ソフトウェア工学　52 — 3
（１９８７・２・１３）

ディジタル信号処理システムの開発環境の設計

稲村　浩[†]、山田　広司[†]、中野　秀男[††]

カノープス電子株式会社[†]　　大阪大学工学部通信工科[††]

本報告では、エンジニアリングワークステーションあるいは上位パーソナルコンピュータの持つハードウェアの能力とソフトウェア環境に注目し、これらをハードウェアの核としたディジタル信号処理システムを、使いやすさ、プロトタイピングの容易さ、処理の高速さを目標として設計する。特にそのシステムのソフトウェアの核となる信号処理用言語については、信号処理用の分野で多用される基本演算を言語仕様に含め、またデータオリエンテッドな演算、すなわち配列やデータの型に対して効率的に取り扱える演算を採用しており、ＦＯＲＴＨインタプリタを言語仕様のベースとして採用していることも合せて、使いやすく高速で、しかもソフトウェア開発の行いやすいものとなっている。

THE DESIGN OF A DIGITAL SIGNAL PROCESSING SYSTEM

Hiroshi INAMURA[†], Hiroshi YAMADA[†] and Hideo NAKANO[††]

[†]Canopus Electronics Co., Ltd., 1-4-30, Nishiokamoto, Higashinada, Kobe, 658 Japan
[††]Faculty of Engineering, Osaka University, 2-1, Yamadaoka, Suita, Osaka, 565 Japan

In this paper, we present the design of a digital signal processing system which is capable of fast processing, easy prototyping and which has a high degree of user friendliness.

The system is designed for use on engineering work station and personal computer which has high quality hardware and software environments. The software of the system is based on a signal processing language which allows most of the basic operations necessary in signal processing, to be peformed in the format of a language. Also, calculations are data oriented, i.e. manipulation of data stored in arrays and calculation involving various data types can also be performed effectively. The signal processing language, which is based on the FORTH, is easy to use, has high performance and provides a way for quick software development.

1　はじめに

近年、ソフトウェア開発をともなうシステムの構築がますます増価しており、それに対する開発環境の必要性が認識されてきている。ソフトウェアの作成が中心となるシステムでは、ＵＮＩＸや最近ではＳｍａｌｌｔａｌｋ－８０のような優れた環境があり、また、それに対する理論的なアプローチとしても、従来からあるＷａｔｅｒ　Ｆａｌｌ型の開発プロセス・モデルだけでなく、プロトタイピング手法や、ソフトウェアの部品化、再利用、オブジェクト指向やルール指向の考え方など新しい概念が提案され実現されてきている。

このような考え方は、どのようなシステムの構築にも使える有用な広い概念であるが、現在のところまだ対象がソフトウェアの専門家に限られており、応用分野においては使用する者の技術レベルやハードウェア環境など現実とのギャップが大きく、一般的なものとなってはいない。しかし、こうした概念は対象とする応用分野やさらにシステムを限定し、そのシステムの利用者やそのシステムを使って開発を行う者の要求を洗い出すことによって実現が可能であり、そうして作成されたシステムはその分野での優れたシステムとなると思われる。本報告では、そのようなシステムとしてディジタル信号処理システムを取り上げる。

ディジタル信号処理の分野では、応用分野の急激な拡大とニーズの多様化に応えてさまざまな計測器や処理解析装置が開発されてきているが、さらにそれらの装置とコンピュータを組合せたシステムも多く使用されている。しかし、ディジタル信号処理技術が一般的な技術になってきたとはいえ、まだなお高度な技術的な内容を含んでおり、コンピュータを使用したシステムでは、特にソフトウェア作成の点でそれを使用する平均的な技術者には負担が大きく、また、この分野の技術を提供する側の者にとっても、ソフトウェア技術の水準やマンパワーの点で問題が多い。そこで、エンジニアリングワークステーションと呼ばれるコンピュータ、あるいは上位パーソナルコンピュータがパーソナルユースとしては十分な処理能力と上質なソフトウェア環境を持っており、これらをハードウェアの核としてディジタル信号処理システムあるいはその開発環境が実現できると考え、使いやすさ、プロトタイピングの容易さ、処理の高速性を目標として設計を行う。

まず第２章では、ディジタル信号処理の現状について説明し、ディジタル信号処理システムに対する要求をシステムへのかかわりかたの異なる３つのタイプの技術者や研究者の立場から考察し、問題点を指摘する。第３章では、それまでの要求や問題点に対する１つの答として、エンジニアリングワークステーションや上位パーソナルコンピュータを核としたディジタル信号処理システムを提案する。このシステムのソフトウェアの核である信号処理用言語（ＳＡＬ）の設計などについては第４章で述べる。

2　ディジタル信号処理システムへの要求

ディジタル信号処理とは、電圧や長さなどのアナログ量をディジタル化し、数値化することによってディジタル量として処理して二次的な情報を得ることである。この分野の技術は、数年前までは高度かつ特殊な技術分野であったが、ハードウェア、ソフトウェア両面の技術的進歩と応用分野の開拓により、一般的でしかも安価な技術となってきている。そして、それらを使用し応用するために計測器や解析装置が数多く開発され使用されていが、応用分野は急速に拡大しており、高度なあるいは複雑な応用分野には、通常の計測器だけでは要求をみたすことができないため、コンピュータによって処理を支援する傾向がみられる。しかし、こうした計測器とコンピュータを組合せたシステムではソフトウェアの開発が必須であり、またこの点で現在のシステムは使用する者の要求を満足するものとなってはいない。

ディジタル信号処理（以下単に信号処理）を行うシステムにはさまざまな要求があり、ここではそれらの要求を整理し、計測器とコンピュータを組合せた信号処理システムの問題点を調べてみる。

まず、信号処理に携わる技術者やシステムのユーザを、その目的や信号処理技術とのかかわりあいについて分類してみると、次のように分類できる。

タイプＡ－現場で信号処理を行う技術者
タイプＢ－信号処理システムを開発する技術者
タイプＣ－信号処理あるいはそれが不可欠な分野の研究者

タイプＡの技術者は、処理すべきデータを持っており、それらを処理、解析して新たな情報を得ることを目的としている。このタイプの技術者は必ずしもソフトウェア技術に精通しておらず、計測器とコンピュータを使用したシステムではソフトウェア作成の点で負担が大きい。しかも、現場で得られたデータ迅速に対処することが必要であるために、結果が直接的に得られる専用化されたシステムを使用している。しかし、要求の変化に柔軟に対応で

きないシステムに不満を持っており、要求に柔軟に対応でき、また自分たちでソフトウェアを作成できるような理解しやすく使いやすい、しかも大きな能力を持ったシステムを望んでいる。

タイプBの技術者は、タイプAの技術者やタイプCの研究者のために、信号処理、解析を行うシステムを開発し提供することを目的としている。このタイプの技術者はソフトウェア技術、信号処理技術の両面に精通しており、コンピュータを使用したシステムを十分に活用する能力を持っている。彼らは、与えられた仕様を満足するシステムを効率的に開発することが必要であるが、拡大する信号処理システムへの要求に対応するにはマンパワーの不足が深刻な問題となっているため、より効率的な開発手段を求めている。

タイプCの研究者は、タイプA、タイプBの技術者のための新たな技術を提供することを目的としている。このタイプの研究者は、信号処理技術には特に精通しており、さらにソフトウェア技術にも精通していることが多い。彼らの作業では、実験によって得られたデータをさまざまな側面から解析し、結果を整理し、新たな実験を行うことに多くの労力が費やされており、種々の実験が効率的に行え、解析能力の高いシステムを望んでいる。

これらの信号処理技術者の要求に応えるために、限定された機能と能力を持つ計測器と一般的なコンピュータを組合せたシステムが使用されてきたが、そうしたシステムには以下のような問題点がある。

（1）現場の技術者や研究者の要求の変化に簡単には対応できない。

コンピュータを使用したシステムでは、要求の変化に対応するにはソフトウェアに関する知識が不可欠であり、また手間も多くかかる。そのため、タイプAの技術者には技術的な困難さがある。タイプBの技術者にはマンパワーの点で個々の技術者の要求に応えることはできず、タイプCの研究者には作業効率の悪いシステムとなっている。

（2）使いにくい。

タイプAの技術者はソフトウェア技術がなくても使いこなせるシステムを望んでおり、タイプCの研究者は大量のプログラムを書かなくて実験的な作業が簡単に行えるような柔軟性のあるシステムを望んでいる。

（3）ソフトウェアの開発効率が低い。

システムのソフトウェア依存度が高いにもかかわらず、記述言語やコンピュータのソフトウェア環境が信号処理に向いていないため、ソフトウェアの生産性が低く、開発効率が上がらない。また、処理を行うためのプログラムの量が多いために、プロトタイピングの手法が使いにくく、実験的なプログラムも作成しにくい。

（4）コンピュータの能力が十分でない。

使用されているコンピュータは特別な場合を除いては、比較的安価なパーソナルコンピュータであり、大量のデータを高速に処理することが要求されることの多い信号処理の分野での用途には能力が十分ではない。

3　エンジニアリングワークステーションや上位パーソナルコンピュータを核とした信号処理システム

上で述べた要求に応え、問題点を解決するための信号処理システムとして、計測器や処理解析装置をコンピュータによって支援するのではなく、十分な能力をもったコンピュータによる処理を前提とし、その周辺装置として目的に応じた計測器を接続したシステム（図1）が、要求に対する柔軟さやデータ処理能力の点で望ましいと考えられる。こうしたシステムではハードウェアの核となるコンピュータの選択が重要であるが、エンジニアリングワークステーション（EWS）や上位パーソナルコンピュータ（UPC）が、以下のような理由で現実的な選択であると考えている。

- 高性能なCPUを搭載しており、データを高速に処理する能力を持つ。
- 内部および外部に大容量記憶をもっており大量のデータを効率よく処理できる。
- 高解像度グラフィックディスプレイを持っており、解析結果をビジュアルに表示することができる。
- 各種ユーティリティなど上質なソフトウェア環境を持っている。
- マルチウィンドウやマウス、豊富な周辺機器など上質なユーザインタフェースを持っている。
- パーソナルユースが可能なサイズとコストである。
- A／D変換器、D／A変換器、各種計測器とのインタフェースが容易である。

こうしたシステムは環境としてのハードウェアや基本ソフトウェアの点では十分な能力を持っているが、このままでは特にソフトウェアの点で信号処理向きではなく、要求に対して柔軟で使いやすくしかも開発効率の高い信号処理向きのソフトウェア環境が必要である。そのためのアプローチとして、個々の応用分野ごとに別々のソフトウェアを用意するのではなく、高度なソフトウェア技術がなくても使用でき、あるいは現場でも要求に合せてソフト

ウェアが簡単に作成でき、さらに、より高度な要求にも対応できるような、信号処理記述の能力の高い信号処理用言語を用意することを考える。

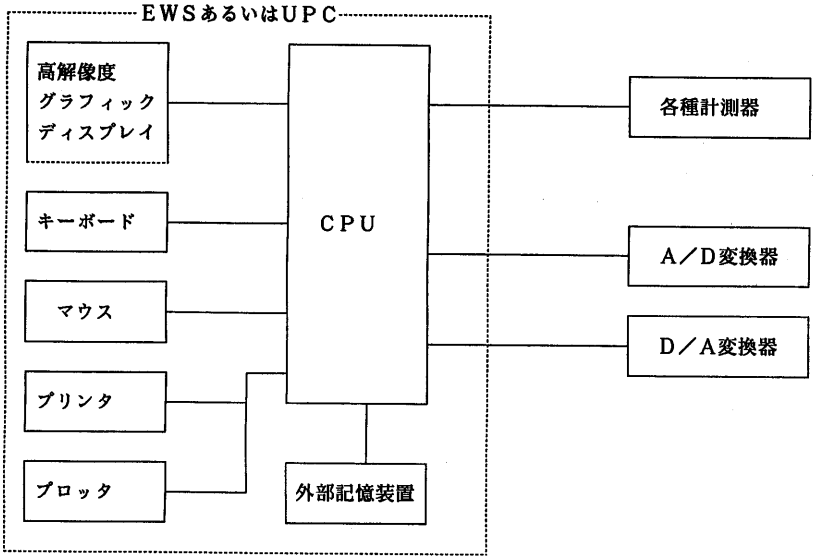

図1　EWSやUPCを核とした信号処理システム

4　信号処理用言語の設計

ここでの信号処理用言語は信号処理、特に信号解析を主な目的とし、信号処理の分野で多用される処理（たとえば高速フーリエ変換やフィルタリング、データのグラフィック表示）を言語仕様に含めることで、そのためのプログラムを作成することなく使用でき、しかも高速に処理できることを狙った言語である。この言語を設計するにあたって次の設計方針を設定した。

（1）ソフトウェアに関する知識が十分でなくても使用できること
（2）ソフトウェア開発の効率が高いこと
（3）ソフトウェアの実行効率が高いこと
（4）高度なマンマシンインタフェースをもっていること

（1）については、ソフトウェア技術に必ずしも精通していなくても使用できる、あるいはソフトウェアの開発が可能である、ことがシステムを使用する側とシステムを開発する側の両方からの要求である。そのためには、プログラムを書かなくても使用できるように簡易言語あるいはコマンドの集まりとしての一面を持っていることが望ましい。（2）については、ソフトウェアの生産性に関することであり、開発者側、研究者側からの要求である。言語の記述能力が高いこと、プロトタイピングが容易であることが重要である。そのためには、信号処理の分野で使用される基本演算はすべて用意されていることとともに、汎用プログラム言語として機能も必要である。（3）については、大量のデータを扱うことが多いため、処理速度が速いことが重要である。（4）については、マンマシンインタフェースが使いやすさに大きな影響があり、EWS、UPCの持つマルチウィンドウなどの上質なユーザインタフェースを前提として言語を設計することで実現できると考えている。以下ではこの設計方針に従って設計した信号処理用言語SAL（Signal Analyzing Language）の言語仕様に関する特徴を述べる。

4－1　FORTHを言語仕様のベースとして採用

SALはFORTHを言語仕様のベースとしたインタプリタとして設計されている。FORTHは必ずしも一般的な言語とはいえないが、その簡潔な表現と高速さには定評がある。プログラミング言語としてのFORTHの受

入れにくさの原因として考えられることに、逆ポーランド記法がある。ＦＯＲＴＨでの演算や処理の単位はワードと呼ばれ、ワード処理はスタックにプッシュされたデータに対して作用し、結果もスタック上に置かれる。そのため、オペレータはオペランドの後ろに置かれる。たとえば、ＡとＢの和にＣを乗ずる、は、

```
Ａ Ｂ ＋ Ｃ ＊
```

と表現され、結果はスタックに置かれている。この表現は、一般的な表現である

```
（ Ａ ＋ Ｂ ） ＊ Ｃ
```

と比べると多少慣れを必要とするが、一方、信号処理の分野でよく行われる、データに対して次々に演算を施してデータを変換加工する際の表現の簡潔さに大きな利点があると考えている。たとえば、Ａ／Ｄ変換して得られた波形データを、フィルタリングし、フーリエ変換して周波数スペクトルを得て、さらにパワースペクトルに変換し、それを表示する、という例はＳＡＬでは、

```
A/D FILTER FFT POWER DRAW
```

という表現になる（実際にはいくつかのパラメータが必要である）。ここでＡ／Ｄ、ＦＩＴＥＲ、ＦＦＴ、ＰＯＷＥＲ、ＤＲＡＷはそれぞれＡ／Ｄ変換、フィルタリング、高速フーリエ変換、パワースペクトルへの変換、データの表示を行うワードであり、実行に必要な一時配列、たとえば波形データやパワースペクトルを格納する配列、は自動的に生成、消滅される。これを従来の他の言語のようにサブルーチンコールあるいは関数コールで実現すれば、

```
A/D (A)
FILTER (A, B)
FFT (B, C)
POWER (C, D)
DRAW (D)
```

のようになり、本質的には必要ではない一時配列（Ａ、Ｂ、Ｃ、Ｄ）を明示する必要があり、表現も簡潔ではない。

また、ＳＡＬの例ではワードをコマンドと考えれば単にコマンドを処理の順に並べるだけで多くの処理が記述可能である。ＳＡＬはダイレクト・モードが使えるインタプリタであるので、ワード（コマンド）を入力したらすぐに実行されるようにできる。したがって、グラマーレスな簡易言語としても使用できる。このことは、現場での技術者にとって受入れやすいだけでなく、研究者にとってもいろいろな実験がプログラムを作成せずに簡単に行えるという点で大きな利点となる。もちろん上の例はプログラムとして実行することもでき、そのほうがこの処理を繰り返し行うには便利である。

さらにＦＯＲＴＨを言語仕様のベースにしていることの利点に自己増殖性が挙げられる。ＦＯＲＴＨではワード列を新たなワードとして定義でき、このワードは他のワードとまったく同等に使用できる。上の例で、ＤＲＡＷ．ＰＯＷＥＲというワードを、

```
： DRAW. POWER FFT POWER DRAW ；
```

と定義すると、ワードＤＲＡＷ．ＰＯＷＥＲは、データをフーリエ変換して周波数スペクトルを得て、さらにパワースペクトルに変換し、それを表示する、という新たなワードとして使用できる。このように、ユーザの目的に合ったワード（コマンド）を自由にしかも簡単に作成できるため、ソフトウェアのライブラリ化、カスタマイズは容易である。また、単一の機能を持ったワードを組合せてより機能の高いワードを定義することで、ソフトウェアの部品化がおこなわれ、再利用も行いやすい。このためソフトウェアの開発効率は高く、プロトタイピングの手法も有効に行える。

なお、ＳＡＬやＦＯＲＴＨは一般のプログラミング言語と同様にループや分岐などの制御構造も具えており、より複雑なソフトウェアも作成できる。

4－2　データオリエンテッドな演算

ＳＡＬでは、信号処理分野で多用される演算や処理は、他のより一般的な演算とともにあらかじめワードとして定義しておく。これによってそのルーチンを作成しなくても信号処理プログラムを作成できる。これらのワードはすべて演算子として使用でき、フーリエ変換を行うワードは加算を行うワードと同じように式の中で使用される。また、これらのワードはデータオリエンテッドな動作をする。すなわち、演算用ワードは単純変数と配列の両方に適用できるように、あるいは整数型データと実数型データ、複素型データのいずれにも適用できるように、データには値とともに配列のサイズや数値の型をも持たせておき、ワード内で自動的に判別して処理する。これにより、たとえばある配列Ａのすべての要素を２倍して使用する場合は、

```
Ａ ２ ＊
```

とすればよい。この表記法はＡが単純変数であっても、配列であってもよく、他の演算の結果であってもよい。さらにデータの型が何であってもよい。信号処理の分野では配列データをまとめて処理することが多く、配列についての演算を行えるようにしておくことで、配列要素をアクセスするためのループをなくすることができ、記述がより簡潔になる。

データには値や配列のサイズ、数値の型とともに、データの種類（波形データなのかパワースペクトルなのか、などを表す）や単位（Ｖなのかｄなのか、などを表す）なども含めておけば、データ表示ワードでデータの種類によって適切な表示形式を自動的に選択できるようになるし、罫線に目盛をつけるときに単位も表示できるようになる。これによって、１つのデータ表示ワードですべてのデータに対応でき、ユーザからは１つのワードを覚えるだけでどのデータも表示できるようになる。このようなデータの抽象化は、ユーザの作成したワードにもそのまま適用されることになるので、単にソフトウェアの作りやすさだけでなく、ソフトウェアの部品化や再利用にも大きな効果を持つ。

4－3　高速かつ豊富な演算子

ＳＡＬでは四則演算のような基本演算の他に、信号処理向けの演算を用意する。信号処理に固有な演算子（ワード）は２０ワード程度（一部を表１に示す）であるが、信号処理や信号解析をより効率的に行うために、通常の科学技術計算で使用される各種関数（三角関数や指数関数など）、複素数演算を支援する演算、配列演算を支援する演算、統計処理を支援する演算など多数の演算子をワードとして用意する。また、グラフィックスやファイル入出力のためのワードなども用意する。

こうした豊富な演算用のワードがすでに用意されているために、ＳＡＬのプログラムは単に機能を組合せるだけのものとなる。また、これらワードは上で述べたようにデータオリエンテッドな演算子であるとともに、Ｃ言語やアセンブリ言語で記述するために高速に動作する。そのためにインタプリタ系言語でありながらコンパイラ系言語と同等の実行速度が実現でき、インタプリタの使いやすさとコンパイラの高速さを両立させている。このことは、ソフトウェア技術に精通していない技術者でも高速なプログラムを作成できることにもなる。なお、ＳＡＬには通常のプログラミング言語のように配列の要素にアクセスするためのワードも用意されているため、特殊な処理も記述できる。

4－4　マルチウィンドウなどのユーザインタフェース

最近のＥＷＳやＵＰＣは、ビットマップディスプレイを使用したマルチウィンドウをサポートしているものがほとんどである。ＳＡＬではこのマルチウィンドウ環境を前提としている。すなわち、いくつかのウィンドウを開き、複数の処理、解析結果を別々のウィンドウにグラフィック表示できるようにする。数値やプログラムリスト、ヘルプメッセージ、コマンド入力などの文字表示にも用途別のウィンドウを割当てておく。これによって、プログラムリストを表示したために処理結果がスクロールして失われることがなく、またよく参照される情報を常に表示しておくこともできる。。各ウィンドウは自由に配置できるため、見やすさはもちろん、レポートの作成しやすさにも効果は大きい。レポート作成に関しては、ディスプレイ画面上で結果を確認したうえでプログラムを変更することなくプリンタやプロッタ、あるいはファイル、周辺機器に出力できる（出力のリダイレクト）ようにしておくことで、作業効率を大きく改善できる。なお、マルチウィンドウの操作（ウィンドウをの開閉、移動）に関しては、ダイレクト・コマンド（ワードのダイレクト・モードでの実行）やプログラム内で操作可能であるだけでなく、マウ

スやキーボードなどによって直接に操作できることが、使いやすさの点で有効である。

表1　信号処理に固有なワードの例

FFT	高速フーリエ変換を行う
IFFT	高速逆フーリエ変換を行う
POWER	パワースペクトルに変換する
PHASE	位相スペクトルに変換する
FIR	FIRフィルタリングを行う
IIR	IIRフィルタリングを行う
COR	自己（相互）相関を得る
COH	コヒーレンスを得る
TRFN	伝達関数を得る
OSC	正弦波、三角波、矩形波を生成する
A／D	A／D変換を行って波形データを入力する
D／A	D／A変換を行って波形データを出力する

5　おわりに

本報告では、ディジタル信号処理の分野でのさまざまな要求を満たすシステムとして、エンジニアリングワークステーション（EWS）や上位パーソナルコンピュータ（UPC）を核とした信号処理システムを考え、このシステムのための信号処理用言語SALを設計した。我々はこのシステムが使いやすくソフトウェア開発効率も高いシステムになりえると考えており、現在SALの開発を行っている。稿末に開発中のSALの簡単な実行例を示す。この例のような簡単でしかもよく使用される応用では10行程度のプログラミングですみ、しかもユーザは画面に表示されたプログラムリストと実行結果を同時に見ながら、自分の意図する処理が行われたかを確認できる。このように、要求段階から実行までが統合された優れた開発環境ということができる。

参考文献

[1] A.V.Oppenheim, R.W.Schafer, 伊達玄訳,「ディジタル信号処理（上）（下）」,コロナ社

[2] 石田春久,「UNIX」, 共立出版

[3] 竹内郁雄,「Smalltalk」, bit, 1983年11月号～1984年1月号

[4]「32ビット高速機の登場で用途を広げるエンジニアリング・ワークステーション」, 日経エレクトロニクス, 1986年12月1日号, pp. 97～113

[5] 井上外志雄,「拡張性のある言語FORTH」, bit, 1981年12月号, pp. 20～33

付録

（例題）１００ＫＨｚのサンプリングレートで１０２４点のＡ／Ｄ変換を行って波形データを入力し、その波形とパワースペクトルを表示する。次にＦＩＲ型フィルタを通した波形とパワースペクトルを表示する。

（プログラム例）

```
: SAMPLE                        :'  ワードＳＡＭＰＬＥの定義の始まり
        0 0 500 500 1 PLANE     :'  プレーン１の座標を設定する
        500 0 1000 500 2 PLANE  :'  プレーン２の座標を設定する
        .HANNING FFT.WINDOW     :'  ＦＦＴの窓関数をハニングに設定する
        1.0E5 A/D.FREQ          :'  サンプリング周波数を１００ＫＨｚに設定する
        1024 0 A/D DUP DUP      :'  チャネル０を１０２４点Ａ／Ｄ変換する
        6 DRAW.COLOR            :'  表示色を６（黄色）に設定する
        1 SCALE&DRAW            :'  罫線と波形をプレーン１に表示する
        FFT POWER TODBV         :'  ＦＦＴを行い、パワースペクトルを求める
        2 SCALE&DRAW            :'  罫線とパワースペクトルをプレーン２に表示する
        LP.COD FIR DUP          :'  ＦＩＲ型ローパスフィルタを通す
        7 DRAW.COLOR            :'  表示色を７（白色）に設定する
        1 DRAW                  :'  波形をプレーン１に重ねて表示する
        FFT POWER TODBV         :'  ＦＦＴを行い、パワースペクトルを求める
        2 SCALE&DRAW            :'  パワースペクトルをプレーン２に重ねて表示する
;                               :'  ワードＳＡＭＰＬＥの定義の終わり
```

（実行例）