DOI: 10.51790/2712-9942-2020-1-1-2

THE CONCEPT OF MULTISTAGE INTEGRATED CHIP ROUTING WITH VIRTUAL CHANNELS

Sergey E. Vlasov[1], **Maksim M. Godovitsyn**[2,a], **Nikolay V. Starostin**[2,b]
[1] *Federal State Institution "Scientific Research Institute for System Analysis of the Russian Academy of Sciences", Moscow, Russian Federation, vlasov@niisi.ru*
[2] *Lobachevsky State University of Nizhni Novgorod, Nizhniy Novgorod, Russian Federation*
[a] *maxim.godovicyn@gmail.com,* [b] *nvstar@mail.ru*

Abstract: the study solves the problem of circuit routing in ICs. We propose a multistage approach based on interconnection mesh reduction. It expands the chip interconnecting capability by introducing extra channels. The interconnection mesh reduction significantly accelerates the existing routing algorithms. The introduction of more channels makes it possible to reduce the routing problem in a complex interconnect space topology by removing the new routes from the extra (virtual) channels.

Keywords: circuit routing, integrated circuit, multistage method, virtual channels.

Cite this article: Vlasov S. E., Godovitsyn M. M., Starostin N. V. The Concept of Multistage Integrated Chip Routing with Virtual Channels. *Russian Journal of Cybernetics*. 2020;1(1):8–15. DOI: 10.51790/2712-9942-2020-1-1-2.

КОНЦЕПЦИЯ МНОГОУРОВНЕВОЙ ТРАССИРОВКИ ЦЕПЕЙ ИНТЕГРАЛЬНЫХ СХЕМ С ИСПОЛЬЗОВАНИЕМ ВИРТУАЛЬНЫХ КАНАЛОВ

С. Е. Власов[1], **М. М. Годовицын**[2,а], **Н. В. Старостин**[2,б]
[1] *Федеральное государственное учреждение «Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук», г. Москва, Российская Федерация, vlasov@niisi.ru*
[2] *Нижегородский государственный университет, г. Нижний Новгород, Российская Федерация*
[а] *maxim.godovicyn@gmail.com,* [б] *nvstar@mail.ru*

Аннотация: рассматривается проблема трассировки цепей интегральной схемы. Предлагается многоуровневый метод, в основу которого положены идеи редукции сетки трассировки, а также расширения коммутационных ресурсов кристалла за счет введения дополнительных каналов. Редукция сетки трассировки обеспечивает существенное сокращение времени работы известных алгоритмов трассировки. Введение дополнительных каналов дает возможность свести задачи поиска трассы в сложной топологии коммутационного пространства к задаче вытеснения построенных трасс с дополнительных (виртуальных) каналов.

Ключевые слова: трассировки цепей, интегральная схема, многоуровневый метод, виртуальные каналы.

Для цитирования: Власов С. Е., Годовицын М. М., Старостин Н. В. Концепция многоуровневой трассировки цепей интегральных схем с использованием виртуальных каналов. *Успехи кибернетики*. 2020;1(1):8–15. DOI: 10.51790/2712-9942-2020-1-1-2.

Introduction

Circuit routing is an integrated part of the typical standard cell integrated circuit (IC) design process. Its purpose is the development of the conductive tracks topology. The tracks connect the contact pads of the circuit components. Each circuit is a continuous system of conductors, commonly referred to as a track. In any acceptable connection topology, no conductors belonging to different tracks may contact each other.

The key routing problems are high dimensionality and complexity of circuits, miniaturization of chip elements, which results in placing more items and the chip packing density increase. Considering the NP-completeness [1] of the routing problem and focusing on the signature large dimensionality of the actual problems, the requirements to applied routing algorithms increase. The situation is complicated by the fact that for submicron chips the interference of adjacent tracks should be taken into account. So, despite the variety of available methods and tools for solving this class of problems, developing a fast and high-quality router tool is a relevant task.

This paper presents a multistage routing method. It uses not one but many interconnection meshes to reach the balance between the solution search performance and quality.

The Routing Problem

The input data for the routing problem are layers, horizontal and vertical tracks that form the interconnection mesh; no-metallization areas on the interconnection mesh including grounding and power supply circuits; contact pads of the chip components assigned to the interconnection mesh nodes; a list of the chip circuits indicating the contact pads; metallized tracks of some circuit subset. A track is a continuous system of metal conductors linking all the circuit contact pads. It passes only within the specified interconnection mesh avoiding the no-metallization areas and tracks of other circuits. The key routing task is developing the tracks of all circuits.

Exact and approximate routing algorithms are mostly of academic interest. They cannot be used in real life to solve practical problems because of the exponential growth of computational costs as the dimensionality increases.

The major useable heuristic routing algorithms can be divided into two classes. The first class consists of algorithms that search for solutions directly on the interconnection mesh. Such algorithms are channel routing, backbone routing [2, 3], Lee routing [4, 5], etc. The key issue with these algorithms is early track placement leading to deadlocks. As a result, it is not uncommon for a poorly located track to block the routing of other circuits.

The second class includes two-stage algorithms [6, 7]: first, tracks are placed on a continuous plane (making it possible to draw a third track between any pair of non-intersecting tracks.) Then they carry over the solution from continuous to discrete space of the interconnection mesh. In a sense, a circuit layout plan is developed in the first stage. Still, the basic routing deadlock problem at the second stage exists.

In real-life applications, algorithms of both classes often produce incomplete tracks. It is solved by repeatedly re-routing the tracks automatically (expert-assisted), either on the entire chip or in selected areas. Such solutions are costly and make the entire chip design process more expensive.

Fast Multistage Routing

To improve the algorithm performance, routing space is usually divided into large fragments (blocks, areas) and interconnect areas that connect the blocks with a mesh of macro discrete elements. At the global routing stage, only interconnect circuits linking elements from different blocks are implemented. At the detailed routing stage, local circuits are implemented within a single fragment. The key problem of such a two-level approach [8] is a progressive track placement. It greatly reduces the routing space available for placing subsequent tracks.

We propose to use the well-known multistage concept [9] for routing assuming that we can remove the negative aspects of the concept by manipulating the interconnection mesh, including its reduction and expansion.

With the multistage approach, we build a coarse interconnection mesh (interconnection mesh reduction.) It is generated by combining adjacent tracks (vertical or horizontal metallization tracks within the entire chip) into groups, with the same number of tracks in each (if possible.) Such groups form channels in the original interconnection mesh. The result is a new coarsened interconnection mesh, where each node and each edge corresponds to a certain fragment of the original interconnection mesh. Let us assign a potential number of tracks within the corresponding fragment to each edge of the mesh. We call this number “edge routing ability”. It is defined as the product of the number of tracks in the group and the number of metallization layers.

Similarly, we can perform a second reduction after the first one, and so on until the required number of reductions is reached. The result of the first stage is a system of interconnection mesh reductions described by graphs of different sizes and characterized by different levels of detail.

To route a single track, we propose to use a multistage Lee algorithm. First, it routes tracks on a coarse mesh without considering topological intersections of conductors from different circuits (pseudo routing) but taking into account the edge routing capacity. In this case, the resulting track, in terms of a rough interconnection mesh, is a continuous fragment of the original interconnection mesh. Lee algorithm builds a track within it.

It is quite obvious that at some stage the algorithm will not be able to find a continuous track within the selected fragment. In this case, the fragment is expanded (in the extreme case it can cover all the elements of the given mesh) and the solution is generated beyond the mesh fragment represented by the track found on the coarse interconnection mesh.

As a result, the solution found on the original interconnection mesh is locked and the corresponding routing capabilities of the coarse versions of the interconnection mesh edges are adjusted.

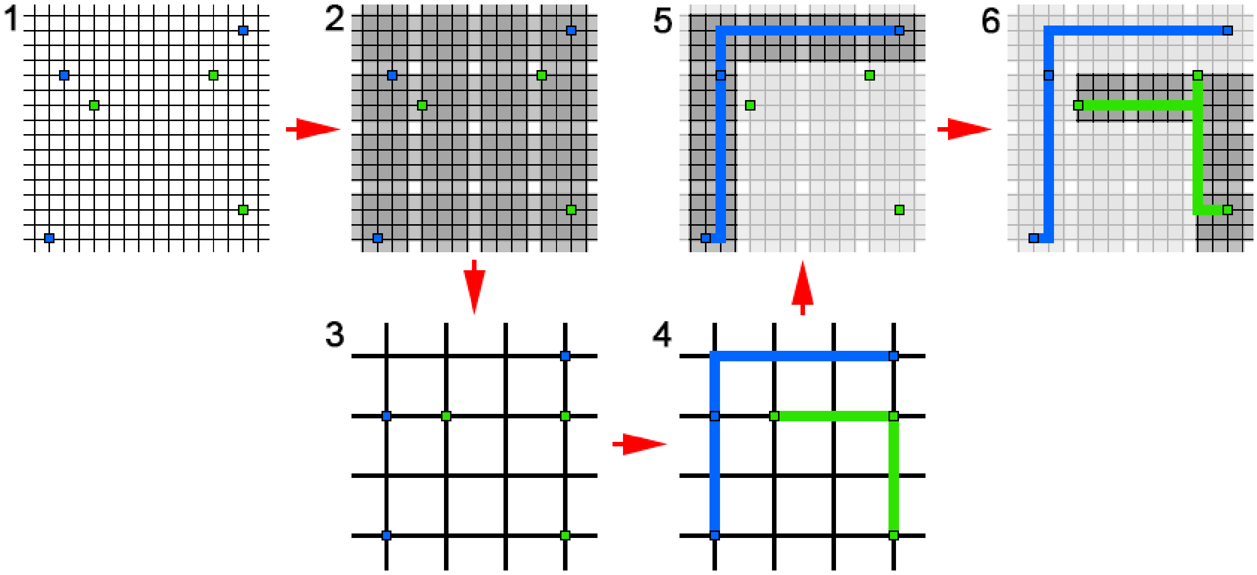


Figure 1. *Multistage Routing Steps*

The figure (refer to Fig. 1) is a chart of basic multistage routing stages (two levels are presented as an example.) The input data (stage 1) include the interconnection mesh, a list of circuits for each circuit, a list of contacts, and their location relative to the interconnection mesh tracks. Next (stage 2), the tracks of the original mesh are combined into groups (channels) and the interconnection mesh reductions are generated. The figure shows one reduction (stage 3.) In real life, there can be several, and their number is an input variable of the algorithm.

The reduced mesh is used for routing (stage 4.) At this stage, a general solution layout is generated. It is expressed as a system of tracks on the reduced mesh. At the same time, each such track describes a system of the original mesh fragments. With a high probability, the circuit would pass through these fragments. At this stage, the tracks can intersect, and their conflict-free configuration is produced as the reduced solution is carried over to the original mesh (stage 5.)

The general solution is progressively carried over to the original interconnection mesh (steps 5 and 6), then adjusted and optimized to meet the track topology requirements and to mitigate the interference of the adjacent conductors.

High-Quality Routing with Virtual Channels

The multistage approach described above accelerates routing. It uses the chip switching capacity uniformly. However, as noted above, the Lee algorithm in the multistage approach may fail to generate the desired track within a selected fragment on the original interconnection mesh. Let us focus on this problem.

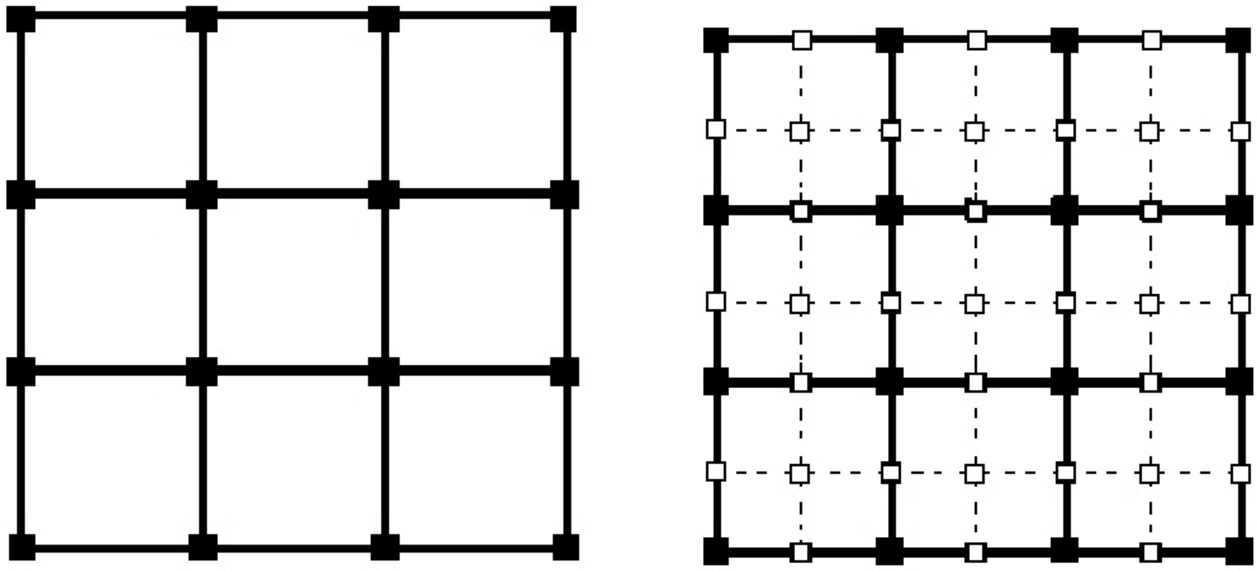
We analyzed various solutions for the incomplete track problem and came to some conclusions as follows. First, rerouting on the original mesh is usually inefficient. Second, the application of topological rerouting methods in metric space involves the problem of mapping a continuous solution onto discrete space. However, the generation of circuits in continuous space is limited only by the topology of the circuit graph (hypergraph.) If a solution exists, then an efficient algorithm for finding it also exists. The reason is the continuous space properties: it is always possible to draw a third track between any pair of non-intersecting tracks. We will use this feature to solve the re-routing problem.

To expand the switching capability it is proposed to introduce new tracks into the existing interconnection mesh. We will call them virtual channels. For this purpose, we will insert an additional virtual channel between each pair of adjacent existing tracks. The channel would to some extent represent the continuous space properties.

As a result, the rerouting procedure is as follows. In the first stage, virtual channels are added to the switching capacity and the partially generated tracks. In the second stage, an incomplete circuit is selected and its track is found. In the third stage, fragments of the tracks that coincide with the virtual channels are removed. The second and third stages are iterated over all the unsolved circuits. Let us consider all the proposed technology stages in detail.

Adding Virtual Channels to the Interconnection Mesh

To temporarily expand the chip switching capability we introduce extra (virtual) channels between each pair of adjacent existing tracks. Fig. 2 shows an interconnection mesh before (a) and after (b) the introduction of virtual channels.



a. *b.*

Figure 2. *Adding virtual channels to the interconnection mesh*

As a result, we get a new interconnection mesh (refer to Fig. 2) that has both real nodes (solid filled) and virtual ones (unfilled.) The latter is connected to the vertical and/or horizontal virtual channels. The interconnection mesh edges that are identical to the virtual nodes connected to two (vertical and horizontal) virtual channels are called “virtual” (dotted lines.)

Routing on a Mesh with Virtual Channels

It is proposed to apply the Lee algorithm to a single circuit routing on this mesh [5]. The algorithm is a BFS (breadth-first search) mesh pathfinding method. Given the nature of BFS, the Lee algorithm guarantees that its result has the min start-to-target path length. The Lee algorithm concept is simulating a wave coming from the source to target nodes, which are subsequently connected by a conductor.

A progressive multi-contact circuit routing is proposed. An initial contact pad is selected and permanently placed. Next, the wave routing algorithm is used to route tracks from all the non-metallized contacts to the metallized ones on this circuit. Each fragment found is added to the track. The last stage is repeated until all the contacts in the circuit are connected, or it is found to be impossible.

It is proposed to extend the classical Lee algorithm [8] to account for the weights of the interconnection mesh edges. The weights can be considered as “distances” between adjacent nodes. Let us limit the use of virtual channels by artificially increasing the weights of virtual edges on the interconnection mesh.

Removing Tracks from Virtual Channels

After successful routing on a mesh with virtual channels, it is required to remove the metallized tracks from all the virtual channels without violating the continuity of every generated track. For clarity, we will consider routing for single metallization layer chips.

Consider a special case when we want to remove metallization from a single node of the interconnection mesh while maintaining the integrity of the respective track. We will consider only track adjustment options when the metallization area is moved in a given direction from a given node to unoccupied adjustment nodes of the interconnection mesh. For the sake of certainty, at this stage, we will consider only 3 directions to move the metallization "up" (see Fig. 3.)

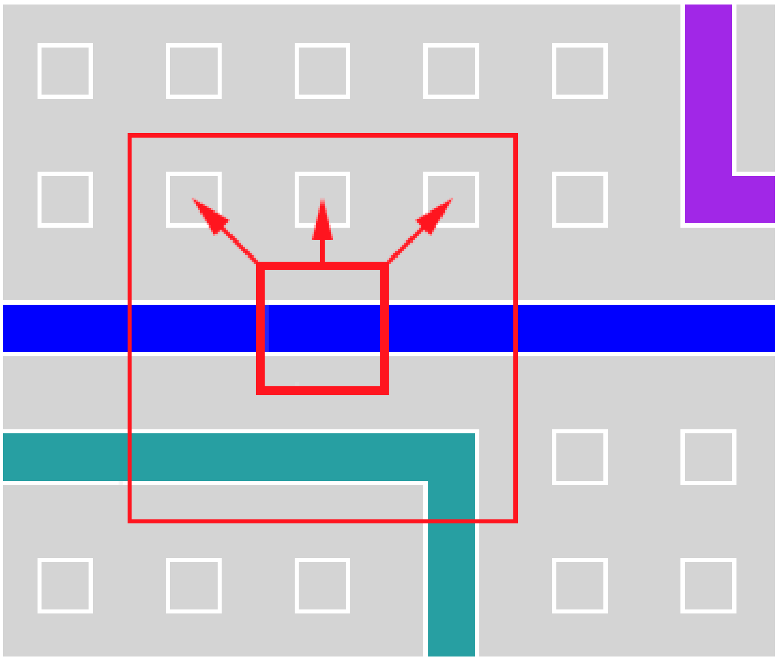


Figure 3. *A metallized mesh node and the directions of the metallization movement*

It is proposed to solve this particular problem with the NODE algorithm.

NODE Algorithm

1. Generate a new track configuration: the metallization is moved in the specified direction from the current node to all adjacent mesh nodes.
2. Check the track continuity. If the track is discontinuous, the partial problem has no solution. Restore the original track configuration exit with the "NO SOLUTION" result. If the track is continuous, go to Step 3.
3. Optimize the track removing excess metal as long as it does not affect the track continuity.
4. Check the new track for intersections with other tracks. If there are any intersections, the original track configuration is restored and the procedure is terminated with the "SOLUTION found" result indicating the list of conflicting mesh nodes.
5. The procedure is terminated with "SOLUTION FOUND" and a list of metallized mesh nodes is displayed.

NOTE. The NODE algorithm performance can be significantly optimized. We can build a database storing all possible track configuration options (256) and pre-configured solutions for each option. In this case, the algorithm is reduced to trivial retrieval of a pre-configured solution for a given configuration.

Consider a more general problem when we need to remove metallization for a single virtual channel of the interconnection mesh while maintaining the integrity of all tracks. To solve this problem, we will use the NODE algorithm for solving a specific problem. For clarity, let us assume that we need to remove metallization from a horizontal track. In this case, we should move the metallization from the virtual channel. It means that if the metallized mesh node is above the virtual channel, the metallization should be moved "up", otherwise "down".

TRACK Algorithm

```
1. Denote queue of nodes that require metallization movement as QUEUE. Add all metalized nodes adjacent to the metallization of the virtual channel to the QUEUE.
2. Check if metallization can be carried over from the virtual channels:
    a. Check if it is possible to move the metallization “up”: the NODE algorithm is run for each metalized node of the virtual channel. If the result for each node is “SOLUTION FOUND”, then exit with the result “SOLUTION FOUND”.
    b. Check the possibility of metallization carryover “down”: for each metalized node of the virtual channel, run the NODE algorithm. If the result for each node is “SOLUTION FOUND”, then exit with the result “SOLUTION FOUND”.
3. If the queue is empty, then exit the algorithm with the result “SOLUTION NOT FOUND”.
4. Retrieve a node from QUEUE. If the node has no metallization (it is possible), then go to step 3. Otherwise, go to step 5.
5. For a node, run the NODE algorithm. If the result is “NO SOLUTION”, go to step 3. If the result is “SOLUTION FOUND”, add the conflicting nodes to QUEUE and go to step 3. If the result is “SOLUTION FOUND”, go to step 6.
6. Generate a set of adjacent nodes extending from the found metallization nodes towards the virtual channel.
7. Add the set of adjacent nodes to QUEUE and go to step 2.
```

The finiteness of the TRACK algorithm is obvious: either all the metallized nodes of the virtual channel are removed (problem solved), or the node queue will eventually be empty (since the interconnection mesh is finite.) The proposed TRACK algorithm is easily transformed for handling vertical virtual channels.

The most general problem statement involves metallization removal from several interconnection mesh tracks while maintaining the continuity of each track. It is proposed to solve this problem by sequentially removing metallization from the virtual channels.

TRACKS Algorithm

```
1. No-metallization virtual channels are removed from the interconnection mesh.
2. Metallized virtual channels are checked one by one in some order. For each virtual channel:
    a. The TRACK algorithm is run. If the result is “SOLUTION NOT FOUND”, then exit with the result “SOLUTION NOT FOUND”.
    b. Remove the virtual channel from the interconnection mesh.
3. Exit with the result “SOLUTION FOUND”.
```

Computational Experiment

The algorithm described above was implemented in C#. A test application was developed to test the proposed multistage routing procedure.

Table 1 lists the results produced by the multistage routing algorithm with simulated test problems. The experiments were conducted on a PC with the following specifications: Intel i7 2 GHz / 16 GB, MS Windows 7x64.

Table

Test case				Run time (sec)	
Circuits	Contacts	Nodes per layer	Layers	Lee algorithm	Multistage algorithm
10^4	5×10^4	10^6	3	1,841	310
10^4	10^5	10^6	3	6099	855
10^4	5×10^4	10^7	3	2721	322
10^4	5×10^4	10^6	5	4155	582

Routing Algorithms Comparison

The multistage algorithm reduced the interconnection mesh once, reducing its size to 100x. As a result, the multistage algorithm is on average 700% faster than the classical Lee algorithm. It should be noted that the quality of the solutions produced by the presented algorithms differs slightly.

The screenshots below present an example of the routing algorithm with virtual channels.

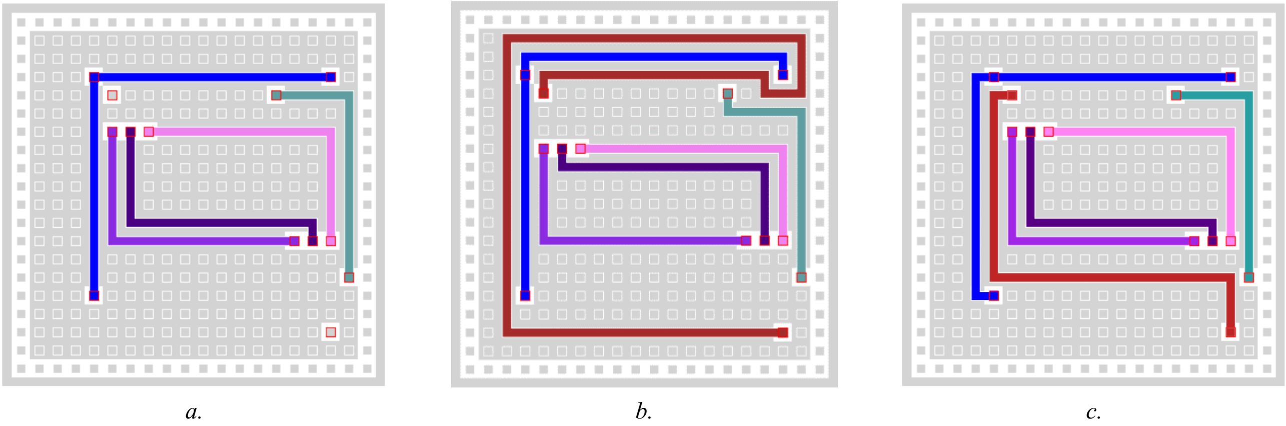


Figure 4. *Routing with virtual channels*

Fig. 4 shows an example produced by the routing algorithm with virtual channels. The image shows the initial data. Note that there is a no-route circuit for which the Lee algorithm fails to find a solution. Image b shows the result produced by the routing algorithm with virtual horizontal channels. Image c shows the result produced by the routing algorithm with the virtual vertical channels. In both cases, the circuit that was not routed by the Lee algorithm was successfully generated while maintaining the continuity of the originally routed tracks.

To summarize, we note that the experimental studies confirmed the assumption that the proposed multistage routing procedure is highly promising for real-life submicron IC design applications.

Conclusion

The study objective is improving the efficiency of IC circuit routing. We propose a multistage routing procedure that saves time and computational burden. The proposed multistage routing uses virtual channels and intended for better utilization of the available switching capacity. It is planned to test and fine-tune the presented algorithms as applied to solving real-life problems of submicron IC design.

REFERENCES

1. Garey M. R., Johnson D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979. 584 p.
2. Mikami K., Tabuchi K. A Computer Program for Optimal Routing of Printed Circuit Conductors. *Proc. IFIP Congress*. 1968:1475–1478.
3. Hightower D. W. A Solution to Line-Routing Problems on the Continuous Plane. *Proc. 6th Annual Design Automation Conference (DAC '69)*. 1969:1–24.
4. Hadlock F. O. A Shortest Path Algorithm for Grid Graphs. *Networks*. 1977; 7(4):323–334.
5. Lee C. Y. An Algorithm for Path Connections and Its Applications. *IRE Transactions on Electronic Computers*. 1961; EC–10 (3):346–365.
6. Hama T., Etoh H. Topological Routing Path Search Algorithm with Incremental Routability Test. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 1999; Feb:142–150.
7. Starostin N. V., Balashov V. V. The Use of Hypergraphs for Solving the Problem of Orthogonal Routing of Large-Scale Integrated Circuits with an Irregular Structure. *Journal of Communications Technology and Electronics*. 2008;53(5):589–593. DOI: 10.1134/S1064226908050185.

8. Batishchev D. I., Starostin N. V., Filimonov A.V. Two-Level Evolutionary Genetic Tracing of Electrical Circuits on a Graph Model. *Problems of Advanced Micro- and Nanoelectronic Systems Development (MES)*. 2008;1:61–64. (In Russ.)
9. Batishchev D. I., Starostin N. V., Filimonov A. V. Multilevel Genetic Algorithm for Hypergraph K-way Partitioning. *Proceedings of Saint Petersburg Electrotechnical University Journal*. 2007;1:3–13. (In Russ.)