3

LOGIC SYNTHESIZERS,
THE TRANSDUCTION METHOD
AND ITS EXTENSION, SYLON

Saburo Muroga

Department of Computer Science
University of Illinois at Urbana-Champaign
1304 W. Springfield Ave., Urbana, IL 61801, USA

ABSTRACT

Logic synthesizer, the Transduction method, was developed in the early 1970s at the University of Illinois, but its usefulness has been only recently recognized in the industry. The original Transduction method handles only NOR gates, though it is still useful for the design of ECL networks. So, since then, SYLON has been developed as its extension, focusing on the design of CMOS circuits. Currently SYLON has three programs, i.e., XTRANS, DREAM, and REDUCE. Each has its own unique features. The basic concepts of the Transduction method and SYLON are discussed along with recent results.

3.1 INTRODUCTION

Automated design of logic networks has been attempted since the early 1960s because such logic synthesizers can reduce design time and mistakes in logic design. But it has been more difficult than automation of lower level designs, such as routing or electronic circuit simulation. Since the beginning of the 1960s, IBM has pushed research of design automation in general. Outside IBM, the author's group probably has been the only one that has continued to work on automated design of logic networks (as outlined in [28]). During the 1970s, assembling integrated circuit packages which contain standard logic networks, instead of designing logic networks by designers by themselves, became the standard practice of manufacturing of computers due to the advent of integrated circuitry. Logic networks which differentiate computers of a manufacturer from
competitors were not many and small and consequently could be designed by hand with satisfactory results. Then, during the 1980s, a single chip could contain a lot more logic gates due to the progress of VLSI technology. Also due to the progress of fabrication technology of VLSI chips, semi-custom design of VLSI chips, or so-called ASIC, such as gate arrays became the most economical approach of manufacturing of chips in small production volume. During this period, the number of logic gates which a single gate array could contain enormously increased and the logic design became beyond manual design. Thus, research on automated logic design came back. IBM, for example, resumed active research of automated design of logic networks since 1980. During the 1990s, automated logic design by logic synthesizers will be extensively used in industry.

This paper presents logic synthesizers of compact multi-level networks, the Transduction method and its extension, SYLON, which we have been developing for years. Multi-level networks can be faster and more compact than two-level networks (or PLA’s) but is inherently a more difficult problem.

3.2 TRANSDUCTION METHOD

The design of a minimal logic network has been one of the most important objectives of switching theory. However, there have been no general design methods known in switching theory that design minimal networks with a mixture of different types of gates, which are more complex than AND or OR gates, and also can take into consideration the arbitrary constraints, such as maximum fan-in and fan-out restrictions, because it was difficult to formulate the concept of minimization within the framework of Boolean algebra which has been a major mathematical tool in conventional switching theory. Having worked on the application of integer programming to logic design since 1965 (summarized in [28]), however, we found that logic design of a minimal network under arbitrary network constraints was computationally feasible with integer programming when networks consist of simple gates, such as NOR gates, NAND gates, AND gates, OR gates, or their mixture. We can minimize the number of gates, the number of connections, or a mixture of them with different weights. We can derive in reasonable processing time such minimal networks under constraints such as maximum fan-in, maximum fan-out, or maximum number of levels, if the networks do not require many gates. This integer-programming logic design method is useful for cells or modules which consist of a small number of logic gates.