Behavioral Synthesis of Combinational Logic Using Spectral-Based Heuristics

M. A. THORNTON
Mississippi State University
and
V. S. S. NAIR
Southern Methodist University

A prototype system developed to convert a behavioral representation of a Boolean function in OBDD form into an initial structural representation is described and experimental results are given. The system produces a multilevel circuit using heuristic rules based on properties of a subset of spectral coefficients. Since the behavioral description is in OBDD form, efficient methods are used to quickly compute the small subset of spectral coefficients needed for the application of the heuristics. The heuristics guide subsequent decompositions of the OBDD, resulting in an iterative construction of the structural form. At each stage of the translation, the form of the decomposition is chosen in order to achieve optimization goals.

Categories and Subject Descriptors: B.6.3 [**Logic Design**]: Design Aids

General Terms: Design, Optimization

Additional Key Words and Phrases: Automatic Synthesis, decision diagrams, design aids, logic design, spectral methods

1. INTRODUCTION

A synthesis technique using a subset of spectral coefficients is described in this paper. Previous results contain the development of methodologies to compute spectral coefficients using decision diagrams [Clarke et al. 1993; Thornton and Nair 1995], however most spectral-based methodologies require the entire spectrum to be computed [Edwards 1977; Hurst et al. 1985]. Although the computational method presented in Thornton and Nair [1995] reduces the complexity from exponential to polynomial in terms of

Authors' addresses: M. A. Thornton, Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39762; email: m.thorton@computer.org; V. S. S. Nair, Department of Computer Science and Engineering, Southern Methodist University, Dallas, TX 75275; email: nair@seas.smu.edu.

the number of primary inputs for most functions, the entire spectrum of a function still contains an exponential number of coefficients. This fact provides the motivation for developing a method that uses a subset of spectral coefficients to perform logic synthesis. By using a subset of spectral coefficients, each value may be calculated quickly and the overall number of computations is no longer exponential.

It has been shown that all 2^n spectral coefficients are required to uniquely represent a Boolean function when the Walsh family of transformation matrices are employed [Karpovsky 1976; Davio et al. 1978]. Thus, a method that uses a subset of coefficients must necessarily employ heuristics, since an exact solution cannot be obtained. The use of heuristics in the synthesis of logic functions is very common and has led to some of the most successful tools available today [Brayton et al. 1984; Brayton et al. 1987; Muroga et al. 1989].

The method presented in this paper is developed to produce multilevel circuits that may be optimized for area, device, and interconnection minimization. The optimization for timing versus area presents a well-known tradeoff. In order to ensure minimal delay, a two-level circuit composed of a maximally reduced set of implicants is the best that can be achieved in terms of critical path length. Alternatively, minimization of area and interconnections generally require a multilevel circuit so that intermediate term sharing between a set of reduced implicants may be exploited.

The approach used in this method is to generate a multilevel circuit while also providing shorter paths for more critical inputs (those that become stable later than other inputs). This is reasonable since many real world design problems are specified by considering some valid input signals to be present at the inputs of the circuit before others. With this scheme, area optimization is the dominating synthesis parameter; however, some degree of timing constraint can be included by ordering the inputs according to their times of arrival. Finally, circuit testability can be enhanced in this synthesis methodology by producing circuits that minimize internal fanout.

This method is designed to be the intermediate step between a behavioral description of a logic function and an initial circuit structure to use as input to other timing or area optimizers. Many of the current popular minimizers require an initial form of the circuit as input [Brayton et al. 1987; Muroga et al. 1989]. The final synthesized output can be adversely affected if a highly inefficient input circuit was provided. Furthermore, the optimizations are usually performed by local changes over portions of the circuit, this approach provides an initial circuit structure that was formulated using the global information present in digital logic spectral coefficients.

The remainder of this paper is organized as follows: First, the synthesis method is presented and an examination of the optimization criteria is provided. Next, the development of the method is described in detail. The formulation of the spectral heuristics is explained and the use of the

decomposition methods at each stage of synthesis is discussed. Following the discussion of the philosophy behind the technique, implementation issues are discussed, including the program flow. Finally, some examples of this method are presented.

2. DESCRIPTION OF THE SYNTHESIS METHODOLOGY

This synthesis technique produces a circuit by determining an output gate first and working back toward the inputs. The output gate is chosen by using the information contained in the subset of spectral coefficients, commonly referred to as the Chow parameters [Chow 1961]. Based upon the properties of the Chow parameters, a set of heuristic rules are applied to choose the appropriate gate. The heuristics are formulated such that the chosen gate will be maximally correlated to the entire function, and hence the remaining portion of the function is simplified. In order to take advantage of the efficient method for computing the spectral coefficients, the intermediate functions, as well as the initial input function, are represented in terms of OBDDs.

At each stage of the synthesis, once the output gate is chosen, at least one primary input is removed from each remaining intermediate function. Therefore, the range of the intermediate function always decreases by at least one half. The particular input that is chosen for removal is determined by the optimization criteria. If timing optimizations are desired, the slower arriving inputs are removed first, resulting in fewer gates in their propagation path. If area and interconnection minimizations are required, the input is chosen using the principle of maximal subfunction independence, resulting in the intermediate functions being as simple as possible. Since at least one primary input is discarded at each step in the processing flow of the synthesis technique, convergence is guaranteed.

The reason the Chow parameters were chosen, rather than some other higher ordered spectral coefficient, is that we needed a correlation measure relative to a specific primary input. A higher ordered coefficient would give a correlation measure relative to a specific function of a subset of the primary inputs. In fact, there are cases when higher ordered coefficients are useful as secondary measures such as those used in Thornton and Nair [1995] to detect the dependence of a structural circuit on various parity functions.

2.1 Optimization Criteria

In the past, area minimization was generally measured as the number of implicants in a minimized cover of a function and the minimization of gates and interconnections was measured as the total number of literals in a minimized cover [Brayton et al. 1984]. The most common way of incorporating this type of optimization is by using the concept of “don’t cares” [Brayton et al. 1987; Rudell and Sangiovanni-Vincintelli 1985]. Another popular method, exploited by many researchers, is the use of “permissible functions” [Muroga et al. 1989].

The minimization criteria used in this implementation is the creation of intermediate functions that are as degenerate as possible. Since a single primary input is guaranteed to be removed at each stage of the synthesis, the resulting intermediate functions will contain at least one less primary input. However, if the intermediate functions also become degenerate, additional inputs may be discarded, resulting in significantly simpler functions remaining to be realized. Further, determination of which, if any, of the inputs are redundant is implicitly achieved, since the intermediate functions are represented by OBDDs formed by applying the *RESTRICT* operation on the original OBDD [Bryant 1992]. This occurs because an OBDD is defined as a reduced BDD with a specific variable ordering [Bryant 1992]. Thus, a redundant input cannot be contained in an OBDD. The test for maximum subfunction degeneracy validates the use of the heuristics for choosing the primary input to decompose about, since it is guaranteed to produce subfunctions that depend on fewer primary inputs rather than a random selection of a primary input.

In addition to the exploitation of intermediate function degeneracy, interconnection optimization is achieved through the structure that a circuit synthesized by this technique must have. Since each stage allows a single primary input to be discarded from the next synthesis step, a characteristic overall circuit structure results. In most cases, primary inputs are discarded through the use of the Shannon decomposition [Shannon 1938], Various forms of this decomposition formula imply the structure of each intermediate portion of the resulting circuit. Figure 1 depicts three possible forms for a single iteration of this synthesis technique. By choosing forms that incorporate a fanout of two, or even restricting those forms to no fanout, interconnections can be minimized, and are also made local to the current area of the circuit being synthesized. This method prevents gates near the input side of the circuit from directly driving gates near the output end of the circuit. Thus, gates that are not close together are decoupled within the resulting circuit, minimizing interconnection complexities.

Although the prototype implementation of this method concentrated on the incorporation of area minimization, a timing optimization criteria could be incorporated by exploiting the order in which primary inputs are discarded from each intermediate function. When the designer supplies the OBDD of the circuit to be synthesized, timing information must also be specified. Specifically, the inputs must be grouped into classes that are ranked by the speed at which they will appear at the inputs to the resulting circuit. Each class may contain a single input, implying a strict timing order of arrival or, at the other extreme, a single class may be specified, inferring that all signals will be present at the same time. In effect, the specification of these classes dictates the delay versus area tradeoffs in the final result. If a strict ordering is supplied, the synthesizer is forced to discard the primary inputs from the intermediate functions in a specific order. Therefore, area minimization is achieved only through the choosing of the particular output gate at each stage. It should be noted that the

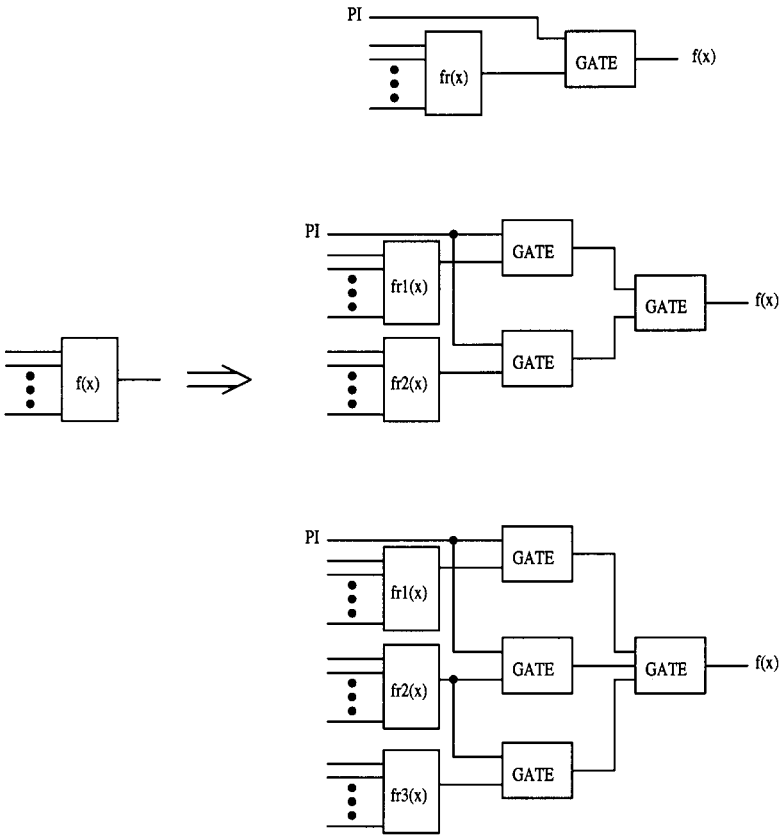

Fig. 1. Diagram of a single iteration of the heuristic synthesis method.

heuristics used to determine the output gates were developed with area minimization in mind, so that a strict ordering does not necessarily cause the resulting circuit to be overly large, but just removes a degree of freedom by not allowing the synthesizer to choose the most prudent input to discard. Alternatively, if all inputs are in the same class, implying that all will arrive at the input of the circuit simultaneously, the synthesizer is allowed to choose the input to discard that will most likely result in an intermediate function with a high degree of redundancy, as well as to choose which type of output gate to use. The practical way to specify the timing classes is to order only those inputs that are critical in terms of arrival time and to place all others in the same class. This will not only allow the synthesizer to enforce the timing criteria but also give it maximum freedom for minimizing the resultant area.

The third optimization criteria is that of testability. It is well known that completely fanout-free (CFOF) circuits are highly testable, since they only require a number of test vectors to detect any single stuck-at fault in the circuit equal to the number required to test for a single stuck-at fault at the primary inputs [Pradham 1986]. By choosing the decompositions at each

stage of the synthesis to be such that no fanout is generated, the resulting circuit will have no internal fanout and be highly testable.

2.2 Spectral Heuristics for Decomposition

This synthesis methodology employs a set of heuristics based upon properties of the Chow parameters. The set of heuristics is used to choose the output gate at each level of synthesis. The basis for the heuristics for the choice of the output gate is the examination of the Chow parameters for all possible Boolean functions of two variables. It is essential that the correct gate be chosen when only two primary inputs remain to ensure that the synthesis algorithm terminates and does not oscillate when this terminal condition occurs. The algorithm is thus guaranteed to converge, since a primary input is discarded at each intermediate stage, and at the terminal stage when only two primary inputs remain, a gate is guaranteed to be chosen that will result in termination of the algorithm. The following section describes the details on the development of these heuristics and lists them in a table.

2.3 Maximum Subfunction Independence

The other main idea in the implementation is the maximal redundancy test used to determine which primary input to discard. Since the function to be realized is in OBDD form, it is very efficient to apply the OBDD *RESTRICT* operation for an input variable. The *RESTRICT* operation returns a OBDD with a logic 1 or 0 substituted for all instances of a specified input variable. When the *RESTRICT* operation is applied, the returned OBDD will always depend on one less variable and, in many cases, several other inputs will also become redundant. The maximal redundancy algorithm computes the OBDDs for the restriction of each input and chooses the input that results in the most redundancy.

3. DEVELOPMENT OF THE TECHNIQUE

The input to the synthesis program is an OBDD representing the circuit to be synthesized. A queue is maintained that points to each intermediate OBDD to be synthesized. Initially, the OBDD of the entire function is placed in the queue. At each stage of the synthesis, an OBDD is popped from the queue. If the OBDD depends on two or more inputs, the Chow parameters are computed. Based upon the Chow parameter heuristics, an output gate is chosen. Once the output gate is chosen, the primary input to discard from the remainder functions must be obtained. If there is a timing optimization, the primary input corresponding to the largest arrival time is chosen. Otherwise, the maximal redundancy test is applied to choose the appropriate primary input to be discarded.

3.1 Formulation of the Heuristics

The heuristics were derived by observing the Chow parameters for all 16 possible Boolean functions of two-variables and by exploiting the properties

Table I. Heuristics and Rules for Synthesis Methodology

Main Heuristic	Secondary Heuristic	Function Choice
$\|S(0)\| = 2^n$	$S(0) < 0$	$f = 1$
	$S(0) > 0$	$f = 0$
$\|S(x_i)\| = 2^n$	$S(x_i) < 0$	$f = \bar{x}_i$
	$S(x_i) > 0$	$f = x_i$
$\|S(x_i)\| = 2^n - \|S(0)\|$	$S(0) < 0$ and $S(x_i) < 0$	$f = \bar{x}_i + f_1$
	$S(0) < 0$ and $S(x_i) > 0$	$f = x_i + f_0$
	$S(0) > 0$ and $S(x_i) < 0$	$f = \bar{x}_i \cdot f_0$
	$S(0) > 0$ and $S(x_i) > 0$	$f = x_i \cdot f_1$
$S(0) > 0$	$\sigma < 0$	NOR
	$\sigma > 0$	AND
	$\sigma = 0$ and $S(x_1) \geq 0$	AND
	$\sigma = 0$ and $S(x_1) < 0$	NOR
$S(0) < 0$	$\sigma < 0$	NAND
	$\sigma > 0$	OR
	$\sigma = 0$ and $S(x_1) \geq 0$	OR
	$\sigma = 0$ and $S(x_1) < 0$	NAND
$S(0) = 0$	$S(x_i) = 0 \forall i$	XOR
	$\sigma > 0$	AND
	$\sigma < 0$	OR
	$\sigma = 0$ and $S(x_1) \geq 0$	XNOR
	$\sigma = 0$ and $S(x_1) < 0$	XOR

of spectral coefficients. The set of rules are organized in a hierarchical manner, so that the rules providing the simplest residual OBDDs are chosen first. Table I contains the list of heuristics and rules used in the synthesis tool. The value σ is defined as the sum of the first-order spectral coefficients as given in Eq. (1) and f_{0i}, f_{1i} represent the Shannon cofactors, $f_0 := f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)$ and $f_1 := f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$.

$$\sigma = \sum_{i=1}^{n} S(x_i) \tag{1}$$

To illustrate how the heuristics were chosen, consider the Boolean functions and their associated Chow parameters given in Table II. There are 16 entries in Table II corresponding to all possible functions of two variables.

As an example, consider the function, $x_1 + x_2$. The zero order spectral coefficient, $S(0)$, is less than 0 and the sum of the first-order coefficients, σ, is greater than 0. Therefore, whenever $S(0) < 0$ and $\sigma > 0$, the OR gate is chosen as the dominant function.

Table II. Chow Parameters for all Boolean Functions of Two Variables

Function	Chow Parameters			σ	Gate
	$S(0)$	$S(x_1)$	$S(x_2)$		
0	4	0	0	4	constant 0
x_1x_2	2	2	2	6	AND
$x_1\bar{x}_2$	2	2	-2	2	NOR
x_1	0	4	0	4	literal x_1
$\bar{x}_1x_2$	2	-2	2	2	NOR
x_2	0	0	4	4	literal x_2
$x_1 \oplus x_2$	0	0	0	0	XOR
$x_1 + x_2$	-2	2	2	2	OR
$\overline{x_1 + x_2}$	2	-2	-2	-2	NOR
$\overline{x_1 \oplus x_2}$	0	0	0	0	XNOR
$\bar{x}_2$	0	0	-4	-4	literal $\bar{x}_2$
$x_1 + \bar{x}_2$	-2	2	-2	-2	OR
$\bar{x}_1$	0	-4	0	-4	literal $\bar{x}_1$
$\bar{x}_1 + x_2$	-2	-2	2	-2	OR
$\overline{x_1x_2}$	-2	-2	-2	-6	NAND
1	-4	0	0	-4	constant 1

3.2 Shannon Decomposition Forms

Once the dominant output gate is chosen, the function must be decomposed into two residual functions. The decomposition method used in this synthesis tool is based on variations of the Shannon decomposition formula when the chosen output gate is not the XOR or XNOR. The Shannon form was chosen, since at least one primary input is guaranteed to be discarded. In order to accommodate the various output gate forms, the Shannon decomposition Boolean formula was rearranged into several forms.

When the output gate is chosen as the XOR or XNOR, a spectral-based decomposition is attempted first. Usually, the spectral-based decomposition is very effective in terms of partitioning the primary inputs; however, in those cases where inferior decompositions are achieved, rules based on various Shannon decompositions are used. When the Shannon forms are used, the form that contains residual OBDDs with the smallest number of dependent variables is chosen.

The idea behind the spectral-based decomposition method is to determine two partitioned subfunctions such that one depends only on highly correlated primary inputs while the other depends upon inputs with a small correlation. When the dominant gate is chosen to be the XOR, the function, f, is partitioned in the form as shown in Eq. (2). The subfunction, g, is formed by evaluating f with all highly correlated inputs set to a logic 0 as given in Eq. (3).

$$f = g \oplus h \tag{2}$$

$$g = f(0,0,0, \ldots, x_{n-i}, x_{n-i-1}, \ldots, x_n) \tag{3}$$

The choice of the highly correlated inputs is made by choosing those inputs whose corresponding spectral coefficients have a magnitude greater than $|2^{n-1}|$. Stated mathematically, the criteria for choosing the highly correlated inputs is given in expression 4. Once g is computed, the corresponding h subfunction is obtained directly by evaluating Eq. (5).

$$|S_f[x_i]| > 2^{n-1} \rightarrow use\ 0\ for\ x_i \tag{4}$$

$$h = f \oplus g \tag{5}$$

4. IMPLEMENTATION

This synthesis technique was implemented using the *C* programming language. The basic data structure used in the implementation is a first-in first-out (FIFO) queue that contains pointers to the intermediate OBDDs. Initially the queue is initialized to point to the OBDD representing the function to be realized. At each stage of the synthesis, the OBDD pointed to at the top of the FIFO is operated upon and an output gate is chosen for implementation. If any remainder OBDDs are created, pointers corresponding to them are inserted at the tail of the FIFO. The synthesis is complete when the FIFO becomes empty indicating the entire circuit has been built.

4.1 Examples and Results

Several of the *ISCAS*85 and *IWLS* benchmark circuits were synthesized using this technique. In order to determine the relative effectiveness of this technique, the *IWLS* benchmarks were synthesized using the *mis*II tool from Berkeley and mapped to a small cell library identical to one used for the heuristic method.

Two different *mis*II scripts were used to synthesize these benchmarks. The first script consisted of an *ESPRESSO* minimization followed by the *mis*II *simplify* and *sweep* commands. The second *mis*II script omitted the *ESPRESSO* simplification and consisted of the *collapse*, *simplify*, and *sweep* commands. Table III contains a summary of these results. The column labeled '*mis*II/ *ESPRESSO* Size' contains the number of logic gates *mis*II, required when the first script was used as input. The fifth column labeled '*mis*II Size' contains the number of gates, the resulting circuit required when the *ESPRESSO* minimization was not used. Finally, the column labeled 'Heuristic Size' contains the number of logic gates in the resulting circuit using the synthesis method just described.

This comparison was chosen because the prototype synthesis tool described here is currently implemented to operate on single output functions only. Future plans include the incorporation of other decision diagram types such as shared BDDs [Minato et al. 1990] to handle multioutput circuits. By selecting outputs from the benchmarks in Table III that result in nontrivial single-output functions and synthesizing with both our method and *mis*II, a preliminary evaluation of the effectiveness of our approach was achieved.

Table III. Comparison of Spectra-Based Heuristic Method with *mis*II

Circuit Name	Output Name	Number Inputs	*mis*II Size	*mis*II/ *ESPRESSO* Size	Heuristic Method Size
5*xp*1	*output* 1	7	14	12	25
9*sym*	*output* 1	9	262	103	194
*alu*4	*output* 2	14	21	26	13
bw	*output* 1	5	65	41	14
*con*1	*output* 1	6	8	6	11
*duke*2	*output* 4	22	263	122	11
*misex*3	*output* 1	14	96	373	16
*rd*53	*output* 1	5	18	7	9
*rd*73	*output* 1	7	114	116	125
*rd*84	*output* 1	8	98	213	253
*sao*2	*output* 4	10	43	73	11
*vg*2	*output* 2	25	35	82	46
*apex*1	*output* 12	45	96	74	38
*misex*3*c*	*output* 1	14	34	28	34
*xor*5	*output* 1	5	4	54	4
clip	*output* 4	9	81	97	170

These results are mixed, in that our method does significantly better and significantly worse in several cases. For those circuits whose Shannon cofactors about some input variable become very small (depend on as few as possible inputs), our method does quite well. This is because the principle of maximum redundancy is incorporated into our method. Also, circuits with XOR operations tend to do well in this method, since techniques for detecting XOR [Thornton and Nair 1995] are incorporated into the tool.

Table IV contains a summary of the results obtained when significantly larger circuits are synthesized. The first column contains the name of the benchmark circuit. The remaining columns contain the name of the output, the number of primary circuit inputs, and, the resulting circuit size in terms of logic gates using the same cell library as in the previous results.

The experimental results are mixed, some of the results are good while others indicate poor performance. The good results are attributed to the high degree of redundancy present in the initial netlist. Likewise, some results are very bad, such as the 11,000 gates required to implement output 329*gat* of benchmark circuit *c*432. After careful study of the log files created during the synthesis runs, several characteristics are noted. Circuits that have Shannon cofactors with a lot of redundancy, that is, cofactors that depend on significantly fewer primary inputs tend to minimize well with this method. Also, circuits that have a lot of XOR dependence tend to do well, since this method (and spectral methods in general) are good at detecting and exploiting this dependence.

In cases where the synthesis performed poorly, the results turn out to be trees of Shannon decompositions, although they are present in a mixture of forms, with subsequent variable dependencies of size $n - 1$ where the previous stage depends on n variables. This is the worst-case scenario,

Table IV. Experimental Results of the Spectral-Based Heuristic Logic Synthesizer

Circuit	Output	Inputs	Size
$c432$	$329gat$	27	11009
$c880$	$879gat$	44	174
$c880$	$863gat$	36	113
$c880$	$768gat$	10	17
$c2670$	401	11	20
$c3540$	399	26	60
$c3540$	364	25	29
$c5315$	1002	9	15
$c5315$	871	24	81
$c5315$	661	24	49

since all functions can be realized as trees of Shannon decompositions, but such realizations require large gate counts. This result is not overly surprising, since this method was designed to exploit redundancy through the principle of maximum subfunction independence, as described in a previous section. Due to the good performance on some circuits and the very poor performance on others, this method can be used as an optimization step in a larger synthesis tool made up of other optimization routines, or it can be used to initially transform a circuit in OBDD form into a netlist suitable for input to some other synthesis tool.

In terms of timing and memory requirements, all of the above data was generated using a 100 MHz Sun SPARCstation 20 with 160 MB of internal RAM. Although the total memory usage for each example was not explicitly measured, none of the synthesis runs above resulted in page swapping and the wall clock times averaged from a few seconds to several minutes. For those examples with extremely large gate counts, the run times are much larger. As an example, the extreme case of $c432$, output $329gat$ required 10 hours to complete.

5. CONCLUSIONS

A method for transforming a behavioral description to a structural one, without an intermediate flattening process, has been developed. Furthermore, for some classes of circuits, this transformation has very good area minimization properties. By using an initial representation in OBDD form coupled with the use of a small subset of spectral coefficients, the computational hurdles of past spectral synthesis techniques are overcome. Furthermore, the heuristics are formed such that a moderate degree of optimization is incorporated during the behavioral synthesis phase described in this paper. The result is a structural circuit that is a good initial representation for input to other structural circuit optimizers.

The area optimization results are mixed. Some benchmark circuits compare favorably with a standard synthesis tool while others require a significantly larger gate count. The circuits that yield small gate counts with this technique generally have a high degree of maximum subfunction independence as described here.

This work will be expanded by developing further heuristics and incorporating them into the synthesis tool. Multioutput circuits can be handled using this method through the incorporation of shared BDDs rather than the OBDDs discussed here. In addition to incorporating more heuristics, other decomposition methods will be investigated for inclusion into the tool. Some of the poorer experimental results are due to reliance on the Shannon-based decompositions, thus other forms of decompositions will likely improve these results.

REFERENCES

BRAYTON, R. K., HACHTEL, G. D., MCMULLEN, C. T., AND SANGIOVANNI-VINCENTELLI, A. 1984. *Logic Minimization Algorithms for VLSI Synthesis*. Kluwer Academic Publishers, Hingham, MA.

BRAYTON, R. K., RUDELL, R., SANGIOVANNI-VINCINTELLI, A. L., AND WANG, A. R 1987. Mis: A multiple-level logic optimization system. *IEEE Trans. CAD CAD-6*, 6, 1062–1081.

BRYANT, R. E. 1992. Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Comput. Surv. 24*, 3 (Sept. 1992), 293–318.

CHOW, C. K. 1961. On the characterization of threshold functions. In *IEEE Special Publication*. 34–38.

CLARKE, E. M., MCMILLAN, K. L., ZHAO, X, FUJITA, M., AND YANG, J. 1993. Spectral transforms for large boolean functions with applications to technology mapping. In *Proceedings of the 30th International Conference on Design Automation* (DAC'93, Dallas, TX, June 14–18), A. E. Dunlop, Ed. ACM Press, New York, NY, 54–60.

DAVIO, M., DESCHAMPS, J.-P., AND THAYSE, A. 1978. *Discrete and Switching Functions*. McGraw-Hill, London, UK.

EDWARDS, C. R. 1977. The design of easily tested circuits using mapping and spectral techniques. *Radio Electrical Eng. 47*, 7, 321–342.

HURST, S., MILLER, D., AND MUZIO, J. 1985. *Spectral Techniques in Digital Logic*. Academic Press, Inc., Orlando, FL.

KARPOVSKY, M. G. 1976. *Finite Orthogonal Series in the Design of Digital Devices*. John Wiley & Sons, Inc., New York, NY.

MINATO, S.-I., ISHIURA, N., AND YAJIMA, S. 1990. Shared binary decision diagram with attributed edges for efficient Boolean function manipulation. In *Proceedings of the ACM/IEEE Conference on Design Automation* (DAC '90, Orlando, FL, June 24-28), R. C. Smith, Ed. ACM Press, New York, NY, 52–57.

MUROGA, S., KAMBAYASHI, Y., LAI, H. C., AND CULLINEY, J. N. 1989. The transduction method—design of logic networks based on permissible functions. *IEEE Trans. Comput. 38*, 10 (Oct. 1989), 1404–1424.

PRADHAN, D. K., Ed. 1986. *Fault-Tolerant Computing: Theory and Techniques; vol. 1*. Prentice-Hall, Inc., Upper Saddle River, NJ.

RUDELL, R. AND SANGIOVANNI-VINCINTELLI, A. L. 1985. Espresso-mv: Algorithms for multiple-valued logic minimization. In *Proceedings of the IEEE Conference on Custom Integrated Circuits*. IEEE Computer Society, New York, NY, 230–234.

SHANNON, C. E. 1938. Symbolic analysis of relay and switching circuits. *Trans. AIEE 57*, 713–723.

THORNTON, M. A. AND NAIR, V. S. S. 1995. Parity function detection and realization using a small set of spectral coefficients. In *Proceedings of the ACM/IEEE International Workshop on Logic Synthesis* (May 1995). ACM Press, New York, NY.

THORNTON, M. A. AND NAIR, V. S. S. 1995. Efficient calculation of spectral coefficients and their applications. *IEEE Trans. CAD/ICAS 14*, 11 (Nov.), 1328–1341.

Received: June 1996; revised: November 1996; accepted: May 1998