一种 RTL 级数据通路 ODC 低功耗优化算法

孟建熠,丁永林,严晓浪,葛海通
(浙江大学超大规模集成电路设计研究所,浙江杭州 310012)

摘 要: 本文提出了一种具有高计算效率和低硬件开销的门控时钟低功耗优化算法.该算法在 RTL 级搜索数据通路的不可观察性(Observability Don't Care).采用 RTL 级逻辑信号总线 ODC 模型和基于路径 ODC 的有向图遍历模型,减少了 ODC 计算负荷,提升了计算效率,使 ODC 适用于超大规模集成电路的低功耗优化.引入数据通路 ODC 条件概率作为门控信号产生的重要依据,对 ODC 条件概率高的通路优先插入门控逻辑,可以极低硬件开销实现高效门控时钟网络.实验结果显示,本算法与传统 ODC 算法相比计算负荷平均降低 8 倍,功耗平均下降 12.35%,面积开销平均减少 13.44%.

关键词: 数据通路低功耗;总线 ODC 模型;路径 ODC 模型;ODC 条件概率
中图分类号: TN47 **文献标识码:** A **文章编号:** 0372-2112 (2010) 07-1654-06

A RTL Level ODC Algorithm for Data Path Low Power Optimization

MENG Jian-yi, DING Yong-lin, YAN Xiao-lang, GE Hai-tong
(*Institute of VLSI Design Zhejiang University, Hangzhou, Zhejiang* 310027, *China*)

Abstract: This paper proposes an efficient and low-overhead optimization algorithm for clock-gating based low power design. It searches the Observability Don't Care (ODC) conditions of data path at RTL level. It analyzes the ODC conditions of RTL logic level signal which is defined as Bus-ODC model. Data path is cut down into several short paths and computed ODC condition separately, which is defined as Path-ODC model. These two models can reduce the ODC computation load and improve the computation efficiency, and make it sufficient for VLSI low power optimization. Probability of ODC conditions is also proposed in this paper and used as an important basis of clock gating logic synthesis. It is preferred to insert clock gating logic into the data path with high probability of ODC condition. Probability driven clock gating logic synthesis improves the efficiency of clock gating network with tiny hardware overhead. Compared with the result of traditional ODC algorithm, computation load of RTL level ODC algorithm is only one-eight of that, power can be reduced by 12.35% and hardware overhead is cut down by 13.44% in average.

Key words: low power data path; bus-ODC model; path-ODC model; probability of ODC condition

1 引言

随着工艺不断进步,集成电路的集成度和复杂度不断提高,芯片及其嵌入式系统的功耗不断增长[1].数据通路是集成电路的重要组成部分,在嵌入式处理器中数据通路功耗约占总功耗的 36%[2].由于数据通路通常随着系统功能的变化而变化,其形式灵活、差异性大,特征提取相对比较困难,因此数据通路的低功耗技术是低功耗设计领域的难点.

数据通路低功耗优化算法的主要目标是寻找通路的空闲状态,消除空闲状态下冗余计算产生的功耗.优化算法总体分为三大类:有限状态机分析法、符号分析法和 ODC 分析法.有限状态机分析法[3]将输入的变化建模为有限状态机模型,通过分析和提取可使输出保持不变的状态,关闭数据通路冗余计算.其缺点是状态的存储与表达以输入宽度 2 的指数级增长,运算和存储量往往超出实际可控制的范围.符号分析法以分析数据通路的布尔函数关系来提取门控条件,方法有二元判决图(BDD)和代数决策图(ADD).BDD 或 ADD 均基于底层门级电路的布尔函数,计算负荷高且占用大量内存空间,极易造成内存空间爆炸,实践表明该方法只适合于规模较小的电路[4].ODC 分析法以统一的 ODC 算术模型对通路的临时不可观察条件进行分析,仅分析提取数据通路中可进行通路控制的逻辑,进而实施低功耗策略.ODC 分析方法算法相对简单,内存占用量较低,且硬件开销也较小,基本适合超大规模集成电路低功耗设计需要.文献[5]通过反向追踪数据通路的拓扑结构进行 ODC 条件的提取和综合,得到门控时钟的控制逻辑.该方法存在两方面问题:首先其 ODC 的计算基于网表,逻辑上相关和相交的数据通路 ODC 条件需被重复计算,

收稿日期:2008-11-03;修回日期:2010-03-16

运算负荷大且效率低下;其次,在 ODC 条件提取中,仅通过逻辑复杂度等静态属性决定门控时钟的插入容易降低门控时钟对功耗的控制效率.文献[6]提出了一种仅通过分析数据通路动态仿真结果,寻找与 ODC 条件最接近的一个控制信号作为 ODC 控制信号的近似方法,虽然简化了 ODC 的提取,但是该方法过于依赖仿真结果,且由于 ODC 条件被过分简化,导致门控单元对功耗的控制力下降.

本文提出了一种基于 RTL 级电路描述的 ODC 分析模型及其实现方法,其主要特征包括:在 RTL 级电路描述中,提取和合并相同逻辑通路,采用基于总线 ODC 计算模型对逻辑通路进行 ODC 联合计算,减少需进行 ODC 计算的通路总量;将完整逻辑通路划分为若干短逻辑通路,分别对短逻辑通路进行 ODC 条件计算,并采用深度优先的有向图路径遍历算法,对路径重合或逻辑上相交的通路直接复用重合部分通路的 ODC 计算结果,避免了相交路径 ODC 条件的重复计算,提高 ODC 的计算效率;引入逻辑通路的 ODC 概率作为 ODC 条件提取的依据,着重优化 ODC 概率高的数据通路,提高门控时钟对功耗的整体控制效力,有效平衡面积和功耗两个因素.该算法简化了 ODC 计算过程,降低了计算负荷,增强了门控时钟优化的方向性,提高了门控时钟效率.

2　算法理论

数据通路的功能可建模为 $z=f(x)$,x 表示输入数据,$f(x)$表示数据通路的行为,z 表示输出结果.本文采用参考文献[4]的原始 ODC 模型,如式(1)所示:

$$\mathrm{ODC}(x)=\mathrm{ODC}_{f(x)}(x)+\mathrm{ODC}(z) \tag{1}$$

若将数据通路扩展为一组输入信号与一组输出信号的计算关系,设 $x[n-1:0]$表示逻辑上相关的 n 个输入信号,$z[m-1:0]$表示 m 个逻辑上相关的输出结果信号,则对于输入信号 $x[n-1:0]$的 ODC 公式定义如式(2)所示:

$$\begin{aligned}\mathrm{ODC}(x[n-1:0]) &= \prod_{i=0}^{n-1}\mathrm{ODC}(x_i)\\ &= \prod_{i=0}^{n-1}\left(\mathrm{ODC}_{f(x_i)}(x_i)+\prod_{j=0}^{m-1}\mathrm{ODC}(z_j)\right)\end{aligned} \tag{2}$$

联合式(1)发现,计算一组逻辑上紧密相连的输入信号 $x[n-1:0]$的 ODC,首先需计算出各输入位关于运算函数 $f(x)$与输出结果 z 的 ODC,然后对各个输入位的 ODC 取逻辑与.

在实际的数据通路设计中,数据通常按组(总线)的方式进行运算与传递,因此数据通路功能函数 $f(x)$对总线信号的各比特的功能相同,其 ODC 也相同,因此 $f(x)$对于输入 $x[n-1:0]$的 ODC 可抽象为 $\mathrm{ODC}_{f(x)}(x[n-1:0])=\mathrm{ODC}_{f(x)}(x[i])$.与输入组信号类似,输出组信号 $z[n-1:0]$各数据位的 ODC 也可以被统一抽象为 $\mathrm{ODC}(z[n-1:0])=\mathrm{ODC}(z[i])$,则输入组信号的在逻辑层面的 ODC 计算可表示为式(3):

$$\begin{aligned}\mathrm{ODC}(x[n-1:0]) &= \mathrm{ODC}_{f(x)}(x[n-1:0])\\ &+\mathrm{ODC}(z[n-1:0])\end{aligned} \tag{3}$$

这种逻辑层的抽象 ODC 模型定义为总线 ODC 计算模型,极大的简化了数据通路的建模.如图 1 所示的电路如根据式(2)推理,其 ODC 计算通路达 32 条,而采用本文提出的总线 ODC 模型(式(3)),ODC 计算通路仅 1 条,路径计算总量减少为传统算法的 $1/n$(n 表示输入宽度).RTL 级的抽象描述为数据通路的总线提取与合并提供了良好的支持,总线 ODC 模型将有效减少 ODC 的运算负荷,极大的简化计算复杂度,提高 ODC 计算效率.

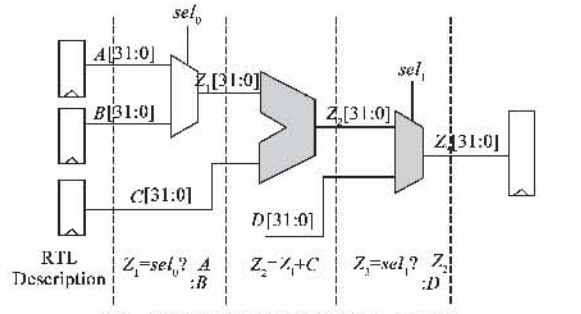

图1　典型总线数据通路及其RTL描述

传统 ODC 计算基于完整的物理路径,路径之间 ODC 计算相互独立.但在实际电路中,路径与路径存在着大量的交叉与共享的短路径.如图 1 中的路径 $A\rightarrow Z3$ 与路径 $B\rightarrow Z3$ 之间存在相交的短路径 $Z1\rightarrow Z3$.将整条数据通路划分为若干短的路径,相交路径复用部分短路径的 ODC 结果,将极大的减少 ODC 条件提取的次数,提升全数据通路 ODC 计算效率.设 x 为路径输入,x_n 为数据通路上各个中间节点,z 为输出,则可获得基于路径 ODC 计算模型如式(4)所示:

$$\begin{aligned}\mathrm{ODC}_{x\rightarrow z}(x) &= \mathrm{ODC}_{x\rightarrow x_1}(x)+\mathrm{ODC}_{x_1\rightarrow x_2}(x_1)+\cdots\\ &+\mathrm{ODC}_{x_n\rightarrow z}(x_n)\\ &=\mathrm{ODC}_{f_0(x)}(x)+\mathrm{ODC}(x_1)+\mathrm{ODC}_{f_1(x_1)}(x_1)\\ &+\mathrm{ODC}(x_2)+\cdots+\mathrm{ODC}_{f_n(x_n)}(x_n)\\ &+\mathrm{ODC}(z)\end{aligned} \tag{4}$$

在式(4)中,一条长路径被切分为若干逻辑上首尾相接的较短路径,路径的串联特性决定各短路径的 ODC 逻辑和即为长路径的 ODC,即只要某段通路被关闭(ODC 为 1),整条通路被整体关闭.由于数据通路中各个节点对于信号传输是透明的,因此中间节点的 $\mathrm{ODC}(x_n)$为 0,则路径 ODC 公式可以简化为式(5).数据

通路 ODC 的计算简化为各个独立短路径传递函数的 ODC 和输出 z 的 ODC 计算.提取数据通路的独立短路径,计算独立短通路的 ODC 结果,然后对短通路的 ODC 进行综合得到完整数据通路 ODC 结果是本文的提升 ODC 计算效率的又一有效方法.逻辑结构越复杂,相交的路径越多,则通过路径 ODC 模型提升性能越多.

$$\mathrm{ODC}_{x\to z}(x)=\mathrm{ODC}_{f_0(x)}(x)+\mathrm{ODC}_{f_1(x_1)}(x_1)+\cdots+\mathrm{ODC}_{f_n(x_n)}(x_n)+\mathrm{ODC}(z) \tag{5}$$

ODC 是数据通路电路级的静态属性,在 ODC 条件满足情况下,数据通路的相关冗余操作均可被消除,从而降低功耗.数据通路 ODC 条件何时成立及其概率大小是电路动态属性,其受到控制逻辑属性以及输入激励的共同影响.数据通路的 ODC 概率越高表示空闲状态越多,则通过关闭数据通路所降低的功耗越明显.本文提出的 RTL 级 ODC 低功耗优化算法在进行低功耗优化时引入了数据通路的 ODC 概率作为是否插入门控时钟的重要依据,优先优化 ODC 概率高的数据通路,以增加低功耗设计的针对性,可利用更少的硬件资源开销实现对功耗的高效控制.

3 算法实现

本文提出的数据通路 ODC 功耗优化算法基于对 RTL 逻辑信号的 ODC 分析与计算,不仅可保持 ODC 计算的准确性,同时可提高 ODC 计算的效率,其总体流程如图 2 所示.主要步骤包括数据通路的提取,数据通路的结构分析与 ODC 计算,ODC 条件概率的生成,ODC 控制逻辑的综合及门控时钟的实现.

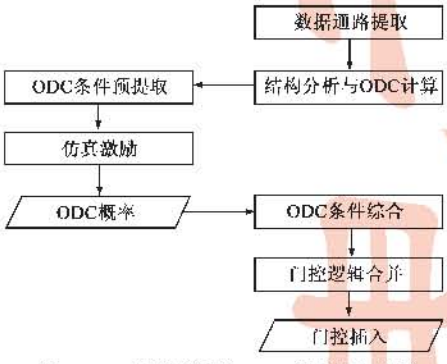

图2 RTL级数据通路ODC功耗优化流程

在 ODC 计算模型中,数据通路的起点和终点均为寄存器,提取数据通路首先从提取 RTL 描述中的寄存器开始.RTL 描述中寄存器采用时钟沿触发的寄存器变量进行建模,通过搜索与匹配即可得到电路中的所有寄存器.然后运用 RTL 级结构分析工具 Vperl[7]获得数据通路的结构信息.本文提出的 ODC 功耗优化算法将逻辑上相关的寄存器比特在 RTL 级合并抽象建模成总线模型.

在实际的电路中,计算单元的 ODC 条件为 1 的发生概率非常低或者恒为 0,如乘法功能单元对于 x_1 的 ODC 条件为"$x_2==0$";加法器由于输入透明传输至输出其 ODC 恒为 0.因此在本文提出的数据通路 ODC 计算中,将忽略计算单元的 ODC 计算,仅提取数据通路中的选择控制逻辑.因此忽略路径中运算单元 $f(x)$ 对于输入 ODC 条件的影响,仅分析通路中的选择器对 ODC 计算的影响,可保证结果精确的前提下,有效简化路径 ODC 的计算.

运算单元和选择器按照特定顺序的组合构成复杂的数据通路.在 ODC 计算流程中,数据通路从终点往起点的方向进行回溯计算.从终点寄存器的角度观察,数据通路可以被层次化为选择器与运算单元不断交替逻辑网表,如图 3 所示.由于数据通路的 ODC 计算顺序固定从目的寄存器到起始寄存器,故逻辑网表可以被抽象为顶点为目的寄存器的有向图数据结构[8],如图 3 所示,m_n 为选择器,c_n 为运算单元,z 为路径终点寄存器,a、b、c、d 和 e 分别表示路径起始寄存器.

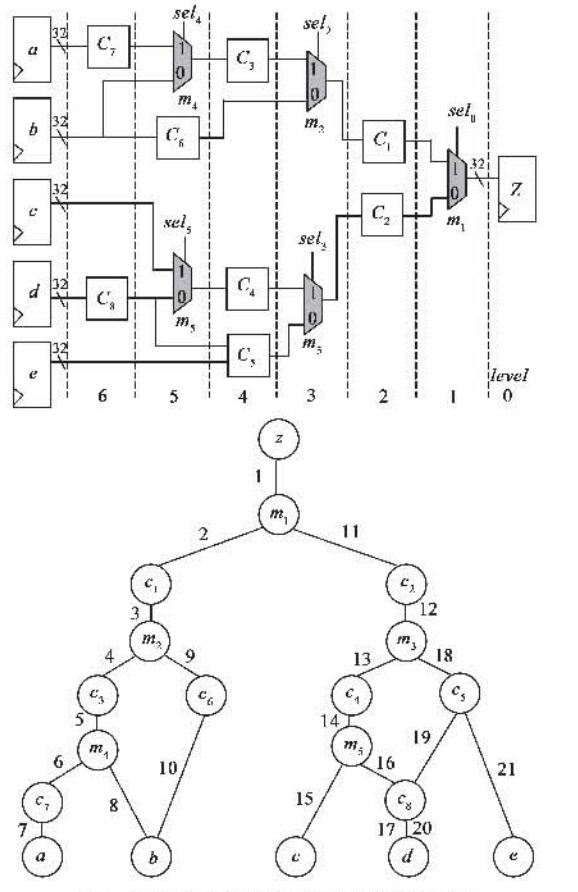

图3 数据通路的层次化结构及有向图遍历

在有向图的数据结构中,运算单元和选择器组成了图的各个节点,运算单元与选择器之间的通路构成了图的边.通路 ODC 的计算过程,演变为遍历图的顶点

(路径终点寄存器)到终点(路径起始寄存器)的所有逻辑路径,计算路径中的 ODC 值,然后进行综合的过程.在路径的遍历中,选择器和运算单元均可产生路径分支,选择器的 ODC 值需进行独立计算,运算器 ODC 的值恒为 0.

```
1:  ODCTraverse (RTLFilelist, CurrentNode, Level) {
2:    Level + + ;
3:    BranchNum = AnalysisBranches(RTLFilelist,
4:                                 CurrentNode,
5:                                 BranchElement);
6:    for(i = 0; i < BranchNum, i + + ){
7:      if(BranchElement[i].type = = REG) {
8:        FullPathODCSave(Level - 1);
9:        return;
10:     }
11:     else if(BranchElement[i].type = = MUX) {
12:         ODCCondition = ODCExtract();
13:         ODCConditonSave(ODCCondition, Level);
14:         CurrentNode = BrancheElement[i] - > Node;
15:         ODCTranverse(RTLFilelist, CurrentNode, Level);
16:     }
17:     else {
18:         ODCConditonSave(0, Level);
19:         CurrentNode = BrancheElement[i] - > Node;
20:         ODCTranverse(RTLFilelist, CurrentNode, Level);
21:     }
22: }
```

图 4　路径 ODC 计算的伪代码

在图 4 所示的路径 ODC 计算伪代码中,完整路径的 ODC 计算通过对各短路径进行深度优先的 ODC 递归计算所得,图 3 各边上的数字表示深度优先算法的计算顺序.在每个节点处,首先通过 *AnalysisBranches*()函数搜索当前节点的分支数量同时返回各个子分支节点的属性;然后根据节点属性分别对各个子分支进行 ODC 递归计算.在计算中,若子分支节点为选择器则通过 *ODCExtract*()提取 ODC 条件,将 ODC 条件通过 *ODCConditionSave*()函数保存至 ODC 条件堆栈中,然后递归进入下一层寻找 ODC 条件,直到找到数据通路的起点寄存器为止.若分支的子节点为运算单元,则忽略 ODC 的计算,直接递归进入下一层计算.ODC 条件堆栈是以结构层次(Level)为索引的数据堆栈,当前层存在 ODC 条件,则压入 ODC 条件,否则压入 0 表示无 ODC 条件.若遍历到节点为数据通路起始寄存器(REG),根据当前的层次信息,从 ODC 条件堆栈中读出之前保存的所有前续短通路 ODC 条件,然后利用函数 *FullPathODCSave*()进行临时保存,当前通路 ODC 条件提取结束.ODC 的路径搜索与计算过程中,采用了深度优先的算法和 ODC 条件堆栈,该方法可使得有相同短路径的数据通路的计算被合并,降低数据通路遍历过程中的计算开销.

本文在 ODC 条件提取的基础上,增加了通路 ODC 概率作为门控时钟插入的辅助条件,优先对 ODC 概率高的数据通路增加门控,提升门控时钟的效率.通过深度优先算法得到 ODC 的基本条件后,对这些基本条件分别进行概率分析,然后综合得到整条数据通路的 ODC 条件概率.本文采用 SystemVerilog 的 Coverage 分析方法对各个 ODC 条件进行监测,对每个子条件自动产生 coverpoint.通过对电路施加随机仿真激励,自动获取各个通路中各个短路径的 ODC 概率.

从 ODC 条件到门控信号产生的过程如图 5 的伪代码所示.路径 ODC 概率与逻辑复杂度共同作为控制逻辑产生的依据.存在 ODC 条件的数据通路中,只有当通路的 ODC 概率大于 ODC 概率阈值 P_T 时,ODC 的条件才被综合.用户可根据资源约束和功耗约束调整阈值 P_T 大小,从而达到资源与功耗的平衡.对于逻辑复杂度大的 ODC 条件,本文采用了循环近似逼近的方法进行提取,即如果综合后的实际逻辑复杂度大于逻辑复杂度阈值 LC_T,则通过 *RmMiniProbability*()函数删除通路中 ODC 概率最小的子条件,重新综合 ODC 条件,直到逻辑复杂度低于 LC_T 或者 ODC 条件为 0.与传统 ODC 条件逻辑复杂度超出阈值就丢弃的方法相比,该方法对于控制逻辑复杂的数据通路仍可保证一定的功耗控制,可在有限硬件资源约束下,最大限度控制电路功耗.

```
1:  ODCLogicGen (RegisterODCRaw) {
2:    index = 0;
3:      while(CurReg = GetReg(RegisterODCRaw, index)){
4:        If(CurReg - > ODCProbability > P_T){
5:          LogicComplex = GetODCComplex(CurReg);
6:          While(LogicComplex > LC_T &&
7:                CurReg - > ODCConditionNum ! = 0){
8:            RmMiniProbability(CurReg);
9:            LogicComplex = GetODCComplex(CurReg);
10:       }
11:       if(CurReg - > ODCConditionNum ! = 0)
12:           GenODCCondition(CurReg);
13:       }
14:       index + + ;
15:     }
16: }
```

图 5　ODC 条件综合过程伪代码

4　实验结果与分析

本文选择 ITC99 基准电路[9]作为目标电路,采用本文提出的基于概率的总线和路径 ODC 模型对其进行优

化,并将结果与文献[6]相比.首先运用总线和路径模型进行 ODC 条件提取与分析,获得基于总线短路径 ODC 分析模型的计算负荷,并与传统 ODC 分析模型进行比较.然后自动生成概率分析模型并施加随机激励,分析各个基准电路的 ODC 条件概率分布情况.最后结合 ODC 条件概率进行门控时钟的优化,在各基准电路中插入门控时钟,并与传统 ODC 优化算法优化结果进行功耗和面积的比较.

表1 总线 ODC 模型与传统 ODC 模型运算负荷对比

基准电路	目标路径总量			ODC 计算总量		
	传统	本文	改进(倍)	传统	本文	改进(倍)
bch_03	242	86	2.81	700	101	6.93
bch_04	286	38	7.53	550	40	13.75
bch_05	238	58	4.10	417	72	5.79
bch_07	438	61	7.18	635	70	9.07
bch_08	132	27	4.89	195	34	5.74
bch_09	255	37	6.89	512	46	11.13
bch_10	300	165	1.81	462	204	2.26

改进 ODC 计算策略,降低 ODC 计算负荷是本文提出的总线短路径 ODC 优化算法的重要目标.分别设计实现了基于文献[5]的传统 ODC 算法和基于总线短路径 ODC 模型的算法,并对各个基准电路进行分析优化,获得两种算法各自的运算负荷.由于仿真平台硬件和系统差异,仿真时间无法真实反映两种算法的运算负荷,本文以 ODC 目标路径总量和 ODC 计算总量作为优化算法运算负荷量化比较的重要参数.ODC 目标路径总量指为完成电路优化所需搜索的所有路径的总和;ODC 计算总量指完成电路优化所需进行 ODC 条件计算次数的总和,以 ODC 提取函数被调用的次数来量化.表1显示基于总线 ODC 算法模型的目标路径总量平均降低为传统 ODC 算法的 1/5 左右;ODC 计算总量平均降低为传统 ODC 算法的 1/8.结果显示通过总线 ODC 模型合并计算路径,共享相交路径的 ODC 计算结果可有效降低运算负荷.

在 ODC 条件提取之后,为每个 ODC 条件自动生成了 Coverage 分析模型.然后对各个基准电路施加随机测试激励,联合 Coverage 分析模型进行功能仿真.最后利用 SystemVerilog 的 Coverage 统计工具,获得通路 ODC 条件成立的概率.对 ODC 的条件概率进行整理,获得各个基准电路的 ODC 条件概率算术平均值分布情况如图6所示.各个测试基准的平均 ODC 条件概率在 27.59%与 50.74%之间.

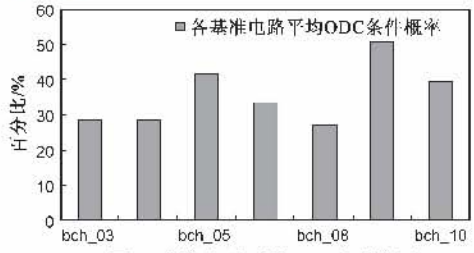

图6 基准电路平均ODC条件概率

根据电路中 ODC 条件概率的分布情况,选择合适的 ODC 概率阈值 P_T 可实现在一定面积约束下功耗降低最大化.本文分别选择 P_T 为 10%和 30%,揭示不同 P_T 对于功耗和面积的影响.另设置 LC_T 为 8,对 ITC99 电路进行 ODC 功耗优化.在 TSMC0.13um 工艺下对优化后的电路进行综合和功耗仿真得到各个结果如表2所示.传统 ODC 算法的数据来自文献[6].当 P_T 为 10%时,电路功耗比采用传统 ODC 优化算法平均下降 13.71%,面积平均减少 9.37%;当 P_T 为 30%,电路功耗比传统 ODC 优化算法平均下降 12.35%,但面积却平均减少了 13.44%.随着 P_T 从 10%增加至 30%,虽然功耗略微回升了 0.65%,但面积开销却下降了 4.07%,单位门控逻辑所获得的功耗降低越明显.用户可根据面积约束,有效设置 P_T 约束,最终达到面积与功耗的平衡.

表2 基于 ODC 概率的优化算法在不同 PT 下的功耗面积优化结果

基准电路	传统 ODC 算法		P_T 为 10%的基于概率的 ODC 优化算法				P_T 为 30%的基于概率的 ODC 优化算法			
	面积 [um²]	功耗 [uW]	面积 [um²]	-Δ [%]	功耗 [uW]	-Δ [%]	面积 [um²]	-Δ [%]	功耗 [uW]	-Δ [%]
bch_03	1640	513.4	1489	9.20%	430.23	16.20%	1295	21.01%	451.23	12.11%
bch_04	4726	911.9	4120	12.83%	875.42	4.00%	4120	12.83%	875.42	4.00%
bch_05	4073	299.3	3942	3.21%	253.51	15.30%	3942	3.21%	253.51	15.30%
bch_07	3173	476.7	2950	7.03%	426.65	10.50%	2745	13.48%	432.08	9.36%
bch_08	1450	241.8	1240	14.50%	220.28	8.90%	1170	19.32%	228.98	5.30%
bch_09	1561	450.8	1461	6.38%	377.77	16.20%	1376	11.85%	382.86	15.07%
bch_10	8855	1242	7756	12.41%	927.90	25.29%	7756	12.41%	927.90	25.29%

5 结论展望

本文通过总线 ODC 计算模型,有效减少了电路中 ODC 计算路径总量;通过短路径 ODC 计算模型,共享了相交短路径的 ODC 计算结果.通过基于概率的 ODC 优化策略,着重优化 ODC 概率高的通路,使门控时钟硬件资源分配向空闲状态概率高的通路倾斜,提高了门控硬件资源的使用效率.

目前 RTL 结构路径提取通过现有工具 Vperl 获得,Vperl 对于 RTL 代码风格有一定约束,对高层次行为描述代码的理解能力暂时比较有限,因此在 Vperl 基础上进一步提升算法对抽象 RTL 代码结构信息的识别能力是将来算法改进的重要方向.

参考文献:

[1] 李曦,王志刚,周学海,王煦法.面向低功耗优化设计的系统级功耗模型研究[J].电子学报,2004,32(2):205 – 208.
LI Xi, WANG Zhi-gang, ZHOU Xue-hai, WANG Xu-fa. Study on system-level power model for low power optimization[J]. Acta Electronica Sinica, 2004, 32(2): 205 – 208. (in Chinese)

[2] Scott J, Lee L H, Arends J. Designing the low-power M * CORE architecture[A]. Proceedings of the 25th Annual International Symposium on Computer Architecture[C]. Barcelona: IEEE Press, 1998. 145 – 150.

[3] Oliveira A L, Monteiro J C. Finite state machine decomposition for low power[A]. Proceedings of the 35th Conference on Design Automation Conference[C]. San Francisco, California, 1998. 758 – 763.

[4] Benini L, Micheli G D, Macii E, et al. Symbolic synthesis of clock-gating logic for power optimization of synchronous controllers[J]. ACM Transactions on Design Automation of Electronic Systems, 1999, 4(4): 351 – 375.

[5] Babighian P, Benini L, Macii E. A scalable algorithm for RTL insertion of gated clocks based on ODCs computation[J]. IEEE Transactions on Computer-Aided Design, 2005, 24(1): 29 – 42.

[6] Babighian P, Kamhi G, Vardi, M. PowerQuest: Trace driven data mining for power optimization[A]. Proceedings of Design, Automation & Test in Europe Conference & Exhibition[C]. IEEE Press, 2007. 1 – 6.

[7] Yan Xiaolang, Yu Longli, Wang Jiebing. A front-end automation tool supporting design, verification and reuse of SOC[J]. Journal of Zhejiang University Science A, 2004, 5(9): 1102 – 1105.

[8] Aho A V, Hopcroft J E, Ullman J D. 数据结构与算法(影印版)[M].北京:清华大学出版社,2003.

[9] Davidson S. ITC99 Benchmark[EB/OL]. http:// www.cerc.utexas.edu/itc99-benchmarks/bench.html.

作者简介:

孟建熠　男,1982 年 1 月出生于浙江省上虞市,博士.主要研究方向为高性能低功耗嵌入式 CPU 设计研究和超大规模集成电路设计技术.
E-mail: mengjy@vlsi.zju.edu.cn

丁永林　男,1981 年 12 月出生于浙江杭州.2006 年毕业于浙江大学电路与系统专业,获得硕士学位.主要从事嵌入式处理器的设计制造有关研究.
E-mail: yonglin_ding@c-sky.com