Interconnection Length Estimation During Hierarchical VLSI Design

Axel Heß and Gerhard Zimmermann
Informatik, University of Kaiserslautern

Abstract

A lognormal interconnection length distribution function is derived from a large number of VLSI layouts. The assumption of a Weibull distribution could not be confirmed. Hierarchical slicing trees are derived from structural descriptions of the sample systems by recursive partitioning. Using the earlier proposed area estimation results, a new model for interconnection length estimation based on slicing trees is proposed, also resulting in a lognormal length distribution with parameters close to the layout results. We therefore believe that the results are a step forward towards understanding and predicting interconnection lengths and logical path delays before layout.

Introduction

In the realm of deep submicron chip technologies, performance becomes more and more determined by interconnection lengths at all levels of the layout hierarchy. Therefore, performance driven design primarily seeks to minimize interconnection related delays on the critical paths of digital systems. This effects placement and routing.

Since there are limits to the possible reduction of interconnection length, critical paths are also subject to optimization during logic design. For this purpose, possibly critical paths have to be known before layout. Therefore, the estimation of interconnection lengths prior to layout is an important task for reliable delay calculations during logic design.

Another application of interconnection length estimation is the control of performance driven layout with hierarchical methods. A critical path or even a single net on a critical path can be distributed over several levels of a hierarchy or several modules at one level. While the design of one module proceeds, estimates of the fragments of paths or nets in other modules are necessary to control this design.

Many other applications of such estimations can be found. All have in common that good estimations either improve the design quality or reduce the design effort by reducing the number and cost of design iterations.

This paper concentrates on the estimation of interconnection length on hierarchically composed ASICs in standard cell technology with optional macro cells. Off-chip interconnections are not considered. From layout experiments simple interconnection length models are derived and compared to experimental results. The models are not based on Rent's rule, but on our area estimation model that has been successfully applied to predict module areas during chip planning[1]. The paper is based on a Ph.D. thesis which contains more details and results [2].

But a word of caution is necessary at this point. While area estimation gains some of its precision from integrative effects, resulting in the averaging out of errors, interconnection length results are mainly maximum length values with very little averaging. Thus the prediction of the length of an individual net or path is nearly impossible within reasonable error margins. Instead we predict lengths distributions. From these distributions we can predict the probability of the maximal length of an individual net.

Basic Models

Let us simplify the problem by assuming that the digital system is *synchronous* and the performance directly depends on the path delay of the longest logic path between clocked storage elements. This defines the critical path. Clock schedules and distribution are not considered here.

The *path delay* is the sum of logic element delays and net delays. A *net delay* depends on the net topology and the length of the *net segments* and can be calculated using Elmore's or related models. Thus the problem can be specified as predicting the topology and geometry of all nets on the critical path.

In practice the critical path is not known a priory and therefore all paths have to be considered. With the above mentioned caution about results on individual nets we can expect a set of possibly critical nets using probabilistic methods.

We further assume a layout hierarchy of *rectangular cells* that recursively partition the chip. The leaves of the hierarchy are either standard or macro cells. Only *slicing geometries* are assumed. These restrictions are necessary in order to be able to model the geometry as a tree structure. Most layouts follow these restrictions. It has been shown that layouts outside of these restrictions gain only minor advantages that are not significant relative to the prediction tolerances.

Under these restrictions our *area estimation model* [1] provides some of the basic geometric parameters. It will be explained in a few paragraphs.

Starting point of all estimations is a hierarchically composed structural description of the system, as schematic diagrams or netlists. Each component of the structure hierarchy will be related to a component of the layout hierarchy.

The area estimation of a single component or module starts by recursive bipartitioning the structure, using any good mincut or ratio-cut algorithm. This results in a tree with the module as root and its subcomponents as leaves. Subcomponents can be either other composed modules, standard, or macro cells.

Partitioning tries to minimize the number of signal connections or nets between the children of the currently partitioned tree node. This results in a concentration of nets towards the leaves of the tree and thus in the smaller subtrees, as should be expected in a good layout. Since smaller subtrees are correlated to smaller areas, this results in shorter netlengths for as many nets as possible. Therefore, the tree represents features of a good layout

These features are of statistical nature. Different partitioning algorithms or executions will result in different trees. The final layout will most certainly result in another tree if the layout is sliced, also strongly depending on the chosen layout algorithm. On the other hand, the results of benchmark experiments with the leading layout algorithms show differences in total area of a few percent, which is less than the area prediction tolerance. We can therefore assume that any good partitioning algorithm will result in the same predicted area within the range of intrinsic tolerances. Our experiments have confirmed this assumption [2].

The next step is the estimation of the *shape function* of a node in the tree from the shape functions of it's children. Shape functions express the area values for different aspect ratios of a cell. It is represented by all possible length (x) and width (y) values of a rectangular cell, including added empty space in either x or y to extend a cell to a given dimension. If we define the shape function as the smallest possible y-values for a continuous range of x-values, a step function results.

In order to calculate the shape function of a node with two subnodes, the subnode shape functions have to be added. If the subnodes are placed on top of each other (horizontal common edge or cut line), the y-values have to be added for equal x-values. Otherwise x and y have to be exchanged. Since we do not know the orientation of the cut line, we execute both additions and select from the two resulting shape functions the steps with the smaller area. We call the result the optimal shape function. The optimal orientations are assigned to its steps.

We can calculate the shape functions of all internal nodes of the tree including the root, if the shape functions of the leaves are given. The sequence of calculation is bottom-up or from the leaves to the root. The same bottom-up sequence can also be applied in a hierarchy of modules, since the size of the bottom leaves, either standard or macro cells, are known.

If we select a shape for the root of the hierarchy, for example of the chip, we find the optimal orientation of the step of the corresponding shape function and, proceeding top-down, we can find the corresponding optimal orientations of all nodes. This brings us close to the topology of floor plans. The only missing parameter is the position of the adjacent subnodes relative to the cut line, top/bottom or left/right. This is called the *ordering*. The only way to determine this would be a complete placement. The estimations are therefore based on *oriented, unordered slicing trees*. This results in some of the estimation tolerances.

If we assume wiring channels between cells, channel space has to be added to the shape functions in x and y, resulting in additional tolerances. This is also the place where area and netlength estimation are closely related. We will therefore postpone this part until we explain the new model.

Previous work

We will only give a very short overview over the netlength estimation work. Many papers are based on the well known *Rent's Rule*: $P = k \cdot M^r$

Donath used it to calculate the upper limit of the average netlength for r>1/2 [3]: $\bar{R} \sim M^{r-1/2}$

More sophisticated models followed by Suen [4], Schmidt [5], and e.g. Feuer for r>1/2 [6]:

$$\bar{R} = \delta \cdot \sqrt{2} \cdot \frac{2 \cdot r \cdot (3 + 2 \cdot r)}{(1 + 2 \cdot r) \cdot (2 + 2 \cdot r)} \cdot \frac{M^{r-1/2}}{1 + M^{r-1}}$$

Because of the technology at the time, these models where mainly applied to PCBs. Because we need maximum values, the average is not sufficient to determine the length of the critical path.

Therefore, Sastry and Parker used Rent's rule to determine the netlength distribution in gate arrays [7]. A stochastic model resulted in a Weibull-distribution for the netlength which was experimentally confirmed:

$$F(x) = 1 - e^{-k \cdot x^r}$$

This results in an average value of:

$$\bar{R} = \frac{1}{r} \cdot \left(\frac{1}{k}\right)^{1/r} \cdot \Gamma\left(\frac{1}{r}\right)$$

The parameters r and k have been determined experimentally by fitting the curve to a measured distribution. The model was further improved by Gura and Abraham [8].

Sechen derives netlength distributions for two extreme cases. Either all cells connected by a net are randomly distributed over a square grid or clustered together as close as possible for an optimal layout [9]. Hamada et.al. refine this principle by topological distances and combine it with the Weibull distribution assumption [10].

Pedram and Preas finally determine the netlength in optimized standard cell blocks from assumptions about pin distributions and channel geometries [11]. In the last three approaches the actual netlists are used for the estimation.

Experimental Results

Because there is little experimental evidence about netlength distributions of real layouts under different conditions we executed many layouts. Despite this large effort it is impossible to say that the sample is sufficiently large. All statements made in this paper are therefore only valid in relation to this sample.

The sample consists of three VLSI chips in 0.7 micron standard cell technology. They have been designed in two levels of hierarchy. They are part of a high performance parallel Volume Visualization Architecture DIV^{2}A [12]. Table 1 summarizes the main parameters of the three chips.

The flexible blocks consist of standard cells. The macro cells are memory blocks from a generator. All layouts have been conducted using shape function generation, top down chip planning, cell synthesis, and chip assembly in the PLAYOUT system [13]. For the purpose of standard cell placement in the cell synthesis phase, Gordian [14] and Timberwolf [15] have been adjusted to and reimplemented in PLAYOUT to guarantee comparable results.

Table 1: Chip Parameter

chip name	# of flexible blocks	# of macro cells	# of nets at chip level	total # of std cells	total # of nets	avrg # of cells/ block	avrg # of nets/ block
adr	12	0	1484	7862	9033	655	752
vox	26	6	2030	14227	16899	547	650
seg	7	6	1178	3508	4232	501	604

In order to gain a large statistical variation, all standard cell block have been synthesized with both algorithms, different aspect ratios, and different pin distributions.

One of the first surprising results was the shape of the netlength distribution. We could not fit a Weibull distribution as we expected. Figure 1 shows measurements and the best fitting Weibull distribution for one of the standard cell bocks. The measurements summerize over 100 layouts. The length units on the horizontal axis is one standard cell height. All other blocks showed similar strong deviations. The Weibull distribution assumes too many very short nets that are not found in our standard cell layouts.

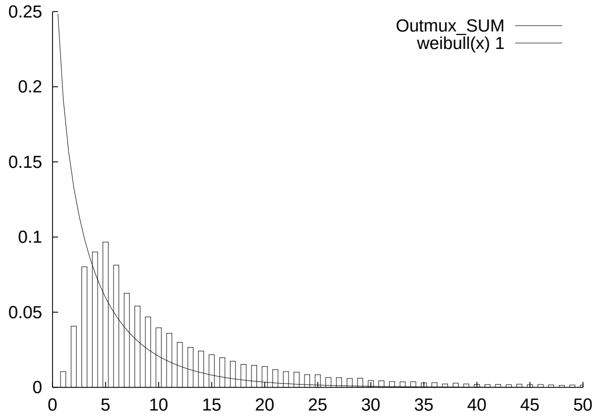

Figure 1: Measured netlength distribution and best fitting Weibull density function

We therefore tried to find a distribution function that fits our results. The result is a lognormal distribution with the density function:

$$f(x) = \frac{1}{\sqrt{2\cdot\pi}\cdot\sigma\cdot x}\cdot e^{-\frac{(\ln(x)-\mu)^2}{2\cdot\sigma^2}}$$

The two parameters can be determined with:

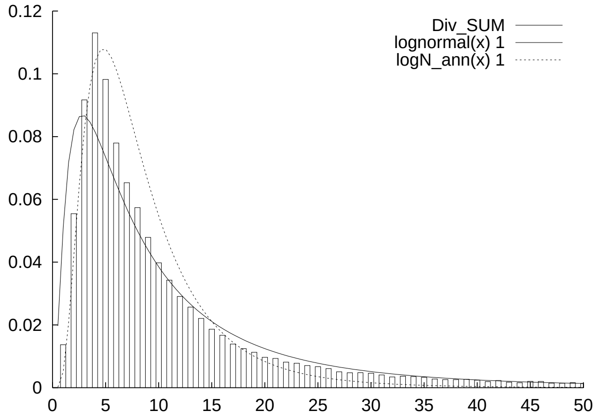

Figure 2: Measured netlength distribution and lognormal density functions

$$\sigma^2 = \ln(\frac{s^2}{m^2}+1) \text{ and } \mu = \ln(m) - \frac{1}{2}\cdot\ln(\frac{s^2}{m^2}+1)$$

m is the arithmetic mean of the measured distribution and s the variance. The resulting density function is shown as a continuous curve in Figure 2, together with measurements.

The curve fits long nets perfectly, but deviates for shorter nets. In order to improve the fitting for short nets we calculated the optimal parameters using a least square method. The result is the dashed curve which fits best in the short range but not so well in the rest of the range. Since the interesting range for critical path is that of long nets we decided to use the lognormal distribution with parameters calculated from average and variance.

Netlength Model

The next question is: How can we find a hypothetical model that will allow us to predict netlength distributions similar to the measured ones. This would be a step forward to understanding netlength as related to layout techniques. It should at least allow us to predict probabilities of upper limits of path lengths.

Starting information is the hierarchical structure description of the digital system. If we have done no chip planning or layout yet, we have no information about shapes and pin distributions with the exception of standard and macro cells. We have knowledge about the chip technology, e.g. the number of signal routing layers, and the available layout tools. We should also have knowledge about other layouts in the same technology with the same tools. The latter knowledge can be used to refine the parameters in the estimation models.

Routing Space Estimation

Before we can estimate netlengths, we need estimated layout dimensions. We already explained the basic optimal addition of shape functions, but postponed the addition of routing space.

For the purpose of this paper we assume two signal routing layers that are partially blocked over standard and macro cells. The layers are primarily assigned to the two routing directions. Here we also assume two hierarchy levels. The primitives of the lower level are standard cells. The primitives of the upper level are flexible cells (standard cell blocks) and macro cells, composing the chip. The extension to more hierarchy levels is no problem.

We further assume that a major part of the routing is channel based and the rest will use feedthroughs through or over cells. Feedthroughs are either intrinsic parts of the standard or macro cells, or result from empty space that is added to cells to create rectangular shapes in a slicing structure, or feedthroughs are added on purpose in standard cell rows for nets crossing rows. More than two routing layers can be modeled as additional feedthroughs. Unused feedthroughs are modeled as transparency. For standard cells we also assume equivalent pins on opposite sides. We distinguish row based and block based design styles.

As it was explained in section "Basic Models", the structure of a composed cell is recursively bipartitioned to create a tree. By optimal shape function addition and by adding appropriate routing areas, a slicing tree with shape functions at each node is calculated. The steps of the shape functions carry orientation information. Thus, for a given shape of the root, the orientation of all cut lines of the corresponding floorplan can be derived and a floorplan representative can be constructed with correct node dimensions but random ordering. Also, the transparency of each node is calculated.

We postponed the discussion of the wiring areas that have to be added to the node areas. Each net either uses up transparency or requires additional tracks in routing channels between cells. For this purpose we partition every n-point net into 2(n-1) segments and determine the node in the slicing tree which should provide track space for the segments. Since we do not know the ordering, the routing cannot be determined more precisely. Figure 3 shows a sample floorplan and its slicing tree with three different nets.

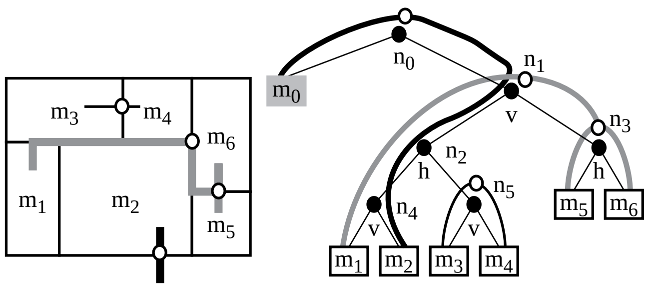

Figure 3: Definition of segments

Net (m_3, m_4) is a 2-point net crossing the vertical cut of node n_5 between two leaf cells. It is split into two segments at the node that represents it's cut line during partitioning (*cut line node*). This point is marked by an open circle. Cell m_0 represents the surrounding cells. Thus net (m_2, m_0) is an external net. Net (m_1, m_5, m_6) is split into four segments. Three of the segments connect a leaf cell to the corresponding cut line node. The remaining segment (n_1, n_3) is treated differently because both ends connect internal nodes. In any case routing space for segments is always added at the cut line node. In the case of external nets, tracks are to be supplied by the root node of the composed cell, in this case n_1, amd not by n_0.

The length of the segment and the space for the tracks are related. The width of a track is the routing grid and can therefore be represented by an integer track number. Since every node in the tree is - by the definition of slicing - a rectangular cell, the length of a track is either the horizontal or vertical dimension x or y of the node. It thus can be derived from the shape function of the node. A net segment on average does not need a full track, leaving room for more than one segment per track. This probability is expressed by a track demand factor between 0 and 1.

A segment will typically require parts of horizontal and of vertical tracks. The track demand factors for these two requirements will be different, dependend on the main direction of the segment in the floor plan. This is assumed to be orthogonal to the cut line node orientation. These factors also differ between different layout styles.

If a segment connects to a standard or macro cell, the connection point is a pin and therefore on the cell border. The positions of the pins are fixed and detours may be necessary. Standard cells and macro cells have to be distinguished because of different pin positions and equivalent pins. Also, segments in channels between standard cell rows differ from segments crossing rows.

If a segment connects to a flexible cell, it can be assumed that pin positions are adjusted in a top-down design process such that they minimize the segment length. In bottom-up design processes, flexible cells have to be treated as macro cells, since the best position for the pins cannot be determined.

These examples show that we are dealing with a number of different track demand factors. There is no room here to go into more details, but by some geometric reasoning we found a set that produces estimated shape functions that come very close to measured ones [2]. Further improvements could be achieved by adjusting the factors to sample layouts.

This short excursion into shape function estimation was necessary to understand the calculation of the representative floorplans. We first calculate the shape function of the root cell as explained and add space for the remaining external nets. This is the estimation of all possible actual shapes including all routing spaces. We select one of the shapes and construct a floorplan by randomly choosing the orientations. The routing space of the internal nodes is equally distributed to the corresponding child nodes. Thus all nodes and the leaf cells get all the routing space and thus actual node dimensions. This is not possible during the bottom-up shape function estimation process, but necessary to predict the lengths of the segments.

Segment Length Estimation

The reason for splitting nets into the small segments are delay calculations using RC-trees and Elmore's or similar delay models. The further feature sizes are reduced, the more important these models are. In the example in Figure 3 node n_3 is a necessary splitting point. n_1 is not necessary for delay calculation, but simplifies the model.

Let us start with the simple case of a net connecting two horizontally adjacent cells. Figure 4 shows the possible segment types on both sides of the cut line. Let the transparency of the cells be zero. There are only three types of segments: Type a connects to adjacent sides, type b to sides orthogonal to the cut and type c to opposite sides. By any combination of the different types we get nine net-types connecting A and B.

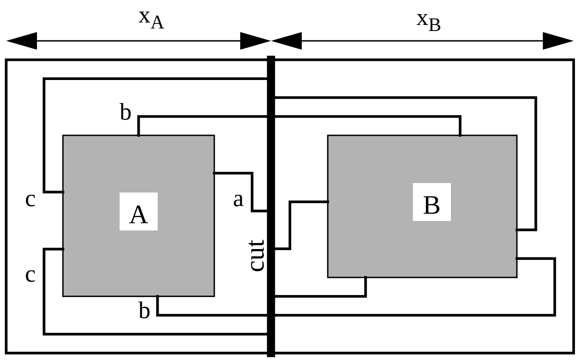

Figure 4: Segment types connecting adjacent cells

Depending on the type of cells A and B, the frequency and the lengths distributions of the segment types will differ. Standard cells (assuming horizontal rows) have no signal pins on vertical sides and thus no type a or c segments. The pins on the horizontal sides can be assumed to be equally distributed. The dimensions x_A and x_B are the same as A's and B's. For the purpose of this estimation we only compute average lengths $\overline{lx}_A = 0.5\ x_A$ and $\overline{lx}_B = 0.5\ x_B$.

For macro cells we assume the pins to be equally distributed on all four sides. It is obvious that the average length of type a and c together is also near 0.5 x_A or 0.5 x_B. We can thus assume an average segment length factor for nets orthogonal to the cut line of 0.5 for standard and macro cells.

For flexible cells as leaf cells we assume that the pins of nets connecting A and B are closer to the cut line. We have shown in [16] that for type b segments a factor of 0.33 can be applied, assuming a linear pin density function with the maximum towards the cut line. Also, assuming three times more segments of type a than of type c, a minimal factor of 0.25 could be justified for both together. Counting all together we assume a factor of 0.3 for flexible cells.

Similar factors can be derived for the average segment length parallel to the cut line. For type a nets the factors range between 0.33 for macro and 0.25 for flexible cells.

These considerations apply to very short segments in adjacent cells and just give a feeling for the type of model and the assumed factors. For segments connecting more distant nodes in the tree, we extend the model, as shown in Figure 5.

xn and yn are the dimensions of the cut line node's direct child node that contains the segment. The cell node has the dimensions xc and yc. We divide the segment in two parts for each direction. The first part brings the segment from a pin to the nearest point (black circle) of the hatched routing rectangle with the dimensions xr and yr. This part is cell type dependent. The above factors derived from Figure 4 can be applied. The second part within the routing rectangle only depends on the dimensions xr and yr. The horizontal length is xr, for the vertical length we only need 0.5 yr, because we distribute the length on both sides of the cut line.

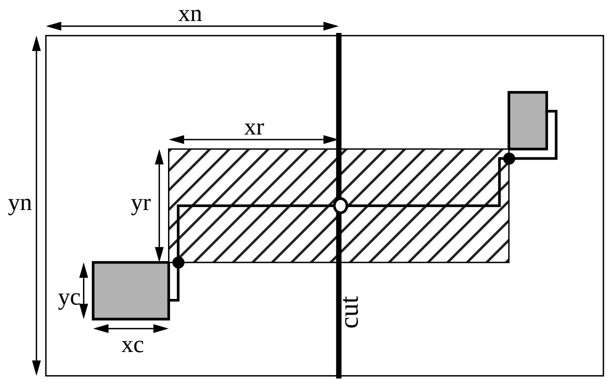

Figure 5: Segment model

For xr and yr we need average values. The maximum values xm=max(xr) and ym can be derived by subtracting the cell dimensions xc and yc from the enclosing node dimensions xn and yc, assuming the cells in opposite corners. The minimum values are zero. If we assume a distribution of the positions of connected cells relative to each other as either equally distributed or tending to be closer together, we assume $\overline{xr}$ = (0.5.. 0.3)xm and $\overline{yr}$ = (0.3..0.25)ym for vertical cuts, again using the above model. For horizontal cuts x and y are exchanged.

One special case has to be treated. If the cell is an internal node in the tree, an for example n_3 in Figure 3, other factors for the cell part of the segment have to be applied than for leaf cells. First, the segment has to reach into the node to connect to internal segments. This makes it longer. Second, the internal segments of the node are extended and will have points not far from the routing rectangle. Thus the segment will be shorter. We found values between the ones for macro and flexible cells to be appropriate.

The total segment length is the sum of the four average values. Each netlength is the sum of its segment lengths. Using this model and averaging over different partitionings and different aspect ratios, we achieve the netlength distribution shown in Figure 6. It is the same standard cell block as in Figure 2, only the average is taken over a smaller sample, explaining the larger variations. The similarity between both curves is relatively good for short nets and very good for long nets which are more important for critical path evaluations. The two curves are fitted in the same way as in Figure 2

A good figure for comparing measurement and estimation are the length limits l_{90}, for which 90% of the nets are shorter than the limit. Using l_{90} as a netlength in calculations we can guarantee with a probability of 90% that a netlength is shorter than this limit. In the same way we can calculate

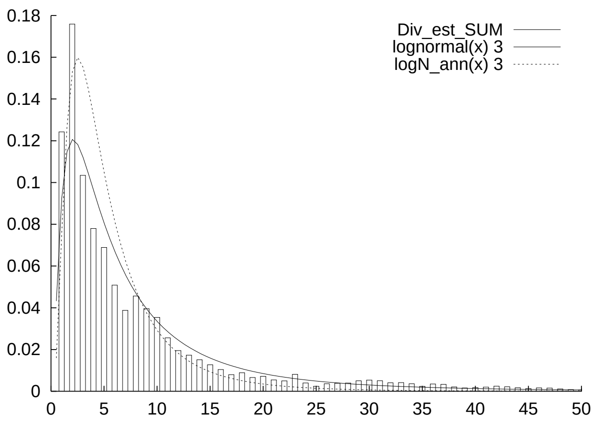

Figure 6: Estimated netlength density

limits for any given probability, once we have the two parameters of the estimated lognormal function. Table 2 shows some results.

Table 2:

cell	l_{90}_est	l_{90}_lay	l_{90}_lay1	l_{90}_lay2
Div	22	27	27	25
MLin	19	37	40	31
Myr	10	16	14	25
Nkr	28	22	19	31
Quad	15	18	18	18
Rgb	27	22	20	29

l_{90}_lay is the average over all layouts. l_{90}_lay1 and l_{90}_lay2 are averages over the results of two different layout algorithms to show how much results can differ using different tools. This puts the deviations between the estimated l_{90}_est and the layout values into proportion. It also shows that there is room for improvement by fitting the used factors to specific layout environments.

Similar estimation results can be obtained for segments, but we had no means to compare to measured segment lengths.

Conclusion

The main result of this experiment is the type of netlength distribution we found. The estimation model shows the same result. Both results may be typical for the chosen sample of circuits. But we believe that the result is typical for standard cell layouts.

The estimated length limit l_{90} seems to be pessimistic but a relative reliable prediction. It is based on the netlist and the estimated size of a block. It can be further refined, but there will always be a tolerance because there are large tolerances between different layouts of the same block.

We may derive x_{90} and y_{90} values for segments of nets for the purpose of RC-tree delay calculations. We could extract the estimated lengths of individual nets instead of only giving distributions, but such results would be very unreliable because they would be based on a specific partitioning. Further research would be necessary to find l_{90} values for individual nets assuming other partitionings.

The delay on logic paths is very roughly related to the sum of the net lengths on the path. If we use l_{90} values for each net, the sum will have a higher probability than 90%.

References

[1] G Zimmermann: "*A New Area and Shape Function Estimation Technique for VLSI Layouts*", Proc. 25th Design Automation Conf., p. 60-65, 1988

[2] A. Heß: "*Abschätzungstoleranzen im Hierarchischen VLSI-Entwurf*", Ph.D. Thesis, Univ. of Kaiserslautern 1999

[3] W. E. Donath: "*Placement and Average Interconnection Lengths of Computer Logic*", IEEE Trans on Circuits and Systems, Vol. Cas-26, No. 4, April 1979, p. 272-277, 1979

[4] L.-C. Suen: "*A Statistical Model for Netlength Estimation*", Proc. 18th Design Automation Conf., p. 769-774, 81

[5] D. C. Schmidt: "*Circuit Pack Parameter Estimation Using Rent's Rule*", IEEE Trans. on Computer-Aided Design, Vol. Cad-1, No. 4, October 1982, p. 186-192, 1982

[6] M. Feuer: "*Connectivity of Random Logic*", IEEE Trans. on Computers, Vol. C-31, No. 1, 1982, p. 29-33, 1982

[7] S. Sastry, A. C. Parker: "*Stochastic Models for Wireability Analysis of Gate Arrays*", IEEE Trans. on Computer-Aided Design, Vol. Cad-5, No. 1, p. 52-65, 1986

[8] C. V. Gura, J. A. Abraham: "*Average Interconnection Length and Interconnection Distribution Based on Rent's Rule*", Proc. 26th Design Autom. Conf., p. 574-577, 1989

[9] C. Sechen: "*Average Interconnection Length Estimation for Random and Optimized Placements*", Proc. International Conf. on Computer-Aided Design, p. 190-193, 1987

[10] T. Hamada, C.-K. Cheng, P. M. Chau: "*A Wire Length Estimation Technique Utilizing Neighborhood Density Equations*", IEEE Trans. on Computer-Aided Design of Integr. Circ. and Syst., Vol. 15, No. 8, p. 912-922, 1996

[11] M. Pedram, B. Preas: "*Interconnection Length Estimation for Optimized Standard Cell Layouts*", Proc. International Conference on Computer-Aided Design, p. 390-393, 1989

[12] J. Lichtermann: "*Eine Architektur zur Echtzeitvisualisierung von Volumendaten*", Ph.D. Thesis, Univ. of Kaiserslautern, 1998

[13] G. Zimmermann: "*PLAYOUT - A Hierarchical Design System*", Information Processing 89, G. Ritter (ed.), Elsevier Science Publ. B. V., p 905-901, IFIP, 1989

[14] J. M. Kleinhans, G. Sigl, F. M. Johannes, K. J. Antreich: "*GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization*", IEEE Trans. on Computer-Aided Design, Vol. 10, No. 3, S. 356-365, 1991

[15] C. Sechen, K.-W. Lee: "*An Improved Simulated Annealing Algorithm for Row-Based Placement*", Proc. Intern. Conf. on Computer-Aided Design, p. 478-481, 1987

[16] W. Hebgen, G. Zimmermann: "Hierarchical *Netlength Estimation for Timing Prediction*", Proc. 5th ACM/SIGDA Physical Design Workshop, p 118-125, 1996