PLANAR TRANSISTOR NETWORK VISUALIZATION ALGORITHM

Rafael H. da Silva, Vinicius Callegaro, André I. Reis, Renato P. Ribas

Instituto de Informática, UFRGS
Av. Bento Gonçalves, 9500 – CEP 91501-970, Porto Alegre, Brazil
{rhsilva, andreis, rpribas}@inf.ufrgs.br

ABSTRACT

The visualization of switch networks is a very interesting tool for analysis and verification of logic cells generated automatically. In this context, the use of graph theory is very useful to attain this objective. The proposed algorithm satisfies this necessity to represent complementary series/parallel and partially "bridge" network logic styles, avoiding any wire crossing. A tool prototype is available.

1. INTRODUCTION

CMOS design is currently the most used and well established logic style applied by the modern industry of microelectronics. Meantime, the handcraft design as well as the execution of all different design flow steps and tasks is prone of mistakes. In this sense, it is important to build CAD tools that provide good assistance in the creation of projects of integrated circuits.

An import help is offered by the visualization of transistor networks that represent CMOS logic gates. This problem consists in transforming switch networks in a pleasant visual description that contains all network components appropriately connected by wires (lines).

This work presents a methodology to automatically generate the visual representation of logic networks from a textual description. In order to achieve this goal, some concepts from graph theory has been applied. In literature, we have found few publications that use graph theory and graph drawing. Thus, this knowledge needs to be adapted for application in the proposed tool.

Next session presents a brief definition about CMOS logic gates. Then, the proposed algorithm to obtain the components position is described. Finally, the results and conclusions about this work are outlined.

2. CMOS LOGIC GATES

CMOS logic gates can be divided in two types of logic styles known as complementary series-parallel topologies (denominated here as SP) and "bridge" or non-series-parallel networks (referred here as NSP).

As described in [1], SP logic style is an arrangement of series and parallel transistors in two separated logic planes: PMOS pull-up and NMOS pull-down ones. An example of this style occurs in the NOR transistor network visualization, as illustrated in Fig. 1.

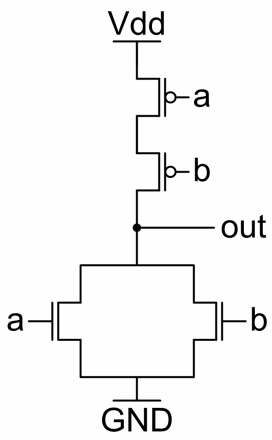
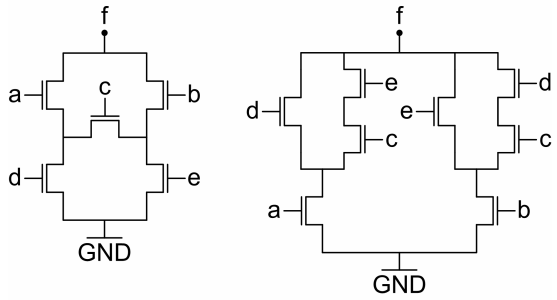


Figure 1 – CMOS pull-down and pull-up plane of NOR gate.

The NSP logic style, in turn, allows sometimes to build logic functions using less transistors than SP approach [2]. Fig. 2 shows a case that the SP pull-down plane requires 8 transistors to represent the function, while the same logic behavior in NSP needs only 5 transistors. However, the NSP cannot be obtained through logic equations.



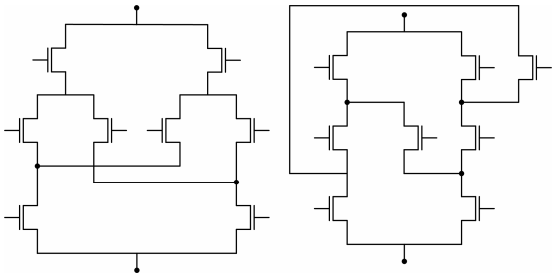
(a) (b)

Figure 2 – Function f=!(a*d+a*c*e+b*e+b*c*d): (a), bridge-based arrangement (b) SP logic style.

Moreover, other logic styles as pass-transistor logic (called as PTL) have been exploited to guarantee some electrical advantages. PTL proposes to combine PMOS and NMOS devices in the same plane as described in [3].

Switch networks can be also qualified as planar network or non-planar one. Planar networks attend to the essential characteristics: There is a way to represent the

network in a plane avoiding transistors crossing and with all terminals (Vdd, Gnd an output node) positioned in contact with the external face. It is observed in SP networks. A non-planar network is showed in Fig. 3. This situation usually occurs in NSP arrangements.



(a) (b)

Figure 3 – No planarity conditions, (a) transistors crossing, (b) terminal node is surrounded.

3. PROPOSED ALGORITHM

The input of the method is a textual description of the transistor network. This description is translated into a multi-graph representation in order to be manipulated. The algorithm generates the visual representation by compressing the switch arrangement, through series and parallel device associations, in order to identify the network type SP or NSP. In the case of a SP network, the algorithm is applied directly by decompression (expanding) the structure. Otherwise, the NSP network is transformed or shared in SP networks. The flowchart in Fig. 4 represents these steps.

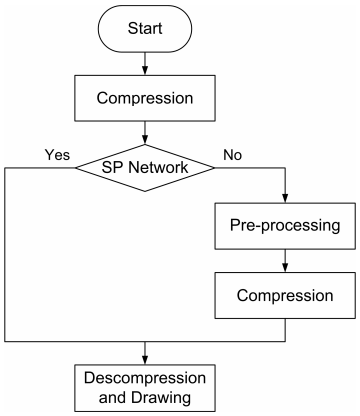


Figure 4 –Algorithm flowchart.

3.1 Data Structure

The procedure maps a network to a multi-graph representation. A multi-graph G consists in a triple of nodes (vertexes) set V(G), an edges set and a relation for each edge has two vertex, that in this work need to be distinct. A detailed explanation about graph theory and multi-graphs can be found in [4] and in [1], respectively.

Nodes labeled in network are mapped as V(G). The transistors (or switches) are equivalent to the edges. Also, a relation that links each transistor to two distinct vertexes exists. Moreover, the structure is extended to have the vertex positions in the drawing and the edges to have their dimensions. The only warn about this mapping is which the terminal nodes are kept as specific references, definition not covered in graph theory. Initially, the graph drawing techniques were studied to construct a good planar network representation, avoiding transistors crossing. But, the graph theory and algorithms do not work with the definition of such special nodes or terminals. Without these restrictions, terminals can be surrounded by transistors. This situation must be avoided.

3.2 Compression

In the first step, a compression series-parallel is accomplished to obtain the dimensions of each edge generate by compression. This data are necessary to discover the position of each component, as illustrated in decompression step.

In the series-parallel compression discussed in [1], all multiple edges joining the same two vertexes (parallel edges) are merged into single edge. In the same way, the edges connecting vertexes with 2-degree (series edges) are merged into single edge. Notice that this merge causes the suppression of the 2-degree vertex. This procedure is only allowed if the vertex is not a special vertex. The relationship between compressed edges and the original ones are saved to be used in the decompression step. Recursively, it is done until no edge compressions are detected.

3.3 Setting edges dimension

Once finished the compression, it can obtain the edges dimensions. For definition in this work, “height” is the number of transistors in series and “length” is the number of transistors in parallel that belong to an edge. All original transistors have their dimensions equal to one. The evaluation begins by the first edges generated through the first parallel compression. In this case, it must sum the length of all compressed edges to obtain its length. To achieve its height, the algorithm has to find the highest in its compressed edges. Next, the edges generated by the first series compression are examined. The length is extracted through longest in its compressed edges. To obtain the height, it must sum the heights of all its compressed edges. It is done until all edges have its dimensions known.

3.4 Detecting the network

After the compression, the network type needs to be identified because the next procedure only works using SP network. To draw the NSP networks, a pre-processing is necessary. This task can be made by counting the number of edges. In the case that the graph presents only

compressed edges then the network is a SP type, otherwise the network is a NSP one.

3.5 Finding the SP networks components position

Initially, the first node treated must be a terminal node. It is necessary because graph theory treats all nodes as a set without any order relation. Meanwhile, this characteristic allows the reuse of this procedure to obtain the SP arrangements. Moreover, it permits to draw only a plane with the same procedure. The visual generation consists in the following sub steps, always saving the transistor vertexes position.

The network decompression is done to obtain the transistors positions. In series decompression case, in order to determine which edges recovered must be drawn first, it is necessary to localize decompressed edges that have relation with the actual node. Next, it determines the new actual node equal to another decompress edge vertex and use the edge height to obtain the next position. Then, the same process is done to other edges.

In series decompression case, the edge length is used to define its position. For each decompressed edge to take advantage of its length to obtain the next position of the next parallel transistor.

Other components as terminal nodes and lines can be easily drawn using the vertexes position.

Fig. 5 and Fig. 6 illustrate the process to obtain the positions, which is done until all transistors have their positions.

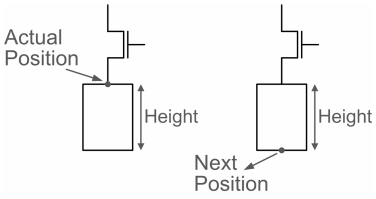


Figure 5 – Process to obtain the series decompression position.

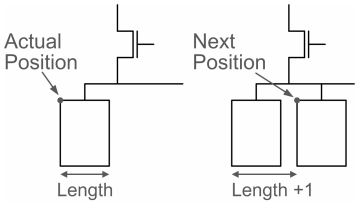


Figure 6 – Process to obtain the parallel decompression position.

3.6 Pre-processing procedure to NSP networks

The main objective is to transform the NSP network in portions composed by SP networks. Fig. 2a shows one example of NSP network that will be divided in SP networks.

To do it, this procedure needs to identify all edges to interconnect the terminal nodes, known as paths in graph theory, and the common edges to all ways (named here as common edges).

It identifies two types of way. A direct way is defined as the path that has the minimal edges set generate a path not using other edges of other direct paths, except for the common edges. The bridge way was those not classified and has least one edge not used by another direct way or bridge way.

After, the procedure finds the relation between the types of ways. Firstly, choose a bridge way and encounter the most common sub way with a direct way (the largest compartment of continuous edges). Then, the second most common sub way to another direct way that doesn't have the edges of last comparison needs to be encountered. In the bridge way, edges that were not contained by any direct way will be considered as bridges.

Finally, each bridge is transformed as SP network having terminal nodes (only two) defined as the vertex contained with the direct way. The common edges are reorganized as SP networks having as terminal nodes its only two vertexes. Moreover, it divides the direct paths in sub paths. The direct sub path is also transformed in SP arrangement, considering its terminal nodes the vertexes to the common edges or that are network terminal nodes. The last step, it is compressed one more time all networks to eliminate eventual series or parallel edges generated to the network division. The result of the application of this procedure in Fig. 2a network is showed in Fig. 7.

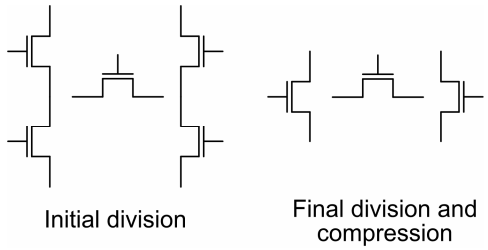


Figure 7 – Pre-processing task.

3.7 Using the components position procedure for NSP networks

This application targets SP networks generated by common edges or direct sub paths. Only the same concepts introduced, by the series-parallel decompression, is necessary in order to set each position of the NSP network.

However, bridge arrangements need to wait until their terminal nodes have information about their position to be drawn. Moreover, bridging devices are drawn perpendicularly, generating a swap in their length and height.

4. RESULTS

The viewer prototype developed is able to represent SP networks without transistors crossing, providing clear illustration of series or parallel links, as showed in Fig. 8. Moreover, it is also possible to represent planar NSP

networks that has the same degree (number of incidents edges in a vertex) in Vdd/Gnd and output nodes, and do not present a NSP network generated by a bridge, as demonstrated in Fig. 9.

Java language has been utilized to develop this tool. Because, this language allows the object orientation method, important to offer an easy maintenance in the tool, and provides the portability to different operating systems.

Unfortunately, this algorithm cannot be compared with others. Because, in all publications read, they do not deal with drawing switch avoided any wire crossing. This difficult was one of motivations to do this publication.

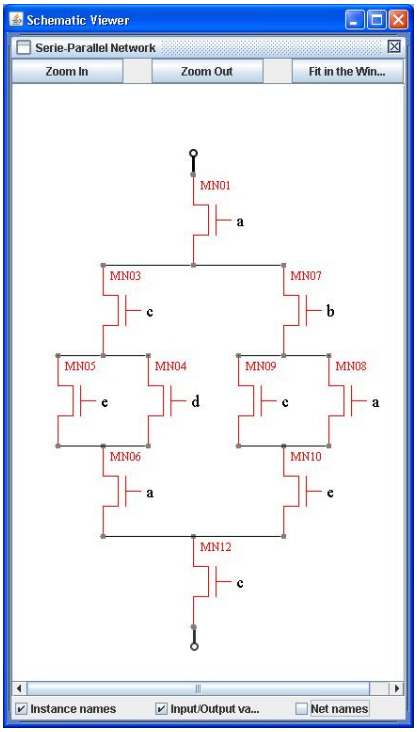


Figure 8 – SP network, NMOS pull down plane.

5. CONCLUSION

The positioning algorithm that draws SP networks and NSP ones has been presented. This paper demonstrates a solution to avoid switch crossing in SP networks and a partial solution to avoid the same situation in planar NSP network. A schematic viewer prototype is ready to use.

6. ACKNOWLEDGEMENTS

This work has been developed in cooperation with Nangate Inc., including financial support.

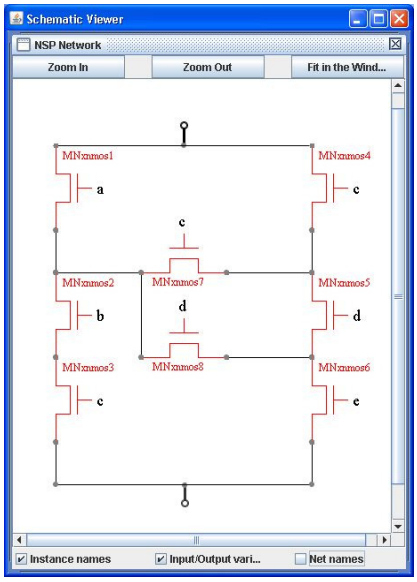


Figure 9 – NSP network, NMOS pull down plane.

7. REFERENCES

[1] V. Callegaro, L. S. da Rosa Jr, A. I. Reis, R. P. Ribas, "A Graph-Based Solution For Dual Transistor Network Generation", *Student Forum on Microeletronics 2008 (*SForum 2008), September 2008.

[2] D. Kagaris, T Haniotakis. A Methodology for Transistor-Efficient Supergate Design. *IEEE Transactions on VLSI Systems*, vol.15, n.4, pp. 488 – 492, April 2007.

[3] L.S Jr. da Rosa; F.S Marques, F.Schneider, R.P. Ribas,; A.I. Reis, A "Comparative Study of CMOS Gates with Minimum Transistor Stacks". *Symposium on Integrated Circuits and Systems Design 2007* (SBCCI'07), pp. 93 – 98.

[4] West, D. B. *Introduction to graph theory*, Prentice Hall, Upper Saddle River, 2001.

[5] Weste, N.H.E. *CMOS VLSI Design: A circuits and Systems Perspective*, Pearson Education Inc, 2005.