异步电路的静态数据流图模型及其性能分析

晋 钢 王 蕾 王志英
(国防科技大学计算机学院 长沙 410073)

摘 要 静态数据流图是异步电路的一种抽象模型,具有灵活性高、易于理解的优点。基于静态数据流图的一种形式化的执行语义,提出了一种适合于性能分析的静态数据流图的Petri网模型,并基于该模型提出了一种性能评价方法。该方法具有速度快、灵活性高的优点,特别适合大规模异步电路设计早期的性能分析。该模型比静态数据流图的传统Petri网模型在规模上小一倍,而且避免了引入非标准的read-arc。通过实验,该模型和性能评价方法的有效性得到了充分的验证。

关键词 静态数据流图,执行语义,性能分析模型,性能评价函数

Performance Evaluation Method for Asynchronous Circuit Based on Static Data Flow Structure

JIN Gang WANG Lei WANG Zhi-ying
(School of Computer Science, National University of Defense Technology, Changsha 410073, China)

Abstract The Static Data Flow Structure (SDFS) is an abstract model of asynchronous circuits, which is very flexible and understandable. A Petri net model which is used for performance analysis was introduced. And a fast performance evaluation method was also introduced, which is very suitable for the very large scale asynchronous circuit design. The size of this model is just half of the traditional Petri-net model for the asynchronous circuits, and the read-arc which was introduced in the previous Petri-net model was eliminated in this model. At last, several experiments were presented to prove the validity of this model.

Keywords Static data flow structure, Execution semantics, Performance analysis model, Performance evaluation function

1 引言

随着集成电路技术的发展,同步集成电路设计方法遭遇到了很多诸如时钟扭曲等设计难题。为解决日益严重的时钟扭曲问题,设计人员引入了复杂的平衡时钟树,这也导致了芯片面积功耗越来越高的问题。现代超大规模同步集成电路中,时钟树要占到1/3的面积和将近一半的功耗。随着工艺尺寸的进一步缩小,工艺变化对于集成电路设计的影响日益严重,这导致更多的设计时间和芯片资源将被耗费在解决这些问题上。异步电路是一种没有全局时钟,而采用局部握手协议替代全局时钟的电路设计方法,可以天然地避免由于引入全局时钟而导致的各种设计问题。而且异步电路在电路没有数据处理的空闲时间不产生动态功耗,因而具有低功耗的优势,在现阶段功耗越来越成为电路设计者的重要设计考虑,因此异步电路有着强大的吸引力。目前异步电路设计方法得到了广泛的关注,并取得了很大的进展,如曼彻斯特大学的Amulate系列处理器[1]等。

但是由于传统的异步集成电路设计方法需要设计者将功能和时序同时进行考虑,而不能像同步电路设计那样设计者可以将功能和时序分别进行考虑。在功能设计阶段,设计者就需要将大量的设计时间和精力放在保证电路的时序正确上;而且异步电路设计缺乏有效的描述语言和模型支持,在设计过程中往往采用Petri网、STG(Signal Transition Graph)图以及进程代数(CSP)等晦涩的描述工具进行设计,这些都大大增加了异步电路的设计难度。为了降低异步电路的设计难度,静态数据流图模型被引入异步电路设计过程,该模型可以描述异步电路的拓扑结构和行为。就像RTL描述在同步电路设计过程中的作用一样,静态数据流图可以作为异步电路设计的基础,在异步电路设计过程中有着广泛的应用。

由于异步电路中没有同步电路中的全局时钟,很难对异步电路像对同步电路一样进行静态时序分析。在设计的早期阶段,对电路的性能进行分析十分重要,而针对静态数据流图,现在仅有一些基于模拟的性能分析方法,不利于在设计优化过程中快速得到电路的重要性能参数。为了解决这个问题,本文在静态数据流图的形式化定义的基础上,提出了一种适合于对其进行性能分析的性能分析模型。

本文第2节介绍了静态数据流图的形式化定义;第3节形式化地定义了一种静态数据流图的执行语义;第4、5节建立了性能分析模型并引入了相应的性能评价函数;第6节给出了该模型的一些试验结果;最后是结论。

到稿日期:2009-01-09 返修日期:2009-03-15 本文受国家"八六三"高技术研究发展计划基金项目(2007AA01Z101),国家自然科学基金(60773024)资助。

晋 钢 博士研究生,主要研究方向为计算机体系结构、异步微处理器和异步集成电路设计关键技术研究,E-mail:jingang.lucky@gmail.com。

2 静态数据流图模型

Steve Furber 在文献[2]中提出了异步电路的静态数据流图模型(SDFS),并给出了一个非形式化的定义。就像在同步电路设计过程中采用寄存器传输模型(RTL)作为电路的高级抽象模型一样,SDFS 是一种异步电路数据通路的高级抽象模型。文献[3]给出了 SDFS 的形式化模型,定义 SDFS 如下。

定义 1(静态数据流图,SDFS) 是一个有向图 $G=\langle V, E, D, M_0\rangle$,其中 V 为节点集合,E 为边集合,D 为一个值域,M_0 是该图令牌(token)的初始赋值。

在 SDFS 中,有两种不同的节点,一是表示组合逻辑的节点 l,二是表示寄存器及锁存器的节点 r,并且 $V=r\cup l$。只有寄存器对应的节点才能存储令牌。

定义 1 给出了静态数据流图的形式化模型,该模型借用了有向图,并使用令牌表示静态数据流图中数据的流动。令牌在图中不同节点间流动,表示数据在相应节点被存储或者处理。一个组合逻辑节点代表了某种逻辑运算,而一个存储节点则代表了数据在该节点被存储,并可以被后续处理过程使用。为了完整描述静态数据流图,还有几个相关定义需要说明。

定义 2(前集和后集) 在静态数据流图中,一个节点 v 的前集定义为 $\cdot v=\{u\in V|(u,v)\in E\}$,其后集定义为 $v\cdot=\{u\in V|(v,u)\in E\}$。

静态数据流图中除了存储节点外,其余节点都必须有前集和后集。没有前集的存储节点称为源节点,没有后集的存储节点称为漏节点,分别用来描述电路的输入节点和输出节点。

图 1 是一个静态数据流图的例子,静态数据流图形式化地定义了电路的拓扑结构,是一种分析异步电路的有效模型。

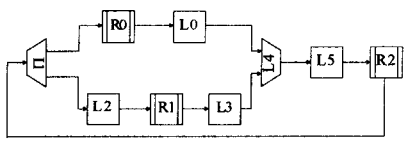

图 1 简单的静态数据流图

静态数据流图可以在不同层面定义电路的拓扑结构。在高级层面,每一个组合逻辑节点可以代表一个组合逻辑模块,每一个存储节点可以代表一组寄存器。这样静态数据流图可以被异步电路设计者用来设计电路的高级功能模型。同样,静态数据流图也可以表达电路的门级模型,每一个组合逻辑门都表示为一个组合逻辑节点,每一个寄存器或锁存器都被表达为一个存储节点,这样一个门级网表就可以表达为一个相应的静态数据流图,并可以通过该静态数据流图对原电路进行验证和分析。

3 执行语义模型

静态数据流图仅仅描述了电路的拓扑结构,对于数据如何在电路中传递,数据如何在寄存器中存取等行为都没有定义。文献[3]给出了静态数据流图 3 种不同的语义模型,分别对应了不同复杂度的电路实现方式,这里给出最基本的传播语义模型,该语义模型定义了普通异步流水线的行为模型。

定义 3(路径) 一个节点序列 $(z_0, z_1, z_2, \cdots, z_{n-1}, z_n)$,若满足任意 $(z_{i-1}, z_i)\in E$,则称该序列为这个静态数据流图的一个路径,记为 $\sigma(z_0, z_n)$。

定义 4(投射,project) 一条路径 σ 对一个节点集合 X 的投射定义为 $\sigma\downarrow X=Set(\sigma)\cap X$,其中 $Set(\sigma)$ 为路径 σ 中节点的集合。

定义 5(R-前集和 R-后集) 为了描述的方便,定义一个节点的 R-前集和 R-后集,一个节点 v 的 R-前集定义为 ${}^*v=\{r\in R|\exists\sigma(r,x):\sigma(r,x)\downarrow R=\{r,x\}\cap R\}$;类似地一个节点的 R-后集定义为 $v^*=\{r\in R|\exists\sigma(x,r):\sigma(x,r)\downarrow R=\{x,r\}\cap R\}$。

在定义了上述概念之后,就可以定义令牌在静态数据流图中传递的语义模型了。可以定义多种不同的语义模型,这里介绍一种最基本的传递语义模型[3]。该语义模型是最基本的静态数据流图的语义模型,事实上该模型定义了 Muller 流水线的行为[4]。该语义模型的形式化描述如下。

定义 6(标记) 静态数据流图 G 的一个标记定义为一个映射关系 $M:R\to\{0,1\}$,即存储节点集合到有限集合 $\{0,1\}$ 的一个映射,表示了每个存储节点是否还有令牌,并且每个存储节点存储令牌数量的上限为 1。

定义 7(组合逻辑节点的计算和复位) 组合逻辑节点的计算状态可以定义为一个映射,$\Sigma:L\to\{0,1\}$,其定义了每个组合逻辑节点 $l\in L$ 是否已经完成了计算工作并输出了结果。如果该组合逻辑节点完成了计算并输出了结果,则 $\Sigma(l)=1$;反之,如果该节点还没有完成计算,则称该节点处于复位状态,有 $\Sigma(l)=0$。

一个处于复位阶段的组合逻辑节点 $l\in L$,若有 $\forall k\in\cdot l\cap L, \Sigma(k)=1$,即该节点前集中所有组合逻辑节点都完成了计算,并且 $\forall k\in\cdot l\cap R, M(k)=1$,即该节点前集中所有存储节点都被标记,则该节点 l 可以开始进行计算。反之,若有 $\forall k\in\cdot l\cap L, \Sigma(k)=0$,即该节点前集中所有组合逻辑节点都处于复位状态,并且 $\forall k\in\cdot l\cap R, M(k)=0$,即该节点前集中的存储节点都未被标记,则该节点 l 可以进行复位。

定义 8(存储节点的使能和失能) 初始情况下,所有被标记的存储节点都处于使能状态,所有未被标记的节点都处于失能状态。当且仅当 $\forall l\in\cdot r\cap L, \Sigma(l)=1$,即该节点前集中所有组合逻辑节点都完成了计算,并且 $\forall l\in\cdot r\cap R, M(r)=1$,即该节点前集中所有的存储节点都被标记,则该存储节点 r 被使能;反之,当且仅当 $\forall l\in\cdot r\cap L, \Sigma(l)=0$,即该节点前集中所有组合逻辑节点都处于复位状态,并且 $\forall l\in\cdot r\cap R, M(r)=0$,即该节点前集中所有的存储节点都未被标记,则该存储节点 r 被失能。

定义 9(令牌的传递) 如果一个存储节点 r 处于使能状态,当且仅当 $\forall r'\in{}^*r, M(r')=1$ 并且 $\forall r'\in r^*, M(r')=0$,即 r 所有的 R-前集中的存储节点全部被标记,而 r 所有的 R-后集中的存储节点全部未被标记,则一个令牌可以被传递到该使能的存储节点 r,即 $M(r)=1$。反之,如果一个存储节点 r 处于失能状态,当且仅当 $\forall r'\in{}^*r, M(r')=0$ 并且 $\forall r'\in r^*, M(r')=1$,即 r 所有的 R-前集中的存储节点全部未被标记,而 r 所有的 R-后集中的存储节点全部被标记,则一个令牌可以被移出到该失能的存储节点 r,即 $M(r)=0$。

值得注意的是,对于源存储节点,由于既没有前集也没有 R-前集,所以源存储节点总是处于使能状态,并且一旦它的

R-后集中的存储节点都处于未标记的状态，则该节点可以产生一个令牌。同样的，对于漏存储节点，如果它处于被标记的状态，并且它的 R-前集中所有的存储节点都未被标记，则可以从该存储节点中将令牌移去。

定义 3 到定义 9 完整地定义了静态数据流图的一种执行语义，该执行语义是最简单的一种执行语义。为了改善电路的执行效率，在此基础上文献[3]还提出两种其他的较为复杂的执行语义模型，该模型可以改善模型的执行效率，可以描述执行效率更高的 Anti-token 流水线和 Counterflow 流水线。本文基于以上基本的执行语义模型，提出了一种基于 Petri 网的性能分析模型，以快速地计算出模型的最大延迟时间，从而有效地指导基于静态数据流图的异步电路设计过程。

4 性能分析模型

基于模型进行性能分析在异步电路设计过程中有重要的意义，由于在异步电路设计过程中没有时钟周期的概念，对于性能的建模就显得尤其重要。尤其是在设计的最初阶段，缺乏具体的电路实现，基于一定的模型估算最终电路的性能对于整个设计过程有着重要的意义。而静态数据流图的结构较为复杂，虽然可以精确地描述异步电路的执行，却不太适合于进行快速的性能分析。文献[4]在静态数据流图的基础上定义了一个与其等价的带 read-arc 的 Petri 网模型 CPN(Circuit Petri Net)，用于模型的验证工作。但由于该模型引入了 read-arc，其是一种非标准的 Petri 网模型，因此很难基于该模型进行性能评估。为解决这个问题，在前文定义的执行语义模型的基础上，本文提出了一种适于性能分析的时延 Petri 网模型。

定义 10(时延 Petri 网) 时延 Petri 网是一个五元组 $\Pi=\langle P,T,F,\Delta,M_0\rangle$。其中 $P=\{p^1,p^2,\cdots,p^m\}$是位置的有限非空集合；$T=\{t^1,t^2,\cdots,t^n\}$是变迁的有限非空集合。$F\subseteq(P\times T)\cup(T\times P)$称为流关系。$\Delta$ 是变迁的实施时间函数。$M_0:P\rightarrow\{0,1\}$是 Petri 网的初始标识函数。

定义 11(异步电路的平均周期) 即异步电路所对应的时延 Petri 网模型所有回路平均周期的最大值，描述了异步时延电路的平均性能。

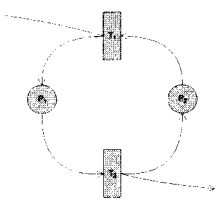

图 2 寄存器对应的 Petri 网子结构

为了计算静态数据流图的平均周期，首先需要将静态数据流图转化为适合进行性能分析的时延 Petri 网模型。整个静态数据流图由两种节点组成，即组合逻辑节点和存储节点。每个组合逻辑节点完成特定的逻辑计算功能，当输入端的数据都准备好之后该组合逻辑节点就可以开始计算，在经过一定的延迟之后产生输出结果。这种行为模式和 Petri 网中的变迁非常详细，可以用 Petri 网中的变迁来表示一定的组合逻辑运算；而对应的，电路中的寄存器存在两种不同的状态，即有数据锁存在寄存器中或者无数据锁存。依据这些性质，可以把寄存器建模成一个小的 Petri 网子结构，可以在性能上模拟寄存器的行为。图 2 是一个寄存器及其对应的 Petri 网子结构，该 Petri 网子结构由两个变迁节点和两个位置节点构成。两个位置节点 P_1 和 P_2 分别代表了寄存器中有无数据两种状态，而 T_1 变迁节点表示一个数据被锁存到该寄存器中，T_2 节点表示数据被输出，寄存器返回到无数据存储的状态。

由于静态数据流图中的每个组合逻辑节点都转化为 Petri 网中的一个变迁节点，而每个存储节点可以转化为由两个变迁节点和两个位置节点构成的子结构，该子结构中的位置节点只有一个输入和一个输出，在 Petri 网中没有选择出现，最后得到的是一个标记图(Mark Graph)。

依据这些想法，本文定义了静态数据流图的性能评估 Petri 网模型 PEPN(Performance Evaluation Petri Net)。

定义 12(静态数据流图的性能评估 Petri 网模型，PEPN) 静态数据流图的性能评估 Petri 网模型是一个时延 Petri 网 $TP=\{P,T,F,\Delta,M_0\}$，各个组成部分的含义和普通时延 Petri 网一致。

该 Petri 网由两种不同的结构互联而成，一种是如图 2 所示的代表存储节点的子结构，一种是代表组合逻辑节点的变迁，这两种结构之间通过一个位置节点根据静态数据流图的连接关系互联在一起。为了性能分析的需要，需要根据在静态数据流图中的节点的延迟值，对 PEPN 中的相应变迁节点的延迟值进行赋值。

由于静态数据流图可以对门级电路进行建模，对于门级电路，也可以得出同样的性能评估 Petri 网模型，从而对门级的异步电路描述进行建模并进行性能评估。图 3 是一个门级电路及其性能评估 Petri 网模型。

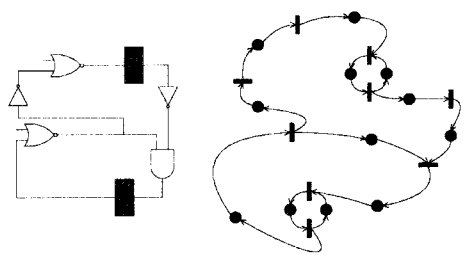

图 3 一个示例电路及其性能评价 Petri 网模型

5 性能评价函数

对于时延 Petri 网，设矩阵 $C=[c_{ij}]$是一个 $m\times n$ 阶整数矩阵，每一个元素 $c_{ij}=c_{ij}^{+}-c_{ij}^{-}$，其中 c_{ij}^{+}是变迁 i 和其输出位置 j 的有向弧的权值；c_{ij}^{-}是变迁 i 和其输入位置 j 的有向弧的权值。

定义 13(S-不变量) 时延 Petri 网的 S-不变量是一个 m 维向量 y，并且满足 $A^Ty=0$。直观地说，一个时延 Petri 网的 S-不变量对应的是 Petri 网中的一个位置集合，在任何位置触发序列中，该位置集合中的位置节点的令牌总数是个常量。文献[5]介绍了求取一个 Petri 网 S-不变量的算法。

定义 14(T-不变量) 时延 Petri 网的 T-不变量是一个 m 维向量 x，并且满足 $Ax=0$。直观地说，对于一个回到 Petri 网初始状态的变迁触发序列，一个时延 Petri 网的 T-不变量对应于每个变迁在该触发序列中被触发的次数。

定义 15(时延 Petri 网的最大平均周期的下限) 基于以上关于 S-不变量和 T-不变量的定义，文献[6]给出了时延 Pe-

tri 网最大平均周期的下限。

$$\tau_{\min}=\max_i\{\frac{y_i^T(C^-)^TDx}{y_i^TM_0}\}$$

其中，D 是变迁的实施时间(即延时值)θ_i 组成的对角矩阵。M_0 是一个 m 维向量，对应每一个位置中的令牌数目初始值。

该算法中，最大值要取遍 S-不变量的最小支持集。文献[6]指出，一个有 n 个位置节点的 Petri 网的 S-不变量的最小支持集中元素的数量为($n/\lceil n/2\rceil$)，其中$\lceil * \rceil$对一个实数进行上限取整。可以看出，在最坏的情况下，S-不变量的最小支持集以指数规模随着 Petri 网规模的增大而增大。

由于本文提出的性能评估 Petri 网模型(PEPN)是一个标记图(Mark Graph)，上述问题可以转化为一个线性规划问题[7]。该线性规划问题描述如下：

$$\tau=\max Y^T(C^-)^T\theta$$

$$C\cdot Y=0$$

$$st.\ Y^T\cdot M_0=1$$

$$Y\geqslant 0$$

其中，Y 是线性规划的变量，θ 是由变迁的实施时间构成的 n 维整数向量。由于线性规划问题具有处理速度快的特点，该方法可以应用于较大规模的静态数据流图模型，可以对门级电路的静态数据流图模型直接进行性能分析。

6 实验结果

为了验证本文提出的模型的有效性和执行效率，本文对一些典型的测试电路进行了试验。测试所使用的电路均选自标准测试电路集 ISCAS'89，选择了其中 9 个较有代表性的电路作为测试电路。这 9 个不同的测试电路有着不同的电路结构和规模，可以较为完整地测试本文所提出模型的有效性。

实验运行平台的 CPU 为 Pentium M 2.0GHz，2GB 内存。首先将测试电路的门级网表转化为静态数据流图，然后分别提取对应静态数据流图的 CPN 模型和 PEPN 模型，试验结果比较了不同模型的大小，即两种模型中变迁节点和位置节点的数目，并且得出了对 PEPN 模型进行性能评估所需的运算时间。表 1 列出了实验的结果。

表 1 PEPN 模型和 CPN 模型的比较

	CPN		PEPN	
测试电路	变迁	位置	变迁	位置
s27	36	40	16	18
s298	280	300	147	153
S344	396	412	190	653
S349	404	428	191	201
S386	342	380	171	193
S420	496	524	250	297
S510	440	452	223	231
S526	470	488	235	237
S1488	2656	2728	665	871

从该结果可以看出，采用 PEPN 模型，可以有效地减小静态数据流图的性能评价 Petri 网模型的规模，运算速度非常快，可以应用到迭代型的优化算法中。采用 PEPN 模型，不仅避免了引入 read-arc，而且大量地减少了最终模型中变迁和位置节点的数目。但由于 PEPN 模型只考虑了对系统中和性能评估相关的系统行为进行建模，而忽略了很多静态数据流图执行模型中的细节，这样使得该模型只适合于对静态数据流图进行性能分析，而不能用于对静态数据流图的一些性质进行形式化检验。总而言之，PEPN 模型可以很高效地解决静态数据流图(SDFS)的性能分析问题，它和 CPN 模型一起，可以很完整、高效地完成基于静态数据流图(SDFS)的异步电路形式化验证和性能分析工作。

结束语 静态数据流图(SDFS)是异步电路的一种描述工具，可以从高层到底层的多个层次形式化地描述异步电路。相对于传统的 STG 或者 Petri 网而言，静态数据流图有着灵活性和可用性强的优点，非常直观，很容易被设计人员掌握。但是静态数据流图由于有着自身特殊的定义和语义，很难使用传统的 Petri 网分析工具和性能分析方法。为了解决静态数据流图的性能分析问题，本文提出了一种可以对静态数据流图(SDFS)进行性能分析的 Petri 网模型 PEPN。本质上，该 Petri 网模型是一个标记图，可以利用线性规划求得该模型的最大平均周期的下限。相对于静态数据流图(SDFS)的复杂 Petri 网模型(CPN)来说，该模型的规模更小，而且避免了 CPN 中引入的 read-arc，更适合于对静态数据流图的性能评价。但 PEPN 模型所做的简化同样导致 PEPN 模型不能够表达 CPN 模型所能表达的静态数据流图执行语义模型的一些细节，从而不适于对静态数据流图模型进行形式化验证。通过对这两个模型的综合运用，可以很好地解决静态数据流图的分析和验证问题。

参 考 文 献

[1] Furber S B, Day P, Garside J D, et al. The design and evaluation of an asynchronous microprocessor[C]// Proc. Int'l. Conf. Computer Design. October 1994: 217-220

[2] Sparso J, Furber S B. Principles of Asynchronous Circuit Design: A Systems Perspective[M]. Kluwer Academic Publishers, 2001

[3] Sokolov D, Poliakov I, Yakovlev A. Asynchronous data path models[C]// Proc. International Conference on Application of Concurrency to System Design (ACSD). 2007

[4] Poliakov I, Sokolov D, Mokhov A. Workcraft: A static data flow structure editing, visualisation and analysis tool[C]// Proc. Petri Nets and Other Models of Concurrency(ICATPN). 2007

[5] Silva M, Colom J M. Convex Geometry and Semiflows in P/T Nets: A Comparative Study of Algorithms for Computation of Minimal P-Semiflows[C]// Lectures Notes in Computer Science. vol. 483, Berlin, Germany: Springer-Verlag, 1991: 79-112

[6] Murata T. Petri nets: Properties, analysis and applications[J]. Proc. IEEE, 2006, 77(4): 541-580

[7] 王蕾. 异步嵌入式微处理器设计和分析关键技术研究[D]. 长沙：国防科学技术大学，2006

(上接第 196 页)

[9] 史开泉，朱常青. 双枝模糊集(VI)[J]. 山东工业大学学报，1999(1): 52-62

[10] 史开泉，刘华文. 双枝模糊集(VII)[J]. 山东工业大学学报，1999(3): 233-240

[11] 史开泉，解树霞. 双枝模糊集(VIII)[J]. 山东工业大学学报，1999(6): 178-186

[12] 史开泉，李岐强. 双枝模糊决策与决策识别问题[J]. 中国工程科学，2001, 1(3): 71-77

[13] 史开泉. 双枝模糊决策及其在市场位置选择中的应用[R]. 济南：山东大学自动化工程系，2000

[14] 陈守煜. 系统模糊决策理论与应用[M]. 大连：大连理工大学出版社，1994: 12-22