Test Pattern Generation for Circuits Using Power Management Techniques

João P. Marques Silva
CEL/IST/INESC
1000 Lisboa, Portugal
jpms@inesc.pt

José C. Monteiro
IST/INESC
1000 Lisboa, Portugal
jcm@inesc.pt

Karem A. Sakallah
University of Michigan
Ann Arbor, MI 48109-2122
karem@eecs.umich.edu

Abstract

Power dissipation has recently emerged as one the most critical design constraints. A wide range of techniques has already been proposed for the optimization of logic circuits for low power. Power management methods are among the most effective techniques for power reduction. Nevertheless, power management techniques based on inhibiting the clock signal create some interesting testability problems. In this paper, we propose an approach for the complete testing of power-optimized circuits using these techniques. Moreover, we describe an ATPG tool, TG-GRASP, that incorporates several powerful search pruning techniques, which are particularly useful for circuits with many hard-to-detect faults. As a result, we are able to present results that show that by using TG-GRASP, ATPG for the power managed circuit is not significantly more difficult than for the original circuit.

1. Introduction

Automatic Test Pattern Generation for combinational circuits has been the subject of intensive research in recent years. As a result of this continued effort, it is now generally accepted that the large number of existing algorithmic techniques permits ATPG tools to efficiently handle most combinational circuit designs. Moreover, extensive experiments have also been conducted to justify using chosen algorithmic approaches instead of others. In this paper we show that recent synthesis algorithms challenge some of these established ideas. First, we show that synthesis algorithms can yield circuits with a large number of hard-to-detect redundant faults. Indeed, this is the case with synthesis for low-power circuits, where certain low-power design techniques yield many redundant faults. Second, we show that circuits synthesized for low-power may pose significant difficulties for existing ATPG algorithms.

Low-power synthesis algorithms are of primary importance for several reasons. On one hand, the rapid increase in clock frequencies, chip complexity and scale of integration is creating significant heat dissipation problems. On the other hand, we have the proliferation of portable devices. For personal communication applications like hand-held mobile telephones, battery lifetime may be the decisive factor in the success of the product. Power reduction techniques have been proposed at all design levels, from system to device. It has been demonstrated at the gate and system levels that large power savings are possible merely by cutting down on wasted power, commonly referred to as power management.

Several methods have been presented that perform the shutdown of a section of the circuit on a clock-cycle base. Depending on some input conditions, the clock driving some of the registers in the circuit is inhibited, therefore reducing the switching activity in the fanout of those registers. These techniques are referred to as *data-dependent* power management techniques.

Data-dependent power management techniques create some interesting testability problems. In order to detect the input conditions that allow for the disabling of the clock signal, some extra circuitry has to be added to the original circuit. Since the functionality of the circuit is not being altered, this logic is redundant, thus dramatically reducing the observability of some of the nodes in the circuit. Consequently, for these circuits a significant number of redundant faults must be handled by the ATPG tool, thus posing new challenges to those tools.

In this paper, we first propose an approach for the complete testing of power-optimized circuits using data-dependent power management techniques. This approach is based on a two input vector sequence. The first input vector guarantees that we can always set the output of the latches to the required value, even if the clock is disabled in the second input vector.

Two different ATPG tools are then used to tests circuits using power management, TEGUS [7] and TG-GRASP [6]. In particular, TG-GRASP is a SAT-based ATPG tool that incorporates several powerful search pruning techniques [4], which are particularly useful for handling circuits with many redundant faults. Experimental results obtained on the MCNC'91 benchmark circuits indicate that both ATPG tools are efficient for circuits using power management, and that TG-GRASP is the more robust ATPG tool of the two, being more insensitive to the larger number of redundant faults in circuits using power management.

Throughout the paper precomputation-based synthesis for low-power is assumed. Complete details of this approach can be found in [1, 2, 5]. The ATPG tool TG-GRASP is described in [6]. Complete details of the ATPG approach and ATPG tool proposed in this paper can be found in [5].

2. ATPG for Power Managed Circuits

Data-dependent power management techniques use mechanisms to prevent transitions in some logic signals from propagating to combinational logic circuits, thus reducing power consumption. As a consequence, some input combinations to the combinational logic block may no longer be possible, hence dramatically reducing the effectiveness of ATPG programs. The major test generation problems arise from the fact that the precomputation logic will prevent some input combinations to the logic block from happening. For this reason, in order to fully test the power managed circuit, we may need two input vectors. Still, even using two input vectors for testing, the precomputation logic introduces many redundant faults. Redundant faults can create difficulties for most ATPG programs as they try to prove that the fault is indeed redundant.

Whenever power management is asserted, some input transitions are prevented from reaching a portion of some combinational logic block. This may impede certain input combinations at this logic block from happening. We propose to solve this controllability problem by using a two input vector sequence. As observability of the outputs is assumed, the result of the test can be verified after the second input vector.

There are two situations to consider. First, assume that the fault we are considering can be detected using some input combination that disables the precomputation logic. Then, the first input vector is of no relevance since the second input vector will be loaded into the input registers and thus we are able to set the inputs of the combinational logic to any value that is required to detect the fault.

Now, consider that some fault can only be detected by input combinations that assert the precomputation logic. In this case, we build the first input vector such that precomputation is disabled and load the desired values to the registers disabled by precomputation. Next, the second input vector needs only have the correct values for the remaining registers since the first registers will already have the correct values and will not be disturbed since the precomputation logic will be active.

We outline a circuit transformation technique for the automatic generation of the two input vector sequence used in the testing of power managed logic circuits. This will allow us to use combinational ATPG techniques, thus avoiding the more computationally expensive sequential test generators.

The proposed transformation consists of duplicating the inputs of the original circuit. The subset of inputs used in the precomputation logic will always be present at the input of the combinational logic.

The remaining inputs of the second input vector go to the precomputation logic and the output of this logic will decide if these values or those of the first input vector are the inputs to the circuit.

We can now run standard ATPG tools on the circuit after transformation to obtain values for both input pattern that detect any fault in the original combinational logic block and associated precomputation logic. However, we have duplicated the number of inputs, increasing dramatically the input search space of the ATPG tool. Additionally, the circuit after transformation will have a significant amount of redundancy, further complicating the problem for the ATPG tool.

3. Experimental Results

In this section we compare the testability of a subset of the circuits in the MCNC'91 combinational benchmark set before and after adding data-dependent power management. In Table 1 we present the literals (Lits) in the circuit and its power dissipation (Power). All statistics is this table were obtained using SIS [3]. Also in Table 1, we show the power savings obtained after precomputation is applied to each circuit. Under **Precomputation** [1], we give the number of literals in the precomputation logic and the power of the power managed circuit. We can see that the size of the precomputation logic is in general much smaller than the original circuit. The last column of Table 1 indicates the percentage savings in power obtained through precomputation. As we can observe, power reductions of up to 77% are possible.

To measure the testability of the circuits, we have used two different ATPG tools, namely the SAT-based ATPG tools TEGUS [7], which is included in the SIS package [3], and TG-GRASP, described in [6].

Circuit	**Original**		**Precomputation**		**%Red**
	Lits	**Power**	**Lits**	**Power**	
9symml	327	1837.1	40	1487.4	19.0
alu4	1150	4525.6	8	3939.7	12.9
apex2	390	2322.7	4	1201.2	48.3
cht	329	2310.9	1	1797.8	22.2
cm138a	46	299.5	3	172.2	42.5
cm150a	104	1005.0	1	682.5	32.1
cm163a	90	832.6	0	646.1	22.4
cmb	99	800.2	10	413.4	48.3
comp	188	1712.6	13	720.9	57.9
comp16	413	2597.8	17	907.9	65.1
cordic	136	1197.3	17	729.7	39.1
cps	1979	4518.8	26	2879.5	36.3
cu	102	761.7	5	510.1	33.0
dalu	1611	7014.2	20	3638.4	48.1
duke2	716	2243.7	20	1527.6	31.9
e64	379	2554.7	5	573.7	77.5
i2	369	7748.5	22	2520.0	67.5
majority	21	220.6	1	167.0	24.3
misex1	94	623.6	6	547.0	12.3
misex2	163	1223.0	16	968.0	20.9
misex3	1111	3473.1	2	2503.9	27.9
mux	89	990.4	0	677.9	31.6
sao2	223	1052.9	2	446.1	57.6
seq	2590	7494.2	0	3923.1	47.7
term1	288	1993.8	6	1581.4	20.7
too_large	234	2396.4	1	1561.1	34.9

Table 1: Power reduction through precomputation

TEGUS is implemented in the C programming language, whereas TG-GRASP is implemented in C++. The CPU times reported are for a Sun 5/85 machine with 64 MByte of physical memory. Both tools were compiled with the same optimization options. In Table 2 the ATPG results for the original MCNC'91 benchmark circuits are shown. The circuit transformation described Section 2 and in [5] was applied to the precomputed circuits and the ATPG programs were run on the modified circuit. The ATPG results after precomputation are also given in Table 2. In each table **#F**, **#R**, **#A** and **CPU** denote, respectively, the total number of faults, the number of redundant faults, the number of aborted faults and the CPU time for each tool. For each benchmark *all* faults are targeted. This solution permits a larger set of faults to be studied and guarantees that each ATPG tool is presented with exactly the same set of faults.

From the results shown several conclusions can be drawn. First, TEGUS and TG-GRASP have similar performances in both types of circuits, with and without power management. Nevertheless, it is clear that TG-GRASP performs better in circuits using power management, which have a large number of redundant faults. In

Circuit	Before Precomputation								After Precomputation							
	#F	TEGUS			TG-GRASP			ratio	#F	TEGUS			TG-GRASP			ratio
		#R	#A	CPU	#R	#A	CPU			#R	#A	CPU	#R	#A	CPU	
9symml	752	2	0	15.9	2	0	18.3	1.15	932	42	0	160.2	42	0	65.6	0.41
alu4	2722	26	0	195.6	26	0	195.4	1.00	2984	106	0	238.8	106	0	234.9	0.98
apex2	948	3	0	77.1	3	0	25.3	0.33	1492	249	0	142.1	249	0	79.5	0.56
cht	820	0	0	3.5	0	0	5.6	1.60	1468	317	0	10.6	317	0	17.2	1.62
cm138a	124	0	0	0.4	0	0	0.4	1.00	190	27	0	1.1	27	0	1.2	1.09
cm150a	232	0	0	1.8	0	0	2.0	1.11	526	142	0	7.9	142	0	7.3	0.92
cm163a	220	0	0	1.2	0	0	1.4	1.17	392	77	0	3.7	77	0	3.8	1.03
cmb	248	0	0	1.2	0	0	1.7	1.42	460	96	0	7.5	96	0	7.3	0.97
comp	480	1	0	6.2	1	0	6.6	1.06	956	207	0	28.1	207	0	26.0	0.93
comp16	960	0	0	15.7	0	0	17.2	1.10	1448	220	0	173.8	220	0	53.4	0.31
cordic	342	0	0	3.2	0	0	3.3	1.03	616	121	0	14.7	121	0	13.2	0.90
cps	4642	2	0	146.6	2	0	199.5	1.36	4996	152	0	457.9	152	0	351.2	0.77
cu	262	7	0	1.3	7	0	1.5	1.15	444	86	0	4.7	86	0	5.3	1.13
dalu	3742	1	1	313.8	2	0	201.0	0.64	4794	519	1	513.1	520	0	468.1	0.91
duke2	1708	0	0	41.7	0	0	53.7	1.29	1970	123	0	85.1	123	0	104.9	1.23
e64	1142	0	0	27.3	0	0	34.6	1.27	2122	430	0	128.0	430	0	139.0	1.09
i2	762	2	0	17.1	2	0	18.7	1.09	3696	1325	0	425.1	1325	0	357.7	0.84
majority	54	0	0	0.1	0	0	0.2	2.00	116	30	0	0.5	30	0	0.4	0.80
misex1	224	0	0	1.9	0	0	1.9	1.00	278	26	0	2.7	26	0	2.9	1.07
misex2	422	0	0	2.5	0	0	3.0	1.20	752	141	0	10.3	141	0	12.9	1.25
misex3	2590	7	0	105.8	7	0	130.2	1.23	2792	96	0	145.9	96	0	170.1	1.17
mux	202	0	0	1.4	0	0	1.5	1.07	494	140	0	6.7	140	0	6.5	0.97
sao2	532	3	0	8.1	3	0	10.2	1.26	672	64	0	14.4	64	0	16.2	1.13
seq	5912	4		1375.4	4	0	378.6	0.28	6536	277	0	838.7	277	0	593.4	0.71
term1	708	6	0	6.2	6	0	7.6	1.23	1130	200	0	21.1	200	0	22.4	1.06
too_large	1132	15	0	27.5	15	0	35.0	1.27	1706	276	0	74.9	276	0	92.5	1.23

Table 2: ATPG experimental results

this situation, TG-GRASP is in general more efficient than TEGUS. Furthermore, TG-GRASP is more robust, since it aborts no faults and its run times grow proportionally to the number of faults. In contrast, TEGUS aborts one fault, and in general its run times experience significant variations when power management is added to a given circuit. Finally, we note that additional results given in [5] suggest that structural ATPG algorithms perform significantly worse in the presence of precomputation logic.

4. Conclusions

Logic synthesis techniques for low power can significantly change the testability of a circuit. In this paper we propose an ATPG methodology and an ATPG algorithm for testing circuits synthesized for low-power. Experimental results, obtained on a large number of benchmarks, as well as those given in [5] validate the proposed approach.

References

[1] M. Alidina, J. Monteiro, S. Devadas, A. Ghosh and M. Papaefthymiou, “Precomputation-Based Sequential Logic Optimization for Low Power,” *IEEE Transactions on VLSI Systems*, vol. 2, no. 4, pp. 426-436, December 1994.

[2] J. Monteiro, J. Rinderknecht, S. Devadas and A. Ghosh, “Optimization of Combinational and Sequential Logic Circuits for Low Power Using Precomputation,” in *Proceedings of the 1995 Chapel Hill Conference on Advanced Research on VLSI*, pp. 430-444, March 1995.

[3] E. Sentovich, K. Singh, C. Moon, H. Savoj, R. Brayton and A. Sangiovanni-Vincentelli, “Sequential Circuit Design Using Synthesis and Optimization,” in *Proceedings of the International Conference on Computer Design*, pp. 328-333, October 1992.

[4] J. P. M. Silva and K. A. Sakallah, “GRASP—A New Search Algorithm for Satisfiability,” in *Proceedings of the International Conference on Computer-Aided Design*, November 1996.

[5] J. P. M. Silva, J. C. Monteiro and K. A. Sakallah, “Test Pattern Generation for Circuits Using Power Management Techniques,” Technical Report RT/06/97, INESC, Portugal, April 1997.

[6] J. P. M. Silva and K. A. Sakallah, “Robust Search Algorithms for Test Pattern Generation,” in *Proceedings of the Fault-Tolerant Computing Symposium*, June 1997.

[7] P. R. Stephan, R. K. Brayton and A. L. Sangiovanni-Vincentelli, “Combinational Test Generation Using Satisfiability,” *IEEE Transactions on Computer-Aided Design*, vol. 15, no. 9, pp. 1167-1176, September 1996.