Chapter 1
Introduction

Sunil P. Khatri and Kanupriya Gulati

With the advances in VLSI technology, enhanced optimization techniques are required to enable the design of faster electronic circuits that consume less power and occupy a smaller area. In the VLSI design cycle, significant optimization opportunities exist in the logic design stage. In recent times, several research works have been proposed in the area of logic synthesis, which can prove to be very valuable for VLSI/CAD engineers. A solid understanding and sound implementation of these advanced techniques would enable higher levels of optimization, and thus enable better electronic design. This text is a systematic collection of important recent work in the field of logic design and optimization.

Conventional logic synthesis consists of the following phases. Given an initial netlist, technology-independent optimizations [1, 3, 4] are first carried out in order to optimize various design criteria such as gate, literal, and net count. Both Boolean and algebraic techniques are used, such as kernel and cube extraction, factorization, node substitution, don't care-based optimizations [2]. During the logic decomposition phase, large gates are decomposed into smaller gates, which allows for efficient technology mapping and technology-dependent optimizations. Finally, technology mapping is applied on the decomposed netlist which is followed by technology-dependent optimizations. This book presents recent research in some of the above areas.

In order to enhance the scalability and performance of logic synthesis approaches, newer optimization styles are continually investigated. Boolean satisfiability (SAT) plays a big role in some of the recent logic optimization methodologies. Therefore, this edited volume also includes some of the latest research in SAT techniques. Further, several non-CAD systems can be viewed as an instance of logic optimization, and thus these too can take advantage of the rich body of recent research in logic synthesis and optimization. Such non-traditional applications of logic synthesis to specialized design scenarios are also included in this volume.

S.P. Khatri (✉)
Department of ECE, Texas A&M University, College Station, TX, USA
e-mail: sunilkhatri@tamu.edu

S.P. Khatri, K. Gulati (eds.), *Advanced Techniques in Logic Synthesis, Optimizations and Applications*, DOI 10.1007/978-1-4419-7518-8_1,

The approaches described in this text are enhanced and archival versions of the corresponding original conference publications. The modifications include enhancements to the original approach, more experimental data, additional background and implementation details, along with as yet unpublished graphs and figures.

The different sections of this volume are described next.

1.1 Logic Decomposition

This section discusses the latest research in logic decomposition. The first chapter investigates restructuring techniques based on decomposition and factorization. In this chapter the authors describe new types of factorization that extend Shannon cofactoring, using projection functions that change the Hamming distance of the original minterms, to favor logic minimization of the component blocks.

The next chapter uses reachable state analysis and symbolic decomposition to improve upon the synthesis of sequential designs. The approach described uses under-approximation of unreachable states of a design to derive incomplete specification of the combinational logic. The resulting incompletely specified functions are decomposed to optimize technology-dependent synthesis. The decomposition choices are implicitly computed by using recursive symbolic bi-decomposition.

The third chapter in the topic of Boolean decomposition employs fast Boolean techniques to restructure logic networks. The techniques used are a cut-based view of a logic network, and heuristic disjoint-support decompositions. Local transformations to functions with a small number of inputs allow fast manipulations of truth tables. The use of Boolean methods reduces the structural bias associated with algebraic methods, while still allowing for high-speed.

The fourth chapter investigates Ashenhurst decomposition wherein both single and multiple output decomposition can be formulated with satisfiability solving, Craig interpolation, and functional dependency. In comparison to existing BDD-based approaches for functional decomposition, Ashenhurst decomposition does not suffer from memory explosion and scalability issues. A key feature of this approach is that variable partitioning can be automated and integrated into the decomposition process without the bound-set size restriction. Further, the approach naturally extends to nondisjoint decomposition.

The last chapter in the Boolean decomposition section focuses on scalability and quality of Boolean function bi-decomposition. The quality of a bi-decomposition is mainly determined by its variable partition. Disjoint and balanced decompositions reduce communication and circuit complexity and yield simple physical design solutions. Furthermore, finding a good or feasible partition may require costly enumeration, requiring separate decomposability checks. This chapter uses interpolation and incremental SAT solving to address these problems.

1.2 Boolean Satisfiability

In the area of Boolean satisfiability, this book presents some key ideas to make SAT more effective. The first chapter studies resolution proofs using boundary points elimination. Given a CNF formula F, boundary points are complete assignments that falsify only certain clauses of the formula. Since any resolution proof has to eventually eliminate all boundary points of F, this approach focuses on resolution proofs from the viewpoint of boundary point elimination. The authors use equivalence checking formulas to compare unsatisfiability proofs built by a conflict-driven SAT-solver. They show how every resolution of a specialized proof eliminates a boundary point, and how this enables building resolution SAT-solvers that are driven by elimination of cut boundary points.

The next chapter presents a methodology called SAT sweeping for simplifying And-Inverter Graphs (AIGs) by systematically merging graph vertices in a topological fashion starting from the inputs, using a combination of structural hashing, simulation, and SAT queries. This chapter presents the details of a SAT-sweeping approach that exploits local observability don't cares (ODCs) to increase the number of vertices merged. In order to enhance the efficiency and scalability of the approach, the authors bound the ODCs and thus the computational effort to generate them. They demonstrate that the use of ODCs in SAT sweeping results in significant graph simplification, with great benefits for Boolean reasoning in functional verification and logic synthesis techniques.

SAT-solvers find a satisfiable assignment for a propositional formula, but finding the "optimal" solution for a given function is very expensive. The next chapter discusses MIN-ONE SAT, an optimization problem which requires the satisfying assignment with the minimal number of ones, which can be easily applied to minimize an arbitrary linear objective function. The chapter proposes an approximation algorithm for MIN-ONE SAT that is efficient and achieves a tight bound on the solution quality. RelaxSAT generates a set of constraints from the objective function to guide the search, and then these constraints are gradually relaxed to eliminate the conflicts with the original Boolean SAT formula until a solution is found. The experiments demonstrate that RelaxSAT is able to handle very large instances which cannot be solved by existing MIN-ONE algorithms. The authors further show that RelaxSAT is able to obtain a very tight bound on the solution with one to two orders of magnitude speedup.

The last chapter in the Boolean satisfiability category presents an algorithm for MaxSAT that improves existing state-of-the-art solvers by orders of magnitude on industrial benchmarks. The proposed algorithm is based on efficient identification of unsatisfiable subformulas. Moreover, the new algorithm draws a connection between unsatisfiable subformulas and the maximum satisfiability problem.

1.3 Boolean Matching

Three research works are presented under the Boolean matching category. The first work proposes a methodology for Boolean matching under permutations of inputs and outputs that enables incremental logic design by identifying sections of netlist that are unaffected by incremental changes in design specifications. Identifying and reusing the equivalent subcircuits accelerates design closure. By integrating graph-based, simulation-driven, and SAT-based techniques, this methodology makes Boolean matching feasible for large designs.

The second approach in the Boolean matching category is DeltaSyn, a tool and methodology for generating the logic difference between a modified high-level specification and an implemented design. By using fast functional and structural analysis techniques, the approach first identifies equivalent signals between the original and the modified circuits. Then, by using a topologically guided dynamic matching algorithm, reusable portions of logic close to the primary outputs are identified. Finally, functional hash functions are employed to locate similar chunks of logic throughout the remainder of the circuit. Experiments on industrial designs show that together, these techniques successfully implement incremental changes while preserving an average of 97% of the pre-existing logic.

The last approach discussed in the Boolean matching section proposes an incremental learning-based algorithm, along with a Boolean satisfiability search, for solving Boolean matching. The proposed algorithm utilizes functional properties like unateness and symmetry to reduce the search space. This is followed by the simulation phase in which three types of input vector generation and checking methods are used to match the inputs of two target functions. Experimental results on large benchmark circuits demonstrate that the matching algorithm can efficiently solve the Boolean matching for large Boolean networks.

1.4 Logic Optimization

The first advanced logic optimization approach presents algebraic techniques to enhance common sub-expression extraction to allow circuit optimization. Common sub-expression elimination (CSE) is a useful optimization technique in the synthesis of arithmetic datapaths described at the RTL level.

The next chapter investigates a comprehensive methodology to automate logic restructuring in combinational and sequential circuits. This technique algorithmically constructs the required transformation by utilizing Set of Pairs of Function to be Distinguished (SPFDs). SPFDs can express more functional flexibility than traditional don't cares and have been shown to provide additional degrees of flexibility during logic synthesis. In practice, however, computing SPFDs may suffer from memory or runtime problems. This approach presents Approximate SPFDs (ASPFDs) that approximate the information contained in SPFDs by using the results

of test-vector simulation, thereby yielding an efficient and robust optimization platform.

The third chapter presents an approach to enhance the determinization of a Boolean relation by using interpolation. Boolean relations encapsulate the flexibility of a design and are therefore an important tool in system synthesis, optimization, and verification to characterize solutions to a set of Boolean constraints. For physical realization, a deterministic function often has to be extracted from a relation. Existing methods are limited in their handling of large problem instances. Experimental results for the interpolation-based relation determinization approach show that Boolean relations with thousands of variables can be effectively determinized and the extracted functions are of reasonable quality.

In this section, the fourth approach presented is a scalable approach for dual-node technology-independent optimization. This technique scales well and can minimize large designs typical of industrial circuits. The methodology presented first selects the node pairs to be minimized that are likely to give gains. For each node pair, a window or a subnetwork is created around the nodes. This windowing is done in order to allow the approach to scale to larger designs. Once the subnetwork is created, the Boolean relation, which represents the flexibility of the nodes, is computed. During this process, early quantification is performed which further extends the scalability of the approach. The Boolean relation is minimized, and the new nodes replace the original nodes in the original circuit. These steps are repeated for all selected node pairs. The authors experimentally demonstrate that this technique produces minimized technology-independent networks that are on average 12% smaller than networks produced by state-of-the-art single-node minimization techniques.

1.5 Applications to Specialized Design Scenarios

This volume presents applications of logic synthesis in non-traditional CAD areas. The first approach investigates techniques for synthesizing logic that generates new arbitrary probabilities from a given small set of probabilities. These ideas can be used in probabilistic algorithms. Instead of using different voltage levels to generate different probability values, which can be very expensive, the technique presented in the chapter alleviates this issue by generating probabilities using combinational logic.

The next chapter presents a gate-level probabilistic error propagation model which takes as input the Boolean function of the gate, the signal and error probabilities of the gate inputs, and the gate error probability and produces the error probability at the output of the gate. The presented model uses Boolean difference calculus and can be applied to the problem of calculating the error probability at the primary outputs of a multilevel Boolean circuit. The time complexity of the approach is linear in the number of gates in the circuit, and the results demonstrate

the accuracy and efficiency of the approach compared to the other known methods for error calculation in VLSI circuits.

In the third chapter in the applications category, a novel realization of combinational logic circuit is presented. In this approach, logic values 0 and 1 are implemented as sinusoidal signals of the same frequency that are phase shifted by π. The properties of such sinusoids can be used to identify a logic value without ambiguity, and hence a realizable system of logic is created. The chapter further presents a family of logic gates that can operate using such sinusoidal signals. In addition, due to orthogonality of sinusoid signals with different frequencies, multiple sinusoids could be transmitted on a single wire simultaneously, thereby naturally allowing the approach to implement multilevel logic. One advantage of such a logic family is its immunity from external additive noise and an improvement in switching (dynamic) power.

The last chapter focuses on asynchronous circuit design issues. In Systems-on-a-Chip (SOCs) and Networks-on-a-Chip (NoCs), using globally asynchronous locally synchronous (GALS) system design for pausible clocking is widely popular. This chapter investigates throughput reduction and synchronization failures introduced by existing GALS pausible clocking schemes and proposes an optimized scheme for more reliable GALS system design with higher performance. The approach minimizes the acknowledge latency and maximizes the safe timing region for inserting the clock tree.

References

1. Brayton, R.K., Hachtel, G.D., Sangiovanni-Vincentelli, A.L.: Multilevel logic synthesis. In: Proceedings of IEEE, 78(2):264–270 (1990)
2. Hassoun, S. (ed.): Logic Synthesis and Verification. San Jose, CA, USA (2001)
3. Sinha, S., Brayton, R.K.: Implementation and use of SPFDs in optimizing Boolean networks. In: Proceedings of International Conference on Computer-Aided Design, pp. 103–110. Paris, France (1998)
4. Wurth, B., Wehn, N.: Efficient calculation of Boolean relations for multi-level logic optimization. In: Proceedings of European Design and Test Conference, pp. 630–634. (1994)