PARALLEL PROCESSJNG OF A MULT'ILAYER ROUTING PACKAGE

M K Srinivas
Computer Aided Design Lab
Indian Institute of Science
Bangalore, India

T Ratdhakrishnan
Computer Science Dept.
Concordia University
Montreal, Canada

Abstract

With the increasing density of components on Printed Circuit, Boards (PCBs) and the advancement of fabrication technologies for multilayer PCBs, improvement of speed and techniques for the Computer Aided Design of multilayer PCBs has become a major area of research. The paper discusses the parallel processing of layering and routing algorithms for design of multilayer PCBs on a network of small computers with a moderately high speed communication medium. Multilayer PCB design consists in partitioning a netlist into as many layers as necessary, and routing one or two layers at a time. The routing processes of the different netlists is entirely independent of each other.

1 Introduction

With the advancement of electronics, layering and routing of multilayer PCBs have become a common requirement even in developing countries. In these countries, availability of special purpose high speed computers intended for PCB layering and routing is rare. Several small companies involved in the PCB making business cannot afford such computing facilities. On the other hand it is not uncommon to find several PC/AT class machines to be readily available in such small companies. It is in this context,, that we consider parallel processing of the layering and routing algorithms with the use of several PCs to be useful. The PCs are connected by means of a high speed (10 MBPS) Local Area Network (LAN) like Ethernet.

Based on the well known algorithm for routing due to Lee[3], a software system was developed at the Indian Institute of Science, Bangalore in India. The system[2] is useful for the design of two layer PCBs. Several timing analysis and measurements have been carried out on this package. In this paper we examine the parallel processing, of the layering and routing of multilayer PCBs, based on the above experience. First, we consider partitioning of the given netlist into concurrently processable netlists. Routing in such layers; are carried out concurrently. Communication between different routers and the

layering package takes place through message passing over the high speed LAN. Secondly we consider the cost of partitioning the netlist. Because this task consumes non-trivial amount of time, we consider methods for overlapped execution of partitioning and routing in selected layers. The intention is to reduce the overall time required to solve the layering and routing problem. Finally, we describe a parallel processing system implemented on a network of Xenix based machines (PC/AT) which are interconnected by a 10 MBPS Ethernet.

2 Layering

Layering consists of partitioning a given netlist into many netlists - one for each layer (or a pair of adjacent layers). Layering may consist in partioning the netlist into a given number of layers or may consist in finding the number of layers to to satify some objective. In most cases the former approach is more desirable and feasible for fabrication. This approach involves tentatively assigning each route to one of the layers of a given multilayer board in such a way as to minimize the wiring difficulties on each layer. Every wire (route) is assumed to be drawn as a straight line between the two pins involved. The problem now becomes one of locating the wires on different layers so that on each layer the number of crossovers is minimized. The fact that the straight (Euclidean) lines cross does not mean that the actual routes on the final layout must necessarily cross, but it does imply a potential conflict, its the reverse condition is true.

The layering algorithm from[1] is briefly explained below:

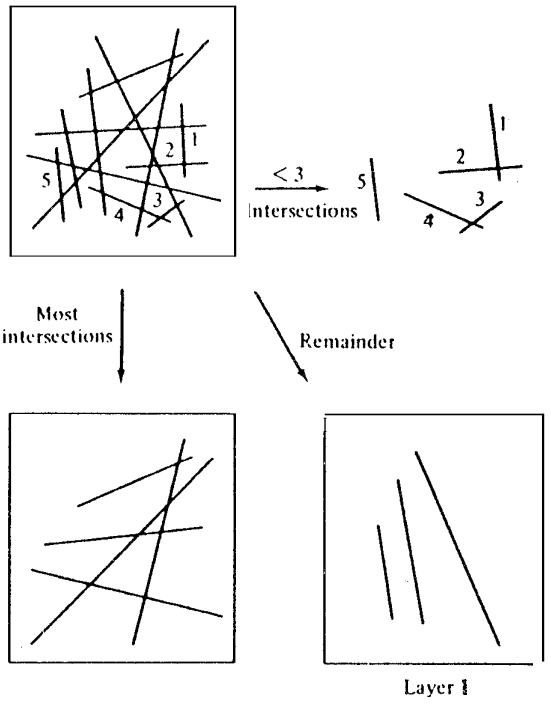


Figure 1: First step in layering

If a set of wires is to be partitioned between k layers, then any wire which intersects less than k others may be ignored. Suppose a wire x intersects with $k-1$ other wires, then wire x can be ignored. The reason is, we' have k layers, and so the $k-1$ wires which are intersecting the wire x can be put in atmost $k-1$ layers and even then we have one more layer left, in which the wire x can be located.

Consider the set of wires shown in Fig 1 to be located on three layers. Five wires can be removed as they intersect greater than three other wires. The order in which these wires were removed must be remembered as they must be relocated in the reverse order finally. In the remaining set of wires each wire will intersect atleast k (three) others. Now start removing wires from this set, placing them on layer

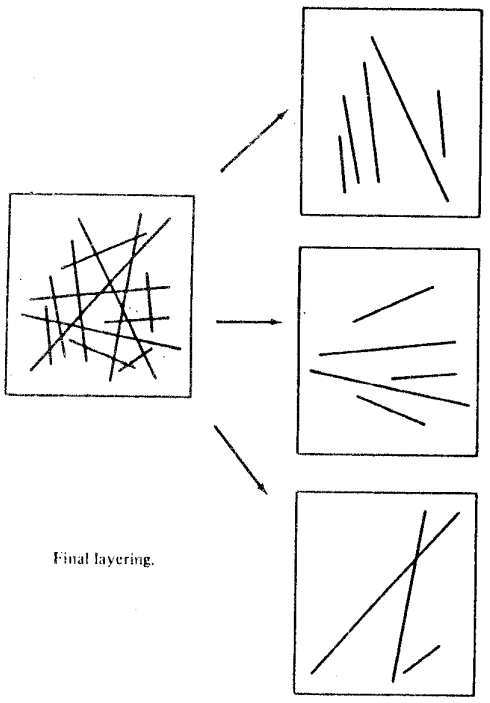


Figure 2: Final layering

two until no intersections remain on layer one. Now the entire process is repeated assuming two layers, and so on until until all the layers are exhausted. Finally all of the wires removed by the application of the first rule are p aced in the reverse order from that in which they w ere removed. The final layering is shown in Fig 2.

It can be noted here that the locating (assignment) of wires to layers is incremental, so that as soon as the wires of layer one is known, routing of these wires can start immediatly on some other processor in parallel with the layering process. Only at the end of the layering process a few more wires may be added to layer one by which time the routing process is still not completed.

An ordering algorithm which finds the order in which the wires have to be routed in each layer helps in higher completion rates for the router. The simplest and the most efficient way of ordering is to route smallest routes first [2].

3 Routing

Routing is the process of formally defining the precise conductoir paths on the PCB to achieve the indicated electrical connections subject to some imposed constraints. The constraints could be, number of feedthroughs for multilayer boards, spacing between conductor paths, width and length of conductor paths, number of layers, fanout at each pin, etc,. Most of the above mentioned constraints can be easily handled by a grid based block router which is actually an application of the shortest path maze running algorithm [3] due to Lee.

A two layer routing algorithm with automatic via minimisation has been implemented in a PCB design package[2] at the CAD Lab., IISc, Bangalore. It can be used to route a single layer or two layers at a time by the control of a logical parameter. It can also be used to do a single layer routing. Two versions of routing has been implemented in this package.

1. A terminal to terminal route finding
2. Finding routes to the nearest equipotential point

The package provides a fully automatic, semi automatic and full manual routing capabilities. For mote details of this package please refer [2].

4 Sequential processing for multilayer PCB design

The layering, ordering and routing algorithms are integrated in a design package such that the partitioned netlists are routed on the same machine one after another for every layer. A few case studies were done on a layering algorithm implemented on a PC for design of multilayer PCBs. The results of the case study for a four layer **PCB** with a total of 79 routes is given below:

Time for layering and ordering – 4 min on a PC/XT.

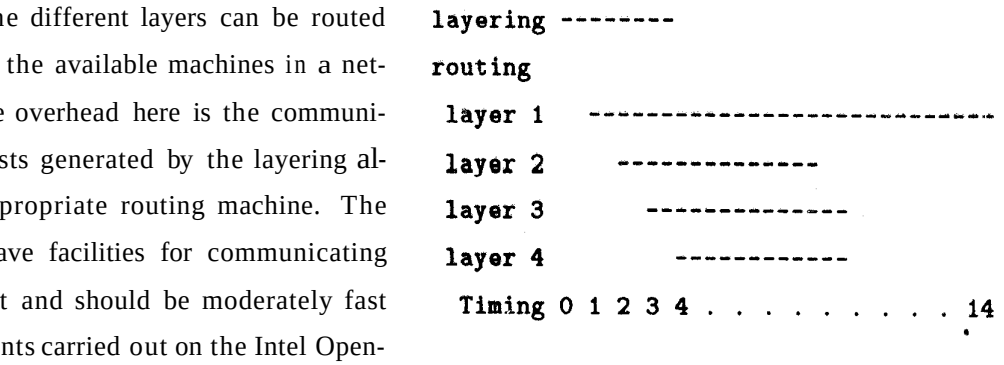
Layer No.	Time in min.	Routed/Ifnrouted
1	13	32/5
2	7	16/1
3	7	12/2
4	6	7/4

Total time = 4 t 13 t 7 + 7 t 6 = 37 *min.*

5 Parallel processing of layering and routing on a network of PCs

The netlists for the different layers can be routed simultaneously on the available machines in a network of PCs. The overhead here is the communication of the netlists generated by the layering algorithm to the appropriate routing machine. The network should have facilities for communicating on a virtual circuit and should be moderately fast enough. Experiments carried out on the Intel Opennet network of Xenix based PCs at the Concordia University suggest that the time for communication is a linear function of the number of bytes transferred with no other load on the network, and it takes about 20 secs for 1000 bytes between two machines, and 22 secs for sending 1000 bytes each to two other machines simultaneously from a single machine.

One of the machines in the network is made as the layering/routing machine and the remaining machines are made to run the routing in single or two layer. Since the netlists for the different layers are generated incrementally for one layer after another, they are immediatly communicated to the appropriate routing machine and routing can start immediatly for that layer on that machine, The routing of the last layer can be done on the same layering machine or can be communicated to some other routing machine, The whole process can be speeded up further if the individual machines were to be specialised in hardware'for layering and routing, The speed one would achieve with layering and routing on a network of PCs or specialised machines with the example timings given in the previous section is as follows:



speedup = 37/14 > 2.5 for four machines.

6 Conclusion

Based on our experiences with the multilayer PCB design package running on a single PC, and experiments carried out on a network of Xenix based PCs we have proposed the feasibility of a multilayer PCB design system on a network of PCs with a speedup factor of more than 2.5 for four processors, in comparison with having a multilayer PCB design package on a single machine. The speedup is limited by the number of layers to be routed and the time taken for routing the layer which takes the longest time. This is because of the fact that the loads on the routing machines are not balanced and the layering algorithm does not consider this criteria of load balancing.

References

[1] Melvin A Breuer, *Design automation of* digital *systems, theory and techniques*, Vol. 1,Prentice Hall, Englewood Cliffs, 1972.

[2] Srinivas M K, Reddy S S, Rajat Moona, *Implementation issues of a two layer block router based on Lee's algorithm on Personal Computers*, IEEE Region 10 Conf., TENCON 87, Aug 28-30, 1987, Seoul, pp 774-778.

Lee C Y, *An algorithm for path connections and its applications*, IRE Trans. Electronics and Computers, Sep 1961, Vol. 10, pp 346-365.