CADプログラム言語としてのProlog
—PLA設計問題への応用—

古関 義幸

（日本電気 ㈱ C&Cシステム研究所 ）

1. まえがき

100 万素子をのせたVLSIチップの実現が夢ではなくなったが、その設計は、CADによる所が大である。しかし、このCADのソフトウェアの開発コストは膨大であり、これを削減する事が大きな課題である。本稿では、最近注目を浴びているProlog言語を、PLA(Programmable Logic Array)のフォールディング問題とマスクレイアウトの生成に適用した結果をふまえ、CADソフトウェアを書くためのプログラム言語としての立場からProlog言語の利点と問題点について述べる。

2. PLA 設計

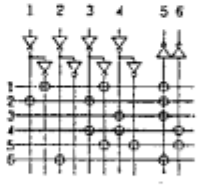
PLA 設計においては、論理式あるいは状態せん移表からPLA パターンを作成し、PLA マスクレイアウトを作成する。また、フォールディング（重みこみ）と呼ばれる処理により、面積の縮小を行う事がある。

3. フォールディング問題への応用

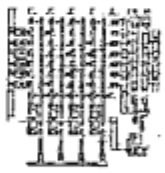
PLA のフォールディングとは、同じ積項線を共有しない入、出力をペアにして、対向するように配置する事によって、PLA アレイの面積を小さくする処理である。図1に元のPLA パターン、図2(a) にその回路図、同図(b) にフォールディング処理をした回路図、(c)(d)にこれらのNMOSマスクレイアウトを示す。ここでは、入力2と3、入力1と4、出力5と6がペアとして選ばれている。

```
in    1. 2. 3. 4:
out   5. 6:

          1234   56

/*  1*/ 0.0. : 1. ;
/*  2*/ 1.1. : 1. ;
/*  3*/ ...1 : 1. ;
/*  4*/ ..11 : .1 ;
/*  5*/ ..00 : .1 ;
/*  6*/ .1.. : 1. ;
```

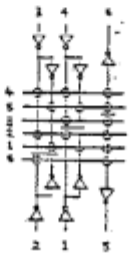
図1



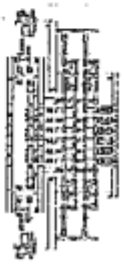
(a)



(c)



(b)



(d)

図2

(a)

(b)

図3

このフォールディング問題は、NP完全問題であるとされている。従って、ヒューリスティックを用いて多項式オーダで解く方法がいくつか提案されている。中でもHachtel等は、グラフの問題に帰着させて解くアルゴリズムを提案している[Hach82]。図3(a) が図2のPLA を表したグラフで、各ノードは入、出力(1-6) に対応し、各エッジは2つのノードが同じ積項線を共有する事を示す。ここでの問題は、以下の条件を満足するように、このグラフに有向エッジを加えて、図3(b) のようなグラフを作る事である。

(1) すでにエッジのある所には有向エッジをおかない。

(2) エッジと有向エッジを交互にたどるループがない。

(3) 有向エッジの数を最大にする。

この問題をProlog(Shapeup) でプログラムした結果が表1である。このように、C言語に比べて、行数で約1/3 、コーディング工数で約1/2 で実現できた。しかし、実行時間は、約60倍であった。また、処理できるPLA の大きさにも限度がある。

表1

	行数	実行時間	工数
Prolog	約 350行	約120 秒	約 6 人日
C言語	約1000行	約 2 秒	約 14 人日

（行数はコメントを含む。実行時間はVAX11/780 UNIX4.1bsdでLoad Average=2程度の混雑。 データは入出力数（ノード数）21、積項数13、エッジ数155 の小規模ＰＬＡ。工数はコーディングのみでデバッグを含まない。）

4. PLA レイアウト生成への応用

図１のようなPLA パターンを入力として、図2(c) のようなPLA のレイアウトを図４のようなCIF(Caltech Intermediate form)[M&C80]で生成する処理をProlog(Shapeup) を用いてプログラムした。行数は270 行であり、同様のプログラムをC言語で書くとすると、やはり、数倍の行数を必要とすると思われる。実行時間は表１と同じデータに対して、20秒程度であった。

```
DS 1000 ;
9 Pla ;
C 100 T 0 15000 ;
C 20 T 15750 15000 ;
C 20 T 11750 15000 ;
C 20 T 7750 15000 ;
C 20 T 3750 15000 ;
C 14 T 0 27000 ;
C 14 T 0 23000 ;
C 14 T 0 19000 ;
C 12 T 15750 27000 ;
C 12 T 11750 27000 ;
C 107 T 23000 15000 ;
C 106 T 23000 15000 ;
C 104 T 15750 19000 ;
C 104 T 11750 21000 ;
C 104 T 15750 21000 ;
C 107 T 23000 19000 ;
C 106 T 25000 19000 ;
C 105 T 11750 23000 ;
C 105 T 15750 23000 ;
C 104 T 7750 25000 ;
C 107 T 25000 27000 ;
C 106 T 23000 23000 ;
DF ;
```

図４

5. まとめ

上記の２つのPLA 設計用プログラムは、Prologを用いる事によって、いずれも少ない工数、少ない行数で実現する事ができた。これは、Prologの強力なリスト処理機能と、ユニフィケーション機能によるところが多いと思われる。Prologの大きな特長であるバックトラック機能は、ローカルな処理（集合演算、リスト処理等）で生かされているが、大局的な処理の制御には、ほとんど生かされていない。その理由は、大局的な制御にバックトラックを使うと余りにも動作が複雑になりすぎ、プログラマの管理能力を越えてしまうからである。

これらのPLA 設計問題は、データ量も高々数百程度であり、処理内容も比較的単純である。しかし、現在のPrologの処理系では、扱えるデータ量、実行速度共に、この程度の処理が限界のようである。レイアウト問題のように、数千、数万のデータを処理しなければならないＣＡＤの分野の処理を全てProlog で処理する事は現在の処理系では不可能に近い。

実行速度はPrologの一番大きな問題点である。処理系の動作を考えずに宣言的にプログラムを書くと、データ量に比例した処理時間で実行できる処理できえ、２乗あるいは３乗のオーダの処理になってしまう事がある。これを避けるように注意深く処理系の動作を考えながら手続き的にプログラムを書いたとしても、リスト処理あるいはユニフィケーションの内部処理のために、処理系を含めた全体の処理のオーダが、従来の言語で書いた場合に比べて大きくなってしまうのを避けられない事が多い。

今後、コンパイラ、専用ハードウェアによる並列処理等の工夫により、実行速度も改善されていくだろう。ただ、それと同時に、手続き的処理機能の導入あるいは他のデータ構造（アレイ、集合等）の導入、または他言語とのリンク機能といった実用的な機能の拡充が期待される。これらの処理系の改善により、Prolog言語はＣＡＤソフトウェアの工数削減に充分寄与できるものと考える。

[謝辞]

日頃ご指導頂く、当研究所、鷲沢田部長、後藤課長、田田主任に感謝致します。

[参考文献]

[Hach82] Hachtel, G. D, et al., "An Algorithm for Optimal PLA Folding," IEEE trans. on CAD, Apr. 1982 pp.63-77.

[M&C80] Mead, C. A., Conway, L. A., INTRODUCTION TO VLSI SYSTEMS, Reading, Addison-Wesley, 1980.