Rectangle-based Technique for Extracting Objects from Polygon Layouts

Yong-Seok Choi, Ik-Jae Chun, and Bo-Gwan Kim
Dept. of Electronics Eng., Chungnam National University
220 Gung-dong, Yusong-gu, Taejon, 305-764, Korea
Tel. +82-42-821-7707, Fax.: +82-42-823-5436
e-mail : sfmania@cnu.ac.kr , s_amity@cnu.ac.kr, bgkim@cnu.ac.kr

Abstract: A new and efficient layout object extraction technique is presented in this paper, which is to be used in technology migration or hard-IP reuse. The technique models the original polygon layout represented by a set of edges as a set of connected or overlapped rectangles. The rectangle model provides easy and efficient recognition, extraction, resizing, and compaction of layout objects such as transistors, contacts, and wires.

Experiments on several designs from simple standard cells to more complex designs using standard and/or custom cells demonstrate the effectiveness of our technique.

1. Introduction

Advances in IC process technology have enabled the design of complex systems on a single chip. ASIC designers need to be able to mix and match pre-designed, best-in-class functional blocks from many providers if they are to meet design deadlines with limited design engineering resources. Meanwhile, there exist a very large number of circuits which should be reproduced and reused for still a long time in the future but with different technologies. The solution is a design methodology shift based on the reuse and implementation of IP's.

One effective method for reusing existing hard IP's (designs represented as mask layouts) is layout-to-layout migration [1~6]. Migrating a layout means adapting it to newer process design rules. This method does not require a major change in the design methodology and is an easier way to implement a reuse methodology. Layout designs are migrated for various reasons: for design reuse in system-on-a-chip applications, for shorter time to market, for the second-source production which is popular among fabless semiconductor companies that try not to be dependent on one silicon vendor, for cost reduction with a smaller die by a process with smaller minimum feature size, and for performance improvement and power reduction. Thus there exists a strong need for fast automatic methods of technology migration for layouts .

First of all for migration of hard IP's, exact symbolization or abstraction of a given design is needed. Retargeting is done through either layout resysthesis[2] or layout compaction[4,5,6] from this abstraction. The latter approach seems to be better in our view since it can better preserve the original designer's intension and verified results embedded in the layout but hard to be documented.

This paper focuses on our approch to abstraction or object extraction.: section 2 on object extraction, section 3 on experimental results, and conclusions in section 4.

2. Extraction Technique

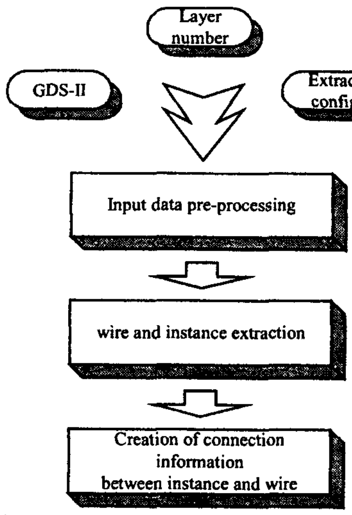
In general, the method for retargeting circuit in reuse of hard IP has been mainly done through linear scaling or by manual work. However, as the minimum feature size is further reduced in DSM (Deep Sub-Micron) technology, it is impossible to generate a feasible layout by the linear scaling. Although the linear scaling preserves the original shape of the layout, compaction ratio is poor since the smallest reduction ratio is used when layers or components shrinks differently in the new technology.

Non-linear layout migration technologies are largely classified into two categories: polygon-based and object-based. Polygon-based approach [5] relies on layout compaction of the original design with a new design rule. The characteristics of this approach are very similar to those of linear scaling: it preserves the original shape, it operates on the flattened layout rather than on the original hierarchical design, separate resizing of circuit element for performance is not easy. Meanwhile, object-based approach [1,2,4] models the original design by a collection of objects and resynthesizes or optimizes the target layout depending on the abstraction levels of the objects. Although the resulting layout shape might be quite different from the original shape, layout can be more optimized through the flexibilities given to extraction and resynthesis/optimizaton step. This approach also permits separate scaling of each device for performance optimization in new technology.

Our system takes the object-based approach because we believe that separate resizing for peformance tuning and preservation of hierarchical structure for efficiency with very large design are important in DSM technology.

Definition of objects and extraction and resizing of objects is the first important step in the object-based approach. The key points of our approach are that wires are extracted as rectangles, not by lines or paths, so that wiring and overall layout shape can resemble the original design, thus preserving much of the original designer's intention, and that hierarchial structure is maintained so that large design is migrated efficiently. Figure 1 showes our object extraction flow.

Input to the system is an original design in GDSII format. Also given is the layers information and design rules for both the source and target technologie. User aslo provides guidelines for object extraction and resizing. The guidelines are defined in the configuration file, which includes the rule for defining the objects, rules for resising, and rules for layer conversion between the source and target technologies.


Layer number
GDS-II
Extract config
Input data pre-processing
wire and instance extraction
Creation of connection information between instance and wire
Figure 1. Layout Object Extraction Flow

The preprocessing step includes merging of connected polygons into a big one in counter-clockwise direction, redefinition of cell boundaries, and relocation of each instances of such cells so that layouts drawn outside the cell bounding box can be can be treated uniformly.

Instances and wires are then extracted from the preprocessed data and resized into the target technology. Connection between instances and wires is checked as the last step, and its information is modified such that the layout model conforms to the model used in layout compaction. This step is described in detail in this paper.

Output is either in customized format used in the following layout compaction/optimization program or in GDSII format for the verification using DRC and LVS programs.

2.1 Object Definition

In our object-based approach the layout design is viewed as a collection of objects, or placement of instances connected by vertical or horizontal wires. This form of layout view is widely used in layout compaction [7].

The objects are classified into two classes accordding to the behavior during resizing and compaction.

Instances are objects with fixed size such as transistors, contacts, and vias. Once extracted and resized its shape and size does not change. They can only move for compaction. Jog is another special instance type, which is defined as a point joining two or more wires in different directions or two wires of different widths in the same direction.

A wire is a directionized rectangle to both ends of which objects are connected. The width is fixed after extraction and resizing, but the length can change for better layout during compaction.

Instances are recognized and extracted first from the polygon layout data given, and then wires are extracted from the remaining portion of the layout data.

2.2 Instance Extraction

Instance types are defined by the user in a configuration file with Boolean expressions of composing layers. The Boolean expressions for a contact and a transistor are as follows for example:

Contact = contact layer & lower layer & upper layer (& user define layers)

Transistor = poly-silicon layer & diffusion layer (& user define layers)

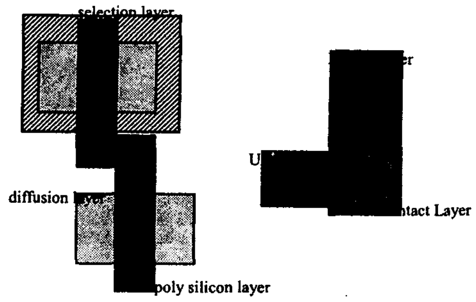


Figure 2. Instance Extraction

Instances are extracted from the preprocessed merged polygon data. Figure 2 shows instance extraction examples.

After instances are extracted, corresponding portion of the layers is deleted for the next step of wire extraction. Specially marked as "internal" are edges of the remainging and possibly separated polygons which are created by instance extraction. These are used to determine the direction of wires during wire extraction.

Multiple contacts are merged into a big one if they are aligned at the same potential. This is to give more chances for compaction. Thus the number of contacts in the resized objects can be regenerated after compaction. For example, appropriate number of well-contacts are generated after the cell width is determined. The multiple contacts can also be replaced by a contact if guided so by the configuration rule.

2.3 Wire Extraction

Recognition and extraction of wires is much more difficult and important for compaction than that of instances.

The algorithm used for wire extraction is similar to the minimization of 2-level Boolean expressions. The polygon is first divided into rectangles, as shown in Figure 3, using virtual grids of the polygon.

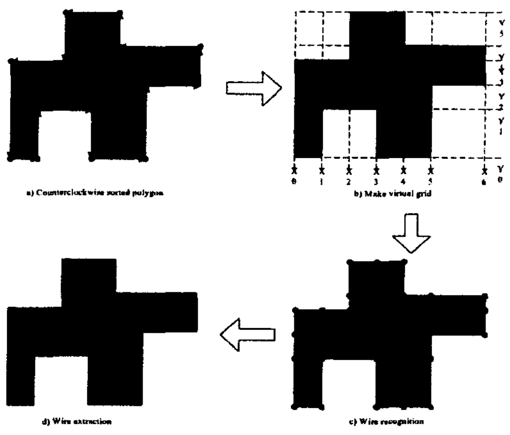


Figure 3. Wire Extraction

Virtual grids are made at the polygon's bending points when traversing the polygon edges. Rectangles obtained by virtual grids are called minimal rectangles, from which maximal rectangles are obtained by merging minimal rectangles. A minimal set of maximal rectangles, which is called a cover, is then obtained.

The direction of each maximal rectangles in the cover is iteratively determined depending on the directions of neighboring rectangles. The direction also depends on the locations of connection points to instances, the location if internal edges, and on the direction of the overlapping neighbor rectangles. Jogs are then defined as the overlapped area of wires.

2.4 Hierarchical Connection Extraction

Preserving the hierarchical information is an another important issue for efficient layout migration of very large designs. Each leaf cell is processed just one time so that all of its instances have the same layout after migration and computation time is reduced.

Electrical connection between hierarchical cells is implicitly specified in layout. Hierarchical connection points are determined by overlapped or abutted layers which are electrically conducting and reside on two different levels. Electrically conducting layer pairs are specified by the user in the configuration file. The hierarchical connection is preserved by replacing the overlapped area by a jog on both levels and by a dummy wire between the two jogs. Thus the two jogs on different levels move together during layout compaction. In some cases a portion of the layout in the lower level is copied onto the upper level to meet design-rules.

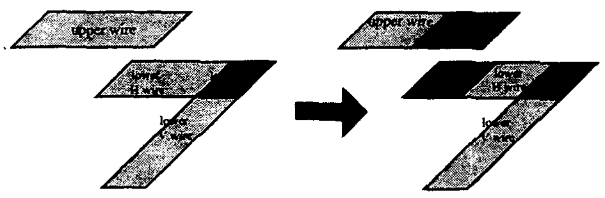


Figure 4. Extracting Hierarchical Connection

2.5 Creation of Connections between Instances and Wires

The connection of instances and wires is checked as a last step. Normally each wire has two objects at the two ends of the wire and each objects has only one wire in its each side. For other special cases as shown in Figure 6 wires and jogs are modified or newly introduced to satisfy the normal condition. If instances are located in the non-end side of wire, the wire is separated, jogs are inserted, and new wires from the jogs to instances are introduced at that points. If there is no instance at an end side of a wire, a jog with zero width is inserted to maintain the original layout shape. If two instances are stacked at the same location, dummy wire in introduced between them.

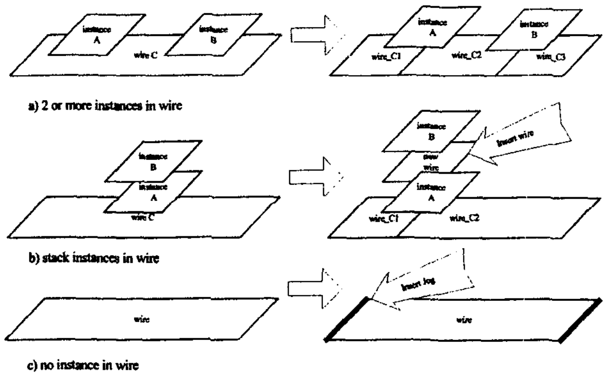


Figure 5. Connecting Instances and Wires

3. Experimental Results

Our technique has been successfully tested on 0.6um and 0.25um standard libraries of 94~223 cells, a 0.6um data-path cell library, and several macro blocks composed of custom cells in addition to standard/datapath cells.

For the correctness of migration, design rules are checked and net-lists are compared with the original designs after extraction/resizing and also after layout compaction.

Figures 6 and 7 show an original flip-flop layout and its extracted layout.. Figures 8 and 9 show an original layout of UART core consisting of about 20,000 equivalent gates in 2-level hierarchical structure and its extracted layout, which demonstrates that hierarchy is preserved.

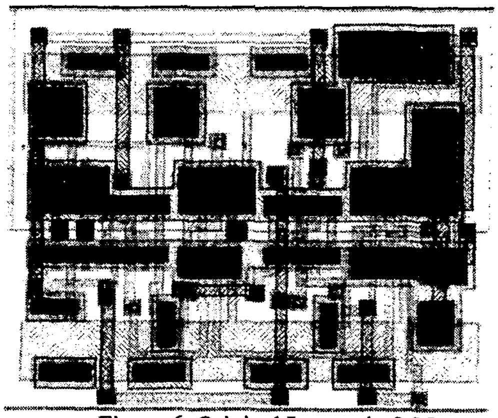


Figure 6. Original Layout in 0.25um

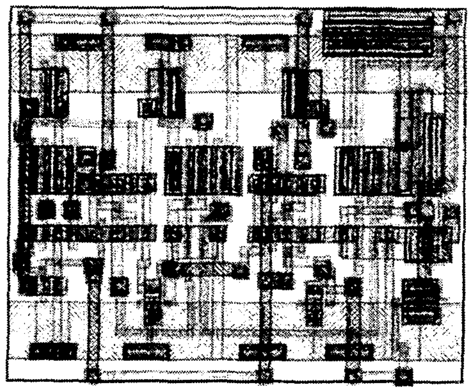


Figure 7. Extracted Objects

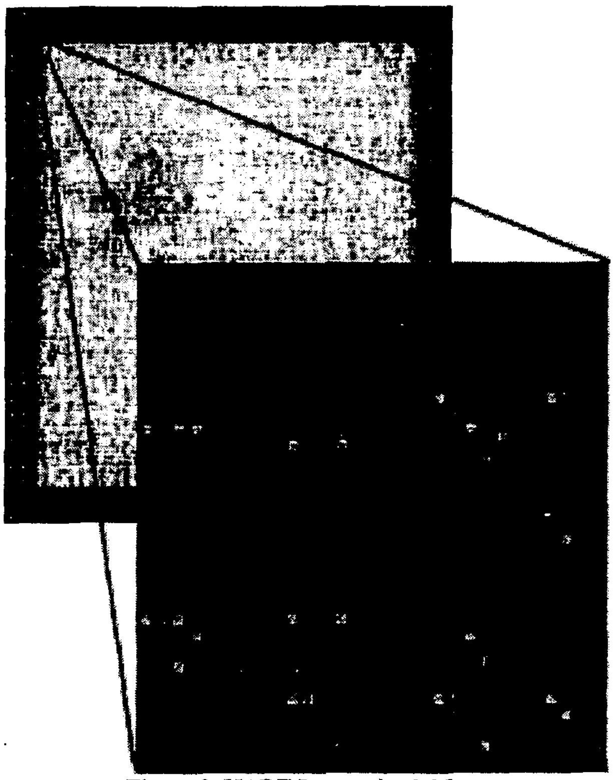


Figure 8. UART Layout in 0.25um

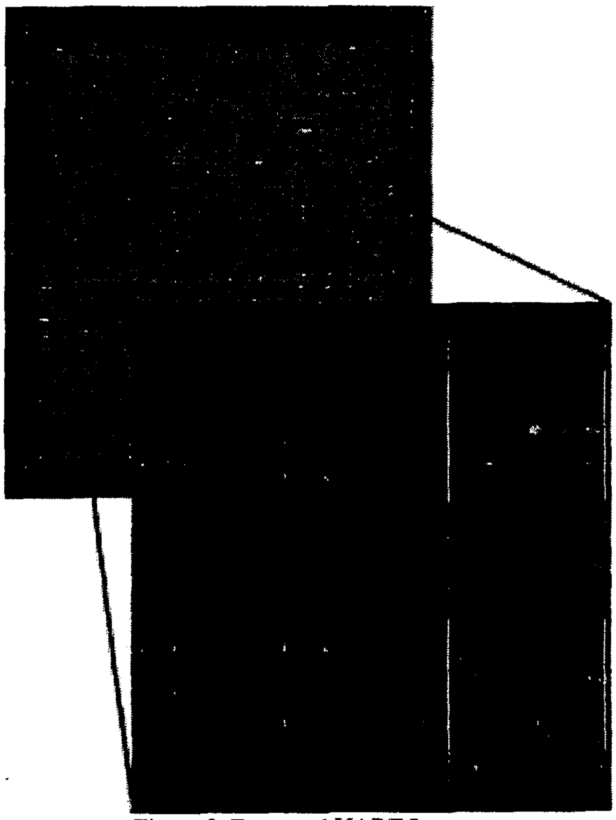


Figure 9. Extracted UART Layout

4. Conclusions and Further Work

We have proposed a new technique for recognizing and extracting layout objects such as transistors, contacts, and wires. The technique, based on rectangle representation of layouts rather than polygon representation, can efficiently handle resizing as well as recognition and extraction of instances and wires. It also provides an easy way to extract hierarchical connection information, which is necessary to reuse very large designs with hierarchical structure.

Most of the migration steps are flexibly controlled by user-supplied rules. Instances are defined by instance-composing rules. Resizing can be defined for each type of instances by a specific ratio or by a fixed size. With a ratio-based resizing of wires, widened wires in the original layout for the signal integrity can be maintained after migration.

We have also noticed that extracted and resized designs are well optimized by a layout compactor that is programmed based on the algorithm in [7].

At present, our technique is not able to recognize objects of non-manhattan shape. Although such layouts are normally not recommended in DSM technology, it is desirable to support them since many previous designs have such shapes or just for the completeness of the technology migration algorithm.

Acknowledgement

This work was supported in part by IDEC at KAIST and BK21 Human Resource Development Consortium for Information Technology of Chungnam National University.

References

[1] Ikjae Chun, *Layout Retargeting for Standard Cell Library*, thesis of master degree, Chungnam National Univ.,February 2000.

[2] John Lakos, "Technology Retargeting for IC Layout," *IEEE Design Automation Conference*, pp. 460 -465, 1997.

[3] Alcantara, "A Novel Circuit Extraction Tool Based on X-Spans and Y-Spans," *IEEE Proceedings of EUROMICRO-22*, 1996.

[4] M. Fukui et. al., "Layout Abstraction and Technology Retargeting for Leaf Cells," *IEICE Trans. on Fundamentals*, Vol. E81A, No. 12, pp. 2492-2500, December 1998.

[5] P. K. Kar and S. K. Roy, "TECHMIG: A Layout Tool for Technology Migration," *IEEE 12th International Conference on VLSI Design*, pp. 615-620, January 1999.

[6] Y. Shigehiro et. al., "Automatic Layout Recycling Based on Layout Description and Linear Programming," *IEEE Trans. on CAD*, Vol. 15, No. 8, August 1996.

[7] H. Shin and A. Sangiovanni-Vincentelli, "Two-dimensional Compaction by 'Zone Refining'," *IEEE Design Automation Conference*, pp. 115-119, June 1986.