基于乘积项的双逻辑实现探测算法

叶锡恩, 毛科益, 夏银水
(宁波大学电路与系统研究所, 浙江宁波 315211)

摘 要: 在基于函数最小项的双逻辑探测算法中, 由于函数最小项个数将随着变量数的增加而急剧增大, 使得算法因时间或空间的占用过大而失效. 为此, 本文提出了基于函数最简项的快速探测算法, 弥补了其缺陷. 由于基于最小项和最简项的探测算法所适用的函数范围都有一定的局限性, 因此进一步提出了将两种算法综合运用的策略, 可有效改进探测效果. 经 MCNC Benchmark 上电路的判定测试, 验证了其有效性.

关键词: Boolean 逻辑; Reed-Muller 逻辑; 汉明距离; 探测算法
中图分类号: TN432; TP331 文献标识码: A 文章编号: 0372-2112 (2009) 05-0961-05

Algorithms for Detecting Dual Logic Based on Product Term

YE Xi-en, MAO Ke-yi, XIA Yin-shui
(*Institute of Circuits and Systems, Ningbo University, Ningbo, Zhejiang* 315211, *China*)

Abstract: In the dual logic detecting algorithm which based on miniterms, the exponential expansion of minterms makes the minterms based algorithm work slowly and exhausts the memory. To accelerate the detection speed for the large functions, we proposed a novel detecting algorithm which based on cubes. Furthermore a strategy of mixed using of cube based detection and minterm based detection is proposed to get a good result. The experimental results on NCMC benchmark prove our algorithm and strategy effectiveness.

Key words: boolean logic; reed-muller logic; hamming distance; detection algorithm

1 引言

集成电路设计中一个非常重要的环节是电路的逻辑综合与优化. 在逻辑综合和优化中, 最为重要的技术之一是逻辑函数的优化. 这是由于更具优化的逻辑函数可直接获得优化的逻辑电路. 数字电路既可以用基于 AND/OR/NOT 运算基的 Boolean 逻辑来实现, 也可以用基于 AND/XOR 运算基的 Reed-Muller (RM) 逻辑来实现[1~4]. 然而目前, 几乎所有的 Electronic Design Automation (EDA) 工具均是基于 Boolean 逻辑发展而来的, 这意味着对那些适于用 RM 逻辑实现的函数无法得到逻辑最优化. 同样, 对于那些适于用 Boolean 逻辑实现的函数若用 RM 逻辑实现也得不到函数的最优化[5]. 事实上, 对于大部分内含 XOR 逻辑的电路用 AND/XOR 两种运算基实现并非是最优化的, 而最为理想的是用基于 AND/OR/NOT/XOR 四种运算基来实现, 为此把由 Boolean 逻辑和 RM 逻辑相结合的逻辑称之为双逻辑. 发展基于双逻辑的自动综合工具亦是必然趋势, 是解决当今集成电路设计发展中遇到的各种挑战的关键, 而本文所要研究的逻辑探测技术是其重要组成部分.

在设计一个较大的数字系统中, 电路结构的优化过程是非常耗时的, 因此, 对于任意给定电路, 设计者希望预先知道采用哪种逻辑来实现会使电路结构得到最大限度的优化, 这就需要对给定电路进行逻辑预判断, 即称为逻辑探测技术. 目前国际上有关这方面的研究刚刚起步, 相关的研究成果比较少. 其中文献[6]对此问题作了一定的研究, 提出了一个用于判断逻辑函数是否可表示成 $f=(g \oplus h)+r$ 的充分条件, 并形成了算法. 但该算法仅考虑了函数 f 中的 $g \oplus h$ 这部分函数, 而没有说明剩下的 r 部分函数的结构特点及与前面部分函数的关联性, 因而, 在实用中存在一定的缺陷. 针对这一点, 文献[7]提出了一个基于函数最小项的逻辑探测算法, 同时给出了判定的充分条件, 该算法对给定逻辑函数能进行正确的逻辑判定. 但当函数变量数增加到 20 以上时, 部分函数的最小项数目超过百万而使算法失效, 程序在计算机上运行不出结果. 鉴此, 提出基于函数最简

收稿日期: 2008-04-30; 修回日期: 2009-01-13
基金项目: 国家自然科学基金 (No. 60676017); 浙江省自然科学基金 (No. Y106818); 国家教育部留学回国基金; 浙江省科技厅项目 (No. 2007C24017); 宁波市科技局项目 (No. 2006A610091)

项的逻辑探测算法. 因为, 对于输入变量数较多的函数, 其对应的最小项数目庞大, 但相应的最简项数目要少得多. 如 MCNC benchmark 上的输入变量数为 23 的 cordic 电路, 它的第一个输出函数的最小项个数为"7806464", 而相应最简项个数仅为"179". 因此, 对于输入变量数较多的电路, 需要考虑从函数的最简项对其进行逻辑模式的探测; 而对于输入变量数相对较少的电路, 可以选择基于最简项, 也可以选择基于最小项对其进行逻辑模式的探测. 将两种算法综合运用, 可使算法的探测效果达到最优.

2 基于最小项的探测算法

在 Boolean 逻辑里, XOR 是一个复合逻辑运算, 即由 AND/ OR/ NOT 运算基的组合来实现. 而在 RM 逻辑里, XOR 是作为一个基本运算的单元, 与 AND/ OR/ NOT 运算基具有同等地位. XOR 逻辑的特点是输入项之间的汉明距离为 2, 即两个汉明距离为 2 的输入项可用 XOR 逻辑表示.

文献[7]提出的基于函数最小项的探测算法是直接从 XOR 逻辑的特点出发展开研究的. 通过一系列相关的定义, 给出适于双逻辑实现的判断条件.

定义 1 对于一个给定的 n 变量函数, 记任意两个最小项 m_i 和 m_j 的下标对应的二进制码分别为 $x_{n-1}\cdots x_1x_0$ 和 $y_{n-1}\cdots y_1y_0$, 码组 $x_{n-1}\cdots x_1x_0$ 和 $y_{n-1}\cdots y_1y_0$ 之间的汉明距离为 $H_{m_im_j}$, 即得最小项 m_i 与 m_j 之间的汉明距离为 $H_{m_im_j}=\sum_{k=0}^{n-1}(x_k\wedge y_k)$(其中"$\wedge$"表示异或).

定义 2 逻辑函数任意两个最小项 m_i 和 m_j, 记对应在卡诺图上只包含这两个最小项的最小卡诺圈为 $sup(m_i, m_j)$. 若 $H_{m_im_j}=2$, 则记为 $sup(R_i, R_j)$; 若 $H_{m_im_j}=1$, 则记为 $sup(B_g, B_h)$; 单个最小项记为 S_k.

判断条件: 记 n 变量函数 f 最小项个数为 N_{cube}, 其中 $sup(R_i, R_j)$ 个数记为 N_R, $sup(B_g, B_h)$ 个数记为 N_B, S_k 的个数记为 N_S; 且 $N_{cube}=2\times N_R+2\times N_B+N_S$ (N_R, N_B, $N_S\in\{0, 1, 2, \cdots\}$, $N_{cube}<2^N$); 若函数 f 满足下列条件, 即可判定该函数用双逻辑实现比用 Boolean 逻辑实现简单:

(1) $N_R>N_S$.

(2) 当 $N_R=N_S$ 时, 至少存在一项 S_k 不与任意 $sup(R_i, R_j)$ 中的 R_i 和 R_j 项相邻.

上述算法可实现对任意给定函数是否适于双逻辑实现的判定. 其算法时间复杂度与给定函数的最小项数 N_{cube} 有关, 上限为 $O(N_{cube}^2)$, 算法对计算机内存所需为 N_{cube}. 可见, 该算法具体运行对象是函数的所有最小项. 因此, 当函数最小项个数随着变量数的增加而急剧增大时, 算法将失效. 鉴此, 本文考虑从函数的最简项着手, 分析函数的逻辑模式.

3 基于最简项的探测算法

3.1 定义

在数字电路中, 逻辑函数有各种表示方法, 较为常用的有逻辑真值表、逻辑函数式、逻辑图和卡诺图等. 其中逻辑函数式是指把输入与输出之间的逻辑关系写成 AND/ OR/ NOT 运算的组合式, 即逻辑代数式. 在进行逻辑运算时常能看到, 同一个逻辑函数可以写成不同的逻辑式, 而这些逻辑表达式的繁简程度又相差甚远, 最简单的为函数的"最简项之和"形式, 定义如下:

定义 3 在 AND-OR 逻辑函数式中, 若其中包含的乘积项已经最少, 而且每个乘积项里的因子也不能再减少时, 则称此逻辑函数式为最简形式, 其包含的任意乘积项均为最简项.

定义 4 对于一个给定的 n 变量函数, 记任意两个输入项 X、Y 对应的输入码分别为 $x_{n-1}\cdots x_1x_0$ 和 $y_{n-1}\cdots y_1y_0$ ($x_i, y_i\in\{0, 1, -\}$, $i\in\{0, 1, 2, \cdots, n-1\}$), 码组 $x_{n-1}\cdots x_1x_0$ 和 $y_{n-1}\cdots y_1y_0$ 之间的汉明距离为 H_{XY}, 即得输入项 X 与 Y 之间的汉明距离为 $H_{XY}=\sum_{k=0}^{n-1}(x_k\wedge y_k)$ (其中"$\wedge$"表示异或, "-"异或"-"结果为零).

定义 5 对于给定的以最简项形式表示的逻辑函数, 记函数包含的最简项个数为 N_{SOP}; 对任意两个最简项 A 和 B, 若 $H_{AB}=2$, 则记为 $pair(A, B)$; N_{pair} 表示 $pair(A, B)$ 的个数.

函数的最简形式可通过各种化简方法得到, 然而, 这里所说的函数最简形式是基于 Boolean 逻辑而言的. 因此, 这一"最简"是相对的. 若基于 Boolean 逻辑的函数最简式中存在 $pair(A, B)$, 则说明该函数基于双逻辑还可以进行化简, 且存在的 $pair(A, B)$ 越多, 函数可化简的空间越大[8].

如图 1 所示的两个四变量函数, 基于 Boolean 逻辑的最简式分别为: $f_1=\bar{a}c+a\bar{c}+b\bar{c}d+bcd$、$f_2=a\bar{b}\bar{d}+\bar{a}bd+a\bar{b}c+abc+a\bar{b}c$. 根据定义 4、5 得, 函数 f_1 中最简项 $\bar{a}c$(0-1-) 和 $a\bar{c}$(1-0-)、$b\bar{c}d$(-100) 和 bcd(-111) 之间的汉明距离均为 2, 故记为 $pair(\bar{a}c, a\bar{c})$、$pair(b\bar{c}d, bcd)$. 函数 f_2 中最简项 $a\bar{b}\bar{d}$(00-0) 和 $\bar{a}bd$(01-1)、abc(111-) 和 $a\bar{b}c$(100-) 之间的汉明距离均为 2, 故记为 $pair(a\bar{b}\bar{d}, \bar{a}bd)$、$pair(abc, a\bar{b}c)$. 由此可得基于双逻辑的函数表达式分别为: $f_1=a\oplus c+b\cdot\overline{c\oplus d}$、$f_2=a\cdot\overline{b\oplus d}+a\bar{b}c+a\cdot\overline{b\oplus c}$. 可见, 基于双逻辑实现的函数 f_1 和 f_2 比基于 Boolean 逻辑实现的要简单.

3.2 判断条件和算法实现

基于上述观察, 即可给出相应的判断条件.

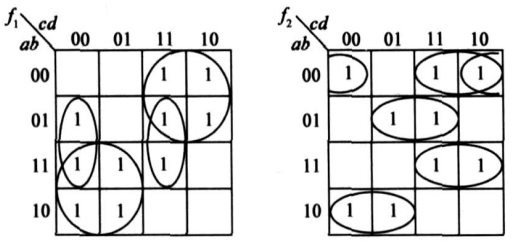

图1　两个四变量函数的卡诺图

判断条件: 对于任意给定的逻辑函数, 若其基于 Boolean 逻辑的最简形式中存在汉明距离为 2 的最简项对 $pair(A, B)$, 即可判定该函数适于用双逻辑实现.

这一判断条件显然是成立的, 因此, 不作证明.

算法实现: 根据判断条件, 可给出如下算法步骤:

(1) 用 C 语言编写一个程序, 将逻辑函数的最小项形式转换为最简项形式; 并将所有最简项建立成一个动态链表, 链表上的每一项均由"0"、"1"、"-"中的一种或几种组成, 如 $\overline{ac}$ (0-1-)、$\overline{abd}$ (00-0) 等; 用两个指针 $p1$、$p2$ 分别指向链表的第一、二项.

(2) 检测 $p1$、$p2$ 所指的两个最简项对应码之间的异或位个数和相等位个数, 分别记为 one_num 和 eq_num, 异或位个数即为最简项之间的汉明距离 H, 异或位的检测用两个对应位相加是否为"1"来实现.

(3) 若汉明距离 H 等于 2 且其他对应位均相等, 即符合 $pair(A, B)$ 形式, 将其从动态链表上删除, 同时最简项个数 N_{SOP} 加 1, 然后返回步骤 2 继续新的检测; 否则就不符合 $pair(A, B)$ 形式, 指针 $p2$ 指向下一项, 同样回到步骤 2 继续检测; 直到所有 $pair(A, B)$ 被探测到为止. 期间最简项删除以及指针的调整通过调用子函数实现.

(4) 若最简项个数 N_{SOP} 的值大于 0, 即可判定该函数适于用双逻辑实现.

算法实现的伪代码如下所示.

```
struct cube{ num; * next;} //存放最简项的结构体
struct cube * creat() //建立动态链表的子函数
struct cube * del() //删除最简项的子函数

{
  struct cube * head, * p1, * p2, * p1_copy;
  unsigned long int one_num, del_flag = 0, eq_num;
  unsigned long int i, H = 0, N_SOP = 0;
  head = creat(); //调用子函数

      p1 = head;
      p2 = p1 -> next;
      while(p1 -> next != ZERO)
      {
  while(p2 -> next != ZERO)
  {
  /* * 检测两个最简项对应码之间异或位个数和相等位个数* * /
      one_num = 0; eq_num = 0;
      for(i = 0; i < num_in; i + + )
      {
        if((p1 -> num[i] - '0' + p2 -> num[i] - '0')
= = 1)
          one_num + + ;
        else if(p1 -> num[i] = = p2 -> num[i])
          eq_num + + ;
      }
      H = one_num;
  if((H != 2) | | (num_in != (eq_num + H)))
    p2 = p2 -> next;
  /* * 若汉明距离为 2, 且其他对应位均相等, 即符合
pair(X, Y)形式* * /
  else
        {
        head = del(head, p1 -> num);
        head = del(head, p2 -> num);
        del_flag = 1;
        p2 -> next = ZERO;
        N_SOP + + ;
        }
  }
if(del_flag = = 1)
  {p1 = p1_copy; p2 = p1 -> next; del_flag = 0;}
else
  {p1 = p1 -> next; p2 = p1 -> next;}
  }
  if(N_SOP > 0) printf("easier by Dual logic \ n");
  else printf("easier by Boolean logic \ n");
  return 0;
}
```

根据上述步骤即可实现对任意给定函数是否适于双逻辑实现的快速判定. 同理本算法的时间复杂度与给定函数的最简项数 N_{SOP} 有关, 其上限为 $O(N_{SOP}^2)$. 在设计实际电路中, 可综合运用上述两种逻辑探测算法对所设计的逻辑电路进行预判断, 以简化电路的综合优化过程, 使其能快速而准确的找到最优逻辑结构. 一般来说, 基于最简项的探测算法的探测过程较为简单快速, 因此, 对于给定逻辑函数, 可先运用基于函数最简项的探测算法对其进行判断, 然后再运用基于最小项的探测算法来判断.

4 实验结果

所提出的算法已用 C 编程实现, 并对 MCNC91 的部分 Benchmark 进行测试. 为便于比较, 将同时给出基于函数最小项和最简项的探测算法的测试结果. 由于这两个算法均为充分条件, 因此, 对于某些用基于最小项的探测算法可正确判断的测试电路, 用基于最简项的探测算法却不一定能判断出来; 反之亦然. 表 1、表 2 分别给出了基于函数最小项和最简项探测算法的具体测试数据.

表 1 基于最小项探测算法的测试数据

样例	输入变量数	输出变量数	N_{aube}	N_R	N_B	N_S
newill	8	1	142	4	66	2
newtag	8	1	234	0	116	2
9sym	9	1	420	0	210	0
life	9	1	140	70	0	0
max46	9	1	62	27	1	6
clip	9	5(1)	256	107	17	8
		5(2)	256	106	20	4
		5(3)	256	97	27	8
		5(4)	256	72	51	10
		5(5)	256	0	127	2
sym10	10	1	837	0	372	93
t481	16	1	42016	11575	9414	38
parity	16	1	32768	16384	0	0
cordic	23	2(1)	7806464	—	—	—
		2(2)	827904	281344	132608	0

表 1 中第一列为测试电路名称, 第二、三列为测试电路的输入、输出变量数, 其中第三列括号内数字表示多输出函数的第几个输出函数, 第四、五、六、七列分别表示电路输出函数的最小项个数及包含的 $sup(R_i, R_j)$、$sup(B_g, B_h)$、S_k 的个数. 表中倒数第二行出现的“—”表示未得到测试结果. 其中 $sup(R_i, R_j)$ 的个数越多, 说明用双逻辑实现比用 Boolean 逻辑实现要简单得越多. 可见, 所测电路中的 life、max46、clip、t481、parity、cordic 均适合用双逻辑实现, 这与其它文献给出的结果相一致[5,6,9].

表 2 中第一列为测试电路名称, 第二、三列为测试电路的输入、输出变量数, 第四列表示电路输出函数的最简项个数 N_{SOP}, 最后一列是函数所含 $pair(A, B)$ 数. 函数所含 $pair(A, B)$ 数越多, 说明用双逻辑实现比用 Boolean 逻辑实现要简单得越多. 可见, 所测电路中的 newtag、9sym、life、max46、clip、sym10、parity、cordic 均适合用双逻辑实现, 这与其它文献给出的结果相一致[5,6,9].

由上述测试结果可见, 两种算法所适用的函数范围具有交集但不重叠, 其中 9sym 和 sym10 这两个电路用基于最小项的算法判断不出来, 这是因为函数不存在 $sup(R_i, R_j)$ 结构, 但用基于最简项的算法很快就能判断出来. 相反, 电路 t481 用基于最小项的算法能判断出来, 但用基于最简项的算法判断不出来. 另外, 电路 cordic 的第一个输出函数用基于最小项的算法探测未能得到测试结果, 但用基于最简项的算法进行探测, 却能快速的得到相应的测试结果. 因此, 将两种算法结合起来使用, 可使所适用的函数范围达到最大.

表 2 基于最简项探测算法的测试数据

样例	输入变量数	输出变量数	N_{SOP}	N_{pair}
newill	8	1	22	0
newtag	8	1	14	3
9sym	9	1	87	21
life	9	1	140	70
max46	9	1	47	16
clip	9	5(1)	21	4
		5(2)	44	8
		5(3)	45	14
		5(4)	37	5
		5(5)	20	0
sym10	10	1	423	210
t481	16	1	481	0
parity	16	1	32768	16384
cordic	23	2(1)	179	80
		2(2)	1027	513

5 结论

本文直接从 XOR 逻辑的特点出发, 即两个汉明距离为 2 的输入项可以由 XOR 逻辑表示, 在文献[7]的基础上, 进一步提出了基于函数最简项的逻辑探测算法, 它同样用于对函数适于双逻辑实现的判断. 且该算法对函数的逻辑判断快速准确, 更弥补了文献[7]存在的算法可能失效的问题. 基于这两种算法均只具有充分性, 因而各自所能判断的函数范围都有一定的局限性, 对于部分适于用双逻辑实现的函数未能判断出来. 但若将两种探测算法结合起来使用, 即先对函数进行基于最简项的快速逻辑探测, 然后再对其进行基于最小项的逻辑探测, 这样可使所适用的函数范围大大增加. 当然, 进一步放宽充分条件将是下一步的研究工作, 而本文所做的工作将是今后研究适于双逻辑实现的自动综合工具的重要技术组成部分.

参考文献:

[1] Xia Yinshui, Wu Xunwei, Almaini A E A. Power minimization of FPRM functions based on polarity conversion[J]. Journal of Computer Science and Technology, 2003, 18(3): 325- 331.

[2] Xia Yinshui, Ye Xien, Wang Lunyao, et al. Novel synthesis method of mixed polarity Reed-Muller functions[A]. Proceedings of third IASTED conference on Circuits, Signals, and Systems[C]. Marina Del Rey, USA, 2005. 148- 153.

[3] 姚茂群,方平,陈偕雄.基于表格法的RM展开系数与或符合展开系数的转换[J].浙江大学学报(理学版),2006,33(4):417-419.
Yao Mao-qun, Fang Ping, Chen Xie-xiong. Conversion of RM expansion coefficients and OR-coincidence expansion coefficients based on tabular method[J]. Journal of Zhejiang University(Science Edition), 2006, 33(4): 417-419. (in Chinese)

[4] D Debnath, T Sasao. A new equivalence relation of logic functions and its application in the design of AND-OR-EXOR networks[J]. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 2007, E90-A(5): 932-940.

[5] T Sasao. A Design Method for AND-OR-EXOR Three-Level Networks[A]. Proceedings of International Workshop on Logic Synthesis[C]. Lake Tahoe, California, 1995. 811-820.

[6] E Dubrova, T Bengtsson. An algorithm for detecting XOR-type logic[A]. Proceedings of 5th International Workshop of Applications of the Reed Muller Expansion in Circuit Design[C]. Starkville, Mississippi, USA, 2001. 271-276.

[7] 夏银水,毛科益,叶锡恩.逻辑函数适于双逻辑实现的探测算法[J].计算机辅助设计与图形学学报,2007,19(12):1522-1527.
Xia Yin-hui, Mao Ke-yi, Ye Xi-en. Detection algorithm for logic function to benefit from dual logic implementation[J]. Journal of computer-aided design & computer graphics, 2007, 19(12): 1522-1527. (in Chinese)

[8] M Maxfield. A Reed-Muller extraction utility[OL]. http://www.highbeam.com/doc/1G1-18371291.html, EDN, 1996-4-11.

[9] D Debnath, T Sasao. GRMIN: A heuristic simplification algorithm for generalized Reed-Muller expression[A]. Proceedings of Asia and South Pacific Design Automation[C]. Makuhari, Japan, 1995. 341-347.

作者简介:

叶锡恩　男,1955年生于浙江宁波,教授,现为宁波大学电路与系统研究所硕士生导师,研究方向为低功耗集成电路设计.
E-mail: yexieng@nbu.edu.cn

毛科益　女,1982年生于浙江宁波,现为宁波大学硕士研究生,研究方向为低功耗逻辑综合与优化.
E-mail: keyimao@yahoo.com.cn

夏银水　男,1963年生于浙江余姚,教授,现为宁波大学电路与系统研究所博士生导师,研究方向为低功耗逻辑综合与优化.
E-mail: xiayinshui@nbu.edu.cn