高速汎用信号処理装置ＫＳＰの マイクロプログラム制御

所 真理雄　内田俊一　森秀樹　金子憲幸　島田幹郎　相磯秀夫
（慶応義塾大学　工学部）

1．はじめに

CoolyとTukeyによる高速フーリエ変換（ＦＦＴ）アルゴリズム[1][2]の開発により，ディジタル計算機による信号処理が広く用いられるようになった。また，近年のパターン情報処理の必要性に伴い，信号解析技術に対する重要性は，ますます増大し，特に大量データに対する実時間処理能力が重要視されるに至った。このため処理方法が，汎用計算機の機械語プログラムによるものから，汎用計算機に信号処理用のマイクロプログラムを追加したり，簡単なハードウェアを付加したものへと発展し，さらに最近では，信号処理のための専用ハードウェア・プロセッサ[3][4]が開発され，注目を集めている。

最近の信号処理装置に要求される事項は，処理速度に関するもの，演算処理能力に関するもの，システム構成の柔軟性に関するものに，大きく分類することができる。

i) 処理速度　処理速度に関して，1024点の標本のＦＦＴの場合を考えると，現状では汎用計算機のプログラムでは約1 sec，マイクロプログラムや簡単なハードウェアを付加したものでは約100 ms，専用ハードウェア・プロセッサでは約10 ms程度で処理される。実時間処理や大量データに対する処理ではさらに高速性が要求されている。

ii) 演算処理能力　信号処理においては一般にフーリエ変換がその基本的な演算として用いられている。これによってフーリエ逆変換，畳込積分，自己及び相互相関，伝達関数，コーヒレント関数，パワースペクトラム等の有用な演算を行うことができる。また，応用分野によっては各種ウィンドイング，平均化，積算，ピーク値ホールディング等の演算も必要となる。そして，これらの演算に加えて，今後どのような種類の演算が要求されるかを予測することは，極めて困難である。このために，今後の信号処理装置は，広範な演算に対して適応できるような柔軟性を備えていなければならない。

iii) システム構成　信号処理装置は，その応用分野によっていろいろなシステム形態を取ることが予想される。たとえば，システムの効率を上げるために他の汎用計算機とのマルチプロセッサ構成を採る場合や，単体構成で，A/D，D/Aコンバータ，ディスプレイ装置等，多種類の入出力機器を接続する場合等が考えられる。このようなシステム構成に関する種々の要求に柔軟に対処できる装置でなければならない。

今後の信号処理装置は単に高速であるだけでなく，上述の3つの要求を満たしていなければならない。

著者らが，現在開発中の高速汎用信号処理装置（ＫＳＰ：Keio Signal Processor）は，各種演算処理能力及びシステム構成の柔軟性を備え，さらに，

1024点のFFTを 約3mS，また，1024点の2チャンネルのFFTについては 約8.5mSで計算することができるものである。 本資料では KSPについて，主にそのマイクロプログラム制御を中心に報告する。

2．演算処理の高速化とパイプライン処理

演算処理を高速化するための要因としては，記憶装置のアクセス時間及びサイクル時間，演算器での演算時間，そして演算処理方式を挙げることができる。 記憶装置に関しては，できる限り高速であることが望ましい。 最近，各種の高速半導体記憶装置が開発されているが，高速のものは，まだ高価である。 またFFTの演算等においては，順々にアクセスされる記憶番地に局所性がなく，バッファリング等の技術が簡単に応用できず，容量の大きな記憶装置全体を高速化しなければならず高価になってしまう。

FFTやIFFT（Inverse FFT）は，複素数 A，B，W に対し，基本的には，A＋B，（A＋B）×Wの演算を，繰返し行うことにより行なわれる［図1］。すなわち，複素加算及び複素乗算からなる演算器を高速化する必要があるわけだが，実際，このような演算をある程度以上高速化することは，極めて困難である。 FFTをはじめとする信号処理演算では，一般に規則正しく並んだ大量のデータに対し同一の演算を施し，全体に対しての処理時間を短くすることが重要となる。このような場合には，パイプライン方式が極めて有効である。

KSPにおいては，記憶装置を3個の独立した記憶バンクに分割し，これらを同時にアクセスすることにより高速性を得た。 また，演算を継承的な演算要素での演算時間が，記憶装置の読みだし及び書き込みに要する時間に等しくなるようにし，演算器と記憶装置を連結したパイプラインを構成し，さらに高速化を図った。このようなパイプライン処理方式を採用することにより，KSPでは，比較的安価に高速性を得ることが可能となった。

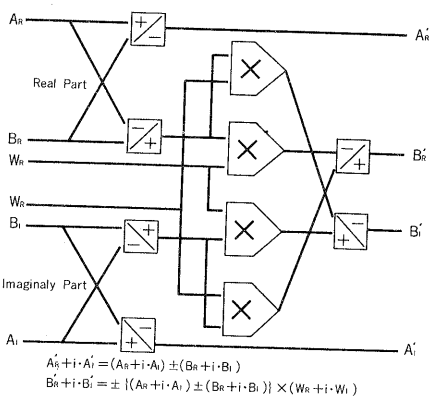

Fig. 1. Radix 2 complex arithmetic unit (CAU)

3．演算処理の汎用化とKSPの構成

信号処理装置の演算処理に対する汎用化は，i)演算器の汎用化，ii)演算要素の任意接続による汎用化，iii)任意の演算器の導入による汎用化 の3点から考えることができる。

i)の方法による演算処理の汎用化は，第1章に述べたとうり，今後いかなる演算処理が要求されるかを予測することが困難なこと，及び，信号処理装置は高速性をめざすものであるという理由から，極めて実現性が乏しいと思われる。

ii)の演算要素の任意接続による汎用化とは，数種類の基本演算要素を適当な個数だけ備えて，これを適当に接続して目的とする演算機能を持ったパイプラインを構成する方法で，大型計算機等の高速化に用いられている方法である［6］。ii)の方法は，i)の方法と同様な理由からどのような基本演算器をいくつ

備えるべきかを決定するのが困難であり，また演算要素の接続のためのスイッチが莫大なものとなり，複雑さが増し高価になってしまう。

(iii)の方法は，任意の演算器に対してその入出力インターフェイスと制御に関する仕様を定め，各種演算器に対して統一化することにより実現可能である。この方法は，その仕様に従うかぎり，どんな演算器でも信号処理装置の中に組込むことができ，演算器内においてもかなり自由に最適化が行えるという利点があり，最も汎用性を備えているということができる。

KSPは，(iii)の方法を採用している。すなわち，図2に示すようなバス構造を採り，これらのバスと演算器間のインターフェイス仕様を定め，使用目的に合致した演算器をバスに接続できるようにした。標準演算器として，FFT演算器（FFTU），複素数用算術論理演算器（DALU），制御用算術論理演算器（CALU）が用意されている。またこのほかに，これらの演算器をはじめ今後接続されるであろう各種演算器に対して，それぞれの処理に対応した記憶番地パタンを発生するために，強力な番地生成ユニット（AGU）があり，KSPにおける演算処理の汎用化に大きな役割を果している。さらに，計算機間通信接続器（CCC）や，入出力インターフェイス・ユニット（INTFU）を接続することにより，他の計算機との接続や，入出力機器の制御が可能となり，マルチプロセッサ・システムや単独システム等，各種のシステム構成をとることが可能となっている。

4．パイプライン処理方式とマイクロプログラム制御

任意の演算器の導入を許すような open-ended な装置においては，広範な演算器に対し適応性を持った制御方式を採用しなければならない。また，制御の統一化や，実装の容易さ，価格の点からも，このような装置の制御には，マイクロプログラム制御方式が適しているといえる。

一般にマイクロプログラム制御方式においては，制御に柔軟性を持たせるための方法として，マイクロ命令の情報ビット数を多くするというやり方がある。しかしながら，このような方法は，KSPのように演算処理等の汎用化とともに，高速性を重視する装置には，必ずしも適当ではない場合がある。

一方，パイプライン処理では，それぞれの演算要素が，次々と入力されるデータに対し，同一の演算を行うという特徴を持っている。従って，これを

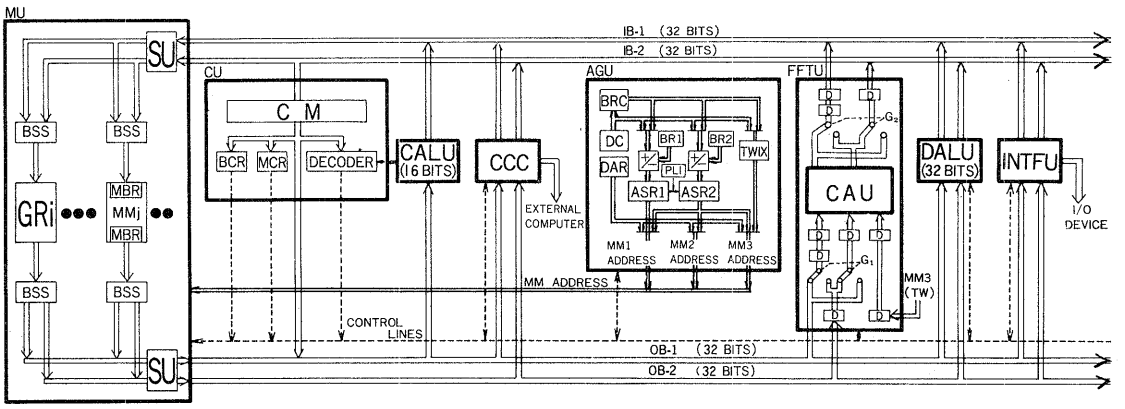

Fig. 2. CONFIGURATION OF THE PROCESSOR.

制御という観点から見ると，1つのデータ集合に対する処理において，その処理の間中不変な制御と，その途中において頻繁に変更しなければならないような制御に分けることができる。すなわち，処理の間中一定のものとしては，i)演算器と記憶装置を結ぶデータ・バス，ii)データの表現形式，iii)演算器で実行する演算の種類，iv)記憶番地生成のパターン 等が挙げられる。また，処理中に頻繁に変えるものとして，v)記憶番地の更新，vi)記憶装置の読出し書込み，vii)データ転送のタイミング，viii)状態のセンスと命令実行シーケンスの決定 等がある。ここで前者を静的制御（static control），後者を動的制御(dynamic control)と呼ぶことにする。このように制御が静的なものと動的なものに分けられるとき，レシデュアル制御技術（residual control technique）を適用することができる。すなわち，処理装置の構成等を指定する制御用レジスタを設けて，これに静的制御のための情報をデータとして転送し，構成を設定し，その後，動的制御をそのようにして設定された論理的構造に対して行うという技術である［7］［8］。

パイプライン処理の場合，制御用レジスタへのパラメータの設定は，1つの演算処理に先立って，1度だけ行えばよく，このための演算処理速度の低下は殆んどない。また，動的制御を演算処理の種類から大部分独立にすることができ，このためマイクロ命令ビット幅を小さくすることが可能となり，命令の実行を高速化できる。また，各種演算に対し広範な融通性を備えることができる。

パイプライン処理の実行の過程は，図3に示すような3つの段階から成る。まず第1は，記憶装置からデータを読出し，演算器に送り込んでいるが，まだ最後の演算要素までデータが満たされず，記憶装置へのデータ書込みは行なわれない段階である。第2は，すべての演算要素にデータがつまっており，記憶装置ではデータの読出しと書き込みの両方が行われている段階である。第3は記憶装置からのデータの読出しは終うし，演算要素中に残されているデータを処理しながら記憶装置へ書き込むという段階である。これらの段階をそれぞれ pre-process，main-process，post-process と呼ぶことにする。これら3つの段階では，記憶装置の読出し，書き込みに関する制御，及び終了条件が異るだけであり，その終了条件は，AGUが与えてくれるので，各段階とも，小さな動的制御のループにより実行できる。このために，パイプライン処理全体を制御するマイクロプログラムのステップ数も極めて小さくすることができる。

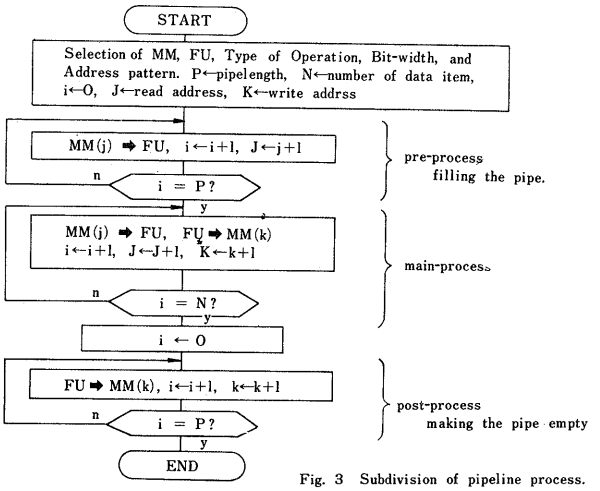

Fig. 3 Subdivision of pipeline process.

以上述べたように，制御の適応性を幅広いものにするという要請から採用されたマイクロプログラム制御方式は，パイプライン処理に対してはレシデュアル制御技術が適用できるために，マイクロ命令

のビット幅を小さくすることができ，マイクロプログラムのステップ数も少くて済むという利点を生む。このため制御記憶の容量も小さくすることができ，高速な制御記憶の実装も容易となる。KSPでは，このような制御方式を採用している。

5. 1語データの演算と割込処理

第4章においては，1つのデータ集合に対し，1種類のパイプライン演算処理を行う場合について述べたが，このような演算をいくつかのデータ集合に対して行う場合や，実行する演算処理の種類がいくつもある場合には，その処理の途中で中間結果や各種の状態を判別する必要が生じる。そして，このような場合には，独立した1語づつのデータを取扱うことが多い。しかしながら，パイプライン演算器は，このような独立した1語のデータの処理には，全く不適当である。すなわち，実行する演算処理自体と比較して，このために行う初期設定等に要する手続きの方がはるかに複雑となり，全体としてみるとオーバヘッドが無視できない程大きくなってしまう。

また，他の計算機との交信や入出力装置の制御のためには，割込みに対し高速で応答する必要がある。そして，このような制御を行うためには，1語づつ独立したデータの演算処理を頻繁に行うことになる。もし，割込みにより起動されるマイクロルーチンが，1語づつのデータの演算のためにパイプライン演算器を用いるとすると，割込み時の状態を退避しなければならず，このためには非常に複雑な操作が必要となる。これを避けるためには，割込の受付を，一つのパイプライン演算終了まで待たせるという方法があるが，この方法では，割込みへの高速応答が不可能となってしまう。

これらの問題を避けるため，KSPでは，制御用算術論理演算器（CALU）を設けた。CALUは，独立した1語づつのデータの演算を行うことを目的とし，その内部状態は4ビットしかなく，パイプライン演算器と異り，初期設定を行わず用いることができる。また演算を行う際にも，パイプライン演算器等の状態を変えることがないよう工夫されている。また，番地生成ユニット（AGU）についても，初期設定なしに記憶装置内を参照できるように，直接番地レジスタ（DAR）を設けている。

6. KSPの制御のためのハードウェア

KSPの制御を行うハードウェアは大部分が制御ユニット中に納められているが，レシデュアル制御を用いたことや，パイプライン処理の特殊性のため，それ以外の部分にも重要な制御のためのハードウェアが分散している。これらについて，制御ユニット，番地生成ユニット，演算器等に分けて説明する。

6.1 制御ユニット（CU）

CUは，32ビット／語で，4K語までのRAMを実装できる制御記憶（CM），制御記憶番地レジスタ（CMAR），プログラム・ループの作成やサブルーチン・リンケージに用いる戻り番地レジスタ（BAR），割込み時に戻り番地を拡例しておく割込番地レジスタ（IAR），バスのレシデュアル制御に用いるバス制御レジスタ（BCR），記憶装置のレシデュアル制御に用いる記憶制御レジスタ（MCR），マイクロ命令デコーダ及びマイクロ命令の実行や実行順序の決定を行う回

路，割込み受付のための回路等から成っている。CUは，マイクロ命令の実行を高速化するために，それ自身がパイプライン化されている。

レジデュアル制御のためのレジスタのうち，BCRは，入出力バスのそれぞれに対し，汎用レジスタ（GR）や，記憶バッファレジスタ（MBR）のどれを接続するか，また，どの演算器を接続するかの指定を行う。MCRは，記憶装置の各バンクごとに，アクセスの有無，ビット幅等の指定を行う。またこのほかにBCR，MBRは，バス選択スイッチ（BSS）やスキュー・ユニット（SU）の制御も行う。BSSやSUは，バス上を転送するデータのビット幅（16ビットまたは，32ビット）の制御や，32ビット・データの上位16ビットと下位16ビットを入替るといった操作を行う。

KSPでは，割込み受付のためのハードウェアが準備されており 割込みが受付られると，マイクロプログラムの制御は特定の番地へ移され，戻り番地はIARへ格納される。割込みのレベルは4段階に分けられており，これらは，i)ハードウェア・エラーおよび電源異常，ii)コンソール割込み，iii)プログラム・エラーおよびアドレス・エラー，iv)外部割込み から成っている。

6.2 番地生成ユニット（AGU）

AGUは，各種演算処理に適合する番地パタンを発生できるが，これらのうちのどのパタンを発生するかの指定や，その初期値の設定は，制御用レジスタによって行われる。AGUは，以下の3つの状態に応じて，フラグ・ビットを立てる機能を有し，これは，プログラム・ループをつくる時に利用される。

状1は パイプ長カウンタ（PLC）がゼロになった事を示す状態である。PLCには，任意の値をセットすることができ，処理したデータ数を示すデータ・カウンタ（DC）を増加させるのと同期して1づつ引かれる。この状態は，pre-process や post-process 等の小さな do-loop をマイクロプログラムで構成する場合に利用される。 状2は，DCの値が，前もってセットされたデータ・サイズ・インジケータ（DSI）の値に等しくなった事を示す状態で，これはmain-process を構成する場合に利用される。 状3の状態は，FFT またはIFFTのすべてのステージが終了した事を示すもので，この演算のみ用いられる。 このような機能によって，KSPはパイプライン処理を高速に実行することができる。

6.3 演算器

CALUを除くすべての演算器は，その演算器で行う演算を指定するための制御用レジスタと，演算器の状態を示す状態レジスタを備えている。演算器内の制御は，主にこれらのレジスタの値に従って行われる。

6.4 計算機間通信接続器（CCC）と入出力インターフェイス・ユニット（INTFU）

CCCおよびINTFUは，KSPにおいては，演算器とみなされほぼ同様に制御される。CCCは外部計算器とのデータ転送の制御を行うほかに，CMKマイクロプログラムをロードするときには，CUを制御する役目を持っている。これにより，ダイナミックにマイクロプログラムを書換えることができる。INTFUは，入出力装置とのデータ転送の制御を行い，マイクロ命令のループで行うブロック転送と，割込みによる1語づつの転送の機能を持っている。

7. KSPのマイクロ命令

KSPのマイクロ命令は1語が32ビットから成り，12種の命令が準備されている。これらの命令は，その使用目的により，パイプラインの動的制御のための命令，静的制御のための命令，1語の独立したデータ処理用の命令，状態センスや分岐を行う命令，エミュレーションのための命令に大別できる。

これらの命令の実行時間は，命令によって異り，それぞれ1ないし3サイクルの基本サイクルを持つ。KSPでは，各種の状態をセンスし条件が一致した場合を，トラップが発生したと呼んでいるが，このような場合および，その命令の実行中に割込みが発生した場合には，命令の実行時間は増加し，それぞれ基本サイクル＋1サイクル，基本サイクル＋2サイクルとなる。これは，マイクロ命令の実行がパイプライン化されているため，その実行順序を変えるとすでに先取りされている命令を捨て，新たに読みなおさねばならないためである。

マイクロ命令の実行は，通常は，現在実行中の命令の次の番地へと進んで行く。この順序が変わるのは，割込みが発生した場合および，条件判定機能のある命令ではトラップの発生した場合であるが，KSPでは，プログラム・ループを作りやすくするために，このほかに次のような機能が加えられている。すなわち，命令のコントロール・フィールド内に，戻り番地セット・ビットおよび戻り指定ビットというものがあり，戻り番地セット・ビットが立てられていると，その命令の番地＋1が，戻り番地レジスタにセットされる。そして，この命令よりいくつか後の命令で，その戻り指定ビットが立っていると，制御は次の番地へは移らず，戻り番地レジスタにセットされていた番地へと戻るようになっている。この機能とトラップの機能を組合せ用いることにより，各種のプログラム・ループが容易に構成できる。

以下，それぞれのマイクロ命令について説明する。

7.1 パイプライン処理の動的制御のための命令

パイプライン処理の実行に際して，記憶の読出し書き込み，バス上のデータ転送，演算実行のタイミング等の制御を行うための命令として，MD命令がある。

i) MD： MD命令は，記憶制御ビット，バス制御ビット，トラップ条件フィールド，トラップ飛先番地フィールド，コントロール・パルス・フィールド，コントロール・ビット・フィールドより成る。記憶制御ビットやバス制御ビットは，これらが立っていると記憶制御レジスタやバス制御レジスタの内容に従って，記憶装置の読出しや書き込み，バス上のデータ転送が行われる。トラップ条件フィールドに書かれた条件を番地生成ユニットのフラグビットが満たすと，制御はトラップ飛先番地フィールドで指定された番地へと移される。コントロール・ビット・フィールドは，戻り番地機能に関するビットのほかに，割込禁止ビット，番地更新ビット，直接番地増加ビット等がある。コントロール・パルス・フィールドは，演算器のタイミングを制御するためのもので，演算器によって意味が異る。MD命令は，KSPにおいて最も重要な命令である。

7.2 パイプライン処理の静的制御のための命令

これは，パイプライン処理の開始に先立って，初期パラメータを各種制御レ

ジスタへセットするための命令で，EMIT命令，SS命令がある。

ii) EMIT： EMIT命令は，演算器指定フィールド，演算器内制御レジスタ指定フィールド，データ・フィールド，コントロール・ビット・フィールドから成り，この命令が実行されるとデータ・フィールドの内容が指定された演算器の制御レジスタにセットされる。

iii) SS： SS命令は，演算器指定フィールド，演算器内制御レジスタ指定フィールド，ソース・レジスタ指定フィールド，およびコントロール・ビット・フィールドから成り，この命令が実行されるとソースレジスタ指定フィールドに示された汎用レジスタ又は記憶バッファレジスタの内容が，指定された演算器の制御レジスタに移される。

7.3 1語データ処理のための命令

1語づつ独立したデータを処理するための命令としては，FS，LR，SR，ALOPの4つの命令がある。

iv) FS： FS命令は，演算器指定フィールド，演算器内制御レジスタ指定フィールド，ディスティネーション・レジスタ指定フィールド，コントロール・ビット・フィールドから成り，この命令が実行されると指定された演算器の状態レジスタの内容が，ディスティネーション・レジスタ指定フィールドに示された汎用レジスタ又は記憶バッファ・レジスタに移される。

v) LR： LR命令は，記憶番地フィールド，ディスティネーション・レジスタ・フィールド，コントロール・ビット・フィールドから成り，この命令を実行すると，直接番地レジスタに記憶番地フィールドの内容がセットされ，その後に，直接番地レジスタで示された記憶番地から読出しが行われ，これがディスティネーション・レジスタに移される。

vi) SR： SR命令は，記憶番地フィールド，ソース・レジスタ・フィールド，コントロール・ビット・フィールドから成り，この命令を実行すると，直接番地レジスタに記憶番地フィールドの内容がセットされ，その後，ソース・レジスタの内容が記憶装置の直接番地レジスタで示された記憶番地へ書込まれる。

vii) ALOP： ALOP命令は，制御用算術論理演算器(CALU)を用いた演算を実行させるための命令で，2つのソース・レジスタ・フィールド，ディスティネーション・レジスタ・フィールド，演算指定フィールド，コントロールビット・フィールドから成る。この命令が実行されると，2つのソース・レジスタの内容に対して指定された演算が実行され，その演算結果に応じて，ゼロ，負，オーバ・フロー等の状態が，演算器の状態レジスタにセットされる。演算結果は，ディスティネーション・レジスタへ書き込まれる。

以上，4つの命令は，記憶制御レジスタや，バス制御レジスタを使用せず，これらの内容を変えない。

7.4 状態判断及び分岐のための命令

状態の判断及び分岐のための命令として，KSPでは，RFI，JOT，JNT，SKPの命令が準備されている。

viii) RFI： RFI命令は，命令コードのみの命令であり，この命令が実行されると，直接番地セーブ・レジスタの内容が直接番地レジスタへ復帰され，割込番地レジスタへ退避されこれに番地へと制御が移される。

ix) JOT： JOT命令は，演算器指定フィールド，トラップ条件フィールド，分岐番地フィールド，コントロール・ビット・フィールドから成り，この命令が実行されると演算器指定フィールドで示された演算器の状態レジスタとトラップ条件が比較され，条件が満たされると分岐番地フィールドに示された番地に制御が移る。

x) JNT： JNT命令はJOT命令と全く同じフィールドを持っており，これは，トラップ条件が満足されない場合に，分岐がおこる。KSPでは，命令実行がパイプライン化されているため，トラップが発生して実行順序が変わると，命令実行時間が長くなり，パイプライン演算処理時間に影響する。このため，JOT，JNTの両方の命令が準備されている。

xi) SKP： SKP命令は，演算器指定フィールド，トラップ条件フィールド，コントロール・ビット・フィールドから成り，この命令が実行されると，トラップ条件フィールドと演算器の状態レジスタの比較が行われ，その条件の満たされ具合によって，この命令の番地＋1から＋5までの，5つの番地のうちのいずれかへ分岐する。これは，入出力機器の状態判別等に用いられる。

7.5 エミュレーションのための命令

各種信号処理のために，目的に合ったマイクロプログラムを，その都度コーディングすることは，作業能率が悪いので，信号処理向きの高級言語を準備しておくことが望ましい。このような高級言語をマイクロプログラムでエミュレートする場合の便宜のために次の命令が準備されている。

xii) AM： AM命令は，記憶番地フィールド，記憶装置の読み書き指定ビット，コントロール・ビット・フィールドから成り，この命令が実行されると直接番地レジスタに記憶番地フィールドの内容がセットされ，その後に，記憶装置の直接番地レジスタの示す番地の内容を記憶バッファ・レジスタへ読みだしたり，又，逆に，記憶バッファ・レジスタの内容を記憶番地へ書き込まれたりする。

8. おわりに

各種の振動解析やパタン情報処理の分野においてみられる信号解析技術の進歩はめざましいものがあり，これに伴って，このために用いられる道具も，より高級なものが要求されるようになった。これらの主なものとして，機械振動や音声信号等の実時間処理のために必要とされる高速性と，図形情報等の多次元データを扱うための大量データ処理能力の要求を挙げることができる。そしてこれらの要求は，しばしば，通常の汎用計算機の処理能力を越えるものとなっている。このために，近年専用ハードウェア・プロセッサが，いくつか開発され，市販されるようになった。一昔前までは，このような専用プロセッサの開発は，非常に高価なものとなり，軍用等の特殊用途以外では，一般的な道具として用いるには無理があった。しかしながら，近年における，ディジタル計算機向きの処理アルゴリズムの発達，計算機アーキテクチャの研究成果の蓄積，及び各種半導体素子の高速化，高集積化は，このようなプロセッサの開発コス

トを大幅に低減させ，現在ではその価格は実用的な道具として十分適用するまでになりつつある。

本論文で述べたKSPは，このような各種技術の蓄積の上に立ち，高速性と汎用性を主眼として設計したものである。特にパイプライン処理の高速性とマイクロプログラム制御の柔軟性，汎用性という両者の長所を融合させ，専用プロセッサにありがちな汎用性，拡張性の欠如という欠点を補っているという点は，最も工夫した点であった。高速性という点に関しては，依然として半導体素子，特に記憶素子の速度が支配的であり，改善の余地を残している。又，このような高速プロセッサを使い易くするという点から，信号処理向高級言語等の研究開発は重要であり，ハードウェアの特長をナニ分に生かすソフトウェアの存在は，今後の専用プロセッサにとって必要不可欠のものとなると思われる。

謝辞

KSPは，現在，タケダ理研(株)において，具体化が進められている。設計並びにその実現に際し御協力下さった武田社長はじめ，開発に御尽力戴いた研究者の皆様に深謝致します。また御検討戴いた相磯研究室の皆様にも，お礼申し上げます。

参考文献

1. J.W.Cooly and J.W.Tukey:"An algorithm for the machine calculation of Complex Fourier series",Mathematics of Computation,9,p.297,1965

2. W.T.Cochran,et al.:"What is the fast fourier transform ?",IEEE Trans.,AU-15,2, p.48,1967

3. R.R.Shively: "The time saver: A digital processor to generate spectra in real time",IEEE Trans.,C-17,5,p.485,1968

4. H.L.Groginsky and G.A.Works: "A pipeline fast Fourier transform",IEEE Trans., C-19,11,p.1015,1970

5. H.Aiso,M.Tokoro,S.Uchida,H.Mori,N.Kaneko,M.Shimada: "A very high-speed micro-programmable pipeline signal processor",IFIP Congress '74,1974

6. C.J.Purcell: "The Control Data STAR System---- Principle of Operation", Control Data Corporation,Dec.,1968

7. H.W.Lawson Jr. and B.K.Smith: "Functional characteristics of a Multilingual Processor", IEEE Trans.,C-20,7,1971

8. V.R.Lessor: "An Introduction to the Direct Emulation of Control Structure", IEEE Trans.,C-20,7,p.751,1971

9. 所，内田，森，金子，島田，相磯："パイプライン計算機とマイクロプログラム"，ダイナミックマイクロプログラミングシンポジウム報告集，P.111～117，情報処理学会プログラミングシンポジウム委員会，1973年7月

10. 相磯，他："高速汎用信号処理装置（KSP）の概要"，昭和49年度信学会大会，1974年7月

11. 内田，所，金子："KSPの制御"，昭和49年度信学会大会，1974年4月

12. 所，森，島田，金子："KSPのFFT演算装置(FFTU)"，昭和49年度信学会大会，1974年7月

13. 金子，所，内田："KSPのアドレス発生機構"，昭和49年度信学会大会，1974年7月

14. 内田，馬場："KSPのための計算機間通信接続装置(CCC)"昭49年信学会大会，1974年7月

15. 森，島田，金子，所，内田，相磯："KSPのシュミレータ"昭和49年度信学会大会，1974年7月