Multi-output Timed Shannon Circuits*

Mitchell A. Thornton
Mississippi State University
Mississippi State, Mississippi
mitch@ece.msstate.edu

Rolf Drechsler
University of Bremen
Bremen, Germany
drechsle@informatik.uni-bremen.de

D. Michael Miller
University of Victoria
Victoria, BC, Canada
mmiller@csr.uvic.ca

Abstract

Timed Shannon circuits have been proposed as a synthesis approach for a low power optimization technique at the logic level since overall circuit switching probabilities may be reduced. An improvement in the application of this principle for multi-output circuits is presented. Techniques that trade area for power reduction and a method for minimizing the overall circuit switching probability are also included. Experimental results are given and analyzed for these techniques.

1 Introduction

The introduction of timed Shannon circuits [5] was motivated by the need to automatically synthesize circuits with low power dissipation characteristics. In the original work on timed Shannon circuits, problems involving the synthesis of multi-output Boolean functions were noted and three methods for handling this situation were proposed; the use of *Multi-valued Terminal Binary Decision Diagrams* (MTBDDs) [3], time multiplexing of individual function outputs, and, the use of "disambiguation circuitry" necessitating the detection of "shared" BDD graph portions.

Here an alternative method to these approaches is presented by considering the representation of the characteristic function. This approach allows for inherent BDD graph sharing to occur and does not require time multiplexing of the circuit outputs or the addition of extra disambiguating circuitry. An analysis of the extra circuitry required to represent the characteristic function is given and alternative techniques for area reduction are also presented.

In particular, the characteristic function of an n-input, m-output Boolean function is represented in a BDD. Such characteristic functions are traditionally represented as *multi-valued input* (MVI), binary-valued output functions with the single MV input representing the m Boolean outputs interpreted as an integer. In the method presented here, the m binary outputs are represented as non-terminal BDD vertices that are incorporated into the BDD representing the function. It is shown that through use of simple adjacent level variable swapping algorithms, a BDD can be formulated that allows for the direct mapping method described in [5] to be applied such that no conflicts occur in the circuit outputs. Therefore, no disambiguation circuitry, timed output multiplexing, or, shared BDD subgraph detection is required.

*This work was supported in part by NSF grants CCR-0000891 and CCR-0097246

Several extensions of this technique are also examined. In particular, we incorporate a "mixed" form of BDD mapping that sacrifices some power savings for area minimization where internal BDD vertices are either mapped as described in [5] or as the more common 2:1 multiplexer mapping. We also incorporate an alternative cost function to the sifting algorithm that minimizes overall switching probability rather than the overall BDD vertex count in an attempt to further reduce power dissipation [6].

The next section provides a brief overview of the mapping technique originally proposed in [5] followed by a description of our extension utilizing the characteristic function. Next, we describe the "mixed" mapping method that sacrifices power minimization for overall transistor count reduction in the resulting circuit. A simple modification to the sifting algorithm that causes minimization of switching probability is also described. Finally, experimental results are given for a variety of benchmarks circuits and conclusions are drawn.

2 Shannon Circuit Mapping

It is well known that the paths from the root of a BDD [1] to a terminal represent a set of disjoint cubes and that for a given variable assignment, a unique path is activated in the graph. This principle, in conjunction with a circuit mapping technique that replaces non-terminal vertices with combinational logic whose signals propagate from the top of the graph toward the terminal vertices resulted in "timed Shannon circuits" as described in [5]. A significant savings in dynamic power dissipation was noted for such circuits since switching only occurs along two paths in the resulting circuit when a new set of circuit inputs are applied; first the previous path is switched off (all internal circuit nodes at logic-1 go to logic-0) and then the new path is activated. The inclusion of an "enable" signal ensures that only two paths are switched and led to the name "timed Shannon circuits". A simple example is shown in Figure 1.

It is noted that a variety of subcircuits can be used in addition to the AND/OR in Figure 1 such as NAND/NAND, NOR/NOR and AND/XOR as shown in Figure 2. These circuits all have dual-rail ouputs, however a single-rail solution is easily obtained using a cone detection algorithm with the initial dual rail solution. As an example, Figure 3 contains the single-rail circuit corresponding to the BDD in Figure 2.

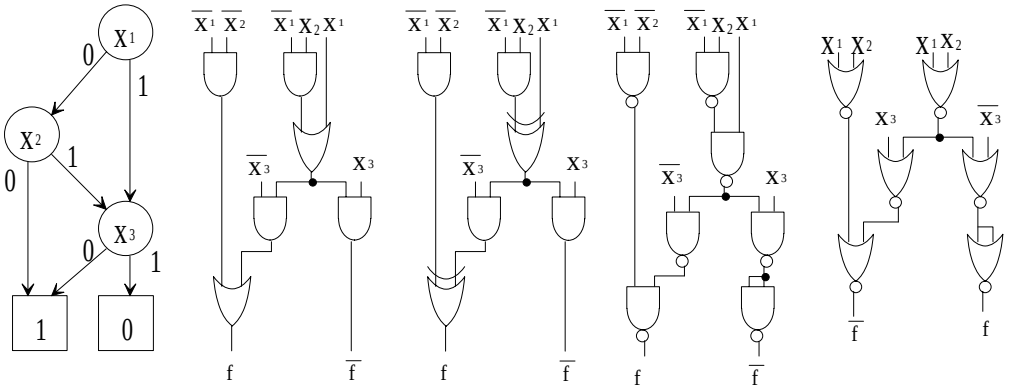

Figure 2: Various BDD Vertex Mappings

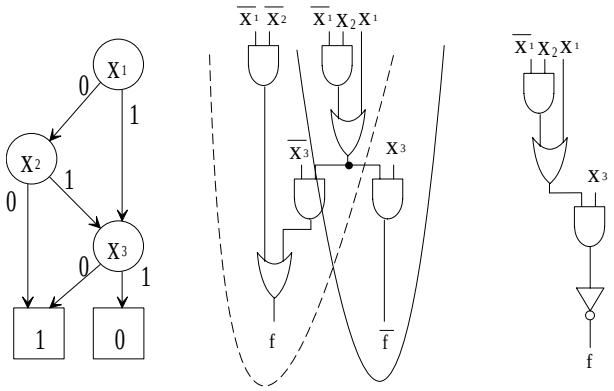

Figure 3: Cone Detection and Single-Rail Output Circuit

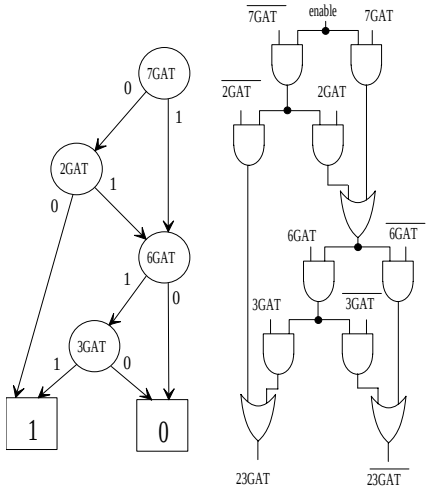

Figure 1: Timed Shannon Circuit Using AND/OR Mapping

2.1 Characteristic Function

A Boolean function, $f : B^n \rightarrow B^m$ may be represented by a characteristic function, $F : B^{n+m} \rightarrow B$, whose on-set, F^{ON}, corresponds to a complete cover of the truth table for f and all remaining points in B^{n+m} are considered to be members of the set, F^{OFF}. In terms of a BDD representation, the original BDD representing f is augmented with the inclusion of m non-terminal vertices corresponding to each unique output (these are referred to as "output nodes" for brevity). Figure 4 shows how the BDD for f can be augmented to form one for F.

By applying the circuit synthesis method reported in [5] to the BDD for F, inherent subgraph sharing occurs for all distinct outputs of f. The only modification is the mapping that corresponds to the m added "output nodes". In this case each output node is mapped to an multi-input OR gate (or tree of OR gates) whose output provides the overall circuit output.

2.2 Mixed Mapping Technique

In the mapping technique described in the previous section the resulting size of the netlist is dependent upon the size of the BDD representing the characteristic function. This provides motivation for reducing the size of the BDD. One approach is to form a group of output vertices in the BDD and then to perform group sifting [9] to position the output variables such that the overall BDD is minimal in size. In general the output variables are positioned in an intermediate level of the BDD. In this case all non-terminal vertices below the output nodes are mapped to 2:1 multiplexer structures while all those above are mapped as described previously.

It is easily shown that output nodes appearing at the top or bottom level of a BDD representing a characteristic function may only occur with unity multiplicity. However, when the output nodes reside in the middle of the characteristic function BDD, a

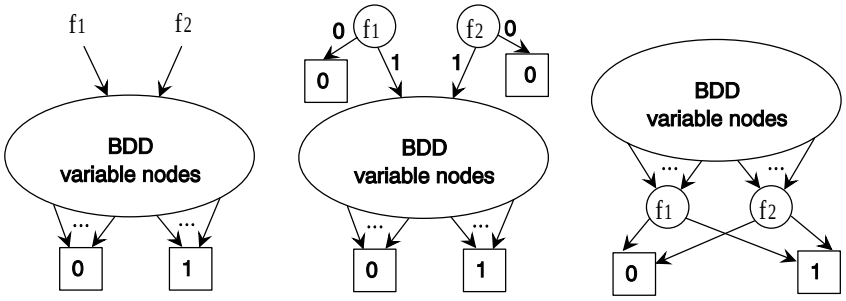

Figure 4: Creation of the Characteristic Function BDD

given output vertex may appear several times. In this case, the mapping procedure for the outputs must modified as shown in Figure 5.

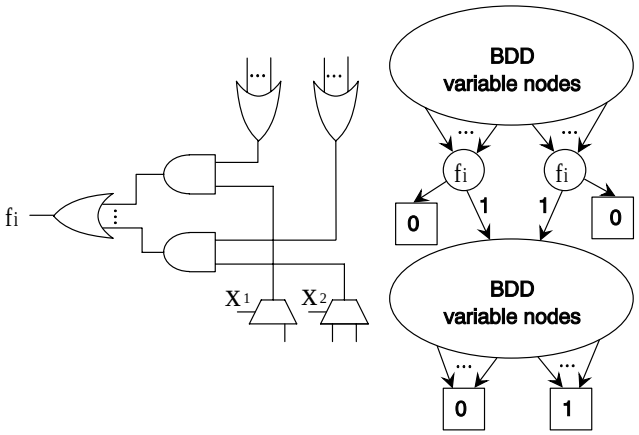

Figure 5: Output Node Mapping in Mixed Case

The mixed mapping method sacrifices some of the savings in power dissipation that originally motivated the development of timed Shannon circuits since those vertices below the output nodes are mapped to multiplexers. However, savings can result in terms of the overall BDD size and the critical path length of the resulting circuit.

2.3 Minimization of Switching Probability

Dynamic power dissipation of static CMOS circuitry is characterized by current spikes that occur when an internal net charges or discharges. Because this occurs when the input signals change value which, in turn, cause internal circuit nets to also change their logic values, the *switching probability*, P_{sw} can be used to compute power dissipation measures. In general, P_{sw} is difficult to compute without some knowledge of the temporal correlation of the input sequence [7]. If it is assumed that all inputs are equally likely to change in a statistically independent manner, a worst-case estimate for P_{sw} may be computed based on the structure of a BDD.

The sifting algorithm [11] can be modified to minimize the sum of all the P_{sw} estimates at each vertex rather than the BDD size as described [6]. A version of the mapping tool described above was created that incorporated this type of minimization and experimental results are provided for this case in the next section.

The output probability of a Boolean function f, denoted as $P[f]$ is the probability that f has a value of "1" at some arbitrary time of observation [10] [2] [4]. Consider a function f having the output probability $P[x]$ for the input variable x and the output probabilities $P[f_0]$ and $P[f_1]$ for the corresponding cofactors f_0 and f_1. In terms of a BDD, this relationship is shown in Figure 6.

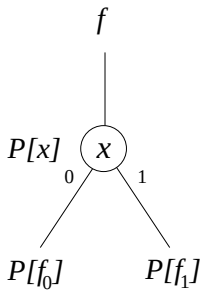

Figure 6: Switching Probability in BDD Vertex

The switching probability, $P_{sw}[f]$ of f, is the parameter of interest. Switching occurs if and only if the value of f changes from 0 to 1 or 1 to 0. We note that the probability that a function is 0-valued is given as the probability that the variable, x and the cofactor f_0 are 0-valued or that the variable, x is 1-valued but the cofactor, f_1 is 0-valued. A similar statement can be made for the probability that $f = 1$. These relationships are given in the following equations.

$$(1 - P[x])(1 - P[f_0]) + P[x](1 - P[f_1]) \quad for \;\; f = 0 \qquad (1)$$

$$(1 - P[x])P[f_0] + P[x]P[f_1] \quad for \;\; f = 1 \qquad (2)$$

Now, consider the value of f at two different observation times f^{t1} and f^{t2}. f is considered to "switch" if the value of $f^{t1} \neq f^{t2}$. Table 1 enumerates the possible states of f at two subsequent observation times, $t1$ and $t2$:

It is now possible to derive an upper bound of the switching probability based on the output probabilities given in Table 1. This is a worst-case value since there are inherent assumptions on the uncorrelated input signals to the circuit. Hence the switching probability $P_{sw}[f]$ of f can be computed as:

$$P_{sw}[f] = P[f^{t1} = 0 \cap f^{t2} = 1] + P[f^{t1} = 1 \cap f^{t2} = 0] \qquad (3)$$

$P[f^{t1} \cap f^{t2}]$	f^{t1}	f^{t2}
$((1-P[x])(1-P[f_0])+P[x](1-P[f_1]))^2$	0	0
$((1-P[x])(1-P[f_0])+P[x](1-P[f_1]))((1-P[x])P[f_0]+P[x](P[f_1]))$	0	1
$((1-P[x])P[f_0]+P[x](P[f_1]))((1-P[x])(1-P[f_0])+P[x](1-P[f_1]))$	1	0
$((1-P[x])P[f_0]+P[x](P[f_1]))^2$	1	1

Table 1: Probabilities at Subsequent Observations.

Using the expression in Table 1 and substituting them into Equation 3, we have the result:

$$P_{sw}[f] = 2((1-P[x])(1-P[f_0])+P[x](1-P[f_1])) \times((1-P[x])(P[f_0])+P[x](P[f_1])) \quad (4)$$

Equation 4 is expressed in terms of cofactors making it easy to compute during the adjacent level changes of variable nodes that occur when a BDD is undergoing sifting. By keeping the overall sum of the P_{sw} values at each BDD vertex an estimate of switching activity in the mapped circuit can be obtained. The sifting algorithm was modified to use $P_{tot} = \sum_{|E|} P_{sw_i}$ as a value to be minimized where $|E|$ is the total number of edges in the BDD rather than the usual metric of the overall vertex count, $|V|$. This modified cost function allows the BDD to grow in size during the sifting process, however excessive growth is not noted since the addition of new vertices leads to the addition of more P_{sw_i} values in the accumulated P_{tot}.

As is shown in the results section, the BDDs typically increase in size by a small amount and in some cases the use of the modified cost function allows for a smaller BDD to result as compared to the use of traditional sifting.

2.4 Multivalued Logic

This technique may easily be generalized to handle the case of multivalued logic (MVL). While BDDs are the most popular decision diagram structure, there has also been interest in *multivalued decision diagrams* (MDDs) [8] [12]. In producing a circuit composed of MVL gates, it is assumed the following basic gate types are available:

- MIN gates - the gate output is minimum of its input values
- 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, that is 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 MDD are mapped to small sets of MVL logic gates, producing a k-output circuit. If the MVL function being computed is $f(X)$, then the k outputs of the resulting circuit correspond to the characteristic functions of f, that is, $J_0(f(X)), \ldots J_{k-1}(f(X))$. The circuit thus outputs a form of 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 MDD, allowing the mapping technique to take full advantage of any advances in the area of decision diagram minimization.

3 Experimental Results

These techniques are implemented using the *CUDD* package [14] and options are included to map to AND/OR, NAND/NAND, NOR/NOR and AND/XOR subcircuits. After netlist mapping occurs, a secondary traversal of the netlist is invoked and circuit simplifications are performed where possible reducing the overall gate count. Output is generated as both a logic gate and as a SPICE transistor level netlist (the SPICE netlist utilizes a library of static CMOS logic gate cells). A maximum fanin parameter allows for the control of the size of the logic gates. For example, during AND/OR based mapping, very large OR gates can result since a single input is required for each BDD edge that points to a common vertex. By limiting the gate fan-in, a large m-input OR gate is replaced by a tree of OR gates.

Experimental results showing the resulting area of mapped benchmark circuits when the characteristic function is used and all output vertices are shifted to the bottom of the graph are given in Figure 2. All circuits were verified to be functionally equivalent to the original netlist by using the *SIS* command *verify -m bdd* [13]. The columns labeled **ORIG. SIZE** and **CHAR. SIZE** refer to the size of the BDD representing the function and that representing the characteristic function when all output vertices are shifted to the bottom of the graph and the other non-terminal vertices have been sifted. The column labeled **TIME** contains the amount of CPU time required to parse the original netlist, convert to a BDD and then the characteristic function BDD, move the output nodes to the bottom of the graph, invoke sifting and finally to generate the output netlist. The remaining columns indicate the number of logic gates and the transistor count of the resultant output netlists.

To reduce the size of the BDD representing the characteristic function and hence the resulting mapped netlist, the mixed mapping method was also implemented where a group is formed for the output vertices and they are positioned in the overall BDD so as to minimize the total node count. Table 3 contains the BDD size and netlist component count when this method is used.

In the next set of results shown in Table 4, the technique for minimizing the sum of the switching probabilities (assuming statistical independence and equally likely switching of the input variables) was used as a cost function for minimizing the BDD representing the characteristic function. These results were then mapped to netlists using the technique previously described and

Table 2: Mapping Results Using the Characteristic Function

NAME	ORIG. SIZE	CHAR. SIZE	TIME	NAND/ NAND		NOR/ NOR		AND/ OR		AND/ XOR	
				GATES	TRANS.	GATES	TRANS.	GATES	TRANS.	GATES	TRANS.
C17	7	13	0.01	16	68	23	76	17	68	17	102
C432	1210	1409	0.33	15776	4068	15790	4075	14006	3172	20642	3172
C499	26408	105320	129.2	304262	1163084	304294	1163148	276874	1108320	276874	1732858
C880	8412	12745	5.54	37889	150106	37922	150172	30105	134642	202164	30105
C1355	29562	108680	135.1	315663	1215422	315695	1215486	291642	1167392	29164	1845690
C1908	6253	24022	12.73	68940	264496	68965	264546	61700	250058	61700	390086
C2670	3981	6145	5.16	17072	65140	17137	65270	14387	59800	14387	91168
C3540	23829	58545	51.47	165056	621328	165080	621376	124561	540444	124561	769048
C5315	1778	11211	5.59	30668	118136	30844	118488	24448	105760	24448	159514
C7552	8408	13002	11.94	38100	151230	38212	151454	34242	143568	34242	228768
pair	3442	9193	3.84	24937	95632	25073	95904	18244	82332	18244	119164
vda	497	1260	0.16	2977	10906	3016	10984	2049	9078	2049	12588
des	3041	9468	2.19	25577	99976	25957	100736	20868	90584	20868	138388
rot	6008	8783	2.90	24736	96746	24845	96964	20068	87468	20068	133188

Table 3: Mapping Results for the "Mixed Mapping" Method

CIRCUIT NAME	BDD SIZE	BDD DIFF.	TOTAL GATES	NET. DIFF.
C17	13	0.0	17	0.00
C432	1409	0.0	3393	-75.7
C499	105320	-2.0	284234	2.7
C880	11674	-8.0	33732	12.1
C1355	108680	0.0	296357	1.6
C1908	24022	0.0	63028	2.2
C2670	6115	-9.8	14763	2.6
C3540	58547	0.0	133287	7.0
C5315	3541	-68.4	7581	-69.0
C7552	10397	-20.0	28676	-16.3
pair	7119	-22.6	15460	-15.3
vda	776	-38.4	1700	-17.0
des	5495	-42.0	11955	-42.7
rot	8069	-8.1	20704	3.2

the resulting size of the BDDs and netlists are compared when using sifting for BDD size reduction. It is noted that some of the resulting BDDs are actually smaller since the criteria of minimizing P_{sw} allows the BDD to grow in size during the minimization process hence allowing it to escape a local minimum (in terms of graph size) that it would otherwise be forced to converge to with conventional BDD sifting. This table contains columns giving the benchmark circuit name, followed by the sizes of the BDDs and netlists for each minimization method respectively.

4 Conclusions

Extensions of the technique for producing timed Shannon circuits based on the use of a BDD representing the characteristic function of a multi-output circuit have been presented. This method has some advantages over the originally proposed methods for dealing with multi-output circuits since time multiplexing of outputs may be avoided and the use of "shared graph" detection algorithms and disambiguation circuitry is not required. Extensions to the mapping method in the form of a "mixed mapping " approach and a modified sifting procedure for reducing estimated switching probabilities were also presented. Experimental results were provided for moderately sized benchmark functions to illustrate the utility of the approach.

References

[1] R.E. Bryant. Graph - based algorithms for Boolean function manipulation. *IEEE Trans. on Comp.*, 35(8):677–691, 1986.

[2] S. Chakravarty. A testable realization of CMOS combinational circuits. In *Int'l Test Conf.*, pages 509–518, 1989.

[3] E.M. Clarke, K.L. McMillan, X. Zhao, M. Fujita, and J. Yang. Spectral transforms for large Boolean functions with application to technology mapping. In *Design Automation Conf.*, pages 54–60, 1993.

[4] R. Krieger. PLATO: A tool for computation of exact signal probabilities. In *VLSI Design Conf.*, pages 65–68, 1993.

[5] L. Lavagno, P. McGeer, A. Saldanha, and A.L. Sangiovanni-Vincentelli. Timed shannon circuits: A power-efficient design style and synthesis tool. In *Design Automation Conf.*, pages 254–260, 1995.

[6] P. Lindgren, M. Kerttu, and M. A. Thornton. Low power optimization technique for bdd mapped circuits. In *Int'l Workshop on Logic Synth.*, pages 221–230, 2000.

[7] R. Marculescu, D. Marculescu, and M. Pedram. Efficient power estimation for highly correlated input streams. In *Design Automation Conf.*, 1995.

[8] D. M. Miller and R. Drechsler. Implementing a multiple-valued decision diagram package. In *Int'l Symp. on Multi-Valued Logic*, 1998.

[9] S. Panda and F. Somenzi. Who are the variables in your neighborhood. In *Int'l Conf. on CAD*, pages 74–77, 1995.

Table 4: Comparison of Resulting Netlists When Minimizing BDD Size Versus P_{sw}

CIRCUIT	Minimized Area		Minimized P_{sw}	
5xp1	123	214	111	207
alu4	903	2150	903	2152
apex1	2357	4182	2392	4266
apex2	531	1485	561	1330
apex4	1488	3189	1488	3189
apex5	1624	3291	1703	3322
b12	89	156	88	153
bw	173	214	173	204
clip	149	330	136	309
con1	20	35	20	35
cordic	84	198	84	198
cps	2543	3709	2641	3435
duke2	688	793	688	961
e64	2212	2208	2212	2143
ex1010	2090	4998	2110	5056
ex4p	560	1322	638	1492
ex5p	709	1111	682	1048
inc	66	102	66	102
misex1	77	115	76	115
misex2	184	214	188	197
misex3	833	1546	825	1543
misex3c	802	1865	792	1865
o64	133	473	133	473
pdc	4642	10290	4642	10290
rd53	34	64	34	64
rd73	54	124	54	124
rd84	77	170	77	170
sao2	113	313	113	216
seq	2423	4286	2424	4316
spla	222	223	222	179
sqrt8	46	81	46	81
squar5	63	86	66	88
t481	35	88	35	88
table3	1725	2886	1725	2886
table5	1821	3146	1822	3150
vg2	235	419	222	391
xor5	12	26	12	26

[10] K.P. Parker and E.J. McCluskey. Analysis of logic circuits with faults using input signal probabilities. *IEEE Trans. on Comp.*, 24:573–578, 1975.

[11] R. Rudell. Dynamic variable ordering for ordered binary decision diagrams. In *Int'l Conf. on CAD*, pages 42–47, 1993.

[12] T. Sasao and J.T. Butler. A method to represent multiple-output switching functions by using multi-valued decision diagrams. In *Int'l Symp. on Multi-Valued Logic*, pages 248–254, 1996.

[13] E. Sentovich, K. Singh, L. Lavagno, Ch. Moon, R. Murgai, A. Saldanha, H. Savoj, P. Stephan, R. Brayton, and A. Sangiovanni-Vincentelli. SIS: A system for sequential circuit synthesis. Technical report, University of Berkeley, 1992.

[14] F. Somenzi. *CUDD: CU Decision Diagram Package Release 1.1.1*. University of Colorado at Boulder, 1996.