みんぱくリポジトリ
国立民族学博物館学術情報リポジトリ National Museum of Ethnolo

マイクロコンピュータ複合体上でのFORTHの開発とグラフィックスへの応用

メタデータ	言語: jpn 出版者: 公開日: 2016-03-08 キーワード (Ja): キーワード (En): 作成者: メールアドレス: 所属:
URL	http://hdl.handle.net/10502/00005939

マイクロコンピュータ複合体上でのＦＯＲＴＨの開発とグラフィックスへの応用

久保正敏（京都大学工学部）

1．はじめに

我々の研究室では，種々のマイクロプロセサ複合体を開発し，計算機アーキテクチャの構成方法に関する研究とその応用を行なっている。特に，市販のマイコンシステムを結合するのではなく，特殊な目的向きにハードウェアシステムを独自に設計開発するので，そのソフトウェアシステム開発には，既存・市販のソフトウェア開発支援ツールをそのままでは利用できないきらいがあった。

本稿では，特殊なハードウェア環境におけるソフトウェア開発手法の例としてグラフィックス向きのマイクロプロセサ複合体におけるＦＯＲＴＨシステムの独自開発の経験と，その応用としてグラフィックデザインシステムを紹介する。

2．マイクロコンピュータ複合体の構成

図1に，グラフィックス向きのマイクロプロセサ複合体のハードウェア構成を示す。本来このシステムは，ホスト計算機のカラーグラフィックターミナルとして計画され，ホストから与えられるモデルに従ってその3次元カラー画像を生成するディスプレイプロセサとして必要な機能を分解し，それらをファームウェアで実現した構成となっていて，画像演算装置の役割を果たす並列プロセサシステムの制御方式に特徴があり，その方式に基づいてG-PSYCOと名付けられている。

G-PSYCOの構成要素のうち，コントロールプロセサは核となるモニタある

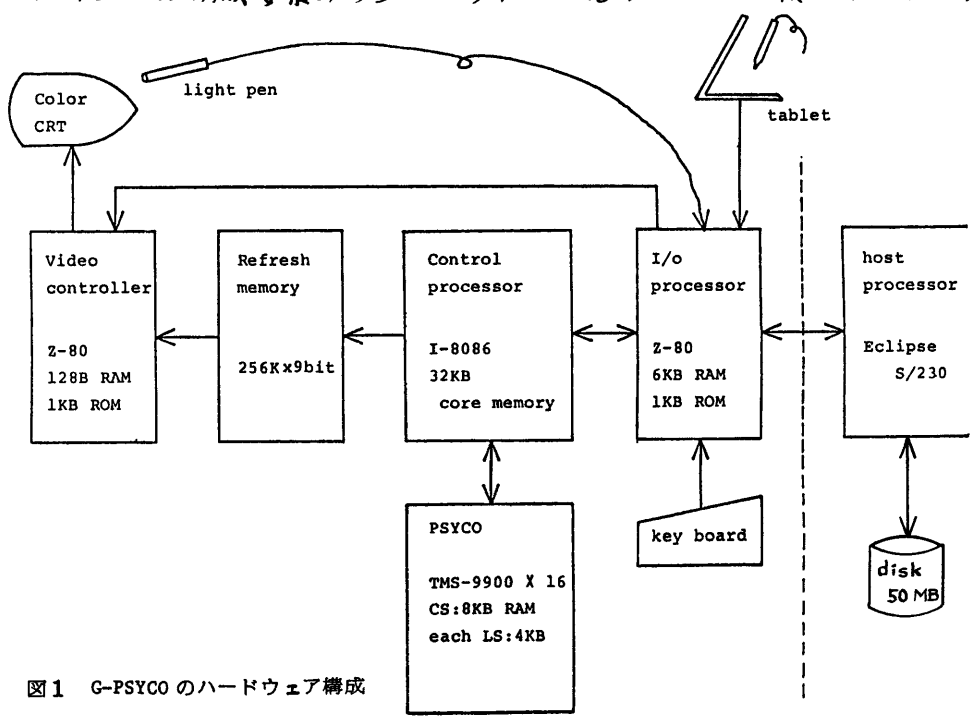


図1　G-PSYCOのハードウェア構成

いはタスクスケジューラ，並列マイクロプロセサシステム PSYCO は高速演算装置，I/O プロセサは I/O ハンドラとして機能する。

2.1 I/O プロセサ

I/O プロセサは，マイクロプロセサ（Z-80，6KB RAM，1KB ROM）で構成され，タブレット，キーボード，ライトペン等が接続されていて

- G-PSYCO とホスト計算機 Eclipse S/230 との通信制御
- 複数 I/O 機器の管理

を行なう。I/O プロセサのメモリ空間の一部 4KB 分は，コントロールプロセサと共有されており，コントロールプロセサから見れば I/O プロセサはマルチプレクサチャネルの役割を果たす。共有メモリは，コマンド及びチャネルステータスワードの授受のためのメールボックスと，データ授受のためのバッファとして用いられる。

2.2 コントロールプロセサ

コントロールプロセサは，ホスト計算機から与えられる画像モデル，I/O 機器から与えられるユーザコマンドに従い，必要な画像演算を演算装置を用いて実行し，演算結果を編集して表示画像データ，即ちディスプレイファイルを画像メモリ上に作成する。この画像演算装置が並列マイクロプロセサシステム PSYCO である。

この様に，コントロールプロセサは，画像生成に必要なタスクを並列処理可能な子タスクに分解して PSYCO に分配し起動をかけ，それらが終了すると新たなタスクを分配するというタスクスケジューラの機能，子タスクが生成したデータを編集するデータマネージメントの機能を持つ。

コントロールプロセサは，PSYCO とのデータ授受，画像メモリへのデータ書込みを行なうので，データ転送のオーバヘッドを少なくするために，我々は，コントロールプロセサが PSYCO のメモリ及び画像メモリを自分のメモリ空間として直接アクセスできる構成を採用している。即ち，コントロールプロセサは，PSYCO 及びビデオコントローラとメモリを共有し，また前述の様に I/O プロセサとも I/O バッファを共有している。従って，大きなメモリ空間を直接アクセスする能力が要求されるので，コントロールプロセサには Intel 8086 を用いている。

2.3 ビデオコントローラ

画像メモリを順次読出し，D/A 変換後カラー CRT へ送出し，画像を表示する。CRT は 20 インチ高解像度カラーモニタ TV である。画像メモリには 256K×9bit のダイナミックメモリを用い，1 画素には R,G,B 各 3 bit を充て，512×512 の解像度を得ている。画像リフレッシュは CRTC チップを用いて行なっている。

グラフィック表示に文字を重畳表示するため，画像メモリの他に 2KB のキャラクタ格納用 RAM とキャラクタジェネレータを置き，この制御のためにマイクロプロセサ（Z-80，128B RAM，1KB ROM）を用いている。

2.4 PSYCO

PSYCO は，SIMD 型処理向きに構成された共通バス方式の並列マイクロプロセサシステムである。SI を共有メモリに置き，共通バスで結合したもので，各プロセサのローカルメモリに MD が置かれる。共通バス上で命令フェッチが衝突しない様に，SI は DIF（Data Independent Flow）プログラムと呼ぶ独特の形に変換されて共有メモリに格納されているが，本稿ではその詳細を省略する。

3. FORTHの特徴とその利用

前述した通り，G-PSYCO システムの中心となるのはコントロールプロセサであり，各種応用ソフトウェアを開発する際に，当初は当研究室でホスト計算機 Eclipse S/230 上に開発したI-8086のクロスアセンブラを利用していた。しかし，G-PSYCO の各サブシステムを統括監視しながらソフトウェア開発を行なうのには不便であった。すなわち，特殊なハードウェア環境に対処できるアセンブリ言語の機能を持ちながら，複雑高度なソフトウェア開発を見やすく行なうための高級言語機能も合せ持ち，また各サブシステムのソフトウェア機能を会話的にデバッグしながら開発する過程に適したソフトウェアシステムが必要であった。

我々は，これらの要求を満すものとして，FORTHシステムを採用し，I-8086上に独自に開発することにした。

3.1 FORTHの特徴

1) Indirect Threaded Code

FORTHシステムにおいては，プログラムはすべて仮想計算機の持つ1つのプログラムカウンタに従って実行される。この仮想プログラムカウンタは，順次実行される手続きを間接的に指している。図2に示す様に，ある手続きはそれを構成する手続きのアドレスから成り，下位の手続きはそれを構成する更に下位のアドレスから成る，という多重連鎖になっている。このアドレス連鎖の末端が，実行可能な機械命令である。

FORTHでは，この様な手続きの1つ1つをワードと呼び，ワードの集まりをディクショナリと呼んでいる。ディクショナリは必要に応じていくつかのボキャブラリに分割できる。

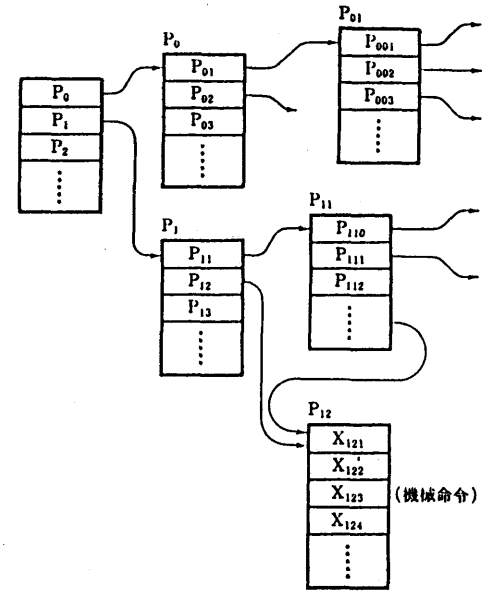


図2　間接連鎖によるプログラム

ワードは具体的には図3に示す構成となっており，ディクショナリとして登録・検索する際に必要なヘッダ領域と，そのワードを構成するサブワードのアドレスの並びから成る本体とに分けられる。

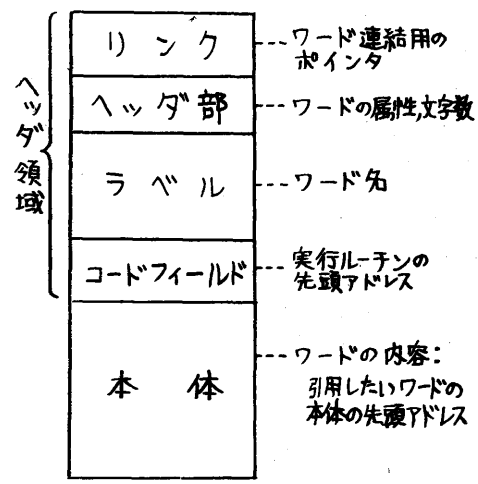


図3　ワードの内部構造

2) スタックの使用

FORTHシステムでは，リターン・スタックとパラメータ・スタックの2つを用いてプログラムを実行する。プログラマは，通常，パラメータ・スタックのみを意識する。演算対象はすべてパラメータ・スタックのトップに置かれているものとして，演算を記述する。即ち，FORTHは，逆ポーランド記法に従うプログラミング言語である。このことによって，プログラマは常にスタック操作を意識してプ

ログラム記述しなければならないが，その反面 当然コンパイラが簡単になる。また，演算は主にスタックを使用するために変数の使用が減少し，メモリ必要量が減少することになる。

3）対話型

ソーステキストは端末から入力されると即時にコンパイルされ，即実行させることができる。既に定義された手続き，即ちワードを用いてのみ，新たなワードを定義し，コンパイル，実行を行なうので，完全なボトムアッププログラミングの形態である。従って，種々の基本機能を組み上げ，実行チェックし，試行錯誤を繰返しながら進行するソフトウェア開発に適している。簡単な例として，スタックトップの値が奇数かどうかを判定するワード ODD? の定義と実行の様子を示す。

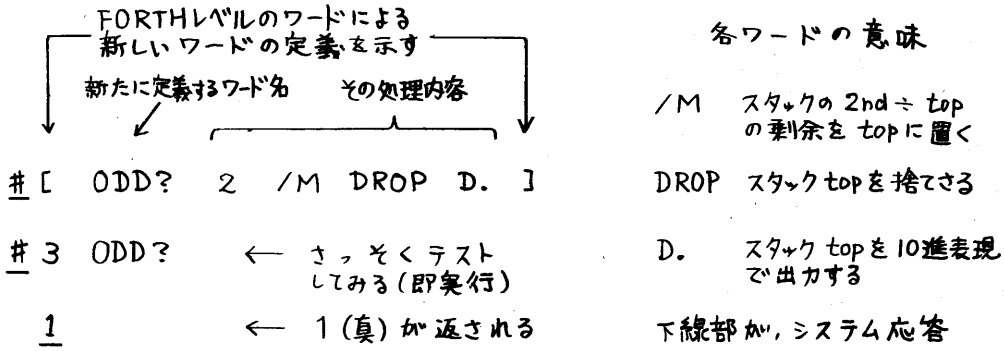


図4 対話型ワード定義とその実行の様子

4）拡張性

ＦＯＲＴＨシステムにおいては，システムプログラムもユーザプログラムも，すべて同じ構造のワードで構成され，何ら差異はない。また，新たにディクショナリに付加されたワードとそれ以前に登録されたワードとの差異もない。また，ＦＯＲＴＨのシステムプログラムの大部分はＦＯＲＴＨ自身で記述されているので，機械命令で記述された基本ワードに注意しさえすれば，移植性は極めて良い。

この様に，移植性，拡張性に優れているので，特殊なハードウェア環境におけるソフトウェアシステムの独自開発ツールとしては有効である。また，アセンブリ言語記述によってワードを定義することもできるので，高速な処理を要請される部分や，特殊なハードウェア操作等も，アセンブリ言語でプログラムし，容易にＦＯＲＴＨシステムにリンクできる。

5）構造化プログラム

プログラムフロー制御用のワードとして，ＩＦ－ＴＨＥＮ－ＥＬＳＥ，ＤＯ－ＬＯＯＰ文等が用意されているので，構造化プログラムにも対処できる。

6）長所と短所

長所としては，オブジェクトがコンパクトになる，アセンブリ言語記述とのリンクが容易，移植性・拡張性に優れる，等が挙げられる。

短所としては，逆ポーランド記法に慣れないとプログラミングしにくい，タイプレス言語である，アセンブリ言語より実行速度が劣る，等が挙げられるが，ボトムアップ的なソフト開発には適したツールであると言える。

4. G-PSYCO上でのFORTH(G-FORTH)の開発

4.1 FORTH処理系の内部構成

一般的にFORTH処理系に必要な構成要素は，図5に示す通りであるが，これらをG-PSYCOの各要素に割当てて構成し，G-FORTHと名付けている。

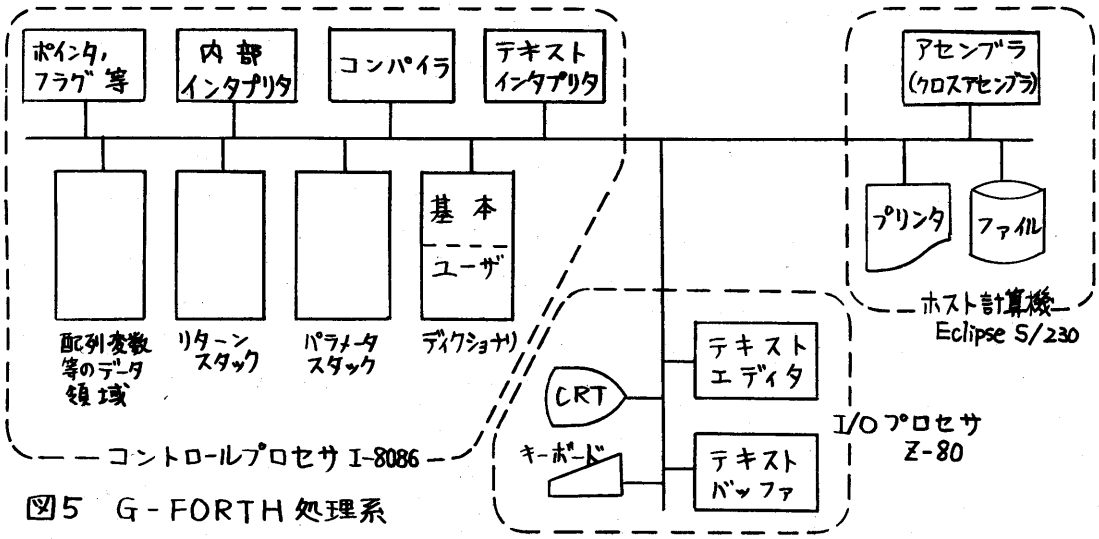


図5 G-FORTH処理系

4.2 テキストインタプリタとコンパイラ

FORTH処理系の中心となるテキストインタプリタ(外部インタプリタ)とコンパイラのアルゴリズムを図6，図7に示す。処理系は常にテキストインタプリタモードにあり，ユーザのテキスト入力を待つ。

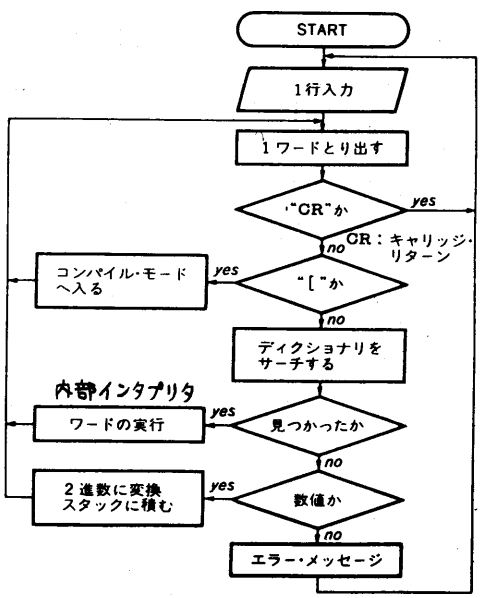


図6 テキストインタプリタのアルゴリズム

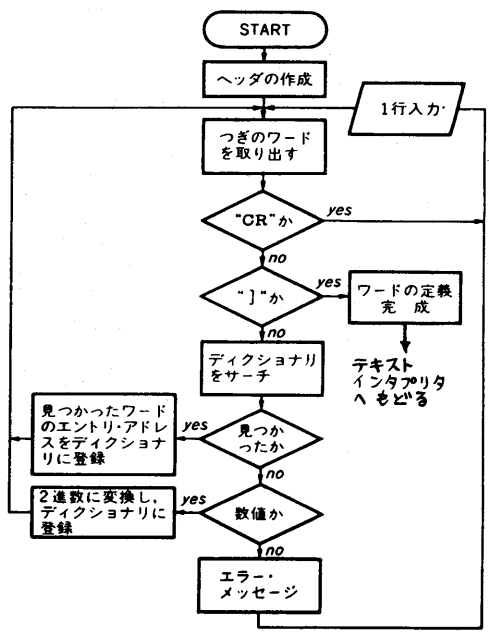


図7 コンパイラのアルゴリズム

4.3 内部インタプリタの動作

G-FORTHの内部インタプリタは，I-8086が持つレジスタを次の様に割当てて実現されている。

BX：次に実行するワードのアドレス（インストラクションポインタ）を保持
SI：内部インタプリタのエントリアドレスを保持
BP：リターン・スタック・ポインタ（正確には，ラベルSTACK＋BPの内容）
SP：パラメータ・スタック・ポインタ

内部インタプリタの動作を示す例として，WOVER，MULT5M（メモリのある番地の内容を5倍する）の2つのワードが定義された時のオブジェクトコードと，内部インタプリタの構造を図8に示す。各ワードのオブジェクトコードは，図3で示した様にヘッダ領域と定義本体とから成るが，G-FORTHではこれらを分離して別々のメモリ領域に置いている。これは，ヘッダ領域が必要なのはコンパイル時のみでありターゲットプログラム化した場合には不要であること，及び，プログラム（ディクショナリ）格納用のメモリ領域を拡張するための一方法として，I-8086の持つメモリセグメント概念を利用して別のセグメントに追い出すこと，の2点を考慮したためである。図8では，内部インタプリタ動作の理解を助けるために，FORTHレベルのワードの定義本体とアセンブリ言語レベルのワードの定義本体とが，別の領域に置かれている如くに描かれているが，実際には，ディクショナリ中に両者が混在している。

4.4 G-FORTH の開発

図8に示した通り，FORTHレベルのワードの定義本体は，アドレスポインタの並びであり，その順序は定義時のワードの順序に一致している。従って，

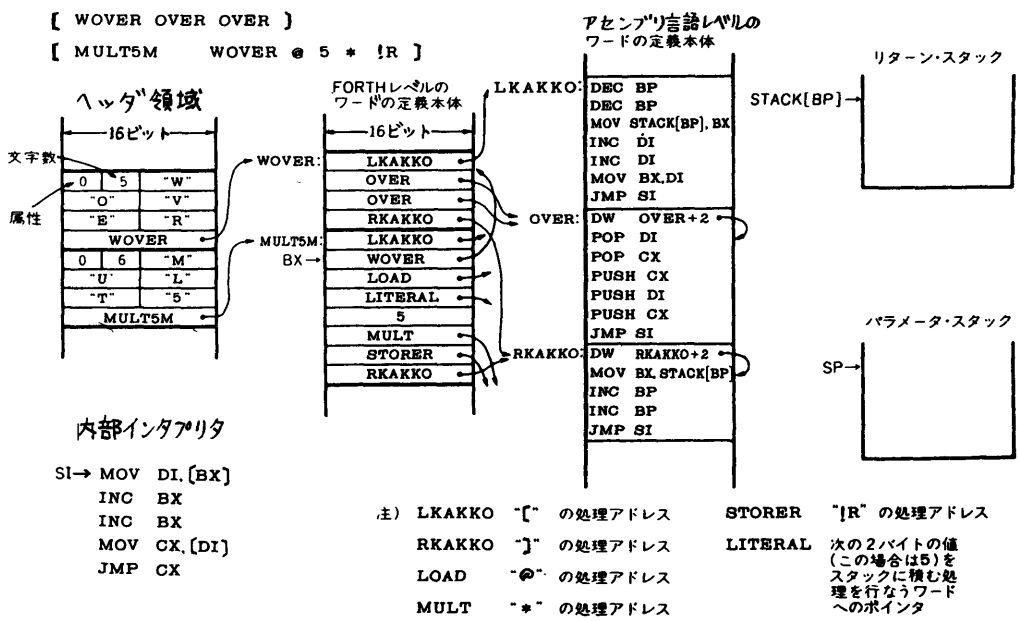


図8 内部インタプリタとオブジェクトコード

FORTHレベルのワードは，それを構成する定義ずみのワード名をDW（Define Word）を用いて図9に示す様にアセンブリ言語で記述してアセンブルしてやれば，正しくアドレスポインタの並びにコンパイル，即ち実行可能な形式に変換できることになる。

G-FORTHの開発は，次の様に行なった。まず，必要最小限の基本ワードをI-8086のアセンブリ言語で記述し，次いでこれらを用いて，コンパイラ，テキストインタプリタをFORTHレベルで記述した。これを図9の様にDWを使ってそのままの順序で書き直し，アセンブリ言語レベルのワードとともに，ホスト計算機上のクロスアセンブラによってアセンブルし，コンパイラおよびテキストインタプリタのオブジェクトコードを得た。この様に，FORTHの特徴である，ITC（Indirect Threaded Code）を利用し，比較的短期間（2ヶ月人）でG-FORTHを開発した。

```
WOVER:   DW LKAKKO
         DW OVER
         DW OVER
         DW RKAKKO

MULT5M:  DW LKAKKO
         DW WOVER
         DW LOAD
         DW LITERAL
         DW 5
         DW MULT
         DW STORER
         DW RKAKKO
```

図9　WOVER, MULT5Mのアセンブリ言語による表現

5. G-FORTHを用いたグラフィックデザインシステム

G-FORTHの応用例として，我々が開発したグラフィックデザインシステムを簡単に紹介する。このシステムは，次に示す様な，デザインシステムに必須な特徴を満たすことをねらっている。

① 図面の作成過程が階層的であること。
② 階層的な図面作成過程をサポートする操作の簡潔さ。
③ 対話性の良さ。

①の特徴に対応して，図形を，次の様に階層的にとらえるのが良いであろう。

1) 図面：図面は，部分図の集合，または図面の集合である。
2) 部分図：基本図形の集合であり，平行移動，回転などの変形の対象となる。
3) 基本図形：点，直線，円，円弧，楕円，半楕円。これらを描く基本ワードをアセンブリ言語で記述し，G-FORTHに付加。

この様な図形のとらえ方に対応し，図面は図10に示す手順で作成され，これらの手順を指示する対話用各種コマンドを用意し，それらコマンドのインタプリタをG-FORTHで記述している。PART，MODIFYは，それぞれ部分図内部データ作成フェーズ，部分図内部データ変更フェーズのインタプ

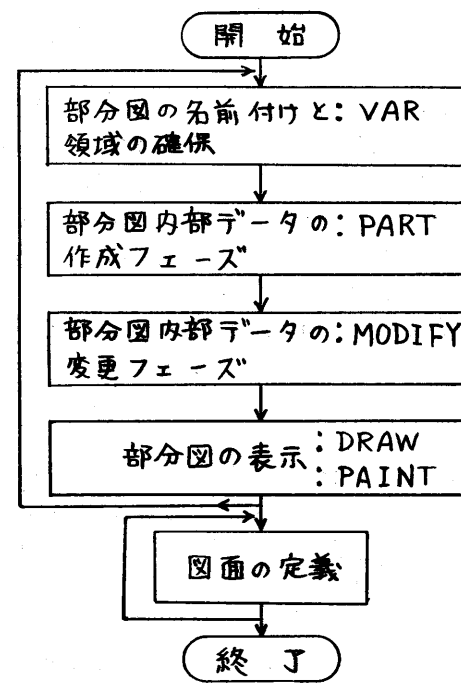
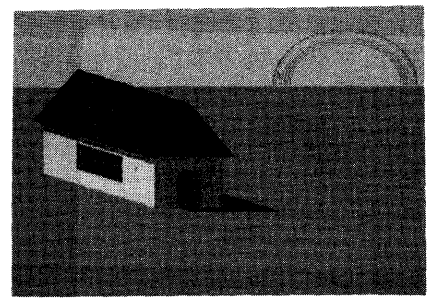
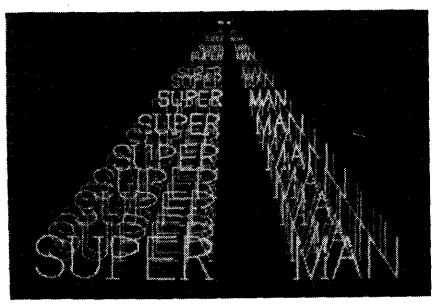


図10　図面作成の手順と，それに対応するワード

リタを定義したものである。DRAWは，部分図内部データを作成し，G-FORTHに付加した描図用基本ワードを用いて線図形を描くワード，PAINTは，線図形で囲まれた枠内に色を塗るワードである。DRAWやPAINTの系列によって定義されたあるワードは，対応する部分図の組合せで得られる1つの図面を定義したものに相当する。さらに，図面を組合せて新たな図面を定義できる。

この様にして得られたグラフィックデザインの例を，図11に示す。





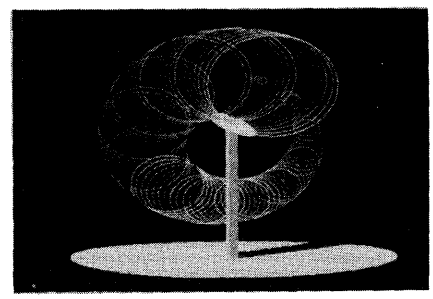


図11　グラフィックデザインの例

6. おわりに

本稿では，マイクロプロセサ複合体G-PSYCOにおけるFORTHシステムG-FORTHの開発経緯と，応用例としてグラフィックデザインシステムを紹介した。FORTHが，特殊なハードウェア環境におけるソフトウェア開発ツールとして適している点を利用し，比較的短期間でこれらの開発に成功した。この例の他に，G-FORTHを用いて，3次元カラーグラフィックスシステムが開発されている。

最後に，これらの開発を担当した本学学生　星野　寛氏に感謝する。

参考文献

1. 星野，久保，大野：G-FORTHを用いたグラフィックイメージデザインシステム，情報処理学会第22回全国大会，1981
2. 久保，星野：8086用G-FORTHとグラフィックスへの応用，インターフェース，1981年8月号
3. 杉本，久保他：研究室内マルチマイクロプロセサ複合体，情報処理学会マイクロコンピュータ研究会資料，18-4，1981
4. 井上外志雄：拡張性のある言語FORTH，bit，Vol.13，No.4，1981
5. 西原清一他：対話型説明図作成システムの構成について，情報処理学会第22回全国大会，1981