第 33 卷 第 2 期 计 算 机 工 程 2007 年 1 月
Vol.33 No.2 Computer Engineering January 2007
·博士论文· 文章编号：1000—3428(2007)02—0009—03 文献标识码：A 中图分类号：TP391

基于 CDFG 的 SoC 验证方法及其分割与搜索算法

李德识，曹 阳
（武汉大学电子信息学院，武汉 430079）

摘 要：随着芯片复杂度以及市场对集成电路上市时间要求的不断提高，对 SoC 设计方法和验证方法带来了巨大的挑战。控制数据流图可用于系统建模、软硬件功能划分、系统综合与验证等多个环节。该文针对 SoC 验证的需要，利用 CDFG，研究了基于 CDFG 的验证体系，给出了 CDFG 的几种定义，讨论了 CDFG 的表示方法，提出了基于 CDFG 的验证流程，研究了基于 DFS 的生成树算法、CDFG 的分割算法和 CDFG 的搜索算法，并以实例说明了这些算法在验证流程中的作用。

关键词：控制数据流图；SoC；验证；深度优先搜索算法

CDFG Based SoC Verification Method and CDFG Division and Searching Algorithm

LI Deshi, CAO Yang
(College of Electronic Information, Wuhan University, Wuhan 430079)

【Abstract】 With the increasing of ASIC complexity and the demand for time to market, great challenges for SoC(system on chip)design and verification have to be faced. CDFG(control data flow graph)can be used in system modeling, software and hardware partition, synthesis and verification. Aiming at the final goal of SoC verification, the paper discusses the system verification based on CDFG by raising the level of abstraction, a new verification flow is presented. By analyzing CDFG's definition and its representation, the tree creating algorithm based on depth first search principle, CDFG division algorithm and CDFG searching algorithm are given. Then some examples of these algorithms are illustrated, which specify the operation of these algorithms in SoC verification flow.

【Key words】 Control data flow graph(CDFG); System on chip (SoC); Verification; Depth first search algorithm (DFS)

在 SoC 设计过程中，控制数据流图(Control Data Flow Graph, CDFG)有着广泛的用途，可用于：系统特性描述，系统软硬件功能划分，设计优化与系统综合，子图匹配及等效性检查，仿真调度优化和测试与验证。在系统描述建模后，进行功能验证，以验证系统描述以及功能模型的正确性，减少系统级设计错误，是非常必要的。利用 CDFG 实现功能验证是一种满足上述要求的重要手段。

CDFG 不仅能够描述系统硬件逻辑功能，同时能够描述软件流程，适宜于与硬件描述语言 HDL，或软件语言 C/C++一起，进行转换、调度与验证。System C 在 C/C++的基础上，增加了硬件端口、通道及时序等描述方法，能够对硬件系统进行建模。利用事务级抽象，能实现事务级设计与验证，当设计层次到达 RTL(Register Transfer Level)级后，在 CDFG 上增加时序信息，亦能实现 RTL 级的功能验证。

基于 CDFG 的验证流程如下：

(1)需要将用高级语言(如C/C++语言)描述的系统说明转换成为CDFG图；

(2)再进行功能验证。功能验证属于白盒或灰盒验证的方法。

CDFG 不仅能够通过程序代码转换，通过进一步的抽象，它也能描述事务级功能，其内部可以包括 CDFG 子图，对于 CDFG 子图或局部功能模块的验证，是系统验证的基础。

1 当前研究状况

在 SOC 及 ASIC 设计、仿真与验证方面，利用 CDFG 作为一种手段，许多大学和机构在该方面开展了研究，并取得了相应的研究成果。相关的研究情况如下：

部分研究侧重于CDFG的子集生成，如Darko等研究了基于子集剪切的SOC程序调试方法，讨论了完备子图的定义和子图生成算法[1]；Miodrag Potkonjak等研究了利用CDFG进行行为表示。

一些研究面向利用CDFG进行功能划分和系统综合，如Naotaka Ohsawa等研究了利用CDFG直接定向的方式[2]；Euiseok等研究了在异步综合中，利用CDFG进行分布式电路自动生成的方法；Gero Dittmann研究了数据流图(DFG)库的组织问题[3]。

利用 CDFG 进行特性表达研究，Reinaldo 等提出了一种高层次设计的内部表示方法；Peter Oehler 等研究了基于混合数据流图(HDFG)的模拟/数字混合综合方法；Sandhya Seshadri 等采用变量的静态单一指派方法；Zhu Ming 等研究了基于 CDFG 的特性分类和定义。

利用 CDFG 进行设计优化的研究，如 Michel 研究了利用 CDFG 将应用映射到 FPGA 的方法；Peter Voigt Knudsen 等研究了利用 CDFG 进行软硬件划分的方法，用于处理器通信性能的估计；Elie Torbey 等利用遗传算法，研究了多时钟选择与综合问题。

本文针对 SOC 高层次抽象与验证，首先讨论了 CDFG 的

基金项目：国家“863”计划基金资助项目(2002AA1Z1490)
作者简介：李德识(1968－)，男，副教授、博士后，主研方向：SoC 建模与设计理论，传感器网络，嵌入式系统；曹 阳，教授、博导
收稿日期：2006-03-27 **E-mail：**lds@eis.whu.edu.cn

定义及其表示，随后重点讨论 CDFG 的分割算法和搜索算法，最后提出了利用 CDFG 进行系统功能验证的方法。

2 CDFG 及其表示

CDFG 可以看成是由控制流图和数据流图组合而成的，可以粗略地分为两个部分——控制部分和数据部分。对于控制部分，主要反映事件及交互关系、响应时间、事件或处理的优先级。对于数据部分，可以看成运算或处理实体，其功能依赖于输入和输出，对每个事件或处理都是平等的，其特性是存储或处理的效率(需要的时间)。

2.1 CDFG 的定义

这里给出了数据流图、控制数据流图的定义如下：

定义 1 一个数据流图 DFG 是一个二元组 G(V，E)构成的图，其中 V 是图的节点构成的集合，E 是节点之间的边构成的集合。

定义 2 一个数据流图 DFG 是一个三元组 Ω(N，E，O)，其中 N 是功能处理节点构成的集合，E 是节点之间的边构成的集合，O 是每个节点处理功能的集合。

定义 3 一个控制数据流图 CDFG 是一个四元组 Γ(S，C，Z，E)，其中 S 是控制分支节点构成的集合，C 是状态节点构成的集合，Z 是非控制分支节点及非状态节点构成的集合，E 是节点之间的边构成的集合。

定义 4 一个控制数据流图 CDFG 是一个二元组 Σ(U，E)，其中 U=Ω∪Γ 为单元的集合，单元可以是 DFG 单元类型、CDFG 单元类型或二者的组合类型，E 是单元之间的边构成的集合。

定义 4 给出了用于高层抽象、能够实现系统级描述的分层控制数据流图的表达方式。

2.2 CDFG 的表示

CDFG可以进行高层次的表示，一个CDFG节点不仅能够表示简单的操作，而且能够进行函数调用[4]，内部可以包括多个CDFG子节点，称为分层控制数据流图。根据不同的需要，可以采用其中的子集作为CDFG的表示。

根据 CDFG 节点的特征，可以将节点划分为不同类型，包括 DFG 节点、循环体节点、状态节点、分支节点、子 CDFG 节点、功能(函数)调用节点、输入/输出节点等。

对每个 CDFG 节点，分别赋予相应的特征参数，这些参数包括节点类型、节点读/写集合、所属 CDFG 图或子图、功能(函数)调用变量、测试变量、输入/输出映射变量、节点执行时间特征等。其中，节点执行时间特征用节点执行时钟周期数表示，可以适应不同系统时钟的要求。对每个 CDFG 边，也分别赋予相应的特征参数，这些参数包括边的类型、源及目标节点、边上传输的变量集合、边上信息传输所需要的时钟周期数等。

3 CDFG 的分割

对于一个复杂的设计，生成的 CDFG 通过共享合并处理后，能够提高抽象的程度，优化设计，简化了 CDFG 结构，但对于功能验证仍然存在巨大的困难。为此，针对局部功能验证的需要，为达到提高验证覆盖率的目的，研究了 CDFG 分割算法。

3.1 基于 DFS 的生成树

CDFG 是一个有向无环图，表示为 G(V，E)，其中 V 表示图的顶点，E 为图的边。能否将 CDFG 转换为树结构？该问题可以转换为图的分割，可以利用深度优先搜索(Depth First Search，DFS)算法，进行有向图搜索，剪掉部分前向边、后向边和横跨边，以生成多个树，以及由树构成的森林(Forest)。

生成树算法的伪码表示如下：

{输入 CDFG 构成的有向无环图；
执行 DFS 算法，记录节点搜索号(DFN)和回溯节点的顺序号；
根据节点搜索号(DFN)，建立搜索顺序表；
生成树及森林；
记录各树的前向边集合、后向边集合、横跨边集合；
记录树间连接边集合；
}

一个图到生成树的实例见图 1，其中图 1(a)为CDFG图，图 1(b)为生成的两棵树和构成的森林，其中的号码 1~12 为DFS搜索序号DFN。该算法复杂性为Θ(|V|, |E|)[5]。

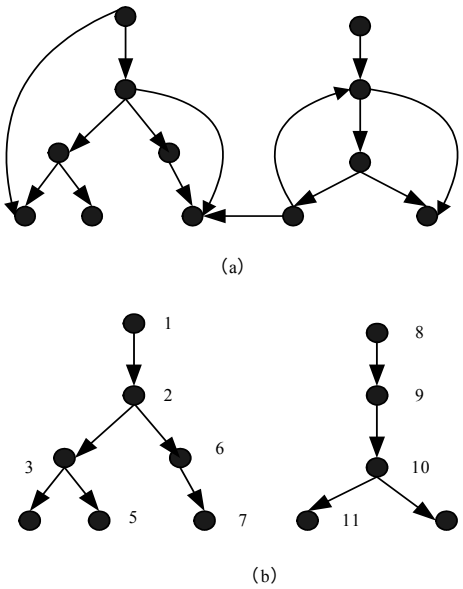

图1 DFS搜索及生成树

3.2 树的分割

利用 CDFG 生成的树和森林以及节点搜索顺序表，将一棵树分割成为不同的部分，以进行功能验证。该问题可以化为树枝的裁剪问题。

采取的方法是：先找到自树根节点开始的最大左边树，根据 DFS 过程中记录的回溯节点 DFN，查寻是否存在右边树枝，若存在，则该右边树枝作为裁剪枝。上面所谓的左边或右边树枝是二叉树的概念，对于多分支的节点，只把最早选择的分支看成左边树枝，其他树枝均视为右边树枝。

树分割算法的伪码表示如下：

{输入生成树算法的树、搜索表和回溯节点的顺序号；
对森林中的任一棵树，若节点在该树内，则
搜索所有回溯节点，若其父节点有右边树枝，则
该右边树枝作为裁剪枝；
记录裁剪节点变量；
裁剪后树枝的剩余部分构成左边树枝；
构成各树的左边树枝和右边树枝的集合；
}

经过树分割算法的处理后，图 1 对应的树得到了分割，其左边和右边树枝见图 2。

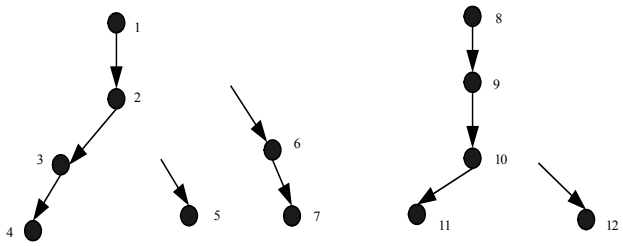

图2 树分割处理生成的树枝

由于需要处理的树不一定是均衡树，分割后生成的树枝长度可能各不相同，且长度差距比较大。在实际处理中，可通过串联节点的合并，减少串联节点的数量；同时经过串联节点合并处理后，树枝长度很长的树枝一般为左边树枝，根据设定的树枝长度(深度)，进行单树枝的裁剪，可以达到相对均衡裁剪树枝长度的目的，使得局部功能的验证易于进行。

由于使用了回溯节点的顺序号，树分割算法不需要搜索全部节点，只需搜索回溯节点的父节点即能完成分割，该算法复杂性为O(|V|)。

4 CDFG 搜索算法

在验证过程中，有时需要对特定变量或处理过程进行检查，或需要对局部程序加重验证。根据条件获取选择节点所属的部分，是范围树的搜索问题，或者转化为寻找这些节点构成的最小树。

线段树是一个二叉树 T(a，b)，它给出了两个整数参数 a 和 b，v 表示其根，S[v]= a 表示一个区间的起点，E[v]= b 表示一个区间的终点。对于二叉树，查询被搜索节点的并集，就可以获得最小树。

由于 CDFG 生成的树一般不是二叉树，因此不能直接使用该方法。我们利用 CDFG 生成树和树分割的结果，进行节点搜索，以建立最小生成树，并获取其根。

CDFG 搜索算法的伪码表示如下：

{输入被搜索节点的号码(DFN)和分割树生成的左边树枝和右边树枝的集合；

取被搜索节点的任意两个节点，若节点属于一个树枝，则删除被搜索节点号码(DFN)大的节点；

否则，若两个节点所属树枝的根是否在一个树枝上，则取右边树枝根节点及被搜索节点号码(DFN)小的节点；

删除两个节点；

若被搜索节点数大于 1，则继续搜索，否则该节点即为被搜索节点构成树的根节点；

获取最小树及树枝；

}

对于节点不属于同一个 CDFG 生成树的情况，该算法将不能得到有效结果。图 2 中，若查询节点 4、5 和 7 构成的最小树，得到根节点为 2，由节点 2~7 构成了最小树。

5 基于 CDFG 的验证流程

基于 CDFG 的验证流程见图 3，在图 3 中，首先将用高级语言描述的系统说明转换为控制数据流图，随后进行 CDFG 的优化与抽象，达到优化设计、提高抽象层次、便于等效性检查等目的，其核心是节点的共享与合并。对经过优化的 CDFG 进行系统级验证，核心是系统功能和性能是否达到设计要求，是否与设计等效。

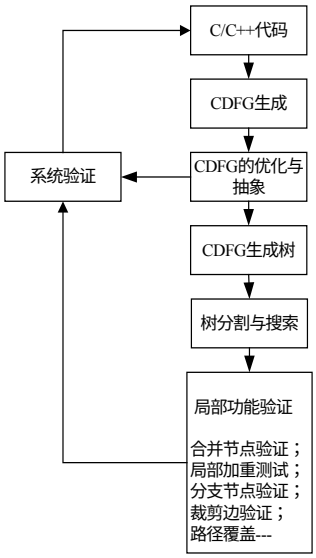

图 3 基于 CDFG 验证流程

将 CDFG 转换为树结构，经过树分割后，可进行多个单向处理流程的功能验证与仿真，将复杂问题简单化。同时，能够实现路径的覆盖。

对于具体的验证，除树枝的验证作为验证的重要内容外，在 CDFG 生成树阶段被裁剪掉的边需要单独验证。串联合并和同步合并节点的验证也是必须的。

对于特定区域的测试，通过 CDFG 搜索，构造最小生成树，对该树进行局部验证实现。完成了局部功能验证及路径覆盖后，可以进行自底向上的验证，即先进行模块(树枝、边)的功能验证，再进行系统功能验证，这样局部经过验证的功能可以实现设计重用。

6 总结与展望

本文针对 SOC 验证的需要，开展了 SOC 高层次验证相关方法的研究工作，系统地研究了基于 CDFG 的验证体系，提出了基于 CDFG 的验证流程，研究了基于 DFS 的生成树算法、CDFG 的分割算法和 CDFG 的搜索算法。

今后计划在基于 CDFG 验证方法的工具原型、SOC 性能验证等方面进一步开展相关研究与开发工作。

参考文献

1 Kirovski D, Potkonjak M, Guerra L M. Cut-based Functional Debugging for Programmable Systems-on-chip[J]. IEEE Transactions on Very Large Scale Integration Systems, 2000, 8(1).

2 Ohsawa N, Hariyama M, Kameyama M. High-performance Field Programmable VLSI Processor Based on a Direct Allocation of a Control/Data Flow Graph[C]. Proceedings of the IEEE Computer SoCiety Annual Symposium on VLSI, 2002-04.

3 Dittmann G. Organizing Libraries of DFG Patterns[C]. Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, 2004-03.

4 Knudsen P V, Madsen J. Graph Based Communication Analysis for Hardware/Software Codesign[C]. Proc. of CODES ’99, Rome, Italy, 1999-05.

5 Levitin A. Introduction to the Design & Analysis of Algorithms[M]. Pearson Education, 2003-10: 162-173.