A CIRCUIT COMPARISON SYSTEM WITH RULE-BASED FUNCTIONAL ISOMORPHISM CHECKING

Makoto Takashima*, Atsuhiko Ikeuchi*, Shoichi Kojima**,
Toshikazu Tanaka**, Tamaki Saitou*** and Jun-ichi Sakata***

* Toshiba Corporation, Integrated Circuit Division
580-1, Horikawa-cho, Saiwai-ku, Kawasaki 210, Japan
** Toshiba Corporation, Systems & Software Engineering Laboratory
*** Toshiba Microcomputer Engineering Corporation

Abstract

This paper describes a circuit comparison system which compares two networks and points out inconsistencies. A new approach is used to handle functionally isomorphic circuits which most conventional programs can not handle. Three techniques are included: network reduction, graph isomorphism-based comparison and rule-based functional isomorphism checking for inconsistencies. This system is efficient even for large networks and can eliminate false errors in a flexible manner.

1. Introduction

Manual and semiautomated layout design are still major methods to design LSIs which require minimum chip area or high performance. In such design methods, layout verification is indispensable to remove human errors before mask fabrication. Under today's LSI complexity, manual checking is no longer a reliable method and so automated layout verification plays an important role.

Circuit comparison is a typical connectivity verification method, which extracts a network from layout and compares it with a reference network usually extracted from schematics. In general, network comparison is a cpu intensive problem which is known as a graph isomorphism problem. Therefore, conventional computer programs focussed their attentions on the efficiency to solve the problem. However, in addition to this, a sophisticated comparison program must handle topologically non-isomorphic but functionally isomorphic circuits, because layout designers often perform functionally equivalent transformations in order to optimize electrical performance or layout area. This means not only transistor permutations but also more complex transformations. Many comparison algorithms have been proposed [1]-[8], but only [8] is known to the authors that describes the above capability. Other programs only handle topological isomorphism or consider transistor permutations at most. On the other hand, a rule-based expert system has been applied to circuit comparison [9] suggesting the possibility of handling functional isomorphism. However, a practical use of this method is limited to relatively small circuits.

This paper describes a circuit comparison system called CCOMP/EX. A new approach is used which is based on the following techniques: (1) transistor network reduction to cope with transistor permutations (2) network comparison based on a graph isomorphism algorithm (3) a rule-based method to check the inconsistencies against functional isomorphism.

This approach enables CCOMP/EX to be efficient even for large networks and provides a flexible capability to eliminate false errors due to functionally isomorphic transformations.

2. System Overview

Figure 1 illustrates the overall system configuration. CCOMP/EX inputs two networks: an actual network usually extracted from layout and a reference network usually extracted from schematics. However, in practice, CCOMP/EX can accept any combination of input networks and layout versus layout comparison is also an important application of CCOMP/EX.

Circuit extraction from layout is performed by EMAP [10]. EMAP extracts devices and their connectivity with parameters such as device size and node capacitance in a technology independent manner. Extracted circuit information is stored in the database for use by CCOMP/EX and other verification tools. CCOMP/EX is also designed to support not only MOS circuits but also bipolar circuits and their hybrid circuits.

Network extraction from schematics is performed by a schematic capture system implemented on engineering work-stations. Extracted network data is output as TDL (Toshiba Standard Net Description Language) or SPICE descriptions. In case of TDL, a macro expansion program is used with a library to decompose logic components into transistor circuits.

CCOMP/EX is composed of several programs which perform network data input, reduction, comparison, functional isomorphism check and error output. As

a result of comparison, two files are created: a matched element file and an unmatched element file. These files are modified by the functional isomorphism checking subsystem and finally used by output programs. CCOMP/EX output methods include a list, a plot and a graphics display. Error devices and nodes are output on a reference network side as well as on an actual network side.

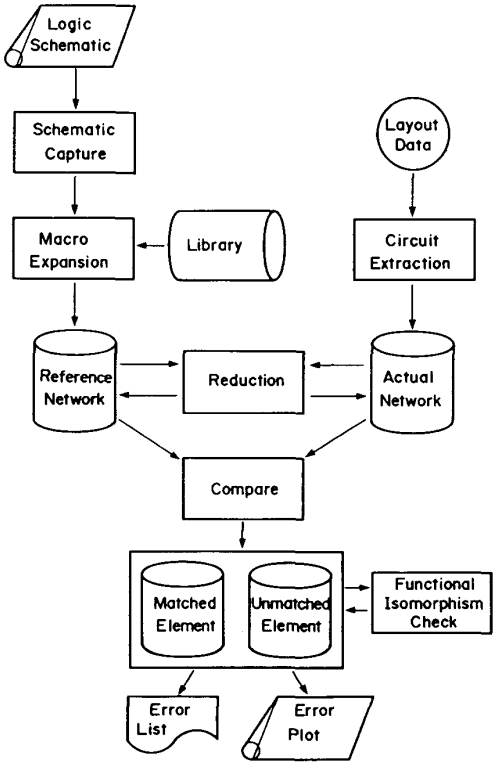

Figure 1 System configuration

3. Reduction

The primary purpose of network reduction is to translate input transistor networks into permutation free networks. An additional advantage of this process is that it can considerably reduce the size of networks. Figure 2 illustrates example CMOS circuits which are functionally equivalent but topologically different. In the figure, the following permutations have been performed: in the pull-down part, between a terminal A and a set of terminals (B, C); in the pull-up part, between terminals B and C.

To resolve such topological inconsistencies, the reduction program translates series-parallel connected transistors into pseudo-logic gates. This process consists of a series and parallel reduction processes applied alternately. Each process replaces series or parallel connected transistors by a single transistor and generates a pseudo-logic gate. When series or parallel connected transistors are redundant, that is their gate terminals are common, they are reduced without generating a pseudo-logic gate. Figure 3 illustrates a reduced circuit which is translated from both circuits shown in figure 2.

In the reduction process, compound device size is calculated and is associated with the pseudo-logic gate. The following equations are used:

$Ws = Min\ (Wi);\ Ls = Ws * \Sigma\ (Li/Wi)$

$Lp = Min\ (Li);\ Wp = Lp * \Sigma\ (Wi/Li)$

where Ls (Ws) means compound channel length (width) for series connected transistors and Lp (Wp) means compound channel length (width) for parallel connected transistors.

The reduction program also handle bipolar devices. However, for bipolar transistors, only redundancy elimination is perfomed. Resistors in series-parallel structures are reduced to a single resistor with a compound resistance value.

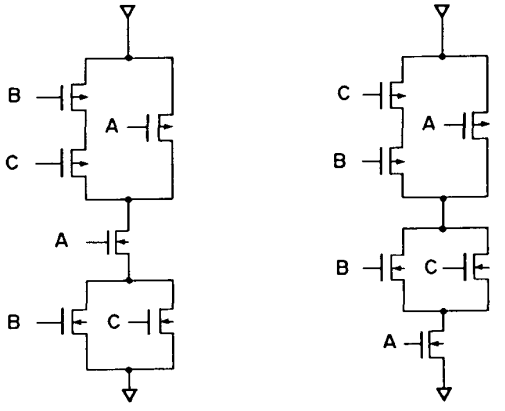

Figure 2 Functionally equivalent but topologically different CMOS circuits

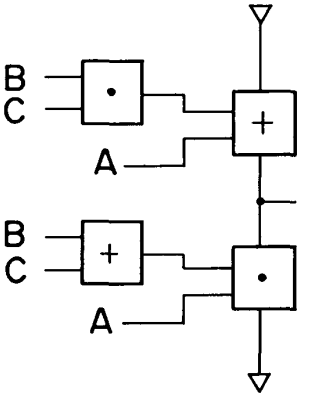

Figure 3 Reduced circuit for circuits shown in figure 2

4. Comparison

The comparison algorithm used in CCOMP/EX is based on a graph isomorphism algorithm which can satisfy the following requirements:
(1) to compare large networks efficiently
(2) to compare whole networks using only initial matching data for pads
(3) to point out inconsistent devices and nodes with high precision

The algorithm is nearly the same that has been described in the author's previous paper [11], but the following modifications have been newly incorporated: First, a graph representation is applied to reduced networks instead of transistor networks. Second, in order to handle bipolar devices which have three terminals to be distinguished, a weighted graph representation is used where each edge is associated with an integer weight according to device terminal types. Accompanied with this, a key value Adj which is used to partition vertex groups is extended to 3-tuple.

As an option, device size comparison is supported, which is performed for matched devices including pseudo-logic gates under user specified error tolerance. Since this check requires device size data, it is limited to comparison for layout extracted networks and spice descriptions.

5. Rule-Based Functional Isomorphism Checking

As functionally isomorphic transformations other than transistor permutations can not be coped with by the reduction process, they are pointed out as inconsistencies by the comparison program. However, for most designers, such inconsistencies should not be output as errors. The subsystem described in this section finds out such inconsistencies and eliminates them from the unmatched element file.

To implement this function, a rule-based method has been adopted, because such inconsistencies are caused by expert techniques to optimize the design and it seems to be difficult to devise an algorithm which can solve the problem to a full extent. On the other hand, a rule-based system provides a mechanism to simply implement the function and gradually increase the system's capability.

This subsystem consists of three parts as illustrated in figure 4. A preprocessing program extracts subcircuits which include unmatched elements as constituents and then extracts their corresponding subcircuits from another network to make pairs of subcircuits. Here, a subcircuit is defined as a set of transistors which are connected with the source or drain terminal and bounded by a power supply node, a ground node and gate terminal connected nodes other than the output node. Note that, in practice, the subcircuit extraction is performed on reduced networks. A corresponding subcircuit is extracted using a matched element in the other subcircuit as a clue. After that, if there remains a matched element whose pair is not included in the corresponding subcircuit, it becomes a new clue to continue the extraction process. This process is repeated until such a situation is resolved.

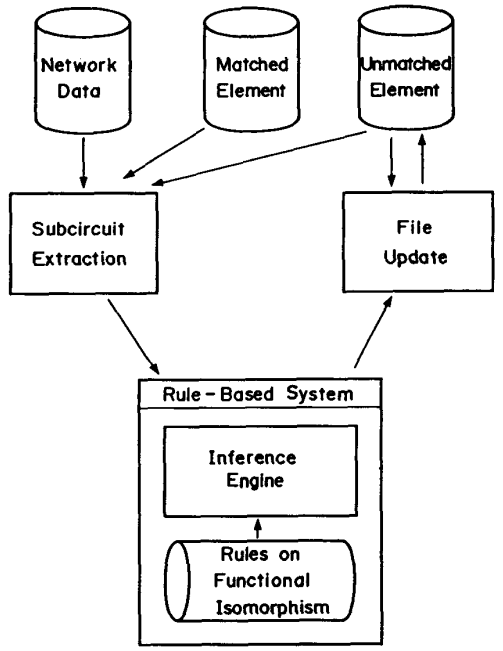

Figure 4 Configuration of functional isomorphism checking subsystem

The rule-based system is implemented using ARCH [12], an inhouse developed expert system building tool. It is composed of a rule base, an inference engine and a working memory and thus supports a production system. The following scheme is used to build the rule-based system:

(1) Represent pairs of functionally isomorphic circuits and input and output node correspondences between the subcircuits as patterns in the rules.

(2) Represent each pair of extracted subcircuits and node correspondences made by the comparison program in a working memory.

(3) In the above context, testing functional isomorphism is reduced to pattern matching between a working memory and rules, which is performed by an inference engine.

When representing a circuit in the rule, there are several problems to be solved. First, permutable terminals must be properly represented. As for this, two formats, a permutable format and a non-permutable format, have been discussed in [9]. In our system, a permutable format is used to reduce the number of rules to be prepared. For example, a 2-input pseudo-logic gate is described as follows:

```
(Element ^name <el> ^type NESRG
         ^out  <n0> ^inum  2 ^side <side>)
(Connect ^node <n1> ^elem <el> ^term IN)
(Connect ^node {<n2> > <n1>} ^elem <el> ^term IN)
```

In the above description, the first condition element represents attributes of the component with its output terminal and the following two condition elements represent permutable input terminals. A predicate ">" is used for preventing duplicate matching, that is, (<n1>, <n2>) matches (n1, n2), (n2, n1), (n1, n1) and (n2, n2) without it. Second, since the number of condition elements allowed in a rule is limitd, a large subcircuit must be partitioned into several rules. To realize this, a hierarchical matching method has been used, where each rule replaces a section of a subcircuit by a block with input and output terminals when a match is occurred.

The rule-based system operates with an input and output procedures. The input procedure reads a pair of subcircuits data with node correspondences into a working memory at the start of each inference process. The output procedure is called, when a match is found, to write the subcircuit data to an output file. Using the file, a post-processing program modifies the unmatched element file.

Currently, rules on CMOS circuits are entered in the rule base. Two examples are illustrated in figure 5 at a transistor level. The current system does not handle rules which extend over several subcircuits. For example, 1-stage inverter is functionally equivalent to 3-stage inverters, but this rule can not be handled because 1-stage inverter forms a subcircuit. However, it is possible to improve the system to support such rules.

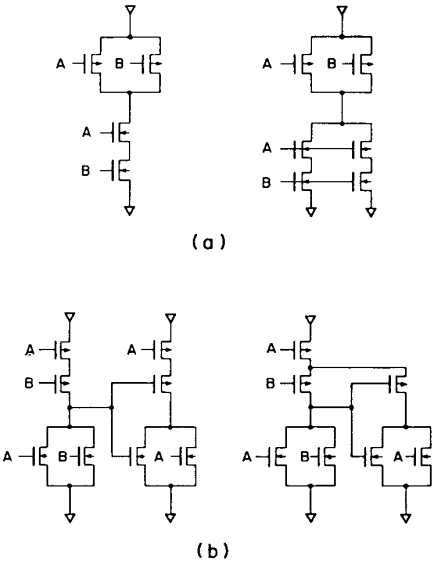

Figure 5 Examples of functionally isomorphic circuits entered as rules

6. Results

All the programs except the rule-based system are implemented on a 14 MIPS mainframe computer using FORTRAN. This part of the system has been applied to a large number of real LSI chips. Table 1 lists the execution time for the reduction and comparison programs on several chips. From this result, it can be seen that the cpu time required for reduction is sufficiently small and the time complexity for the comparison program is normally nearly linear with the number of transistors.

The rule-based system is implemented on AS3000, a 2 MIPS engineering workstation, which is connected to a mainframe computer through a local area network. A prototype system has been developed and applied to several CMOS LSI chips for evaluation. Table 2 lists the run time for the rule-based system with the number of input and matched subcircuits and the maximum subcircuit size. The cpu time required for preprocessing on a mainframe computer is also listed in the table. For all cases, false errors included in the input subcircuits have been all successfully detected. The result also shows that the cpu time remains within a practical range even when a large subcircuit is included.

Table 1 Run time for reduction and comparison

chip	transistors	reduction cpu time (sec.)	comparison cpu time (sec.)
A	7480	2	26
B	35120	10	137
C	61000	21	300

Table 2 Run time for the rule-based system

Chip	trans.	input subc.	max. size of subc.	matched subc.	cpu time (sec.)	cpu for pre-proc.
D	5600	9	55	0	751	2
E	7800	14	16	4	156	3
F	15000	32	16	1	192	8

7. Conclusion

A circuit comparison system has been described focussing on the techniques to handle functionally isomorphic circuits. Network reduction to obtain permutation free networks, graph isomorphism-based comparison and rule-based functional isomorphism checking for inconsistencies have been used. Owing to this approach, CCOMP/EX can eliminate false errors in a flexible manner while keeping efficiency even for large networks. The algorithm-based part of CCOMP/EX has been in

production use for two years contributing to provide functional chips at the first design cycle. The rule-based method has been shown to be practical in performance through the evaluation of a prototype system. Our future work includes adding rules through its praccical use in the production environment.

Acknowledgement

The authors would like to thank Mr. K. Yoshida for his encouragement and useful comments on this paper. They would like also to thank Mr. M. Koike and Mr. T. Iio for their helpful discussions.

References

[1] N. Kubo and I. Shirakawa, "A Fast Algorithm for Testing Graph Isomorphism", Proc. ISCAS, pp. 641-644, 1979

[2] I. Ablasser and U. Jaeger, "Circuit Recognition and Verification Based on Layout Information", Proc. 18th Design Automation Conference, pp. 684-689, June 1981

[3] N. Miyashita, T. Watanabe, M. Endo and S. Yamada, "A new CAD System for Automatic Logic Interconnection Verification", Proc. ISCAS, pp. 114-117, 1981

[4] R. L. Spickelmier and A. R. Newton, "WOMBAT: A New Connectivity Verification Program", Proc. ICCAD 83, pp. 170-171, November 1983

[5] E. Barke, "A Network comparison Algorithm for Layout Verification of Integrated Circuits", IEEE Trans. on CAD, vol. CAD-3, no. 2, pp. 135-141, april 1984

[6] T. Sakata and A. Kishimoto, "A Circuit Comparison System for Bipolar Linear LSI", Proc. 22nd Design Automation Conference, pp 429-434, June 1985

[7] J. D. Tyger and R. Ellickson, "Efficient Netlist Comparison Using Hierarchy and Randomization", Proc. 22nd Design Automation Conference, pp. 702-708, June 1985

[8] Y. Shiran, "YNCC: A New algorithm for Device-Level Comparison Between Two Functional Isomorphic VLSI Circuits", Proc. ICCAD 86, pp. 298-301, November 1986

[9] R. L. Spickelmier and A. R. Newton, "Connectivity Verification Using a Rule-Based Approach", Proc. ICCAD 85, pp. 190-192, November 1986

[10] T. Mitsuhashi, T. Chiba, M. Takashima and K. Yoshida, "An Integrated Mask Artwork Analysis System", Proc. 17th Design Automation Conference, pp. 277-284, June 1980

[11] M. Takashima, T. Mitsuhashi, T. Chiba and K. Yoshida, "Programs for Verifying Circuit Connectivity of MOS/LSI Mask Artwork", Proc. 19th Design Automation Conference, pp. 545-550, June 1982

[12] A. Nakamura, et al., "Rete Compiler of Production System ARCH" and "Interpreter of Production System ARCH", Proc. 34th National Conference of Information Processing Society of Japan, pp. 1611-1614, March 1987