Logic Synthesis Based on the Structure of an Ordered DD*

M. A. Thornton, D. M. Wessels
Department of Computer Science and Computer Engineering, University of Arkansas
313 Engineering Hall, Fayetteville, Arkansas, USA 72701-1201

Telephone: 501-575-5159 - Fax: 501-575-5339 - email: mitch@engr.uark.edu

Introduction: Digital logic synthesis is an integral step in modern integrated circuit design. Coupled with the popular usage of synthesis techniques is the use of BDDs, which are data structures that represent Boolean functions [1]. These directed acyclic graphs (DAGs) have generated great interest due to their ability to represent many Boolean functions in a very compact manner and the existence of associated algorithms for their manipulation.

Although BDDs have traditionally been viewed as purely behavioral entities, it is well known that they can represent "trees of multiplexers" due to the correspondence between their structure and the Shannon decomposition. With this point of view, a minimized BDD can represent a correspondingly small structural representation of a circuit in the form of a network of multiplexers.

The motivation for developing a BDD based synthesis tool based on the structure of the DAG, is given by the fact that many good variable ordering heuristics have been developed recently. By using BDD minimizing methods as the underlying mechanism for area optimization, this synthesis optimization is effectively translated into the BDD variable reordering problem. Such mappings have other desirable characteristics as well, such as the ability to produce high-speed, dual-rail configurations and to generate low-power PTL based circuits [2][3].

BDD Mapping Technique: The technique described here operates on BDD edges rather than vertices as in the well-known "tree-of-multiplexers" approach and can provide good results with respect to synthesis optimizations such as area minimization. This technique exploits the fact that, for a given satisfying variable assignment, the beginning of a directed edge implies that the edge is part of a specified path taken in a traversal of the graph if the variable denoted by the connected vertex is at the appropriate Boolean value. Therefore, the beginning of a directed edge may be mapped to a simple pass transistor or an AND gate. Likewise, the endpoint of the edge indicates that the successor variable is to be tested for the activated edge. Since a successor variable vertex may be connected to many predecessor edges, and that variable is tested if any of the edges are activated, all endpoints may be logically ORed together. These observations form the basis of the edge mapping technique that is stated algorithmically below:

* This work was supported by the National Science Foundation under grant MIP-9633085

1) For each edge insert a 2-input AND gate driven by the corresponding node literal with polarity determined by the value of the BDD edge activation.

2) For each non-rooted vertex, insert an *m*-input OR gate driven by the outputs of the preceding AND gates representing edges that point to the current node.

3) Simplify by removing all gates with a single input and map to the desired type of gates or PTL style.

We note that this method results in a circuit with output lines that structurally correspond to the terminal vertices of the BDD. This is in contrast to the technique that maps BDDs into multiplexer trees which results in a circuit with output lines corresponding to BDD initial vertices. The technique described here maps the BDD (various variants of the data structure such as OBDD, SBDD and FBDD are applicable) into an intermediate multilevel AND/OR structure initially. Since AND/OR structures may not be desirable, the technique is easily extended to map into an equivalent NAND/NAND or NOR/NOR or PTL type network through subsequent transformations. Figure 1 illustrates the edge mapping technique and shows an OBDD with the corresponding intermediate AND/OR circuit. Figure 2 shows the resulting AND/OR, NAND/NAND and NOR/NOR simplified circuits.

The edge mapping method ensures that each net in the resulting circuit is at a logic-1 only when the corresponding edge in the BDD is traversed for a given input assignment. This leads to the desirable property of allowing output probability factors to be assigned to each net in the intermediate AND/OR circuit since statistical independence is preserved. The statistical independence property arises from the fact that at no point can signals dependent on the same-polarity input variable converge within the mapped circuit.

In addition to the ability to produce gate level netlists, the technique also provides for the ability to produce PTL type netlists exclusively or a hybrid combination of Pass-Transistor Logic (PTL) and static gate type circuits. Due to the dual-rail nature of the circuit produced by the edge mapping method, other styles of PTL such as complementary PTL (CTL), double PTL (DPL), energy economized PTL (EEPL), swing restored PTL (SRPL) and push-pull PTL (PPL) [5] are possible in the resulting circuit.

Circuit Optimizations: As in the development of all synthesis systems, the incorporation of target netlist optimizations is of paramount importance. This approach for the generation of a synthesized circuit has some unique characteristics that allow for optimizations with respect to power dissipation, area and delay.

Power: Inherent in gate-level power constrained synthesis is the need to estimate power dissipation prior to the generation of the circuit to be implemented. In terms of CMOS technology, it is well known that power dissipation is directly proportional to the sum of all internal net switching probabilities denoted as p_{swtch}. In the edge mapping approach described above, we note that the output probability, p_{out}, is easily computed through a simple traversal of

the BDD or the intermediate AND/OR netlist that is generated [4]. A rough estimate of power dissipation may accomplished by computing p_{swtch} in terms of p_{out} as $p_{swtch} = 2[p_{out} - p^2_{out}]$. Although this approximation assumes that each input is statistically independent and equally likely to be a "0" or "1" and hence ignores the much more difficult problem of estimating dependencies due to temporal and spatial correlations.

The other method that may be employed for power optimization is to target PTL based netlists which are well-known for low power dissipation. However, problems with internal signal degradation must be considered. This is typically accomplished through the judicious insertion of buffers throughout the resulting netlist.

Area: Minimization of the BDD size directly results in a reduction of transistors in the netlist. This observation effectively translates the area optimization problem into the BDD variable reordering one. It is also noted that the BDD edge mapping technique described above results in a dual-rail circuit. In terms of a gate-level implementation (and some PTL styles), this means that the "cone" utilizing the smallest number of components from either the inverted or non-inverted output may be used alone resulting in a single-rail circuit requiring fewer transistors.

Delay: In terms of delay minimization, we exploit the fact that the BDD representing the function to be synthesized structurally represents the target netlist. Thus, by annotating the BDD directly with "slack parameters" equivalent to those used by other synthesis tools, additional information is available for guiding variable reordering of the BDD before mapping to the target netlist.

Multi-Valued Logic: The technique described above may be easily generalized to handle the case of multi-valued logic. While binary decision diagrams have been the primary focus of decision diagram research, there has also been interest in multi-valued decision diagrams [6][7].

In producing a circuit composed of multi-valued logic gates, it is assumed the following basic gate types are available:

- MIN gates - the gate output is minimum of its input vales
- MAX gates - the gate output is maximum of its input values

It is also assumed that the characteristic functions are available for each of the primary inputs, i.e. the set of $J_j(x_i)$ values such that $J_j(x_i)=k-1$ if $x_i=j$, and $J_j(x_i)=0$ otherwise.

The edges of an ordered multi-valued decision diagram are mapped to small sets of logic gates, producing a k-output circuit. If the function being computed is $f(X)$, then the k outputs of the resulting circuit correspond to the characteristic functions of f, i.e. $J_0(f(X)), \ldots J_{k-1}(f(X))$. The circuit outputs thus form a *1-of-k* code, where the i^{th} output of the circuit is logically true if and only if the decision diagram would evaluate to logic value i.

The resulting circuit provides output in a *1-of-k* form for a diagram operating on k-valued logic, and the circuit size is linear in the size of the original decision diagram, allowing the mapping technique to take full advantage of any advances in the area of decision diagram

minimization. We then consider implementing the multi-valued decision diagram as a binary logic circuit.

Conclusion: A technique has been presented for mapping binary and multi-valued decision diagrams to combinational digital logic, composed of either multiple-valued logic gates (using MIN, MAX), or binary logic (using AND/OR gates) has been presented. The approach differs from the well-known technique of mapping graph vertices into multiplexers by concentrating on mapping graph edges instead. The resulting circuit may be realized as either a dual-rail or single-rail circuit and the target circuit can be in the form of a logic gate network, a PTL style circuit, or a hybrid combination of the two.

References

[1] Bryant, R. E., Graph-Based Algorithms for Boolean Function Manipulation, *IEEE Transactions on Computers*, vol. c-35, no. 8, pp. 677-691, August 1986.

[2] Yano, K., Sasaki, Y., Rikino, K. and Seki, K., Top-Down Pass-Transistor Logic Design, *IEEE Journal of Solid-State Circuits*, vol. 31, no. 6, June 1996, pp. 792-803.

[3] Bertacco, V., Minato, S., Verplaetse, P., Benini, L. and De Michelli, G., Decision Diagrams and Pass Transistor Logic Synthesis, Technical Report: CSL-TR-97-748, Stanford University, December 1997.

[4] Thornton, M. and Nair, V., Efficient Calculation of Spectral Coefficients and their Application to Combinational Logic Synthesis, *IEEE Trans. on CAD/ICAS*, vol. 14, no. 11, November 1995, pp.1328-1341.

[5] Zimmerman, R. and Fichtner, W., Low Power Logic Styles: CMOS Versus Pass-Transistor Logic, *IEEE Journal of Solid-State Circuits*, vol. 32, no. 7, pp. 1079-1090, July 1997.

[6] Miller, M. and Drechsler, R., Implementing a Multiple-Valued Decision Diagram Package, Proceedings of the International Symposium on Multiple-Valued Logic, 1998.

[7] Sasao, T. and Butler, J., A Method to Represent Multiple Output Switching Functions by Using Multiple-Valued Decision Diagrams, Proceedings of the International Symposium on Multiple-Valued Logic, 1996.

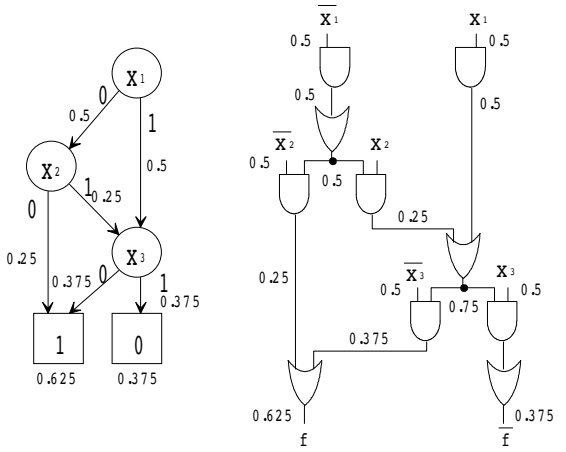

Figure 1: OBDD Mapped into Intermediate AND/OR Structure

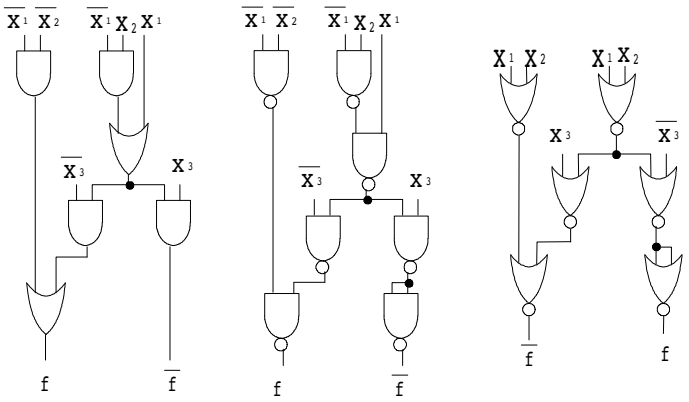

Figure 2: Simplified AND/OR, NAND/NAND and NOR/NOR Representations