

基于穷举和回溯的自反馈测试生成算法的实现

杨 鹏, 邝继顺

(湖南大学 计算机与通信学院, 长沙 410082)

摘 要: 提出了一种基于穷举和回溯的自反馈测试生成算法, 并在 Xilinx Virtex - II Pro 开发环境下实现了测试生成算法. 穷举和回溯算法用软件设计实现. 算法中状态矩阵的计算和检测用硬件设计实现. 系统的整体设计在型号为 XC2VP30 的实验板上进行了验证.

关键词: 内建自测试; 自反馈测试; 穷举; 回溯; 片上可编程系统

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2011)03-0094-05

Realization of Self-feedback Test Generation Algorithm Based on Exhaustion and Backtracking

YANG Peng, KUANG Ji-shun

(College of Computer and Communication, Hunan University, Changsha 410082, China)

Abstract: This article proposed a self-feedback test generation algorithm based on the exhaustion and backtracking, and realized it in the development environment of Xilinx Virtex-II Pro. The exhaustion and the backtracking algorithms were realized with the software, while the computation and the examination of state matrix in the algorithm were realized with the hardware. The system was validated on the XC2VP30 experiment board.

Key words: build-in self-test; self-feedback test; exhaustion; backtracking; SOPC

1 引言

传统的内建自测试^[1] (Build-In Self-Test, BIST) 方法一般是采用线性反馈移位寄存器 (Linear Feedback Shift Register, LFSR) 减少测试数据存储量, 它并没有考虑被测电路本身的资源. 被测电路除了输入输出端口以外, 其内部节点也是一种可以利用的资源. 自反馈测试^[2] 就是在考虑如何充分利用这些内部节点的情况下, 一种新颖的内建自测试方法.

自反馈测试方法将内部节点视为可利用的资源, 通过捕获电路内部对应当前测试向量的部分节点响应, 反馈到原始输入端得到下一个测试向量, 使得电路在施加了初始向量后, 可以自动的生成一个完整的测试集. 该方法的核心是寻找反馈方式, 反馈

方式决定由被测电路自动生成的测试集.

在不给定测试集的情况下, 文献[3]和文献[4]分别研究了用软件和硬件实现的基于遗传算法的测试生成方法. 遗传算法本身是一种近似解法, 不能精确找到最优的反馈方式. 为了得到精确的解集, 文中提出了一种基于穷举和回溯的自反馈测试生成算法. 由于该算法的计算量较大, 文中利用硬件具有并行性的特点, 设计了一种基于片上可编程系统^[5] (System on a Programmable Chip, SOPC) 的测试生成系统, 可以降低算法的时间复杂度且提高系统的设计灵活性及性能.

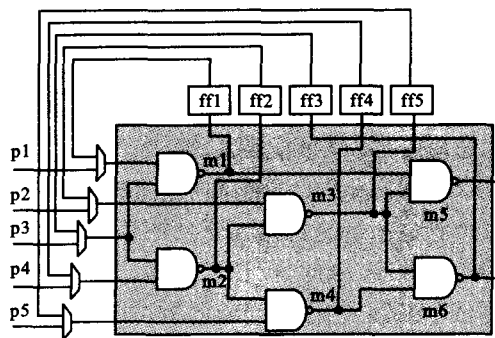
2 自反馈测试方法

下面以 C17 电路为例介绍自反馈测试方法的基本原理. 图 1(a)给出了一种 C17 电路的自反馈方式测试结构. C17 电路的自反馈测试结构包括 C17

电路、5个触发器和5个二选一的多路开关这三个部分。触发器捕获的内部节点响应值组成被测电路的下一个测试向量。这种测试结构通过控制多路开关的选择端,分别把种子向量和由内部节点产生的测试向量送给被测电路的原始输入端。首先从6个内部节点中找出5个节点,然后这5个节点按照合适的顺序依次反馈到电路的原始输入端。在种子的作用下,由这5个内部节点的响应值组成第二个测试向量施加到电路原始输入端,然后再次取出这5个节点的新的响应值组成第三个测试向量,依此类推。那么,要解决的主要问题是寻找这5个内部节点,以及按照什么样的排列顺序把它们施加到电路的原始输入端,才能产生更多更有效的测试向量。如果用穷举法选择抽点自反馈方式,共有720种选择。

已知C17电路的某一测试集为 $T = (t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7) = (00000, 11011, 11100, 01110, 10011, 11101, 01100, 11110)$ 。测试集 T 可以检测所有固定型故障。图1(b)给出了每个测试向量及其节点响应的对应关系,其上半部分是测试向量,下半部分是对应向量的内部节点响应,箭头表明了测试向量与节点响应一一对应的关系。按照图1(a)的反馈方式,如果把向量(00000)作为初始种子加载到电路的原始输入端,则电路内部节点 m_1, m_2, m_6, m_4 和 m_3 的值分别是1、1、0、1、1。也就是说,(11011)将作为下一个向量输入到原始输入端。依照这种反馈方式,在施加种子向量(00000)后,图1(a)的测试结构依次产生的向量分别为(11011)、(11100)、(01110)、(10011)、(11101)、(01100)、(11110)。综上可知,电路在施加了第一个测试向量 t_0 后,由5个触发器分别从电路内部节点(m_1, m_2, m_6, m_4, m_3)捕获的响应值正好组成 t_1 ,然后通过多路开关可以将触发器的值作为 t_1 施加给C17的输入端。可以看到,在 t_1 被自动施加后,用同样的触发器组合方式可以依次得到 t_2 到 t_7 的各个测试向量。这样这种结构只需存储 t_0 ,而不必存储整个测试集,从而可以减少存储测试向量的硬件代价。

自反馈测试方法的核心思想是在给定测试集的情况下,寻找一个适当的测试向量加载顺序和被测电路中的一组内部节点,即寻找一种反馈方式,使得从外部对被测电路施加了第一个种子向量后,通过反馈能够产生并施加这个测试集中的其它向量。为了得到这个适当的测试向量加载顺序,文中采取一种穷举测试向量组合的方法进行求解。



(a) 自反馈测试方法的测试结构

t0	t1	t2	t3	t4	t5	t6	t7	
0	1	1	0	1	1	0	1	p1
0	1	1	1	0	1	1	1	p2
0	0	1	1	0	1	1	1	p3
0	1	0	1	1	0	0	1	p4
0	1	0	0	1	1	0	0	p5

m1	m2	m3	m4	m5	m6	s0	s1	s2	s3	s4	s5	s6	s7
1	1	0	1	1	0	1	0						
1	1	1	0	1	1	1	1	0					
1	0	0	1	1	0	0	0	1					
1	0	1	1	0	0	1	1						
0	1	1	0	0	1	1	1						
0	1	1	0	1	1	1	1	0					

(b) 测试向量和测量响应

图1 C17电路的一种自反馈测试结构与测试向量和测试响应

3 自反馈测试生成算法

3.1 穷举

穷举测试向量组合就是把被测电路的测试向量以各种顺序进行排列,因为不是每种顺序的队列都是所求的解,所以需对每种顺序加以判断才能得到满足要求的解。判断排列的每种顺序是否满足要求,需从测试向量和节点响应入手。图1(b)给出了C17电路的每个测试向量及其节点响应的对应关系。

3.1.1 问题描述

假设被测电路的测试集为 $T\{t_0, t_1, t_2, \dots, t_{n-1}\}$, 对应的内部节点响应集为 $S\{s_0, s_1, s_2, \dots, s_{n-1}\}$ 。被测电路的节点个数为 M , 输入个数为 L 。算法定义一个 $M \times L$ 的状态矩阵 H , 它是一个只包含0、1元素的矩阵, 其行和列分别对应电路的内部节点和输入。 H 中的元素1表示对应的节点能够反馈给对应的输入, 元素0则表示对应的节点不能够反馈给对应的输入。例如, $H_{ij} = 1$, 表示节点 m_i 能够反馈给输入 p_j ; $H_{ij} = 0$, 则节点 m_i 不能够反馈给输入 p_j 。另外,

算法定义种子向量对应的状态矩阵是元素为全 1 的矩阵。

对测试向量进行排列时,测试向量的排列顺序通过状态矩阵加以判断。那么,算法中的一个重要问题就是求解状态矩阵。对测试向量进行排列时,可以发现每个测试向量能否排入队列依赖于它之前测试向量的排列顺序。因此,计算当前测试向量的状态矩阵需要三个参数,它们是当前的测试向量、前一个测试向量对应的节点响应和前一个测试向量对应的状态矩阵。

3.1.2 状态矩阵

假设有一个三输入和三内部节点的电路,测试向量 t 为(011),前一个测试向量对应的内部节点响应 s 为(101),前一个测试向量对应的状态矩阵记为 H_0 。计算测试向量 t 对应的状态矩阵 H 的过程如下: t 与 s 先进行计算,其结果再与 H_0 进行计算。计算时,分别把 t 的每一位和 s 的每一位进行同或操作,其结果再和 H_0 的每一位进行与操作,这样就得到 t 对应的状态矩阵 H 。图 2 为状态矩阵求解示例。

$$\begin{matrix} & 0 & 1 & 1 \\ \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} & \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} & \Rightarrow & \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} & \cdot & \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}_{H_0} & \Rightarrow & \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}_H \end{matrix}$$

图 2 状态矩阵求解示例

状态矩阵的列对应电路的输入。如果出现某列为全 0,则表示没有任何一个节点可以反馈到这列对应的输入。因此,根据 H 来判断 t 的顺序。如果 H 中只要出现某列为全 0,则 t 不可以加载到这个顺序中;否则, t 可以加载到这个顺序中。

由图 2 可知,状态矩阵的计算可描述为公式(1):

$$H_{ij} = (t_j \odot s_i) \cdot H_{0ij} \quad (1)$$

其中, t_j 是测试向量 t 的第 j 位, s_i 是前一个测试向量对应的节点响应 s 的第 i 位, H_{0ij} 是前一个测试向量对应的状态矩阵的第 i 行第 j 列的元素, H_{ij} 是状态矩阵的第 i 行第 j 列的元素。

公式(1)通过转换可写成公式(2):

$$\begin{cases} H_{ij} = s_i \cdot H_{0ij}, & t_j = 1 \\ H_{ij} = \bar{s}_i \cdot H_{0ij}, & t_j = 0 \end{cases} \quad (2)$$

按照公式(2)计算状态矩阵,如果 t 的某一位为 1,那么 s 和 H_0 对应的那一行进行按位与操作;如果 t 的某一位为 0,那么把 s 按位取反后,再和 H_0 对应的那一行进行按位与操作。

3.1.3 算法描述

穷举算法给每个测试向量定义一个序号,序号的起始值为 0;一位标志位,0 表示对应测试向量未排入队列,1 表示对应测试向量已排入队列。算法定义参数 n 为测试向量个数,定义数组 $dl[i]$ 用于存储测试向量的序号,定义数组 $flag[i]$ 用于存储测试向量的标志位信息,定义数组 $H[i]$ 用于存储测试向量的状态矩阵,定义数组 $T'[i]$ 用于存储能够排入当前位置的测试向量的序号,这里 $i=0,1,\dots,n-1$ 。算法的输入 m 为队列中穷举操作的起始位置。根据以上分析,给出穷举算法如下:

```
exhaustive (T,S,m,n)
begin
  for p←m to n-1 do
    T'←search(flag)
    for i←0 to n-1-p do
      t←T[T'[i]]; s←S[dl[p-1]]
      H←cal_state_matrix(t,s,H[p-1])
      test←detect_state_matrix(H)
      if test=1 break
    end if
  end for
  if test=0 break
  else flag[T'[i]]←1; dl[p]←[i]; H[p]←H
  end if
end for
end
```

上述算法中,search 函数通过对测试向量标志位的判断,寻找出能够排入当前位置的测试向量的序号。cal_state_matrix 和 detect_state_matrix 函数分别是计算状态矩阵函数和检测状态矩阵函数,它们在 IP 核中设计实现。

3.2 回溯

从组合原理可知,如果测试向量的个数为 n ,队列的组合数最多就有 $n!$ 种。找队列的思想是队列中每个位置按序号从小到大尝试测试向量,保证队列中的每个位置都能包含每个测试向量,从而保证遍历 $n!$ 种测试向量的组合。测试生成算法需要找出所有满足要求的队列,且需要遍历 $n!$ 种测试向量的组合,当某种排列不成功时,必然会涉及到对排列的回溯。

回溯算法给定一个虚拟序号,其值为 n ,作为回溯的标志和整个算法的结束标志。算法定义参数 Q 为回溯位置的标号,定义回溯位置的测试向量 t 的

序号为 j . 即当队列尝试到第 Q 号的位置 ($Q < n$), 且其测试向量的序号为 j , 其后不能排入测试向量, 则回溯测试向量 t . 根据以上分析, 给出回溯算法如下:

```

backtrack (Q,j)
begin
while(Q<n)
begin
p←Q
flag[j]←0
if p=0 dl[0]←j+1; flag[j+1]←1
else
for i←j+1 to n-1 do
if flag[i]=1 continue;
else t←T[i]; s←S[dl[p-1]];
H←cal_state_matrix (t,s,H[p-1])
test←detect_state_matrix(H)
if test=1 flag[i]←1; dl[p]←i; H[p]←H
end if
end if
if test=1 break; end if
end for
end if
if dl[0]= n break; end if
if dl[p]! = j exhaustive (T,S,p+1,n)
end if
Q←p-1
end
end

```

4 算法的实现

本系统在 Xilinx Virtex - II Pro 开发环境下选用 FPGA 芯片 XC2VP30 实现算法. 该芯片内嵌了 2 个采用高性能 RISC 技术、频率高达 400MHz 的 PowerPC 405 内核. PowerPC 405 是专门为嵌入式应用而设计的高性能 32 位 PowerPC 系列处理器芯核. 存储器资源丰富, 其中块 RAM(BRAM)容量为 2448KB, 分布式 RAM 容量为 428KB. 开发采用 Xilinx 公司的 EDK10.1 (嵌入式开发套件) 和 ISE10.1 (集成软件环境) 工具软件.

系统框图如图 3 所示. PowerPC 处理器是此嵌入式系统的核心, 它通过 PLB 总线实现与外围设备的通信. 软件部分的开发主要是用 C 语言编程实现穷举和回溯算法, 在处理器上运行程序. PLB 总线

是处理器内部总线, 它在处理器和高性能外围设备之间提供高速接口. 系统的程序存储器和数据存储器都放在片内, 用 BRAM 来实现. 硬件部分的开发主要是用 Verilog HDL 语言设计实现状态矩阵的计算和对矩阵信息的检测, 在用户 IP 上设计实现. IP 核^[6]和内存之间的数据通信方式选择直接存储器访问 (Direct Memory Access, DMA) 模式. DMA 模式的数据传输通过一个 DMA 控制器实现. RS232 串口设备输出系统实验结果.

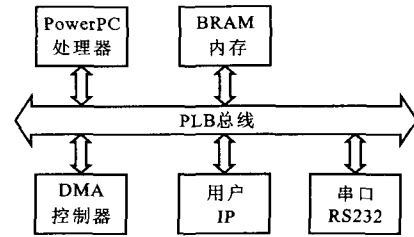


图3 系统框图

软硬件通信模式需要 DMA 控制器的支持, 其工作原理如下:

(1) CPU 在 DMA 控制器中装入必要的输入输出控制数据, 例如传输方式、IP 核所需数据的内存起始地址、IP 核名称和成组传输字节数等.

(2) 启动 DMA 控制器和 IP 核.

(3) DMA 控制器负责将数据从内存传送到 IP 核.

(4) IP 核获得数据并计算. IP 核执行完后, 向 CPU 发起中断, 通知 CPU 计算已完成, 并将结果发送给 DMA.

(5) CPU 将结果要写入的内存地址告诉 DMA 控制器, DMA 控制器将结果写入内存.

系统设计的 IP 核是实现状态矩阵计算和矩阵信息检测的功能. 根据公式 (2), 基本运算单元可以设计成如图 4 所示. 运算单元有三个输入端和一个输出端, 其中输入端 t_j 作为控制端, 对应测试向量的每一位; 输入端 s_i 对应节点响应的每一位; 输入端 H_{0ij} 对应前一个测试向量的状态矩阵元素; 输出端 H_{ij} 对应当前判断的测试向量的状态矩阵元素. 因为状态矩阵的列对应被测电路的输入, 且 t_j 作为一列的控制位, 所以按每列进行计算较容易实现. 按每列进行计算时, 因为 IP 核的输入 s_i 、 H_0 和输出 H 三者具有 M (节点个数) 位, 所以系统可以设计实现一个包含 M 个基本运算单元的 IP 核, 它能同时计算 M 位数据. 矩阵信息的检测用一个比较器实现. 比较器的一个输入端固定为零值, 另一个输入端为计

算所得的数据,其输出为系统的测试值. 它的功能是检测状态矩阵计算得到的这一列是否为全 0,是,则返回 0 值;否,则返回 1 值.

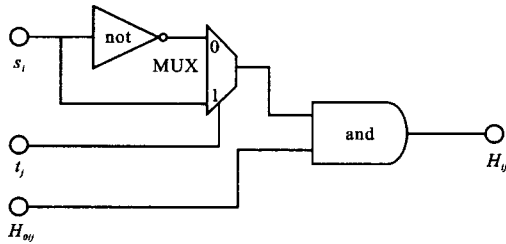


图 4 IP 核的基本运算单元结构

整个设计在嵌入式开发工具 EDK 的平台上进行开发与验证. 为了验证系统和测试系统的整体性能,文中用 ISCAS85 Benchmark 中的 4 个标准电路对系统进行了测试,并在 Xilinx Virtex - II Pro 系列嵌入式开发板上进行系统验证实验. 实验结果通过开发板的串口输出,并且通过串口调试助手进行观察.

本实验用 C 语言实现算法,用 Verilog HDL 语言设计 IP 核. 实验板选择 Virtex - II Pro 系列中型号为 XC2VP30 的器件,其处理器频率为 300MHz,总线频率为 100MHz. 表 1 以 C432 电路为例,列出 IP 核的器件使用情况的综合报告. 可以看出,文中设计的 IP 核占用的系统资源是比较少的.

表 1 IP 核的系统综合报告

所选器件	2vp30ff896-7
硅片数目	1066 out of 13696 7%
硅片触发器数目	1190 out of 27392 4%
四输入查找表数目	1294 out of 27392 4%
最大时钟频率	170.237MHz

表 2 为 4 个标准电路的实验结果. 对比 C6288、C880 和 C432 这三个电路可知,系统运行时间总体上随测试向量数的增加而增加. 这是因为要遍历的组合最多有 $n!$ (n 为测试向量数)种,所以测试向量数是影响时间效率的主要因素. 对比 C6288 和 C880 这两个电路可知,C880 的系统运行时间比 C6288 少. 这是因为 C6288 的测试向量数虽然比 C880 少,但 C6288 的节点数却远多于 C880,使得矩

阵计算的规模较大,所以被测电路的输入数和节点数也是影响时间效率的因素.

表 2 测试结果表

电路	测试向量个数	输入个数	节点个数	测试向量最长组合数	最长组合个数	系统运行时间
C17	8	5	6	8	1	1.5s
C6288	12	32	2448	10	76507	8.14h
C880	16	60	469	9	3	7.32h
C432	27	36	311	8	3117	9.33h

5 结束语

文中设计了一种基于 SOPC 的测试生成系统,实现了一种基于穷举和回溯的自反馈测试生成算法. 系统实现了一种用软件设计算法和用硬件设计 IP 核做算法中数据处理的嵌入式系统结构. 实验结果表明,文中方法能够准确地找到被测电路的所有测试向量加载顺序,即能找到被测电路的所有反馈方式的解集.

参考文献:

[1] Chen Chien In Henry. Efficient approaches to low-cost high-fault coverage VLSI BIST designs[J]. IEEE Transactions on Aerospace and Electronic Systems, 1998:63-70.

[2] 邝继顺,刘钦,尤志强. 由被测电路自己产生测试向量的自动测试生成[C]// 第五届中国测试学术会议论文集. 苏州:2008:30-35.

[3] 肖媛. 基于遗传算法的自反馈测试生成方法研究[D]. 长沙:湖南大学,2009.

[4] 钟汝刚. 基于遗传算法的测试生成在 FPGA 上的研究与实现[D]. 长沙:湖南大学,2009.

[5] 胡建萍,崔晓松,潘松,等. 基于 SOPC 技术的导航解算系统[J]. 微电子学与计算机,2008, 25(6): 218-221.

[6] 付华杰,刘丽君,张遂南. 高速 USB IP 核的设计与开发[J]. 微电子学与计算机,2008, 25(7): 127-129.

作者简介:

杨 鹏 男,(1986-),硕士研究生. 研究方向为 SOC 测试与设计.

邝继顺 男,(1959-),教授,博士生导师. 研究方向为集成电路设计与测试.