Simulation Standard

TCAD Driven CAD

A Journal for CAD/CAE Engineers

Connectivity Options for DRC Spacing Check Operations in *Savage*

An IC layout has to satisfy many technological design requirements. One of these is spacing, i.e., a layout object has to be separated from another one by some minimal distance.

In addition, any spacing requirement (e.g. minimal distance between layout objects) varies depending on the particular features of the layout objects, e.g. the layer the objects belong to, the purpose of these objects, etc.

A rule set (or script) of DRC commands allows a designer to separate objects with particular features and then to check them according to technological requirements.

An important characteristic of a layout object is that it belongs to an electrical node as almost every geometrical object is a part of an electrical circuit, e.g. a wire, a drain, a gate, etc. The technological requirements for objects can be different depending on whether these objects belong to the same electrical node.

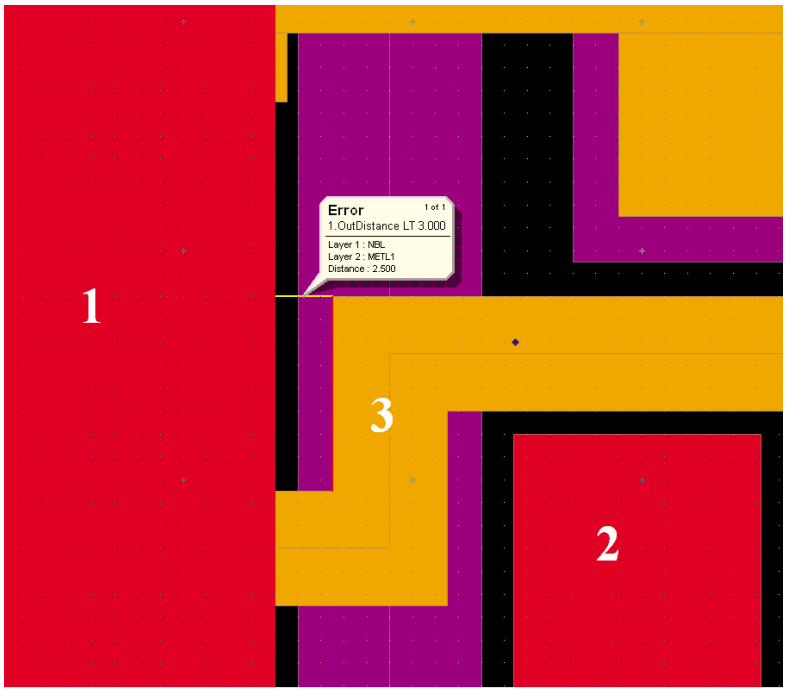

Figure 1. DRC Spacing Check operation with the option C'.Object 1 and 2 belong to the layer NBL, object 3 - to the layer METL1. Object 1 and 3 belong to the same electrical node.

The information that defines whether layout objects belong to the same electrical node is called "connectivity data". This data can be used along with any spacing requirements during the rule checking

If a designer is going to use DRC spacing check operations which take into account the connectivity information the option C or C' must be specified. The option C in a DRC check operation means the operation checks a spacing requirement only between the objects that belong to different electrical nodes. The option C' means the operation checks the spacing requirement only between objects that belong to the same electrical node.

Continued on page 2....

INSIDE

SILVACO INTERNATIONAL

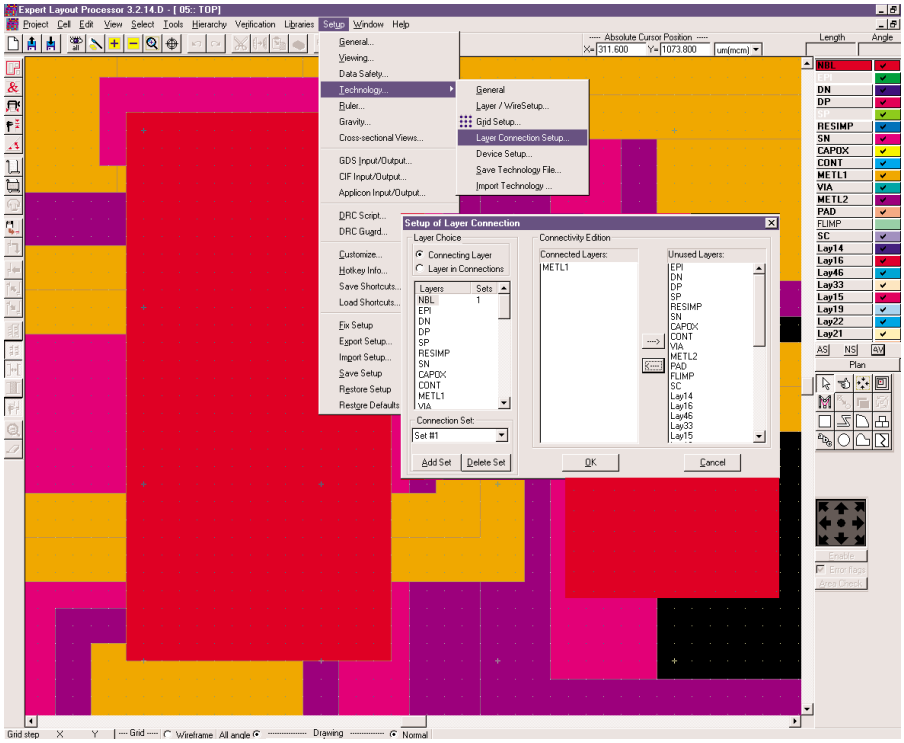

Figure 2. Dialog "Layer Connection Setup".

two steps, the designer can run a rule set which uses the check operations with the connectivity options.

If a designer tries to start a rule set with check operations that use the connectivity options but the connectivity information was not, then a window with a warning message appears (see Figure 4). This window suggests three choices:

- prepare connectivity information based on existing Layer Connection Setup and continue running rule set;
- ignore connectivity in commands; it means that the option C or C' will be ignored in DRC check operations;
- stop processing the DRC script

For example:

```
Outdistance: Options=(C',.), Layer1=NBL,
Layer1=METL1, .;
```

This operation will check the outer distance between every two layout objects, one of which belongs to layer NBL, another to layer METL1 and both these objects have to be in the same electrical node (see Figure 1).

An important stage of the usage of the connectivity information in DRC check operations is its preparation.

Firstly a designer has to define electrical dependencies between layers. This can be done e.g., from the user's interface: Setup>> Technology>> Layer Connection Setup (see Figure 2). The next step is to build the connectivity information: Verification>>Node probing>> Reextract Connectivity (see Figure 3). After the above

Continued on page 5....

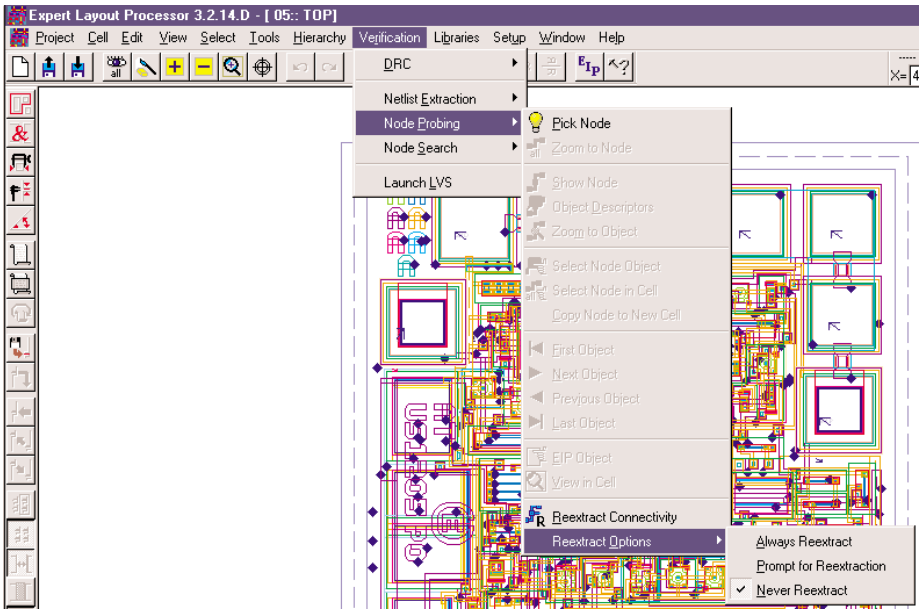

Figure 3. Menu Item to build connectivity information.

Recent Improvements in *CELEBRITY* Tools

Most of recent improvements in ***Expert, Savage, Guardian*** and ***Maverick*** are aimed at the enhancement of the usability of these tools. All features described below are developed according to the requests of our customers.

1. Improvements In Expert Layout Editor

1.1. Rotation by 90 or 45 degrees with respect to a selected point

This operation is made much easier. Now the "Edit>>Rotate" tool obeys the angle mode (Edit>>Angle Mode>>All/90/45). If the 90-deg. mode is on, then after clicking the center of rotation the cursor jumps to the four orthogonal angles only. Similarly, under the 45-deg. mode the cursor jumps to the 8 directions.

1.2. Chip Rover dumps coordinates in user units

"Dump" command of the ChipRover now writes the positions of cell instances in the current measurement units. Previously it wrote in database units.

1.3. Cell name length restriction for gdsII files

A warning is added when an attempt is made for writing a gdsII file with cell names longer than 32 characters. Some CAD systems fail to read such gdsII files.

1.4. Bindkeys instead of mouse clicks

The new command **Edit>>Left Mouse Click** is added. If a bindkey is assigned to it, this bindkey may be used instead of left mouse clicks when drawing the shapes and in some other operations.

At the same time, the **<Enter> key acts as the right mouse click** to terminate the drawing of polygonal shapes: region, wire, ruler and cutting line.

These replacements are especially useful if the mouse is not stable or you are using the tablet with the stylus.

1.6. License Releasing for Idle Application

"**SFLM idle time**" option is added to **Setup>>General**: if an application is idle during this time, then its license is released.

Zero value for idle time means that the license is released only when the application terminates

1.7. Infix Mode for Ruler

A new option is added to ruler setup: **Setup>>Ruler>> Infix Mode.**

In this mode, if the ruler is started by its bindkey, then the starting point is taken at the mouse position at the moment of hitting the bindkey (without mouse click). In this way, the measurement requires one mouse click less. If the ruler is started from menu, then you still need to click the first ruler point.

1.8. Duplicate/Move with Rotation/Flip

If you assign shortcuts to "Rotate-90", "Flip Vertically" and "Flip Horizontally", then you may click them in the course of the "duplicate" or "move" operation, so that the result will be a combined transformation of shift/rotation/mirroring.

Moreover, right mouse clicks during the duplication/move perform "Rotate-90".

1.9. Technology File Format Change: Layer Stipple Pattern Stored Directly

Notice: The new format of technology file can be still used with older versions of Expert: new fields will be ignored.

(This is a general rule for ***Expert*** technology files.)

Previously, technology files stored only the name of the layer's stipple pattern. Pattern itself was to be loaded from the currently active stipple library. As a result, a substitution of the current stipple library led to substitution of stipple patterns.

Now stipple libraries are used only for the initial selection of the pattern. Once it is selected, it is stored in the technology and may be substituted only from the Layer Setup.

In particular, this change of the technology file format enabled the following useful feature:

1.10. Loading/Saving Layer Style (Color, Filling) using Layer Plans

Expert Layer Plan files (*.elp) now store layer color and stipple pattern data, so that you can replace layer style without changing the technology file. This is done as follows.

- load a project;
- set your favorite color and stipple patterns;
- open the Layer Plan panel, either from menu View>>Layer View>>Layer Plan or clicking the "Plan" button in the Layer Bar;
- add a new plan named, e.g., "my_style-1";
- save it. In this example, by default it will be saved in a file "my_style-1.elp".

From now on, if you load a project with a compatible technology (i.e., with the same layer names), you may load the "my_style-1.elp" layer plan to set your layer style.

Note: When a layer plan is loaded, it does not change appearance of the layers not listed in it.

1.11.Default "FULL__" and "DATA__" Layer Plans

When a project is loaded, Expert automatically creates layer plans with the names "FULL__" and "DATA__". FULL__ layer plan shows all layers present in the technology. DATA__ layer plan shows only layers that contain geometry in the current cell.

These layer plans are automatically updated when you open a cell or select one of these plans from the layer plan drop-down list in the Layer Bar.

1.12. Xi-procedure for layer attributes (visibility/selectability)

A new procedure for setting layer attributes is added.

In this version only visibility and selectability may be set.

```
layer_set_view_attr("layer_name", LVA_VISIBLE, flag);

layer_set_view_attr("layer_name", LVA_SELECTABLE, flag);

 flag  may be TRUE or FALSE

If layer_name is an empty string, then visibility/selectability is set for all layers. Example:

layer_set_view_attr("", LVA_SELECTABLE, TRUE); Sets all layers selectable.
```

1.13. get_point(): xi-procedure for getting mouse position

If it is called from xi-script then it expects a mouse click in the layout.

It returns the POINT object with coordinate values in current measurement units for the cursor position. It takes into account the active reference point.

When get_point expects a mouse click, you may still use viewing commands (zoom in/out, pan, scroll) and ruler. If any other Expert's interactive operation is selected (e.g. box creation or stretch, etc.), then the return of get_point() is undefined.

Usage example:

```
P1=get_point();

P2=get_point;

Box (P1.X) (P1.Y) (P2.X-P1.X) (P2.Y-P1.Y);
```

If get_point()is called and rather than clicking the mouse you hit <ESC> key, the function will return a point with minimum coordinates. Their exact values depend on the current database unit and measurement unit values. For example,If DB unit is 0.001 and measurement units are microns, get_point() will return the point (-1000000, -1000000).

Notice that when get_point() expects the input of the point you can still use the following operations:

- viewing (panning, zooming)
- Ruler
- Pick Layer
- operations in the Layer bar

1.14. Control for same GDSII number/datatype in technology file

Control for correctness of technology data is enhanced:

If the option: **Setup>>Technology>>General>>**"Allow layers with same GDSII numbers and datatypes" is NOT checked, then operations **Project>>New** and **Project>>Load** that use a technology file with a pair of layers with the same GDSII numbers and datatypes will fail with the corresponding message.

In previous versions of Expert this control was performed only during technology creation / modification using **Setup>>Technology>>Layer Setup.**

This control prevents unintentional data change during export into the gdsII format: two or more Expert layers can be written into one gdsII layer.

If it is required to merge layers during gdsII export, it is recommended to use a layer remapping table, see Setup>>GDS I/O>>Output.

1.15. Replacing layer colors, stipples and GDSII numbers in technology

The commands below are a temporary workaround for some most common requests until the replacement of technology data without closing the project will be implemented.

The **View>>Layer View>>Layer Lists** submenu contains the following commands:

```
Load Layer Colors
Save Layer Colors
Load Layer Stipples
Save Layer Stipples
Load Layer GDSII Numbers/Datatypes
Save Visible Layer List
```

The "**View>>Layer View>>Layer Lists>>Save...**" commands save the colors or stipples, or gdsII numbers/datatypes of the visible layers into files with the extension *.ell (Expert layer list). (Layer visibility is set using the Layer bar).

Use these "**Save...**" commands to see the format of these files (a very simple one).

Notice that the stipple layer list uses stipple names and requires a proper stipple library to load stipples by names.

NOTE. Recall that the "Save Visible Layer List" command is useful for creating layer remapping tables.

2. Improvements in Savage DRC System

2.1. New commands in Savage DRC runset

2.1.1. Top_Cell: <Name of Top Cell>;
<Name of Top Cell> - the name of the cell that will be taken as top for the execution of the DRC runset for the given project.

This command must be placed before any executable commands in the script.

The DRACULA->Savage converter converts this command from DRACULA's command "**PRIMARY**".

2.1.2 Error_Limit: <Number>;
<Number> is the maximal number of errors reported by one DRC check command.

This command overrides the "Errors number limit" setting in Setup>>DRC Script during DRC script running.

The DRACULA->Savage converter converts this command from DRACULA's command "LIMIT-DRC-ERROR".

2.2. "Convert layer checks from DESCRIPTION BLOCK"

This option is added to "Verification>>DRC>>Script Panel>>Options".

If this option is checked then DRACULA's check commands from DESCRIPTION BLOCK

(FLAG-ACUTEANGLE, FLAG-SELFINTERS, FLAG-OFF-GRID, FLAG-NON45)

are converted into the corresponding Savage check commands for each input layer.

2.3. DRACULA's command "SELECT NOT ..."

This command (not supported before) is converted now into the corresponding Savage commands.

2.4. Options for DRC Progress Indicators

Options for DRC Progress Indicators existing in "Verification>>DRC>>Script Panel>>Run Options" dialog are added to Setup>>DRC Script to set Progress Indicator options permanently and to save/restore to/from configuration files and/or WinNT registry.

...continued on page 2.

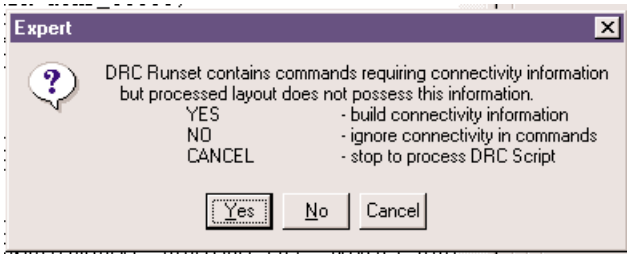

Figure 4. Warning Message.

Another way to prepare the connectivity information is to import a DRC rule set and a technology description from another (non-Expert) system. To do that, a designer needs to perform the following steps:

1. Convert a foreign rule set (e.g., `net.dsf'): Setup>>Technology>>Import Technology>> Netlist Definition (see Figure 5).
2. Reload the layout project with a new converted technology file (*.tcn).
3. Run a derivation rule set (net_*.dsf) to create all derived layers which do not permanently exist in the project.
4. Build the connectivity information (see above "Reextract Connectivity").
5. Run a DRC rule set (drc_*.dsf).

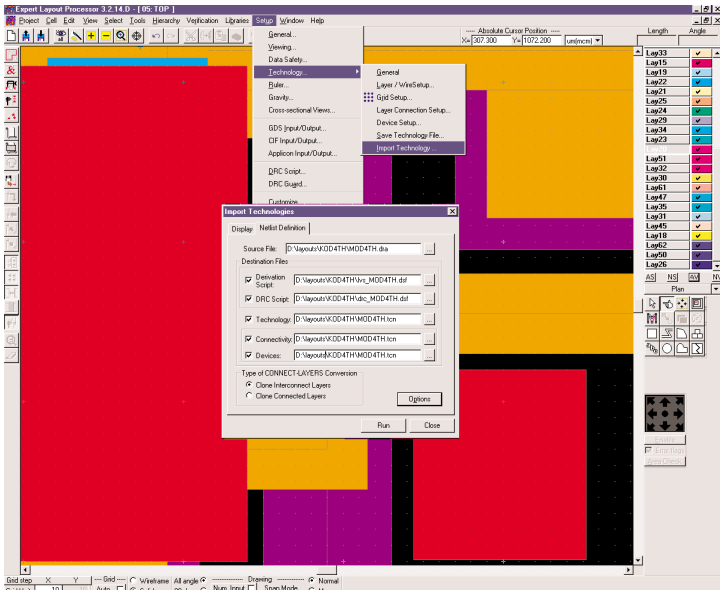

Figure 5. Dialog to Import DRC Rule Set with connectivity information.

Optimal Packing of Orthoblocks For ULSI Floorplanning

1. Introduction

Virtually all latest and prospective enhancements of ***Expert*** Layout processor (for example, semi-automatic floorplanning) are based on sophisticated geometrical models. In some cases some advanced mathematical research is necessary to develop efficient algorithms. This article is devoted to the problem of optimal packing of orthoblock which is originated from several areas of VLSI design automation.

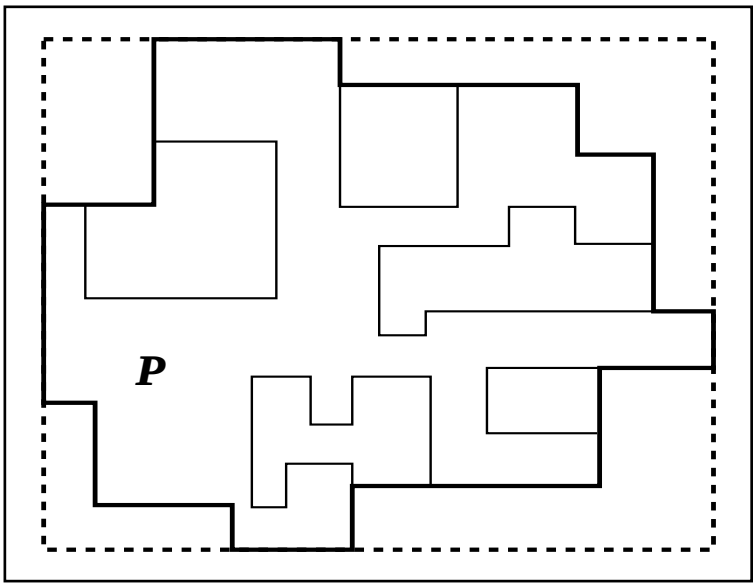

Figure 1. An orthogon *P*, orthoconvex hull of orthogon *P* (outlined with bold lines) and rectangular hull of orthogon *P* (outlined with dotted lines).

Cutting, packing and placement problems appear under different names in literature and come from various fields of production, design, engineering, planning and operations research. In spite of their variety, these problems have essentially much in common. H. Dyckhoff [2] was probably the first to take an attempt to develop a comprehensive topology integrating the various kinds of cutting and packing problems. Note this topology has not list its significance until now and we refer the reader to the survey [2] for a unifying framework of cutting and packing problems as well as for an overview of methods for their solving. Different special aspects of the cutting and packing topic related to our packing problems are also concerned in [6,9]. Our paper is devoted to an extension of the hierarchical merging method which has been used so far for solving placement and layout problems [1,3,4,10] in the plane to a special packing problem in 3 dimensions. Informally speaking, the problems is to pack given items (material pieces) into a container with a minimum area base. Our problem has the following features.

1. Unlike the problems considered in [2], no container is given in advance, but it is to be constructed (designed) as a result of solving the problems. Container design must meet the following requirements: – container must be cylinder, with its bottom being a rectilinear polygon (so-called) orthogon, see below;
   - container's geometric shape determined by the shape of its bottom should be "simple" enough in common sense;
   - material consumption to manufacture a container defined by its lateral area and, consequently, determined by the perimeter length of its bottom should be reduced;
   - the bottom area should be minimized to save the pallet space when loading containers on a pallet.

We define the container bottom to be an orthoconvex polygon (see below) to settle contradictions among the above container's properties by a proper compromise.

2. The item in our problem are assumed to be "thin", i.e. to be cylinders with rectilinear bases and "small" heights to provide the practical adequacy of our model. This implies that the items can be assumed to be planar objects, more precisely rectilinear polygons. Taking into consideration this circumstance and following Dyckhoff [2], we should speak about our problem as a 2.5-dimensional problems rather than 3-dimensional one.

Note at last we deal with arbitrary rectilinear items whereas a majority of papers published until now are concerned with rectangular items.

2. Packings With A Minimum Area Base

We will consider geometric objects that have axis parallel edges. A polygon with axis parallel edges is called an *orthogon* (Figure 1). An orthogon *P* can be given by its clockwise ordered vertex list *VList(P)* and the translation vector which determines its current position with respect to its initial one. An orthogon *P* is said to be *orthoconvex* (Figure 2) if the intersection *P* and a horizontal or vertical line is connected, i.e. is empty or is a segment.

Rank *r* of *orthoconvex* P is the minimal number of rectangles R1, R2, ...,Rr which do not intersect *P* and complete it to rectangle. In other words:

$P \cup (\cup R_i$, for i=1,2,...,r) is rectangle

$R_i \cap R_j = \phi$ for i,j=1,2,...,r, i !=j.

$R_i \cap R = \phi$ for i=1,2,...,r.

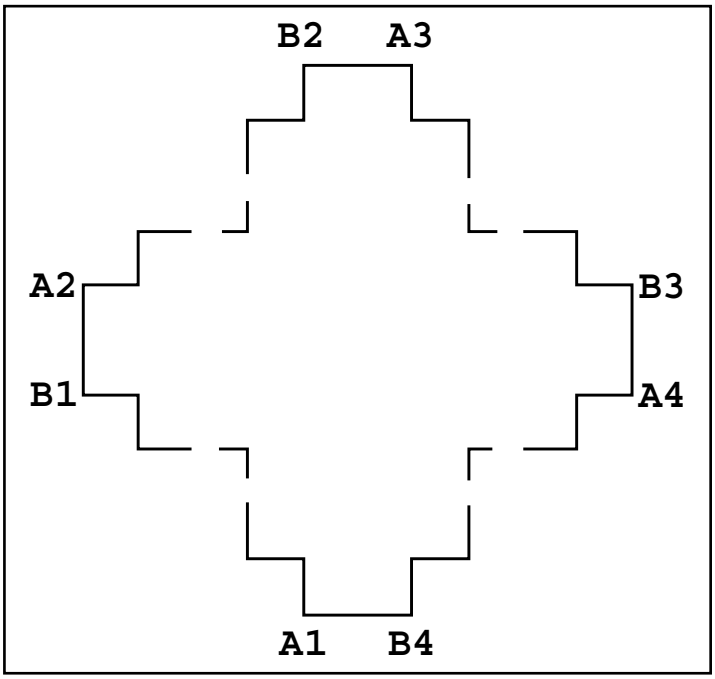

Figure 2. General structure of orthoconvex.

This definition of rank is equivalent to one introduced in [5]. Rank *r* of *orthoconvex P* and number k of its vertices are related by equation

$k=2r + 4$.

Let us denote the interior of set *A* by int(A) and boundary of *A* by fr(A). A set of rectangles

R = $\{R_i\}$, i=1,2,...,n is partition of orthoconvex P if

1. $\text{int}(R_i) \cap \text{int}(R_j) = \phi$, i !=*j*, *i*,*j*=1,2,...,n

2. $\cup \{R_i, i=1,2,...,n\} = P$

There is an evident relation between the structure of *orthoconvex* and its partition. So we will use same letter to denote both the *orthoconvex* itself and its partition. rectangles R_i are called atoms of the partition. The rank of partition is the rank of the partitioned block.

Articles[1,4] propose an approach to find the "best" placement of isotetic blocks based on the sequence of their pairwise merging.

The *orthoconvex hull OH(P)* of an orthogon *P* is defined to be the smallest orthoconvex orthogon containing *P*. The r*ectangular hull RH(P)* is the smallest axis-oriented rectangular that contains *P* (Figure 1).

By a *placement* of a set $S = \{P_1,...,P_m\}$ of orthogons in the plane E^2 we mean a mapping τ which determines an orthogon $\tau(P_i)$ is in E^2 for P_i, $1 \le m$ with the help of translation P_i by a vector v_i.

Remark 1. *We emphasize that our definition of orthogons placement allows overlapping of orthogons* $\tau(P_i)$, i=1,...,*m*.

We define an *orthoblock B* in Euclidean space E^3 to be a point set $B \approx \{(x , y, z) \mid (x, y) \in$ P is in *OXY*, $a \le z \le b\}$, where *P* is an orthogon. The orthogon *P* is called the bottom and $h = b - a$ the height of orthoblock *B*.

Let $M = \{B_1,...,B_m\}$ be a set of orthoblocks whose bottoms $P_1,...,P_m$ have $n_1,...,n_1$ vertices, respectively, $\Sigma_{i=1}^{m} n_i = n$.

By a packing of the set *M* in E^3 we mean a mapping τ which determines a orthoblock τ (B_i) for each B_i, $\approx 1,...,m$ with the help of translation by a vector v_i, with int $\tau(B_i) \cap$ int $\tau(B_j) \approx$ 0,1 $\le i,j \le m$, $i \ne j$, (int *B* denotes the interior of orthoblock *B*). Unless otherwise explicitly stated we assume that rotations of orthoblock are not allowable.

Let $M = \{B_1,...,B_m\}$ be a set of orthoblock in E^3. We define a *container C*(*M*) of orthogons set *M* to be an orthoblock which contains blocks $B_1,...,B_m$, with its bottom being orthoconvex.

The *main problem* is to find a packing of orthoblock set $M = \{B_1,...,B_m\}$ such that the area of containers bottom is minimal. We also refer to the main problem as POO (packing of orthoblocks into an orthoblock).

Notice that container height and, hence, the container volume is not assumed to be minimized in the main problem.

In general case optimal solution is very difficult to find and requires an algorithm based on exhaustive search approach. But in most practical cases an acceptable approximate solution can be obtained much faster and easier. There are number of purely heuristic algorithms designed to solve the packing problem which work very fast. There are also some heuristic modifications of exhaustive search algorithms, which substantially increase algorithm's performance at cost of quality of result.

One of the popular approaches here is the pairwise merging technique. An algorithm based on this technique works in the following manner. Starting with initial blocks it generates new ones by merging them in pairwise manner. Blocks in each pair are attached to each other without overlapping at different relative positions thus creating a number of combined blocks. Such combined blocks are referred to as clusters. If cluster still fits in target area it is called feasible cluster and participates in subsequent mergers to form larger clusters and so on. Note that any feasible cluster as well as any initial block alone represents more or less satisfactory solution of the problem. When the algorithm is not able to generate any new cluster the best solution found so far is reported as final result.

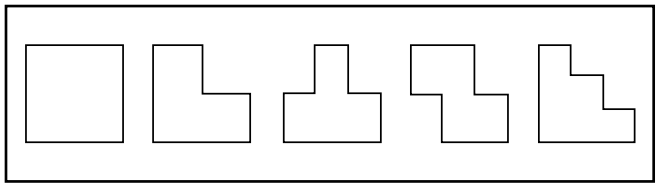

Figure 3. Simplest blocks of bounded rank.

The main obstacle in the way of practical implementation of this algorithm is merging operation . A number of problems complicate the implementation of this operation and the whole algorithm. The most interesting of them is the following.

Any two geometrical objects, even pair of rectangles, can be merged into a new one in an infinite number of ways. While initial blocks are rectangles and are relatively easy to deal with on first iterations of the algorithm, clusters that appear later can have very complex shapes. From the other side, it is easy to see that all initial blocks and intermediate clusters are orthoblocks. There are theoretical results which show that in order to find an optimal solution in this case we can limit ourselves to consideration of a finite number of relative positions of such objects -so called contact concurrent positions. However, the number of such positions can still be very large, which results in generation of large amounts of new feasible clusters. In practice the amount of intermediate data generated by the algorithm can easily overflow memory in any modern computer. Not to mention that algorithm's iteration time grows at a very high rate.

A simple way to reduce the number of variants generated by merge function is the following.When some cluster s is generated at particular iteration of the algorithm, all previously built clusters t, such that cluster s dominates over cluster t, can be discarded without compromising the quality of final solution. In fact, in any placement that contains cluster t it can be naturally replaced with cluster s. Such replacement can only increase the quality of the solution. Note that according to definition of domination relation any cluster s dominates over itself. It means that by discarding dominated clusters the algorithm discards identical clusters as well. The discarding results in substantial decrease in number of clusters built by the algorithm thus reducing the number of variants to be checked and increasing the overall performance of the algorithm.

Amount of internal waste in cluster can be used in two simple heuristic criteria which significantly reduce number of variants processed by the algorithm. The main idea here is to detect and discard intermediate clusters that are unlikely to be part of optimal solution. The criterion is applied to clusters generated by merge operation. If a cluster meets the criterion it participates in subsequent mergers, otherwise the cluster is discarded because of high percentage of internal waste.

These criteria are heuristics, and on particular input data they can "cut off" optimal solution of the problem. But with algorithms described below they provide excellent results in terms of performance and quality of solution. There are also more complex and more precise criteria for discarding clusters that cannot be part of optimal solution . Branch-and-bound technique can be naturally used in pairwise merging algorithms.

3. Special Cases of the Main Problem

Let us specify three cases of POO by means of the following conditions.

1. Both the container and orthoblocks $B_1,...,B_m$ are parallelepipeds (problem PPP).
2. The container is a parallelepiped (problem POP).
3. Orthoblocks $B_1,...,B_m$ are parallelepipeds (problem PPO).

It follows immediately from the setting of POO that its solution can be found if the following problem POO2 has been solved. Problem POO2 is to determine a placement τ^* of the bottoms $P_1,...,P_m$ of orthoblocks $B_1,...,B_m$ such that the area of the orthoconvex hull covering the bottoms placed attains its minimum:

$$Ar\{OH(\bigcup_{i=1}^{m} \tau^*(P_i))\} \approx \min Ar\{OH(\bigcup_{i=1}^{m} \tau(P_i))\},$$

where $Ar\{P\}$ is the area of orthogon P.

Let us consider the following three special cases of POO2 such that their solving provides solutions to PPP, POP, PPO.

1. Both the container bottom and the orthoblocks bottoms $P_1,...,P_m$ are rectangle (problem PPP2).
2. The container bottom is a rectangle (problem POP2).
3. The orthoblock bottoms are rectangle $R_1,...,R_m$ (problem PPO2).

Turn to problem PPP2 which is to find a placement τ^* of rectangles $R_1,...,R_m$ such that the area of the rectangular hull containing the rectangle is minimum and compute $Ar\{RH(\cup_{i=1}^{m} \tau^*(R_i))\}$. The problem is *NP*-hard under additional condition int $\tau(R_i) \cap$ int $\tau(R_j) = 0$, $1 \le i, j \le m$, $i \ne j$ [5] which is a mandatory requirement in two-dimensional cutting stock problems. In our case, where overlappings are allowable, the problem PPP2 can be easily solved int he following way.

Let l_i and h_i denote horizontal and vertical size of rectangle R_i, $1 \le i \le m$. Obviously, the smallest rectangle hull has the sizes $L = max_i\ l_i$ and $H = max_i\ l_i$. Placement τ^* can be realized by placing the corresponding, for example south-western vertices, at the same point (Figure 2.)

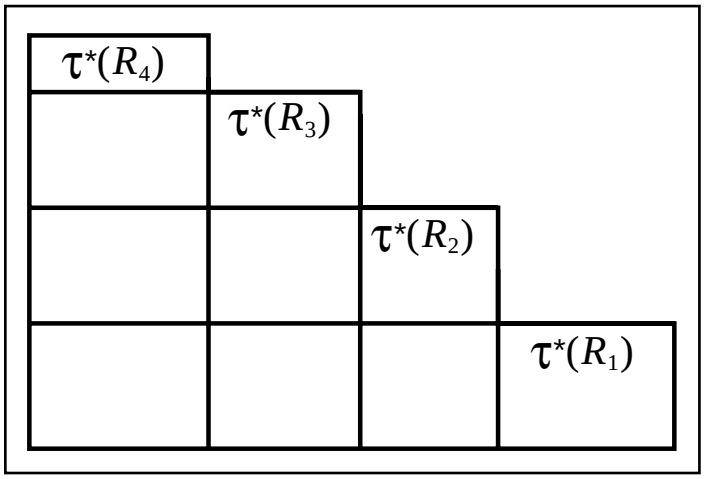

Figure 4. Placement τ* is an exact solution of problems PPP2 and PPO2.

The size of the hull can be computed in $O(m)$ time by comparing all values l_i, h_i, $1 \le i \le m$. That leads us to the following statement.

Theorem 1. *Problem PPP2 and consequently PPP can be solved in O(m) time.*

Theorem 2. *Problem POP2 and consequently POP can be solved in O(m+n) time.*

Proof. Solving problem PPO2 falls into two stages: computing the rectangular fulls for orthogons $P_1,...,P_m$ in $O(n)$ time and then solving problem PPP2 for these hulls in $O(m)$ time.

Turn to problem PPO2. Observe that, for placement τ* shown in Figure 2 the area of union $\cup_{i=1}^{m} R_i$ is minimal. Since the is union is an orthoconvex orthogon, placement τ* is an exact solution of problem PPO2. The vertex list *VList* for this orthogon can be computed n $O(m \log m)$ time by means of the sweep line method [7,8]. The area of the hull can be determined in $O(m)$ time by using the vertex list. Hence, the following statement is true.

Theorem 3. *Problem PP02 and consequently PPO can be solved in) (m* log *m) time.*

Generally, problem POO2 seems to be much more difficult than its special cases considered above. In this situation, one has to engage decomposition approaches. One of them, so-called hierarchical merging method [4], reduces a large-size problem of simultaneous placing orthogons to iterative placements of orthogon paris. At every iteration, a placement of two orthogons to minimize the area of their common hull is determined. This hull is an orthogon which represents the pair of block in later iterations. Selection of orthogons to be merged into a new orthogon (their common hull) is realized with the help of estimating the area waste when merging. These considerations motivates the study undertaken in the next section.

4. An Algorithm of Exact Solving Problems POO for Two Orthoblocks

Our idea is to take an attempt to distinguish from an infinite set of all placements a finite set (we call them *concurrent*) which contains at least one optimal solution.

A placement of orthogons P_i and P_j is, by definition, *concurrent* if *x-coordinates* of some vertical edges of different orthogons concur and the same holds for y-coordinates of some horizontal edges of P_i and P_j.

Note once more that we consider placements that allow overlappings of orthogons placed.

Theorem 4. *There exists a concurrent placement τ* of orthogons P_i and P_j such that*

$$Ar\{OH(\tau^*(P_i) \cup \tau^*(P_j))\} =$$
$$\min_{\tau} Ar\{OH(\tau(P_i) \cup \tau(P_j))\}$$

Theorem 4 serves as the basis of the following algorithmic result.

Theorem 5. Problem POO2 for two orthogons Pi and Pj with ni and nj vertices and consequently POO for two orthoblocks can be solved in $O(n_i^2 n_j^2 (n_i+n_j))$ time and $O(n_i+n_j)$ space.

References

[1] Azarionok A.S., Klebanovich D.M., Krikun V.S., Miatselski M.M., and others. Computer-aided design of VLSI circuits: a method for hierarchical generating a layout pattern on the base of arbitrarily sized blocks, Preprint N 16(326), Institute of Mathematics, AS BSSR, Minsk, 1988 (in Russian).

[2] Dyckhoff H. A topology of cutting and packing problems, Europ.J. OR, v 44, 1990. pp. 145-160.

[3] Martyuchik V.M., Miatselski M.M., Proth J.M. Computing placements of a pair of geometric objects under the criterion of intersection area minimization, Computational Mathematics and Mathematical Physics, N 5, 2000.

[4] Miatselski M.M., Krikun V.S. A method for allocation isothetic blocks, Preprint N 27(427), Institute of Mathematics, Minsk, 1990 (in Russian)

[5] Murata H., Fujiyoshi K., Nakatake S., Kajitani Y. Rectangle-packing-based module placement, Proc of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD'95), 1995.

[6] Nelissen J. New approach to the pallet loading problem, Reports RWTH, Aachen, 1994.

[7] Ottman T., Soisalon-Soininen E., Wood D. Partitioning and separating sets of orthogonal polygons, Information Sciences, v. 42, 1987, pp 31-49.

[8] Preparata F.P.,Shamos M.I. Computational geometry. An introduction, Springer-Verlag, New York, 1985.

[9] Scheithauer G., Terno J. the C4-heuristic for the pallet loading problem, Preprint Math-NM-06-1995, TU Dresden, 1995.

[10] Martynchik V., Miatselski M., Proth J.M. Three-dimensional packings of orthoblocks // Proceedings of the International Workshop "Discrete optimization methods n scheduling and computer-aided design". Minsk, Republic of Belarus, September 5-6, 2000, p.147-150.

Calendar of Events

December	January
1	1
2	2
3	3
4 IEDM - San Francisco, CA	4
5 IEDM - San Francisco, CA	5
6 IEDM - San Francisco, CA	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	27
28	28
29	29 Design Con - Santa Clara, CA
30	30 Design Con - Santa Clara, CA
31	31 Design Con - Santa Clara, CA

Bulletin Board

Silvaco Introduces QUEST

RLCG High Frequency Parasitic Extractor, ***QUEST*** is the first of a new generation of tools developed exclusively for pay-per-use under e*ECAD. (www.eecad.com)

Integration Group Moves to New Building

Silvaco acquired an additional building, Bldg #26, to accommodate the rapid expansion of the Integration group in recent months. The building has been remodeled and developers have now settled in.

Visit Us at EDA Techno Fair in Japan

Come and see the latest Silvaco software developments demonstrated by experienced application engineers from our Silvaco Japan office. Five Silvaco presentations are scheduled during the conference to reveal the latest development and strategic planning. Our partners from ***e*ECAD*** will also be present.

For more information on any of our workshops, please check our web site at ***http://www.silvaco.com***

Hints, Tips and Solutions

Mikalai Karneyenka, Applications and Support Engineer

Q: If I try to close a project, *Expert* gives me a prompt: "Save changes in a project?", but I remember that I didn't do any editing of cells. When can be the reason. The second question is: how do I know which cells were modified?

A: Expert saves changes of the the following items in the project file:

- cell geometry
- project "table of contents" (list of cells and some other data)
- project technology
- layer plans.

In you case, you could change a layer's color or added a new layer plan, and this would trigger the mentioned warning.

To know the list of changed cells, use the command Project>>Save Cells. It will show the list of all changed cells, and you can selectively save some of them.

Q: In trying to debug the procedure I noticed that the command line is fairly limited in how long a command can be. I tried to copy/paste the long search statement into the command line and it cut it off and would not allow me to extend it

A: This limitation for the command-line length is set deliberately: you better see the whole command you are typing. For a more advanced usage, you may switch to the console mode of command-line input by clicking the "Console" button on the command-line bar. You will see the cpnsole window, shown in Figure 1.

This window has both vertical and horizontal scroll bars and allows you to type or "cut'n'paste" a command of any length. Additional benefits are:

- you can see the whole history of your commands, together with responses;
- you can edit any of these commands and re-run them by hitting the <Enter> with the cursor positioned in the corresponding line.

By clicking "Command line" button you may return to the simpler command-line mode. (For example, to make the screen less cluttered with windows.)

Q. We've installed all *CELEBRITY* products and wished to try cross-probing between *Expert* and *Guardian*. But it was hard to find necessary options to switch this functionality on. Can't you give us a clue?

A. To access cross-probing (or inter-tool communication) capability you should perform the following:

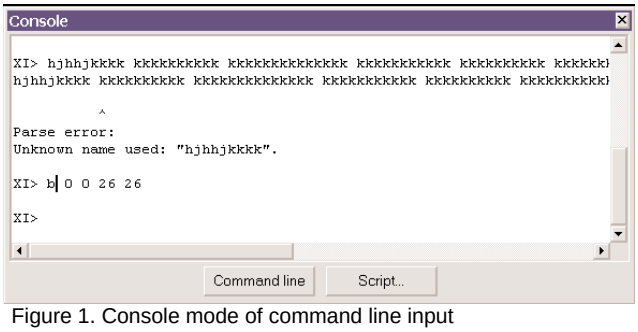

Figure 1. Console mode of command line input

1. Make sure that the ITC tool bar (buttons with the white lamp and gray lamps) is on the screen (if it is not, it using the View>>Show Bars command)
2. Register the layout cell for crossprobing by pressing the Set/Reset Crossprobing button (the button with the white lamp). The button must become green.
3. Load the required project (in ***Expert*** or ***Maverick***) and open the cell to extract. Make extraction using the Verification>>Netlist Extraction>>Run command, but make sure first that options "Annotate Layout" and "Rebuild All Derived Layers" are switched ON.
4. Start ***Guardian*** (either standalone or by ***Expert's*** Verification>>LVS command).

5. Using LVS Setup>>Project Settings, select:
   Schematic netlist as the 1st netlist and
   Layout (extracted) netlist as the 2nd netlist
6. Run LVS.
7. Start Action>>LVS_Navigator. Press RESTART button on the LVS Navigator panel.

If the default settings are used, then the first pair of matched nodes will be highlighted in:

1)Match report (*.MTC file)
2)Both netlists
3)Layout cellview (if they are loaded).

All other pairs matched nodes are browsed using "Next" and "Previous" buttons.

Call for Questions

If you have hints, tips, solutions or questions to contribute, please contact our Applications and Support Department
Phone: (408) 567-1000 Fax: (408) 496-6080
e-mail: *support@silvaco.com*

Hints, Tips and Solutions Archive

Check our our Web Page to see more details of this example plus an archive of previous Hints, Tips, and Solutions
www.silvaco.com

Join the Winning Team!

Silvaco Wants You!

- Process and Device Application Engineers
- SPICE Application Engineers
- CAD Applications Engineers
- Software Developers

fax your resume to:
408-496-6080, or
e-mail to:
jobs@*silvaco.com*

Opportunities worldwide for apps engineers: Santa Clara, Phoenix, Austin, Boston, Tokyo, Guildford, Munich, Grenoble, Seoul, Hsinchu. Opportunities for developers at our California headquarters.

SILVACO
INTERNATIONAL

USA HEADQUARTERS

Silvaco International
4701 Patrick Henry Drive
Building 2
Santa Clara, CA 95054
USA

Phone: 408-567-1000
Fax: 408-496-6080

sales@silvaco.com
www.silvaco.com

CONTACTS:

Silvaco Japan
jpsales@silvaco.com

Silvaco Korea
krsales@silvaco.com

Silvaco Taiwan
twsales@silvaco.com

Silvaco Singapore
sgsales@silvaco.com

Silvaco UK
uksales@silvaco.com

Silvaco France
frsales@silvaco.com

Silvaco Germany
desales@silvaco.com

Products Licensed through Silvaco or e*ECAD

e*ECAD
Vendor Partner