NEW APPLICATIONS OF THE AUTOMATIC LAYOUT GENERATION TOOL CDF-2

Felipe R. Schneider, João D. Togni, Renato E. B. Poli, Júlio C. Silvello, Renato P. Ribas, André I. Reis

Instituto de Informática – UFRGS
Porto Alegre, RS – Brazil
{felipers, togni, rpoli, silvello, rpribas, andreis}@inf.ufrgs.br

ABSTRACT

This paper describes a newer version of so-called CDF, a CAD tool for automatic standard-cell layout generation. The additional features of the CDF-2 tool include a novel cell level logic synthesis method, the possibility to generate non-complementary PMOS and NMOS networks, cell generation targeting different logic families (static and dynamic, single- and dual-rail topologies), faster transistor placement algorithms, and an electrical diagram visualization from the network description. The improvements of the tool allow new fields of application such as synthesis of functional blocks for asynchronous design as well as the accomplishment of straightforward benchmark flow for logic synthesis. It is also useful as a cell server for library-less design flows.

1. INTRODUCTION

The cell-based methodology is widely used in IC design. In this approach, each time the IC process or design technology changes an entire set of cells, which compose a cell library, must have its mask layouts regenerated. Hence, automating the whole process of synthesizing cell libraries will reduce the time-to-market of state-of-the-art IC projects. Thus, CAD tools that rapidly accomplish the task of developing leaf cells for libraries are as strategic as the fabrication technology is.

The cell generation flow can be divided in individual steps, so there are commercial and academic CAD tools developed to automate this flow. However, nowadays none of these softwares is compromised on accomplishing all the steps in a single tool. The available tools usually address their efforts either to logic synthesis (optimal transistors network generation) or to physical synthesis (optimal layout generation).

This way, this newer version of CDF [1] still keeps the characteristic of keeping together in the same environment procedures of logic and physical synthesis leaf cell level. However, new features were added to the tool which made possible extending the usability of the tool to other application fields. The role of CDF-2 in IC's design flow is presented in Fig. 1.

2. NEW FEATURES

Transistor-level logic synthesis process consists, basically, on translating a high-level description into an optimized transistors network (netlist), taking into account some project constraints. Automating this process speeds-up the time of converting a Boolean function into a netlist.

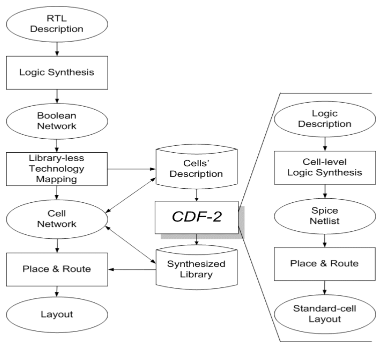


Fig. 1 – The role of CDF-2 in IC's design flow.

CDF-2 allows the high-level description of logic cells throughout either truth-table or Boolean expression. Furthermore, a set of different logic synthesis methods and logic families are available. This availability of different logic synthesis methods are due to the fact that each method presents better results to different sets of Boolean functions and applications [2]. The methods CDF-2 is working nowadays are Quine-McCluskey [3], Muroga. [4], Pass Transistor Logic [5] and CMOS from BDD [6]. The latter was developed in our research group.

As the tool starts from a logic description, it is able to target different logic families, because the P and N planes are generated directly from the logic equations. CDF-2 is able to derive these transistor configurations directly to the equation and add the extra transistors that define the logic family.

The tool is also able to deal with non-complementary P and N transistor networks. These topologies will happen when the original truth table contains high impedance outputs. The high impedance is considered logic 0's when synthesizing the PMOS plane and logic 1's when synthesizing the NMOS plane. This way, when a combination that results in high impedance occurs, none of the planes will conduct.

The main objective of transistor placement algorithms, for one-dimension layout style, is to find an optimal chaining of transistors. So, to solve the problem of finding an optimal chaining, different transistor-level placement algorithms were implemented [7].

The output of CDF-2 logic synthesis process is an optimized transistors network formatted as a Spice netlist. This resulting network agrees with all previously provided parameters which will define the topology and performance characteristics of the desired cell. So, to agree with the academic purposes of the proposed tool, CDF-2 incorporates a graphical viewer of the electrical diagram which corresponds to the netlist generated on the logic synthesis process.

3. APPLICATION FIELDS

With the features described in previous section, CDF-2 by this time can be employed in a wider set of applications. Among these are straightforward benchmark flow for logic synthesis, use as cell server for technology mapping and synthesis of functional blocks for asynchronous design.

Benchmarking is important for comparison of different methods for circuit synthesis. For instance, many papers about logic synthesis use literal count as a figure of merit. However, this position has limited validity under design restrictions.

The computer architecture community used Millions of Instructions Per Second (MIPS) as a figure of merit to compare processor performance for a long time. Today it is well accepted that the performance (time to run a program) of a processor is given by:

$$Time = \frac{Instructions}{program} \ x \ \frac{clock\ cycles}{instruction} \ x \ \frac{seconds}{clock\ cycle}$$

The same way, the number of literals is not a valid figure of merit. The area of a given circuit is given by:

$$Area = (\#cells)\ x\ (average\ cell\ area)\ x\ (routing\ overhead)$$

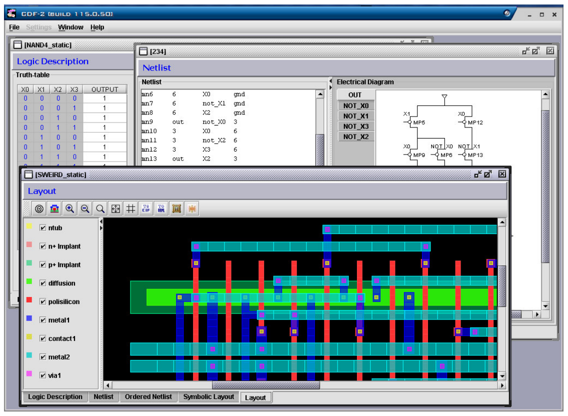
This way, CDF-2 is a valuable tool to be used in a benchmark flow, because it is able to give comparative area for cells targeting different logic families. Besides, as the cells generated by the tool are compatible with the standard-cell design style, the routing overhead may be obtained by using standard place and route tools.

Another application of the tool is as a cell server for library-less circuit design [8]. This way, it opens the way to perform research on technology mapping targeting virtual libraries. This research may be done using mixed logic families and synthesis methods.

Finally, asynchronous circuit synthesis is based on cells that contain asynchronous memorization features, and also can be done by our tool. This means that asynchronous cells truth-tables, like a Muller cell truth-table, will be composed of input combinations for which the outputs may be forced to zero or one or memorized. This way, the tool may be used to generate asynchronous cells automatically.

4. CONCLUSIONS AND FUTURE WORKS

In this paper, we have described a tool for automatic standard-cell layout generation. The main characteristic that distinguishes this tool from the available ones is the possibility to generate a mask layout using an ordinary logic description as starting point (Fig. 2). Furthermore, the tool implements new features which increases the possibility to extend its usability for purposes like benchmarking for logic synthesis, design of functional blocks for asynchronous circuits and use as cell server for library-less design flows.



X0	X1	X2	X3	OUTPUT
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1

```
mn6     6      X0      gnd
mn7     6      not_X1  gnd
mn8     6      X2      gnd
mn9     out    not_X0  3
mn10    3      X0      6
mn11    3      not_X2  6
mn12    3      X3      6
mn13    out    X2      3
```

Fig.2 – The CDF-2 automatic layout generation tool.

CDF-2 is also scalable, once that it was developed to be used as a platform to integrate different research works of the group in a single tool. Hence, as the tool has been designed namely with academic purposes, it has a GNU General Public License, which means that it can be freely downloaded at the group homepage [9].

5. ACKNOWLEDGEMENTS

The authors would like to thank the support provided by CNPq, FAPERGS and PROPESQ-UFRGS.

6. REFERENCES

[1] J.D. Togni, F.R. Schneider, V.P. Correia, R.P. Ribas and A.I. Reis. "Automatic Generation of Digital Cell Libraries", *SBCCI*, Porto Alegre, 2002

[2] F.R. Schneider, V.P. Correia, A. I. Reis, and R.P. Ribas, "Comparing Transistor-level Implementations of 4-inputs Logic Functions", *ACM/IEEE 11th IWLS*, June 2002, pp. 361-365.

[3] E.J. McCluskey, "Minimization of Boolean Functions", *Bell Systems Technical Journal*, November 1956, pp. 1417-1444.

[4] S. Muroga, *Procedure 4.2.1: Design of Logic Networks with Negative Gates in Single-Rail Input Logic*, VLSI System Design: When and How to Design Very-Large-Scale Integrated Circuits, John Wiley & Sons, 1979.

[5] P. Buch, A. Narayan, A.R. Newton, A. Sangiovanni-Vincentelli, "On Synthesizing Pass Transistor Networks", *ACM/IEEE IWLS*, 1997, pp. 101-108.

[6] R.E.B. Poli, F.R. Schneider and A.I. Reis. "Unified Theory to Build Cell-Level Transistors from BDDs", *SBCCI*, São Paulo, September 2003.

[7] F.M. Trindade, F.R. Schneider and A.I. Reis. "Euler Path Search for Cell Width Minimization". *Student Forum 2003*, São Paulo, September 2003. **Under Submission.**

[8] S. Gavrilov et al. "Library-Less Synthesis for Static Combinational Logic Circuits", *ICCAD*, 1997.

[9] LAGARTO – Layout Automatic Generator Tool Project. Available at: <http://www.inf.ufrgs.br/gme/lagarto>