Library Free Technology Mapping

A.I. Reis[1,2]
R. Reis[1]
D. Auvergne[2]
M. Robert[2]

Abstract

This paper presents an efficient method for mapping a set of Boolean equations onto a set of Static CMOS Complex Gates (SCCGs) under a constraint in the number of serial transistors. This Library Free Technology Mapping (LFTM) approach uses a virtual library of SCCGs available through a layout generator, instead of using a limited set of pre-characterized cells. Our goal is to use a virtual library of SCCGs to perform the mapping at the transistor level, in order to fit the topological constraints imposed by the CMOS technology. Limitations of previously proposed techniques to perform Library Free Technology Mapping are discussed. The proposed method, based on an one-to-one association of CMOS transistors with Binary Decision Diagram arcs, is not dependent on the initial ordering of Boolean equations. Experimental results comparing this technique to previously published ones indicate that it generates good-quality solutions.

Keywords: Technology Mapping, Complex gates, BDDs, automatic synthesis, cell libraries

1 INTRODUCTION

The automatic synthesis of integrated circuits involves the generation, treatment and optimization of several intermediate descriptions at different levels of abstraction. At the back end of this process, it is necessary to choose the physical elements that will be used to implement the final layout. This task is usually known as technology mapping and it was pioneered in the works of Keutzer (1987) [5] and Detjens (1987) [4]. Formally, technology mapping is the task of mapping a set of Boolean equations onto a set of given physical elements to minimize a total given cost function. In a typical application, the set of Boolean equations does not have any technology information and comes from a logic synthesis tool. The set of physical elements is normally a library of pre-characterized cells for a given target technology. Each cell is associated with some technology specific information such as its area, delay, power, load and drive capabilities. The technology

[1] {andreis,reis}@inf.ufrgs.br, Institute for Informatics/UFRGS, Brasil
[2] LIRMM , UMR 5506 Université MontpellierII.

mapping tool chooses a set of cells from this library in order to obtain a circuit equivalent to the original Boolean description. The goal of this process is globally minimize some of the technology dependent cost functions (area, delay or power). Obviously, the quality of the final set of physical elements depends on the quality of the initial set of Boolean equations and on the quality of the library, as well as on the ability of the mapping tool to use the cells available in the library.

This paper presents an efficient method of mapping a set of Boolean equations onto a set of Static CMOS Complex Gates (SCCGs) under a given constraint in the number of serial transistors. This approach is implemented in a tool called TABA and it is conceived to be in connection with a leaf cell generator. Our goal is to use a virtual library of SCCGs (available through the leaf cell generator) and to perform the mapping at the transistor level, in order to fit the topological constraints imposed by the CMOS technology.

This paper is organized as follows: Section 2 details the formulation of the Library Free Technology Mapping (LFTM) problem and discusses the limitations of previously proposed approaches. Section 3 presents our approach to Library Free Technology Mapping . The validation of the proposed method is given in section 4, where the performances obtained by TABA on a set of benchmarks are compared to previously published results. Finally, we conclude on the benefit of using LFTM and on its application to circuit design.

2 FORMULATION OF THE PROBLEM

LFTM relies on the extensive and efficient use of a large virtual library of SCCGs instead of using a pre-characterized library. Let us discuss some topics and present terms related to this problem.

2.1 Static CMOS complex gates

A Static CMOS Complex Gate (SCCG) is composed of two complementary serial/parallel networks of transistor switches connecting the output of the SCCG with the logic signal sources. The network between the output and the logic 1 source (Pull-up network, or P plan) is composed of PMOS transistors while the network connecting the output to the 0 logic source (Pull-down network, or N plan) is composed of NMOS transistors. Each input of the SCCG controls a dual pair composed of a NMOS and a PMOS transistor.

2.2 CMOS Virtual libraries

We will denote by SCCG(n, p) the set of the SCCGs with a maximum of $n(p)$ serial NMOS (PMOS) transistors, respectively. Note that a given limitation in the number of serial transistors induces a finite set of SCCG configurations to be available as a virtual library.

From Detjens (1987), the cardinality of the set SCCG(4,3) is |SCCG(4,3)|=396. The library SCCG(2,2) is composed of seven gates : a one-input gate (inverter), two two-input gates (2-input-nand and 2-input-nor gates), two three-input gates and two four-input gates, as shown in figure 1.

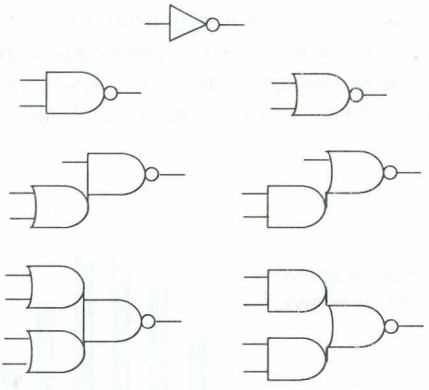


Figure 1 - SCCG(2,2) virtual library.

2.3 Previous approaches to Technology Mapping

Technology mapping needs the application of two distinct operations : matching and covering. Matching is the recognition of the equivalence between a portion of the initial network and library cells. Covering is the task of choosing the best set of cells whose interconnection represents the original circuit among the several possibilities returned by the matching phase. Now we will discuss the limitations of applying current techniques to perform LFTM.

2.3.1 Pattern Matching

Pattern matching techniques were used in the pioneering works of Keutzer and Detjens [4,5]. The difficulty to apply pattern matching techniques to LFTM is that the number of patterns is greater than the number of SCCGs. DAGON applies pattern matching on a DAG reduced to a forest of trees. The number of necessary tree patterns is not linear with the library size, and 1171 different patterns are needed to represent a subset of SCCG(4,4) containing only 123 SCCGs Detjens (1987). Pattern matching is limited by the size of the library.

2.3.2 Boolean Matching

Pattern matching is unable to treat a great number of (logically) different cells. Recently Maillot [6] proposed a Boolean matching technique in which the logic function of a sub-set of the subject description is compared with the logic function of a library element, by using ROBDDs canonicity [3]. Even if there is only one logic function for the element, the number of pin assignments to match a cell is exponential with its number of inputs. In [3] the libraries were limited to gates with 11 or less inputs and in practice, only gates with a maximum of 6 inputs were used. Figure 2 shows the limitations of using only six-input cells with respect to the library SCCG(4,4). Boolean matching is limited by the size of the cells in the library.

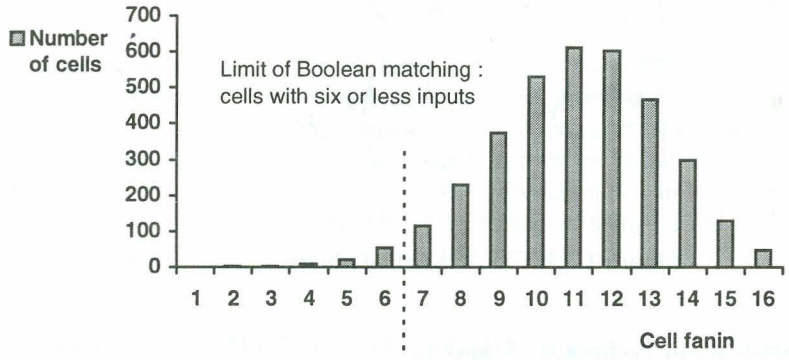


Figure 2 - Fan-in distribution of SCCG(4x4) virtual library.

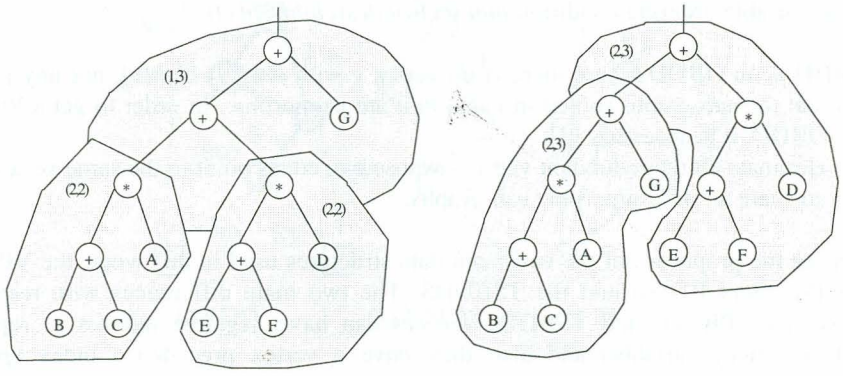
2.3.3 Dynamic covering

The covering process is normally done by dynamic covering of a tree with trees. Dynamic covering gives the optimal solution to the problem. Unfortunately, this is partially true, because the optimal solution of the covering problem depends on the initial ordering of the tree. Figure 3 shows optimal covers of two trees representing the same logic function (the cost function is the number of cells from SCCG(3,3), disregarding inverters).

2.3.4 Greedy structural covering

The method proposed in [1] is also based on tree representations. Anyway, there is no need of recognizing patterns because the use of inversions is limited to root and leaf nodes, allowing easy calculation of the number of serial transistors. The approach is based on decomposing trees into sub-trees by locally taking into account the number of serial transistors in the sub-trees. The method proposed in [2] is a similar method based on a

graph representation. The main problem of the greedy structural approaches is that only the local limitation of serial transistors is considered while constructing a SCCG structure.



(a) Bad initial ordering (b) Good initial ordering

Figure 3 - Dynamic covering of a tree with trees.

3 LIBRARY FREE TECHNOLOGY MAPPING

This section presents our approach to the Library Free Technology Mapping (LFTM) problem. Sub-section 3.1 reviews some basic concepts about BDDs and introduce TSBDDs. Our algorithm is presented in sub-section 3.2.

3.1 Binary Decision Diagram basics

Terminal suppressed BDDs (TSBDDs) are briefly presented in [8]. Bryant [3] has introduced ROBDDs as a canonical form to represent Boolean functions, by proving that there is a unique correspondence between a ROBDD and a Boolean function for a given variable ordering. TSBDDs are appropriate for CMOS technology mapping because a TSBDD will always match a SCCG and the transistor topology of the PMOS and NMOS transistor networks can be obtained by isomorphism with BDD arcs.

A Binary Decision Diagram (or **BDD**) is a rooted directed acyclic graph whose vertex set is V. Each non-leaf or non-terminal vertex has as attributes a pointer *index(v)* • {1, 2, ..., *n*} to an input variable in the set $\{x_1, x_2, ..., x_n\}$, and two children *low(v), high(v)* • V. A leaf or terminal vertex v has only one attribute value *value(v)* • *B*, where *B* is the binary set {0,1}.

In this paper, we say that every arc whose *head* is *v* and whose *tail* is *low(v)* is a S0 arc. Also, every arc whose *head* is *v* and whose *tail* is *high(v)* is a S1 arc.

An **OBDD** is a BDD where any non-terminal vertex pair {*v, low(v)*} and {*v, high(v)*} obeys the variable ordering condition *index(v)<index(children(v))*.

A **ROBDD** is an OBDD where there is no vertex *v* with *low(v)=high(v)*, nor any pair {*v, u*} such that the sub-graphs rooted in *v* and in *u* are isomorphic. In order to get a ROBDD from an OBDD, it is necessary [3]:

- to eliminate all the redundant vertices whose two edges point to the same vertex,
- to share all the isomorphic sub-graphs.
-

Now let see the properties of the two main data structures used in this work, the VPBDDs (Vertex Precedent BDDs) and the TSBDDS. The two main differences with respect to BDDs is that VPBDDs and TSBDDs vertices can have negative indexes to represent negated (inverted) variables and also they have a vertex precedence index *vp(v)* to represent the order the vertices appear in the graph.

A **VPBDD** (Vertex Precedent BDD) is a BDD where:
1) there is a path that passes through all non-terminal vertices, i.e. either *vp(low(v))=vp(v)+1* or *vp(high(v))=vp(v)+1*. This condition is called vertex precedence.
2) there is no vertex *v* with *low(v)=high(v)*.
3) the graph is serial/parallel, i.e. for each node v, the nodes *av* above *v* (defined by *vp(av)<vp(v)*) cannot have *vp(low(av))* neither *vp(high(av))* between the interval]*vp(v), max(vp(low(v)), vp(high(v)))*.

A **TSBDD** is defined as a VPBDD that has the following four properties:
1) Only S0 arcs connect the 1-terminal vertex, and they are suppressed without loosing information.
2) Only S1 arcs connect the 0-terminal vertex, and they are suppressed without loosing information.
3) All arcs having the same tail are either S0 arcs or S1 arc.
4) There is a path that passes through all non-terminal vertices (vertex precedence condition, as for VPBDDs).
5) The graph is serial/parallel, as for VPBDDs.

In short, if a VPBDD obey to the previous four conditions, then it is a TSBDD and it is possible to associate CMOS transistors directly (one-to-one association) with its arcs to obtain SCCGs [8], as shown in figure 4.

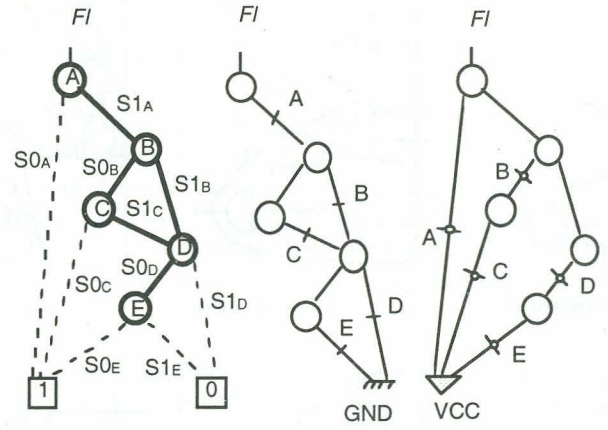


Figure 4 - Generation of a SCCG from a TSBDD of the function $Fl = \overline{A.(B+C)(D+E)}$.

3.1.3 Implementation issues

The Class of BDDs is implemented in an hierarchical way. As they are serial/parallel graphs, the data structure is composed of a virtual class triangle having two derived classes: BDD_node and BDD. These classes use a common interface given by triangle class. A BDD is composed of a list of triangles, a parallel node pointed by all triangles in the list and a serial node pointed by one of the triangles of the list. At each level the sons are not allowed to have the same parallel node as their father. If this situation happens the son is deleted and it is substituted by its sons (and one hierarchical level disappears).

3.2 Technology Mapping in TABA

Our LFTM tool (TABA - Transistor Assignment with TSBDDs representing Static CMOS Complex Gates) is presented here. Figure 5 shows the steps of the algorithm through an example. The input is a Boolean network composed of simple gates (5.a) that is decomposed in terms of logic cones (5.b) where the gates are interconnected with fan-out 1. For each logic cone, three steps are performed. First, a TSBDD representing the logic cone is constructed (5.c), next there is a limitation in the number of serial transistors taking into account polarity assignment (5.d) and finally the SCCGs are generated by association of transistors with TSBDD arcs (5.e). The final result is CMOS network (5.f) containing SCCGs obeying the serial transistor constraint used in step 5.d. Next we will discuss these steps in more detail.

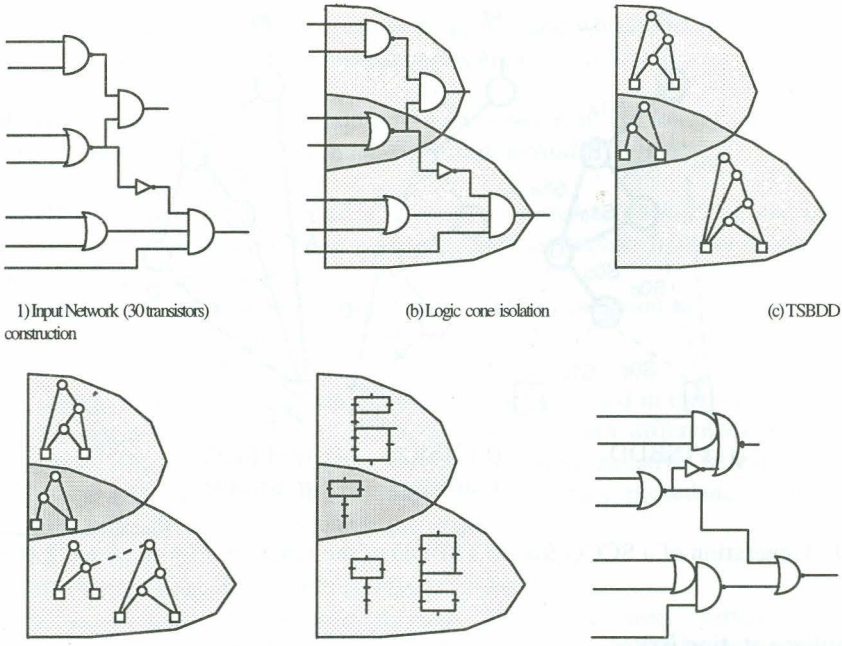


Figure 5 - Steps of TABA technology mapping.

3.2.1 VPBDD construction

The first step in TSBDD construction is the creation of a TSBDD for each cell in the initial network. As the initial network contains only simple gates, this operation is trivial. Afterwards, for every two combinational cells interconnected with fan-out 1, the following procedure is performed: 1) The two cells are deleted. 2) A new VPBDD is created by collapsing the two VPBDDs of the cells. 3) A new cell is created to encapsulate the VPBDD resulting from step 2. This procedure is repeated till no more combinational cells interconnected with fan-out 1 are found. The resulting structure is a set of VPBDDs interconnected with fan-out greater than one (otherwise they would be collapsed). In order to obtain a network of SCCGs under a constraint of serial transistors, two operations are need. First, the number of serial arcs must be limited to respect the constraint of serial transistors. Second, the polarity assignment must transform the VPBDDs in TSBDDs, in order to obtain a description that is realizable in CMOS technology. The quality of the final network will depend on the number of cuts inserted to limit serial arcs of the original VPBDD, as well as on the number of inverters used to obtain a polarity assignment realizable in CMOS technology.

3.2.2 Limitation of serial transistors

Fortunately, the number of serial arcs obtained when transforming a VPBDD in a TSBDD does not change with the chosen polarity. Polarity assignment will only change the plan where the transistors will be in serial/parallel. When using libraries that are closed under inversion, like SGGG(2,2), SCCG(3,3) and SCCG(4,4) these two problems can be treated separately. This sub-section introduces an algorithm that minimizes the number of cuts inserted to limit the number of serial arcs, while the next sub-section will treat about polarity assignment.

The algorithm for serial transistor limitation is based on the way the number of serial arcs is calculated in a VPBDD. The VPBDD is hierarchical and there are two costs by level : CS and CP, which represent the number of serial and parallel transistors, respectively. The final costs being CS and CP of the upper level. Leaf nodes have CS and CP set to 1.
The cost of non-leaf levels are calculated as follows:

$$CS_{father} = \sum CP_{sons}$$

$$CP_{father} = MAX(CS_{sons})$$

The serial transistor limitation is performed in one level at a time. The algorithm descends till it finds a level where CS is greater than the serial transistor constraint and where all sons have CS and CP OK. Then the algorithm chooses a sub-set of the sons where the sum of CPs is the maximum value equal or smaller than the serial transistor constraint that it is possible to obtain. This sub-set is then substituted by a single node. The extracted subset obeys to the serial transistor constraint, and the process is repeated over the remaining VPBDD till it also obeys to the serial transistor constraint. As the sons can be selected in any order, the algorithm is not dependent on the initial order of the Boolean equation as it happens in dynamic covering.

3.2.3 Polarity assignment and SCCG generation

The first step of the polarity assignment algorithm is the assignment of a cost for each net in the network. Nets with high fan-out have a zero cost (because they will need buffer insertion, so the cost of getting the inverted signal is zero). The polarity of each cell is set in order to eliminate every inverter inside the logic cones, because the nets inside the logic cone are not charged nets. At the end of the process, the inverters will be at the borders of the logic cone, that are charged nets or inputs of the circuit, or both. If the cost of the cone is high, the polarity of every cell is changed and this will lead to a polarity assignment with lower cost and without inverters inside the cone.

3.2.4 Design Environment

TABA is conceived to be integrated in a design flow composed of a logic synthesis tool at the front end and a layout generator at the back end. At the moment, we are using SIS as the logic synthesis tool and TROPIC [7] as the layout synthesis tool.

4 VALIDATION AND RESULTS

Comparative results between TABA and [2] are shown in table 1. For a set of nine IWLS93 benchmarks mapped on SCCG(4,4) TABA reduces the number of transistors by an average of 17%. In the better case, TABA wins by 36% and in the worst case TABA looses by 12%. The comparisons are only presented for the SCCG(4,4) because this is the only SCCG set used for the results presented in [2].

Table 1: Comparison between Berkelaar (1988) [2] and TABA

	results from Berkelaar			*results from TABA(4,4)*		
Circuit	*trans*	*SCCGs*	*invs*	*trans*	*SCCGs*	*invs*
5xp1	302	31	8	302	25	26
9sym	620	63	22	404	30	35
bw	532	63	22	516	59	39
duke2	1224	142	53	1110	131	92
misex1	132	14	5	148	13	12
rd53	120	8	1	82	11	6
rd73	274	24	10	174	19	16
rd84	384	40	18	290	30	25
sao2	486	53	22	362	32	33
Total	4074			3388		
%Total	100%			83%		

Table 2 shows comparative results between TABA and [1]. These benchmarks are first mapped on SCCG(3,3), because the results from [1] use only this set. In the better case,

TABA wins by 70% and in the worst case TABA looses by 8%. TABA reduces again the number of transistors by an average of 17% when compared to [1]. This average goes to 24%, when compared to TABA mapping on SCCG(4,4).

Table 2: Comparison between Abouzeid (1992) [1] and TABA

	Abouzeid		*TABA(3,3)*		*TABA(4,4)*	
Circuit	*trans*	*SCCGs*	*trans*	*SCCGs*	*trans*	*SCCGs*
5xp1	408	49	326	31	302	25
9sym	714	77	448	41	404	30
bw	510	64	552	68	516	59
misex1	212	30	164	17	148	13
rd53	182	22	86	12	82	11
rd73	576	63	186	22	174	19
rd84	1020	110	302	33	290	30
sao2	504	59	406	43	362	32
hyeti1	9714	1003	8704	889	7880	683
hyeti2	6500	685	5740	602	5208	469
Total	20340		16914		15366	
%Total	100%		83%		76%	

5 CONCLUSION

The Library Free Technology Mapping problem was introduced and discussed. We have presented a method for mapping a set of Boolean equations into a set of Static CMOS Complex Gates (SCCGs) based on a new class of Binary Decision Diagrams, the TSBDDs.

This method was implemented in a tool called TABA and it has two main features. First, it exploits the re-ordering of the structure to obtain a minimal number of complex gates. Second, it takes into account polarity assignment at early stages and inverter minimization

is carried out together with buffer insertion and fan-out limitation. Experimental results have demonstrated that this method gives a better reduction in the overall number of transistors when compared to previously published results for Library Free Technology Mapping.

References

[1] Abouzeid, P.; Leveugle, R.; Saucier, G. and Jamier. R. (1992) Logic synthesis for automatic layout. *Proc. of EUROASIC*, 146-51.

[2] Berkelaar, M. and Jess, J. (1988) Technology mapping for standard-cell generators. *ICCAD*, 470-3.

[3] Bryant, R.E. (1986) Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, vol. C-35, n° 8, 677-91.

[4] Detiens, E.; Gannot, G.; Rudell, R.; Sangiovanni-Vinccentelli, A.L. and Wang, A. (1987) Technology mapping in MIS. *Proc. of ICCAD*, 116-9.

[5] Keutzer, K. (1987) DAGON: Technology Binding and Local Optimization by DAG Matching. *Proc. of 24th DAC*, 341-7.

[6] Mailhot, F. and DeMicheli, G. (1993) Algorithms for technology mapping based on binary decision diagrams and on Boolean operations. *IEEE Transactions on CAD for IC and Systems*, vol. 12, n° 5, 599-620.

[7] Moraes, F.; Azemard, N.; Robert, M. and Auvergne, D. (1993) Flexible Macrocell Layout Generator. *4th ACM/SIGDA Physical Design Workshop*, 105-16.

[8] Reis, A.; Robert, M.; Auvergne, D. and Reis, R. (1995) Associating CMOS transistors with BDD arcs for technology mapping. *IEE Electronics Letters*, vol. 31, n°14, 1118-20.

Este trabalho recebeu o Best Paper Award on Design Methods and CAD, na conferência VLSI'97, realizada em Gramado, RS, Brasil, agosto de 1997.