

基于近似逻辑的不完全指定 FPRM 函数的功耗优化

尹浩凯, 马雪娇, 夏银水*

(宁波大学 信息科学与工程学院, 浙江 宁波 315211)

摘要: 基于精确逻辑的逻辑综合和优化方法已有广泛的研究, 但有时并非需要精确逻辑, 从而可利用这种特性进行关键指标的综合和优化. 本文提出了基于近似逻辑的不完全指定固定极性 Reed-Muller (Incompletely Specified Fixed Polarity RM, ISFPRM) 逻辑函数的逻辑优化方法. 首先建立基于二级逻辑的功耗估算模型和近似电路的设计效能评估模型, 然后提出基于近似逻辑的 ISFPRM 函数的优化算法, 并用遗传算法加以实现. 所提算法应用于 MCNC 标准电路进行测试, 结果表明在一定的面积约束下, 电路每增加 1% 的误差率获得 12%~18% 的功耗优化.

关键词: 近似逻辑; ISFPRM; 极性搜索; GA

中图分类号: TN47; TP391

文献标志码: A

文章编号: 1001-5132 (2019) 03-0028-07

随着 VLSI 制造技术进入到纳米规模, 诸如缺陷率上升和制造工艺不稳定性增加等原因, 致使工艺制程缩小带来的优势开始减弱. 为此, 研究人员提出了错误容忍和缺陷容忍等技术. 然而, 这些技术往往需要以增加额外硬件为代价. 在不增加电路复杂度的情况下, 近几年又提出了误差容忍用于降低制造工艺升级带来的负面影响^[1].

近似计算广泛应用于包括图像、视频、音频、图形、游戏和无线通信的纠错码在内的领域, 也可以应用于电路设计. 基于近似计算的电路设计是指利用误差容忍, 以适当牺牲电路某一指标的性能来提高电路其他指标水平的设计方法. 而基于精确计算的电路设计是指不引入误差的传统设计方法^[2]. 在逻辑级的电路设计中, 通常将基于近似计算的电路设计方法称为近似逻辑, 而基于精确计算的电路设计方法称为精确逻辑. 由于基于近似逻辑设计的电路在合适情况下可以大幅度降低电路的面积、功耗和时延, 因此引起了众多研究者的兴趣, 并展开了相应的研究工作^[3-7]. Shin 等^[3]利用特殊误差对多级逻辑电路进行优化; Wu 等^[4]利用查表法对 FPGA 进行近似逻辑优化; Jiang 等^[5]将 Booth 乘法器的特定内部传统加法器替换成近似加

法器构建近似 Booth 乘法器; Zou 等^[6]提出基于动态规划的两级近似逻辑综合算法; Wu 等^[7]提出在布尔网络中选取特定节点, 并通过逼近法删除节点, 完成多级逻辑电路的优化.

综上, 目前基于近似计算的优化研究主要集中在传统布尔(Traditional Boolean, TB)逻辑. 但是对于某些电路, 如奇偶校验电路、模 2 运算电路及通信电路, 采用基于异或/与(XOR/AND)的 Reed-Muller (RM)逻辑, 可获得更好的可测性、更小的面积和功耗. 因此, 基于 RM 逻辑的综合和优化研究正日益受到重视^[8-9]. Wang 等^[10]提出了一种自适应模拟退火遗传算法, 用于寻找大规模混合极性 RM 逻辑电路的最佳极性. Debnath 等^[11]提出了一种基于真值矢量和权重矢量的 ISFPRM (Incompletely Specified Fixed Polarity Reed-Muller)展开式最小项算法. 然而, 这些方法都是基于传统精确逻辑的 RM 函数的综合和优化. 相对而言, 基于近似逻辑进行 RM 函数的优化研究还相对缺乏.

本文提出一种基于近似逻辑的 ISFPRM 函数优化算法, 以误差率(Error Rate, ER)为误差容忍的衡量指标, 通过芯片分析确定误差率阈值, 采用遗传算法获得给定误差率阈值下面积和功耗的优化.

收稿日期: 2018-07-31.

宁波大学学报(理工版)网址: <http://journal.gd.nbu.edu.cn/>

基金项目: 国家自然科学基金(U1709218).

第一作者: 尹浩凯(1994-), 男, 浙江衢州人, 在读硕士研究生, 主要研究方向: 数字集成电路逻辑综合和优化. E-mail: 448502171@qq.com

*通信作者: 夏银水(1963-), 男, 浙江宁波人, 博导/研究员, 主要研究方向: 数字集成电路逻辑综合和优化. E-mail: xia yinshui@nbu.edu.cn

1 近似逻辑原理及其面积和功耗估算

近似逻辑函数是指引入误差逻辑项的逻辑函数. 假设待优化逻辑函数 A , 引入误差逻辑项后, A 变为 A' , 则 A' 是 A 的一种近似逻辑函数.

对于逻辑函数 $f = \sum_{i=0}^Q p_i$, p_j 和 p_k 为 f 的任意

2 个乘积项, $j \neq k$, 且 $j, k \in \{0, 1, 2, \dots, Q\}$, 若有 $p_j \cap p_k = \emptyset$, 则称 p_j 和 p_k 互不相交. 如果在逻辑函数中乘积项都互不相交, 则称为基于不相交项的逻辑函数.

基于不相交项的不完全指定的 n 变量 TB 逻辑函数可表示为

$$f(x_{n-1}, x_{n-2}, \dots, x_c, \dots, x_0) = \sum_{i=0}^{W-1} (p_i | 1) + \sum_{j=0}^{D-1} (q_j | y_j),$$

$$1 \leq W \leq 2^n, 1 \leq D \leq 2^n, \quad (1)$$

式中: 累加号表示逻辑“或”运算; $(p_i | 1)$ 表示输入为 p_i 变量组合时, f 的取值为逻辑“1”; $(q_j | y_j)$ 为不确定项, 其中 y_j 表示输入为 q_j 变量组合时 f 的取值, 且 y_j 可以取逻辑“1”或“0”. 不相交乘积项之间具有相互独立的性质, “或”和“异或”互换不会改变逻辑函数的正确性, 因此, 可转换为基于不相交项的不完全指定 RM 逻辑函数表示:

$$f(x_{n-1}, x_{n-2}, \dots, x_c, \dots, x_0) = (\oplus \sum_{i=0}^{Z-1} e_i) \oplus$$

$$(\oplus \sum_{j=0}^{V-1} (u_j | d_j)), 1 \leq Z \leq 2^n, 1 \leq V \leq 2^n, \quad (2)$$

式中, $\oplus \sum_{i=0}^{Z-1} e_i$ 和 $\oplus \sum_{j=0}^{V-1} (u_j | d_j)$ 分别为由 $\sum_{i=0}^{W-1} (p_i | 1)$ 和 $\sum_{j=0}^{D-1} (q_j | y_j)$ 不相交项转换得到的异或形式. 由于没有对变量的取值形式进行限定, 因此变量 e_i 和变量 u_j 都可以以原变量或者反变量的形式出现.

如果引入误差项, 式(1)可以转换为下式:

$$f(x_{n-1}, x_{n-2}, \dots, x_c, \dots, x_0) = \sum_{i=0}^{W'-1} (f_i | 1) + \sum_{j=0}^{D-1} (q_j | y_j),$$

$$1 \leq W' \leq 2^n, 1 \leq D \leq 2^n. \quad (3)$$

由于引入了误差项, 使得 $\sum_{i=0}^{W'-1} (p_i | 1)$ 中某些 f 输出的逻辑值由“1”变成逻辑“0”, 而有些 f 的输出逻辑值由“0”变成逻辑“1”, 因此转换为

$\sum_{i=0}^{W'-1} (f_i | 1)$. 由于不确定项的输出 f 的逻辑值发生变化时, 仍为不确定项, 所以不确定项部分和式(1)相同. 考虑到不相交项之间相互独立, 式(3)可以转换为基于不相交乘积项的不完全指定 RM 的近似逻辑函数.

$$f(x_{n-1}, x_{n-2}, \dots, x_c, \dots, x_0) = (\oplus \sum_{i=0}^{Z'-1} r_i) \oplus$$

$$(\oplus \sum_{j=0}^{V-1} (u_j | d_j)), 1 \leq Z' \leq 2^n, 1 \leq V \leq 2^n, \quad (4)$$

式中, $\oplus \sum_{i=0}^{Z'-1} r_i$ 和 $\oplus \sum_{j=0}^{V-1} (u_j | d_j)$ 分别为由 $\sum_{i=0}^{W'-1} (f_i | 1)$ 和

$\sum_{j=0}^{D-1} (q_j | y_j)$ 不相交项转换得到的异或形式.

研究表明, 对于 RM 逻辑函数, 极性与函数表示的繁简关系密切^[10], 进而影响到电路的性能. 因此, 可通过极性优化进而实现对 ISFPRM 函数的优化. 另外对于 ISFPRM 函数, 由于存在不确定项 d_j , 可利用不确定项可取逻辑值“1”或“0”的特性进行进一步优化^[12].

极性优化指通过搜索极性空间得到最佳极性, 而搜索极性空间需要利用极性转换. 逻辑函数极性转换主要方法有表折叠法和列表法^[13], 这些方法都存在转换速度慢、处理规模小等问题. 为此文献[14-15]提出了一种基于不相交项的快速列表技术. 该算法采用并行处理数据且对输入数不敏感, 从而提高了极性转换速度和处理大规模电路的能力. 利用基于不相交项的快速列表技术对式(4)进行极性优化, 可实现对 ISFPRM 函数的逻辑优化.

在精确逻辑函数中添加误差项可获得近似逻辑函数. 这可通过在逻辑函数初始格式, 如真值表、卡诺图中引入误差项来完成, 这种添加方法可以控制误差个数以及误差位置^[2]. 为避免函数展开为最小项占用大量内存, 可以在不相交乘积项中进行误差添加. 在引入误差项时需要作如下判断: 第一, 随机产生误差项是否位于不确定项, 如果是则无需操作, 因为不确定项可以通过指定加入函数; 第二, 产生的误差项是否在精确函数内, 如果在精确函数内, 则剔除误差项, 否则加入该项.

通常用 ER 和误差距离(ES)^[3]来评估近似逻辑电路的设计效能. ER 指近似逻辑电路产生不正确

输出数占总输出的比例,也称错误频率. ES 指近似逻辑电路产生不正确输出的数值和精确电路产生的输出数值之间的汉明距离. 实际工作中一般将 ER 作为评估近似逻辑电路设计效果的常用指标.

例如,若存在一个 n 输入, m 输出的逻辑函数,每个输出都具有 2^n 个的可添加误差项位置,那么 m 个输出一共有 $2^n m$ 个可添加错误位置. 当误差项个数为 k 时,其中有 t 个误差项位于不确定项,那么误差率为 $(k-t)/(2^n m)$.

为了控制误差率以符合应用的需要,引入误差阈值概念. 误差阈值是指在满足应用需要,引入误差项所产生的误差率的最大值. 误差阈值可以通过分析芯片应用程序确定.

以一个基于近似逻辑的 3 输入 FPRM 逻辑函数的化简为例,如图 1(a),精确逻辑下当极性为 001 或 101 时乘积项项数最少,函数 $f=1\oplus B\bar{C}\oplus B\oplus \bar{C}$;在引入误差项 $\bar{A}\bar{B}\bar{C}$,进行近似逻辑优化后,极性为 010 时乘积项项数最少如图 1(b),函数 $f'=A\oplus \bar{B}\bar{C}\oplus \bar{A}\bar{B}\bar{C}$,此时误差率为 1/8,减少乘积项数 1/4;再一次引入误差项 $\bar{A}\bar{B}\bar{C}$ 后,极性为 011、010、000 和 001 时乘积项项数最少如图 1(c),函数 $f''=A$ 误差率为 1/4,减少乘积项数 3/4. 因此在引入误差后可以带来面积的显著减少,也可潜在减少功耗. 另一方面,如果设定误差阈值为 1/8,则以 $A\oplus \bar{B}\bar{C}\oplus \bar{A}\bar{B}\bar{C}$ 函数代替 $1\oplus B\bar{C}\oplus B\oplus \bar{C}$,如果设定误差阈值为 1/4,则以 A 代替原逻辑函数.

(a)

$\begin{array}{c cc} & BC & \\ \hline A & & \end{array}$	00	01	11	10
0	0	1	0	1
1	1	1	1	1

(b)

$\begin{array}{c cc} & BC & \\ \hline A & & \end{array}$	00	01	11	10
0	0	1	0	0(1)
1	1	1	1	1

(c)

$\begin{array}{c cc} & BC & \\ \hline A & & \end{array}$	00	01	11	10
0	0	0(1)	0	0(1)
1	1	1	1	1

在逻辑级优化中, 一般将乘积项个数作为衡量面积的标准. 以比例形式表示各因素的优化程度来综合面积和开关活动性指标, 综合优化率为

$$O_{PA} = \left(\alpha \frac{N_{\text{approximate}}}{N_{\text{accurate}}} + \beta \frac{E_{\text{approximate}}}{E_{\text{accurate}}} \right) / (\alpha + \beta), \quad (12)$$

式中: $N_{\text{approximate}}$ 和 N_{accurate} 分别为近似逻辑函数和精确逻辑函数的乘积项数; $E_{\text{approximate}}$ 和 E_{accurate} 分别为近似逻辑函数和精确逻辑函数的开关活动性; α, β 为权重系数, 通过对其设置, 实现对某一种性能指标的偏好.

2 基于遗传算法的 ISFPRM 函数优化

本文以面积和功耗的综合成本最小化为目标, 是一个多目标优化问题. 函数的近似逻辑优化主要分 3 个步骤: 误差项添加、不确定项添加和极性转换.

由于遍历计算误差项添加和极性及不确定项添加只适用于小函数, 随着函数规模的增大其计算量呈指数上升. 遗传算法(GA)是一种模拟生物进化的启发式算法, 具有搜索空间大、适用于多目标优化等特点, 因此本文拟采用 GA 进行优化.

由于本文涉及误差项添加和极性、不确定项优化, 为此构建了基于遗传算法的 ISFPRM 函数的近似逻辑优化算法. 先进行误差项添加, 然后进行极性转换和不确定项添加, 得到电路的面积和开关活动性返回给 GA. 通过不断迭代, 直到算法收敛或者达到最大代数后返回结果.

2.1 编码及适应度函数

GA 的染色体编码是混合编码, 分为前中后 3 段编码, 前段编码是代表误差项位置, 用无符号整数表示, 中段和后段编码是表示极性与不确定项的取值, 用二进制数表示.

例, 电路的输入为 8, 输出为 1, 编码为 86 125 240 11101111 10. 其中前 3 位编码是误差项所在位置, 第 4 位到第 11 位是极性编码, 最后 2 位是不确定项取值编码. 编码表示将第 86, 125, 240 位最小项作为误差项加入到函数中, 极性转换至 11101111 极性, 并将第 1 个不确定项加入到表达式, 第 2 个不确定项不加入表达式.

将该染色体对应 O_{PA} 作为衡量染色体优劣的

指标, 通过染色体的编码代表的误差项以及极性和不确定项取值, 可以将函数转换成对应的 ISFPRM 表达式, 然后利用 O_{PA} 计算适应度:

$$F_{\text{fitness}}(i) = 1 / O_{PA}(i). \quad (13)$$

例, $\alpha=1, \beta=1$ 时, 假设精确 ISFPRM 函数在 O_{PA} 取最小时, 对应的乘积项数和开关活动性假设分别为 10 和 15.0. 如果存在一个近似 ISFPRM 函数, 其面积为 15, 开关活动性为 20.0, 那么该近似函数对应的 O_{PA} 为 1.41, 适应度为 0.71. 如果其面积为 5, 开关活动性为 7.5, 那么该近似函数对应的 O_{PA} 为 0.50, 适应度为 2.00.

2.2 遗传算子

遗传算子主要包括选择、交叉和变异 3 种基本算子, 其中选择算子采用传统的轮盘赌法, 根据个体适应度与群体适应度之比确定个体被选择的概率产生新种群.

本文算法利用整数编号查找误差项位置, 所以需要采用混合编码形式. 但整数码在交叉和变异时存在颗粒度过大的问题. 颗粒度大小是指一次操作所移动的位数, 如在 32 位操作系统中移动一个 int 型变量, 每次移动数据量为 32 bit, 而移动一个 Boolean 型变量, 每次移动的数据量为 1 bit, 所以认为 int 型变量颗粒度大, Boolean 型变量颗粒度小. 颗粒度过大将导致每次操作的数据量过大而损失一部分信息量, 而这将会导致算法在后期无法收敛.

综上, 本文算法在交叉和变异时将混合编码转换为二进制编码, 以解决整数编码颗粒大的问题. 但是整数编码转换为二进制数编码后, 位数会出现差异. 因此需要通过用“0”进行补齐(表 1).

表 1 混合码转换为二进制码

码制	编码
混合码	86 125 240 11101111 10
二进制	01010110 01111101 11110000 11101111 10

交叉是将 2 个父代个体的部分结构替换重新生成新的个体, 本文采用多点交叉方法. 首先, 根据交叉概率选择待交换个体 A 、 B 作为父代, 然后随机生成多个交叉点, 奇数交叉点到下一个偶数交叉点内, A' 继承 A 的基因, B' 继承 B 的基因. 在偶数交叉点或染色体头部到下一个奇数交叉点内,

A' 继承 B 的基因, B' 继承 A 的基因. 交叉操作见表 2, # 为交叉点所在位置.

表 2 交叉操作

代 个体	基因
父代 A	01010110 01#111101 111#10000 1110111#1 10
B	11011100 10#000000 000#00001 0000111#0 01
子代 A'	11011100 10#111101 111#00001 0000111#1 10
B'	01010110 01#000000 000#10000 1110111#0 01

变异是通过随机改变个体的基因值, 从而使算法具备局部搜索的能力. 本文使用二进制变异, 对于每一个染色体, 遍历所有基因位, 同时产生一个 0~1 的随机数与变异概率比较, 如果小于变异概率则发生变异, 即 0 变 1, 1 变 0, 见表 3.

表 3 变异操作

个体	基因
A	01010110 01111101 11110000 11101111 10
A'	00010110 11111101 11011111 11010111 10

最后仍然需要将二进制编码转换为混合编码, 在二进制编码中将前段编码和中后段编码进行分离. 并将前段二进制码转换为无符号整数, 见表 4.

表 4 二进制码转换为混合码

码制	编码
二进制	00010110 11111101 11011111 11010111 10
混合码	22 126 233 11010111 10

2.3 程序伪代码

结合提出的面积和功耗估算模型, ISFPRM 函数近似逻辑优化流程, 以及 GA 中的编码, 适应度计算和遗传操作, 就可以实现基于近似逻辑的 ISFPRM 函数的优化. 优化算法的伪代码如下:

GA- ISFPRM-OPT

GA- ISFPRM-OPT(. pla)/ISFPRM 函数近似逻辑优化

```

Int MG; //最大代数
Int Popsiz;
Int Threshold_Errors;
Disjoint_Term=pla_to_disjoint_term(. pla);
Oldpop=initialize(Popsiz, Threshold_Errors);
Com_fitness(Oldpop, Disjoint_Term);
find_best_individs (Oldpop, Best_Chrom,
    Disjoint_Term);
While(MG!=0)

```

```

for(i=0: Popsiz)

```

```

    Mate1=secelet(Oldpop, Popsiz);
    Mate2=secelet(Oldpop, Popsiz);
    crossover_operation(Mate1, Mate2,
        Newpop)
    Muttled1=mutation(Mate1, Mate3,
        Newpop);
    Muttled2=mutation(Mate1, Mate3,
        Newpop);
    Com_fitness(Newpop);
    find_best_individual(Newpop,
        Best_chrom, Disjoint_Term);
    update_population(Oldpop, Newpop);
MG--;

```

```

Return Best_chrom. area;

```

```

Return Best_chrom. sw;

```

```

Return Best_chrom. erros;

```

COM_FITNESS(pop, Disjoint_Term)

```

Disjoint_Termapproximate=add_error_term(pop,
    Disjoint_Term);
Disjoint_TermDC_approximate=add_DC_term
    (Disjoint_Termapproximate, pop);
ISFPRMpolarity=disjoint_term_to_isfprm(pop,
    Disjoint_TermDC_approximate);
ComPA (ISFPRMDC_polarity);
Return Com_fitness(power,area)
Return area

```

```

Return sw;

```

算法从 GA-ISFPRM-OPT() 开始, 计算适应度函数时调用 COM_FITNESS(), 代数通过 MG 表示, 每执行一代, MG 减 1, 直到 MG 等于 0 时迭代结束.

3 实验及实验结果分析

所提算法用 C 语言加以实现. 在 windows 7 的 64 位操作系统, 16 GB 内存和 3.6 GHz 处理器下, 应用于 MCNC (Micro-electronics Center of North Carolina) 基准电路进行测试. 并选取种群规模为 800, 代数为 150, 交叉率为 55%, 变异率为 8%, 可以在适当时间内得到一个较优的解. 信号概率取值为 (0, 1), 跳变密度的取值区间为 (0, 2), 根据输入个数的不同, 随机产生的信号概率密度为 0.04,

0.10; 0.65, 0.54; 0.12, 1.21; 0.89, 1.58; $\dots$, 则三变量的输入信号概率和跳变密度为 0.04, 0.10; 0.65, 0.54; 0.12, 1.21; 其他变量以此类推。

当适应度函数中的权值为 $\alpha=1, \beta=0$ 时, 在不同误差阈值情况下, 以面积最小为目标进行近似逻辑优化, 结果见表 5, 其中 R 为 ER 的大小。

表 5 以面积为目标近似逻辑化简实验结果

电路名称	输入/输出/ 乘积项数	RM 表达式的乘积项数			
		$R=0$	$R=1\%$	$R=1.5\%$	$R=2\%$
newtag	8/1/8	8	7	7	6
newwill	8/1/8	13	11	11	10
newcwp	5/8/8	8	7	7	6
newapla2	6/7/7	7	6	5	5
dekoder	5/8/8	10	9	8	8
rd53	4/7/16	23	18	16	16
squar5	5/8/39	23	21	20	19
misex1	8/7/32	25	23	22	21
节省面积/%			12.82	17.95	22.22

$R=0$ 表示函数经过精确逻辑化简得到的乘积项数, $R=1\%, 1.5\%, 2\%$ 分别表示在 $1\%, 1.5\%, 2\%$ 误差阈值下函数经过近似逻辑化简得到的乘积项数。

节省面积表示在不同误差阈值情况中面积改进百分比, 定义为:

$$S_A = 1 - N_{R>0} / N_{R=0} \quad (14)$$

由表 5 可见, 总体上误差阈值和优化效果成正比, 但是某些电路随着误差阈值的增大, 化简效果并未改善, 如 newwill、newcwp、dekoder、newapla2 电路, 这是因为近似逻辑通过添加误差项进行化简, 而只有符合条件的误差项才可以简化电路。如

果在一种误差项组合已经取得较优解, 再增加误差项反而导致冗余, 不利于优化, 所以出现误差阈值上升, 但化简效果不变的情况。而随着阈值进一步增加, 符合条件误差项增多可能突破瓶颈, 化简效果将再次提升, 如 newwill、newcwp。

当 $\alpha=1, \beta=1$, 以综合成本最小为目标近似逻辑优化数据结果见表 6。节省综合成本表示综合优化效果, 定义为:

$$S_{PA} = 1 - \frac{1}{2} \left(\frac{N_{R>0}}{N_{R=0}} + \frac{E_{R>0}}{E_{R=0}} \right) \quad (15)$$

由表 6 可得, 在这种面积约束情况下, 电路每增加 1% 的误差率可以获得 $12\% \sim 18\%$ 的功耗减少。与面积优化不同, 随着误差阈值提高, 面积和开关活动性其中一个目标优化效果下降, 甚至如 rd53 和 newapla2 优化效果为负, 这是通过牺牲单一目标来提高多目标优化效果。多目标优化时出现面积相同开关活动性不同的情况, 如 newwill。原因是在不同极性下, 虽然乘积项项数相同但是函数发生改变, 即结构发生改变, 造成开关活动性不同。

综上, 本文提出的算法可以较好地解决在给定误差阈值情况下, ISFPRM 函数基于近似逻辑的优化问题, 以较少和可控的误差项个数, 实现 ISFPRM 电路的近似逻辑优化。

4 结语

本文提出了基于近似逻辑的 ISFPRM 函数的功耗优化算法, 主要由 2 方面构成: (1) 近似逻辑原理和 ISFPRM 函数的面积和功耗估算; (2) 基于 GA 的 ISFPRM 函数优化。实验结果表明, 与传统的

表 6 以综合成本为目标近似逻辑优化实验结果

电路名称	$R=0$		$R=1\%$		$R=1.5\%$		$R=2\%$					
	面积	开关活动性	面积	开关活动性	时间/s	面积	开关活动性	时间/s	面积	开关活动性	时间/s	时间/s
newtag	8	7.42	7	6.68	56.321	7	6.68	71.536	6	5.50	75.635	
newwill	14	5.60	13	5.27	317.660	13	4.02	337.738	12	3.83	348.205	
newapla2	8	5.04	7	5.46	177.813	7	5.46	181.043	6	4.82	173.823	
newcwp	8	11.86	6	7.97	49.453	5	8.06	81.233	5	7.77	83.556	
dekoder	13	24.90	10	23.02	122.701	11	20.96	114.130	8	21.24	124.895	
rd53	26	36.25	30	22.21	252.856	21	25.18	279.548	20	19.18	278.693	
squar5	23	37.83	22	35.22	233.588	20	35.16	256.810	19	30.47	256.811	
misex1	37	67.32	33	60.54	465.735	35	52.56	510.356	33	38.21	501.135	
节省综合成本/%				19.15			24.05			29.46		

ISFPRM 函数优化相比, 引入少量误差后可以获得面积和功耗的明显优化。

参考文献:

- [1] Han J, Orshansky M. Approximate computing: An emerging paradigm for energy-efficient design[C]. 18th IEEE European Test Symposium, 2013:1-6.
- [2] Shin D, Gupta S K. Approximate logic synthesis for error tolerant applications[C]. Design, Automation & Test in Europe Conference & Exhibition, 2010:957-960.
- [3] Shin D, Gupta S K. A new circuit simplification method for error tolerant applications[C]. Design, Automation & Test in Europe Conference & Exhibition, 2011:1-6.
- [4] Wu Y, Hen C, Jia Y, et al. Approximate logic synthesis for FPGA by wire removal and local function change[C]. IEEE Design Automation Conference. 2017:163-169.
- [5] Jiang H, Han J, Qiao F, et al. Approximate radix-8 booth multipliers for low-power and high-performance operation [J]. IEEE Transactions on Computers, 2016, 65(8):2638-2644.
- [6] Zou C, Qian W, Han J. DPALS: A dynamic programming-based algorithm for two-level approximate logic synthesis [C]. IEEE 11th International Conference on ASIC, 2015:1-4.
- [7] Wu Y, Qian W. An efficient method for multi-level approximate logic synthesis under error rate constraint [C]. Proceedings of the 53rd Annual Design Automation Conference, 2016:128.
- [8] Rafiev A, Mokhov A, Burns F P, et al. Mixed radix Reed-Muller expansions[J]. IEEE Transactions on Computers, 2012, 61(8):1189-1202.
- [9] Das A, Pradhan S N. Shared Reed-Muller decision diagram based thermal-aware AND-XOR decomposition of logic circuits[J/OL]. VLSI Design, 2016, 2016:3191286 [2018-03-17]. <http://dx.doi.org/10.1155/2016/3191286>.
- [10] Wang X, Li M, He Z, et al. PAOA: A power and area optimization approach of Reed-Muller logic circuits[C]. IEEE 60th International Midwest Symposium on Circuits and Systems, 2017:1394-1397.
- [11] Debnath D. Exact minimization of FPRMs for incompletely specified functions by using MTBDDs[J]. IEICE Transactions on Fundamentals of Electronics, Communications & Computer Sciences, 2005, 88(12): 3332-3341.
- [12] 汪鹏君, 汪迪生, 蒋志迪, 等. 基于 PSGA 算法的 ISFPRM 电路面积与功耗优化[J]. 电子学报, 2013, 41(8):1542-1548.
- [13] Tan E C, Yang H. Fast tabular technique for fixed-polarity Reed-Muller logic with inherent parallel processes[J]. International Journal of Electronics, 1998, 85(4):511-520.
- [14] 王玉花, 王伦耀, 夏银水. 基于不相交项并行列表技术的 FPRM 实现[J]. 电子与信息学报, 2014, 36(9):2258-2264.
- [15] 李新文, 夏银水, 王伦耀. 一种改进的混合极性列表技术[J]. 宁波大学学报(理工版), 2015, 28(1):42-46.

Power optimization of incompletely specified FPRM function based on approximate logic

YIN Hao-kai, MA Xue-jiao, XIA Yin-shui*

(Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo 315211, China)

Abstract: The logic synthesis and optimization methods based on accurate logic have been extensively studied. However, sometimes accurate logic may not be technically needed. As a result, this undesired requirement can be taken advantage of to synthesize and optimize other key issues. In this paper, an optimization method based on approximate logic for the Incompletely Specified Fixed Polarity Reed-Muller (ISFPRM) function is proposed. Firstly, the evaluation model of power consumption based on the two-level logic and the estimation model of design efficiency aiming at the approximate circuit are established. Then an optimized algorithm based on the approximate logic ISFPRM function is proposed and implemented in genetic algorithm. The proposed algorithm is tested under MCNC benchmarks. The experimental results show that under area constraints, every 1% increase in the error rate of the circuit can achieve power saving by 12% to 18%.

Key words: approximate logic; ISFPRM; polarity search; GA

(责任编辑 韩 超)