PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 37, NO. 1, PP. 73-78 (1993)

HYBRID TOPOLOGY DESIGN USING ARTIFICIAL INTELLIGENCE TECHNOLOGY

Gábor GOMBÁS

Department of Electronics Technology
Technical University of Budapest
H-1521 Budapest, Hungary
Tel: (36 1)166-6808/1520, Fax: (36 1)166-6808

Abstract

When composing computer aided design algorithms, it often proves useful to examine the following problems: how does the human designer plan and how does he handle parallelism and foresight. If we want to create a model of human designer planning methodology, we can use the artificial intelligence technology. In this paper we illustrate our approach by an example. The example is an autoplacer program for designing hybrid circuits.

Keywords: hybrid circuit design, artificial intelligence, autoplacer.

1. Introduction

The basic problem is the complexity of a plan. We do not know of any procedure which can handle the topology design in one step. So we need to separate the problem into two parts. The first is to find the best places for the components and the second is to make the interconnection of the pads of the components. Generally the two procedures have different goals. That means, e.g. the placer algorithm does not know anything about the routing method, so it cannot help the routing.

If we want to design thick-film hybrid circuits, we have some other difficulties as well. The general placer and router methods were developed to design printed circuit boards. These models do not suit hybrid circuits because they are not able to handle many similar components, they cannot select the most important components and the size of hybrid components is comparable with that of the wires. Therefore we need to use other algorithms.

2. Information, Rules

There are many kinds of information and rules which need to be considered. This information can be grouped. The first part of the data contains the rules and dimensions defined by the technology (e.g. width of wire,

minimum distance between wires). The second group of information is the topology of components and the board. The third one concerns the connections between the pads of components. In the next group of information there is the experience of manufacturing, and other, not very important properties. Finally, there is information about the style of the autorouter.

3. Representation of the Rules

There are many design rules. One of the main problems is the efficient representation of the rules because they frequently need evaluation. Many rules and dimensions can be represented by using bit maps. For example the map of the placed components is filled on the basis of the topology of the components, increased with a distance defined by a technological rule. We can use this rule as follows: if the distance between components is X, then, at first, we fill the picture of the first component, increased with X, in the first bit map. The second component is represented in the second bit map without modification. Later, every placed component is filled in the first bit map, so this will be the picture of the board (*Fig. 1*).

If we want to know that a selected position for a component is correct or not, we perform a logical AND between the first bit map and the second bit map, moved to the selected position. If we get a result which contains only zeros, that means the position is suitable.

In hybrid circuits, there are two basic types of components: the components which are made with thick-film technology and those which are mounted. The topological design rules of these two parts of components are different, so we use four bit maps.

Another important question is how to take the style of the autorouter into consideration. It must be a very simple model because a more exact model is more complex and it takes much time of computation. But this simple model should be efficient in trying to reserve the place of wires.

A further scaled map can be used to represent the connections between the pads. The measure of each pixel shows how much that point is required for connections. The simplest connections can be Manhattan lines, with the smallest measure. According to the importance, the measure of the lines can be increased incrementally. If, for instance, a node is considered even more important (e.g. it consists of only two pads), its connection route can have even higher weight (*Fig. 2*).

As a result, the scaled map consists of a measure (a number of 4 or 8 bits) for each point. The pixels of the map which are involved by connection are defined by the Manhattan routes. The overlapping Manhattan routes of different connections increase the measure of the points incrementally.

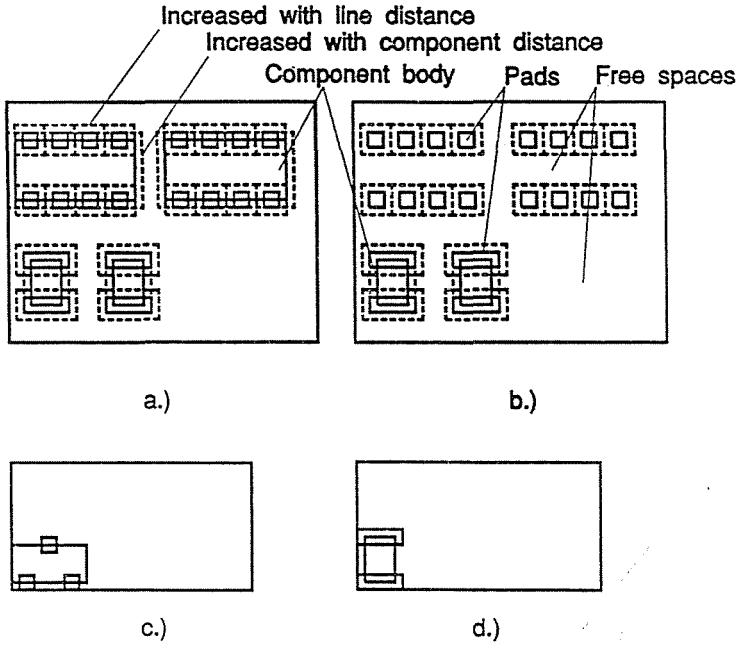

Fig. 1. Representation of rules: bit maps
a) a simple bit map for the placed components;
b) a bit map for the thick film components;
c) actual placed component;
d) actual thick film component.

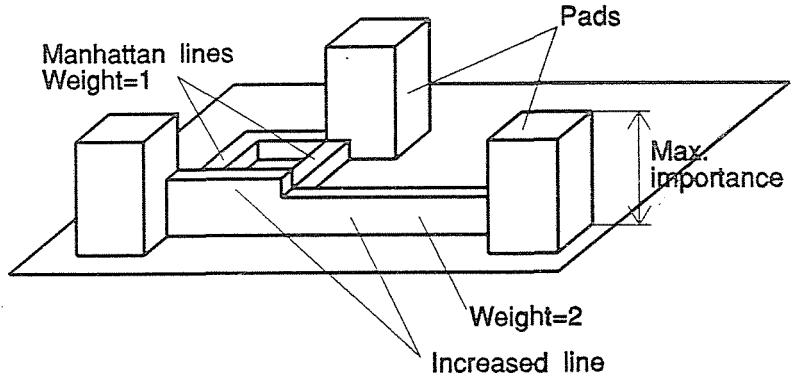

Fig. 2. The scaled map

In this way the scaled map shows the importance of the points. When we want to know how much an area is reserved for connections, we need to

summarize the value of points below the component to be placed. If it is higher than a limit determined by the designer, it means that this place can be accepted but it is not comfortable for the component.

With these methods we can also represent other properties, such as prohibited areas and the dissipation of components.

4. The Best Location

Another important question is where is and how to find the best location for a component. Generally the distances between the pads of each node of the component to be placed are summarized, but this method does not match the philosophy of the autorouter. A good autorouter finds the shortest connection to the given pad of a node. Therefore it is a better estimation to summarize the shortest distances only (*Fig. 3*).

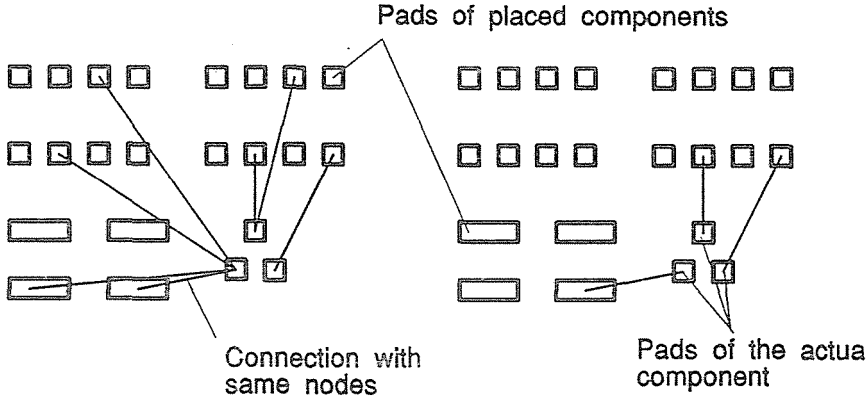

Fig. 3. Estimation of distances:
a) all connections of all nodes of the component;
b) shortest connections of all nodes of the component.

The other aim is to reserve space for wires. Previously we saw the method of calculation of the importance of a place. When we look for the best position for a component, we try to find that location and orientation of the component, which is the nearest to the placed components and has minimal value of importance.

Generally we have got a number of good positions. These positions are not equal, but they are all acceptable and suitable. We need other criteria to make the difference. We can use the rules of practical experiences. At this point the program can apply an expert system.

However, we are still not ready to select the position. Generally some components have similarly good placements and in this stage of the design we do not have sufficient information to decide whether one of the components needs a place more eagerly, than the others do.

5. When Can a Component be Placed?

The placement of a component contains three steps: at first we need to check whether a position is suitable, then the positions are qualified, and, finally, conflicts with other unplaced components are discovered and solved.

A simple algorithm has been found. We try to place all unplaced components. The best 3...5 places of every components are written in the first level of a tree. The current state of the placed components is the root of the tree. If a component can be placed without destroying a good position of any other component, it will be placed. If a component conflicts with another one, but the other has a good position which should not be destroyed by the first component, then this other component will be also placed. Otherwise the algorithm tries to imagine further steps of the design. These steps are represented as the next levels in the tree (*Fig. 4*). On this basis the algorithm can find differences between sequences of the placement of the components.

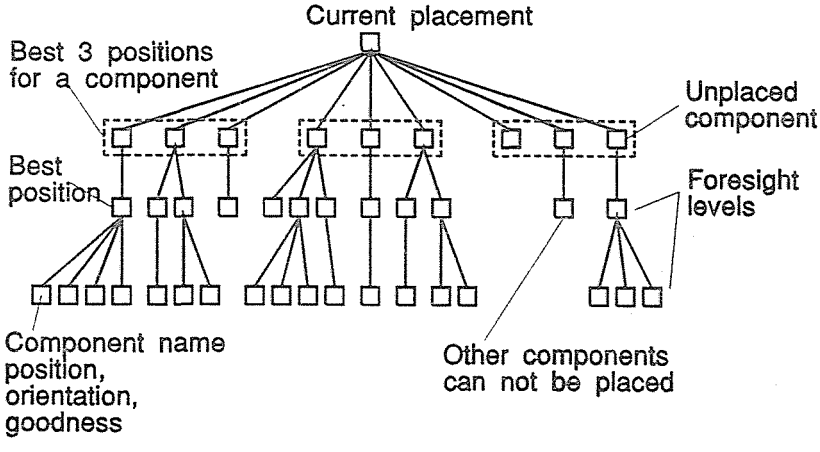

Fig. 4. The preview tree

Each node of the tree represents a component with its position and orientation, and quality of the placement. A component can be found on a level several times because it may have a lot of suitable positions.

The generation and manipulation of the tree are made by a special language, in the same way as in an AI (artificial intelligence) program.

The speed of the program can be increased by adding some heuristics.

6. Conclusions

The results of this method represent several improvements in comparison with earlier methods. This algorithm places the components quasi parallel if it is possible. In other cases, it tries to determine the optimal sequence of placement. It does not destroy any 'comfortable' position of components without checking consequences. Although the running time is generally long, the algorithm solves the problems effectively and efficiently.

The algorithm automatically clusters components with close connections to the same part of the substrate.

Generally it reserves space for wires and so the router can be quick and it generates shorter wires.

These improvements are in close connection with the problem solving method of programs based on artificial intelligence techniques.

We have some experience with general purpose languages of artificial intelligence (LISP, PROLOG, etc.). The main problems are the efficient representation of rules, the working style and the memory management of inference engine. Therefore the program was written in C and assembler languages in object oriented style. The program works like a typical artificial intelligence program but it is more efficient.

References

GOMBÁS, G. – RUSZINKÓ, M. – SZIKORA, B. – ILLYEFALVI-VITÉZ, ZS.: Hy-CAD in Practice, *Thirteenth International Spring Seminar on Electronics Technology,* Göd, Hungary, 15–19 May, 1990.

ILLYEFALVI-VITÉZ, ZS. – GOMBÁS, G. – BARABÁS, T.: Hy-CAD : Hybrid Topology Design, *Thirteenth International Spring Seminar on Electronics Technology*, Göd, Hungary, 15–19 May, 1990.

ODAWARA, G: CAD Systems Using AI Techniques, *Proceedings of the IFIP TC 10/WG 10.2 Working Conference*, Tokyo, Japan, 6–7 June, 1989.

PREAS, B. – LORENZETTI, M.: Physical Design Automation of VLSI Systems, The Benjamin/Cummings Publishing Company, 1988.

ROSSMAN, W. – VOLMERHAUS, E.: Trends of CAD/CAM in Hybrid Circuit Design, *Proceedings of the 7th European Hybrid Microelectronics Conference*, Hamburg (Federal Republic of Germany), 24–26 May, 1989.

SCHILD, H.: Artificial Intelligence Using C Osborne, McGraw-Hill Berkeley, California, 1987. 11.