高技术通讯 2009 年 第 19 卷 第 8 期:850 ~ 854

doi:10.3772/j.issn.1002-0470.2009.08.014

一种基于元操作的宏模块功耗建模方法①

刘晓飞② 张 戈 韩承德

(中国科学院计算技术研究所微处理器技术研究中心 北京 100080)
(中国科学院研究生院 北京 100049)
(苏州中科集成电路设计中心 苏州 215021)

摘 要 为了有效支持系统芯片(SOC)的功耗分析和第三方 IP 的功耗评估,并有效保护知识产权,提出了元操作的概念,给出了一种基于元操作的宏模块功耗建模方法,建立了元操作功耗模型。该模型是一种周期精确的功耗模型,它描述了宏模块每个时钟周期的动态功耗变化情况。文中论述了这一模型的开发方法和使用方法,并指出,建立元操作功耗模型的关键是模块的功能定义、模块的功能到元操作的映射以及门级功耗样本的产生和收集。通过与门级功耗分析的实验数据对比,此元操作功耗模型的功耗分析误差在 4% 以内,功耗分析效率可以提高百倍以上。

关键词 系统芯片(SOC),功耗分析,低功耗设计,设计重用,宏模块,知识产权

0 引 言

随着设计重用技术的广泛采用,系统芯片(system-on-chip, SOC)中使用的第三方 IP 越来越多。这些模块多以硬核的形式出现,一般不提供电路网表和版图等具体信息,这不但无法对硬核 IP 进行功耗评价,而且难以进行 SOC 系统级的精确功耗分析和软件功耗优化。为此,需要建立一种满足下列基本要求的宏模块级的功耗模型:(1)屏蔽模块内部的物理实现信息,有效保护知识产权;(2)能够提供与模块工作状态密切相关的瞬态功耗;(3)功耗描述格式与现有的功耗分析工具和设计流程兼容;(4)相对于门级功耗模型的精度误差小于 10%。

近年来,研究人员在功耗分析领域做了大量的研究工作。指令级功耗模型[1,2]是一种统计模型,主要针对具有复杂功能的处理器模块。由于开发方法和描述方法的限制,即使考虑了指令之间的关联和操作数的影响[3,4],以及流水线的执行过程[5],误差依然超过 10%,并且无法反映瞬时功耗。Givargis 等人建立的宏模块指令级功耗模型[6],最大误差也超过 10%。与指令级模型不同,电容能量模型把模块的功耗描述为电源电压、负载电容和工作频率的函数。从 Wattch 功耗模拟器[7]和 CELL 处理器[8]的电容能量模型可以看出,其不足在于负载电容难以精确估计,模型误差往往大于 10%,而且针对不同的结构需要重复建模。上述两种模型多用于结构设计阶段的功耗评估和软件功耗评价。在物理设计阶段,普遍使用的是标准单元的门级特征化功耗模型。门级功耗模型有专门的描述格式,不同厂家的功耗分析工具可以实现模型数据共享。建模的过程需要大量的晶体管级 SPICE 仿真,一般借助于专门用于电路单元特征提取的电子设计自动化(electronic design automation,EDA)工具来完成。门级功耗分析工具利用门级功耗模型计算出精确的功耗。门级特征化模型的缺点是建模需要完备的门级网表和版图参数,无法描述复杂的宏模块功耗,并且建模效率很低。由于上述方法都无法建立满足 SOC 功耗分析要求的宏模块功耗模型,我们提出了元操作的概念,并建立了一种基于元操作的宏模块功耗建模方法。

本文讨论的功耗建模方法只针对模块的动态功耗。

1 元操作功耗建模方法

1.1 基本概念

为了便于描述建模方法,首先给出与元操作功耗模型相关的几个定义。

① 973 计划(2005CB321600),863 计划(2006AA010201,2007AA01Z112,2007AA01Z114)和国家自然科学基金(60673146,60703017)资助项目。
② 男,1978 年生,博士生,助教;研究方向:SOC 体系结构,集成电路物理设计和低功耗设计技术;联系人,E-mail: liuxf@szicc.com.cn
(收稿日期:2008-10-13)

定义1　宏模块,也称为IP核,是指能够实现某种特定功能,并且可以在专用集成电路(application-specific integrated circuit,ASIC)或者SOC设计中重复使用的电路功能模块。这些模块带有独立的输入输出端口和可配置的内部寄存器,通常以软核、硬核和固核的形式出现。

定义2　宏模块的功能是指宏模块在一个或者多个时钟周期执行的读、写、运算、数据变换等逻辑操作的集合。功耗模型的开发者负责完成模块的功能定义。功能定义要求具有完备性和互斥性。所谓完备性,是指包含逻辑设计阶段实现的全部功能。所谓互斥性,是指任意两种功能所对应的逻辑操作均不完全重叠。

定义3　元操作是指宏模块在一个时钟周期内所进行的逻辑操作。宏模块的某种功能可以用一个或者多个元操作序列来描述。

定义4　宏模块的功耗是指宏模块在某段时间之内为完成某种功能而消耗的能量。本文中建立的功耗模型是一种周期精确的功耗模型,它描述宏模块在一个时钟周期内的逻辑操作所消耗的能量,即元操作的功耗,因此这种功耗模型也称为元操作功耗模型。

定义5　元操作的功耗具有以下性质:

(1)元操作的功耗以时钟周期为单位表示。

(2)元操作的功耗与上一时钟周期的元操作类型有关。

(3)宏模块的可配置内部状态寄存器值的变化会影响元操作的功耗。

(4)宏模块在某一特定应用下的功耗可以用元操作的功耗来表示。

1.2　建模过程

建立元操作功耗模型的关键是模块的功能定义、模块的功能到元操作的映射以及门级功耗样本的产生和收集。实现元操作功耗建模的步骤如下:

(1)完成宏模块的物理设计,取得与版图相一致的门级网表和寄生参数。根据模块的逻辑设计规范,定义宏模块的功能。

(2)编写功能测试向量,使其包括对该模块全部功能的测试,并且每种功能要有足够多的重复测试次数,以便收集丰富的功耗数据样本。测试程序应该包含各种功能的顺序组合。

(3)建立功能并行表,实现功能向元操作的转换。定义元操作,并对其进行编码。元操作的定义及其编码是编写功能监测模块和建立功耗模型的基础。

(4)设计功能监测模块。该模块与测试程序一起运行,它根据端口信号和用户可见的内部寄存器值来确定宏模块的功能。功能监测模块可以用任何一种硬件描述语言、仿真器支持的验证语言或者C语言来实现。

(5)在模块门级仿真的基础上,取得包含门级电路状态变化的值变转储文件和周期精确的功能监测日志文件。根据功能并行表得出周期精确的元操作序列文件。进行周期精确的功耗分析,收集每个周期的功耗数据。

(6)对周期精确的元操作序列文件和功耗数据进行统计分析,按照特定格式建立基于元操作的宏模块功耗模型。

2　关键建模技术

2.1　功能监测与元操作生成

元操作功耗模型是一种周期精确的模型。不论建模的过程,还是使用模型的过程,都需要收集模块每个周期的运行状态,即该模块正在执行什么功能。这是通过功能监测模块来实现的。功能监测模块与结构模拟器或行为级仿真模型相配合,报告每个周期模块的运行状态。在功能监测的过程中,监测模块需要对模块的输入输出端口以及用户可见的可配置寄存器进行跟踪,然后根据模块的设计规范将其每个周期的状态记录到监测日志中。为了有效保护知识产权,功能监测不能使用模块的内部节点信号和对用户不可见的寄存器。同时,监测模块可以以加密代码或编译后的二进制代码的形式提供给用户。如果通过输入输出端口和用户可见的可配置寄存器无法准确监测功能,则需要重新定义模块的功能,或者将模块划分为更小的模块来逐一建模。根据不同的设计需求,功能监测可以运行在不同的抽象层次。建模时的功能监测需要在寄存器传输级(register transfer level,RTL)或者门级进行。

功能定义转化成元操作定义的过程是通过功能并行表来实现的。功能并行表是一张二维表,描述了模块可以在同一个时钟周期并行执行的功能组合。表格的列表示模块的功能,表格的行数是全部并行执行的功能组合的个数。表格的每一行代表一个元操作。建立功能并行表的方法是在表格的第一行写出模块的功能定义名称,然后在表格的每一行把在同一周期可以并行的功能标记出来,在每一行

的右侧为元操作定义一个名字。表 1 是异步串行通信接口(universal asynchronous receiver/transmitter, UART)模块的一张功能并行表(△标记表示对应的功能被执行)。在这个表格中将 UART 模块划分为 5 个功能,总共定义了 13 个元操作,并在表格中用 13 行分别描述。就表 1 中的元操作 BRSO 所在的一行来说,它表示如果某个时钟周期有且只有“APB 读”和“SOUT 输出”两个操作发生,那么该周期的元操作定义为 BRSO。一般来说,功能划分越细,模型精度越高。但功能划分越细,元操作定义就会越多,功耗模型中的数据量就会越大。

表 1 UART 模块功能并行表

APB 读	APB 写	SIN 输入	SOUT 输出	复位	元操作名称
					IDLE
				△	RESET
△					APBR
	△				APBW
		△			SIN
			△		SOUT
△		△			BRSI
△			△		BRSO
	△	△			BWSI
	△		△		BWSO
△		△	△		BRSIO
	△	△	△		BWSIO
		△	△		SIOG

2.2 精确建模技术

元操作功耗模型可以有两种形式。一种是一维模型,即只描述每个元操作的功耗;一种是二维模型,即描述由一种元操作向另一种元操作转化的功耗。显然,二维模型会更精确,建模的难度也比较大。

建立元操作功耗模型需要收集大量的功耗样本。这些样本来源于不同功能测试激励所产生的周期精确的门级功耗数据。门级功耗分析可以借助于成熟的 EDA 工具来完成。在建模的过程中,需要精心编写激励向量,以便得到所有元操作和元操作转化的样本。模型中的功耗数值是功耗样本中同类数据的统计平均值。为了提高模型精度,我们在实际工作中要求每种模型的采样数据均在 1000 笔以上。同时,为了尽量减小操作数对模型精度的影响,我们采用了随机向量生成技术,来避免地址信号和数据信号有规律地变化。

2.3 模型的使用

宏模块的开发人员向用户提供行为级仿真代码、元操作序列监测模块和元操作功耗模型,用于模块的功耗分析。用户在进行功耗分析时,首先利用 RTL 级、行为级的仿真模型或者借助结构模拟器来进行功能监测。在功能监测的基础上,通过功能并行表,分析出电路在每个周期所发生的元操作和元操作的转化情况。然后根据元操作功耗模型,计算出每个周期电路发生的功耗。最后,把各个周期发生的功耗累加,就得到电路在运行期间的总功耗。我们研制了专门的软件工具来使用元操作功耗模型进行功耗分析。这一工具可以高效完成 SOC 芯片硬核部分的功耗分析,并可用于快速评估单一硬核 IP 的功耗。同时,元操作功耗模型也可以描述成与现代集成电路标准单元工艺库文件兼容的格式,配合特定的行为级仿真代码,方便 EDA 软件的使用,从而可以集成到现有的集成电路设计流程中。

3 实验及结果分析

下面是一个针对 UART 模块建立元操作功耗模型的例子。这个模块遵循 AMBA 总线规范,在功能上与工业标准 16650 兼容,作为 SOC 的外围电路来实现串行通信。它通过 APB 总线接口与其它片上 IP 互连,设计的总线时钟频率为 166MHz,最大串行通信速率为 1.5M。IP 共有 13 个输入端口、12 个输出端口,14 个片上寄存器,数据总线的宽度为 32 位,地址总线宽度为 8 位,片上寄存器均为 32 位。片上 FIFO 的宽度为 32 位,深度为 64,串行输入输出的波特率可以通过片上寄存器的配置来调整。模块采用 Verilog 编码,使用 180nm 工艺完成物理设计。

我们利用 Verilog 编写了功能监测模块,该模块直接通过顶层测试模块挂接在 UART 模块上,根据输入输出信号和内部寄存器值的变化来报告周期精确的元操作序列。门级功能仿真采用 Cadence NC-Verilog 完成。门级功耗分析使用的是 Synopsys PrimeTime-PX 工具。整个 IP 划分为 5 个功能,定义了 13 个元操作。为了提高模型精度,我们利用二维查找表描述元操作转化功耗。

为了使用元操作功耗模型计算功耗,我们编写了专用的功耗分析软件 Dpower。软件的输入是元操作序列文件和元操作功耗模型,输出是测试程序运行期间的总功耗和平均功耗。我们把利用该模型计算出来的功耗与采用门级功耗分析方法得出的计算

结果进行对比。通过对包括各种不同功能序列的多个测试程序进行比较,实验结果显示(见表 2),与门级功耗分析相比,元操作功耗模型的功耗分析误差在 4%以内。同时,由于这种分析方法不再需要缓慢的门级模拟过程,不必生成庞大的翻转状态记录文件,避免了门级功耗分析过程,所以分析效率有了显著提高。实验数据中,程序 3 的误差之所以较大,是因为我们在测试程序中故意设计了很多违反逻辑操作规范的误操作,不断导致模块发生功能异常。由于功耗模型对于异常元操作序列的采样偏低,导致误差较大。上述针对 UART 模块开发元操作功耗模型的方法,亦可以用于其它 SOC 外围 IP 模块的功耗建模。我们用十余个模块对元操作功耗模型的建模方法进行了验证,都收到了很好的效果。

表 2　UART 元操作功耗模型与门级模型的实验数据对比

测试程序	仿真时间(ns)	总耗能			运行时间		
		门级(pJ)	元操作(pJ)	误差(%)	门级(s)	元操作(s)	加速比
程序 1	6×1882	25.09	25.35	1.04	31.40	2.19	14.34
程序 2	6×3032	43.74	43.91	0.39	47.50	2.41	19.71
程序 3	6×4921	14.76	15.24	3.25	58.99	2.34	25.21
程序 4	6×32574	518.43	518.02	0.07	315.74	3.35	94.25
程序 5	6×40133	639.99	639.11	0.14	775.30	5.14	150.84
程序 6	6×209251	3358.07	3361.28	0.10	2128.24	18.29	116.36
程序 7	6×288739	4636.02	4640.68	0.10	2671.6	25.06	106.61

4　结　论

本文提出了元操作的概念,并介绍了较为完善的元操作功耗建模方法。使用元操作功耗模型,可以对模块的平均功耗和峰值功耗进行分析。实验数据显示,利用元操作功耗模型针对 SOC 外围 IP 进行建模,在精确度和执行效率上都取得了很好的效果。

本文中建立的元操作功耗模型除了能解决硬核 IP 的功耗分析问题外,也可供软件开发人员来评估软件的功耗。软件代码经过编译器编译之后,由功能模拟器模拟执行,通过功能监测产生元操作序列表,然后利用元操作功耗模型和专用的功耗分析软件来计算平均功耗和峰值功耗。从这个角度来说,元操作功耗模型也可以用于芯片的体系结构设计阶段来估计同类 IP 在不同架构下的功耗。我们期望元操作功耗模型能够成为 SOC 系统低功耗设计的重要基础设施。

利用本文的方法针对片上处理器建立元操作功耗模型的过程要复杂一些。由于处理器的每一条指令可以定义为一个功能,所以功能并行表会比较复杂,元操作的数量众多,使得模型描述变得比较困难。我们尝试采用以下两种方法来解决这一问题:(1)合并功耗比较接近的功能,减少功能定义的数量,从而压缩元操作变换表的规模;(2)将处理器模块进一步划分为更小的模块进行功耗建模。这会使建模的过程变得复杂。在下一步的工作中,我们将研究如何简化处理器元操作功耗模型的建立过程。

参考文献

[1] Tiwari V, Malik S, Wolfe A. Power analysis of embedded software: A first step towards software power minimization. *IEEE Transactions on Very Large Scale Integration Systems*, 1994, 2(4):437-445

[2] Tiwari V, Lee M T. Power analysis of a 32-bit embedded microcontroller. In: Proceedings of the Design Automation Conference, Chiba, Japan, 1995. 141-148

[3] Sarta D, Trifone D, Ascia G. A data dependent approach to instruction level power estimation. In: Proceedings of the IEEE Alessandro Volta Memorial Workshop on Low-Power Design, Como, Italy, 1999. 182-190

[4] Sama A, Balakrishnan M, Theeuwen J F M. Speeding up power estimation of embedded software. In: Proceedings of the 2000 International Symposium on Low Power Electronics and Design, Rapallo, Italy, 2000. 191-196

[5] Kang K, Kim J, Shim H, et al. Software power estimation using IPI (inter-prefetch interval) power model for advanced off-the-shelf processor. In: Proceedings of the 17th ACM Great Lakes symposium on VLSI, Stresa-Lago Maggiore, Italy, 2007. 594-599

[6] Givargis T, Vahid F, Henkel J. Instruction-based system-level power evaluation of system-on-a-chip peripheral cores. *IEEE Transactions on Very Large Scale Integration Systems*, 2002, 10(6): 856-863

[7] Brooks D, Tiwari V, Martonosi M. Wattch: A Framework for Architectural-Level Power Analysis and Optimizations. In: Proceedings of the 27th International Symposium on Computer Architecture, Vancouver, Canada, 2000. 83-94

[8] Chaudhry R, Stasiak D, Posluszny S, et al. A cycle accurate power estimation tool. In: Proceedings of the Asia and South Pacific Conference on Design Automation, Yokohama, Japan, 2006. 867-870

A meta-operation-based method of power modeling for macro modules

Liu Xiaofei, Zhang Ge, Han Chengde

(Research Center of Microprocessor Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080)

(Graduate University of Chinese Academy of Sciences, Beijing 100049)

(Suzhou CAS IC Design Center, Suzhou 215021)

Abstract

To support the power estimation of system-on-chip at the chip level, facilitate the evaluation of the power dissipation of the third party IP modules, and protect the intellectual property of re-use blocks, the paper proposes an efficient power modeling method for macro modules based on the meta-operation definition and gives a meta-operation-based power model. This is a cycle-accurate model. It describes the dynamic power variation of each clock cycle of macro modules. The paper gives the methods to develop and use it, and points out that the key techniques in construction of the model are the definition of the module logical function, the mapping from the module function to meta-operation sequences, and how to collect samples of the gate level power value per cycle of the module. The experimental results show that the estimation errors of the new method are less than 4%, and the efficiency of the power estimation is more than one hundred times of the gate level method.

Key words: system-on-chip (SOC), power estimation, low power design, design reuse methodology, macro module, intellectual property