためには、指数部の大小と、その差（シフト数）の両方を求めなければならない。この方法としていくつかあるが、第１は、減算を行って結果の符号ビットが負のときには２の補数を取る方法（２の補数減算）であり、第２は、大小を検出しその結果に基づいて大きい数から小さい数を引く方法である。しかしいずれの方法も、単なる指数部

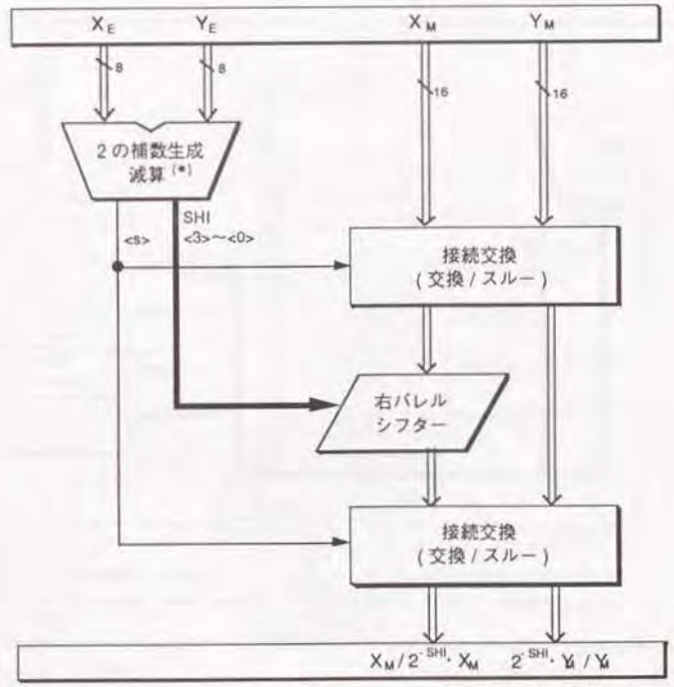
表３.２.２　浮動小数点演加算器の高速化技術

名　　称		内　　容
(A)	並列桁合わせシフト	検出とシフトの並列処理
(B)	桁合わせ減算 （２の補数+シフト+加算）	１の補数+シフト＋補正加算
(C)	高速正規化回路	ビットの反転桁を検出するＡＮＤアレイを最適ビット配分

の減算より、遅延ははるかに大きい。

本技術は、上記の減算と２の補数とシフトの３処理を並列に実行できるように工夫したものである。この構成を図３.２.４に示す[31][32]。仮数部シフタを２個並列に設けていることと、指数部減算器を「１の補数減算器」ですませているのがポイントである。すなわち指数部減算（ＸＥ-ＹＥ）を１の補数減算器で行い、その減算結果を使って、正規化シフトを高速におこなう。また仮数部のシフト結果は、ＸＥとＹＥの大小に応じて２通り準備されており、１の補数減算器のサイン出力によって、大小が定まった時点で選択される構成である。これを式で表現すると、次のようになる。

（＃１）ＸＥ ≧ ＹＥ のとき
　　ＸＭ と 2^{-SHI} ・ ＹＭを選択、
（＃２）ＸＥ ＜ ＹＥ のとき
　　2^{-SHI} ・ ＸＭ とＹＭを選択、



(*) $X_E - Y_E = X_E + \overline{Y}_E + 1$

図３.２.３　従来の桁合わせシフト法

減算を１の補数で行うのは、この減算結果から、正の場合と負の場合のシフト信号（SHI<3>～SHI<0>）を、高速に得られるからである。すなわち、正の場合にはＬＳＢ（Least Significant Bit）であるSHI<0>のみを反転させることにより、また負の場合には全ビットを反転させることにより容易に得られる。２の補数減算の場合にはこのような簡単な規則はない。

また仮数部シフタについては、ＬＳＢから順にシフトされる構成にしてあるので、指数部の減算と仮数部のシフトとを並列に実行できる。このため全体の遅延は、指数部減算器の減算時間と「2-1セレクタ」の通過時間だけになり、従来の方法と比較して、４０％以上の高速化を達成することができる。

（２）桁合わせ減算の高速化

ここでは、シフト数の検出結果に基づき、２つの仮数部（XM、YM）間で、次式で示す「シフト＋減算」処理を行う。

$$X_M+2^{-SHI}\cdot(\overline{Y}_M+1)+2^{-1} \qquad (3.2.1)$$

ここに2^{-1}は、４捨五入丸めを行うための加算項である。これを行う従来の回路を図３.２.５（ａ）に示す。この従来の方法では、２つのキャリー伝搬の演算を必要とする。ひとつは、減数YMの２の補数処理、もうひとつは加算処理である。そしてこの２つの伝搬演算の間に、桁合わせシフトを行わなければならない。このため、上記３つの処理がシリアルに実行されることになり、高速性が制限されている。

この処理を高速化するために考案した、高速桁合わせ減算法（ＨＳＳ：High-speed Subtraction with Scaling）を図３.２.５（ｂ）に示す。この方法は従来の処理を「１の補数＋桁合わせシフト＋ファクターＣを伴う加算」の３処理に置き換えることにより、始めの２の補数演算に相当する部分の演算時間を、１の補数演算の時間まで高速化している。この処理とファクターＣの意味を、（３.２.２）式に示す。

$$X_M+2^{-SHI}\cdot\overline{Y}_M+C+2^{-1} \qquad (3.2.2)$$

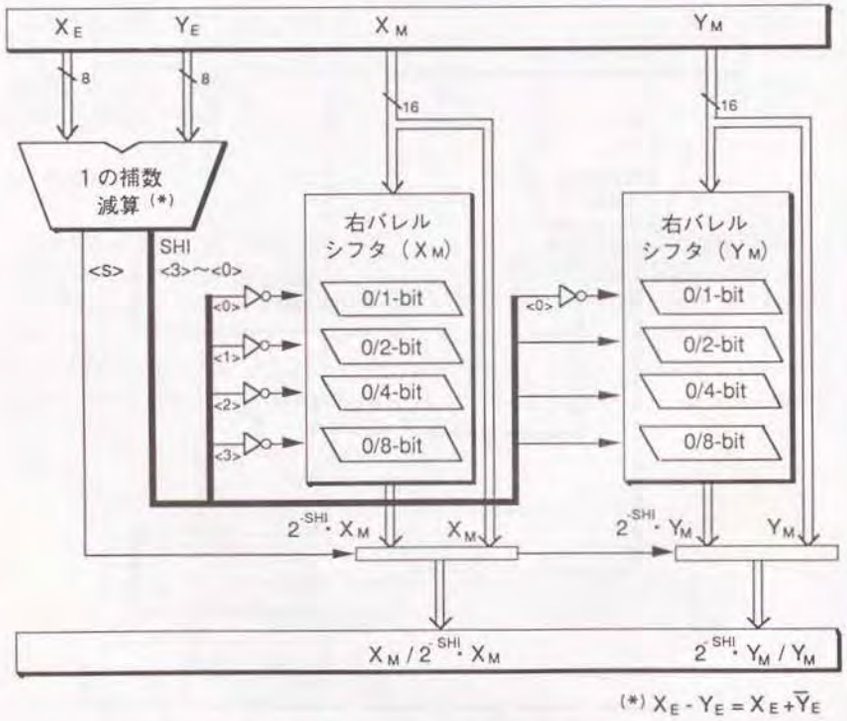


図３.２.４　考案した高速桁合わせシフト法

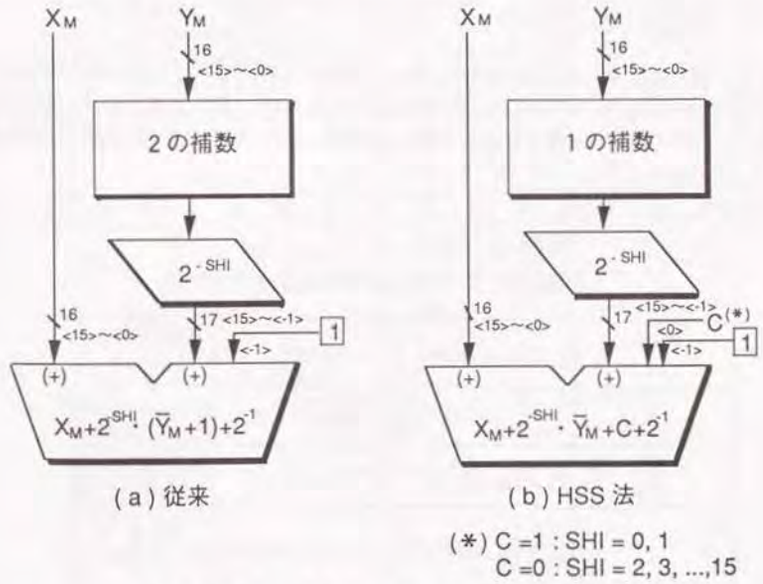


図３.２.５　高速スケーリング減算法 (HSS)

ここに、ファクターCは、

C＝1：（YM＜0 & SHI＝0, 1）のとき
C＝0：（YM＜0 & SHI＝2, 3, ..., 15）または（YM ≧ 0）のとき

である。1の補数処理は全ビットの反転処理のみですむため、2の補数処理のようなキャリー伝搬を伴わない。このため従来方式に比べて高速となる。この効果により、桁合わせ減算器全体として、約30％の高速化が実現できている。

しかしながらこの方法は、「2の補数＋シフト」を「1の補数＋シフト＋ファクターCの補正加算」で置き換えているため、幾分かの誤差を伴う。この誤差の性質を、表3.2.3に示す。これは減数YMのシフト数により、以下の3つの場合に分けられる。

（A）SHI＝0：シフトがない場合であり、誤差は生じない。
（B）SHI＝1：右1ビットシフトの場合であり、従来方式より2^{-1}大きくなる。
（C）SHI＝2～15：右2ビットから15ビットシフトの場合であり、従来方式より2^{-i} (i=2,3,...,15) 小さくなる。

シフト数SHIは、統計的には0から15まで均一に生起すると考えられるので、種々の演算を繰り返したときの統計的な誤差は（3.2.3）式のようになる。

$$2^{-1}+\sum_{i=2}^{15}\left(-2^{-i}\right) = 2^{-15} \cong 0 \qquad (3.2.3)$$

このように、統計的には、累積誤差は無視でき得る程度に小さくなる。通常偏りのある誤差は致命的な欠点となるが、本手法は極めて不偏丸めに近いため、標準規格に基づく演算を要する場合や、特別の高精度を要する場合を除いて問題ないと言える。広く、一般の信号処理へ適用できる技術である。

表3.2.3　HSS法での計算誤差

SHIFT (bit)	Conventional	HSS	Difference (Error)
0	$\overline{Y}_M + 1 + 2^{-1}$	$\overline{Y}_M + 1 + 2^{-1}$	0
1	$2^{-1}(\overline{Y}_M + 1) + 2^{-1}$	$2^{-1}(\overline{Y}_M + 2) + 2^{-1}$	2^{-1}
i (*)	$2^{-i}(\overline{Y}_M + 1) + 2^{-1}$	$2^{-i}(\overline{Y}_M) + 2^{-1}$	$\Sigma(-2^{-i})$ $\fallingdotseq -2^{-1}$

(*) i＝2, 3, ... , 15

（3）高速正規化回路[33]

正規化処理では、仮数部の加算結果を得て、正規化量を検出し、検出量に基づいて正規化シフトを行う。この中で、正規化量を検出する、エンコーダの高速化が課題である。この回路には、加算結果をMSB（Most Significant Bit）から順次見ていって、最初にビット反転が生じた桁を検出し、その出力を2進変換してシフト数を出力する機能が必要である。そして従来は、このビット反転位置の検出回路を、ANDのリップルとEXORアレイからなる回路か、または4ビット単位のCLA加算器を変形した先見検出回路で構成していた。

しかしこれら従来構成では、十分な性能が得られていない。例えば、指数部8ビット、仮数16ビットのデータ構造で、2μmCMOSプロセスを想定したとき、エンコーダ回路の全加算時間に占める割合は、リップル構成の場合47％（65ns / 138ns）、先見検出の場合26％（36ns / 104ns）を占めている。

そこで、ビットの反転桁を検出するANDアレイに、ビット配分を最適化したキャリー

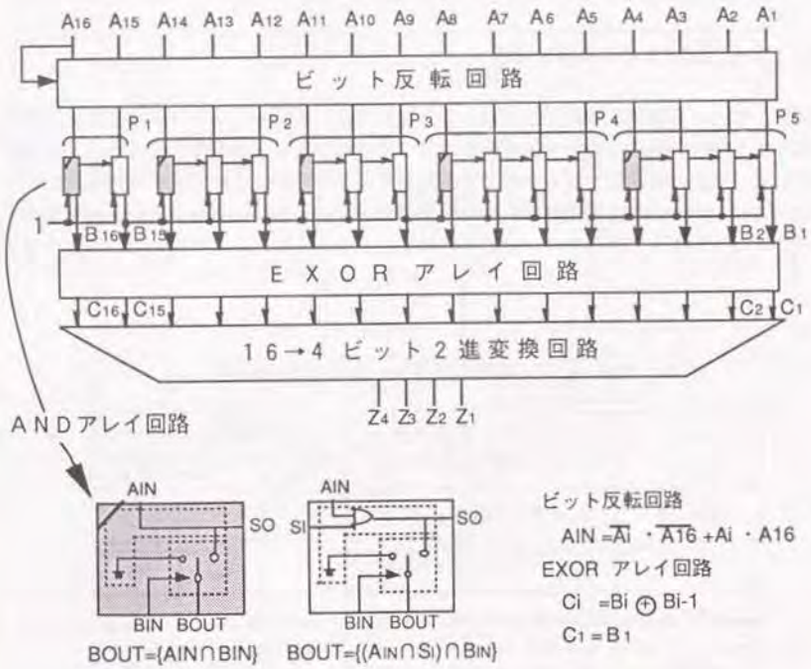


図3.2.6　高速ビット反転検出回路（S1～S6）を用いたエンコーダ

選択加算器による２次元処理の高速化手法を取り入れることとした。この回路を図３.２.６に示す[34]。一例として、１６ビット構成の場合について示してある。まず、１６ビットの入力データＡｉ（i=1,2,...,16）を、常にＭＳＢが”１”となるように規格化（ビット反転回路）して、後段のＡＮＤアレイ回路への入力AINを得る。このＡＮＤアレイ回路が、５つの部分アレイＰｊ（j=1,2,...,5）に分割されているのが要点である。各部分アレイでは、上位アレイのＬＳＢ（Least Significant Bit）出力（BOUT）が、０の場合と１の場合について、アレイ内部の各桁の出力と、下位アレイへの出力とをあらかじめ求めておく。そして、上位アレイからの実際のＬＳＢ出力が決まると、ただちに、真の出力を選択する構成になっている。各部分アレイPjの桁数は、どのアレイから見ても、上位アレイから伝搬してくる選択信号の遅延と、アレイ内部の各桁出力までの遅延とができるだけ等しくなるように最適化している。１６ビット構成の場合、P1＝２ビット，P2＝３ビット，P3＝３ビット，P4＝４ビット，P5＝４ビットである。

このように、各部分アレイの桁数が最適配分されていることと、先見方式に比べてゲートのファンイン数が小さいこととから、高速化されている。先ほどの２μｍＣＭＯＳプロセスの場合、遅延時間は３１ｎｓとなり、従来の先見方式に比べて５ｎｓ（１６％）高速化できている。また回路の素子数から見ても、先見方式より少なく、優位である。リップル方式に比べても僅か１３％（２０ゲート）の増加にすぎない。

（４）　３入力加減算器の高速化技術

固定小数点と同様に、多入力の浮動小数点加算器は、累算やバタフライ演算等の信号処理演算を行う場合、極めて強力である。その第１の理由は、２入力加算器を多数用いる場合と比較して、回路の共通化が図れるので回路規模を小さくでき、その分別の回路を余分に搭載できることである。第２の理由は、ベクトル演算を行うときに、複合演算を利用してパイプライン段数を小さくできるため、演算の絶対遅延を小さくできることである。本節では、

表３.２.４　補数生成ブロックの動作モード

ASU ユニット	バタフライ演算での ADD / SUB 演算モード	１の補数 / スルー		
		X_M	Y_M	Z_M
ASU 0	$C_R = A_R + B_R W_R - B_I W_I$	－	－	○
ASU 1	$D_R = A_R - B_R W_R + B_I W_I$	－	○	－
ASU 2	$C_I = A_I + B_R W_I + B_I W_R$	－	－	－
ASU 3	$D_I = A_I - B_R W_I - B_I W_R$	－	○	○

○：１の補数，　－：スルー

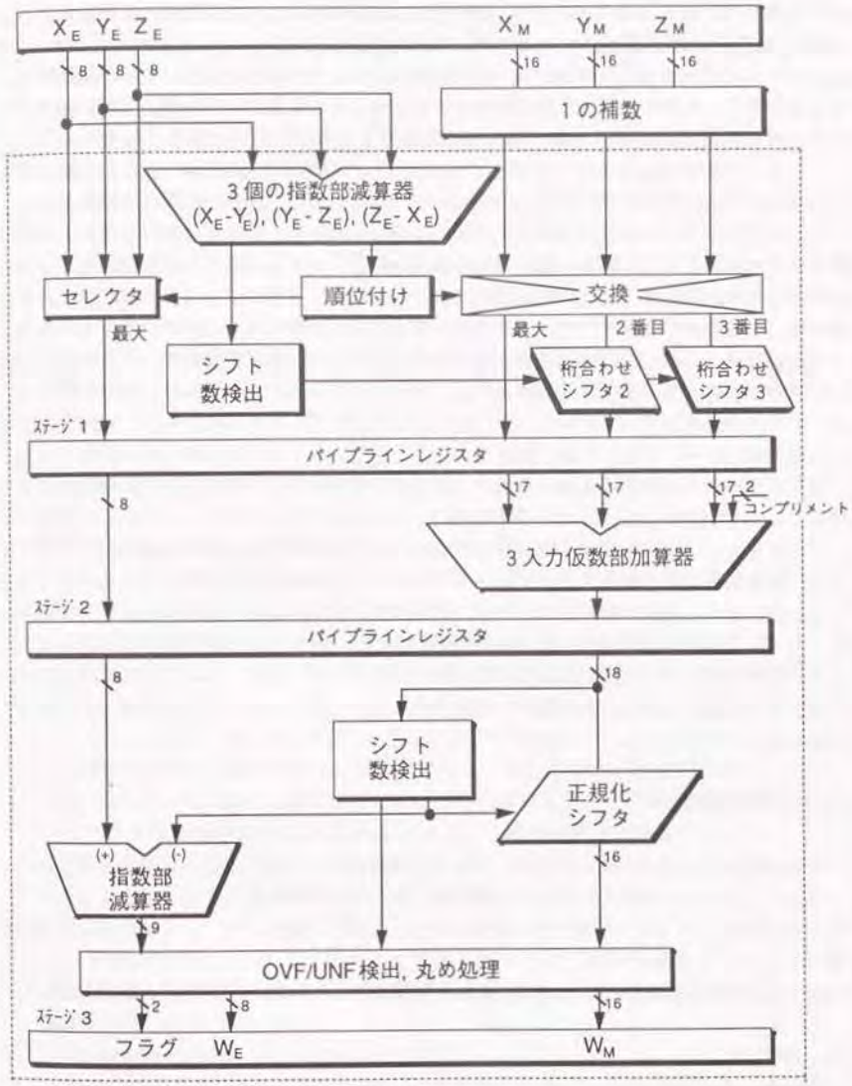


図３.２.７　３入力浮動小数点加減算器のブロック図

複素バタフライ演算を１サイクルで実行できるバタフライ演算ユニットのために開発した、３入力加減算器を取り上げ、その高速設計技術について述べる。

バタフライ演算で必要な４種の加減算器を、表3.2.4に示す。各加減算器の共通性を高めるため、ここでは、３入力加算器の基本ブロックの入力に、２の補数回路を付加し、「加算／減算」に対応して、「コンプリメント（Complement）／パススルー（pass-through）」を選択する構成とした。さらに高速化のため、従来行われている「２の補数＋桁合わせシフト」処理を、「１の補数＋桁合わせシフト＋ＬＳＢへの１加算」処理に置き換えた。これは、前節で述べた、高速桁合わせ減算を３入力に拡張したものである。

この３入力加減算器を３段パイプラインで構成したブロック図を図3.2.7に示す。初段のステージでは、３入力Ｘ，Ｙ，Ｚの桁合わせ処理を行う。まず、３入力の指数部、ＸＥ，ＹＥ，ＺＥの中から最大数を検出する。次にその最大数と、２番数および３番数との差分をそれぞれ計算して、対応する仮数部を差分量だけ右シフトする。この正規化処理は、３入力の指数部の差分計算（ＸＥ-ＹＥ，ＹＥ-ＺＥ，ＺＥ-ＸＥ）を含めて、３数が並列に実行できるため、その処理時間は、２入力加算器の正規化処理の時間とほとんど同等である。また２入力加減算器と比較した回路規模は、差分検出の減算器が３倍、桁合わせシフタが２倍、セレクタ系が２倍、制御系が２倍弱であり、このステージでは、全体でほぼ２倍に収まっている。

２段目のステージでは、３数の仮数部の加算を行う。ＬＳＢへの２本のコンプリメントは、減算のための２の補数である。最後に３段目のステージでは、正規化処理を行う。この回路は２入力加減算器の場合と全く同様である。

以上により、３入力の浮動小数点加減算器を、２入力の場合と同等の３段のパイプラインステージで構成でき、かつ、パイプラインサイクルタイムを、ほぼ同等に高速化できることを示した。また、回路規模の点で、２入力加減算器と比較すると、初段は約２倍、２段目は約1.5倍、３段目は同一のため、加減算器全体として約５０％の増加にすぎない。これは、３入力加減算器を２入力加減算器２個のカスコード接続で構成した場合と比較すると、パイプライン段数は、６段から３段へ、回路規模では、１００％増から５０％増へ、それぞれ大幅に改良されている。

（５）評価結果[30]

本節で提案した浮動小数点乗算器、および加減算器の４つの高速化４技術の効果を確認するため、0.8μm-CMOSにより、乗算器４個と３入力加減算器４個からなる６段パイプラインの複素バタフライバタフライ演算ユニットを試作し測定した。シミュレーションで予測した４０ＭＨｚ動作保証（バタフライ演算ユニットとしては、400MFLOPS：Mega FLoating-point Operation Per Secondと世界最高速の性能）を確認し、高性能を実証した。

3.3　むすび

本章では、高速演算回路の構成技術について述べた。成果は大きく２つに分けられる。１つは、信号処理演算の基本である並列乗算器の高速化、２つは、今後の信号処理で増々重要となる、浮動小数点乗算器と加減算器の高速化である。

並列乗算器では、完全拡張機能を搭載した高速乗算器の構成法を提案し、それを、種々の回路技術とデバイパラメータ評価技術を用いて最適設計し、開発当時、世界で最高速を実現した。この完全拡張機能は、並列乗算器アレイを作って、種々の語長の並列乗算器を構成する上で、有用である。例えば、完全拡張機能のある８ビットの並列乗算器を６４個搭載すると、精度のいらない場合には、８ビット精度での６４並列演算を行えるし、１６ビット精度の場合には、１６並列で演算できる。例としてビデオ符号化を取り上げると、前者は、動きベクトル検出の距離計算に、後者はＤＣＴ（Discreate Cosine Transform）演算にと、それぞれ時分割で使い分けることができる。

浮動小数点乗算器と浮動小数点加減算器では、４つの高速化技術を提案した。従来の、冗長２進演算や超多段パイプライン等の高速化技術が、回路規模の増加が大きいため、信号処理ＬＳＩで使いにくい点を重視し、２の補数形式で、適度なパイプライン段数という範囲内で高速化を図った。このため、従来の浮動小数点演算器に比較して、回路規模的に遜色なく高速化を達成している。非常に実用的な技術といえる。さらに開発した技術を３入力の加減算器へ拡張し、高密度で高速な複素バタフライ演算ユニットを設計した。

複素バタフライ演算ユニットの中で、３入力浮動小数点加減算器に使われている高速化技術の貢献を、従来技術と比較してまとめると、以下のようになる。まず基本構成技術として、以下の効果がある。

・桁合わせ処理　　：４０％の高速化
・桁合わせ減算　　：３０％の高速化
・正規化シフト回路：１６％の高速化

通常、桁合わせ処理が最長遅延になっているので、パイプライン段数を同じにすれば、演算サイクルで４０％の向上が見込める。さらに３入力にしたとき、

・３入力加減算器技術：２入力と比較して遅延の増加はほとんどなし

の効果があるので、従来では、パイプライン段数を増やすのが必須であるが、本技術により、段数を３段に維持することができる。段数の少ないことにより、実際にバタフライ演算を行ったとき、パイプラインオーバヘッドを少なくできる。

上記技術の成果を実証するため、複素バタフライ演算ユニットを、実際に0.8μm-CMOSで試作した。４０ＭＨｚ動作で動作し、４００ＭＦＬＯＰＳの性能（保証値）を達成した。これは世界最高速のＦＦＴ演算性能であり、技術の有用性を実証できた。

3.4　参考文献

[1] G.W.Mclver et al., "A Monolithic 16×16 Digital multiplier", in Proc. IEEE-ISSCC'74, SESSION Ⅳ, (Feb., 1974).
[2] "8×8 multipliers", MMI Data Sheets, (Aug.1977).
[3] J.M.Triboler and R.E.Crociere, "Frequency Domain Coding of Speech", IEEE Trans., ASSP-27, pp.512-530, (1979).
[4] 誉田、板倉、" 音声の適応ビット割当て予測符号化" 、音響学会音声研究会資料、

S80-2, (Apr.1980).
[5] 高橋、中村、電子情報通信学会誌、Vol.73, No.8, pp.836-841, (Aug., 1990).
[6] J. Takahashi, S. Hattori et al., "A Ring Array Processor Architecture for Highly parrallel Dynamic Time Warping," IEEE ASSP-34, pp.1310-1318, 1986
[7] H.Yamauchi, Y.Tashiro, et al., "Architecture and Implementation of a Highly-Parallel Single-Chip Video DSP", IEEE Trans. on Circuit and Systems for Video Technology, Vol.CSVT-SP92, (Apr.,1992).
[8] O.L.Macsorley, "High-speed Arithmetic in Binary Computers", Proc. IRE, 49, pp.67-91, (Jan.,1961).
[9] L.P.Rubinfield, "A Proof of the Modified Booth's Algorithm for Multiplication", IEEE Trans., C-24, pp.1014-1015, (Oct.,1075).
[10] 山内、二階堂、中島、小林、酒井、" ＳＳＴ技術を用いた超高速８×８乗算器" 、昭和５６年度電子情報通信学会半導体材料部門全国大会, 96, (1981).
[11] H.Yamauchi, T.Nikaido, T.Nakashima, Y.Kobayashi, T.Sakai, "10ns 8×8 Multiplier LSI Using Super Self-Aligned Process Technology", IEEE J. Solid-State Circuits, Vol.SC-18,No.2, pp.204-210, (Apr.,1983).
[12] C.S.Wallace, "A Suggestion for Parallel Multipliers", IEEE Trans., EC-13, pp.14-17, (Feb.,1964).
[13] 山内、酒井、二階堂、" ＮＴＬゲートのスピードアップコンデンサの最適化の解析" 、昭和５６年度電子情報通信学会総合全国大会, 335, (1981).
[14] 向井、" 分布しきい値型論理回路" 、電子情報通信学会半導体トランジスタ研究会資料, C-472, p.1268, (1971).
[15] 二階堂、石谷、山内、" 超高速ＬＳＩにおける電流切り替え型回路の設計" 、昭和５４年度電子情報通信学会総合全国大会, 384, (Mar.,1978).
[16] 山内、小林、酒井、" ＳＳＴ-２によるＬＣＭＬゲートの特性" 、昭和５６年度電子情報通信学会総合全国大会, 334, (1981).
[17] 山内、小中、酒井、" 高性能安定化容量を内蔵したＬＣＭＬゲートの特性" 、昭和５７年度電子情報通信学会総合全国大会, 385, (1982).
[18] 山内、小中、酒井、" バイポーラトランジスタ，デバイスパラメータ評価システム" 、電子情報通信学会技術研究報告, SSD81-20, pp.93-100, (1981).
[19] B.Kulke, and S.L.Miller, "Accurate Measurement of Emitter and Collector Series Resistance in Transistors", IRE Proceedings (Lett.), vol.45, p.90, (1957).
[20] W.C.Davidon, "Variable Metric Method for Minimization", A.E.C.R &D.Report, ANL-5990, (1959).
[21] R.Fletcher, M.J.D.Powell, "A Rapidly Convergent Descent Method for Minimization", The Computer Journal, 6, pp.163-168, (1963).
[22] 山内、二階堂、中島、小林、酒井、" 完全拡張機能を内蔵した超高速乗算器ＬＳＩ" 、電子情報通信学会技術研究報告, SSD81-52, pp.7-14, (1981).
[23] B.J.Bencshneider, W.J.Bowhill et al, "A Pipelined 50-MHz CMOS 64-bit Floating-Point Arithmetic Processor", IEEE Journal of SC., vol.24, no.5, pp.1317-1323, 1989.

[24] F.Okamoto, Y.Hagihawa et al, "A 200-MFLOPS 100-MHz 64-bit BiCMOS Vector-Pipelined-Processor" ,ISSCC'91, pp.256-257, 1991.
[25] H.Edamatsu, S.Kuninobu et al, "A 33-MFLOPS Floating-Point Processor using redundant Binary Representation", ISSCC'88, pp152-153, 1988.
[26] O.J.Bedrij, "Carry-Select Adders", IRE Trans., EC-11, No.3, pp. 340-346, 1962.
[27] T.Ikenaga, H.Yamauchi, J.Takahashi, "A Generation Method of Complete Function Test Patterns for Adders", IEICE-FTS-91-27, pp. 63-70, 1991. (in Japanese)
[28] K.Kaneko, T.Okamoto et al, "A VLSI RISC with 20-MFLOPS, 64-bit Floating-Point Unit", IEEE Journal of SC., vol.24, no.5, pp.1331-1340, 1989.
[29] K.Takeya et al, "A Generator for High-Density macrocells with Hierarchical Structure," in Proc. CICC, 1989, pp.23.5.1-23.5.4.
[30] H.Yamauchi, H.Miyanaga, "Architecture of a Floating-Point Butterfly Execution Unit in a 400-MFLOPS Processor VLSI and its Implementation", IEICE Transactions, Vol. E74, No.12, pp.3852-3860, (Nov.,1991).
[31] 山内、金子、" 冗長コードを用いた浮動小数点加算器の検討" 、昭和６０年度電子情報通信学会情報システム部門全国大会, 429, (1985).
[32] 金子、山内、" 高速浮動小数点ＶＬＳＩシグナルプロセッサ：ＤＳＳＰ１" 、電子情報通信学会論文誌, Vol. J72-B-I No.1, pp.67-73, (1989).
[33] 鈴木、山内、" 高速ビット検出回路を備えた浮動小数点加算器" 、昭和５７年度電子情報通信学会通信部門全国大会、No.66, (1982).
[34] 守谷、赤沢、" ＣＭＯＳ高速加算器の構成法" 、昭和５６年度電子情報通信学会半導体材料部門全国大会、No.95, (1981).

第4章　データメモリアドレス生成技術

4.1　ＤＳＰのアドレス生成技術の分類

ＤＳＰのデータメモリのアドレス生成法は、大きく、スカラ処理用とベクタ処理用に分類することができる。これを表4.1.1に示す。スカラ処理では、従来のマイクロプロセッサのアドレス生成と同様、直接アドレス生成と間接アドレス生成とを備えて、ベクトル化できないＭＩＳＣ（MISCellaneous）処理を逐次的に実行できる様にしている。このため、例えばレジスタ間接アドレスを見ると、汎用性が高いＤＳＰほど、バスに繋がるすべてのレジスタを対象に、レジスタ間接アドレスができるようにして、種々のスカラ処理を柔軟にこなせる様にしている。

一方信号処理の最大の特長は、種々のベクタ演算を高速に実行することである。このためＤＳＰは、ベクタ演算の特性に合わせて、必要なデータを連続的にアクセスして、パイプライン演算回路へ流し込む機能を備えている。この必要なデータの連続アクセスの機能が、データメモリの修飾アドレス生成機能であり、ＤＳＰの大きな特長の１つになっている。

この修飾アドレスの生成は、表4.1.1に示すように、対象とする信号処理の領域に応じて、1次元，2次元，3次元に分けて考えることができる。1次元アドレス生成は、1次元ベクタデータを対象とするものであり、音声処理，通信処理等、波形を対象とする従来の信号処理と、画像等の多次元信号処理を、独立な1次元信号処理に分解して処理する場合等、信号処理一般に現われるものである。このアドス生成技術を、4.2節で詳述する。また、この技術を基本にして開発した、音声信号処理ＤＳＰのアーキテクチャ技術を、4.3節で述べる。

２次元アドレス生成は、画像等の２次元データの任意の領域を切り出して処理する場合に必要なものである。本研究では、２次元フーリエ変換でのアドレス生成技術（4.4節）と、画像をブロック単位で処理するアルゴリズムに基づいて開発した、ビデオ符号化ＤＳＰでのアドレス生成技術を取り上げ（4.6節）、詳述する。３次元アドレス生成は、レイトレーシング等、３次元グラフィクスでの空間切り出しで必要であるが、２次元アドレス生成の延長技術と考えてよい。

表4.1.1　データメモリのアドレス生成の分類

アドレッシング種別			内　容	本章の内容
スカラ処理	直接		マイクロフィールドのオペランドで直接指定	
	間接	レジスタ	アドレスが格納されたレジスタを指定	
		メモリ	アドレスが格納されたメモリの番地を指定	
ベクタ処理	修飾	1次元	アドレスポインタの連続移動（信号処理）	音声信号処理ＤＳＰ（4.2節）
		2次元	同上（画像処理）	2次元フーリエ変換ＬＳＩ（4.4節） ビデオ符号化ＤＳＰ（4.6節）
		3次元	同上（3次元グラフィクス）	

4.2　音声信号処理ＤＳＰのアドレス生成技術

4.2.1　音声信号処理でのアドレス生成に関する要求条件

音声信号処理のアルゴリズムでは、メモリ内のデータや、定数テーブルのアクセスに、種々の高度な修飾アドレスが要求される。これらをアドレスモードとして整理し、表4.2.1に示す。

ディジタルフィルタ，窓関数，自己相関等の処理では、アドレスモードは通常の増加・減少だけのインクリメント機能だけで十分である。ただし、処理の並列パイプライン化を図るには、最大3アドレス演算ができなければならない。これは常時、２データを読みだし、パイプライン演算部で加工して、結果の１データを書き込むためであり、独立した3個のアドレスポインタを必要とする。

適応予測符号化は、過去の幾つかのサンプル点の値（データ系列）からの予測値と、現サンプル点の値との差分信号を符号化する方式である。この処理では、データ系列と予測係数列とで積和演算を行って予測信号を計算するが、このとき、サンプルデータを格納しておくデータメモリのアクセスに、高度なアドレス生成機能を必要とする。すなわち、符号化のステップが進むにつれて、予測次数以前のサンプルデータは、漸次不要となるので、メモリ上の不要となったサンプルデータを新しいサンプルデータで置換して、メモリを周期的に使

表4.2.1　音声信号処理のアドレスモード

アドレスモード	ベクタ演算種		アドレスポインタ	演算形式	使用される音声符号化方式
インクリメント	フィルタリング		2	積和	各種
	自己相関		2	積和	
	正規化		2	除算／定数倍	
	窓関数		3	積	
	ダウンサンプリング		2	転送	LSP, APC-AB
サイクル	予測符号化		2	積和	LPC, LSP, APC-AB
モジュロ	ＤＦＴ		2	積和	PARCOR, LSP, APC-AB
	ＤＣＴ		2	積和	ATC
モジュロ ビットリバース	ＦＦＴ	バタフライ	2	積和	
		ビットリバース	2	転送	

用していかなければならない。このサンプルデータの読み出しのとき、予測次数の周期でベースアドレスを更新させるアドレス生成機能（サイクルアドレス）が、必要である。

離散的フーリエ変換（ＤＦＴ）と離散的コサイン変換（ＤＣＴ）は、一定区間のサンプルデータを使って、そのスペクトルを求める処理である。このとき、三角関数テーブルとサンプルデータとの積和演算が行われる。この三角関数テーブルを読みだすとき、高度なアドレス生成機能が必要である。すなわち、三角関数は周期２πの周期関数であり、また値の絶対値に着目すると、偏角０からπまでの周期関数でもある。そこで通常、πの周期性の方を使用して、三角関数テーブルのメモリ容量を節約するが、このテーブルの読みだしのときに、０からπまでのデータをモジュロとして、繰り返し読みだすアドレス生成機能（モジュロアドレス）が、必須である。

高速フーリエ変換（ＦＦＴ）では、各ステージでのバタフライ演算と、最終ステージでデータ系列の順序を入れ替えるビットリバース処理とを行う。バタフライ演算では、係数用の三角関数テーブルの参照に、上記モジュロアドレスが必要であり、またデータ用には、複素数データを読みだすアドレス生成機能が必要である。またビットリバース処理では、当然、ビットリバース用のアドレス生成機能（ビットリバースアドレス）を必要とする。

このように音声信号処理では、インクリメント機能の他に、各種の高度な修飾アドレスを要求とする。表４.２.１では、アドレスモードに加えて、各信号処理に必要なアドレスポインタ数，演算形式，および使用される代表的な音声符号化方式を併記した。

４.２.２　アドレス生成ユニット

（１）基本的考え方[1]

表４.２.２　ＡＡＵに搭載したアドレスモード

アドレスモード	演算内容	適用例
インクリメント	A+I→A	窓関数
モジュロ	MOD(A)→A	DFT,DCT
ビットリバース	BR(A)→A	FFT(ビットリバース)
イデレーション	IT(A)→A	複素ベクトル乗算
		FFT(バタフライ)
ベースアドレス更新	A0+△A→A0	DCT,自己相関
インクリメント更新	I+△I→I	DFT,DCT

A0 ：ベースアドレス　　I ：インクリメント

△A：ベースアドレス増分　I0：インクリメント増分

通常ＤＳＰでは、３個のアドレスポインタを搭載して、３アドレス演算まで可能とするが、この内の１個のみに、本高機能アドレス生成ユニットを搭載し、残りの２つは、インクリメント機能のみとする。この理由は、高機能アドレス生成ユニットは、インクリメントのみのアドレス生成ユニットに比べて、回路規模が３倍程度になるので、必要最低限の所にのみ使用したいことと、通常、リードアドレスの一方のみに高機能アドレスを発生させるだけで、ほとんどすべての信号処理演算に対処できるからである。

高機能アドレス生成ユニットのアドレスモードを表４.２.２に示す。この表のすべてのアドレスを、必要最小限の回路で生成するため、以下の２つの方針を定めた。

（Ａ）基本アドレスモードの組み合わせで、種々のアドレスを生成する。
（Ｂ）プログラム制御の利点を活かし、２重ループ制御と組み合わせる。

（Ａ）は、全アドレスの中で、インクリメントが最も基本的なアドレス演算であることに着目したものである。すなわち、インクリメント機能に、モジュロマスク、ビットリバーシング、イテレーションの各機能を独立にＯＮ／ＯＦＦさせることで、インクリメント、モジュロ、ビットリバース、イテレーションの４アドレスを自由に生成させるものである。具体的には、命令でアドレスモードを指定して、インクリメント量，モジュロ周期，バタフライサイズ，イテレーション周期等のパラメータをセットをすることにより、所望のアドレスが各演算サイクルごとに連続的に生成されるようにする。

（Ｂ）は、ベースアドレスの更新とインクリメントの更新を、２重ループ処理と同期させて行うものである。例えば、自己相関演算の場合には、内側ループで０次の相関係数を求め、次に１次の相関係数を求めるループに入る時に合わせて、開始アドレスを自動的にシフトさせる。また、ＤＦＴ演算でテーブル参照を行う場合、１次係数を求める１回目のアクセスの時には、連続したアドレスで読みだし、２次係数を求める２回目のアクセスの時には、１跳びのアドレスで読みだすというようにインクリメント値を変化させるが、これは、内側ループに入るのに同期して、インクリメント値を増加させることで実現する。

（２）アドレス生成ユニットの構成[2]

イテレーションとモジュロの具体的なアドレス生成法の概念を図４.２.１（ａ）に示す。イテレーションでは、ある１組のアドレスを、ある回数だけ繰り返し生成させるが、特に、アドレスの数と繰り返し回数とが、共に２の冪乗のとき、簡単な回路で実現することができる。今、１組のアドレス（イテレーショングループ：Iteration Group）を、繰り返し回数（イテレーション数：Iteration Number）だけ、繰り返し生成させるとする。これを（４.２.１）式の様に表現する。

$$\text{Iteration Group} = 2^{M} \qquad (4.2.1)$$

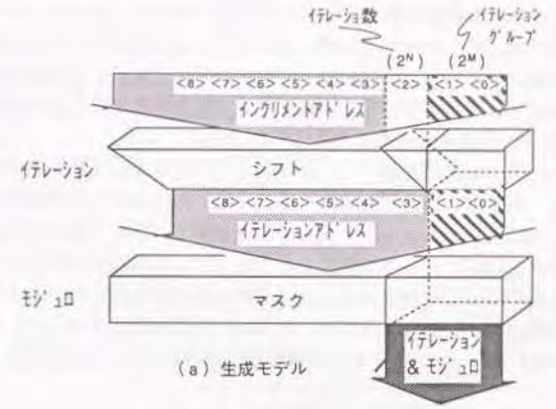
$$\text{Iteration Number} = 2^{N}$$

ここで、下位Ｍビットはそのままにして、上位桁の内、隣接するＮビットが除去されるよ

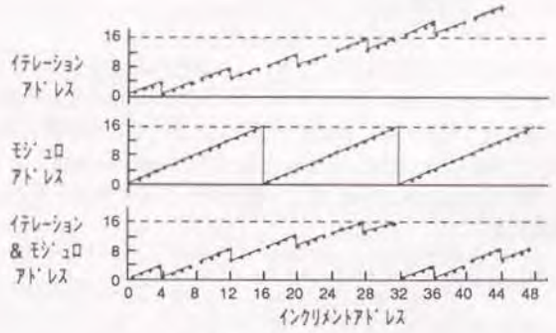
うにシフトする。すると2^Mを組として、除去された桁数の「2の冪乗回」（2^N）だけ、繰り返しアドレスが生成される。図4.2.1（a）では、M＝2，N＝1の場合を示している。

モジュロでも、繰り返し周期を2の冪乗に限定する。この制限を設けることにより、モジュロ周期に対応する上位ビットをマスクするだけで、容易にモジュロアドレスを生成できる。また、このイテレーション回路とモジュロ回路を縦積みとすることにより、「イテレーション＆モジュロアドレス」の組み合わせアドレスも、容易に生成できる。図4.2.1（b）に、M＝2，N＝1、そしてモジュロ周期＝（2^4＝16）のときの、具体的なアドレスの生成例を示した。

以上の方針と原理に基づいて、具体的に設計したアドレス生成ユニットの回路を図4.2.2に示す。3アドレス演算まで可能とするため、3個のアドレスポインタAR0，AR1，AR2を設けている。そしてその内、AR0のみに高機能アドレスを生成させ、残り



（a）生成モデル



（b）生成アドレス

図4.2.1　イテレーション＆モジュロアドレスの生成法

の、AR1とAR2には、インクリメントアドレスのみを生成させる。

AR1とAR2のベースアドレスとインクリメント値は、それぞれBA1，BA2、およびI1，I2である。DBR1は、インクリメント用の加算器ADD2を共通利用して回路規模を削減するために設けたレジスタである。AR1とAR2のアドレス生成回路の最長遅延が、AR0に比べて圧倒的に短いことを利用して、それぞれのアドレスを半サイクルに分けて、時分割で生成している。

AR0では、高機能アドレスを生成するため、インクリメント用の加算器ADD0と、ベースのオフセットを与える加算器ADD1を分離している。BA0、I0は、それぞれ、ベースアドレスとインクリメント値である。DAR0は、インクリメント値を更新していくためのテンポラリレジスタであり、加算器ADD0へ帰還させるアキュムレータの役割と、後段のイテレーション等の複雑なアドレスを生成するための、原アドレスをセットする役割をしている。AIRは、イテレーションのグループの大きさと繰り返し回数を指定するレジスタであり、MCRはモジュロのサイズを指定するレジスタである。ACRは、ビットリバースアドレスを、シフトと加算とモジュロとで生成する手法に基づく[3][4][5]、ビットリバース用のシフトを与えるレジスタである。

本アドレス生成ユニットの機能を、その他のDSPと比較して、表4.2.3に示す。表中のDSSP1（Digital Speech Signal Processor 1）で示したのが、本アドレスユニット搭載のDSPである。DSSP1開発に先行したDSPは、D7720とMB8764のみであり、当時、格段に高機能なアドレス生成機能を有していたことを示している。またDSSPg a、その後のDSPのアドレス生成機能の流れを作ったことがわかる。

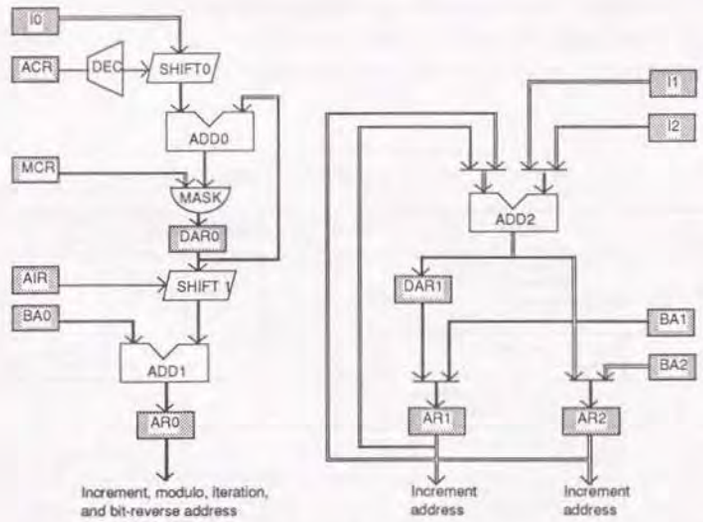


図4.2.2　高機能アドレスユニット（AAU）の構成

4.3　アドレス生成手法の応用例

２章では、ＤＳＰアーキテクチャの発展は、高速化の歴史であることを述べた。そして、時間方向の高速化であるパイプライン技術を支えるのが、データメモリのアドレス生成技術であることを述べた。本節では、4.2節で述べた技術によるアドレス生成回路を搭載し、複雑な音声符号化の実時間処理を実現して、第２世代ＤＳＰの先駆的な役割を果たしたＤＳＳＰ１を取り上げ、構成技術、インプリメント技術、およびその応用技術について述べる。

4.3.1　音声信号処理ＤＳＰへの要求条件の分析[6]

ここでは、アーキテクチャへの要求条件と、基本設計方針について述べる。

（1）要求条件

音声符号化アルゴリズムの処理およびＣＯＤＥＣの装置化の観点から音声信号処理ＤＳＰに対する要求条件を整理すると、表4.3.1（a）を得る。

（1-1）処理能力

ＤＳＰに要求される演算処理能力は、対象とする符号化アルゴリズムの複雑さやＤＳＰの機能分担量などにより決められる。例えば、ＡＰＣ-ＡＢのような高度なアルゴリズムを対象とすると、演算サイクル100ns程度の高い処理能力が要求される。

また音声符号化処理には、ＤＣＴ、ＤＦＴあるいはＦＦＴなどのスペクトル演算，窓関数，相関関数演算など、配列データを扱う専用処理が頻繁に現われるので、これらを専用的

表4.2.3　データメモリアドレス機能の比較

実行アドレス	内　容	D7720	MB 8764	DSSP1	MN 1901	DSP 56000	D77230
(R) @ (オペランド)	レジスタRの内容と命令のオペランド間で演算を施し，その結果をアドレスとする。例：EXOR	○ 部分ワード	○ フルワード				
(R) ±n	レジスタRの内容に±nを加算する。	○ ±1	○	○	○ -3~+3	○ ±1	○ ±1, +2ⁿ
(R) + (R')	レジスタRとR'の内容を加えたものをアドレスとする。			○	○	○	○
モジュロ	最大，最小値間のアドレスを繰返す。	△ 下位4b		○ 2ⁿ	○ 任意	○ 任意	○ 2ⁿ
ビットリバース	LSBとMSBをひっくり返してアドレスのビット順を逆にする			○ ワード処理		○ ワード処理	○
インテーション	ひとまとまりのアドレスを組にして一定回数繰り返す。			○			

に処理するハードウェアを付加して、処理能力の向上を図らねばならない。

さらに、１つのＤＳＰでは処理能力不足となるような符号化アルゴリズムに対処するため、簡単にマルチチップ構成が可能となるアーキテクチャを備えておかねばならない。

（1-2）演算精度，ダイナミックレンジ

音声符号化方式の品質は、量子化雑音のようにアルゴリズム自体から決まる他に、ＤＳＰの数値データ形式，すなわち内部演算における固定小数点、あるいは浮動小数点の語長、小数点位置などにより決められる。対象とする音声符号化アルゴリズムの品質劣化を最小限に抑える範囲内で、これらのデータ形式を明らかにしなければならない。

（1-3）メモリ構成

フレームを扱う音声符号化（数十／数百のサンプルデータをベクトルとしてまとめて扱う符号化）アルゴリズムでは、相関演算、窓関数演算のような配列演算が多い。これを効率的に処理するためには、各種フレーム情報を配列として、これを逐次演算部に供給するようなメモリ構成が必要である。また、Ｘｉ＊Ｙｉ⇒Ｚｉのような２項演算に適した構成にする必要がある。更に、フレーム内のサンプル数が多いアルゴリズムに対しては、データメモリをチップ外部に拡張できる構成にしなければならない。

（1-4）プログラミング

アルゴリズムをプログラミングする観点では、リアルタイム処理を維持する範囲内で、記述が容易で、かつ、ロードモジュールをコンパクトに抑えられることが必要である。

表4.3.1　ＤＳＰへの要求条件とアーキテクチャ

（a）要求条件

	項　目	要　求　条　件
アルゴリズム	処理能力	・専用処理のハード化 ・高速化（双方向通信） ・マルチプロセッサ化
	精度、ダイナミックレンジ	・音声符号化品質の劣化僅少（内部演算，インタフェース）
	プログラミング	・プログラム量の縮小 ・作成の容易さ
	データメモリ	・フレーム処理の効率化 ・外部拡張性
論理化	入出力	・シリアルインタフェース
	同期	・ビット，ワード，フレーム同期
	システム制御	・初期設定，動作モード変更

（b）アーキテクチャ

A．プログラムアーキテクチャ	B．システムアーキテクチャ	C．ハードウエアアーキテクチャ
アドレス方式	マルチ接続機能	パイプライン化
データ形式		
命令種別		マイクロ制御
レジスタ方式		
		専用入出力
	同期処理	
	割り込み	

（1-5）入出力インタフェース

ＣＯＤＥＣとしての装置化を前提とすると、シリアル信号インタフェースの他に、ビット，ワード，フレーム同期用のインタフェースが不可欠である。さらに、装置の初期設定や動作モードの変更などのシステム制御用として、外部割り込み機能が必要である。

（2）　アーキテクチャ設計の方針

前述の要求条件に答えるＤＳＰの基本アーキテクチャを、表4.3.4（b）のように導出した。

（2-1）プログラムアーキテクチャ

（2-1-1）アドレス方式

２項演算を主体とする配列演算のパイプラインを高速化するため、高機能なアドレス生成機能を持った複数のアドレスポインタを設ける。

（2-1-2）データ形式

数値データ形式により、演算の精度あるいはプログラミングの容易性が規定される。一般に従来のＤＳＰでは、固定小数点演算を用いるものが多く、精度の要求される線形分析やパワー計算などについては、倍精度演算で処理してきた。しかし、各種の音声符号化アルゴリズムを対象とするとき、以下の問題が顕在化してきた。すなわち

・アルゴリズムの高度化に伴って、処理のダイナミックレンジが広がり、より長い固定小数点の語長が必要となってきた。

・演算精度を確保するためには、演算毎にデータの小数点位置を移動するスケーリング処理が必要である。このため、処理が複雑になるに従い、プログラミングの最適化に大きな負担がかってきた。

以上を解決するため、浮動小数点演算方式を導入する。

（2-1-3）浮動小数点語長の評価

一般に、ディジタルフィルタのように比較的簡単なディジタル信号処理の場合には、演算の精度とダイナミックレンジの関係は、ある程度理論的に予測できる。しかし、高能率音声符号化ＣＯＤＥＣのような時変型のディジタル信号処理の場合には、シミュレーションを行う以外に有効な手段がない。そこで、オペレーション毎（演算毎）に処理語長を自由に指

定できる、有限語長シミュレータを開発して評価した。解析対象として、代表的な高能率音声符号化アルゴリズムである、16kb/s で3.4kHz帯域のＡＰＣ-ＡＢ方式を選び、特定話者３秒間のシミュレーションにより浮動小数点の語長に対する符号化品質特性を評価した。

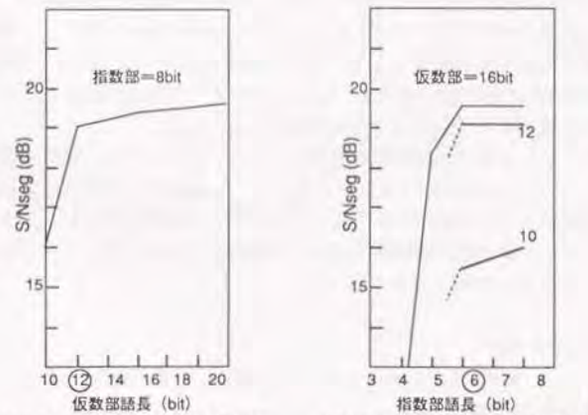
図4.3.1（a）は、指数部ビットを一定とした時の、仮数部ビット数とセグメンタルＳ／Ｎ（S/Nseg）との関係を示したものである。仮数部が１２ビット以上では、品質にほとんど変化がなく、１２ビット未満で急激に劣化していることがわかる。また図4.3.1（b）は、仮数部ビットを一定とした時の、指数部ビット数とS/Nsegとの関係を示したものであり、この場合には、指数部６ビット以上では、品質に全く変化がなく、６ビット未満で急速に劣化することがわかる。以上から、品質劣化がほとんど認められない範囲での最小の語長として、「仮数部１２ビット、指数部６ビット」の１８ビット浮動小数点方式（１２Ｅ６）が、最適であるとが確認できた。

（2-1-4）命令種別

ＤＳＰでは演算が主体であるので、この演算命令を最適化した水平型のマイクロ命令セットを設ける。命令種別としては、この他に、ジャンプやサブルーチンおよびループを制御するシーケンス制御命令や、レジスタへのデータ設定を行うデータロード命令を設ける。

（2-1-5）レジスタ方式

ＲＡＭにより大容量のレジスタを構成し、パイプライン演算器に直結する。２項演算の



（a）仮数部語長による品質劣化特性　　（b）指数部語長による品質劣化特性

図4.3.1　有限語長シミュレーションによる
最適な浮動小数点語長の策定

データを同一演算サイクル内で同時に取り込めるように、２面ＲＡＭまたはデュアルポートＲＡＭ構成とする。またアプリケーションに応じて、レジスタ容量を等価的に拡張できるようにする。

（2-2）システムアーキテクチャ

（2-2-1）マルチチップ拡張機能

１チップのＤＳＰ処理能力を上回るアルゴリズムに対して、マルチ接続機能を設ける。音声符号化アルゴリズムは、比較的容易に機能分割できる構造を持つので、ここでは、ハンドシェーク同期により、簡単にマルチプロセッサに拡張可能な機能を搭載する。

（2-2-2）割り込み

一般のマイクロプロセッサと異なり、音声符号化ＤＳＰでは多重割り込みの必要はないので、初期設定あるいは、ＤＳＰの動作モード設定用の外部割り込み機能のみとする。

（2-3）ハードウエアアーキテクチャ

詳細は次節以降で説明する。ここでは、必要なデータメモリとプログラムメモリ容量の算定法を述べる。図4.3.2は、いくつかの音声符号化アルゴリズムについて、データメモリのメモリマップを作成し、その結果から最小限必要な容量を計算したものである。ＡＤＰＣＭ、ＳＢ－ＡＤＰＣＭ（Sub-Band ADPCM）、音声認識のためのパラメータ分析等、サンプルデータ単位に処理するアルゴリズムでは、５００Ｗ程度あれば良いことがわかる。一方、８ＫＨｚサンプルでフレーム長１６ｍｓ（128 points/frame）のフレーム処理のアルゴリズムでは、最低２ｋＷ程度必要である。以上の要求条件を、ＬＳＩ化を検討している1.2μmルールの制限内で両立させるため、５００Ｗのデータ ＲＡＭをオンチップ化し、外部に４ｋＷまでの拡張メモリを接続できる構成とした。

またプログラムメモリについての結果を、表4.3.2に示す。音声符号化では比較的処理が複雑な、ＡＰＣ-ＡＢとＳＢ-ＡＤＰＣＭについて調べた。いずれも２ＫＷ弱で収まっており、この程度ならばＲＯＭ化することで、オンチップ化可能である。ここでは、高速化のためにプログラムを最適化できる余裕と、より複雑なアルゴリズムへも対処できる余裕を持たせるため、容量を４ＫＷと定めた。

4.3.2　アーキテクチャ

（1）　高並列処理[7][8]

前節での方針に基づいて仕様を決定した音声信号プロセッサＶＬＳＩ（ＤＳＳＰ1）の機能ブロック構成を図4.3.3に示す。各種の信号処理アルゴリズムに効率良く適用できる

ように、３２ビットの水平マイクロ命令を使って並列処理し、高速化を図った。アドレス演算ユニット（ＡＡＵ）、高速パイプライン演算器、パラレルデータバスインタフェース、およびＤＭＡコントローラの搭載により、（Ａ）高機能アドレスの演算，（Ｂ）浮動小数点乗算，（Ｃ）浮動小数点ＡＬＵ演算，（Ｄ）オンチップＲＡＭと拡張ＲＡＭ間のデータ転送，（Ｅ）２チャネルのシリアルデータ入出力の５つを並列に処理できる。これにより、ＦＦＴやＤＦＴのような複素ベクトル演算も効率良く実行できる。

（2）　パイプライン構成

ベクトル演算処理の高速化を図るため、プログラムバス（ＰＢＵＳ）とデータベース（ＤＢＵＳ）からなる２つのバスを用いた高速シーケンス制御システムを採用し、ＤＢＵＳを介さないデータメモリからの専用の読出しバスを設けた。これにより、命令フェッチ，命令デコードとアドレス生成，データメモリ読出し，乗算，ＡＬＵ演算，およびデータメモリ書込みから成る６段のパイプライン構成を実現した。

このパイプラインシーケンスのタイミングを図4.3.4に示す。命令フェッチの次のサイクルで、命令のデコードと同時に、データメモリの読出しおよび書込みで使用するアドレスをＡＡＵから生成する。さらに次のサイクルで、命令をデコードし、データメモリの読出し，書込み，浮動小数点乗算器（ＦＭＰＬ），浮動小数点ＡＬＵ（ＦＡＬＵ）の演算を制御

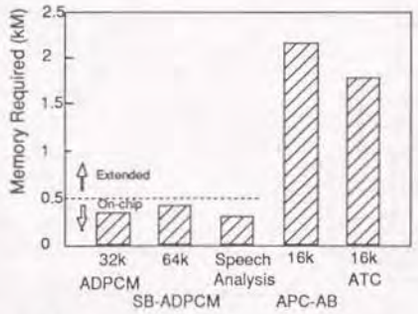


図4.3.2　必要なデータメモリ容量の分析

表4.3.2　必要なプログラムメモリ容量の分析

CODEC	Algorithm	Memory Required
16 kb/s	APC-AB	1.8 kW
64 kb/s	SB-ADPCM	1.6 kW

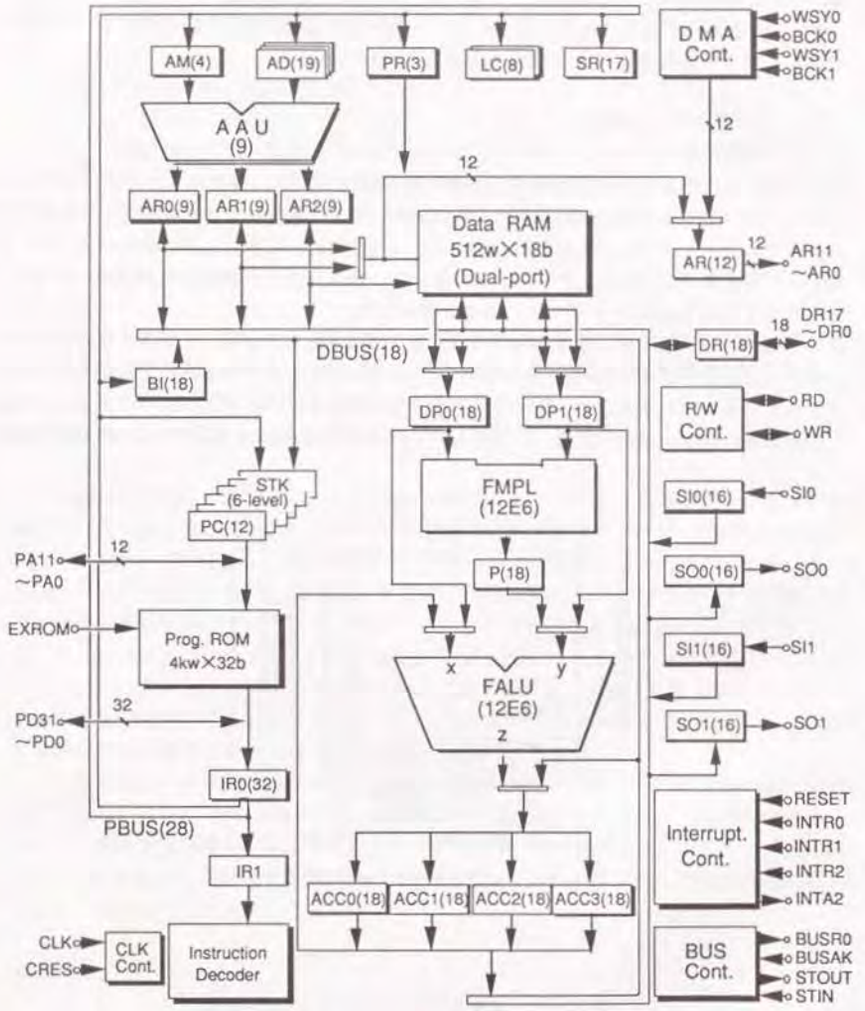


図４.３.３　高速音声信号プロセッサVLSI(DSSP1)の機能ブロック図

する。また、データメモリから読出されたデータは、乗算，ＡＬＵ演算を経て、４サイクル後にデータメモリに書込まれる。このように、命令のデコードとアドレス生成を並列化したパイプライン構成をとっているので、積和演算や複素演算などのベクトル処理を、１クロックサイクルに２回実行できる（20MHz のとき 40MFLOPS）。

（３）　オンチップマイクロプログラムＲＯＭ

高能率ＣＯＤＥＣのようなフレーム処理の符号化では、約２ＫＷのプログラムメモリ容量を必要とし、ベクトル量子化や音声認識ではさらに大きなメモリ容量が必要である。またマシンサイクルを向上させるためには、高速なプログラムメモリのアクセスが要求される。このため、ＤＳＳＰ１では４ＫＷの大容量マイクロプログラムＲＯＭを 2.95mm×1.43mm の面積にオンチップ化した。これにより、各種の信号処理装置に広く適用して、低電力で経済的なシステムを構成できる。なお、プログラム開発時は、プログラムを自由に書き換えねばならないので、プログラムメモリを外部ＲＡＭに切換られる機能を設けた。

４.３.３　ＶＬＳＩインプリメント技術

（１）　正規化浮動小数点ＡＬＵ

浮動小数点ＡＬＵは１０種の正規化浮動小数点演算モードと、１１種のシフト演算，論理演算，固定小数点演算のモードを持つ。浮動小数点ＡＬＵの演算モードとその適用例を表４.３.３に示す。浮動小数点演算では、通常の信号処理プロセッサの基本的なモードのほか

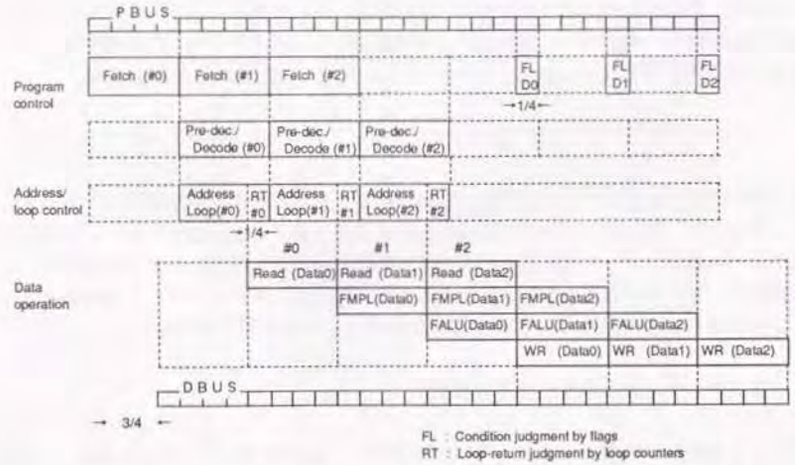


図４.３.４　パイプラインシーケンサのタイミング

に、音声符号化に便利な、絶対値，符号相関，最大値検出，最小値検出の各モードを設け、予測符号化やＬＳＰ（Line Spectrum Pair）分析などに対する処理能力を大幅に向上させた。また固定小数点では、１２ビットの仮数部加算と、６ビットの指数部減算モードを設けた。前者は、ＡＡＵで生成できない複雑なアドレス計算が必要になったとき、ＡＬＵで加算とシフトを繰り返し使用して生成するため、後者は、除算のときに指数部の減算処理が必要となるためである。浮動小数点加算，減算を始めとして、すべてのＡＬＵ演算を１マシンサイクルで実行できるので、高速を損なわずに、プログラム開発の容易性を実現している

浮動小数点ＡＬＵのブロック構成を図４.３.５に示す。浮動小数点ＡＬＵの設計では、本研究で考案した２つの高速化手法（３.２.３節参照）を採用して、高速な正規化浮動小数点加算と減算を実現した。その手法の１つは、ブロックレベルでの並列処理技術であり、指数部減算（SUB1）と仮数部シフト（MSFT）の並列処理、および正規化シフト数の検出（ENC）と正規化シフト（NSFT）の並列処理である。もう１つは、回路技術であり、キャリーセレクトアダーを使った加減算回路（SUB1,ADD/SUB）と高速正規化シフト数検出回路（ENC）である。

以上の結果、浮動小数点加算，減算をはじめとする、21種のすべての浮動小数点ＡＬＵ演算を、マシンサイクル50nsec内（１.２μｍＣＭＯＳプロセス，電源変動等を含んだ保証値）で実行可能とした。

（２）　高機能Ｉ／Ｏインタフェ ース

ＰＣＭ入力を符号化してディジタル回線に出力する符号器用に１チャンネル、その逆の復合器用に１チャンネルの、計２チャンネルのシリアルポートを搭載して、ＣＯＤＥＣの双方向処理を１チップで実現できるようにした。また、シリアルポートと、データ処理のワークエリヤとなるＲＡＭ間（チップ外部の拡張ＲＡＭ間も含む）のデータ転送は、ＤＭＡコントローラを使った割り込み処理で、プログラマが意識しないで処理できるようにした。割り込み処理の高速化のため、ハードウェアによる徹底したサポートを行い、全処理時間を５マシンサイクル以内に抑えた。これは、２チャンネルすべての転送時間を加えても、

５（マシンサイクル）×２（I/O）×２（チャンネル）×50ns＝1μs

となり、音声処理での８Ｋサンプル／秒（125μs）を想定したとき、処理時間の１％以下になることを意味し、転送オーバヘッドをほとんど無視できることを示している。この機能により、フレーム単位のデータ列をワークメモリに格納してから処理するのが容易になり、フレーム処理アルゴリズムを効率よく実行できるようになった。また、システム制御用とマルチチップ動作用に、４レベルの割り込み機能と外部バス管理機能を搭載した。

（３）　ＤＡシステムによるＶＬＳＩの設計

ＶＬＳＩの設計は機能シミュレータ（FOREST），論理シミュレータ（ELSA），自動レイアウトプログラ（CHAMP/ALPHA）[9][10]から成る階層的ＤＡシステムを有効に用い

表４.３.３　浮動小数点ＡＬＵの演算モード

演算モード		演算内容	適用例
浮動小数点	符号反転	-x→z	
	絶対値	|x|→z	パワー検出
	スルー	y→z	
	符号相関	SIGN(y)·|x|→z	予測符号化
	加算	x+y→z	
	減算	x-y→z	
	最大値	MAX(x,y)→z	ピッチ抽出
	最小値	MIN(x,y)→z	零点検出
データ変換	固定→浮動	FLT(x)→z	音声入出力
	浮動→固定	FIX(x)→z	音声入出力
シフト	右1、左1～8	R1,L1～L8	
論理	AND,OR,EOR,NOT		
固定小数点	仮数加算	$x_M+y_M \rightarrow z_M$	アドレス計算
	指数減算	$x_E-y_E \rightarrow z_E$	除算

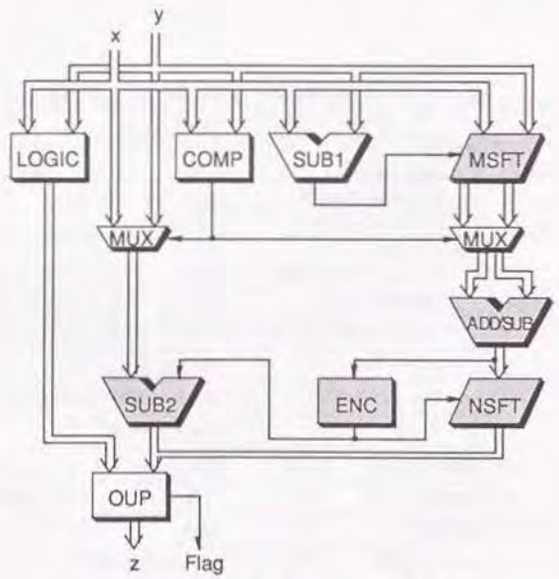


図４.３.５　浮動小数点ＡＬＵのブロック構成

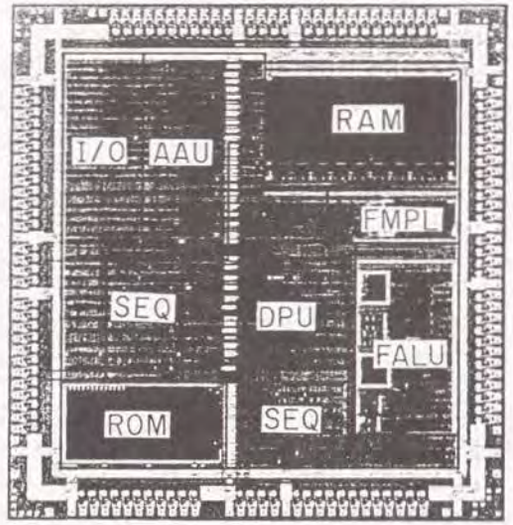


図4.3.6　ＤＳＳＰ１のチップ写真

表4.3.4　汎用音声信号プロセッサVLSI（DSSP1）の諸元

演算器	
浮動小数点ＡＬＵ	18ビット（12E6）,16ビット
浮動小数点乗算器	18ビット（12E6）
アドレスユニット	12ビット
データＲＡＭ	
デュアルポート内蔵	512ワード ×18ビット
外部拡張	4Kワード ×18ビット
マイクロプログラムＲＯＭ	4Kワード ×32ビット
マシンサイクル	50nsec（typ.）
シリアルＩ／Ｏポート	2チャンネル（DMA機能付）
パッケージ	132ピンPGA
電源	5V
消費電力	700mW（50nsec動作時）
素子数	280k
デバイス技術	1.2μmCMOS
チップサイズ	9.2mm×9.6mm

て行った。浮動小数点ＡＬＵのレイアウトでは、まず最長遅延パスを構成する５つのブロック（図4.3.5の網線で示す）を手描きで高密度に作製し、それら５つのマクロと、残りのＳＣ（スタンダードセル）ゲートを一括レイアウトすることで、高速性と設計工数短縮とを両立させた。浮動小数点ＡＬＵ全体で2.3Ｋゲートであり、これを1.88mm×3.09mmの面積（1.2μmＣＭＯＳ技術）に集積化した。チップ全体のレイアウトでも、浮動小数点ＡＬＵ，５１２wのデュアルポートＲＡＭ，４KWのマイクロプログラムＲＯＭ，および乗算器の４つを予め作製したマクロセルとして、残りをＳＣで一括レイアウトした。ＬＳＩ全体で２８０Ｋ素子であり、これを、9.2mm×9.6mmのチップに集積化した。この階層レイアウト手法により、設計期間の大幅短縮と、高速・高密度性を両立できた。開発した音声信号プロセッサＶＬＳＩ（ＤＳＳＰ１）のチップ写真と諸元を、それぞれ、図４３.６と表4.3.4に示す。

4.3.4　命令セットと性能[11]

マイクロ命令は、３２ビットの水平型とした。その構成を図4.3.7に示す。命令は、オペコードで、シーケンス命令（OP1），モード命令（OP2），演算命令（OP3），およびロード命令（OP4）の４種に分類した。シーケンス命令では、ループ，ジャンプ，サブルーチンコール等の各命令種別をＳＱＭフィールドで指定する。最大の特長は、"Loop/Address re-setup" フィールドを使って行う、ループ回数とアドレス初期値の自動更新機能であり、詳細は図4.3.8にて後述する。モード命令では、ループカウンタ，ＡＡＵのアドレスモー

OP1	SQM	Loop/Address re-setup	Branch	Jump address	SRC	—
OP2	—		Loop counter/Address mode			DST
OP3	REP	Loop/Address renew	Multiplexer control	FALU	SRC	DST
OP4	—	Address renew	Immediate data			DST

OP1 : Sequence control　　SQM : Sequence mode
OP2 : Mode setting　　SRC : Source register
OP3 : Data operation　　DST : Destination register
OP4 : Literal load　　REP : Repeat

図4.3.7　マイクロ命令セット

ド，ベースアドレスなどを初期設定する。演算命令では、モード命令で初期設定したループカウンタやアドレスポインタを簡単な指定で自動更新しながら、必要なデータをデータメモリから取り出して各種演算を行う。また、データメモリやレジスタ間のデータ転送を行う。この演算命令での特長は、予め繰り返し回数を指定することで、１つの命令を全くオーバヘッドなしに繰り返し実行できるリピート機能を持たせた（ＲＥＰフィールド）ことである。ロード命令での特長は、データを、チップ外部のメモリへも直接ロードできるようにし、外部メモリを内部メモリの完全な拡張して扱えるようにした点である。

マイクロ命令を用いたプログラム例を図４．３．８に示す。Ｎワードのベクトルデータから（M-1）次の自己相関係数を求める例である。外側のループの繰り返し回数は、M回で一定であるが、内側のループの繰り返し回数は、Ｎ回から（N-M+1）回までしだいに減少していく。さらに、データＢ(I+J) のベースアドレスは、外側のループの繰り返しにつれて１づつ増加していく。このループの繰り返し回数とデータＢ(I+J) のベースアドレスは、先に述べたシーケンス命令の"Loop/Address re-setup"フィールドを使って自動更新できる。また、積和演算は、演算命令のリピート機能を使って、１マシンサイクルで実行できる。

上記マイクロ命令を使ったプログラムの実行例を表４．３．５に示す。複素数演算や多変数演算等を含むベクトル処理が、浮動小数点演算の広いダイナミックレンジで高速に実行で

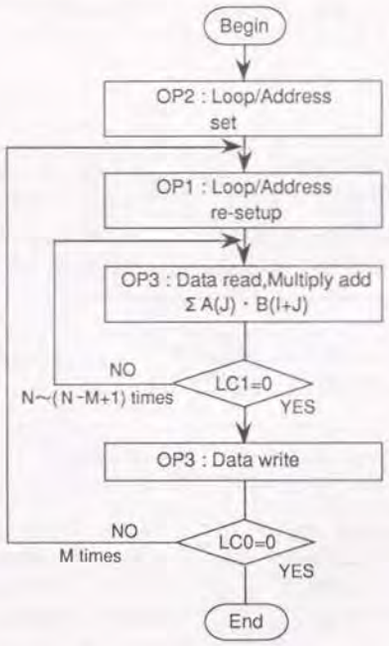


図４．３．８　自己相関関数のフローチャート

きる。また浮動小数点演算に加えて、ループ回数とアドレス初期値の自動更新機能の効果が顕著である。例として、128ワードの複素ベクトル加算を挙げると、実行ステップ数は1294ステップであり、これをアドレス初期値の自動更新機能がない場合と比較すると、ステップ数で37%削減されている。また、128ワードデータを使った、10次の自己相関係数とFIRフィルタの場合には、ループ回数とアドレス初期値の両方の自動更新機能が使えるので、ステップ数で40～60%の削減が達成され、それぞれ1347ステップと3927ステップで実行できる。この実行ステップ数を、演算サイクル50nsのときの実行時間に換算し、両方を表４．３．５に示した。これらにより、ＤＳＳＰ１のマイクロプログラムの高効率性が実証されている。

４．３．５　音声ＣＯＤＥＣへの応用

ＤＳＳＰ１の音声ＣＯＤＥＣへの応用例を図４．３．９に示す。２例共、１．２μｍプロセスではなくて、低価格版として２．５μｍプロセスを使って開発したＬＳＩを、使ったものである。演算サイクルを150nsと、１．２μｍ版の１／３に低下させると共に、集積度の制約から、プログラムメモリを外付けとする構成になっている。

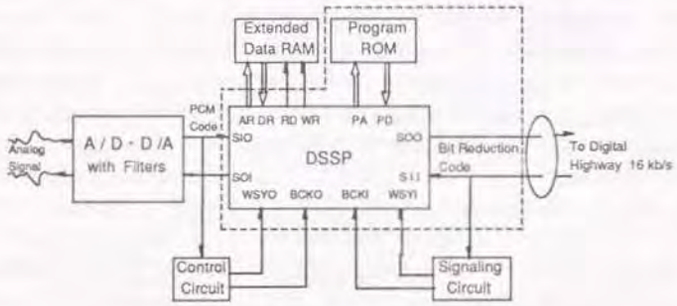
（ａ）は、１６ｋｂ／ｓのＡＰＣ-ＡＢのＣＯＤＥＣである。音声符号化を効率良く処理するベクトル命令と、それをサポートする浮動小数点ＡＬＵ、および高機能アドレスユニットにより、２．５μｍプロセスにも関わらず、音声の符号化・複合化部の処理を１チップで実現している。必要な周辺回路は、プログラムメモリ（32b×4kW），外部拡張データメモリ（18b×4kW），アナログインタフェースとしてのAD/DAと周波数軸上での折り返し防止用のローパスフィルタのみである。ＤＳＳＰ１に２チャンネルのシリアルポートを内蔵させているので、アナログインタフェースとディジタル回線インタフェースへの付加素子を大幅に減らすことができた。ちなみに、ＤＳＳＰ１の性能を２～３μｍ技術による汎用μＰと比較してみよう。代表的なものとして、モトローラのM68000が挙げられるが、当時、

表４．３．５　信号処理のプログラム実行例

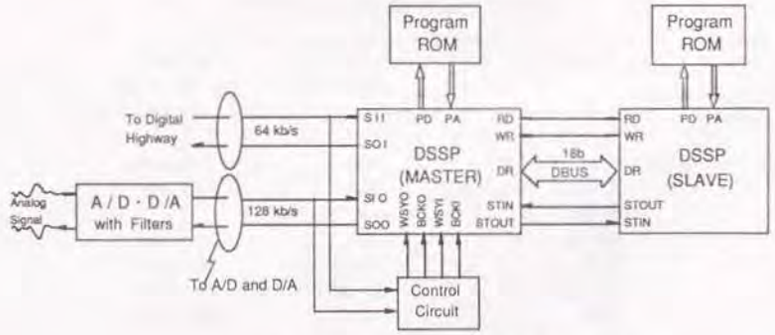
処理内容	ステップ数 (Nワード、M次)	実行時間 (N=128, M=10)
データ転送		
ＲＡＭ-ＲＡＭ間	3+N	6.6μsec
ＲＡＭ-Ｉ／Ｏポート間	4+N	6.6μsec
算術演算		
絶対値	5+N	6.7μsec
加算	7+5N	32.4μsec
複素加算	14+10N	64.7μsec
除算	40N	256.0μsec
応用		
自己相関係数	7+M(6+N)	67.4μsec
ＦＩＲフィルタリング	7+M(8+3N)	196.4μsec

１６ビットで８ＭＨｚ程度の動作クロックであった。これはＤＳＳＰ１と比較して、クロックサイクルはほぼ同等である。しかし、データメモリから２データを読みだしてＡＬＵ演算を行いメモリに格納するまで１０サイクル以上かかること、乗算や浮動小数点演算ではさらにその数倍の時間を要することから、本ＣＯＤＥＣに数十チップを必要とする。またさらに、シリアルポート等の付加回路も必要とするので、とても比較にならない。

（ｂ）は、サブバンドADPCM-CODEC への応用である。7kHz 帯域の高品質音声を16kHz でサンプリングして取り込んだ128kb/s の原データを 64kb/s に圧縮する。この場合は１個のＤＳＳＰ１では処理能力が不足するので、２チップによるマルチチップＤＳＰとした。この、２個のプロセッサを対向させて動作させるマルチチップ接続機能は、一方をマスター、他方をスレーブとし、スレーブ側のリクエストに対して、マスター側がアクノリッジを返すことで、プロセッサ間での直接データ転送を行う機能である。これは外部のバス調停回路なしにマルチプロセッサが構成できるので、小規模なシステムを経済的に構築できる。



(a) 16 kb/s 高能率 CODEC



(b) 64 kb/s 高品質 SB-ADPCM CODEC

図４.３.９　ＤＳＳＰ１の音声ＣＯＤＥＣへの応用例

４.３.６　地下埋設探知装置への応用

（１）　地下埋設物探知装置の信号処理技術

（1-1）パルスレーダ法の原理

図４.３.１０にパルスレーダ法の原理を示す。微小時間幅（約2ns）のインパルスを送信アンテナに給電し、電磁波を発生させる。発生した電磁波は地中を伝搬し、誘電率等電気定数の異なる物体に当たると反射する。この反射した信号は再び受信アンテナでとらえられる。この往復に必要な伝搬遅延時間から埋設位置を計算する方法である。このとき、埋設物の深度L(m)は近似的に次式により求められる。

$$L(m)=V(m/s)\ T(S)/2$$
$$V(m/s)=C(m/s)\ \sqrt{\varepsilon_S} \qquad (4.3.1)$$

ここに、Ｔは入射波と埋設物からの反射波との時間間隔，Ｖは地中での電磁波伝搬速度，Ｃは空気中での電磁波伝搬速度，ε_sは土の比誘電率である。実際の測定は、図４.３.１０の矢印の方向にアンテナを移動させながら（例えば2cm移動するごとに）観測信号を記録する。

（1-2）　埋設管反射派の特徴把握[12][13]

観測信号は、地中のあらゆる不連続点からの反射波が重畳したものであり、埋設管からの反射波も他の不要反射波と同様に、観測信号のある時間範囲に存在する。そこで、観測信号をある時間間隔ごとに分割すれば、そのいずれかの分割波は、埋設管からの反射波を含ん

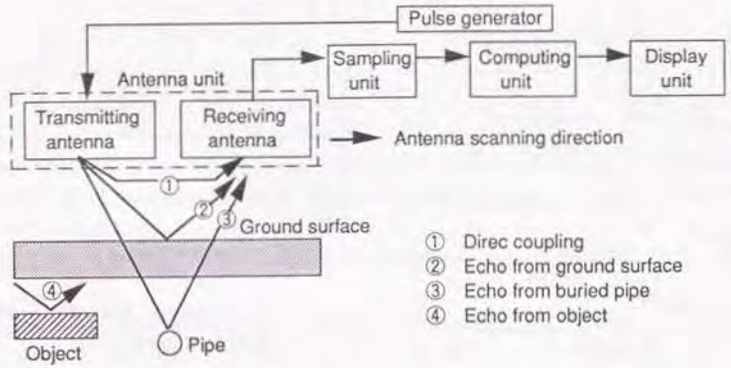


図４.３.１０　パルスレーダ法の原理

でいると考えられる。この分割波の有する特徴を把握すれば、埋設管からの反射波に相当する分割波を識別できる。実際にアンテナから送信される信号は、モノサイクル信号であることから、観測信号を１波長程度の時間幅に相当する一定の幅△Ｔで順次分割する。分割の模様を図４.３.１１に示す。観測信号のいずれかの時間位置に重量している埋設管からの反射波を確実に分割できるように、分割開始と次の分割開始の時間間隔は、分割長△Ｔより小さくしてある。

また一般的に、電磁波の高周波数成分は低周波数成分に比べ伝搬時間に対する減衰率が大きい。そのため、△Ｔも伝搬時間の大小により変化すると考えられる。従って本来なら、伝搬時間が大きいほど△Ｔを大きく設定するほうが高精度に探知できる。しかし、ここでは埋設管を探知するのが目的のため、探知目標の距離は1.5m以内である。このため、△Ｔの伝搬時間に対する変化は、0.2ns程度とあまり差がないので、実用的観点から△Ｔを一定とした。

従来、波形解析には時間領域の手法が多く採用されていた。しかし、波形のもつ周波数領域の特徴をも利用した方が、より一層多くの有効な情報を得ることができる。特に今回のように反射現象が１波長程度の波形が基本であれば、その単位での周波数領域の特徴を把握することが有効である。そこで、これらの分割波の評価は、分割波個々の周波数スペクトルの特徴を抽出・比較することにより行うこととした。まず各分割波に対するパワースペクトルを求める。図４.３.１２にこのスペクトル分布の例を示す。横軸は周波数(50MHz/div)、縦軸は強度である。このスペクトル分布を評価するパラメータは種々考えられるが、通常は以下に示す三つのパラメータにより反射派のスペクトル形状がほぼ決定される。

①中心周波数 fp
②半値幅 W
③直流成分比 Rdc(=Idc/Ip)

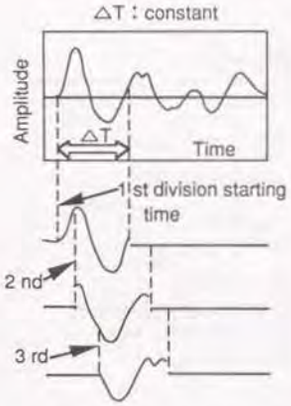
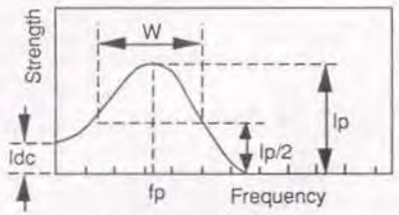


図４.３.１１　観測信号の分割方法



① Spectrum peak frequency　fp
② half width　W
③ DC component ratio　Rdc

$$Rdc = \frac{\text{DC component strength}}{\text{Spectrum peak strength}} = \frac{Idc}{Ip}$$

図４.３.１２　特徴パラメータ

ここで最も重要なことは、埋没物体からの反射波を三つのパラメータでどれだけ精度良く特徴づけることができるかを明確に把握することであるが、多くの実験を行い、ほとんどの場合、９０％以上の確度で十分に特徴づけられることを確認した。

（1-3）　パターン認識の処理フローおよび処理例

上記検討結果をもとに開発した、パターン認識技術の詳細なフローチャートを図４.３.１３に示す。詳細な説明は後に譲り、ここではその概略を述べる。

Step１：アンテナを移動しながら観測信号を178回受信し、地中断面パターンを形成する。以下、この１回の観測信号を１ラインデータと称する。

Step２：地中断面パターンを構成する１ラインデータを時間間隔で分割する．そしてこの分割波を周波数領域に変換し、スペクトラムを得る。

Step３：各分割波のスペクトラムから３つの特徴パラメータを算出する。その値が基準範囲内にあれば、その値を有する分割波を埋設管からの反射波と判断し、抽出する。

Step４：178個すべてのラインデータに対して上記処理を行い，断面パターンを再構成する．

（２）　パターン認識信号処理ボードの設計・開発

前節で述べたパターン認識処理の中で、各分割波の周波数領域への変換処理（Step２）と、周波数領域上での判定処理（Step３）とを高速化するための専用の信号処理ボードを開発した。性能目標として、178ラインデータ（各ラインデータは330ワード）からなる１断面の処理を、準リアルタイムである３０秒以内とした。

Step２は、ＦＦＴを核とする典型的な定形ベクタ演算であり、1024点複素ＦＦＴを2ms強で処理するスループットが必要である。この理由は、１ラインデータ（330ワード）処理をするとき、分割波のずらし時間を６サンプル毎とすると、Step２での１ラインデータ当りの、ＦＦＴと判定の繰返し回数が56回となり、従って、178ラインから成る１断面処理では、約１万回（56×178）の繰返しが必要となるからである。すなわち、３０秒間に１万回の繰り返しを行うことから、１ラインの処理時間は3msとなる。この3msの内、ＦＦＴに2ms、パワー計算に1msを割り当てた。

Step３は、特徴抽出，有効・無効判定等、分岐処理の多い比較的複雑なベクタ演算である。

以上により、Step２は専用のＦＦＴ-ＬＳＩ[14]で、Step３は、（４.３.１～４.３.４節）で述べた音声信号処理ＤＳＰで処理することとした。以下、Step２の処理ボードを、ＤＳＰ０、ステップ３の処理ボードを、ＤＳＰ１と呼ぶ。

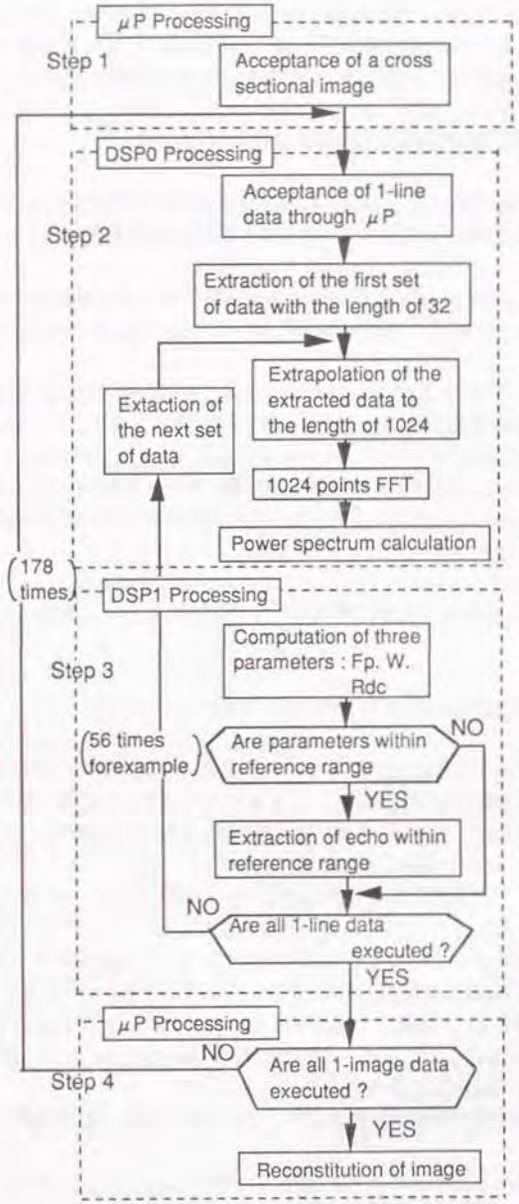


図4.3.13　パターン認識のフローチャート

ＤＳＰ０とＤＳＰ１は、システムバスを介してホストのμＰから制御される。制御の単位は、反射波の１周期（１ラインデータ＝330ワード）またはその集まりである地中断面パターン（178ラインデータ）のいずれでも可能であるが、今回は１周期単位とした。

埋設物探知装置の中に占めるＤＳＰ０ボードとＤＳＰ１ボードの位置づけを図4.3.14に示す。ＤＳＰ０とＤＳＰ１には、それぞれローカルＣＰＵを置き、システムバスを介して非同期に動作させると共に、バススレーブ機能のみとし、独立性を高くした。また、アンテナ系からの反射派データの受理と、ホスト系からの解析パラメータの受理およびホスト系への解析結果の送信とは、ＤＳＰ０およびＤＳＰ１からのバス解放要求に応じて、ホストのμＰが制御する。

ＤＳＰ０とＤＳＰ１は、１パワースペクトラムデータ単位にパイプライン処理される。ＤＳＰ０からＤＳＰ１へのパワースペクトラムデータの転送は、専用のチャンネルを介してＤＭＡ転送する。このとき、ＦＦＴ演算を行うＤＳＰ０側がスループットのボトルネックとなるので、ＤＳＰ１側をチャンネルマスタとした。

本信号処理部を搭載した地下埋設物探知装置全体の写真を図4.3.15に示す。また、この装置により処理された例を図4.3.16に示す。原画像（ａ）が、ＤＳＰ０とＤＳＰ１ボードにより処理され、３０秒後に、（ｂ）に示す結果を出力する。本処理によりノイズが完全に除去されていることがわかる。（ｃ）は（ｂ）の結果をＣＰＵで合成開口処理し、楕円上の反射形状を点図形に直して見やすくしたものである。以上により、音声信号処理ＤＳＰの高速性と汎用性が実証できた。

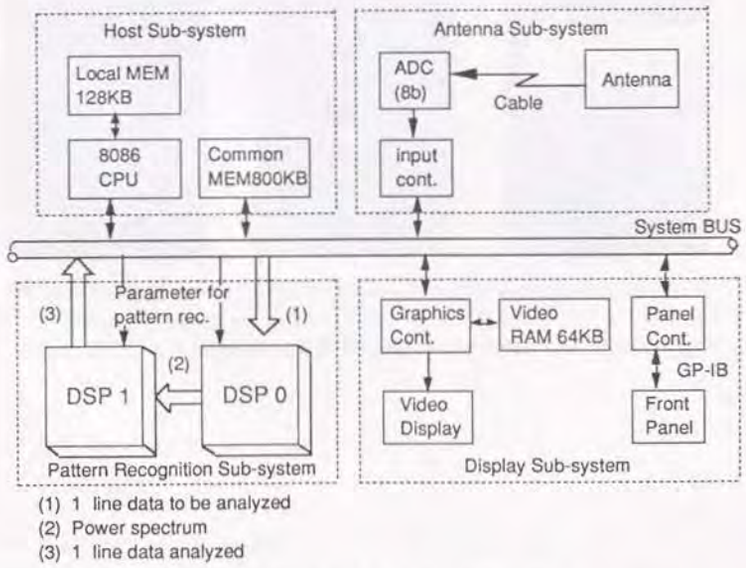


図4.3.14　信号処理部のブロック図

図４．３．１５　地下埋設物探知装置の全体写真

(a) Original Data　(b) Recognized Pattern　(c) Synthetic Aperture

図４．３．１６　地下埋設管検出の処理例

４．４　２次元浮動小数点フーリエ変換ＬＳＩのアドレス生成技術

前節では、汎用性の高い音声信号処理ＤＳＰのアドレス生成技術について検討した。ここでは、ＦＦＴという特有の処理に特化することにより、ＦＦＴ処理スピードが、音声信号処理ＤＳＰの数十倍にできる、ＦＦＴ専用ＬＳＩのアドレス生成技術についての成果を述べる。また次節では、本アドレス生成技術の成果を取り入れた、高速ＦＦＴ-ＬＳＩの構成技術について、その設計法と試作結果を述べる。

４．４．１　本ＬＳＩの目標

ＬＳＩ技術の急速な進展に伴い、ディジタル信号処理ＬＳＩ（ＤＳＰ）も急速な発展をとげている[15]。そして現在、ディジタルフィルタの高速処理を目的として発展してきたＤＳＰが、より複雑で、より高速な処理にも対応できつつある。ディジタル信号処理の基本技術である高速フーリエ変換（ＦＦＴ）もそのひとつである。この場合、ＦＦＴ特有の複雑なアドレス信号を高速に生成することが重要となる。そのため、従来のＤＳＰでは、専用のアドレス生成回路を内蔵し、ＦＦＴの高速化を図っている。例えば、３２ビット浮動小数点ＤＳＰでは、512ポイントＦＦＴを5.2msで[16]、また、３２ビット固定小数点ＤＳＰでは、1024ポイントＦＦＴを2ms[17]で実行する。この性能は、それぞれ、4MFLOPS，25MOPSの処理速度に相応する。こうした高速性能により、音声の実時間信号処理や静止画像処理にＦＦＴを用いることが現実的になってきている。

しかし、こうしたＤＳＰの演算部は、あくまでもディジタルフィルタ用の積和演算器であるため、ＦＦＴとしての性能向上には限度がある。このため動画像処理へＦＦＴを適用するには、充分な性能とはいえない。一方ＦＦＴ専用プロセッサーの開発も行われており、１６ビット固定小数点で、 200MOPS の処理性能が報告されている[18]。このプロセッサーは、基数４のバタフライ演算を基本としているため、基数２に比べて演算量が少なくて済むが、ＦＦＴのポイント数が４の巾乗に限定されるという欠点がある。また１６ビットの固定小数点のため、演算精度上もダイナミックレンジも充分とはいいがたい。

そこで、基数２のバタフライ演算を基本とする浮動小数点ＦＦＴプロセッサーが、必要になってきている。性能目標は、１チップで、毎秒３０フレームの動画像をリアルタイム処理できることとする。現在画像の符号化では、ＤＦＴ（Discreate Fourie Transform）よりむしろＤＣＴ（Discreate Cosine Transform）が使われる傾向にあるが、ＦＦＴ用ＬＳＩを利用してＤＣＴを計算することができるため、上記性能が達成できれば本ＬＳＩの適用領域は広い。

４．４．２　高速フーリエ変換

ＦＦＴのシグナルフローグラフを図４．４．１に示す。ＦＦＴのアルゴリズムは、各種存在し[19]、それらは、少しでも演算量を減らす方向で工夫がなされて来ている。この中で代表的な、Cooley-Tukeyアルゴリズム[20]を図４．４．１に示す。ＬＳＩ化する場合、単に演算量だけでなく、処理の規則性が高速化の大きな要因となる。この点から、Cooley-Tukeyアルゴ

リズムは、規則性と演算量でバランスがとれ、ＬＳＩ化に向いていると言える。一方、乗算回数が、オーダーＮ（O(N)）とさらに少ないWinogradのアルゴリズム[21]なども知られている。しかしこのアルゴリズムは、シグナルフローが複雑なため、アドレス生成と並列演算が共に容易でない。したがってＶＬＳＩに適したCooley-Tukeyのアルゴリズムを採用した。

4.4.3　アドレス生成ユニット

ＦＦＴプロセッサーでは、次の３種類のアドレス生成ユニットが必要である。

（A）外部データメモリのアドレス生成ユニット
（B）内部データＲＡＭのアドレス生成ユニット
（C）内部係数ＲＯＭのアドレス生成ユニット

（A）は、チップの外のフレームメモリへのアドレス生成を行うもので、１次元バタフライ演算に必要な１ライン分のデータをフレームメモリから取り込むときと、処理した１ライン分のＦＦＴデータをフレームメモリへの書き込むときに、使用する。この機能は、２次元ＦＦＴを、１ライン単位の１次元ＦＦＴのパイプライン処理に展開する上で重要である。節を分けて詳述する。

（B）は、１ライン単位の１次元ＦＦＴ演算を行うのに必要な２種類のアドレスを生成する。１つは、バタフライ演算に必要なバタフライアドレスであり、もう１つは、最終段でビットリバースのソーティングを行うのに必要なビットリバースアドレスである。これは音声信号処理ＬＳＩのところで触れたので省略する。（C）は、バタフライ演算で必要な係数を読みだすものであり、モジュロアドレスが基本である。これも音声信号処理ＬＳＩのところで触れたので省略する。

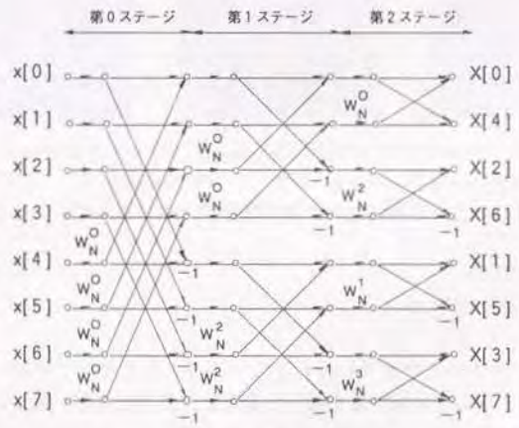


図4.4.1　8ポイントＦＦＴのシグナルフローグラフ

（1）外部データメモリのアドレス生成ユニット

２次元ＦＦＴの演算法として、行方向と列方向の１次元ＦＦＴをカスケードに実行する方法を採用した。この方法は直接２次元ＦＦＴを実行する方法に比べて演算量が少ない[22]ことと、１次元単位で処理できるため、オンチップメモリが少なくてすむ利点がある。しかしこの方法は、先に述べた様に、外部データメモリ（以下外部ＲＡＭ）とのデータ転送の効率化が重要であり、その要は、外部ＲＡＭをアクセスするアドレスユニットである。

外部ＲＡＭから１ライン単位でリード／ライトするときの、アクセスパターンとそのときのアドレス生成式を図4.4.2に示す。フレーム画面を想定した仮想的２次元座標を、１次元実メモリのアドレスに変換するのが主たる機能である。そして、行方向の１次元ＦＦＴを行うときには、行単位にアクセスでき、列方向の１次元ＦＦＴを行うときには、列単位にアクセスできなければならない。これを、フレーム画面のサイズをＮ×Ｎとして式で表すと、次の様に表現できる。

・行方向アクセスのとき：
（Ｌ＋１）番目の行のｎ番目のデータのアドレス＝Ｌ×Ｎ＋ｎ

・列方向アクセスのとき：
（Ｌ＋１）番目の列のｎ番目のデータのアドレス＝ｎ×Ｎ＋Ｌ

上式に基づいて、行方向と列方向の２つの座標変換を統一的にできるアドレス生成ユニットを考案した。これを図4.4.3に示す。行方向のときと列方向のときに対応して、Ｌ値およびｎ値を指定する２つのカウンタ出力のいずれかを選択してＮ倍し、その結果を、選択されなかった方の出力と加算する構成である。Ｎ倍はシフタで簡単に実現している。これはポイント数Ｎが２の巾乗に限られることによる。また加算器も単純なＯＲ回路で実現している。その理由は、加算のオペランドが、Ｎを整数倍した値と（Ｎ-１）以下の整数値とになるため、キャリー伝搬が生じないからである。以上により、高速で小規模に構成できた。

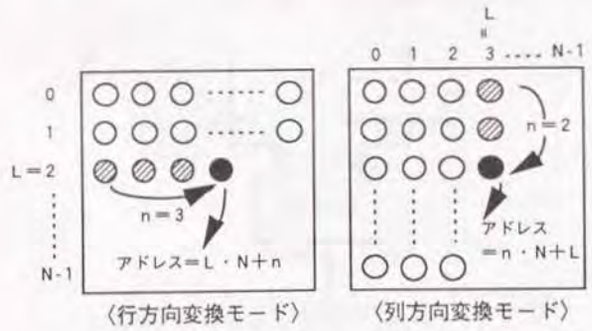


図4.4.2　外部ＲＡＭへのアクセスパターンとアドレス

4.5　2次元フーリエ変換ＬＳＩの構成法

4.5.1　ハードウェアアーキテクチャー

フーリエ変換ＬＳＩのアーキテクチャを考える場合、処理の単位をどの大きさにするかが最も重要である。例えばシストリックアレイで構成した場合、１ビット演算器の様な小さいサイズのアレイにすると、ＦＩＲフィルタのように、近接演算器からの出力データのみを利用して処理できる場合の高速化には有効であるが[23]、ＦＦＴのように、Global Communicationを必要とする演算には不向きである。ただしWARPマシン[24]のように、バタフライステージを基本ユニットとしてシストリックアレイ化すれば、複雑なコネクションを基本ユニットの中に入れることができるため、基本ユニット間でのパイプライン処理に帰着させることができる。このため、ステージ数単位で基本ユニットを増やすことにより、処理能力を比例して向上させることができる。例えば1024ポイントＦＦＴでは、ステージ数が１０であるから、１０倍の高速化が達成できる。すなわち、ＬＳＩ技術を活かして、チップ内部にできるだけ大きい処理ユニットを持ち込むほうが、有利といえる。そこで、0.8μmＣＭＯＳを想定した現実的な解として、複素バタフライを処理ユニットとした。

次に、複素バタフライ単位でパイプライン処理する上で、バタフライステージ間のGlobal Communicationを解決しなければならない。これには、２ステージ分のデータＲＡＭをオンチップ化し、専用のアドレス生成ユニットでバタフライ接続と等価なアドレスを生成して、メモリ内部でデータ交換行うことで解決した。さらにＦＦＴアルゴリズムの特徴を利用して、チップとチップ外部のフレームメモリ間のデータ転送を、効率良く行う方式を導入し、演算器の性能を最大限に引き出せるようにした。

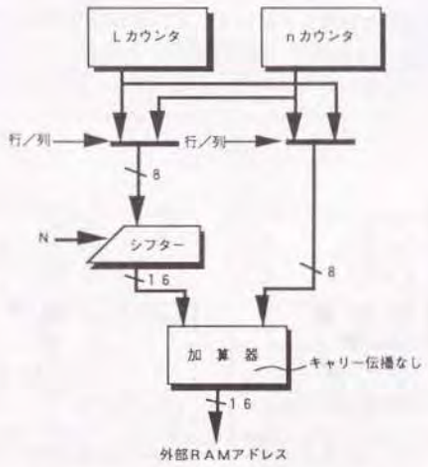


図4.4.3　外部ＲＡＭのアドレス生成ユニット

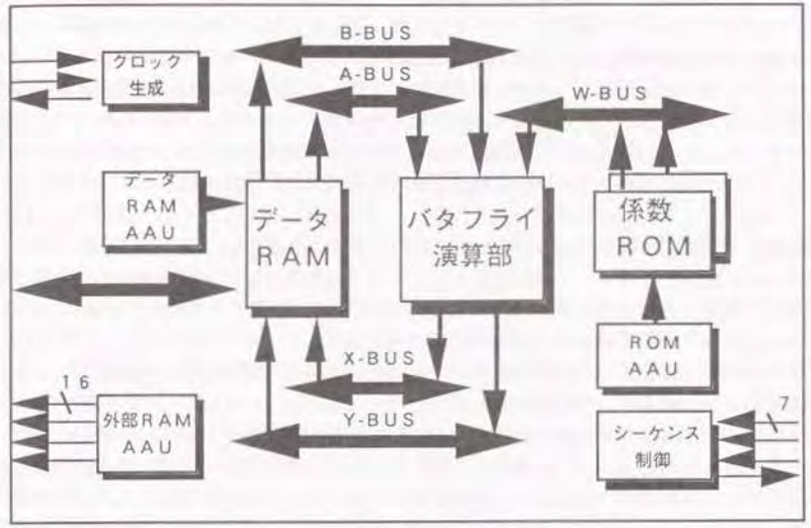


図4.5.1　２次元フーリエ変換プロセッサーのブロック図

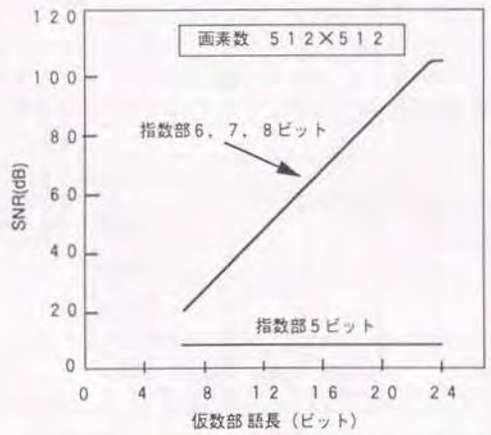


図4.5.2　有限語長シミュレーションによる語長とＳＮＲとの評価結果。画像データに５１２×５１２ポイントの２次元ＦＦＴと逆ＦＦＴを施して評価。

以上の方針に基づいて開発した、2次元フーリエ変換本プロセッサーの構成を図4.5.1に示す。複素バタフライ演算器，データRAM，係数ROM，各種アドレス生成部，シーケンス制御部，およびクロック生成部から構成されている。

複素バタフライ演算器は、浮動小数点乗算器4個と浮動小数点3入力加減算器4個とから構成される。演算語長を決定するにあたって、有限語長シミュレータ[25]を用いて、語長とSNR（Signal Noise Ratio）との関係を評価した。その結果を図4.5.2に示す。この図は、512×512画素の画像データに、演算語長をパラメータとして512×512ポイントの2次元FFTと逆FFTを施した場合のSNRの変化をプロットしたものである。明らかに、指数部の語長は、6ビット以上あれば十分なことがわかる。この結果と、他の浮動小数点DSPとのデータ互換性を考慮して、演算語長を24ビット（仮数部16ビット指数部8ビット）の浮動小数点形式とした。また図4.5.2から明らかなように、採用した語長でSNRは70dBとなり、画像処理に充分な精度を確保できることがわかる。

データRAMは2ライン分の複素データを格納できるダブルバッファー構成とし、チップ外部とのデータ転送と、内部演算との並列処理を可能とした。これにより、外部フレームメモリとのデータ入出力の問題を解決し、1次元FFTの連続処理を実現している。またバタフライ演算部との高並列データ転送を可能とするため、2ポートRAMを2並列で動作させるようにし、等価的に、2READと2WRITEができる4ポートデータRAMユニットの構成とした。そして、このユニットの4ポートが常時動作できるように工夫し、バタフライ演算部の高並列性能が最大限に発揮されるようにした。詳細は、4.5.3節で述べる。

演算部とデータRAMとは、4本のバスで結合されている。A-BUSとB-BUSを介してデータをバタフライ演算部へ転送し、X-BUSとY-BUSを介して演算結果をRAMへ転送する。係数データは、W-BUSを介して係数ROMから演算部へ転送する。また、データRAM，係数ROM，外部RAM用にそれぞれ専用のアドレス生成回路を設け、FFT処理に必要なすべてのデータが、独立かつ並列に読みだされ、そして結果が、書き込まれるようにした。また、FFT処理を指定する1ワードのマクロ命令を用意し、ユーザにとって使い易いインターフェースとした。FFTサイズを指定した後、外部からスタート信号を与えるだけで、シーケンス制御部から必要なアドレスとR/W制御信号等を発生して、自動的に処理を実行する。

4.5.2　バタフライ演算回路の構成[26]

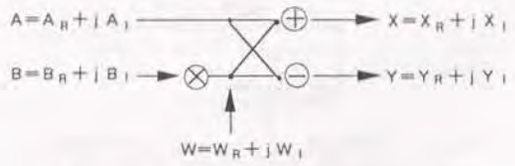


図4.5.3　バタフライ演算のシグナルフローグラフ

（1）全体構成

FFTは図4.5.3に示すバタフライ演算を繰り返すことにより実現される。これを具体的に式で表すと以下のようになる。

$$XR + jXI = AI + BRWR - BIWI + j\,(AI + BRWI + BIWR)$$
$$YR + jYI = AR - BRWR + BIWI + j\,(AI - BRWI - BIWR)$$

すなわち一つのバタフライ演算でBRWR,BIWI,BRWI,BIWRの4つの実乗算、と4つの3入力加減算が必要となる。

これらの演算を1マシンサイクルで実行するバタフライ演算器のブロック構成を図4.5.4に示す。入力データAはA-BUSを介して加減算器（ASU）に送られ、入力データBはB-BUSを介して乗算器に送られる。一方出力データX、Yはそれぞれ X-BUSとY-BUSを介してASUからデータRAMへ送られる。なお係数データはROMからW-BUSを介して乗算器（MPL）に送られる。MPLとASUとの中間にあるM-BUSはバタフライ演算器内に閉じたバスである。

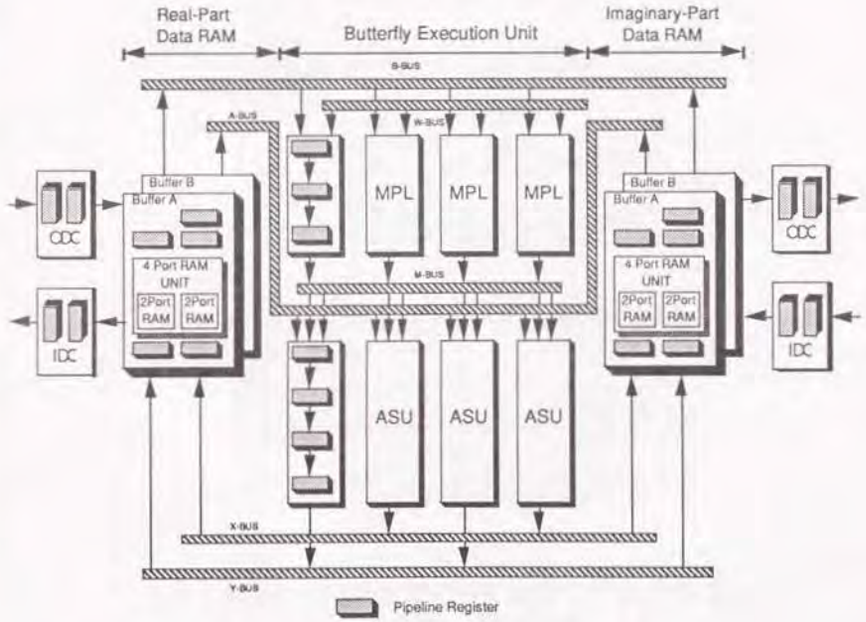
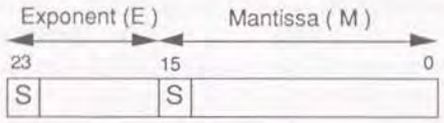


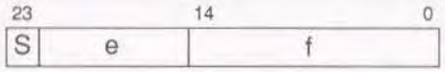
図4.5.4　バタフライ演算器とデータRAM

(1) 2's Floating : 2's Complement Floating-Point (24-bit)



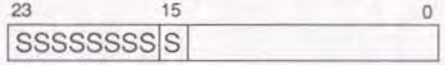
$$D = 2^{E} \cdot M : E = -2^{0}e_7 + 2^{-1}e_6 + \cdots + 2^{-7}e_0$$
$$M = -2^{0}m_{15} + 2^{-1}m_{14} + \cdots + 2^{-15}m_0$$

(2) IEEE Floating : Floating-Point Compatible with IEEE Recommended Single Format (24-bit)



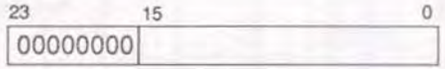
$$D = (-1)^{S} \cdot 2^{e-127} \cdot (1.f)$$

(3) 2's Compliment Fixed-Point (16-bit)



$$D = 2^{15} \cdot (-2^{0}m_{15} + 2^{-1}m_{14} + \cdots + 2^{-15}m_0)$$

(4) Straight Binary Fixed-Point (16-bit)



$$D = 2^{15} \cdot (2^{0}b_{15} + 2^{-1}b_{14} + \cdots + 2^{-15}b_0)$$

図4.5.5　2次元フーリエ変換LSIが処理できる入出力データ形式

浮動小数点乗算器の高速化技術については、すでに3.2.2節で詳述し、その構成を図3.2.1に示した。24ビットの入力データを乗算して24ビットの出力データを得る構成である。演算は2の補数で、2次のBoothのアルゴリズムを採用している。2段パイプライン構成であり、前段の加算器アレイ部では、Carry Save Adderを、また後段の最終加算には、Carry Select Adderを採用した。また3入力の浮動小数点加減算器の高速化技術については、3.2.3の（4）節で詳述し、その構成を図3.2.7に示した。3段パイプラインで構成され、初段で3入力データの桁合わせ処理を、2段目で仮数部の加算を、最終段で正規化と特殊数処理を行う。

（2）データ形式

採用した24ビットの浮動小数点形式（16E8）は、指数部，仮数部共2の補数である。この形式は、乗算，加算，特種数検出等で高速処理が容易な反面、標準浮動小数点形式の汎用マイクロプロセッサや、ＡＤ／ＤＡ変換器とのデータ互換性の点で問題がある。この変換に時間を要すると、内部での高速処理の利点が半減する。

そこで、ＬＳＩの入力と出力にデータ変換回路（ＩＤＣ：Input Data Converter，ＯＤＣ：Output Data Converter）を搭載し、バタフライ演算器のデータ形式と、チップ外部のデータ形式との変換を行うこととした。またこれらの部分を、それぞれパイプライン1段で

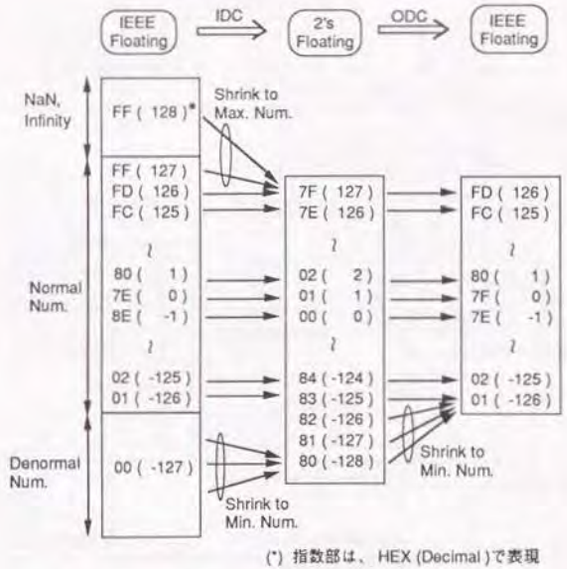


図4.5.6　IEEE浮動小数点と2の補数浮動小数点相互の変換と丸め処理（IDCおよびODC回路）

構成し、パイプライ遅延の増加を最小限に抑えた。

ＩＤＣとＯＤＣで処理できる４種のデータ形式を図４.５.５に示す。２の補数は、ＬＳＩ内部のデータと同一の形式である。IEEE浮動小数点は、IEEE-754規格の32ビット単精度規格の上位24ビットが、コンパチブルとなる形式である。すなわち、指数部と仮数部のデータの並びと、指数部８ビットの構成が同一であり、仮数部は、24ビットのうち上位16ビットがコンパチブルである。16ビットの２の補数固定小数点と、ストレートバイナリとは、それぞれ同一のデータ形式を有するＡＤ／ＤＡ変換器とコンパチブルである。

変換の中で特に、IEEE浮動小数点と２の補数浮動小数点間の変換では、特殊数の扱いに注意しなければ成らない。一つは非数（NaN）、もう一つは仮数部のスティキービットである。この変換時の丸め処理を図４.５.６に示す。

IEEE浮動小数点から２の補数浮動小数点への変換では、非数と無限大とを、２の補数浮動小数点で表現できる最大数に丸めた。また非正規化数は、同じく表現できる最小数に丸めた。このように２の補数系では、非正規化数が丸められているため、演算および、オーバフローとアンダーフローの検出を高速化できる。逆に、２の補数浮動小数点からIEEE浮動小数点への変換は容易である。その理由は、先の有限語長シミュレーションの結果より、ＦＦＴ演算では、指数部が６ビットのダイナミックレンジを有すれば、十分なS/Nを得られ

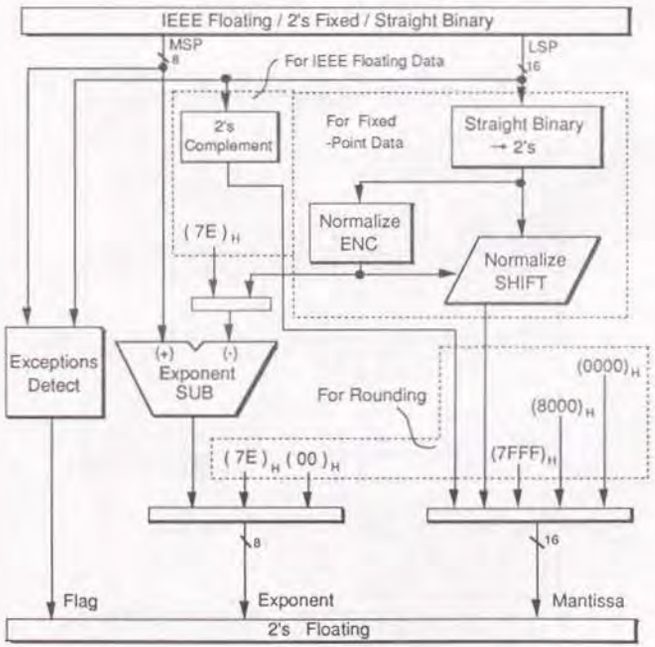


図４.５.７　入力変換回路

ることが示されたので、最大数や最小数の近似が、全体のＦＦＴ演算の精度にほとんど影響しないと言えるからである。そのため、出力のIEEE浮動小数点データには、非正規化数を用いず、図４.５.６に示す用に、正規化数の最小数に丸めて簡単化した。

（２-１）入力変換回路（ＩＤＣ）

入力変換回路の構成を図４.５.７に示す。IEEE浮動小数点数の仮数部は、（2's Complement）ブロックで２の補数に変換される。また、ストレートバイナリのデータは、固定の２の補数に変換されタ後、正規化回路（Normalize ENC + Normalize SHIFT）により、２の補数浮動小数点に変換される。

指数部減算回路（Exponent SUB）は、２つの機能を兼ねている。１つは、IEEE浮動小数点から２の補数浮動小数点へ変換するとき、IEEE 浮動小数点数の指数部の値（バイアス）と、仮数部のスティキィビットを考慮して変換する機能であり、もう１つは、仮数部を正規化するときに必要な、指数部の補正機能である。

（２-２）出力変換回路（ＯＤＣ）

出力変換回路は、指数部加算器（Exponent ADD），２の補数器（2's Complement），右バレルシフタ（Right SHIFT）を主要回路としている。その構成を図４.５.８に示す。指数部加算器と２の補数器は、２の補数浮動小数点数からIEEE浮動小数点数への変換のためであ

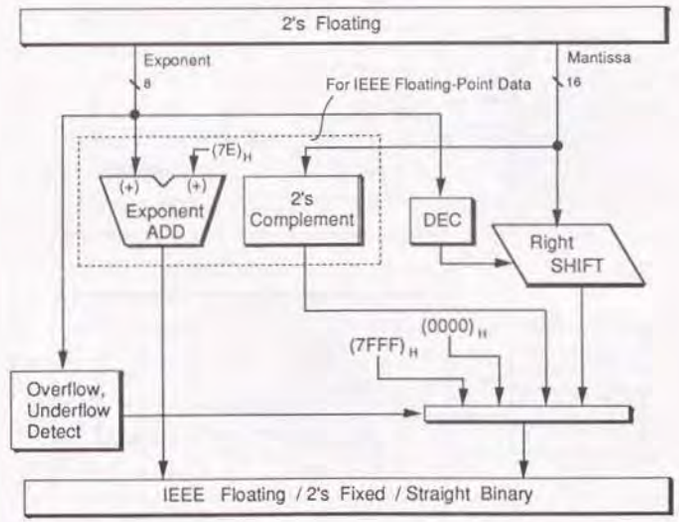


図４.５.８　出力変換回路

る。また2の補数器は、ストレートバイナリへの変換にも使用する。右バレルシフタは、2の補数浮動小数点数を、2の補数とストレートバイナリの2種の固定小数点数へ、変換するときに使用する。

4.5.3　データRAM[27]

ここでは、バタフライ演算回路を最高性能で動作させるように設計した、データRAMの構成法について議論する。データRAMは、ダブルバッファー構成とした。その理由は、2次元データから、行単位で1次元データを取り出しながら、1次元FFTを連続処理できるようにするためである。このバブルバッファの動作を図4.5.9に示す。

図4.5.9の上図では、バッファーAが外部とのデータ送受用として、またバッファーBがFFT演算用として用いられている。このときバッファーAは、まず第（n-3）行のFFT結果を外部メモリに送信し、続いて第（n-1）行のデータを外部メモリから読み込む。この間バッファーBは、バタフライ演算回路との間でデータ転送を繰り返し、第（n-2）行のデータに関してFFT演算を実行している。

次のフレームでは（図4.5.9の下図）、バッファーAとBが切り替わる。すなわち前フレームで計算された第（n-2）行のFFT結果が、バッファーBから外部メモリに送信され、続いて第（n）行のデータを外部メモリから読み込む。この間バッファーAはバタフライ演算回路との間でデータ転送を繰り返し、前フレームで読み込んだ第（n-1）行のデータに関してFFT演算を実行する。こうして1次元FFTを連続して処理することが可

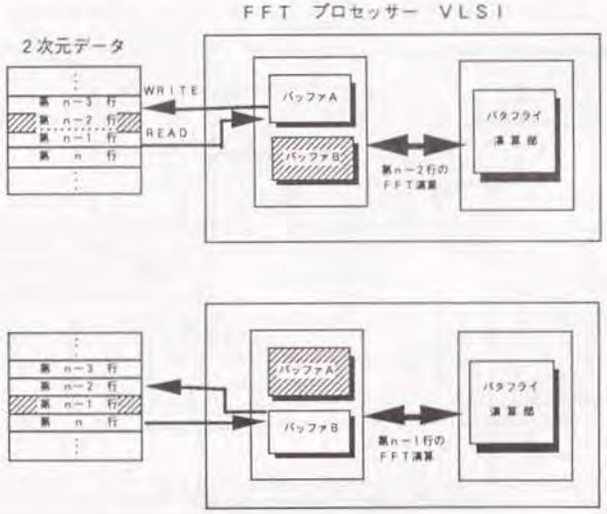


図4.5.9　1次元FFTのパイプライン処理

能となる。

次に外部転送時間と内部演算時間の比について考察する。1次元FFTのポイント数をNとすると、チップと外部メモリ間で、送受合わせて（2N）ワードの転送を必要とする。1ワード転送に1サイクルを要するバースト転送を想定すると、データ転送に必要なサイクル数、MEXTは、

$$M_{EXT} = 2N \qquad (4.5.1)$$

となる。一方FFT演算に必要なサイクル数MINTは、バタフライ演算の数が$N/2 \cdot \log_2 N$であるから、

$$M_{INT} = \frac{N}{2} \log_2 N + \alpha \qquad (4.5.2)$$

となる。ここでαはパイプラインのオーバーヘッドで、Nが大きいと無視できるので、

$$M_{INT} = \frac{N}{2} \log_2 N \qquad (4.5.3)$$

となる。式（4.5.1）と式（4.5.3）とから、内部サイクル数と外部サイクル数との比Rは

$$R = \frac{M_{INT}}{M_{EXT}} = \frac{\log_2 N}{4} \qquad (4.5.4)$$

となる。例えば、N=256ポイントの場合R=2となる。これは、外部転送サイクル数は内部演算サイクル数の1/2であることを意味する。したがって外部サイクルタイムを内部サイクルタイムに比べて2倍遅くしても、全体の性能は変わらない。一般的に、外部サイクルタイムは、内部サイクルタイムに比して高速化するのが難しいので、これは魅力的な性質である。本LSIの場合、動画像の実時間処理を目標にしているので、内部サイクルタイムは、25nsにしなければならない。これに対し、外部サイクルタイムは、50nsとすればよい。

ここでデータRAMの構成の説明に戻る。図4.5.10にバッファー内部の構成を示す。128ワード×24ビットの2ポートRAMを基本単位とし、このRAMを2つまとめて4ポートデータRAMユニットを構成した。そして、この4ポートデータRAMユニットを2つ用いて、それぞれ実部データ用と虚部のデータ用とした。

次に4ポートデータRAMユニットの機能について説明する。このユニットの特長は、4ポートの機能として、読みだし2ポートと書き込みに2ポートまでの機能に限定したことである。この限定により、2ポートRAMを2個並列にするだけで、4ポートデータRAMユニットを実現できる。そのために、FFTのバタフライ演算の特徴に合わせて3つの制約を設定した。以下に、バタフライ演算の2つの特徴と、設定した3つの制約条件を示す。

・FFTのバタフライ演算の特徴

(#1) 最終ステージ以外では、入力データＸとＹのアドレス間距離は常に偶数である。したがって、ＸとＹのアドレスは偶数同士また奇数同士のペアーとなる。

(#2) 最終ステージでは、入力データＸとＹのアドレス間距離は常に１となる。したがって、ＸとＹのアドレスは常に偶数と奇数の組合せとなる。

・設定した制約条件

(#1) バタフライ演算部のパイプライン段数を、奇数とする。

(#2) インプレース演算とする。すなわちバタフライ演算で、入力データをＲＡＭから読みだすアドレスと、結果を書き込むアドレスとは等しい。

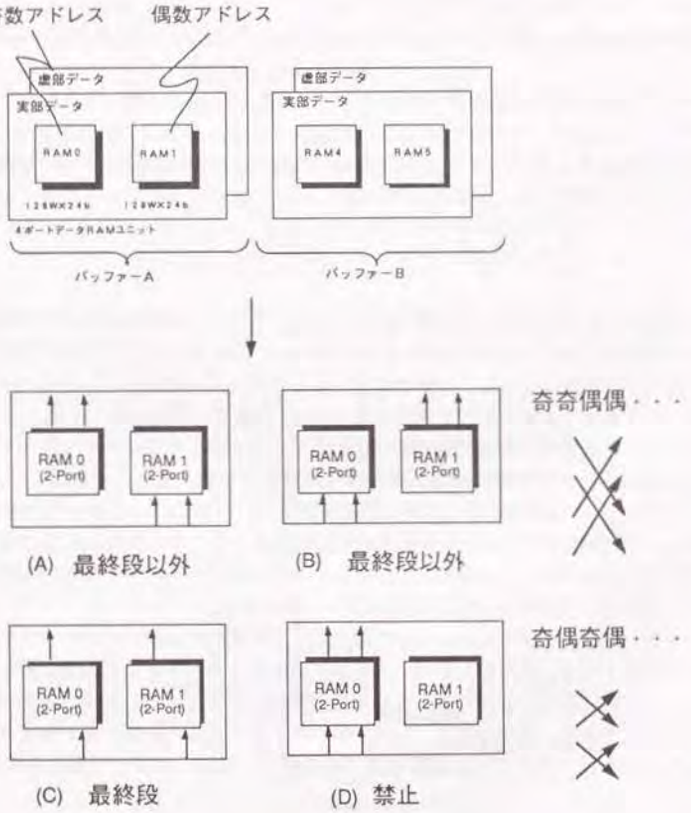


図４.５.１０　４ポートＲＡＭの構成とアクセスパターン

(#3) 一方のＲＡＭに偶数アドレスのデータを格納し、他方のＲＡＭに奇数アドレスのデータを格納する。

以上の特徴と制約により、データＲＡＭへのアクセスパターンは図４.５.１０の（Ａ）（Ｂ）または（Ｃ）のいずれかに限定される。（Ａ）と（Ｂ）は最終ステージ以外のアクセスパターンで、一方のＲＡＭから２つのデータを読みだし、他の一方のＲＡＭに２つのデータを書き込む場合である。また（Ｃ）は最終ステージのアクセスパターンで、２つのＲＡＭがそれぞれ１つのデータを読みだし、１つのデータを書き込む場合である。したがって、全バタフライ演算に渡って、常時４並列アクセスが達成できる。

これに対し、たとえば制約条件の(#1)が満足されないと、ＲＡＭへのアクセスパターンは図４.５.１０（Ｄ）ようになる。これは、いずれか一方の２ポートＲＡＭに４つのアクセスが集中してしまうので実現不可能である。

このように、２ポートＲＡＭを２並列で使った、２リード＆２ライトの４ポートデータＲＡＭを搭載したことと、４ポートＲＡＭへのアクセスが必ず２個の２ポートＲＡＭに分散される仕組みを持たせたこととにより、高速なパイプライン演算を実現した。

４.５.４　命令コード

本ＬＳＩに与える外部からの命令コードは、唯一１ワードのみと、非常に簡単化した。そのフィールド構成を図４.５.１１に示し、各フィールドの説明を行う。

外部メモリ転送サイクルフィールドでは、内部演算サイクルと外部データ転送サイクルとの比を指定する。この値は、第４.５.３節で考察したように、例えば256ポイントの場合２とするのがが最適である。しかしポイント数が小さい場合には、内部演算量が相対的に低下するので、１として外部転送サイクルを上げないと、転送によるボトルネックが発生する。

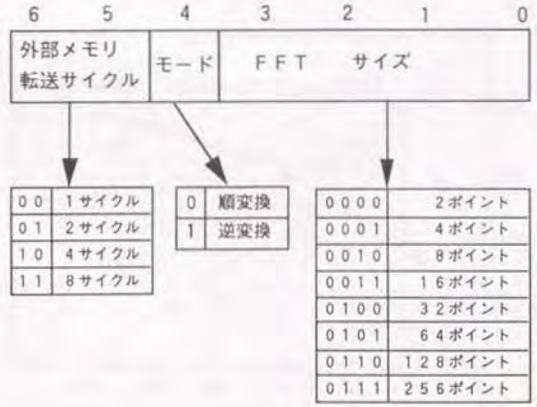


図４.５.１１　命令コード

そこで、外部メモリの速度と転送ボトルネックとを考慮して、最適な値を指定しなければならない。モードフィールドでは、フーリエ変換の順方向／逆方向を指定する。順方向と逆方向の違いは、係数データの虚部の符号にあるので、この指定により、係数テーブルから読みだすデータの虚数部符号反転を制御している。ＦＦＴサイズフィールドでは、文字通り、フーリエ変換のサイズＮを指定する。

以上の指定を行った後スタート信号を入力するだけで、ＬＳＩ内部のシーケンス制御がスタートし、各種アドレスおよびメモリのＲ／Ｗ制御信号等を自動的に生成して、指定したサイズの２次元フーリエ変換を完了させ、結果を外部フレームメモリに格納する。

表4.5.2　ＦＦＴプロセッサーＬＳＩの諸元

プロセス	0．8μmCMOS
素子数	380，000
チップサイズ	11．58×11．58mm^2
クロック	40MHz
ピン数	132
データ形式	24ビット浮動（16E8）
128ポイント1次元ＦＦＴ	12．8μs
256ポイント1次元ＦＦＴ	27．4μs
128×128ポイント2次元ＦＦＴ	3．32ms
256×256ポイント2次元ＦＦＴ	14．1ms

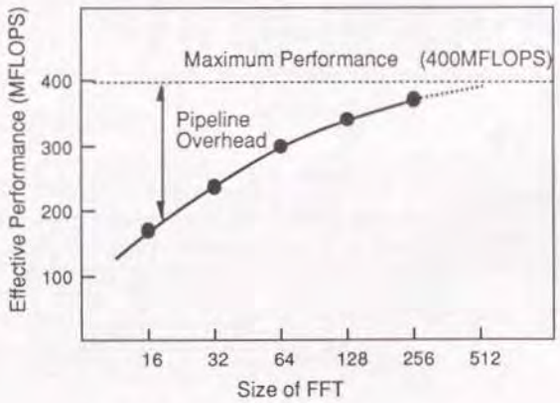


図4.5.12　ＦＦＴサイズとプロセッサの性能

4.5.5　性能評価

２次元フーリエ変換ＬＳＩのアドレス生成技術と、それを使ったアーキテクチャ技術の効果を確認するため、0.8μm-CMOSにより試作を行った。試作ＬＳＩの諸元を表4.5.2に示す。実測により、256×256ポイントの２次元ＦＦＴを14.1msで処理できることを確認した。これは、ビデオレートの実時間で処理が可能なことを示している。

ところで、本ＬＳＩは、データ処理の流れに深いパイプラインを採用しているので、ＦＦＴのバタフライステージ間に、無視しえないパイプラインバブルが発生する。図4.5.1

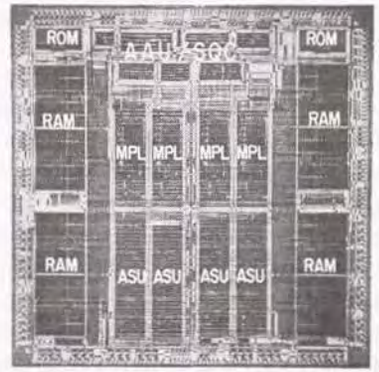


図4.5.13　２次元フーリエ変換ＬＳＩのチップ写真

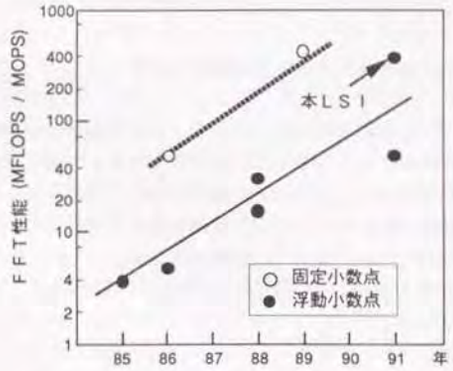


図4.5.14　従来のＦＦＴ専用ＬＳＩ，ＤＳＰ-ＬＳＩによるＦＦＴ処理性能と、本ＬＳＩの性能との比較

2は、1次元ＦＦＴを実行した際のパイプラインバブルによるオーバーヘッドを示したものである。この結果、ＦＦＴのサイズが大きくなると共に、そのオーバーヘッドは減少し、256ポイントＦＦＴでは１０％以下となることがわかる。すなわち、今後ＬＳＩ技術の発展と共に、ＦＦＴは、よりポイント数の大きいものを処理する方向になっていくので、本アーキテクチャによるパイプライン処理のオーバヘッドは、徐々に無視できるものとなる。将来的に、より有利となっていくアーキテクチャと考えて良い。

本ＬＳＩのチップ写真を、図4.5.13に示す。アーキテクチャ設計段階で配置の対称性も考慮に入れているので、高密度なレイアウトが実現されている。各演算器，ＲＡＭ，ＲＯＭは、すべてマニュアル設計である。

従来ＬＳＩと比較した本ＬＳＩの性能を図4.5.14に示す。一般に浮動小数点プロセッサの性能は、固定小数点に比して約1桁劣っている。この理由は、先にも述べたように、ハードウエア資源上の負担が重いことと、パイプライン演算サイクルの高速化が難しいからである。しかしながら、本2次元フーリエ変換ＬＳＩは、3.2節で述べた、浮動少数点演算器の高速化技術，4.4節で述べた、2次元フーリエ変換ＬＳＩのアドス生成技術，および本節のハードウェアアーキテクチャ技術とにより、従来の浮動小数点のＬＳＩに比して1桁近くの高速化を達成し、浮動小数点演算形式であるにも関わらず、現在の最高速の固定小数点プロセッサＬＳＩと同等の処理速度を有している。

以上4.4節と4.5節での成果を、以下に列挙する。

(#1) 複素バタフライ演算を1サイクルで処理する高並列アーキテクチャの提唱。
(#2) バタフライアルゴリズムの特長を利用した、4ポートデータメモリ構成法の提唱、および演算器の並列度と整合したデータ転送方式の考案。
(#3) チップ外部のフレームメモリへの2次元アドレス生成法と、内部データメモリへのバタフライアドレスおよびビットリバースアドレス生成法の考案
(#4) ビデオレートの実時間で2次元フーリエ変換を行うＬＳＩの実現。

4.6 ビデオ符号化ＤＳＰのアドレス生成技術[28][29]

4.6.1 ビデオ符号化ＤＳＰのアドレス生成の要求条件

画像処理の中でもビデオの符号化は、リアルタイム性が要求されるため、ＬＳＩ化を行う上で、非常に厳しい分野のひとつになっている。この負担を大幅に軽減させる画期的な技術が、ブロック単位の処理である。この利点を、例えば、ＩＳＯ（International Organization for Standardization）標準化が進められている、蓄積画像の符号化[30]で見てみる。この場合は、動きベクトル検出とＤＣＴ（Discreate Cosine Transform）が主要なベクトル演算であり、それぞれ、処理の演算量とブロックサイズの関係は、以下のようになる。ただし、ブロックサイズをｎ×ｎとする。

・フルサーチでの動きベクトル検出：ｎの4乗に比例
・ＤＣＴ演算　　：ｎの3乗に比例

すなわち、フレーム単位やフィールド単位の処理では、途方も無い演算量となることがわかる。そこで、ＩＳＯ／ＭＰＥＧ1では、動きベクトル検出のときには１６×１６画素，ＤＣＴのときには８×８画素からなる画素ブロックを処理単位として演算量を削減させている。

そこで、ブロックを単位とするアルゴリズムを効率良く処理できることが重要であり、そのためには、2つのアドレス生成機能が要求される。1つは、外部のフレームメモリから、チップ内部で処理する画素ブロックを切り出してくるための、2次元のブロック切り出し用のアドレス生成である。そしてもう1つは、切り出してきたブロックデータを、演算器のパイプラインへ供給するときに必要となるブロック内のアドレス生成であり、符号化アルゴリズムの種類に応じて種々の機能が要求される。

前者のブロック切り出しは、比較的容易であるが、後者の場合は、チップ内に搭載できるメモリ容量の制限を考慮しながら、種々のアルゴリズムに対処できければならないので、容易ではない。すなわち、以下の2条件を満足しなければならない。

(#1) 種々の符号化アルゴリズムに対処できる汎用性
(#2) メモリ容量の制限を補って、高処理能力化できる機能

次節では、ブロック内のアドレス生成で考案した技術について述べる。また、このアドセス生成技術を使って構成したビデオ符号化ＤＳＰについては、6.6節で述べる。

4.6.2 データメモリのアドレス生成

ここでは、チップに搭載できる限られたメモリ容量を活用して、複数の画素ブロックに渡って並列処理を行う、並列処理用アドレス生成技術について述べる。

データメモリへの要求として最も厳しいのは、広い領域のアクセスを並列に行う場合である。この代表的な例として、動き補償フレーム間予測符号化の動きベクトル検出用距離計算処理が挙げられる。これは、処理フレーム（現フレーム）の中の16×16サイズの画素ブロックを、前フレームの中の探索領域（例えば48×32(～2000)）でスキャンさせながら距離計算を行い、探索領域の中から距離の最も近い画素ブロックを検出する処理である。

この場合に、従来のＳＩＭＤ（Single Instruction Multi Data stream）方式を使って、各ブロック独立に並列演算を行おうとすると、メモリ容量ネックが生じる。なぜなら、ＳＩＭＤ方式では、それぞれのデータストリーム毎に、演算器（以下ＤＰＵ：Data Processing Unit）と、そのＤＰＵにブロックデータを供給するデータメモリ（通常デュアルポートメモリ）が必要だからである。必要なメモリ容量を見積ってみると、それぞれのＤＰＵごとに、処理フレームの画素ブロックデータと、その画素ブロックをスキャンさせる前フレームのスキャン領域データが必要であり、上記の例の場合、ＤＰＵごとに最低2KWのデュアルポートメモリが必要となる。このため、ＳＩＭＤの並列度を4，メモリのデータ語長を16ビットとすると、３２Ｋｂのデュアルポートｒａｍを4バンク搭載しなければならないことになる。これは、0.8μmのＬＳＩ技術の場合に、チップのほとんどの領域がデータＲＡＭで占められることを意味し、実用的でない。ましてや、ＳＩＭＤの並列度を16とすると、0.5μm技術を使っても搭載困難である。

この問題を解決するため、アドレス生成とバス構造とに工夫を行い、それぞれのＤＰＵのスキャン領域を、バンクメモリ間に渡って共通に持たせることにより、ＤＰＵごとの並列探索機能を維持しながら、探索領域のメモリ容量を削減する方法を考案した。この方法を用いると、上記４並列の場合に、２ＫＷ×４の構成を、512Ｗ×４と、ほぼ４分の１まで削減可能となる。

上記のアドレス生成技術に加えて、汎用性を持たせるために、合計３種のアドレスモードを設けた。各アドレスモードの概念を図4.6.1に示す。４個のＤＰＵを想定し、それぞれ、ＤＰＵ０～３と指定する。そして各ＤＰＵに対応して、４バンクのデータＲＡＭ、ＣＭ０～３が存在し、各ＤＰＵｉ（ｉ＝０～３）にＣＭｉが対応している構造である。この内モード（Ａ）とモード（Ｂ）の２種は、従来のＳＩＭＤアドレスモードとバンク間のシリアルアクセスモードである。そしてモード（Ｃ）が、上記提案の機能を満足させるものであり、バンク間並列アドレスモードと呼ぶ。各アドレスモードの説明を以下に行う。

（Ａ）ＳＩＭＤアドレスモード

メモリＣＭ０～３のアドレスが、それぞれ独立である。また、アドレス空間はそれぞれ512Ｗである。このモードは、各ＤＰＵｉに１対１対応でＣＭｉを配置し、それぞれのＤＰＵに独立に画素ブロックを割当てて処理するものである。処理の間各画素ブロックが独立で、かつ必要なメモリ容量が１つのデータメモリに収まる場合に有効である。例としては、２次元フィルタ処理やＤＣＴが挙げられる。

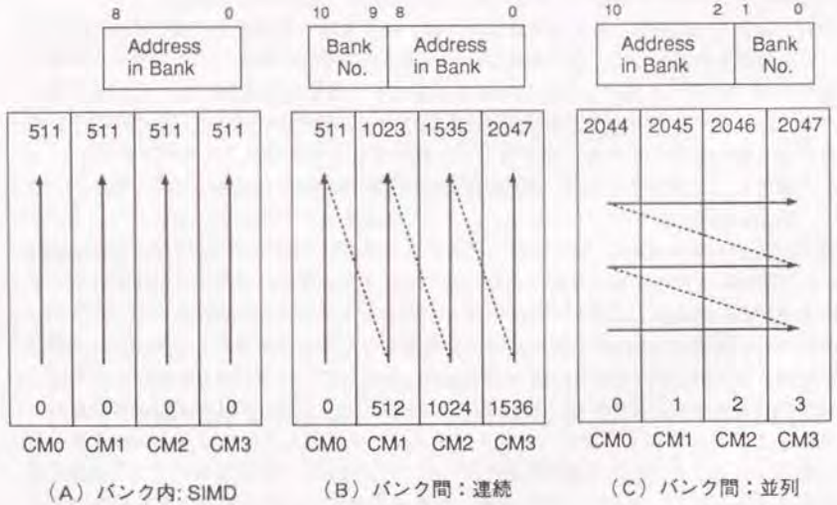


図4.6.1　3種のアドレスモード

これに対応する具体的なアドレス機能は、１並列の場合と同様である。フィルタ処理には、２次元画素ブロックのブロック内スキャンアドレッシング機能を持たせる。またＦＤＣＴ（Fast Discreate Cosine Transform）の処理に対応して、ビットリバースアドレスやバタフライアドレス等、通常ＤＳＰの持つ基本機能を備えさせる。

（Ｂ）バンク間シリアルアドレスモード

このモードでは、512Ｗの４バンクメモリを２ＫＷの連続したメモリとして使用する。オンチップのＤＭＡ（Direct Memory Access）プロセッサを用いて外部メモリとのデータ入出力を行う場合や、２ＫＷの連続したメモリに複数の画素ブロックデータをロードして、ブロック間にまたがる処理を行う場合に適している。ただし並列度はポート当たり１となる。

アドレス生成は、９ビットのバンク内アドレス（512Ｗ）の上位に、２ビットのバンク番号（４バンク）を付与することにより、連続した11ビットアドレスで行う。

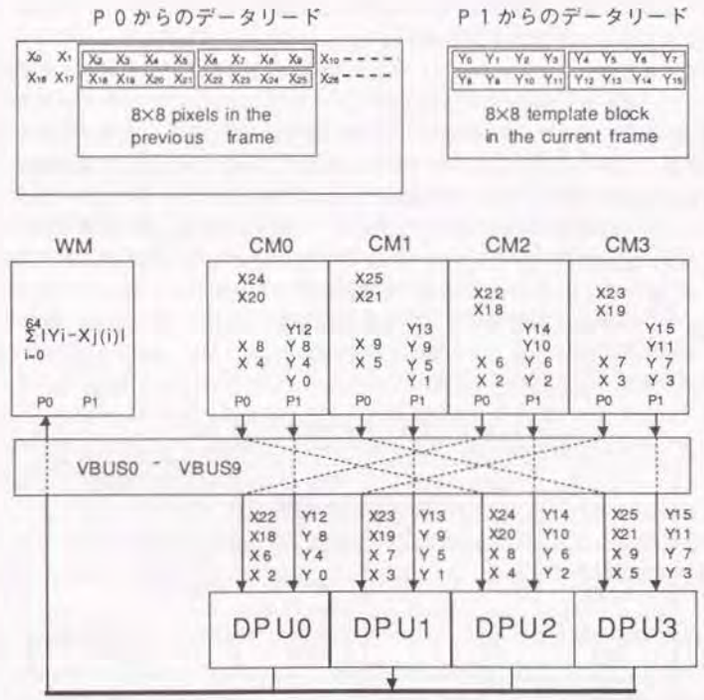


図4.6.2　バンク間並列アドレスモードを使った処理例
（動きベクター検出用距離計算）

（C）バンク間並列アドレスモード

このアドレスモードの具体的な使用例として、動きベクトルの距離計算を取り上げる。計算式を次式に示す。

$$\mathrm{Norm}(j) = \sum_{i=0}^{63} |Y_i - X_{j(i)}| \qquad (4.6.1)$$

ここに、

i ∈ [処理フレームの中の、8×8のテンプレート画素ブロック]
Yi　：8×8テンプレート画素ブロックの画素
Xj(i)：前フレームの探索領域内で、テンプレートの画素 Yi に対応する画素
（サーチ領域は、512W のバンク内メモリ容量より大きい）

（4.6.1）式の距離計算を、4個のデータメモリCM0～3と、4個のDPU0～3とにより並列に実行する。このデータ処理のフローを図4.6.2に示す。各データメモリのP1ポートは、処理フレームの中の8×8テンプレート画素ブロック内のデータを読み、P0ポートは前フレームのサーチ領域のデータを読む。

この時、4個のデータメモリの、それぞれのP0ポートとP1ポートのデータを並列に読み出せるようにするため、以下の2点の工夫を行った。1つはテンプレートの移動に伴って、探索領域の読みだし開始アドレスを動かし、常に対応する開始アドレスから連続する4データを並列に読みだせるようにしたことである。これで、両ポートからそれぞれ4データづつ読みだされるので、同時に8データが読みだされる。

もう1つは、P1からの4個のテンプレートデータとP2からの4個の探索領域のデータとから、それぞれ対応するテンプレートデータと探索領域データとを同一のDPUに取り込むため、CM0～3とDPU0～3とを結合している8本のデータバス（VBUS0～7）を使って、いずれかを並列シフトして取り込むようにしたことである。図4.6.2の例

表4.6.1　3種のアドレスモードの特徴

アドレスモード	並列アドレッシングの数 (#1)	アドレス空間	応　用
バンク内SIMD	4	512 W	・ループフィルタ ・DCT/IDCT
バンク間連続	1	2048 W	・外部メモリからのデータ転送
バンク間並列	4	2048 W	・動きベクトル検出

(#1) 1ポート当り

では、探索の開始アドレスが、2画素ずれたX2からとなっているため、探索領域のデータを読みだすP0ポートからのデータを、サイクリックに2画分シストすることで、テンプレートの画素と対応するようにした。これにより、4組のデータ (Yi and Xj) を4個の DPU へパイプライン状に連続して取り込むことができる。

各DPUはそれぞれ距離計算を行うと共に、DPU間のトリー結合網（6.6節で詳述する）を使って4画素の累算を行う。これにより4画素分の距離の累算値が、常時DPU1に出力される。そしてDPU1の出力は、9番目のデータバスVBUS8を介して、共通のデータメモリ（WM）に書き込まれる。以上述べた3種アドレスモードの特長を、表4.6.1にまとめて示す。

4.7　むすび

本章では、信号処理LSIの高速化技術の中で、高速演算回路を活かしてパイプライン演算を行うときに重要なデータメモリのアドレス生成技術について述べた。まず4.2節で、1次元信号処理用のアドレス生成技術について検討した。これは、すべての信号処理LSIのアドレス生成の基本となるものである。特に音声信号を汎用的に処理するDSPを取り上げ、1次元アドレスを汎用的に生成する方法について検討し、ハードウエアを共通化して効率良く生成する方法を提案した。

次に4.4節と4.6節では、2次元信号処理用のアドレス生成について述べた。4.4節では、特に2次元の高速フーリエ変換に特化した。第1の特長は、チップ外部に2次元のフレームメモリを置き、1次元FFT単位で、チップ内へのデータの取り込み，FFT演算，処理結果のフレームメモリへの書き込みを、インタリーブに行うことができるメモリ構成とそのアドレス生成技術である。第2の特長は、複素バタフライ演算回路をオンチップ化して、複素バタフライ単位に、データメモリと演算器間でパイプライン演算を行えるようにした、データパスの構成法とメモリのアドレス生成技術である。

4.6節では、動画の符号化を行うビデオDSPのアドレス生成について述べた。ここでは、複数の演算器を搭載したSIMDの並列動作を前提とし、複数の演算器に常時データを供給して並列にパイプライン動作させるためのメモリ構造と、そのアドレス制御法を検討した。そして、オンチップ化できるメモリ容量の制限を解決するため、バンク分けしたメモリを、アルゴリズムに合わせて有機的に並列動作させるアドレス制御法を提案し、動きベクトル検出演算等で有効なことを示した。

4.3節では、4.2節でのアドレス生成技術の成果を活かした、音声信号処理DSPのアーキテクチャとその試作結果、および代表的な2種の応用事例を示した。開発当時、最も高性能なDSPであることを示し、アドレス生成技術とアーキテクチャの有効性を実証した。

また4.5節では、4.4節で検討したFFTのアドレス生成技術を活かした、2次元の高速フーリエ変換をビデオレートの実時間で処理できる、浮動小数点の2次元フーリエ変換プロセッサーLSIのアーキテクチャーについての検討結果を述べた。演算器の並列性能を最大限に引き出すための高並列データ転送方法の考案と、各種アドレス生成手法とにより、従来の浮動小数点DSPの2桁上，浮動小数点専用LSIの1桁上の性能である、1チップ

で４００ＭＦＬＯＰＳの性能を、０.８μｍＣＭＯＳで達成した。

４.８　参考文献

[1] H.Yamauchi, et al.,"An 18-Bit Floating-Point Signal Processor VLSI with an On-Chip 512W Dual-Port RAM", IEEE ICASSP'85, p.204, 1985.

[2] 金子、山内、" ディジタル音声信号処理に適した汎用アドレス制御法" 、電子情報通信学会技術研究報告, CAS82-194, pp.57-62, (Mar.,1983).

[3] 金子、山内、" 複素ベクトル演算に適したアドレス制御法" 、昭和５９年度電子情報通信学会総合全国大会, No.337, (Mar.,1984).

[4] 金子、山内、" 複素ベクトル演算に適したアドレス制御法" 、信学論(D), J69-d, 4, pp.634-636, 1986.

[5] 金子、山内、" ビットリバースアドレスの制御法" 、信学論(D), J69-D, 7, pp.1124-1126, 1986.

[6] 渡部、小野、鴇沢、山内、" 音声符号化用プロセッサのアーキテクチャ" 、電子情報通信学会技術研究報告, CS82-87, pp.89-96, (Nov.,1982).

[7] S.Kozuka, N.Watanabe, S.Ono, H.Yamauchi, "Signal Processor Architecture for High Complex Coding", IEEE Global Conference, pp. 45-1-1, (Dec.,1983).

[8] H.Yamauchi, T.Kaneko, J.Takahashi, A.Iwata, "Speech Signal Processor VLSI Family", Proc. International Workshop on Digital Signal Processing, pp.2a 1-4, (Jun.,1985).

[9] K.Ueda, et al., "CHAMP:Chip floorplan for hierarchical VLSI layout design", IEEE Trans. Computer-Aided Design, CAD-4, pp.12-22, (Jan. 1985).

[10] T.Adachi, et al., "Hierarchical top-down layout design method for VLSI chip", Proc. 19th IEEE Design Automation Conf., pp.785-791, (June 1982).

[11] T.Kaneko, et al., "A 50ns floating-point signal processor VLSI, Proc. ICASSP86, pp.401-404, (April, 1986).

[12] 増田、須藤、永島、 "周波数領域でのパターン認識を利用した埋設物探知技術" 、宇宙航行エレクトロニクスシンポジウム, SANE-87-54, (1988-01).

[13] Y.Nagashima, J.Maguda, et al., "Underground radar system utilizing pattern recognation technique in the frequency domain" , Proc, ISNCR-89, pp. 689-692, (Nov.,1989).

[14] 永島、増田、山内、" 地下埋設物探知技術におけるパターン認識技術とその信号処理の高速化の研究" 、電子情報通信学会論文誌, Vol. J-74-C-2 No.5, pp.317-324, (May.,1991).

[15] 持田、雁部、" ディジタル信号処理プロセッサ" 、電子情報通信学会誌７月号, pp.757-765, 1989.

[16] Y.Kawakami, et al., "A 32b Floating Point CMOS Digital Signal Processor", ISSCC Digest of Technical Papers, pp.86-87, (Feb.,1986).

[17] A.Kanuma, et al., "A 16MHz 32bit Pipelined CMOS Image Processor", ISSCC Digest of Technical Paper, pp.102-103, (Feb.1986).

[18] J.O'Brien, et al.,"A 200 MIPS Single-Chip 1K FFT Processor", ISSCC Digest of Technical Papers, pp.166-167, (Feb.,1989).

[19] A.V.Oppenheim and R.W.Schafer, "Discrete-Time Signal Processing", Prentice Hall, New Jersey, 1989.

[20] J.W.Cooley and J.W.Tukey, "AN Algorithm for the Machine Computation of Complex Fourier Series", Mathematics of Computations, Vol.19, pp.297-301, (Apr.,1965).

[21] S.Winograd, "On Computing the Discrete Fourier Transform", Mathematics of Computation, Vol.32, No.141, pp.175-199, (Jan.,1978).

[22] J.S.Lim, "Two-Dimensional Signal and Image Processing", Prentice Hall, New Jersey, 1990.

[23] H.T.Kung and C.R.Lierseron, "Systolic Arrays for VLSI", Sparse Matrix Proceedings, editons I.S.Buff and G.W.Stewart, 1978.

[24] E.Arnould, et al., "A Systolic Array Computer", ICASSP85, pp.232-235, 1985.

[25] 田中、小林、 "信号処理LSI開発用有限語長シミュレータ" 、信学技法CAS85-165, (Jan.,1989).

[26] H.Yamauchi, H.Miyanaga, "Architecture of a Floating-Point Butterfly Execution Unit in a 400-MFLOPS Processor VLSI and its Implementation", IEICE Transactions, Vol. E74, No.12, pp.3852-3860, (Nov.,1991).

[27] M.Miyanaga, H.Yamauchi, K.Matsuda, "A Real-Time 256×256 Point Two-Dimensional FFT Single-Chip Processor", IEEE-ICASSP'91, pp.1193-1196, (May.,1991).

[28] 南、山内、田代、鈴木、笠井、高橋、遠藤、浜口、" ビデオシグナルプロセッサＩＤＳＰのデータフロー制御" 、電子情報通信学会技術研究報告, ICD91-12, pp.25-32, (Apr.,1991).

[29] 山内、笠井、南、田代、鈴木、" 蜜結合型高並列ビデオプロセッサ" 、回路とシステム軽井沢ワークショップ予稿, pp.355-360, (Apr.,1991).

[30] ____, " Coding of moving picture for digital storage media," ISO/IECJTC1/SC2/WG11 MPEG90/176, 1990.