

基于 LUT 的 ATPG 算法研究

王志远,李彩虹,何安平

(兰州大学 信息科学与工程学院,甘肃 兰州 730000)

摘 要: 随着科学技术尤其是半导体工艺与通信技术的快速发展,FPGA 技术日趋成熟,采用 FPGA 的设计愈加广泛.随着系统设计的复杂化,其 FPGA 的验证在整个开发周期中占据的比重越来越大.针对以上背景,本文采用系统功能验证中常用的自动测试模式生成(ATPG)算法为基本框架的 ATALANTA 开源软件,研究基于 FPGA 的基本查找表(LUT)结构的 ATPG 验证技术,提出了一种基于 LUT 的 ATPG 验证算法和这种算法在 ATALANTA 软件中的实现方法,实验结果表明了这种方法的可行性.

关键词: ATPG 算法;ATALANTA;FPGA;LUT 查找表

中图分类号: TP301

文献标识码: A

文章编号: 1000—7180(2017)04—0044—05

A Study of ATPG Based on LUT Technology

WANG Zhi-yuan¹, LI Cai-hong¹, HE An-ping¹

(School of Information Science & Engineering, Lanzhou University, Lanzhou 730000, China)

Abstract: As the science and technology especially the semiconductor and communications technology developing rapidly, the designs based on FPGA technology are adopted much wider and becomes applicable for large system. Large system brings complex designs; the verification of FPGA design occupies larger part of the whole system development. So in this paper, by studying the ATPG technology and the ATALANTA, an open-source ATPG solver, we introduce a new LUT based ATPG method that has been implemented by extend ATALANTA with many new features. The case study in this paper shows the feasibility of our method.

Key words: ATPG;ATALANTA;FPGA;Look-Up-Table

1 引言

基于 FPGA 的 IC 设计越来越广泛,其验证问题成为决定 FPGA 设计成败的关键环节.对集成电路进行测试验证,目的就是为了寻找集成电路设计中存在的缺陷和故障. ATPG^[1]算法是一种测试验证的优秀算法,尤其是对固定型故障^[2]的测试验证得到了广泛应用.文献[3]使用 ATPG 算法对复杂 IC 系统做等价性验证,故障点设置的数量会非常庞大;文献[4]将电路根据黑盒设定划分若干逻辑锥,但对黑盒中电路进行验证时需将三值逻辑转换成二值逻辑,并且必须要确定黑盒功能,算法复杂性非常

高;此外,还有一些其他改进型 ATPG 算法,都是从优化搜索空间和得到更多启发式信息来测试和判断某固定型故障并减少测试产生时的测试时间.文中,我们对 ATPG 算法和 ATALANTA^[5]引入 LUT^[6]技术进行局部改进,减少测试产生时故障数量设定以加速整个测试产生验证过程.最后,对以 ATPG 算法为基本框架的开源软件 ATALANTA 引入 LUT 技术进行改进,最后选取 ISCAS'85^[7]中 10 个 bench 组合电路并随机加入 LUT 模块进行测试验证,对测试验证的结果进行分析以验证我们对 ATPG 算法改进的正确性.

收稿日期: 2016—08—30; **修回日期:** 2016—09—14

基金项目: 国家自然科学基金项目(61073193,61300230);自然科学基金(61402121);中央高校基本科研业务费专项资金(861914)

2 ATPG 算法与 ATALANTA 软件

测试向量自动生成 (Automatic Test Pattern Generation, ATPG), 是由相关软件或求解算法在对集成电路进行测试产生时生成满足要求的解的一个过程。在使用 ATPG 算法对集成电路进行测试验证时分为以下步骤: 第一步, 对要进行验证的待检测电路选取节点进行单固定故障点设置; 第二步, 在进行测试生成时, 使用路径敏化和递归方法通过向主输入方向进行判断即 fault sensitizing 过程, 找到一组能够引起设置的故障值的合适的输入向量集; 若已经搜索完整个向量空间并且没有找到引起故障值的输入向量, 则测试产生过程直接结束; 第三步: 将找到的主输入向量向后传播直至主输出处, 即将故障值通过寻找的向量集向后传播至主输出处, 观察故障值在主输出处的结果值; 将引起此故障值的输出向量同样应用在规范电路的主输入处, 然后在规范电路的主输出处观察结果; 第四步: 若实现电路和规范电路在应用相同输入向量集的情况下, 它们在主输出处的结果相同, 则证明实现电路和规范电路在逻辑功能上是等价的; 否则, 则说明实现电路和规范电路在逻辑功能上是不等价的。

ATALANTA 软件是由美国布拉德利电气与计算机工程部、弗吉尼亚理工学院和州立大学等部门联合研发的一种对电路固定型故障进行诊断^[8]和模拟的自动测试向量生成器。ATALANTA 软件利用 ATPG 生成算法 FAN, 在 ATPG 算法基础上进行了充分蕴含、唯一敏化、多重回溯、分之界限等改进措施来进一步减小测试搜索空间和测试产生时间。因为在对电路进行测试产生时, 充足的回溯次数可以更好的寻找到测试向量。但是回溯次数太大会极大地增加测试产生时间。命令: `./atalanta -b n *.bench`, `-b` 代表 FAN 算法中的回溯, 而 `n` 代表测试产生过程中的回溯次数, 我们可以人为对回溯次数进行设置, 但软件默认为 10。如果在对某个故障进行测试产生过程中回溯次数大于 10, 则将该故障加入到中止故障列表中。

在建立测试向量列表后, ATLANTA 软件使用 FSIM 算法和 HOPE 算法对故障列表中的固定型故障进行故障模拟。在此阶段, ATALANTA 在将测试产生的向量集向后传播至主输出端时, 将检测到的故障从故障列表删除, 缩短测试产生的时间。最后, ATALANTA 软件会对测试向量进行压缩, 并

将每个测试向量检测到的固定型故障数量、识别出的冗余故障、因为回溯次数得到的中止故障和故障覆盖率等测试信息写入到 `*.bench` 电路对应的 `*.test` 与 `*.vec` 文件中。如果需要对电路更详细的测试信息, 可以使用 `./atalanta -l *.log *.bench` 命令生成测试电路的日志文件 `*.log`, 方便使用者了解整个电路的测试信息。

3 基于 LUT 的 ATPG 算法

3.1 LUT 的表示

使用 FPGA 技术模拟实现具有一定功能的电路时, 使用 LUT 技术构造模拟一个具体功能的简单逻辑函数嵌入在设计的电路中。LUT 的原理是, 当输入传到 LUT 时, 通过地址查找将对应的输出值寻址输出在 LUT 模块的输出端, 然后再继续往下一模块传播, 直至整个电路的主输出处。加入 2 输入或门功能的简单 LUT 模块的组合电路如图 1 所示。

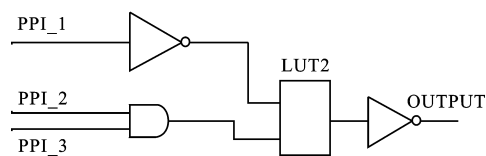


图 1 两输入 LUT 电路实例

图 1 电路中, 2 输入 LUT 逻辑模块用合取范式表示为 $LUT_PPI_1 \vee LUT_PPI_2$, 具体表现为当输入 PPI_1 经过反向器后, 将值传递到 2 输入 LUT 模块的第一个输入 LUT_PPI_1 , 而输入 PPI_2 和 PPI_3 经过逻辑与门后将值传递到 2 输入 LUT 模块的第二个输入 LUT_PPI_2 , 然后 LUT_PPI_1 与 LUT_PPI_2 经过 LUT 模块向后传输至 LUT 模块输出处, 最终经过反向器传递到整个电路主输出处。

LUT 一般是 4 输入结构的功能函数, 为方便理解, 构造了一个 2 输入或门的 LUT 真值表, 下表中 I_0 与 I_1 分别代表 2 输入或门 LUT 的两个主输入, O 代表 2 输入或门 LUT 的主输出, 而表中的具体数值分别代表 2 输入或门所有的输入与其对应的主输出的逻辑关系, 其输入与输出对应情况如图 2 所示。

根据图 2 中的对照表, 可以得出此 LUT 模块当两个主输入为 00 时, LUT 模块输出为 0; 当输入为 01、10 和 11 时, LUT 模块输出为 1。并且, 最后将得到的输出值进行移位运算得到 $4'b1110$ 。其中 4 代表输入向量数量, b 代表二进制。在对 LUT 模块

I1	I0	O
0	0	0
0	1	1
1	0	1
1	1	1

图 2 两输入或门的输入输出对照表

进行测试验证时,将 LUT1110 写入到 bench 电路中,并根据写入的对照关系就可以对 LUT 模块进行测试产生. 根据上述分析,定义了扩展的 bench 电路格式如下所示.

```
# c19
# 4 inputs
# 1 outputs
# 5 gates
INPUT(PPL_1)
INPUT(PPL_2)
INPUT(PPL_3)
INPUT(PPL_4)
OUTPUT(b1)
a1=AND(PPL_1,PPL_3)
a2=AND(PPL_1,PPL_2)
a3= LUT1110(PPL_1,PPL_2,PPL_3,PPL_4)
a4=AND(a1,a2)
b1=AND(a1,a2,a3,a4)
```

如上所示 bench 电路,在进行测试验证含有 LUT 模块的组合电路时,对于基本模块,我们只读入设置的 AND 等逻辑符号;而对于 LUT 模块,因只考虑 LUT 模块输入和输出对应关系并且只在 LUT 模块输入和输出处进行固定型故障设定,针对 LUT 模块输入和输出关系做如下处理: n 输入 1 输出 LUT 模块输入共有 2^n 种,而输出只有 0 和 1 两种. 我们设定 LUT 模块第 i 个输入 P_i ,输入为 1 时,记为 P_i ;此输入为 0 时,记为 $\neg P_i$;然后对所有输入实行与运算得到此输入对应的输出,即 $f_o(i) = (P_i)_1(\neg P_j)_1$,其中 i 表示输入为 1 的输入数量, j 表示为 0 的输入数量,最后输出 $f_o = (f_o(i))_i^n$,且 $i + j = n$. 同理,读入 LUT 模块时,根据 LUT 后面的二进制值寻找到输入与输出的对应关系,如根据 LUT1110 后第一个 1 可以得到两个输入关系 $I1 \wedge I0$,同理最后可得 $f_o = (I1 \wedge I0) \vee (I1 \wedge \neg I0) \vee (\neg I1 \wedge I0) \vee (\neg I1 \wedge \neg I0)$. 在对电路中 LUT 模块进行测试验证时,向前测试产生输入向量和向后传播寻找输入向量可根据此公式进行输入与输出对应关系的查找.

3.2 基于 LUT 的 ATPG 算法

一般而言,门级 FPGA 设计描述是混合了基本逻辑门和 LUT 模块的电路,本文研究混合二者的组合电路模块,采用 ATPG 进行测试验证时,可以把 LUT 模块理解为一个黑盒,它里面包含有多个基本逻辑门,我们关心的只是它的输入与输出以及两者的对应关系和故障点的设置,而不用去关心 LUT 模块里面的具体实现细节. 使用 ATPG 算法对 LUT 模块设置单固定故障时,就像对组合电路中的逻辑门设置故障时一样,只针对 LUT 模块的输入和输出设置 LUT 模块的故障点;在测试验证阶段,将向量向前敏化时,只需到写入的 SRAM 中根据 LUT 模块的主输出去寻找其对应的主输入;同样地,在将测试向量向后传播时,我们只需根据写入到 SRAM 中的输入与输出的对应关系去寻找对应的主输出赋值在 LUT 模块的主输出端. 在整个的测试产生过程忽略 LUT 模块的内部实现细节.

使用 ATPG 算法对包含有基本逻辑门和 LUT 模块的功能电路验证流程如图 3 所示.

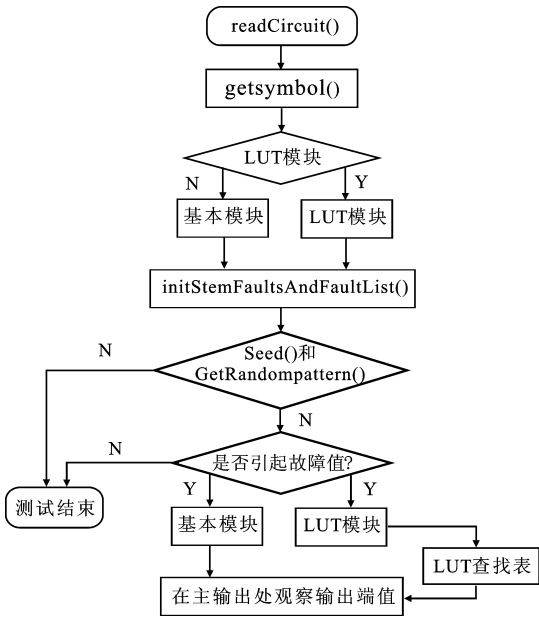


图 3 基于 LUT 的 ATPG 算法流程图

ATALANTA 软件对含有 LUT 模块的组合电路进行验证时,函数 readCircuit()用来添加逻辑门符号、计算逻辑门主输入和主输出的数量、构造电路的基本结构,同时检查 bench 电路的格式是否符合要求;getsymbol()函数读取 bench 文件中的基本逻辑门符号并识别出 LUT 模块的符号 LUT;initStemFaultsAndFaultList()用来对电路中的基本逻辑门和 LUT 模块的输入和输出线设置 s-a-0 或

s-a-1 故障;函数 Seed()和 GetRandompattern()用来生成随机测试向量,建立测试向量空间;函数 initFaultSim_HOPE()用来初始化进行故障模拟时所需电路参数信息,而 injectFault()和 Faulty_Gate_Eval()用来对电路进行故障模拟,在故障模拟过程中,使用函数 gate_eval1()通过查找基本逻辑门的真值表来寻找输入对应的输出,对于 LUT 模块,不是在真值表而是到查找表中来寻找输入对应的输出;函数 print_atpg_head()在 Linux 终端输出 bench 电路的测试信息,如故障覆盖率、中止故障和总故障等故障数量和所用时间等信息.

扩展了 ATALANTA 软件,实现了 LUT 做为基本门级单元的输入和解析,在测试时对于基本逻辑门的输入与输出的对应关系,软件会自动到真值表中查找基本逻辑门的输入与输出的逻辑关系.当软件在测试验证 LUT 模块时,会到查找表中寻找输入与输出的对应关系,而忽略 LUT 模块中的具体实现细节.再将寻找到的 LUT 模块的输入向量向前敏化至电路的主输入处,然后将寻找的引起故障值的主输入处的向量向后传播,当经过 LUT 模块时,会根据输入向量寻找 LUT 模块的输出值,并将此值传播至电路的主输出处.然后在主输出处观察设置的故障点的表现值,并与应用相同输入向量的规范电路的输出值进行比较,验证实现电路和规范电路是否在逻辑功能上等价.

4 测试

ATALANTA 软件对带有 LUT 模块的 FPGA 设计进行测试时,为了缩短组合电路的测试产生时间,我们设置回溯次数为 10,如果在验证测试时,对一个单固定型故障的回溯的次数超过 10 次,则停止对该故障进行测试产生,将该故障在故障列表中删除并加入到中止故障列表中,然后继续从故障列表中选择未测试的故障继续进行测试,当该故障产生测试向量时,会将该向量沿着电路向后传播至整个电路的主输出处,也即故障模拟阶段,在故障模拟阶段若检测到故障列表中存在的故障,则将故障模拟阶段检测到的故障从故障列表删除,并将该向量认为是故障模拟阶段检测到的故障的测试产生向量.这个循环过程在整个测试阶段一直执行,直到将故障列表中的故障全部测试完.

ISCAS'85 由 M. Hansen, H. Yalcin, and J. P. Hayes 提出,并且被证明是在时序分析和测试生成等数字设计领域非常有效的测试集.为了方便

进行测试,本文选取了 ISCAS85 中的 10 个组合电路,并对每个组合电路随机加入 LUT 模块进行测试.下表是对加入 1 个 LUT 模块的 10 个组合电路使用改进的可以识别 LUT 模块的 ATALANTA 软件进行测试的结果,加入的 LUT 模块如下:168=LUT0010000100111111(输入 1,输入 2,输入 3,输入 4),结果如表 1 所示.

表 1 加入 LUT 模块前后故障覆盖率对比

电 路 名 称	加入 LUT 模块 前故障覆盖率/%	加入 1 个 LUT 模 块后故障覆盖率/%
c432	99.046	97.378
c499	96.570	95.698
c880	100.00	98.948
c1355	99.492	98.926
c1908	99.468	98.782
c2670	95.741	95.369
c3540	96.004	94.734
c5315	98.879	98.862
c6288	99.251	99.471
c7552	98.159	97.991

为了验证组合电路中加入 LUT 模块的效果,在之前加入一个 LUT 模块的基础上,对每个组合电路再加入与之前功能不同的另一个 LUT 模块,但是每个组合电路加入的两个 LUT 模块功能是相同的,也即每个组合电路中加入两个 LUT 模块,下表是对加入两个 LUT 模块的组合电路进行验证的结果,并与只加入 1 个功能不同 LUT 模块的组合电路的故障覆盖率进行对比,加入的第二个 LUT 模块如下:1689=LUT1110000100111111 输入 1,输入 2,输入 3,输入 4),结果如表 2 所示.

表 2 加入不同数量 LUT 模块故障覆盖率对比

电 路 名 称	加入 1 个 LUT 模块 前故障覆盖率/%	加入 2 个 LUT 模 块后故障覆盖率/%
c432	97.378	93.554
c499	95.698	94.974
c880	98.948	97.188
c1355	98.926	98.116
c1908	98.782	98.419
c2670	95.369	95.142
c3540	94.734	94.515
c5315	98.862	98.736
c6288	99.471	99.394
c7552	97.991	97.780

通过将上述两张表的结果进行对比,可以看出在将 LUT 技术引入到组合电路中进行测试时,加入 LUT 技术的组合电路的故障覆盖率稍微低于没加入 LUT 模块时的故障覆盖率;同样地,加入两个 LUT 模块的组合电路故障覆盖率略微低于加入 1 个 LUT 模块时组合电路的故障覆盖率,这是因为加入 LUT 模块后尤其是加入 LUT 模块的数量增多时,组合电路中总的故障数量也比之前增多,并且在回溯数量受到限制时,由于电路复杂度增加,对单固定型故障进行测试产生时,需要回溯的次数也再增加,中止的故障数量也在不断变化,因此加入 LUT 模块后故障覆盖率会有所变化.但是总体上,加入 LUT 模块后,对组合电路进行测试诊断时故障覆盖率仍在 90%以上,可以检测到其他类型的故障,因此在总体上是满足要求的.

5 结束语

针对 FPGA 的设计,本文提出了将 LUT 引入 ATPG 算法的技术,并将这种技术整合到 ATALANTA 开源软件中,对包含了 LUT 的组合电路进行测试验证时,这种改进型的 ATALANTA 软件完全达到预期目标,可以检测出各类 LUT 的故障.因此,这种直接操作 LUT 的 ATPG 方法,不仅减少了设计和测试验证人员的工作量,也极大地减少了对待测试电路的测试时间.

参考文献:

[1] 何曦. 基于优化 ATPG 的可测试性设计与实现[D]. 成都:电子科技大学,2016.

[2] 曾文斌. 组合逻辑电路的测试故障模型[J]. 计算机学报,1990,12(7):529-538.

[3] 周宁. 集成电路等价性验证系统设计及其实现[D]. 兰州:兰州大学,2007:33-42.

[4] 岳园. 基于逻辑锥和 SAT 的带黑盒电路等价性验证方法[D]. 兰州:兰州大学,2007:19-31.

[5] Lee H K, Ha D S, On the generation of test patterns for combinational circuits[R]. USA:Dept of Electrical Eng. , Virginia Polytechnic Institute and State University. Technical Report No. 12_93,2002.

[6] 范逸鸣. 基于 LUT 的 FPGA 时序仿真的设计与实现[D]. 西安:西安电子科技大学,2012:7-18.

[7] Hansen M, Yalcin H, Hayes J P. Unveiling the ISCAS-85 benchmarks: a case study in reverse engineering[J]. IEEE Design and Test, 1999, 16(3): 72-80.

[8] 杨世元,胡梅,王红. 模拟电路软故障诊断的研究[J]. 微电子学与计算机,2008,25(1):1-6.

作者简介:

王志远 男,(1989-),硕士研究生. 研究方向为等价性验证算法研究. E-mail:wangzhiyuan14@lzu.edu.cn.

李彩虹 女,(1973-),博士,高级实验师. 研究方向为形式化验证、电网预测、机器学习、无线传感器网络等.

何安平 男,(1981-),博士,副教授. 研究方向为模型检测、等价性验证、异步电路系统相对时序分析等.

(上接第 43 页)

[6] Shi Y H, Eberhart R C. A modified particle swarm optimizer[C]//IEEE International Conference on Evolutionary Computation. Anchorage,1998:69-73.

[7] 陶海龙,李小平,张胜召,等. 基于 IPSO-BP 神经网络的铁路客运量预测[J]. 铁道运输与经济,2011,33(9): 78-82.

[8] 高岳林,任子晖. 带有变异算子的自适应粒子群优化算法[J]. 计算机工程与应用,2007,43(25):43-47.

[9] 王青,伍书剑,李明树. 软件缺陷预测技术[J]. 软件学报,2008,19(7):1565-1580.

[10] 余安雷,皮特常. 基于 PSO-BP 的软件缺陷预测模型[J]. 计算机工程与应用,2013,49(7):64-67.

作者简介:

邓伟康 男,(1990-),硕士研究生. 研究方向为大数据与云计算.

刘 锋 男,(1962-),教授,硕士生导师. 研究方向为并行计算与云计算.

朱二周(通讯作者) 男,(1981-),副教授,硕士生导师. 研究方向为虚拟化与程序分析. E-mail:ezzhu@ahu.edu.cn.