第32卷 第15期 计 算 机 工 程 2006年8月
Vol.32 № 15 Computer Engineering August 2006
·工程应用技术与实现· 文章编号：1000—3428(2006)15—0245—02 文献标识码：A 中图分类号：TP 391

一种低功耗SoC芯片的综合BIST方案

方祥圣[1,2]，梁华国[2]，曹先霞[3]
（1.安徽经济管理学院计算机系，合肥 230051；2.合肥工业大学计算机与信息学院，合肥 230009；3.安徽省公路局培训中心，合肥 230051）

摘 要：提出了一种低功耗的综合BIST方案。该方案是采取了屏蔽无效测试模式生成、提高应用测试向量之间的相关性以及并行加载向量等综合手段来控制测试应用，使得测试时测试向量的输入跳变显著降低，从而大幅度降低芯片的测试功耗。测试实验表明，该方案既能减少测试应用时间，又能够有效地降低芯片测试功耗，平均输入跳变仅为类似方案的2.7%。

关键词：SoC芯片；内建自测试；低功耗

Low-power Synthesis Scheme for SoC BIST

FANG Xiangsheng[1,2], LIANG Huaguo[2], CAO Xianxia[3]
(1. Department of Computer, Anhui Economy Management Institute, Hefei 230051; 2. Institute of Computer and Information, Hefei University of Technology, Hefei 230009; 3. Training Center of Anhui Highroad Bureau, Hefei 230051)

【Abstract】 This paper presentes a low-power synthesis BIST scheme. The scheme adopts some synthesis measures that deletes the void or redundancy testing patterns and increases the relativity of the test vectors and parallel loaded test vectors, so that the power consumption inside the circuit under testing is reduced enormously. This scheme not only decreases testing-time, but also reduces testing-power effectively. The average input switching activity is only 2.7% of the similar type scheme.

【Key words】 SoC chip; BIST; Low-power

近年来，随着SoC芯片的出现，内建自测试（Built-In Self-Test，BIST）成为人们研究的热点。然而，随着集成电路特征线宽的持续减小，尤其到了深亚微米设计阶段，功耗问题却变得越来越突出，因为过大的功耗会严重影响电路的性能，甚至使电路失效[1]。研究显示芯片在测试阶段的功耗是正常工作状态下的功耗的200%[2]。这种功耗的增加主要原因是：(1)芯片在正常工作状态下，其内部工作的模块较少，而测试时则使其尽可能多的模块工作。(2)芯片在测试时各节点的翻转次数应尽可能增多，即翻转明显增加。

最近研究者们提出了一些关于BIST中功耗约简的研究方法。典型工作有：文献[3]提出了基于扫描链的BIST功耗约减技术；文献[4]提出了低功耗的测试向量的自动生成（ATPG）技术；文献[5]提出了控制功耗的静态压缩技术；文献[6]提出了扫描分段控制功耗的技术。

1 CMOS电路的功耗问题

通过研究可知，CMOS电路的功耗主要分为静态功耗和动态功耗。静态功耗主要来源于持续的漏电流，它相对较小；而动态功耗主要来源于片内各节点的1/0，0/1翻转(switching activity，SA）时的短路电流和负载电容的充、放电[8]。在当前的CMOS电路中，动态功耗是主要功耗。

衡量电路功耗特性的重要参数如下：电路节点 i 在一个周期中，功耗消耗可估计为

$$E_i = \frac{1}{2} S_i F_i C_0 V_{DD}^2 \quad (1)$$

其中，V_{DD}为电源电压，S_i是一个周期翻转的次数，F_i为节点的扇出，C_0是最小的负载电容。

由式（1）可知：在测试过程中，WSA(Weighed Switching Activity）是节点i功耗E_i的唯一变量，所以WSA可作为该节点的功耗估计。对于一对连续的输入向量$TP_k=(V_{k-1},V_k)$,电路总的WSA为

$$W(TP_K) = \sum_i S(i,k) F_i \quad (2)$$

其中i是电路中所有节点的个数，$S(i,k)$是由TP_k所激励节点i的翻转次数。

根据式(2)，考虑长度为L的测试向量TS作为电路的输入向量，电路总的WSA为

$$W_{TS} = \sum_k \sum_i S(i,k) F_i \quad (3)$$

根据式(3)可知，常见的降低功耗的方法有：

(1)降低工作电压和测试电压V_{DD}；

(2)测试时降低激励节点的翻转次数，即减少激励的输入跳变。

2 折叠计数器的工作原理简介

折叠计数器是一种可编程的约翰逊计数器，其状态序列的生成，可以通过对线性移位寄存器（LFSR）的反馈电路编程来实现；也可以根据同一个初始状态，规则的翻转其状态位来获得[9]。当折叠计数器从一个初始状态$s\in\{0,1\}^n$开始，产生一个$n+1$状态序列$s=F(0,s),F(1,s),\ldots,F(n,s)$。其中状态$F(i,s)$能够根据初始状态$s$执行多次翻转后推导出，翻转仅仅由位置$j$和状态距离值$i$确定。由下式给出翻转函数。

基金项目：国家自然科学基金资助项目（90407008）；教育部留学回国人员科研基金资助项目（2004.527）；安徽省自然科学基金资助项目（050420103）

作者简介：方祥圣(1969—)，男，硕士生、讲师，主研方向：内建式BIST，嵌入式系统；梁华国，教授、博导；曹先霞，讲师

收稿日期：2006-03-09 **E-mail：fangxiangs@tom.com**

$$inv(j,i)=\begin{cases} j & if \quad j<i \\ i & else \end{cases} \qquad (1\le j\le n,\ 0\le i\le n)$$

折叠序列计算公式：$F(i,s)=(\neg^{inv(1,i)}s_1,\ \neg^{inv(2,i)}s_2,\ldots,\neg^{inv(n,i)}s_n)\quad s=(s_1, s_2, \ldots, s_n)\ (0\le i\le n)$

折叠计数器的控制过程[9]：折叠种子送入LFSR，由折叠控制器(主要由折叠距离计数器、比较器、两输入的多路选择器和位计数器组成)通过异或门控制对初始状态的翻转，实现翻转函数$inv(j,i)$的功能；异或门的输出连至被测电路(CUT)的扫描链，从而将折叠种子展开称n+1 个测试应用序列。

折叠计数器的主要特性：(1)一个折叠种子可以生成 n+1 个折叠计数器序列，每间隔一个序列的两个序列只有一位不同，最后两个相邻序列仅最后一位不同。(2)利用折叠关系可以对任何一个测试集构造一个较小的折叠种子集。(3)折叠种子的展开可以依次连续生成，也可以通过赋给折叠距离计数器的距离值，选择生成对应的序列。

3 本文建议的低功耗 SoC 芯片综合 BIST 方案

使用双种子压缩方案[7]，就是利用折叠计数器特性(2)对确定的测试集T_D进行压缩的。它先将T_D压缩成折叠种子集，再对每个折叠种子运用 LFSR编码成该寄存器种子；当每个种子展开时，必须将一个完整的折叠计数器状态序列生成出来；这样存在以下问题：不同种子所产生的测试序列之间包含大量的无效和冗余的测试序列；且测试序列间跳变频繁；同时测试序列又串行移入扫描链，势必增加CUT中各节点的跳变，从而导致测试功耗过大。因此，删去无效和冗余的测试模式，降低测试模式间的输入跳变，是方案[7]有待改进的问题。为此，本文提出了一种新的控制策略。如下所述：

由于T_D是一个确定的测试集，在构造最小折叠种子集的过程中[9]，每个种子所生成的折叠计数器序列中包含确定的测试模式数是已知的，并且对应的折叠距离值也是确定的。因此，只要确定了所有的种子及其对应的有效折叠距离值后，依据折叠计数器特性(3)，就可生成确定的测试集T_D，从而避免了大量无效和冗余的折叠序列的生成，同时对有效应用模式重新排序后，并行加入扫描链，使得输入跳变大幅度降低，从而达到降低功耗的目的。下面给出每个折叠种子与其折叠距离值的对应关系。首先，按照每个种子对应的确定折叠距离值分组一一对应种子排列，用每个种子对应距离值的个数来控制每组距离值的装载。具体的数据结构关系见图 1，其中折叠种子用$s_i(1\le i\le k)$表示，种子对应的确定折叠距离数用$m_i(1\le i\le k)$表示，具体的距离值用$FD_{ij}(1\le i\le k, 1\le j\le m_i)$表示。

$(S_1, m_1, (FD_{11}, FD_{12}, \ldots FD_{1m1}))$

$(S_2, m_2, (FD_{21}, FD_{22}, \ldots FD_{2m2}))$

$\vdots$

$(S_k, m_k, (FD_{k1}, FD_{k2}, \ldots, FD_{kmk}))$

图 1 折叠种子与折叠距离值关系

在表 1 中给出了一个具体实例，说明建议方案中测试模式生成过程。

例如：种子s_1=（0,1,0,0,0,1,0），s_2=(0,1,1,0,1,0,1)播种时展开的完整测试序列如表 1 所示，其中确定的测试模式是{0100010, 1100010, 1110101, 1110110, 1001010, 1101010}，分别包含在折叠种子s_1与s_2生成的折叠序列中，对应表 1 中灰色背景的序列，s_1和s_2包含的确定测试模式所对应的折叠距离分别是FD_1={0, 2, 5, 6}，FD_2={1, 2, 3, 5}，可见种子s_2的折叠距离 2 和 5 对应的序列与种子s_1的折叠距离 5 和 2 序列重叠，因此，可以删除s_2的 2、5 距离值，则总的确定测试模式对应到（s_1，m_1=4，FD={0, 2, 5, 6}）与（s_2，m_2=2，FD={1, 3}）；为了进一步降低功耗，我们按最小跳变原则对折叠距离FD_{ij}按 0→2→4…→2n→2n-1…→3→1 的规律重新排列次序，即对应于（s_1，m_1=4，FD={0, 2, 6, 5}）与（s_2，m_2=2，FD={3, 1}），那么产生的测试序列为：{0100010, 1100010, 1110110, 1110101, 1101010, 1001010 }。这样测试时生成的总测试模式数是 6，输入跳变数是 6，而原方案[7]中，总测试模式数是 16，输入跳变数是 56，可见本方案的应用模式数仅为原方案[7]的 37.5%，输入跳变仅为原方案[7]的 10.7%。这样既降低了测试时间，又降低 了测试功耗。本方案对测试时间及测试功耗的降低效果是非常显著的，这在后面的实验结果中可看出。

表 1 折叠种子s_1与s_2生成的折叠序列

折叠距	折叠序列(s_1)	折叠序列(s_2)
0	0100010	0110101
1	1011101	1001010
2	1100010	1110101
3	1111101	1101010
4	1110010	1100101
5	1110101	1100010
6	1110110	1100001
7	1110111	1110000

下面给出本文总体方案：

(1)针对硬件故障F_{hard}测试集T⊂{0，1，-}^{n}，构造一个最小的折叠种子集T'⊂{0，1，-}^{m}。

(2)利用 LFSR 对每个折叠种子实施编码，获得一个小的 LFSR 的种子。测试期间经过 LFSR 解压还原成折叠种子。

(3)折叠距离值的选择与折叠距离数的统计。删除在折叠集中无效和冗余的折叠距离值，最终获得的折叠距离值与硬故障集中测试模式是一一对应的。

(4)按数据结构(s_i, m_i, {$FD_{i1}, FD_{i2}, \ldots, FD_{imi}$})排列种子、折叠距离数以及对应的折叠距离值。

(5)按最小输入跳变原则对上述数据结构中的折叠距离{$FD_{i1}, FD_{i2}, \ldots, FD_{imi}$}进行重新排序；测试时将测试模式并行加入扫描链。

本文方案建议的解压结构如图 2 所示。

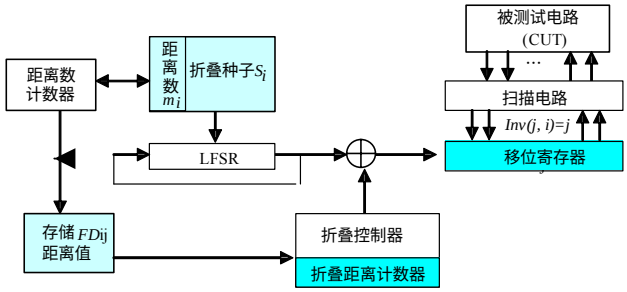

图 2 本文建议的综合方案的解压结构

在此结构中，不同于原方案的是，添加了一个折叠距离数计数器和移位寄存器，折叠距离数计数器用以控制装载和更换种子及对应的折叠距离值。这里仅生成有效测试模式的排序序列，而不是生成折叠种子的完整序列；同时移位寄存器的添加使得测试向量并行加入扫描链，这将会非常有效地降低测试功耗。实验表明该方案的功耗只是原方案[7]的 2.7%，故本文建议方案为一个切实可行的优秀方案。

4 实验验证

本方案采用 ISCAS-89 和 ISCAS-85 组合逻辑电路部

（下转第 249 页）