A Parametrizable Transistor Folding Algorithm Applied to an Automatic Full-Custom Layout Generation Tool*

Fabrício Biolo Bastian, Cristiano Lazzari, Cristiano Santos, Ricardo A.L. Reis
UFRGS – Universidade Federal do Rio Grande do Sul
{fbastian,clazz,clsantos,reis}@inf.ufrgs.br

Abstract

This paper presents a new folding algorithm applied to an automatic layout generation tool. Most of transistor sizing algorithms propose continuous sizing. Nevertheless, in row-based layout synthesis, the variation of transistor sizes may cause non-uniform cell heights that may lead to significant waste of layout area. The proposed folding approach leads to a very simple algorithm with small complexity that is able to obtain very good results.

1. Introduction

Transistor Sizing is used in physical synthesis to reduce the delay of the critical paths in circuits. In row-based layout synthesis, the variation in transistor sizing may cause non-uniform cell heights that may lead to significant waste of the layout area. To utilize the chip area more efficiently, a transistor folding scheme should be introduced to apply this transistor sizing to the layout.

Gupta and Hayes [2,4] and Krzysztof and Berezowski [3] proposed good solutions for cell generator tools. Although, the folding problem in a row-based layout tool is different. Besides, Kim and Kang [1] developed an algorithm for row-based CMOS layout design which considers the folding before the transistor placement. Since our algorithm is applied to a timing-driven automatic generation tool and we do not desire to insert any extra diffusion gaps to the layout, reorder the transistors is not an advantage.

This paper introduces a new transistor folding algorithm that can find the optimal folded transistors arrangement for a given transistor placement, given desired transistors sizes and given initial PMOS and NMOS transistors sizes.

The proposed algorithm is able to fold transistors in a way that the initial transistor placement is not modificated as the previously methods. All the folded transistors remain together in the row, avoiding metal connections between them. It means that the P-plane and the N-plane of a given cell can be folded independently, neither changing the layout topology nor increasing the design complexity (when folded transistors are connected with metal layers).

2. The Folding Algorithm

A folding algorithm was developed to attend all these constrains:

- The folded transistors must remain together in the row;
- The transistor placement should not be modificated;
- And no extra gap must be inserted.

To satisfy the requirements a dynamic folding with static placement approach was chosen, once the transistor placement should not be changed. So, given the placement, the desired size and the initial size of the transistors, the algorithm is able to determine the number of legs and the orientation for each transistor.

The folding is executed in two steps, first statically and then dynamically. In the static part the number of legs of each transistor is determined simply dividing the size of the resized transistor by the folding size. The static folding may introduce diffusion sharing problems, consequently adding extra gaps to the layout.

After the static folding, the dynamic folding is executed. The main idea of the dynamic part is to classify the transistors according with its parity and fold them respecting it in such a way no new diffusion gaps are inserted. The parities of the transistors are considered as follows:

- An *even transistor* is a transistor that, after folded, should provide a path to return to a node. An *even transistor* can be seen as there was no path between two nodes.
- An *odd transistor* is a transistor that, after folded, should provide a path to go to another node. An *odd transistor* can be seen as there was just an edge between two nodes.

* This work is supported by CNPq, CAPES and FAPERGS Brazilian Agencies

An even transistor will always be resized by an even multiplicity as well as an odd transistor will always be resized by an odd multiplicity.

To classify the transistors, the algorithms walks over the undirectional graph applying the concepts explained above for each transistor, considering the given placement and the transistor chain. The classification of a transistor depends on how it is connected to its neighbor transistors. During the classification, the transistors can receive a temporary classification, which can be changed along the analysis.

As the placement is not changed, the algorithm has been simplified to walk over a list reducing its complexity.

3. Results

Table 1 and *Table 2* show the results of the proposed algorithm over two different circuits. Cases when the delay reductions does not correspond to the delay requirement show limitations of the gate sizing tool [5] used on the layout generation tool called *Parrot Punch*[6].

As the algorithm allows that PMOS Plane and NMOS Plane are sized independently, the number of PMOS and NMOS folded transistor may be different. Because of this difference, the layout tool must use doglegs to make the poly routing.

The results presents that good delay optimizations were done with small area penalty, which proves the algorithm's efficacy.

Table 1. Results of the proposed algorithm applied to circuit bw (628 transistors) with the required delay optimizations

Delay Requirement	**No Optimized**	**-10%**	**-20%**	**-30%**
Area	16068um^2	4%	6%	10%
Delay Reduction		-10%	-20%	-25%
Sized Gates		4	8	23
NMOS folded transistors		4	12	42
PMOS folded transistors		2	13	30

Table 2. Results of the proposed algorithm applied to circuit csa8 (288 transistors) with the required delay optimizations

Delay Requirement	**No Optimized**	**10%**	**30%**
Area	6500um^2	1%	3%
Delay Reduction		-7%	-20%
Sized Gates		4	9
NMOS folded transistors		12	25
PMOS folded transistors		2	37

4. Conclusion

A new folding algorithm was presented as well as its results when applied to the automatic layout generation tool.

The algorithm uses a new approach for folding, based on the paritity of the transistors, avoiding diffusion gaps and this way reducing the transistors perimeters, which make the cells faster.

Results show that good delay optimizations can be done with small area penalty.

References

1. Kim, J., Kang, S.M.: An Efficient Transistor Folding Algorithm for Row-Based CMOS Layout Design. Design Automation Conference (1997) 456-459

2. Gupta, A., Hayes, J.P.: Optimal 2-D Cell Layout with Integrated Transistor Folding. ICCAD (1998) 128-135

3. Berezowski, K.S.: Transistor Chaining with Integrated Dynamic Folding for 1-D Leaf Cell Synthesis. EUROMICRO Digital Systems Design Symposium (2001)

4. Gupta, A., The, S-C., Hayes, J.P.: XPRESS: A Cell Layout Generator with Integrated Transistor Folding. European Desing & Test Conf. (1996) 393-400

5. Santos, C. L., Wilke, G., Lazzari, C., Güntzel, J., Reis, R.: A Transistor Sizing Method Applied to an Automatic Layout Generation Tool. Sbcci. São Paulo (2003) 303 - 307

6. Lazzari, C. , Domingues, C. V., Güntzel, J., Reis, R.: A New Macro-cell Generation Strategy for Three Metal Layer CMOS Technologies. Vlsi Soc. Darmstad-Alemanha (2003) 193-197