

# 融合编译功能的组合电路测试集生成方式

李天义<sup>1</sup>, 富 坤<sup>1</sup>, 耿跃华<sup>2</sup>, 徐 丹<sup>1</sup>

(1. 河北工业大学 计算机科学与软件学院, 天津 300401; 2. 河北工业大学 电气与自动化学院, 天津 300130)

**摘 要:** 通过研究一些组合电路测试仪器中测试集的生成方式, 提出一种融合了编译功能的组合电路测试集生成方式. 该测试集生成方式将编译原理与测试集生成算法相结合, 解决了组合电路测试仪器中被测芯片类型不易扩充、硬件电路设计复杂等难点问题. 从被测芯片的功能表达式入手, 详细阐述其词法、语法、语义的分析过程, 并以临界通路法为例, 给出了该测试集生成方式的详细设计方案, 最后根据该设计方案设计出基于 STC89C52 芯片的组合电路测试系统, 并给出了该测试系统的性能测试结果.

**关键词:** 编译原理; 临界通路法; 组合电路; 测试集

**中图分类号:** TP314; TP39

**文献标志码:** A

**doi:**10.3969/j.issn.1671-6833.2014.02.015

## 0 引言

目前, 高校的数字电子实验教学大都以 TTL、CMOS 等标准的集成电路器件为基础<sup>[1]</sup>, 使用前必须判断集成芯片是否已经损坏, 以确保教学实验顺利进行, 为此, 各大高校和企业纷纷设计出各自的数字芯片检测装置, 以达到快速检测该错误的目的<sup>[2-4]</sup>. 经过研究发现, 这些数字芯片检测设备虽然有效地简化了芯片测试方式和步骤, 但仍有很多不足之处亟待解决. (1) 被测芯片数目固定. 各种检测设备中, 被测芯片种类受测试集的限制不能够动态的扩充. (2) 下位机设计电路复杂. 一些设备需要设计电路选择模块来选择要测试的芯片, 使得下位机电路的设计更为复杂, 增加了不必要的电路开销.

笔者从被测芯片测试集的生成方式入手, 采用编译原理<sup>[5-6]</sup>的基本思想与测试集生成算法相结合, 通过被测芯片的功能表达式直接动态生成测试集和结果集的生成方式(简称生成方式), 经程序验证该生成方式有效地解决了上述问题.

## 1 原理与逻辑结构

编译程序的主要功能是把用高级语言编写的

源程序翻译为等价的目标程序, 实现一种语言的翻译, 以便目标机器执行<sup>[6]</sup>. 利用编译程序的这一思想, 结合组合逻辑电路的测试集生成方法, 将编译功能的输入、产出与测试集生成方法的输入、产出进行等价比拟, 得出比拟对象为: 高级语言源程序比拟为组合逻辑电路功能表达式; 目标程序比拟为测试电路的测试集和结果集; 中间的运算变化过程即为笔者提出的融入编译功能的测试集生成方式.

该测试集生成方式的主体部分由文法分析过程和算法结合过程构成. 其逻辑结构如图 1 所示.

文法分析过程中, 功能表达式是按照一定规则输入的字符串; 词法分析程序利用识别功能表达式的有限自动机对功能表达式进行逐一扫描和分解<sup>[7]</sup>, 判断功能表达式是否含有非法字符(即未按照表达式书写规则书写的字符); 语法分析程序利用文法分析表对功能表达式进行分析, 判别其是否为完善的表达式, 并产生相应的归约动作序列<sup>[8]</sup>. 算法结合过程中, 根据对应文法产生式的函数, 利用归约动作序列生成相应表示芯片表达式功能的树(简称功能树); 通过对功能树进行分析、整理, 并结合相应的测试集生成算法生成相应芯片的测试集和结果集.

收稿日期: 2013-11-11; 修订日期: 2013-12-28

基金资助: 国家自然科学基金青年基金资助项目(31100711); 河北省高等学校科学技术研究青年基金项目(20111122).

作者简介: 李天义(1986-), 男, 河北衡水人, 河北工业大学研究生, 主要从事嵌入式系统及应用研究, E-mail: litianyi2060@163.com.

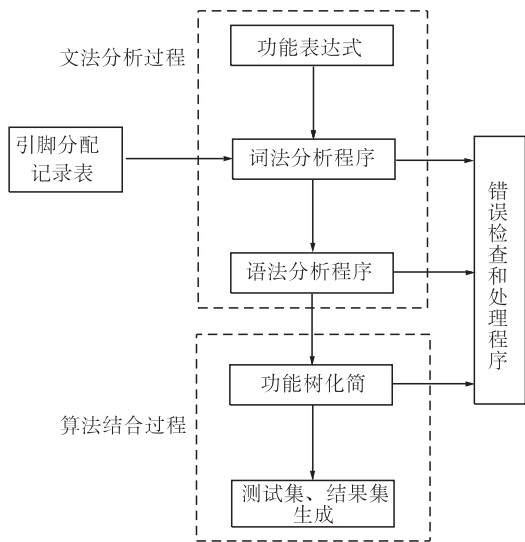


图1 生成方式的逻辑结构图

Fig.1 Logical architecture of the generation

## 2 文法分析过程

### 2.1 构造表示芯片功能的文法

在进行词法分析前首先对被测芯片功能表达式的书写方式进行统一规定。

(1) 芯片引脚名称由任意一位大写英文字母组成；

(2) 运算符有“\*”、“+”、“-”，分别表示“与”、“或”、“非”三种逻辑操作；

(3) 赋值符号为“=”；

(4) 语句终结符号为“；”；

(5) 另外，表达式中编译顺序是从左到右进行，为改变运算的优先级需使用“(”和“)”这两个优先运算符。

芯片功能表达式由上述 5 种基本语法单位组成，如表达式： $Y = -(A * B)$ 。

根据上述表达式的书写规则，构建出满足此规则的 SLR(1) 文法  $G[S']$ 。  $G[S']$  文法如下：

- |                              |                              |
|------------------------------|------------------------------|
| 1. $S' \rightarrow S$        | 6. $T \rightarrow P$         |
| 2. $S \rightarrow PIN = E$ ; | 7. $T \rightarrow - P$       |
| 3. $E \rightarrow T$         | 8. $P \rightarrow PIN$       |
| 4. $E \rightarrow E * T$     | 9. $P \rightarrow (E)$       |
| 5. $E \rightarrow E + T$     | 10. $PIN \rightarrow letter$ |

如上文法  $G[S']$  由 10 个产生式组成。式中终结符号为“letter、=、;、\*、+、-、(、)”；非终结符号为“ $S'$ 、 $S$ 、 $E$ 、 $T$ 、 $P$ ”；开始符号为“ $S'$ ”。终结符号 letter 代表字母 A 至字母 Z 的任一字母。

### 2.2 构造有限自动机

依据上述规定构造有限自动机，词法分析程

序通过有限自动机对功能表达式进行扫描，识别出芯片功能表达式中各个基本语法单位，排除未按照表达式书写规则书写的字符。有限自动机如图 2 所示。

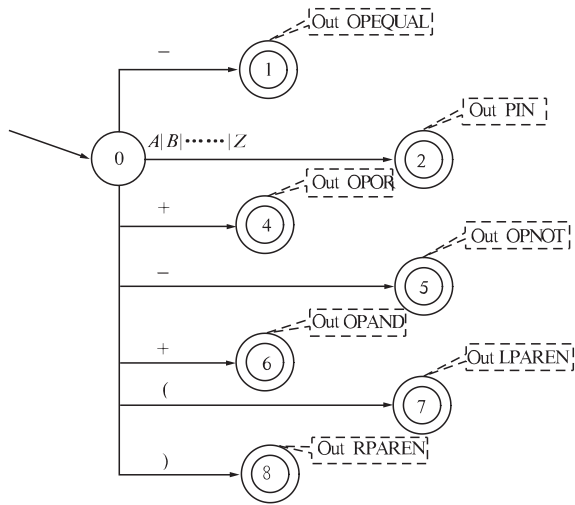


图2 功能表达式的有限自动机

Fig.2 Finite automation of the functional expression

扫描器从初态结点 0 出发，依次扫描功能表达式中的字符，对比有限自动机中的状态转换关系，寻找下一状态。进入终态结点时，表示扫描到基本语法单位。

### 2.3 构造文法分析表

编译功能中语法分析程序的目的是，判别功能表达式是否为完善的表达式，并依据对该功能表达式的归约动作序列，生成满足该表达式功能的树。

语法分析程序由功能表达式符号串、下推分析栈、总控制程序和分析表构成<sup>[9]</sup>。下推分析栈由状态信息  $S_0 \sim S_m$  来刻画；总控制程序依次扫描功能表达式中各个符号，根据当前下推分析栈的状态及正扫描的单词类型，按照分析表的指示完成相应的分析动作。语法分析程序的结构，如图 3 所示。

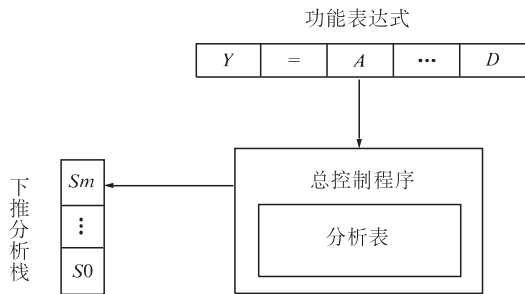


图3 语法分析的结构示意图

Fig.3 Sketch of the syntax analysis

语法分析程序的核心是构造一张分析表,根据文法分析表的构造原则,可构造 SLR(1) 文法  $G[S']$  的文法分析表. 它由两个子表构成:一是分析动作表(即 ACTION 表),另一个是状态转移表(即 GOTO 表). 分析表的构造和使用过程参考文献[8]和文献[9].

以表达式(1)为例,其语法分析过程,如表 1 所示. 在第 23 步,分析动作作为“ACC”,表示接受动作,即芯片的功能表达式没有文法错误. 将表 1 中的归约动作顺序排列可得到归约动作序列:“R10,R10,R8,R6,R3,R10,R8,R6,R4,R9,R7,R3,R2”.

表 1 表达式(1)的语法分析过程  
Tab.1 The syntax analysis process of the expression (1)

步骤	状态	栈中值	余留值	分析动作	下一状态
1	0	#	$Y = -(A * B); \#$	S3	3
2	03	#Y	$= -(A * B); \#$	R10	$GoTo[0, PIN] = 2$
3	02	#PIN	$= -(A * B); \#$	S4	4
...	...	...	...	...	...
21	0245	#PIN = E	;#	S11	11
22	0245 11	#PIN = E;	#	R2	$GoTo[0, S] = 1$
23	01	#S	#	ACC	

3 算法结合过程

目前,常用的测试集生成算法有临界通路法、布尔差分法、D 算法、FAN 算法等<sup>[10]</sup>,另外模拟电路的检测方法也提供了检测思路. 笔者以临界通路法为例说明表达式的功能树与测试集生成算法的结合过程.

3.1 生成表达式的功能树

在进行测试集生成算法结合前,需要根据归约动作序列生成表达式的功能树,以便于实现测试集生成算法.

表达式的功能树,由引脚名称符号、运算符符号和中间符号(中间符号是表示逻辑运算中间值的符号)构成. 在程序中,这些符号用 Note 结构表示. Note 结构中,用 Type 属性表征结点的类型,引脚名称符号用一位英文大写字母表示,运算符符号的“与、或、非”运算分别用“\*、+、-”表示,中间符号用一位英文小写字母表示.

```
struct Note {  
    int NoteID; // 结点标识  
    char Type; // 结点类型,如表示引脚的字母、  
    或运算符  
    bool Input; // 该节点是否是下一运算的输入值  
    int Follow1; // 指向下一节点  
    int Follow2; // " +、*" 运算符才有第二个下一节点  
} Notes[100];  
功能树的生成过程借鉴了编译原理中语义子
```

程序的思想,使不同产生式对应不同操作函数,依照在 2.3 中得到的归约序列,依次调用对应的操作函数,便可生成表达式的功能树. 产生式与操作函数的对应关系,如表 2 所示. 该方法的难点在如何确定各个操作函数的输入参数. 这里通过将操作函数的输入输出统一保存在 Notes 数组中来解决该问题. 若操作函数的输出结果中 Input 属性为 true,则作为下一操作函数的输入参与运算.

表 2 产生式与函数对应关系  
Tab.2 The Correspondence between production type and function

参数个数	产生式	函数名称
0	$PIN \rightarrow letter$	Note PIN_letter()
1	$S' \rightarrow S$	Note S_S(Note s)
	$E \rightarrow T$	Note E_T(Note t)
	$T \rightarrow P$	Note T_P(Note p)
	$P \rightarrow PIN$	Note P_PIN(Note pin)
	$P \rightarrow (E)$	Note P_E(Note E)
	$T \rightarrow -P$	NoteT_Not_P(Note p)
2	$S \rightarrow PIN = E;$	Note S_PIN_E(Note pin, Note e)
	$E \rightarrow E * T$	Note E_E_And_T(Note e, Note T)
	$E \rightarrow E + T$	Note E_E_Or_T(Note e, Note t)

为使功能树达到最简状态,Notes 数组中并不是每个结点都是最终需要的结点. 这里在操作函数中对输入参数进行处理,以达到简化的目的. 以函数 Note T\_Not\_P(Note p) 为例说明其处理流程.

(1)判断当前输入参数 p. Type 属性是否是引脚名称符号,同时,判断该输入参数 p. Follow1

属性所指向的结点的属性是否是运算符号;

(2)若均不是,则使用输入参数  $p$ . Follow1 属性所指向的结点,即  $\text{Notes}[p. \text{Follow1}]$  参与以后的赋值操作;

(3)否则,则使用输入参数参  $p$  与以后的赋值操作;

3.2 生成测试集和结果集

根据临界通路法中敏化立方 (Sensitized Cube) 的概念<sup>[10]</sup>,构造基础门电路敏化立方表,如图 4 所示.

$A$	$B$	$Y$	$A$	$B$	$Y$	$A$	$Y$
$1^c$	$1^c$	$1^c$	$0^c$	$0^c$	$0^c$	$1^c$	$0^c$
$0^c$	$1$	$0^c$	$0$	$1^c$	$1^c$	$0^c$	$1^c$
$1$	$0^c$	$0^c$	$1^c$	$0$	$1^c$		
(a)与门			(b)或门			(c)非门	

图 4 门电路的敏化立方表

Fig. 4 The sensitized table of gate circuit

其它组合逻辑电路敏化表均可由基础敏化立方表来求解.表中  $A$ 、 $B$  为输入端,  $Y$  为输出端, 右上角标有  $c$  的逻辑值为“临界”值. “临界”值是, 当输入端的“临界”值发生改变时, 相应的输出也会改变 (因而输出也是临界).

语义分析程序中采用临界通路法求解测试集和结果集的过程, 就是由语法分析程序得到的功能树出发, 将其逐步拆解为由简单与、或、非门组成的基本敏化表的过程.

(1)将语法分析程序得到的功能树根据运算符功能拆分成仅有与门、或门、非门的机构, 并对每个门电路的输出端命名.

(2)确定输出端为  $0^c$ , 按表 3 中对应关系, 向输入端倒退求测试. 当所有临界值都敏化到输入端时, 将其它未定值一一确定.

(3)确定输出端为  $1^c$ , 重复上述步骤, 直到求得测试集.

例如, 对表达式 (1) 所求得的功能树有以下分析过程. 先确定输出端  $Y = 0$ , 即  $Y = 0^c$ .  $Y$  是由  $M$  经过非门后得到的, 查询基本敏化表可得  $M = 1^c$ ,  $M$  是由  $A$  与  $B$  经过与门后得到, 故查找与门基本敏化表得到  $(A, B) = (1^c, 1^c)$ .

再确定输出端  $Y = 1^c$ , 查表可得  $M = 0^c$ , 在基本敏化表中与门输出端为  $0^c$  时, 有两种输入方案, 故此时  $(A, B) = (1^c, 0^c)$  或  $(0^c, 1^c)$ .

综上可得可得  $(A, B)$  的测试集为  $(1^c, 1^c)$ 、 $(1^c, 0^c)$ 、 $(0^c, 1^c)$ , 与之对应, 结果集  $Y$  为  $0^c$ 、

$1^c$ 、 $1^c$ .

4 性能测试

基于上述设计思想, 设计出了组合逻辑电路芯片测试系统. 该系统上位机使用 C# 语言编写在 Microsoft Visual Studio 2005 开发平台上编译实现, 采用 SQLite 数据库; 下位机使用 STC89C52 芯片作为 MCU、使用 PL2303 芯片作为的 USB 转串口通信的转换芯片. 该系统在 Windows XP 操作系统下测试成功.

分别采用新设计的测试仪器和传统测试仪器 (未采用任何测试集生成算法的仪器) 对部分 TTL 型器件进行测试, 测试结果如图 5 所示.

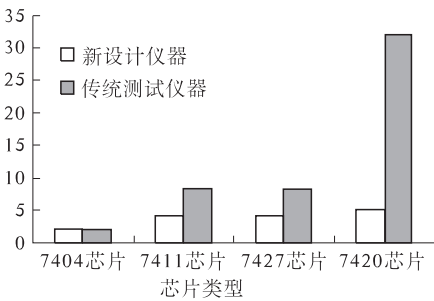


图 5 新设计仪器与传统仪器性能对比

Fig. 5 The comparison of performance between new designed instrument and traditional instrument

图中, 横轴分别是 7404 六反相器芯片、7411 三 3 输入与门芯片、7427 三 3 输入或非门、7420 双 4 输入与非门芯片; 纵轴是对相应芯片进行检测时所需要的测试集的组数. 由该图可以看出, 融合编译功能的测试集生成方式所产生的测试集和芯片的引脚数目没有关系; 测试集的组数和芯片的功能有直接的关系, 并随着芯片功能的复杂度的增加而缓慢增加 (对比传统测试仪器中测试集的组数成指数增加). 该方法合理的节约了测试芯片所需的测试集的组数, 节约了存储空间, 方便了芯片类型的扩充.

另外, 对比传统的测试仪器, 该测试仪采用软件编程代替硬件电路的思路, 将组合电路测试集生成方式与编译功能融合, 省去了结果显示、器件选择等电路部分, 有效简化了硬件电路. 器件扩充方式和测试集存储方式转由上位机软件控制, 使得处理更加方便, 操作也更加灵活、简洁.

5 结论

在理解编译功能原理的基础上, 设计出的融合编译功能的组合电路测试集生成方式, 易于程

序实现,具有很强的可移植性.使用设备时可以通过被测芯片的功能表达式,动态的生成与其对应的测试集和结果集,从而避免了将大量测试集和结果集存入到存储器,造成存储空间紧张的问题,方便了芯片类型的拓展.目前,该编译功能不适用于表达式中同一引脚名称出现多次的情况,下一步将针对此问题做进一步的分析改进.

参考文献:

[1] 王勇. 数字逻辑电路实验系统设计与开发[J]. 实验室研究与探索, 2008, 27(12): 42 - 44.  
[2] 薛延侠, 赫建国. 标准逻辑器件检测系统的设计[J]. 实验室研究与探索, 2010, 29(9): 33 - 35 + 112.  
[3] 刘敬猛, 吴星明, 胡晓光. 模拟集成电路测试仪的设计与开发[J]. 实验室研究与探索, 2009, 28(4): 192 - 195.

[4] 肖宝森. 多功能数字芯片测试仪的设计与应用[J]. 实验技术与管理, 2010, 27(12): 133 - 136.  
[5] 张冬莱, 方习文. 编译原理课程设计的教学实践与改革[J]. 实验室研究与探索, 2012(11): 134 - 137, 153.  
[6] 王改芳, 龚君芳, 李圣文, 等. 编译原理课程实践教学改革探索[J]. 实验技术与管理, 2009(12): 130 - 131.  
[7] 张大方, 张洁坤, 黄昆. 一种基于智能有限自动机的正则表达式匹配算法[J]. 电子学报, 2012(08): 1617 - 1623.  
[8] 杨明, 姜乃松, 蔡维玲. 编译原理简明教程[M]. 北京: 电子工业出版社, 2012: 89 - 93.  
[9] 黄贤英, 曹琼, 王珂柯. 编译原理及实践教程[M]. 2 版. 北京: 清华大学出版社, 2012: 90 - 95.  
[10] 张卫党, 李萍. 基于能量分配提高纠错码误比特率性能的研究[J]. 郑州大学学报: 工学版, 2013, 34(1): 125 - 128.

Generation of Combinational Circuit Test Suite Combined with Compilation Feature

LI Tian-yi<sup>1</sup>, FU Kun<sup>1</sup>, GENG Yue-hua<sup>2</sup>, XU Dan<sup>1</sup>

(1. School of Computer Science and Engineering, Hebei University of Technology, Tianjin 300401, China; 2. School of Electrical Engineering and Automation, Hebei University of Technology, Tianjin 300130, China)

**Abstract:** By researching the generation of test suite in combinational circuit testing instruments, a generation of combinational circuit test suite combined with compilation feature was proposed. The generation of test suite, which combined the compilers principles with test suite generation algorithm, solving some difficult problems in combinational circuit testing instruments; the type of tested chip is not easy to expand, hardware design circuit is complexity, and so on. This paper starts from the functional expression of tested chip, elaborating the analysis process of the lexical, syntax and semantics in detail. And taking critical path method for example, this paper gives a detailed design scheme of the test suite generation. Finally, according to the design, we designed a combinational circuit test system based on STC89C52 chip, and gave the performance test results of the system.

**Key words:** compilation principle, critical path method, combinational circuit, test suite