V.M. Glushan, P.V. Lavrik, A.Yu. Lozovoy, M.V. Rybalchenko

DISTRIBUTED MODE OF TOPOLOGICAL DESIGN OF VLSI BY MEANS OF HIERARCHICAL CLIENT-SERVER ARCHITECTURE

Glushan V.M., South Federal Univesity, (SFedU),
Taganrog, Russia, e-mail:vmglushan@sfedu.ru,
Lozovoy A.Yu., South Federal Univesity, (SFedU),
Taganrog, Russia, e-mail: a.u.lozovoy@gmail.com
Lavrik P.V., OOO «LODOSS»,
Taganrog, Russia, e-mail: levarto@mail.ru
Rybalchenko M.V., South Federal Univesity, (SFedU),
Taganrog, Russia, e-mail: mic_v@mail.ru/

Abstract

Building advanced VLSI engineering design subsystems offers to use hierarchical client-server architecture. The conceptual analysis of such architecture focused on multi-level dichotomous partition of an undirected graph with various number of peaks and local degrees has been performed. Available from the studies the hypothesis confirms that the architecture allows to build VLSI engineering design subsystems characterized by higher performance compared to lumped subsystems.

Key words: VLSI, hierarchical client-server architecture, dichotomous partition, undirected graph, local degree, performance.

INTRODUCTION

Progressive technology development is intimately connected with its continuous loss of simplicity in a quantitative, as well as in a qualitative sense. Proliferation is mostly obvious in computer technologies. The first ENIAC ECM advertised by the end of the 1940s contained 17500 vacuum tubes, 7200 diodes, 1500 relays and could perform 5000 operations per second [1]. Dimensionality of today's VLSI achieves $1{,}5 \cdot 10^9$ and more chip transistors [2], and the performance is million instructions per second. As referred to in the work [3], without the use of VLSI CAD facilities designing a regular structure with less number of transistors, 10^5, required (in the mid-1970s) and wasted valuable time of 120 person years that is comparable with aircraft designing. If abiding by Moore's law [4], one of its statements reads that the number of transistors in a chip doubles approximately every two years, and this trend will continue, VLSI developers and manufacturers will have to chase the performance of CAD facilities for a rather long time.

There are 3 chronological approaches to be distinguished in the design-automation system speeding, namely 1970s and 1980s of the last century (multiprocessors reached their zenith), 1980s and 1990s (developing and applying hardware-based accelerators were under active study), 1990s and present moment (network information technologies). Among other information technologies client server computing gained traction.

The works [5-7] theoretically and experimentally prove that curcuit engineering design subsystems including VLSI can be built based on the distributed client-server technologies. The subject matter of the above papers is the subsystem speeding. The investigations reveals that the client-server architecture contributes to the performance compared to lumped design (in the same computer), but only by several times. Where the number of client computers depending on the complexity of the designed curcuit has an optimal value dwindling slowly with fast increasing of the designed curcuit, and the performance increases several times. The speeding of designing systems is not envisaged as sufficient. Thus there are still approach searches to build subsystems based on client-server architectures providing more significant performance improvement when VLSI design engineering.

I. SERIAL/PARALLEL AND PARALLEL-PARALLEL ARCHITECTURE CAPABILITIES

Using the architectures for distributed design guarantees time advantage by client/server design layering **[**6,7**]**. The layering can be performed in a number of different ways. The first subsystem version applies a model of distributed client-server architecture where the server decomposes the original curcuit, clients place-and-route curcuit elements in each part, the same server performs tracing between curcuit parts (pre-tracing). As the overall designtime for such model is determined by the sequential work time of client and server model parts, the maximum time advantage compared to a lumped subsystem can be obtained by tracing maximum possible number of links on client computers reducing server work in pre-tracing. The above model is said to be serial/parallel.

The above works show the real time advantage greater by times than in a lumped design subsystem. The obtained merit value seems to be limited and it is impossible to get higher values. As analysis given in [8] has shown that the serial/parallel model of distributed subsystem client server architecture possesses improvement reserves. The reserves can be implemented if a serial/parallel model is made into a fully parallel one. It requires that server does not wait for the clients quit but simultaneously, in parallel, executes interblock tracing. Implementing the above idea results in transforming a serial/parallel model into a parallel-parallel one of VLSI engineering design distributed system client-server architecture.

Implementing a parallel-parallel model requires solving the problem of interblock link pre-tracing, performed by the server, synchronous with the problem of curcuit element place-and-route on client computers. How should that be done if the tracing layout in various curcuit parts is not known before the clients quit? However, the topology of isolated traces on their boundaries is of primary importance. It simplifies the problem as you could define the curcuit side (a chip side for VLSI) to output the circuit extended in other curcuit parts. It is in much the same way as conductor strips output to an LRU (line replacement unit) connector or, circuits brought to VLSI processor perimeter primary outputs.

It is just the first stage. The second stage means that if interconnections between curcuit elements defined under the curcuit decomposition are known, the required number of channel backbones can be made preliminary, parallel with client functioning, as is done in channel tracing. This problem should be solved by the server have been out of action.

The experimental studies done in [8] allow for the conclusion that distributed subsystem parallel-parallel architecture is faster but at this stage the tracing quality turns to be lower and it conceivably resulted from the limited amount of investigational studies.

II. CONCEPTUAL BASELINE FOR THE HIERARCHICAL CLIENT-SERVER ARCHITECTURE

A particular kind of this (dichotomous) architecture is presented in Fig. 3. The name of the dichotomous architecture results from its basic architectural elements, namely, server, located on the previous level of a hierarchy, and two clients, located on the following level, informationally related to the indicated server.

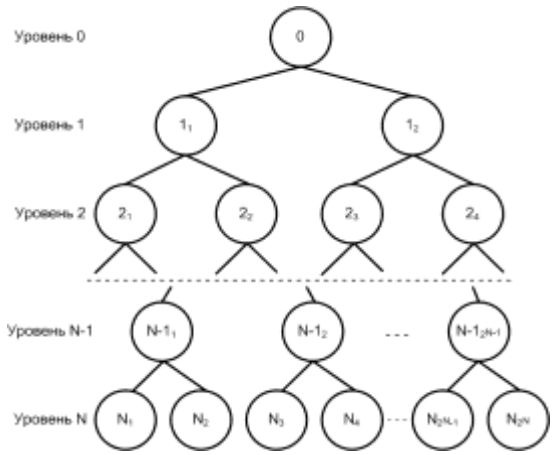

Fig.1. The hierarchical client-server architecture

Fig.1 shows a special case of the hierarchical architecture, each level applies 2^k computers. In general there can be any number of the computers. But the appliance of 2^k computers in particular is explained by two factors. First, as it is shown in [6,7], at a greater range of the curcuit complexity change the number of front-end computers is limited to two or three ones. Secondly, the conceptual analysis of the dichotomous architecture is easier than using another number of the computers. If this analysis demonstrates the efficiency of the hierarchical architecture, following research will contribute to the analysis of the general case.

The design process starts at zero level, where there is a C_0 master server at the top of the dichotomous hierarchical architecture. This server splits the original curcuit into two sub-curcuits and sends each subcurcuit to one of the two front-end computers of the first level, each being the servers for two computers of the second level. The computers of the second level consequently decompose their curcuit part into two sub-curcuits and send them to the computers of the third level. The process lasts until the relevant subcurcuits are received by the computers of the last *N*-level. The computers of the last level are only clients for the previous level.

After the computers of the *N*-level record the relevant parts of sub-curcuits, they start solving distribution problems and tracing tasks. At the end of the process place-and-route data from each pair of front-end computers of the *N*-level come to the respective server of the previous (*N*-1)-level. Each server of the (*N*-1)-level solves the problem emerging two parts of the curcuit of the *N*-level, tracing them. It results in a double part of the curcuit. As each pair of the computers of the (*N*-1)-level is connected to the respective computer of the (*N*-2)-level, every computer solves the problem emerging two double parts of the curcuit. As a result each computer of the (*N*-2)-level has a quadruplicate part of the curcuit. This process continues until the C_0 master server of zero level receives the data from two computers of the first level to create a complete final curcuit.

III. HIERARCHICAL ARCHITECTURE ANALYSIS

A nondirected graph has been applied as a diagram model, its nodes display curcuit elements and the edges of graph show connections (loops) between elements. Goal of the analysis is to define dependencies between the engineering design performance for circuits of diverse complexity and the number of hierarchical layers. The circuit complexity is governed by a number of the *M* nodes and the *R* edges in a modeling graph. Further, the number of edges varies with local degrees of the *L* nodes. Thus, source parameters of the circuit being modeled will be *M* and *L*. If all the nodes have an equal local degree, the number of simulating graph edges according to **[**9] is defined by the following short formula: $R = ML/2.$

The source quantity of simulating graph nodes and edges will be distributed along the computers of the hierarchical architecture tree. More natural and simplier procedure can be performed for the graph nodes equally dispersing them with all the computers of the layer. It is more complicated procedure for the edges even the local degree is equal for all nodes, as a graph partition set obtained will get the same number of nodes in subgraphs but different number of external edges, between subgraphs.

After the analysis of various approaches on the issue, as described in **[**6], the decision was taken to apply a percentage value of the number of external edges partitioning the graph into 2 subgraphs. Where the number of nodes in all subgraphs of the same layer on a hierarchy level and the number of external edges between subgraphs of each pair are taken to be equal. Thus, the following equations are applied as the analysis technique:

- At hierarchy zero-layer y_0 the total number of source graph edges (being interior edges) is determined from the following formula: $R_0 = ML/2.$
- At each subsequent layer y_i, $i = 1,...,n$ the number of external edges between each pair subgraphs and the number of interior edges in each subgraph are respectively determined from the following formulas:

$$R(i)_{внеш} = \alpha R(i-1)_{вн},$$

$$R(1)_{вн} = ... = R(2^i)_{вн} = \frac{R(i-1)_{вн} - R(i)_{внеш}}{2},$$

where α is percentage of the number of external edges between each subgraph pair from interior edgesin all subgraphs of *i* – layer.

For demonstrating the above technique consider the following example: let *M* = 1024, *L* = 6, $\alpha = 0,2.$ Hierarchy layer subgraph assignment of external and interior edges is the following:

Layer 1. $R(1)_{внеш} = 0,2R(0)_{вн} = 0,2 \cdot 3072 = 614;$

$$R(1)_{вн} = R(2)_{вн} = \frac{3072 - 614}{2} = 1229.$$

Layer 2. $R(2)_{внеш} = 0,2 \cdot 1229 = 246;$

$$R(1)_{вн} = ...R(4)_{вн} = \frac{1229 - 246}{2} = 492.$$

Layer 3. $R(3)_{внеш} = 0,2 \cdot 492 = 98;$

$$R(1)_{вн} = ...R(8)_{вн} = \frac{492 - 98}{2} = 197.$$

Layer 4. $R(4)_{внеш} = 0,2 \cdot 197 = 39;$

$$R(1)_{вн} = ...R(16)_{вн} = \frac{197 - 39}{2} = 59.$$

Using the results obtained a tentative complexity of the design process can be calculated applying the 1, 2, 3, 4 etc. layers for hierarchical designing. It is pertinent to note that algorithms of different time complexity can be applied in the design. On practical grounds, as it noted in [10,11], it is appropriate to use polynomial algorithms for allocating, as well as for tracing. In addition two cases will be discussed, both cases lean on squared complexity placement algorithms, the tracing algorithm of the first case is accepted as squared, the tracing algorithm of the first case is accepted as cubic.

The distinctive feature of hierarchical design lies in the fact that the placement algorithm will be applied at the last hierarchy *n*- layer. The design flow in such an architecture starts at the ultimate layer. Both (distribution and tracing) problems for each circuit part (for each subgraph in the case under consideration) should be solved. As far as the problem is solved simultaneously (in a parallel way) in all similar parts, the design complexity at the ultimate layer will be defined by distribution and tracing time of one part.

The distribution and tracing result obtained from two computers of the last layer will be transferred to the previous (parent) computer. It's an obvious point that the parent computer provides external linkage tracing between circuit elements brought from the recent layer. The same subcircuits will be built by two other computers of the ultimate level. So, at the penultimate (*n*-1)-layer the even number of duplicated subcircuits (two subgraphs) with the arranged elements will be formed. These duplicated subcircuits at the next (*n*-2)-layer will obtain a parent computer to route the coming subcircuits. The process will run in a similar way until it comes down to the zero-layer. Fig. 2 illustrates the above process.

Thus, time complexity counting, set as T_i, when design process starts from different layer, may be represented with the following practical relations:

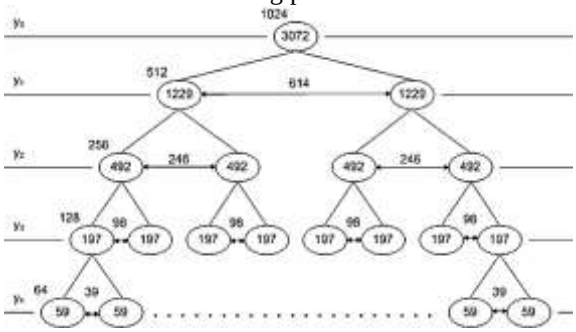

Fig. 2. The example of node and edge distribution in subgraphs at different layers

$$T_{n-i} = M^2(n) + R^2(n)_{вн} + \sum_{i=0}^{n-1} R^2(n-i)_{внеш} \quad (1)$$

Is for quadratic place-and-route, where $M(n)$ and $R(n)_{вн}$ are respectively the number of nodes and the number of edges in each subgraph at the hierarchical n-layer;

$$T_{n-i} = M^2(n) + R^3(n)_{вн} + \sum_{i=0}^{n-1} R^3(n-i)_{внеш} \quad (2)$$

is for quadratic allocation and cubic tracing.

Consider the usage sample of relations (1) and (2) to calculate the design performance of the hierarchical architecture basing on calculation data of subgraph external and interior edges at every layer as shown above for the case where M = 1024, L = 6, $\alpha = 0,2$.

For quadratic place-and-route, applying relation (1), we shall obtain the following:

$$\left.\begin{aligned} T_4 &= 64^2 + 59^2 + 39^2 + 98^2 + 246^2 + 614^2 = 456214; \\ T_3 &= 128^2 + 197^2 + 98^2 + 246^2 + 614^2 = 502309; \\ T_2 &= 256^2 + 492^2 + 246^2 + 614^2 = 745112; \\ T_1 &= 512^2 + 1229^2 + 614^2 = 2149581. \end{aligned}\right\} (3)$$

For quadratic allocation and cubic tracing, by a similar way, applying relation (2), we shall obtain the following:

$$\left.\begin{aligned} T_4 &= 64^2 + 59^3 + 39^3 + 98^3 + 246^3 + 614^3 = 247572466; \\ T_3 &= 128^2 + 197^3 + 98^3 + 246^3 + 614^3 = 254965429; \\ T_2 &= 256^2 + 492^3 + 246^3 + 614^3 = 365523504; \\ T_1 &= 512^2 + 1229^3 + 614^3 = 2088069677. \end{aligned}\right\} (4)$$

Subsequent to the calculation data (3) and (4), but for various values of M, L and α, respective performance curves are plotted shown in Fig.3-6. For all three curves of Fig.3 and 4 M = 1024, L=6, and α for curves 1, 2 and 3 takes the following value: 0,2; 0,3; 0,4.

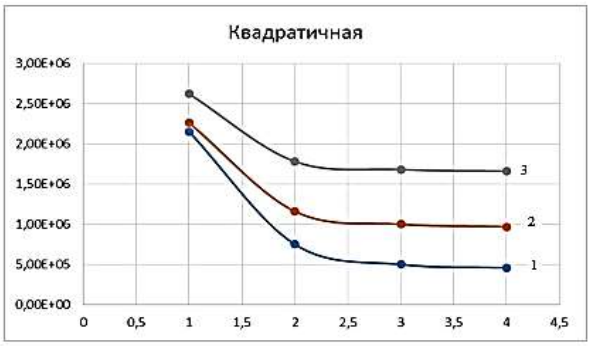

Fig. 3. The design performance with quadratic place-and-route

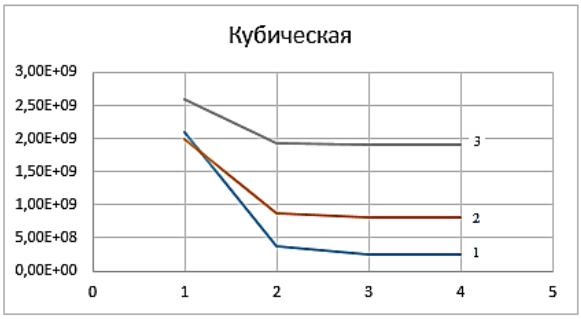

Fig. 4. The design performance with quadratic allocation and cubic tracing

Fig. 5 is built for M = 1024, L= 12, α = 0,2 and quadratic place-and-route. Fig. 6 is built for M = 1024, L= 12, α = 0,2 and quadratic allocation and cubic tracing.

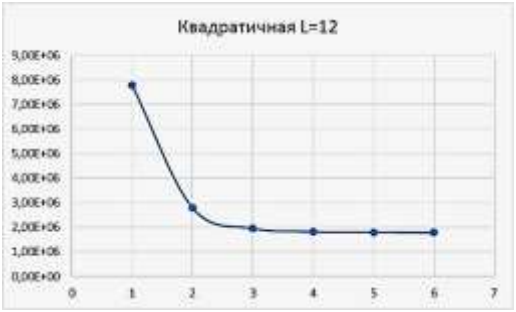

Fig. 5. The design performance with quadratic place-and-route and L= 12

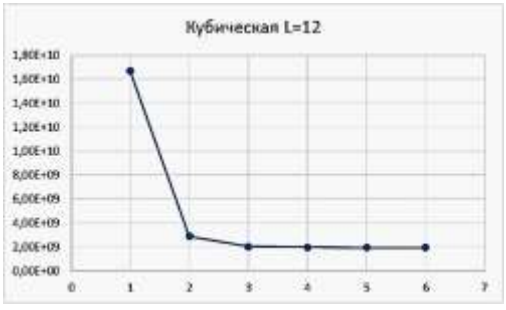

Fig. 6. The design performance with quadratic allocation and cubic tracing and L= 12

IV. CONCLUSION

Analysis of the above curves leads to the following conclusions:

- the design performance tends to decrease rapidly, regardless of the number of elements and saturation of circuit with connections (modeling graph edges) along with the increase of levels number;
- decrease of design performance along with the increase of levels number quickly stabilizes and even at three, four layers becomes unaltered;
- application of more sophisticated tracing algorithms (assuming higher quality results of their work) causes the increase of design effort, the latter, however, quickly decreases as compared to the less sophisticated algorithms with the increase in number of layers;
- design performance increases along with the increase in number of external connections between the parts of the circuit, although the decrease of the design performance, as the number of layers increases, is relatively slow.

From the above we can draw the following conclusion: hierarchical client-server architecture of building of the distributed subsystem design has potentially greater speed as compared to lumped subsystem. Therefore, a more detailed and deep study of it is of both scientific and practical interest.

[1].www.pochitat.com/pervyiy-kompyter-eniac-byil-zapyushhen-14-fevralya.html.

[2].S.V.Asmakov, S.O.Pahomov Hardware 2010. ComputerPress endorses. – SBR.: Piter, 2010. – 416 p.

[3]. E.E. Ivanov, V.N. Briunin. CAD VLSI Development Problems. Electronic Computer Facilities. Collection of articles. Publication 2. Under the editorship of V.V. Przhiyalkosky. M «Radio and Telecommunication». – 1988. p. 114-120.

[4].www.tadviser.ru/index.php/ Articles: Moore's law.

[5].V.M.Glushan, P.V.Lavrik Updates on Client-server Model of Distributed Circuit CAD. /News of Southern Federal University. Engineering and Industrial Technology Sciences. – 2009. – T. 12.

[6]. V.M.Glushan, P.V.Lavrik. Distributed CAD. Architecture and Possibilities. / Monograph. – Stary Oskol: TNT, 2014. – 188 p.

[7]. V.M.Glushan, P.V.Lavrik. Possibilities of the Distributed Topological Layout Design Subsystem built on Client-server Technologies. / Development Issues of Advanced Micro and Nano-electronic Systems. Collected works // Endorsed by member of Russian Academy of Sciences A.L. Stempkovsky. M.: IPPM RAN, 2014, part II.

[8]. V.M.Glushan, P.V.Lavrik. Parallel-parallel Model of Distributed CAD Client-server Architecture/ Congress Works on Intelligent Systems and «IS-IT'12 Information Technology». Scientific publication in 4 books. – M.: Fizmatlit, 2012. – B. 1.

[9]. O.Ore Graph Theory. – 2-nd edition. – M.: Nauka, Chief editorial board of Physico-mathematical Literature, 1980, 336 p.

[10]. V.M.Glushan., R.V.Ivanko Performance Analysis of Distributed CAD. News of TRTU. Special issue «Intelligent CAD». – Taganrog: TRTU publishing company, 2006, No 8.

[11]. V.M.Glushan, P.V.Lavrik Conceptual Analysis and Building Distributed Subsystem of Circuit Computer Aided Engineering. Intelligent Systems. Multi-authored monograph. Otput. 5/ Under the editorship of V.M. Kureichik. – M.: Fizmatlit, 2011. – 262 p.