

マイクコンピュータによるLISP汎用図形編集システム

山本 隆 小林 隆広 青木 由直

(北 大 工)

1 まえがき

最近の16ビット/チップCPUの処理能力の向上には目を見張るものがあり、これらのCPUを用いたパーソナルコンピュータシステムは処理能力の点ではいわゆるミニコンピュータと肩を並べるものとなってきている。特にLISP処理系のように比較的大きな線形メモリー空間を必要とするシステムではこれらの16ビットCPUの特徴を有効に利用でき、従来の8ビットCPUでは困難であった使用システムが実現できる可能性がある。また、最近ではビットマップディスプレイ、インテリジェントマウスといった初歩的なマン・マシンインターフェイスがこうしたパーソナルコンピュータに標準装備されるようになってきており、図形処理端末としての機能も高度になってきている。その結果LSI設計、論理回路設計等に使用されるCADシステムもこうしたパーソナルコンピュータ上で実現できるようになり、ワークステーションとして既に実用化されている例もある。本報告ではそのようなCADシステムの実現例の一つとして16ビットパーソナルコンピュータ上のLISPを宿主言語として開発された汎用図形編集システムについて述べる。本システムはLSIなどの多層構造のバターンデータの対話形式の編集を行うものでありTree状のデータ構造により大規模なバターンデータの取扱いも可能となっており、従来はコストの面で困難であった個人用のCADシステムの実現を目指している。

2 汎用図形編集システム〈AIDS〉の概要

AIDS (Advanced Interactive Designer System) はLSIマスクデータを始めとする多層構造の2次元バターンデータの入力、編集を会話的に行うことを目的として開発された。必要なハードウェアは標準的なパーソナルコンピュータの範囲である事を前提とし、現在は富士通FM-11 (8088 CPU) 上のCP/M-86にインテリジェントレシであるが、同程度のシステムであれば移植は容易である。図形編集のためのポインティングデバイスとしてはインテリジェントマウスを、ディスプレイは内蔵のグラフィック機能を用いる。必要ならばハードコピー端末としてX-Yプロットと接続することができる。AIDSはその豊富なポリタイプデータ構造、Tree状のデータ管理、様々な図形編集コマンドによって特徴づけられる。

2.1 ポリタイプデータ構造

AIDSシステムでは基本データ構造として次の6種をサポートする。

1. LIN型: オートワイヤー
2. SHP型: 閉ループデータ (面積バターン)
3. REC型: 四角型データ (面積バターン)
4. TXT型: 文字ストリング
5. SYM型: シンボルデータ
6. FIG型: 他で定義された図形のネスティング

これらのデータ構造のうち、FIG型を除いた他のタイプではレヤルコードを与えられ、これによって多層構造を表現する。また、各データ型はこれらに独特のプ

ロハタイプを持つてあり、これは入力時に自動的に与えられるが、後で変更することもある。H/G型のデータは他とは異なりレハルコードを持たない。これはH/G型データが他で定義された図形そのものであり、展開されたときにもその中に含まれるポリミタイプデータが既にレハルコードを持つていたりするのである。H/G型データのネスティンクレベルの制限はやはり評価段階を故意に制限することによりリミットを設定することはできる。

2.2 Tree構造の図形データ管理

一般に大規模なLSIや論理図面の作成時に全データを同一レベルで取り扱うのは記憶容量の増大やデータサーチに要する時間が増加する点と好ましくない。本システムでは機能的あるいは局所的なまとまりをモジュールとして図形データを定義しこれを上位レベルで配置し、モジュール間の接続等を行う階層的レイアウトの手法をサポートする。既に定義されている図形データは他の図形中にネストで入るが、この場合、展開されポリミタイプデータとして追加されるのではなく、配置情報のみを入力される。ディスプレイ時にはネスティンクされた図形を横断するとその図形の評価を行い実行シーケンスを表示し他の図形データとの相互関係が表示される。しかしネスティンク図形の内部に及んで編集を行うことはできず、一担そのレベルの編集を終了し、編集するネスティンク図形をオーポニしてから行う必要がある。このようなデータ管理を行う利点としてデータオーバーレイを上げることである。大規模な図形の表示を行う場合、主記憶の制限から全データが同時に主記憶中に存在できなくなるが、この場合でも評価の過程で必要なのはTreeの中の現在注目している枝のみでありそれ以外は主記憶から追いつくことができる。またメモリー等の設計に対して基本セルのマトリクス配置を行うが、この場合もセルデータは全て共通となり記憶容量を節約できる。

2.3 データ編集コマンド及びシンタックス

AIDSシステムの編集操作コマンドは、基本動作を示すキーワード(動詞)とそのコマンド内の細かな分類を示すキーワード(名詞、形容詞)で構成される。動詞としては現在 種のコマンドが定義されているがこれらのうち図形データの操作に関するものに以下に示す。

1. INS系コマンド: 現在オーポニされている図形に対するデータの追加を行う。(Insert)
2. ERS系コマンド: 指定されたエントリを消去する。(Erase)
3. MOV系コマンド: 指定されたエントリを移動する。(Move)
4. STR系コマンド: 指定されたエントリを位相的な情報を変えることなく変形を行う。(Stretch)
5. BRK系コマンド: 指定されたエントリを変形する。(Break)
6. CHG系コマンド: 指定されたエントリに付属するデータ(プロパティ)を変更する。(Change)

これらのコマンド群は更にデータの指定法として直接に1点ディジタイズで指定するか(ENT)あるいはウィンドウに含まれるデータをすべて対象とするか(WIN)を指定する名詞を添える。データ編集の他のコマンド(ファイル操作、ディスプレイ操作、パラメータ設定等)についても同様に動詞-名詞型の構文となっており、いくつかの基本キーワードの学習で操作できるようになっている。

2.4 その他の機能について

本システムはLSI設計を想定して仕様を決められているが、他の分野への応用が可能とするためシンボル型のデータをサポートしている。これは論理回路記号電子回路記号といった、図型として個々に定義するには種類が多すぎるような基本ブロックについては、そのための専用データベースをあらかじめ定義しておきユーザーはそれを自由に使うことができるような機能であり、そのデータはLISPセル領域の外に置くことにより、テキストと同じレベルで取り扱うことができる。このシンボルデータベースを目的にカスタマイズすることにより専用システムが実現できる。この際、基本ブロックは本システムの編集機能を用いてデフォルト値とすることで、そのようにして作成された図形データをテキストに変換するユーティリティプログラムをサポートしている。

3. LISP処理系について

AIDSシステムはグラフィック操作関数を強化したLISP上で実現されている。本章では概略LISP処理系について述べる。

3.1 基本LISP処理系

本システムを開発するにあたり、核言語としてLISPを選んだ理由としては(1)図形データを配列で表現することの困難である。(2)Tree状の図形データの取り扱いが再帰的に処理を必要とする。(3)システムが大規模になると予想される構造的プログラミング手法を取り入れる必要がある。等であることができる。しかし開発の時点で16ビットCPU上のLISPシステムが入手困難であったため当初我々が既に開発していた8080CPU上のLISP処理系(LISP1.5)で基本アルゴリズムを実現し、同時に8086用LISP処理系の開発を行った。我々の開発した8086用LISP処理系は以下にあるような特徴を持つ。

1. セル構造はCAR部, CDR部共に16bit, セル領域は62Kbyte (約15Kセル)
2. Shallow bindingによる変数の管理
3. ハッシュ技法の採用
4. グラフィック操作関数群の組込み, 基本サード関数のSUBR化
5. 表記法はSTANDARD LISP形式

8086 CPUを用いる場合、線形メモリ空間が64Kバイト毎にセグメント化されるため、64KBを越えるセル領域を実現しようとするとオーバーヘッドが増大する。そのため、セル領域は一つのセグメント内(DS)とし、フルワード領域、ハッシュテーブルに更に一つのセグメント(ES)を与えるようなメモリ割当てを採用した。この結果、セル数は15Kセルと極めて小さいものとなり、セルの消費を防ぐためにハッシュ技法を採用し、データに関するマージを行うようにしている。また、数値アトムに関しては32bit整数であるが、数値演算の高速化のため、下位16ビットのみを用いている。LISPの形式は当初LISP1.5風の表記法でスタートしたが、8086初代の段階で大々的構造の変更を行ったため、今後のシステムの移植入点を考慮しSTANDARD LISP⁽²⁾に合わせることにした。本LISP処理系のCP/M-86上でのメモリ割当てを図1に示す。

3.2 グラフィック操作関数

本システムのような図形処理システムでは高速なグラフィックディスプレイ操作を行う基本関数が必須であるが従来、パーソナルコンピュータ上で利用可能な

LISP処理系ではこういった機能をサポートするのは入念でなかった。我々のLISPシステムではその応用目的が明確であるため、必要な機能を合わせて以下に示すような最低限のグラフィック操作関数を定義した。

1. (VISION <list of scale and origin>)
: ビューウィンドウのスケールと原点を指定する。
2. (TURN angle)
: 仮想画面の相対回転。
3. (POSITION (X,Y))
: カーソルを (X,Y) にセットする。
4. (DRAW (X,Y))
: 現在のカーソル位置から (X,Y) へ直線を引きカーソルを移動する。
5. (ERASE)
: 画面を消去する。

これらの関数群は必須であるが、更に高速処理を実現するため、グリッドパターンを表示、カーソルマーク表示といった特殊関数も組み込まれている。

本システムをサポートする仮想画面は X,Y 座標で 16 ビットで表現される空間を取り扱うもので、任意のスケール、位置にビューウィンドウを設定するこゝで出来る。また、回転を行うため、図形のネステイングに際して座標変換は仮想画面の処理の一部として内部で行なわれるため LISP 上の演算として実数計算は現れずに済む。また、マウスからの座標入力に対しては仮想画面上の座標に変換した後、値を返すような形で関数を実行しているため、ここでも座標変換は不要であり、演算はすべて 16 ビット整数の範囲で行なうことが可能となった。

4. AIDS システムの実行状況

AIDS システムの特徴の一つにデータ及びプログラムのオーガナイズ構造があることである。これは機言語である LISP が現在の所コンパイルをサポートしていないため全プログラムを同時にセル領域にロードできず、また大規模な図形データの取り扱いを可能にするために取られた処置である。プログラムのオーガナイズは、各コマンドプロセッサ（動詞名に対応）は 1 個の関数として定義し、共通に使用する関数群を常駐しておき、コマンドの要求があった場合にこれらのコマンドを呼び出すことにより行なわれる。この場合、新たなコマンドの要求があった場合でもその時点で在座しているコマンドが 3 個以下の場合はそのまゝロードするか、そうでない場合は最も以前に使用した関数を追い出し、常に 3 個のコマンドプロセッサがセル領域中に存在するものとする。その結果、編集操作の繰り返しに依りて使われるいくつかの基本コマンドは存在する確率が上がり会話操作のレスポンスが向上している。データのオーガナイズ管理については最後のカーブコレクションで得られたフリーセル数をモニタしそれに基づいて割り当てた時点からオーガナイズが完了する。この場合も古い順にデータを追って

00000	CP/M 86
	SYMBOL DATA (30K)
CS	CODE 領域 (16K)
	WORK 領域 (2K)
DS	CELL 領域 (62K)
ES	Fullword 領域 (32K)
	Hash Table (32K)
SS	STACK 領域 (20K)
40000H	

図1. LISP処理系のメモリレイアウト (CP/M-86 256K 環境)

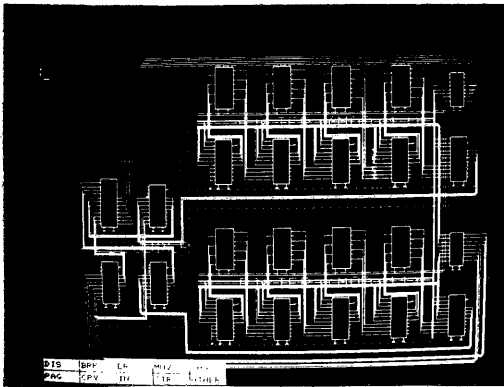


図2-1 論理図面作画の例

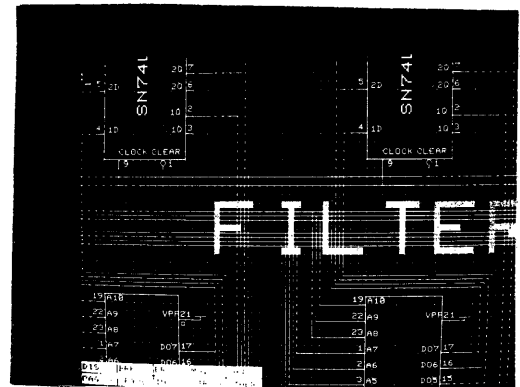


図2-2 左図の一部拡大の例

出し、3Kセル以上のフリーセルを得た時点でデータを取り取りを行う。このような管理を行うことで小規模な図形の時にオーバーレイが起らず極めてスムーズな編集操作を行うことが可能となっている。

AIDSシステムの実行状況を知るためのフローチャートとして動作中のフリーセル数をモニターするのは有効である。現在用いているLISP処理系ではセル領域が15Kセルであるが、常駐関数部のロードが完了した時点で13Kセルがフリーであり、SUBR関数を含めたシステムが2Kセル程度である事が知られる。更に普通の編集コマンドをロードした状態では1Kセル程度のフリーセルが得られ2112から、これが実際にデータ領域として用いられることのできる領域である。現在の内部データ表現では1バイト（1/8型）につき最大7セル、ハッシュによるマージが起った場合5セル消費するから1000~2000バイト程度程度の記述能力はあると考えられる。しかしこれは同一レベルでマージしたデータを入れた場合であり、Tree状のシステムマージを行った場合、その取り扱い可能量は指数関数的に増加するため簡単には述べることができない。しかし現実問題としてCPUの処理速度がそれほど高くないためむしろ表示速度の点から取り扱いうるデータの限界がくるものと考えられる。

図2-1、図2-2に本システムを用いているための論理図面作成の例を示す。この例ではTTL10個、PRAM14個からなる論理回路の完全な接続情報を含むものであり、1画面のリフレッシュに要する時間は約5秒であった。また、この図形データをオープンした状態でフリーセル数はまだ8000セル以上あり、このような用途に対する記述能力は十分であることがわかる。

5. おわりに

本報告ではLISPを用いた実行システムの実現例として図形編集システムを紹介した。残念ながら、用いたLISP処理系は一般的に流通ソフトウェアではなく我々が独自に開発したものである。このシステム自体がまた他のLISP処理系上に移植できるものではない。これは現在入手できるLISP処理系からのターゲットを数式処理、自然言語処理等においてあり、CAD等分野に対するLISP

の記述能力の向上にもかかわらず、その目的に使えるような処理系が入手困難である点に問題があると言える。今後パーソナルユニフォームの図形処理に対する応用の需要は増々高まると考えられるから早く早い時点で組み込み開発としてのグラフィクス操作の標準化が望まれる。また、AIDSシステムは現在の所、核となる図形編集システムのみが稼動してゐるが、今後、プロジェクタ出力、パナールワーク、論理シミュレーション等を同様のLISPを中心として開発する予定であり。現在はCP/M-86上で動作してゐるが8086 CPUはLISPを用ゐる場合アキタウワークが適してゐるとは思はず、今後上位のプロセッサへの移行についても目下検討中である。

参考文献

1. 小林, 山本, 青木 「リスト処理言語による対話型図形処理システム」, 電子通信学会, 情報システム部門全国大会予稿集, 講演番号187, 8858, 99
2. J.B. Marti, A.C. Hearn, M.L. Griss, C. Griss, "STANDARD LISP REPORT", University of Utah, UUCS-78-101