# 用 AIG 推理检验组合电路的等价性

范全润，段振华，徐国培
(西安电子科技大学 计算机学院，陕西 西安 710071)

**摘要**：大部分基于 SAT 的组合电路等价性检验方法是将两个待检验的电路组合成一个 miter 电路，将这个电路变换成 CNF 形式，然后调用一个 SAT 判定器来确定这个 CNF 是否是可满足的. 但是，当 miter 电路被变换成 CNF 之后，就丢掉了电路的结构信息. 针对这种方法的不足，先假定 miter 的输出为 1，然后从 miter 的输出端开始，回溯检查是否存在冲突来判定 miter 的可满足性. 利用 AIG 的特点，把每个节点的四种输入组合归结为一种，从而使推理得到了简化. 实验表明，此方法有更快的处理速度.

**关键词**：组合电路；等价性检验；电路推理；与非图
**中图分类号**：TN47 **文献标识码**：A **文章编号**：1001-2400(2009)05-0877-08

## Combinational equivalence checking based on AIG reasoning

*FAN Quan-run*, *DUAN Zhen-hua*, *XU Guo-pei*
(School of Computer Science and Technology, Xidian Univ., Xi'an 710071, China)

**Abstract**: Most SAT based combinational equivalence checking algorithms combine two circuits into a miter, and then convert the miter into a CNF formula. After that, an SAT solver is invoked to check whether the CNF formula is satisfiable. However, when a miter is converted into a CNF formula, the structure information of the circuit is lost. In this paper, we assume the output of the miter is 1, and then the backtracking method is used for conflict checking to decide the satisfiability of the miter. By using the character of AIG, four input combinations of each node can be reduced into one to simplify the deducing procedure. Preliminary experiments demonstrate the efficiency of our approach.

**Key Words**: combinational circuits; CEC; circuit reasoning; AIG

电子设计自动化软件需要确保电路在综合优化前和优化后是功能等价的. 组合等价性检验(Combinational Equivalence Checking, CEC)比较同一个电路设计的两种不同描述，以检验它们在功能上是否等价. 在电子设计自动化领域，CEC 技术有着广泛的应用. 它可以用来检验组合逻辑电路在经过优化或者人工修改后，是不是还保持功能上的等价. CEC 还可以用在时序电路的等价性检验中. 此外，它还可以用来检验逻辑 Cone 的等价性[1]，从而可用于有界时序等价性检验，无界时序等价性检验，或者模型检验[2-4].

现有的很多 CEC 工具都是基于二叉判定图(Binary Decision Diagram, BDD)或者可满足性(Satisfiability, SAT)的. 对于简化的有序二叉判定图(Reduced Ordered Binary Decision Diagram, ROBDD)，同一布尔函数的 ROBDD 形式是唯一的. 基于 BDD 的组合电路等价性检验方法把待检验的两个电路都变换成 BDD，从而将两个电路的等价性问题归结为检验它们的 BDD 表示是不是同构的. BDD 的主要问题是存在状态爆炸的可能.

近几年来，SAT 算法取得了很大的进步，于是很多研究人员采用 SAT 的方法来实现 CEC. 基于 SAT 的 CEC 方法首先将待检验的两个电路组合成一个电路，这个电路称为 miter. miter 的构造方法是将两个电路的对应输入连到一起，将两个电路中对应的输出连到一个异或门的输入，然后再将所有异或门的输出连到一个或门上，这个或门的输出就是 miter 的输出. 如果待检验的两个电路不等价，那么至少有一个异或门的

**收稿日期**：2008-12-16
**基金项目**：国家自然科学基金资助(60373103, 60433010)
**作者简介**：范全润(1973-)，男，西安电子科技大学博士研究生，E-mail: ynfqr@163.com.

输出为1,从而导致或门的输出(即miter的输出)为1.因此,要检验两个电路是否等价,只要检验miter的输出为1是不是可满足的,如果不可满足,则两个电路是等价的,否则两个电路不等价.基于SAT的方法一般把miter变换成布尔合取范式(Conjunctive Normal Form,CNF),然后调用一个SAT判定器(SAT Solver)来检验这个CNF公式是否可满足.布尔可满足性的判定是NP-complete问题[5],这意味着SAT判定算法的复杂度是指数阶的.

因为在电子设计自动化研究领域的许多问题都可以归结为用CNF表示的SAT问题,所以为了使SAT求解器能处理更多的问题,大部分SAT求解器都是基于CNF而不是电路本身.当miter被变换成CNF后,电路中的结构信息就丢失了.为了利用电路中的结构信息来加快SAT的判定速度,一些研究者建议用电路推理来判定组合电路的等价性[6-9].这些研究的主要问题是不能和EDA工具中的综合工具很好的相结合(例如CSAT[8-9]就只能接受.bench格式的输入),而且没有充分利用特定形式的电路的特点进行推理,效率不高.

miter电路用AIG(And-inverter Graph)[10]来表示.由于越来越多的电路综合工具采用AIG格式,基于AIG格式推理的方法能更好地和综合工具相结合.此外,由于充分利用了AIG的特点来进行电路推理,和现有的基于电路推理的方法相比,本文中的方法更为简单、有效.先假定miter电路的输出为1,然后从电路的输出开始回溯推理,看能不能找到使电路的输出为1的一组原始输入值.如果满足条件的原始输入不存在,说明miter电路的输出值不可能为1,要检验的两个电路是功能等价的.反之,两个电路功能不等价.推理规则充分利用了两输入与门的特点,减小了搜索空间,从而加快了搜索的速度.

## 1 相关概念

在笔者提出的算法中,miter电路是用AIG来表示的.AIGER格式[6]描述了如何用文件来存储AIG.电路的AIG表示存储在一个.aag格式的文本文件中.

### 1.1 AIG

一个组合布尔网络是一个有向无环图(Directed Acyclic Graph,DAG).在有向无环图中,节点对应于逻辑门,有向边则对应于逻辑门之间的连接.节点的输入边称为扇入,节点的输出边称为扇出.网络中没有输入边的节点称为原始输入,没有输出边的节点称为原始输出.原始输入和原始输出定义了电路和外部的连接关系.

AIG是一种特殊的有向无环图.在AIG中,每一个节点要么没有输入,要么有两个输入.没有输入的节点就是原始输入,有两个输入的节点就是两输入的与门.边有可能被取反.如果一条边被取反,则说明该边对应的输入被取反.在AIG图中,用圆圈表示与门,通过给边加上圆点的方式来表示取反操作.图1给出了一个例子,它表示用与非门表示的一个电路及其对应的AIG,该电路相当于一个或门.

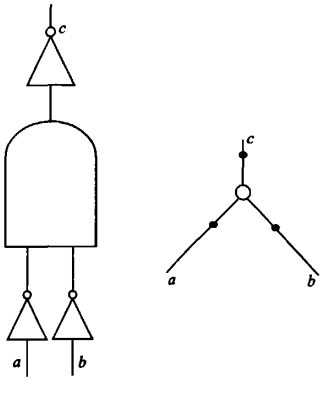

图1 或门的等价电路及其AIG表示

任何组合电路的布尔网络都可以分解变换成AIG.在构造AIG的过程中,采用结构化哈希的方法来确保没有任何两个与门的输入边是相同的.图1给出了或门电路及其对应的AIG表示.

### 1.2 AIGER格式

AIGER格式用多根与非图来表示电路.因为miter电路只有一个输出,所以我们要处理的AIG只有一个根节点.一个AIGER格式的文件以文件格式描述符串开头.格式描述符可能是aag,表示这是ASCII格式,或者是aig,表示是二进制格式.在文件格式描述符之后是一个空格,然后是5个用ASCII码表示的非负整数,$M,I,L,O,$和$A$.$M$是最大变量索引值,表示了变量的个数.电路中有$I$个输入,$L$个锁存器,$O$个输出,$A$个与门.如果所有的变量都被用到,也不存在没有被使用的与门,则应该有$M=I+L+A$.对于用于组合电路等价性检验的miter电路,因为它没有latch,只有一个输出,所以$L$的值应该为0,$O$的值应该为1.文字

(*Literal*)是变量(*Variable*)或者变量的非.例如,假定有一个变量 $a$,则这个变量对应两个文字,即 $a$ 和 $\neg a$. 在 AIGER 格式中,变量用一个正整数索引值来表示,变量对应的文字也用整数来表示,变量对应的正文字用变量索引乘以 2 来表示,负文字则用变量索引乘以 2 再加 1 来表示.例如,可以用整数 3 来作为变量 a 的索引,它对应的文字 $a$ 用整数 6 来表示,$\neg a$ 用整数 7 来表示.

例如,图 1 所示的或门用 AIGER 格式可以描述为

```
aag  3  2  0  1  1
2
4
7
6  3  5
```

在第 1 行中,"aag"表示这个文件的描述是 ASCII 格式,"3 2 0 1 1" 表示变量的最大索引值是 3,电路有两个输入,没有锁存器,有一个输出,有一个与门.第 2 行和第 3 行表示输入(变量 1 和变量 2).第 5 行表示与门,它的输入分别是 3(变量 1 的非)和 5(变量 2 的非).与门的输出为 6.第 4 行表示电路的输出是 7,它是电路中与门的输出的非.

如果用符号 $a$, $b$ 来表示与门的输入,用符号 $c$ 来表示与门的输出,用整数 1,2,3 分别来表示 $a$, $b$, $c$,则在上面的例子中,2 相当于 $a$, 4 相当于 $b$, 6 相当于 $c$, 而 3,5,7 则分别相当于 $\neg a$, $\neg b$, $\neg c$. 最后电路的输出是 $\neg(\neg a \wedge \neg b)$, 等价于 $a \vee b$.

## 2　基于 AIG 推理的组合电路等价性检验

### 2.1　表示 AIG 的数据结构

在算法中,原始输入和与门都用相同的结构体来表示.每个结构体包含四个域,(out, $l$_in, $r$_in, out_value). 其中,out_value 表示该节点的输出值,所有节点的 out_value 域都初始化为 $X$. 对于与门,out, $l$_in, $r$_in 分别表示它的输出,左输入和右输入.在本文中,"左输入"与"右输入"的概念是为了区分一个与门的两个输入,并不表示它们的位置关系.和 AIGER 格式一样[6],用一个整数来表示变量.左输入和右输入都对应相应的变量.如果输入边上没有非门,则表示输入边的域存储的文字就是相应的变量索引乘以 2,如果输入边上有非门,则相应的表示输入边的域存放的是变量索引乘以 2 后再加 1 的值.对于原始输入,将它的 $l$_in 和 $r$_in 设为 0.因为与门的输出是唯一的,所以,与门用对应节点的 out 域来表示.

### 2.2　由 AIGER 格式文件构造 AIG

算法 1 给出了由.aag 文件构造 AIG 的方法.

在.aag 文件中的每个原始输入和与门都被映射为图中的一个节点.对于原始输入,构造一个节点来代表它.原始输入节点的 out 域设为它本身,域 $l$_in 和 $r$_in 则置为 0,out_value 域置为 $X$. 例如,在.aag 格式文件中描述的一个原始输入"2" 被转换成一个节点,(2,0,0,$X$). 对于每个与门要构造一个与之相对应的节点,节点的 out_value 域初始化为 $X$. 其它域则按相应的与门实体来进行设置.例如,在.aag 文件中和一个表示与门的项"6 3 5"会被转换成节点(6,3,5,$X$).

电路的结构图可以从节点的信息中找出来.从根节点开始,对于每一个节点,通过它的 $l$_in 域和 $r$_in 域,就可以找到它的左输入节点和右输入节点,重复此过程,直到找到的节点是原始输入节点为止,就可以得到整个电路表示.因为一个 miter 电路的根节点是惟一的,所以 miter 的输出可以用根节点的 out 域表示.

算法 1——Construct AIG from. agg File

```
/ * create vertex for PIs * /
for i=1 to I do
  vertex=new Vertex();
  vertex.out=readInputEntity();
  vertex.out_value=X;
```

```
    vertex.l_in=0;
    vertex.r_in=0;
    vertex.imply=NULL;
    addItem(htable,vertex);
  end for
  /* create vertex for AND gates */
  for i=1 to A do
    vertex=new Vertex();
    andInfo=readAndEntity();
    vertex.out=andInfo.out;
    vertex.out_value=X;
    vertex.l_in=andInfo.l_in;
    vertex.r_in=andInfo.r_in;
    vertex.imply=NULL;
    addItem(htable,vertex);
  end for
  /* Find root node */
  miterout=readOutEntity();
  if (miterout is odd) then
    root=lookup_htable(miterout-1);
  else
    root=lookup_htable(miterout);
  end if
  root.out=miterout;
  return root;
  算法 2——check vertex
  int check(v)
  {
  v.imply=NULL;
  if v is a primary input node then
    NEW t_imply;
    t_imply.node=v.out;
    t_imply.nodevalue=v.out_value;
    t_imply.next=NULL;
    status=addImply(v.imply, t_imply, 1);
  end if
  if (v.out_value=1) then
    v.lchild.out_value=infer(l, v.l_in);
    v.rchild.out_value=infer(l, v.r_in);
    check(v. lchild);
    status=addImply(v.imply, v.lchild.imply, 1);
    if (status=1) then
      return 1;
    end if
```

```
    check (v. rchild);
    addImply(v. imply, v. rchild. imply, 1);
    if (status=1) then
      return 1;
    end if
  else
    t_imply=NULL;
    v. lchild. out_value=infer(0, v. l_in);
    check (v. lchild);
    status=addImply(t_imply, v. lchild. imply, 1);
    v. rchild. out_value=infer(0,v. r_in);
    check(v. rchild);
    status=addImply(t_imply, v. rchild. imply, 0);
    status=addImply(v. imply, t_imply, 1);
    if (status=1) then
      return 1;
    end if
  end if
  return 0;
  }  .
```

通过这种方式，用. aag 格式文件存储的电路被转换成了一种由单一类型节点表示的有向图. 例如，对于图 1 所表示的电路，假如它是一个 miter 电路，并且用整数 1，2 来代表两个原始输入，用整数 3 来代表电路的输出. 2. 2 节中给出的文件格式的例子实际上就描述了这个电路. 在算法中，这个电路对应的 AIG 有 3 个节点，分别为表示输入 $a$ 的节点(2,0,0,$X$)，表示 $b$ 的节点(4,0,0,$X$)，以及表示与门的节点(6,3,5,0). 此处因为 miter 的输出要为 1，所以与门的输出应该为 0.

### 2.3　基于电路推理的组合等价性检验

#### 2.3.1　基本思路

先假定 miter 的输出为 1，然后从输出开始对 AIG 进行回溯推理，试图寻找一个能使 miter 输出为 1 的一组不冲突的原始输入值. 如果这样的输入值存在，则说明 miter 的输出可以为 1，两个电路不等价. 反之，说明两个电路是等价的.

在两个电路不等价的情况下，找到的不冲突的原始输入值可以作为一个反例，以方便查找导致两个电路不等价的原因.

例如，图 2 就是一个 miter 电路. 和算法描述不同的是，为了方便理解，没有把它转换成与非图. 我们先假定输出 $c$ 为 1，则有 ($d=0, f=1$) 或者($d=1, f=0$). 对于($d=0, f=1$)，由于 $d=0$ 隐含了($a=0, b=0$)，而 $f=1$ 隐含了 $a$ 和 $b$ 中至少有一个要为 1，所以推出矛盾. 同样，对于($d=1, f=0$) 的情况，也必然推出矛盾. 故此 miter 的输出不可能为 1，图中虚线圈出的 Circuit1 和 Circuit2 是等价的.

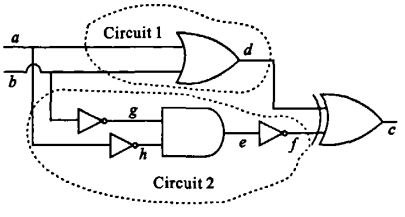


图 2　一个 miter 电路

笔者提出的算法充分利用 AIG 的特点来进行电路推理. 由根结点开始，先检查当前节点，如果当前节点的输出值是 1，则检查它的两个输入是否都能为 1. 如果当前节点的输出值为 0，则需要检查它的两个输入组合能否为(1,0)，(0,1)，或者(0,0). 这 3 种情况可以归约为检查它的两个输入组合能否为 (0,$X$) 或者($X$,0)，因此，只需要检查它的左输入为零时的隐含值或右输入为零时的隐含值是否会产生冲突即可. 如果一个节点的输出值为 $X$，就没必要检查它的隐含值，因为任意的原始

输入值都能使该节点的输出值为 $X$，不可能产生冲突.

因此，无论当前节点的输出是0还是1，对于它的下一级节点都只需要考虑一种组合，即为1的时候考虑它的两个输入同时为1是否会导致冲突，为0的时候考虑两个输入之一为0是否会导致冲突，相当于只需在考虑左右输入的一种组合. 如果不利用电路的结构信息，就需要分别考虑左右输入的4种组合.

大部分情况下，每个节点只需要进行一次处理即可. 但是，如果在一个节点有多个输出，可能需要进行两次处理. 例如，图3中的节点 $c$ 是一个多输出的节点，假定通过上层节点的分析，要求节点 $a$ 和 $b$ 的输出都为0. 如果从节点 $a$ 来看，就要检查它的右输入能否为0，注意到边上有一个非门，因此，需要检查节点 $c$ 的输出能否为1；从节点 $b$ 来看，需要检查节点 $c$ 的输出能否能0. 这样，节点 $c$ 就需要同时考虑输出为0和输出为1的情况.

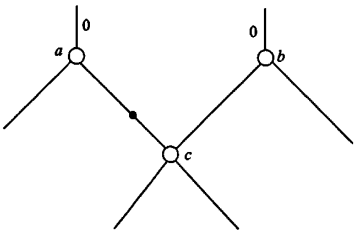


图3 存在多输出的aig节点

综上所述，只有当一个节点同时满足两个条件：(1)是多输出节点；(2)由上层节点推导出的该节点的输出分别为0和1. 满足这两个条件，才需要对输出为0和输出为1分别进行处理. 同时满足这两个条件的节点只占小部分，因此大部分节点只需要处理一次.

如果用标准的SAT方法，每个节点将会变换成三个子句，同时每增加一个节点，会增加一个变量，对每个变量都要考虑其取值为0和取值为1的情况，因此算法的复杂度为 $O(2^{(m+n)})$，其中 $m$ 为输入的个数，$n$ 为节点的个数. 如果电路中的每个门都满足上述的两个条件，笔者提出的算法复杂度也是 $O(2^{(m+n)})$. 但是，在实际电路中，满足上述两个条件的门只占小部分，因此实际的处理复杂度可以大大降低. 实验结果也证实了这一点.

### 2.3.2 推理算法

算法3给出了推理方法的基本框架. 首先，确定根节点值，然后调用check()来检查能否找到使miter的输出为1，且不存在冲突的一组原始输入值. 为了检查是否存在无冲突的一组赋值，每个结点增加一个叫做imply的域. 域imply是一个指针，它所指的链表存储了当前节点所隐含的原始输入的值.

算法3——satCheck()

```
if(root.out is odd) then
  root.out_value=0;
else
  root.out_value=1;
end if
root.imply=NULL;
result=check(root);
if result=1 then
  return EQUAL;
else
  return UNEQUAL;
end if
```

算法4——infer child nodes' value

```
infer(rea_value, flag)
{
if flag is odd then
  cflag=1;
else
  cflag=0;
```

end if

return(cflag XOR req_value);

} .

在算法 2 中，$v$.lchild 和 $v$.rchild 分别表示 $v$ 的左右孩子节点，它们可以由 $v$ 的 $l$_in 域和 $r$_in 域推导出来.

函数 addImply(imply1, imply2, flag) 把隐含值表 imply2 合并到隐含值表 imply1 中，合并的方法由 flag 的值决定. 如果 flag 的值是 0，则两个隐含值表做或运算. 如果 flag 为 1，两个表做与运算. 只有在两个隐含值表做与运算的时候才会产生冲突. 因为一个节点的输入可能是它的孩子节点的输出取反，所以在已知当前节点的输出的情况下，用一个函数 infer()来推断它的孩子节点的输入值，如算法 4 所示.

## 3 实验结果

用 C++语言实现了 miter 可满性检验工具 AIG－SAT，以验证组合电路的等价性.

实验用 ISCAS85 电路作为测试基准电路. 原始电路和优化后的电路都是用 Altera 公司的 QuartusII[11-12]软件来生成的. 通过修改工程配置文件，让 Quartus II 读入用 Verilog 描述的电路，在对电路进行优化之前和优化之后，分别生成对应的. blif 格式文件. 然后利用 ABC[13]把对应的. blif 格式的文件组合成一个 miter，并用一个叫 bliftoaig[10]的工具将 miter 变换成. aag 格式，然后用我们的工具来判定 miter 的可满足性. 为了和基于 CNF 的 SAT 求解器进行比较，miter 被变换成 CNF 格式，然后分别用 MiniSAT 1.14[14]和 PicoSAT 632[15]来判定它的可满足性. 由于 C-SAT 只能接受. bench 格式的输入，所以用了该工具所带的例子中的. bench 文件来测试它的速度. 实验用的计算机配置为 Pentium D 2.8GHz 的 CPU，内存为 1G，操作系统为 Linux，Fedora 7. 表 1 给出了 AIG－SAT 和 MiniSAT，PicoSAT 以及 C－SAT 的运行时间. 对于几个比较小的电路，由于求解时间很短，难以进行比较，故没有将其列在表中. 对于电路 C6288，这几个工具都无法得到结果.

**表 1　CEC 工具运行时间比较**　　s

| 电路 | MiniSAT | PicoSAT | C-SAT | AIG-SAT |
|---|---|---|---|---|
| c1355 | 0.12 | 0.09 | 0.11 | 0.04 |
| c1908 | 0.71 | 0.33 | 0.22 | 0.13 |
| c2670 | 0.05 | 0.06 | 0.16 | 0.01 |
| c3540 | 2.76 | 3.33 | 2.46 | 1.90 |
| c5315 | 0.42 | 0.67 | 1.27 | 0.01 |
| c7552 | 0.72 | 0.67 | 3.05 | 0.04 |
| 平均 | 0.797 | 0.858 | 1.21 | 0.37 |

## 4 结　论

给出了一种方法，用电路推理来检验两个组合电路的等价性. 首先从. aag 文件构造出 miter 的 AIG 表示，然后在 AIG 上利用电路的结构信息进行推理，从而验证 miter 的可满足性.

实验表明，和现有的一些基于 CNF 的 SAT 工具(MiniSAT 和 PicoSAT)以及基于电路推理的工具(C-SAT)相比较，此方法有更快的处理速度. 我们认为，基于 AIG 推理的工具(AIG-SAT)速度更快的原因在于它能更好地利用 AIG 电路的结构信息. C-SAT 虽然也是利用电路的结构信息来进行推理，但由于它考虑的电路是较为通用的表示形式，没有充分地利用 AIG 表示的电路的特点，从实验来看，它的速度实际上比现有的基于 CNF 的 SAT 工具还慢.

但是，本文中只考虑了组合电路的等价性检验问题. 如何将这种方法应用到电子设计自动化的其它领域，如时序电路的等价性检验，还需要进一步的研究.

**参考文献：**

[1] Lu F, Cheng K. Sequential Equivalence Checking Based on K-th Invariants and Circuits SAT Solving[C]// Proceedings

of the High-Level Design Validation and Test Workshop'05. New York: IEEE, 2005: 45-52.

[2] Kuehlmann A. Dynamic Transition Relation Simplification for Bounded Property Checking[C]// Proceedings of the 2004 IEEE/ACM International Conference on Computer Aided Design. New York: IEEE, 2004: 50-57.

[3] 张海宾，段振华. 多速率混合系统的模型检查[J]. 西安电子科技大学学报，2008，35(1):60-64.
Zhang Haibin, Duan Zhenhua. Model Checking Multirate Hybrid Systems [J]. Journal of Xidian University, 2008, 35 (1):60-64.

[4] 郭建，金乃咏. 模型检验中对 CTL 公式的空属性探测[J]. 西安电子科技大学学报，2007，34(5)：794-799.
Guo Jian, Jin Naiyong. Vacuity Detection in Computation Temperal Logic [J]. Journal of Xidian University, 2007, 34 (5): 794-799.

[5] Cook S. The Complexity of Theorem Proving Procedures[C]//ACM SIGACT Symposium on the Theory of Computing. New York: ACM, 1971: 151-158.

[6] Kuehlmann A, Ganai M K, Paruthi V. Circuit Based Boolean Reasoning[C]//Proceedings of the 38th annual Design Automation Conference. New York: ACM, 2001: 232-237.

[7] Kuehlmann A, Paruthi V, Krohm F, et al. Robust Boolean Reasoning for Equivalence Checking and functional Property Verification[J]. IEEE Trans on Computer-Aided Design of Integrated Circuits and Systems, 2002, 21(12): 1377-1394.

[8] Lu F, Wang L, Cheng K, et al. A Circuit SAT Solver with Signal Correlation Guided Learning[C]//Proceedings of the Conference on Design, Automation and Test in Europe. Washington: IEEE Computer Society, 2003: 892 - 897.

[9] Lu F, Wang L, Cheng K, et al. A Signal Correlation Guided ATPG Solver and Its Applications for Solving Difficult Industrial Cases[C]//Proceedings of the 40th annual Design Automation Conference. New York: ACM, 2003: 436-441.

[10] Biere A. AIGER (AIGER is a format, library and set of utilities for And-inverter Graphs (AIGs)) [CP/OL]. [2008-09-30]. http://fmv.jku.at/aiger/.

[11] Altera Corp. Quartus II University Interface Program [CP/OL]. [2008-09-10]. http://www.altera.com/education/univ/research/unv-quip.html.

[12] Pistorius J, Hutton M, Mishchenko A, et al. Benchmarking Method and Designs Targeting Logic Synthesis for FPGAs [C]// Proceedings of the ACM/IEEE International Workshop on Logic and Synthesis. New York: IEEE, 2007: 230-237.

[13] Berkeley Logic Synthesis and Verification Group. ABC: a System for Sequential Synthesis and Verification. Release 70930[CP/OL]. [2008-10-22]. http://www.eecs.berkeley.edu/ alanmi/abc.

[14] Eén N, Sörensson N. An Extensible Sat-solver[C]//SAT 2003, LNCS: 2919. Berlin: Springer, 2004: 333-336.

[15] PicoSAT[CP/OL]. [2008-10-22]. http://fmv.jku.at/picosat/.

(编辑：高西全)

## 简 讯

※ 香港城市大学(City University of Hong Kong)电子工程学院副教授苏庆祥博士于 2009 年 6 月 23 日～25 日来我校讲学访问、合作科研. 苏庆祥副教授 1990 获得香港城市大学学士学位，并于 1995 年获得香港中文大学博士学位. 苏博士的主要研究领域包括：快速自适应算法，统计信号处理，信号检测，参数估计和定位技术等.

※ 爱尔兰利墨瑞克大学 Ian Grout 博士于 2009 年 6 月 29 日－7 月 2 日来我校进行了学术交流和访问. Ian Grout 博士于 1991 年和 1994 年在英国 Lancaster University 分别获得一等荣誉工程学士学位和博士学位. 主要研究方向是 IC 设计及测试.

转自《西电新闻》2009.7.6