Routing Algorithms for Multi-Chip Modules

Jens Lienig, K. Thulasiraman, M. N. S. Swamy

Department of Electrical and Computer Engineering
Concordia University
1455 de Maisonneuve Blvd. West
Montreal, Quebec H3G 1M8, Canada

Abstract

We present routing algorithms for multi-chip modules. Two routing strategies, a channel routing and a grid-based routing, are discussed. The channel routing enables the designer to examine an effective routing during the placement phase. The grid-based routing calculates the net ordering with a new cost function and includes an effective rip-up and reroute procedure. The paper concludes with the presentation of some routing results.

1 Introduction

During the last decade great efforts have been made to transport the high integration degree of integrated circuits into other design levels. Surface mounted technology has become important in increasing the degree of integration, as has the development of new multi-chip modules. A multi-chip module is a special hybrid circuit which allows a compact package of a great number of different components. Such components are mostly chips without packaging, but surface mounted devices are also possible. The interconnections of these chips (here after the word "chips" describes all possible components placed on a multi-chip module) are realized on several layers which are situated between the chips and a ceramic substratum.

There are several new design automation problems associated with multi-chip modules. One of these problems is the lack of suitable layout design algorithms which consider all the constraints relating to multi-chip modules ([1, 2]).

In this paper we present routing algorithms which can be applied successfully in the design of special multi-chip modules. Section 2 describes the routing problem in detail. The routing algorithms are presented in section 3. In section 4 the details of the implementation and some results are shown.

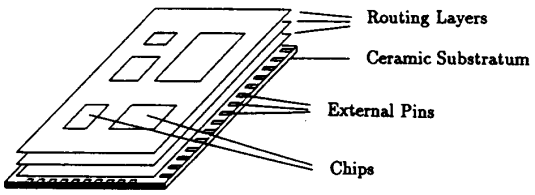

Figure 1: A multi-chip module

2 Problem description

In this paper we focus on a special multi-chip module which is shown in figure 1.

The chips are placed on the top layer called the chip layer. The interconnections are formed on a certain number of layers including the chip layer. The external pins are situated on the edges of the ceramic substratum.

The multi-chip module routing problem is: Given a set of rectangular routing regions with z_{max} routing layers, a set of Nets $\mathcal{N}$ are to be routed within these layers subject to technological, electrical and heat constraints.

Technological constraints which are to be considered during the routing procedure are the maximal via depth, the width of signal and power/ground connections, the size of the vias and the grid size of the layers.

Constructive arrangements (e.g. H-V-routing) and a special interface [3] to check the routing results are created to meet the electrical constraints.

Temperature problems are solved with special vias which are used only for heat exchange between the chips and the ceramic substratum [4].

0-8186-2780-8/92 $3.00 © 1992 IEEE

The objective of routing is an automatic complete routing thereby achieving a proper balance in minimization of routing length, minimization of number of vias and routing in preferred direction of layers.

3 Routing algorithms

3.1 Overview of our proposed algorithms

We introduce two different strategies for the routing of multi-chip modules:

- a gridless channel routing consisting of global and final routing,
- a grid-based routing.

The global routing algorithm generates channels and assigns nets to the channels. This global routing algorithm can also be used during the placement phase [2] to allow an effective estimation of the routability of the current placement structure. Thus it overcomes the traditional separation between placement and routing.

During the final routing the nets are assigned to the tracks in the channels.

The grid-based routing was developed as a second possibility to route multi-chip modules. It is based on a fast line-search algorithm combined with a modified Lee algorithm [5] and an effective rip-up and reroute procedure.

3.2 Placement estimation and global routing

The algorithm for placement estimation and global routing is used either during the placement phase (to estimate the current placement structure only) or after the final placement of the chips. In the latter case the nets are assigned to the channels to prepare the following final routing.

The generation of channels is shown in figure 2.

The edge coordinates of the pads are connected and continued to the edges of the chip layer. These channel borders are transferred to all layers.

The channel coordinate consists of the position of the channel within the channel grid and a parameter k. The sign k enables us to store effectively a number of different features of the channel, e.g. the current capacity of the channel ($k = 1$), the current path search potential ($k = 2$), etc.

This channel model has the following advantages:

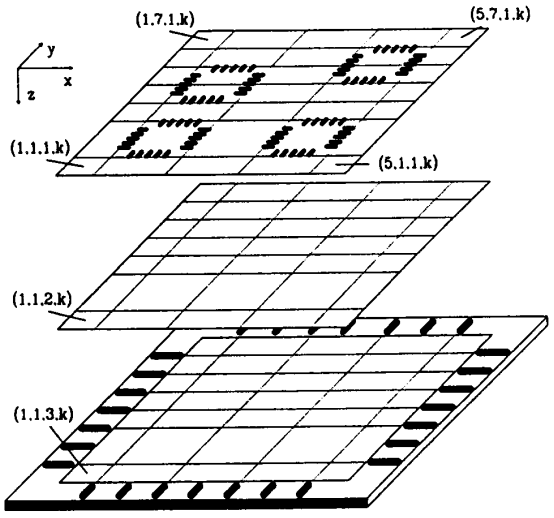

Figure 2: Generation of channels

- channel generation is efficient in terms of time and memory consumption,
- it is usable for multilayers,
- it takes into consideration the whole routing area.

$(2n + 1)^2$ channels are generated on each layer in the worst case, where n is the number of chips. Thus the memory space complexity is $O(n^2)$.

A reservation of tracks is made in a channel according to the number of pins belonging to the border of this channel. So the possibility to connect a pin is guaranteed even in an advanced routing situation.

First the algorithm calculates a starting pin of each net. The second pin of each net is calculated according to the priority chosen by the user. These pairs of coordinates are stored in a list where the pair with the highest priority is situated on the top of the list.

The path search is based on a modified Lee algorithm [5]. The Lee algorithm consists of an extension phase and backtracing. During the extension phase the neighbor cells of a start cell (starting pin) are labelled with the current number of the wavefront, i.e. with number 1 at the beginning. Afterwards all neighbor cells of these marked cells are labelled with the number 2. These extension cycles terminate when the target cell (target pin) is reached or when there are no free cells to be labelled. The backtracing is accomplished by following a sequence of decreasing cell labels starting from the target cell. The shortest path

between the start cell and the target cell is found with the backtracing.

The path between the pair of pins is found by the extension phase in the channel grid. Only channels whose current free channel capacity (number of tracks which are not occupied yet) is greater/equal 1 are considered. If this current free capacity of a channel is very small (i.e. the channel is nearly exhausted) the extension on this channel is delayed to prefer other (longer) ways. The delay factors (number of extension cycles without including the channel) in table 1 has been found to be favorable.

current free channel capacity	delay factor
1 track	4
2 tracks or less than 30%	3
3 tracks or less than 50%	2

Table 1: Delay factors according to the current channel capacity

After this routing a new pin of the current net which is not connected yet is chosen according to the priority. This pin and the nearest channel position already initialized by this net are incorporated into the list.

Because the pins are always located on channel borders, at least two channels per pin are usable to connect this pin. Thus, a shifting of routing channels is possible if two pins could not be connected.

As a result of the global routing the nets are assigned to the channels and the channels are labelled with their occupation. Thus the placement of the chips can be improved to get a better routing result without endangering the already routable nets.

3.3 Final routing

During the final routing the nets are assigned to the tracks in the channel and the length of each track occupation is calculated.

The situation at the end of the global routing is shown in figure 3 on a simple example with four chips and one net a.

At first the channels are combined into the so-called "super channels" according to the routing direction of the layer (see figure 3).

We assign the nets to these super channels in such a way that the fixed segments (segments with at least one pin connection) are first placed. Afterwards the movable segments are assigned whereby the longest

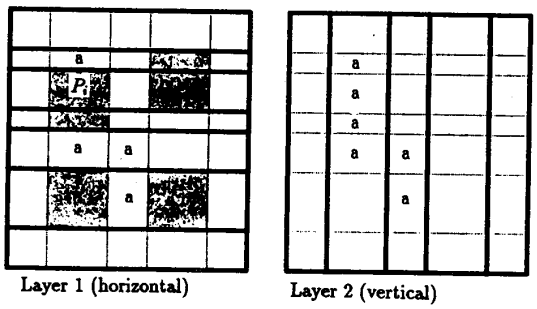

Figure 3: Result of the global routing and the creation of super channels

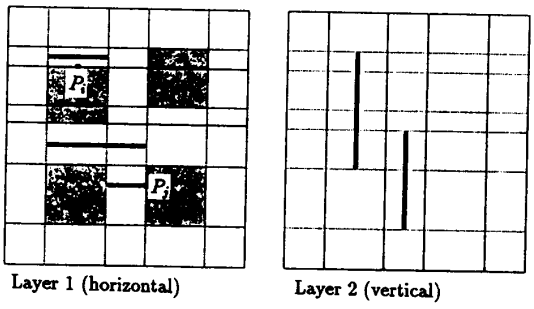

Figure 4: Net assignment

segments are placed in the middle of the channel (see figure 4).

The channel combination and net assignment described above takes place sequentially in each layer starting with the chip layer. Before changing from layer n to layer $n+1$ the layer $n-1$ (if it exists) is finished finally. This layer $n-1$ is superimposed with both layer $n-2$ (if it exists) and layer n to fix the length of the occupied tracks (see figure 5).

The final result is also shown in figure 5.

All layers are checked because of conflicts caused by oppositely located pin connections belonging to the same channel. Then track change is tried. If it is not possible, doglegs are implemented to solve these conflicts.

3.4 Grid-based routing

3.4.1 Introduction

We develop a grid-based routing strategy as another possibility to route multi-chip modules because of its

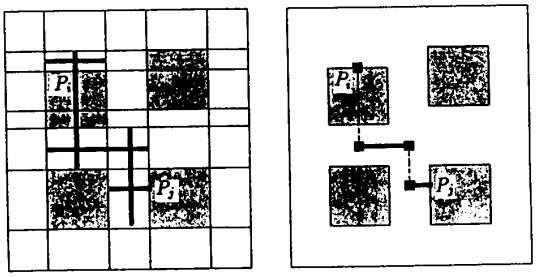

Figure 5: Superimposition and final result

capability to take into account a number of different technological constraints.

The layers of the multi-chip module are covered by equidistant horizontal and vertical grid lines. The size of the grid unit equals the sum of the smallest wire width and the minimum spacing between wires.

To ensure that the sequential net ordering creates nearly optimal routing results, we develop a special cost function to find the best sequence of routing. In addition to this we implement a rip-up and reroute algorithm to rip-up connections already routed.

3.4.2 Cost function

During a sequence of routing an optimal path between two points can cause an unroutable situation later. To avoid this we create special blocked areas. These areas are rectangular areas around the pins which are considered later by the routing algorithm.

In addition to this we introduce a cost function $C(P_i, P_j)$ of every two pins (P_i, P_j) that are to be connected:

$$C(P_i, P_j) = C_{user}(P_i, P_j) + C_{adapt}(P_i, P_j). \quad (1)$$

The parameters of $C_{user}(P_i, P_j)$ are chosen by the user before he/she starts the automatic routing. They include direction costs (preferred directions), length costs (long connections are preferred or v.v.) and shape costs (I-shaped connections are preferred to L-shaped ones or v.v.). Each of these costs can be determined with a cost number between 0 and 100.

$C_{adapt}(P_i, P_j)$ results from the endangering of a connection (P_i, P_j) caused by the current routing situation. $C_{adapt}(P_i, P_j)$ is calculated as follows:

$$C_{adapt}(P_i, P_j) = \frac{100 * N_{free}}{N_{all}} \quad (2)$$

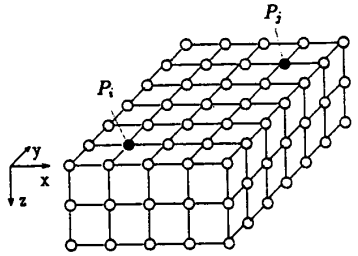

Figure 6: Volume of $V(P_1, P_2)$

where N_{free} is the number of grid points of a volume $V(P_i, P_j)$ which are neither occupied by other already routed nets nor by blocked or unroutable areas. N_{all} is the number of all existing grid points in $V(P_i, P_j)$. $V(P_i, P_j)$ contains the volume which encloses both points (P_i, P_j) with an additional grid unit and with a height resulting from the number of layers (see figure 6).

The two pins (P_i, P_j) with the lowest cost $C(P_i, P_j)$ of their connection have the highest priority to be routed.

3.4.3 Routing procedure

The routing procedure is carried out in 3 passes. In each pass only the failed connections are handed over to the next pass.

Pass 1: Routing according to the preferred directions Among all possible combinations of pin connections (P_i, P_j) of a certain net n the one $(P_i, P_j)_n$ which has the lowest cost function $C(P_i, P_j)_n$ is chosen. These pairs of pins $(P_i, P_j)_n$ of each net n are stored in a list according to their cost value $C(P_1, P_2)_n$. The pair $(P_i, P_j)_n$ with the highest priority (i.e. the lowest $C(P_i, P_j)_n$) is routed next.

First a simple line-search algorithm is applied, which is very fast:

1. If the $x-$ *and* $y-$co-ordinates of P_i and P_j are the same and the maximal via depth is great enough the connection will be realized by a via.

2. If the $x-$ *or* $y-$co-ordinates are the same the connection will be realized by a straight line on the layer of the suitable preferred direction.

3. If neither the $x-$ nor the $y-$co-ordinates are identical the enclosing rectangle is calculated. The connection will be tried along the straight lines of this rectangle considering all possible layers.

With this line-search algorithm non-routable connections (P_i, P_j) are handed over to a grid-based algorithm based on a modified Lee algorithm. The shortest path between P_i and P_j is found by the backtracing after an extension phase. If this connection fails P_i and P_j will be considered in the next pass.

The algorithm calculates a new pair of connecting pins $(P_i, P_j)_n$ by examination of $C_{user}(P_i, P_j)_n$ of all remaining pins of the net n. Note that only P_i must be a pin of n whereas P_j can be any point of the already routed part of n to which $C_{user}(P_i, P_j)_n$ has the lowest value.

The cost $C(P_i, P_j)$ and thus the order of all connections (P_i, P_j) in the list is modified according to the *current* routing situation. Afterwards the new pair $(P_i, P_j)_n$ is inserted in accordance with its cost value $C(P_i, P_j)_n$.

In Pass 1 $(P_i)_n$ is used as target cell for the extension cycles. Start cell is every point of the already routed net n including $(P_j)_n$.

Pass 2: Potential routing The routing algorithm now uses the pins $(P_i)_n$ which are not connected yet as start cells for the extension phase. Target is any point of the same net n. This enables the algorithm to connect these pins $(P_i)_n$ with each other. This increases the possibility of a complete-routed net within the next routing steps.

Pass 3: Routing without preferred directions The preferred directions must be regarded during pass 1 and 2. The preference of directions can now be ignored if this connection failed within pass 1 and 2.

It is known that backtracing is ambiguous. This is used in case of violating the preferred directions in the extension phase. During backtracing, the preferred direction determines the direction of the preceding and the following cells. So segments which are routed against the preferred direction of the layer are minimized in an effective way.

3.4.4 Rip-up and reroute

The rip-up and reroute algorithm calculates a connection between two non-connected pins $(P_i, P_j)_n$ which causes a minimal number of rip-ups of already routed nets. The path search between $(P_i, P_j)_n$ is done by a modified Lee algorithm.

First the algorithm calculates the grid cells of the same potential (i.e. of the same net n) connected with $(P_i)_n$ to create the start cells. Then an extension phase is carried out whereby different potentials

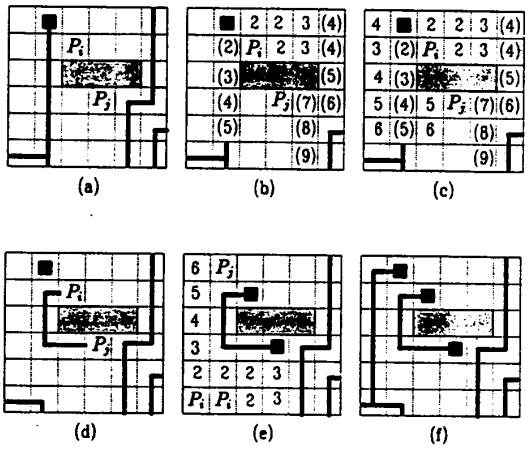

Figure 7: Rip-up and reroute example

(i.e. other nets) can be "climbed". But the further extension caused by these "climbed" cells has to be done only *on* the different connection up to its next pin or next Steiner point (see figure 7 a and b). The extension is done until no new extension cells can be created. In this case all extension cells located on connections are converted into start cells and put in order according to their extension value. The extension starts now with the cell(s) with the lowest extension value (see figure 7 c). Only these cells located on connections are involved in the extension the value of which is equal to the current extension value.

Different connections can be "climbed" again during the extension phase. If there are no free cells left the expansion starts again with the cells located *on* connection(s) according to their order.

The number of expansion cycles (i.e. the number of cut connections) is chosen by the user. The algorithm will stop the expansion if the target cell is reached or the number of cut connections reaches the limit. The connection can not be routed in the latter case and the previous routing situation is created again.

The backtracing is started after the target cell is reached. The connections which are "climbed over" are ripped-up until the next pin or Steiner point (see figure 7 d). These connections are now routed again (see figure 7 e and f). If this routing fails in more than one case the routing result would be worse than before and so the previous routing situation (with *one* open connection) will be restored.

If the rerouting fails in one case a new rip-up and reroute procedure will start. Previous routed connections are labelled to prevent a reciprocal rip-up. The

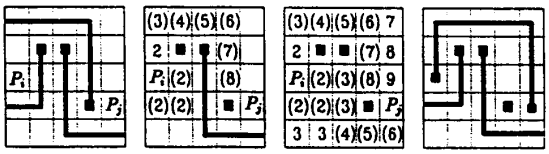

Figure 8: Minimization of the number of ripped-up connections

algorithm stops when all the connections are routed successfully or a limit of computing time is exceeded.

Figure 8 illustrates the reason for "climbing over" a connection only if any other extension is impossible. It guarantees a minimal number of ripped-up connections even if the routing connection needs a longer routing length.

Nevertheless, the routing length is minimized as shown in figure 7. This is caused by the characteristic of the algorithm to "climb down" from a (now cut) connection in the order of the extension values of the "climbing". It also guarantees that the cutting length is as small as possible. This increases the chance of success of rerouting this cut connection.

4 Implementation and results

The procedure was implemented in FORTRAN 77 on an AT Personal Computer.

Table 2 shows the routing results of 3 different multi-chip modules. The running times are calculated on a computer with a CPU-frequency of 16 MHz. We used 4 layers for routing for every multi-chip module.

We reached different routing results with the global routing algorithm. The bad global routing result of the module TU1 results from a placement structure which did not consider sufficiently the generation of channels between the chips. Note the short running times which allow an effective examination of placement structures even within the placement procedure.

Our experimental results show that there is no direct correlation between the routing results of the channel algorithm and the grid-based one. So, we conclude that channel routing is preferable only if the placement structure enables the generation of regular channels. In all other cases the grid-based algorithm is more effective using the channel routing just as a fast placement estimation.

We achieve excellent results by combining both algorithms using the channel algorithm for routing the power/ground nets (straight connections with less bends) and the grid-based one to route the signal nets afterwards.

	S6009	MC29	TU1
Number of chips	21	27	17
Number of pins	481	374	368
Number of external pins	133	112	112
Number of signal nets	167	124	118
Maximal number of pins per net	9	11	6
Size of the module [mm^2]	50x50	50x50	35x35
Global routing result	97 %	100 %	93 %
Not routed connections	8	0	16
Center-to-center track spacing [mm]	0.3125	0.3125	0.175
Number of channels per layer	540	980	529
Running time	2 min	3 min	2 min
Final routing result			
Number of line segments	1191	1092	/
Number of vias	1033	945	/
Route length [mm]	4737	4017	/
Running time	113 min	92 min	/
Grid-based routing result	100 %	99 %	100 %
Not routed connections	0	1	0
Size of the grid [mm]	0.625	0.625	0.350
Number of line segments	1374	1148	934
Number of vias	1297	896	874
Route length [mm]	4646	3658	4106
Running time	19 min	25 min	38 min

Table 2: Routing results

References

[1] J. M. Ho et al., "Layer Assignment for Multi-Chip Modules," *IEEE Transactions on Computer-Aided Design*, Vol. 9, No. 12, pp. 1272-1277, Dec. 1990.

[2] J. Freund, "Placement Algorithms for Multi-Chip Modules," (in German), Ph.D. Thesis, Dresden, University of Technology, 1990.

[3] J. Dammköhler, "ELANA - A System for the Electrical Analysis of Hybrid Circuit-Layout in Computer-Aided Design," (in German), Ph.D. Thesis, Technical University Ilmenau, 1990.

[4] H. Kühn, "Investigation on the Heat Transport Problem of Multi-Chip Modules Using a Finite Element Program," (in German), Ph.D. Thesis, Dresden, University of Technology, 1988.

[5] C.Y. Lee, "An Algorithm for Path Connections and its Applications," *IRE-Transactions on Electronic Computers*, pp. 346-365, 1961.