



(12) 发明专利

(10) 授权公告号 CN 101395609 B

(45) 授权公告日 2012. 05. 30

(21) 申请号 200680044898. 6

(51) Int. Cl.

(22) 申请日 2006. 10. 26

G06F 17/50 (2006. 01)

(30) 优先权数据

11/290, 368 2005. 11. 30 US

(56) 对比文件

US 6868531 B1, 2005. 03. 15, 全文.

US 6643826 B2, 2003. 11. 04, 全文.

(85) PCT申请进入国家阶段日

2008. 05. 30

审查员 吴敏

(86) PCT申请的申请数据

PCT/US2006/060261 2006. 10. 26

(87) PCT申请的公布数据

W02007/102887 EN 2007. 09. 13

(73) 专利权人 飞思卡尔半导体公司

地址 美国得克萨斯

(72) 发明人 迈克尔·W·迪尔 约翰·W·伍尔西

(74) 专利代理机构 中原信达知识产权代理有限

责任公司 11219

代理人 黄启行 穆德骏

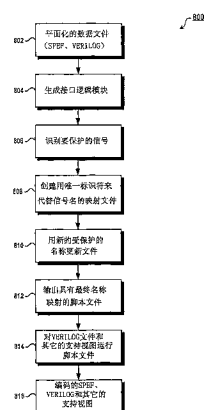
权利要求书 2 页 说明书 7 页 附图 7 页

(54) 发明名称

用于保护 EDA 工具设计视图中的信息的方法和系统

(57) 摘要

在将设计视图转换为电路设计的仿真模型的一部分之前,通过用不相关的唯一标识符 (808) 来代替所选择的实例名和网名,来排除 HDL 接口逻辑模型中的敏感电路设计信息 (806),诸如某些 EDA 工具设计视图中的模块名和结构,并且从而避免该信息的未授权的使用。用于对 IC 设计的不同设计视图中的信号名进行编码的方法包括:提供多个设计数据库中包含的名称列表 (806);将名称列表中的每个名称改变为受保护的名称 (808),以及用每个设计视图数据库中相关联的受保护的名称来代替每个所改变的名称 (810,812,814,816)。



1. 一种用于保护信息的方法,包括:

扫描 IC 设计的第一视图中标识所述 IC 设计中的一个或多个逻辑路径的实例名和网名;

将所述第一视图中的一个或多个所选择的实例名和网名中的每一个映射到与其相关联的名称不类似的相关联的唯一标识符;

在所述第一视图中,将一次或多次出现的所映射名称用它的相关联的唯一标识符来代替,以生成所述 IC 设计的第一受保护的视图,使得所述 IC 设计中的一个或多个逻辑路径不在所述第一受保护的视图中公开;

在所述 IC 设计的第二或更多的视图中,将一次或多次出现的所映射名称用它的相关联的唯一标识符来代替,以生成所述 IC 设计的第二受保护的视图,使得所述 IC 设计中的一个或多个逻辑路径不在所述第二受保护的视图中公开。

2. 根据权利要求 1 所述的方法,其中通过使用密码来从所述第一视图中的实例名和网名中选择所述一个或多个名称。

3. 根据权利要求 1 所述的方法,其中每个名称表示在一个或多个实例名和一个或多个网名之间的等级关联。

4. 根据权利要求 1 所述的方法,其中所述第一视图是接口逻辑模型。

5. 根据权利要求 1 所述的方法,其中所述第一和第二视图是网表。

6. 根据权利要求 1 所述的方法,其中用硬件描述语言写所述第一和第二视图。

7. 根据权利要求 1 所述的方法,其中所述第二视图以 Verilog 进行编码。

8. 根据权利要求 1 所述的方法,其中所述第一视图被格式化为标准寄生交换格式。

9. 一种设计保护系统,包括:

用于扫描 IC 设计的第一视图中标识所述 IC 设计中的一个或多个逻辑路径的实例名和网名的装置;

用于将所述第一视图中的一个或多个所选择的实例名和网名中的每一个映射到与其相关联的名称不类似的相关联的唯一标识符的装置;

用于在所述第一视图中,将一次或多次出现的所映射名称用它的相关联的唯一标识符来代替,以生成所述 IC 设计的第一受保护的视图,使得所述 IC 设计中的一个或多个逻辑路径不在所述第一受保护的视图中公开的装置;

用于在所述 IC 设计的第二或更多的视图中,将一次或多次出现的所映射名称用它的相关联的唯一标识符来代替,以生成所述 IC 设计的第二受保护的视图,使得所述 IC 设计中的一个或多个逻辑路径不在所述第二受保护的视图中公开的装置。

10. 一种用于保护多个数据库中的信息的方法,包括:

提供多个数据库中包含的名称的列表,其中所述多个数据库的每个数据库包括与 IC 设计相关的不同信息;

将名称列表中标识所述 IC 设计中的一个或多个逻辑路径的每一个实例名和网名改变成相应的受保护的名称,使得所述 IC 设计中的一个或多个逻辑路径不公开,以及;

用所述多个数据库的每个数据库中的相关联的受保护的名称来代替每个所改变的名称,使得所述 IC 设计中的一个或多个逻辑路径不在所述多个数据库的任何数据库中公开。

11. 根据权利要求 10 所述的方法,其中所述名称列表包含实例名和网名。

12. 根据权利要求 10 所述的方法,其中所述名称列表中的一个或多个名称表示在一个或多个实例名和一个或多个网名之间的等级关联。

13. 根据权利要求 10 所述的方法,其中所述多个数据库包括接口逻辑模型。

14. 根据权利要求 10 所述的方法,其中所述多个数据库包括网表。

15. 根据权利要求 10 所述的方法,其中用硬件描述语言写所述多个数据库。

16. 根据权利要求 10 所述的方法,其中所述多个数据库中的至少一个数据库被格式化为标准寄生交换格式。

17. 根据权利要求 10 所述的方法,其中所述多个数据库中的至少一个以 Verilog 编码。

18. 一种设计保护系统,包括:

用于提供多个数据库中包含的名称的列表的装置,其中所述多个数据库的每个数据库包括与 IC 设计相关的不同信息;

用于将名称列表中标识所述 IC 设计中的一个或多个逻辑路径的每一个实例名和网名改变成相应的受保护的名称,使得所述 IC 设计中的一个或多个逻辑路径不公开的装置;以及

用于用所述多个数据库的每个数据库中的相关联的受保护的名称来代替每个所改变的名称,使得所述 IC 设计中的一个或多个逻辑路径不在所述多个数据库的任何数据库中公开的装置。

用于保护 EDA 工具设计视图中的信息的方法和系统

技术领域

[0001] 本发明一般涉及数字电路的计算机辅助设计领域,更具体地,涉及保护电子设计自动化(EDA)模型中包含的设计信息。

背景技术

[0002] 诸如多层深亚微米互补金属氧化物半导体(CMOS)技术的现代硅技术的进步应归因于允许在单一芯片上形成更大和更复杂的设计,并且因此使得能够在同一个硅基片上放置整个“片上系统”(SOC)设计。这些复杂的设计可以合并例如一个或多个处理器组件、数字信号处理(DSP)组件、存储器、多个通信接口以及图形支持组件。

[0003] SOC设计的出现使电子系统工业向基于先前存在的电路块或“核心”的使用的新设计模式转移。在这种模式下,通过在同一个硅基片上集成一个或多个先前存在的电路块来装配系统。对于IC设计者来说,合并已开发的组件而不是用每个新IC设计来重新设计所有必需的硬件是更有效和更实用的。对于不同的芯片设计厂商,自然地跟随这种转移,以向开发片上系统(SOC)的系统级集成商或者制造商提供先前存在的电路块,并且这些核心经常涉及由除IC集成商之外的这样厂商或团体拥有的知识产权(IP)。

[0004] 由于市场需求继续推动电子系统制造商更快更有效地开发系统,对于电子系统集成商或制造商来说,接收核心的设计模型以便允许仿真及设计集成SOC是有用的。IC设计仿真模型允许IC设计者知道一个组件是否会与另一个组件一起工作来达到设计者的要求。随着设计和方法的复杂度增加,要求越来越详细和透明的核心设计以便允许使用和集成。不幸地,转移到SOC集成商的模型越精确和详细,对那些设计的诸如被盗和未经授权操作的滥用的风险就越大。

[0005] 核心设计和模型相对容易复制、伪造和重新设计,因为他们物理上是作为存储设备上的数据或代码存在的。此外,由于核心的存在而引起的效率提高同样提供了对这些项目的未授权使用、再使用、转让或销售的动机。结果,在设计和方法的复杂性需要转移大量的信息以便允许SOC设计者使用该核心时,采取努力来保护转移到顾客、厂商和参与SOC设计的第三方的IP变得更加重要。

[0006] 因为仿真模型提供了对设计的操作和能力的有价值洞察,所以那些在设计中拥有IP的人在与合作伙伴之间没有谨慎地拟定法律协议时一般不公开该仿真模型。对于所有方,这样协议的谈判都需要花费时间和金钱。如果SOC设计者想要测试出设计或构思,则达成该协议对于快速、有效并且划算的设计开发是相当有害的。此外,由于他们本身,这样的合法方式是一种有限制的使用,因为检测对专有核心的非法复制、伪造、转让或再利用是困难的。对于确定特定用户是否非法获得核心或者已经从它的电子形式中获得了电路设计中的什么敏感信息是极端的挑战。核心的供应商因此需要一种保护他们的设计的有效方法,以便他们花费在核心的设计、开发以及采购上的资源的权益不被剥夺。

附图说明

[0007] 在下面参照附图的描述中,采用优选的实施例来描述本发明,其中相同的数字代表相同或相似的元件,如下:

[0008] 图 1 是本发明的一个实施例的方框图。

[0009] 图 2 是平面门级 SPEF 的例子。

[0010] 图 3 示出了在 Verilog 视图中的模块或核心设计的例子。

[0011] 图 4 示出了实例名的示范性格式。

[0012] 图 5 示出了依据本发明的优选实施例的由设计保护系统执行的替换方案的映射。

[0013] 图 6 示出了依据本发明的优选实施例的受保护的 SPEF 的例子。

[0014] 图 7 示出了依据本发明的优选实施例的受保护的 Verilog。

[0015] 图 8 示出了依据本发明的优选实施例的用于保护 EDA 工具的设计视图中的保密信息的过程的流程图。

[0016] 在本发明的典型示范性实施例的以下详细说明中,对附图进行参照,该附图构成本发明的一部分,并且其中通过说明的方式示出了可以实践本发明的特定示范性实施例。

具体实施方式

[0017] 在本发明的示范性实施例的以下详细描述中,用足够的细节描述了可以实践本发明的特定示范性实施例,以便使本领域技术人员能够实践本发明,并且应该理解的是,可以利用其它实施例并且在不脱离本发明的精神或范围的前提下可以进行逻辑的、结构的、程序的、机械的、电子的以及其它改变。因此,以下的详细描述并不用于限定含义,而本发明的范围仅由所附的权利要求限定。具体地,使用硬件描述语言 (HDL) 表示法或其它符号表示法来表示 IC 设计或 IC 设计的仿真模型的任何方式都被认为在本发明的范围内,这对于本领域普通技术人员是显而易见的。

[0018] 在如这里描述的一个或多个实施例中,提供了一种用于保护被转换为电路设计的仿真模型的一部分的敏感电路设计信息并且从而避免该信息未授权使用的方法和程序产品,其中敏感信息是诸如在某些 EDA 工具设计试图中的模块名和结构。以该方式,本发明的方法和系统使 SOC 设计者能够使用核心设计的仿真模型来确定在不允许访问模型中包含的机密信息的情况下仿真模型的核心和组件的操作是否满足设计者的需要。这部分地减轻了对 SOC 设计者(或公司)与 IP 所有者之间的机密性协议的依赖,以便保护敏感核心设计信息,该信息否则可能包含在仿真模型中。

[0019] 根据特定电路设计所期望的抽象级别,一般作为可以包含多个步骤的设计过程的一部分来生成虚拟组件块。芯片设计者经常使用电子设计自动化 (EDA) 软件工具来辅助设计过程,并允许在原型设计或生产之前仿真芯片设计。使用 EDA 软件工具的芯片设计通常包括反复的过程,从而逐渐地完善芯片设计。典型地,芯片设计者通过在通常具有高质量图形容量的计算机工作站输入信息而创建一个电路,以便根据需要显示一部分电路设计。一般使用硬件描述语言 (HDL) (诸如 Verilog 或 VHDL) 来使用管理严密的设计方法,例如,通过该方法,设计者通过分级定义电路的功能组件、然后将每个组件分解成越来越小的组件来创建集成电路。

[0020] 通过它们的功能操作以及相应的输入和输出来最初定义集成电路的各种组件。在这些设计阶段期间,设计者通常使用大量的分级信息来构造电路,并且典型地已经提供了

设计中的重要规则。可以以不同方式将数字电路的设计表示或规定在计算机系统的存储器中。每种类型的数字电路表示被称作一个视图。不同的视图包含不同数量和类型的有关电路结构的信息。包含更多有关电路结构的信息的视图需要更多的计算机存储器并且需要更多的计算机时间来构造和操作。

[0021] 根据 HDL 或其它高级描述,典型地通过逻辑合成来确定实际逻辑单元实现,所述逻辑合成将电路的功能描述转换为具体的电路实现。然后对逻辑单元进行“放置”(即,给出电路设计图中的具体坐标位置)和“布线”(即,根据设计者的电路定义连线或连接到一起)。放置和布线软件程序通常接收通过逻辑合成处理生成的平面网表作为他们的输入。该平面的或分等级网表识别来自目标标准单元库中的具体逻辑单元实例,并且描述具体的单元到单元的连接。

[0022] IC 芯片包括在半导体基片的表面上形成的单元以及单元之间的连接。单元是一组被分组来执行某个功能的一个或多个电路元件(诸如晶体管、电容器以及其它成组基本电路元件)。IC 可以包括大量的单元并且单元之间需要复杂的连接。IC 的每个单元可以具有一个或多个管脚,每个管脚进而可以通过电线被连接到 IC 的一个或多个其它管脚。在芯片的表面上还形成将管脚连接到 IC 的电线。

[0023] 网是一组两个或更多个相互连接的管脚和电线,从而连接具有管脚的逻辑电路。因为一般的芯片具有几千、几万或几十万个必须采用不同组合连接的管脚,所以芯片还包括几千、几万或几十万的“网”的定义,或者管脚组的连接。对于芯片来说,网的数量的数量级一般与该芯片上的单元的数量级相同。通常,多数网只包括要连接的两个管脚;然而,许多网包括三个或多个管脚。一些网可能包括数百或几万个要连接的管脚。

[0024] IC 模型是典型地用 HDL 表示的一组相互连接的电路单元和网,并且有时在这里可互换地被称作模块、核心、知识产权(IP)、IP 块、块或组件。核心是可以用硬件实现的(典型地用 HDL 规定的,并且典型地也用高级网表表示的)电子设计。“网表”是包括相连管脚的名称的网(“网名”)的列表,或者是包括与单元的管脚连接的网名的单元(“实例名”)的列表。合成高级 RTL 以便在低级的抽象层生成门级网表。在设计的合成之后,核验门级网表,确定电路的设计图,并且制作 IC。

[0025] 如今,多数数字集成电路(IC)的设计是基于 HDL(硬件描述语言)方法的高度结构化的过程。HDL 方法以通过规范文档中规定的 IC 开始。然后,在被称作寄存器转换语言(RTL)的设计抽象层级,把 IC 规范写成高级 HDL 代码。RTL 代码将电路描述为寄存器、布尔等式以及诸如“if-then-else”指令和复杂事件序列的控制逻辑的集合。在抽象的 RTL 级,一般使用诸如 Verilog-HDL(“Verilog”)或 VHDL 的 HDL 语言来实现 RTL 代码,在这里通过描述当数据在电路输入、输出与时钟寄存器之间流动时对数据执行的操作来规定 IC 设计。在 RTL 层级,设计者必须做出所有的关键设计决定,诸如设计等级和分区、时钟方案、复位方案以及寄存器的位置。所有那些决定都包含并反映在 RTL 代码中。RTL 代码一般是独立于技术,也独立于设计工具。

[0026] 然后合成如用 RTL 代码所表示的 IC 设计,以便生成门级描述或网表。合成是用于将由 RTL 代码表示的设计的结构和功能描述翻译成设计的较低级表示(诸如逻辑级和门级描述)所采用的步骤。IC 设计规范和 RTL 代码一般是独立于技术的。也就是说,该规范和 RTL 代码一般不指定用于实施设计的确切的门或逻辑设备。IC 设计的门级描述可以是依赖

于技术的。这是因为在合成处理期间合成工具可能使用给定的技术库来将独立于技术的 RTL 代码映射成依赖于技术的门级网表。然而,网表可以是通用的或者专用的技术。通用的网表是从还没有与单元的专用技术库相关联的 RTL 代码来创建的网表。专用技术网表或映射网表是在已经将 IC 设计映射到单元的特定专用技术库之后创建的网表。

[0027] 如这里所描述的本发明的实施例适用于各种各样的虚拟组件块,无论他们是否涉及系统芯片设计中使用的系统级宏、超级单元、或嵌入式软件“核心”。虚拟组件块一般可以根据它们的“稳固性”被分类到设计阶段的抽象级为止。例如,虚拟组件块可以被分类为三个稳固类别之一:“软”、“稳固”和“硬”。在从功能描述到特定的制作过程的范围内,稳固性分类表现开发阶段的特色。

[0028] 一般以可合成的硬件描述语言(诸如,例如,VHDL 或 Verilog)的形式将“软”虚拟组件块传递给 SOC 集成者。软虚拟组件是高度灵活的,并且可以是通用的和可配置的。对于普通技术,“稳固”虚拟组件块大部分是为通用技术优化的结构型模型。稳固虚拟组件块的详细级别的范围可以例如从其中经由相对放置的数据路径指定 RTL 子块的区域放置的电路设计到具有完全放置的网表的电路设计。稳固虚拟组件块一般不包括布线并且在整个目标技术上是轻便的。“硬”虚拟组件被映射到专用技术,并且一般是对于目标技术中的功率、尺寸和性能来优化的。硬虚拟组件的例子可以包括自定义(custom)物理设计图或网表,该网表被完全放置、布线和对所给定技术进行的优化。硬虚拟组件块是专用的处理或厂商,并且通常是用诸如 GDS II 的标准设计语言表示的。为了有效地使用硬虚拟组件块,用户一般有权访问连同 GDS II 文件一起的高级行为模型和完整的物理/时序模型。

[0029] 为了简化 SOC 的设计和分析任务,可以将复杂的设计分成块,即,芯片中任何用户或系统定义的层次。在这一点上,块可以用接口模型来代替。有利地,例如,这样的模型可以提供与块的 SOC 集成商交换块的接口信息的紧凑方式。代替块或核心的完整门级网表的接口模型的使用是提高与芯片级分析和模拟相关的性能和容量的关键,尤其对于门的数量超过五百万个门的复杂设计。此外,这样的接口逻辑模型允许芯片设计者保留并保护不需要随该接口逻辑模型(ILM)转换的内部门级设计。

[0030] 自动模型生成可以包括 ILM 的使用。ILM 丢弃与设计上的内部寄存器到寄存器路径有关的网表、制约以及备份注释(back-annotation),从而提供运行和性能方面的显著改进而不牺牲准确性。此外,ILM 只包含原始网表的接口逻辑。例如,仅包含在块上的寄存器到寄存器路径中的逻辑不被放到 ILM 中。ILM 包括时序路径中包含的单元(即,逻辑和相关网(这里称作接口网)的组合),该时序路径开始于输入端口并且结束于沿触发的寄存器(即,时序逻辑)或输出端口。如果在这样的时序路径中遇到透明锁存器,则将它组合逻辑来进行处理,并且路径跟踪继续穿过该锁存器,直到遇到沿触发的寄存器或输出端口为止。

[0031] ILM 可以被用作设计的接口时序的仿真模型表示。可以从设计 Verilog 网表和适当的寄生信息中来生成 ILM 模型。组件的 ILM 模型使能够详细理解和探究组件相对于时序如何操作以及如何与其它组件相互作用。

[0032] 现在参照附图,具体地参照图 1,这里示出了本发明的一个实施例的方框图。用 Verilog 来实现 IC 设计信息(Verilog 设计)102。IC 设计寄生信息(SPEF)104 与 IC 设计信息 102 相关并且是以标准寄生交换格式(SPEF)。Verilog 设计 102 和 SPEF 104 共同为核心或设计块提供设计信息的重要集合。使用标准的 EDA 工具,SPEF 104 可以被转换成 ILM

106, 而 Verilog 设计 102 可以被转换成 Verilog ILM 108。当生成 ILM 时, EDA 工具格式化平面门级网表, 因为有时被 EDA 工具用来电路模拟、时序验证以及电路布置和布线。SPEF ILM 106 和 Verilog ILM 108 的每一个都代表电路设计的模型视图。如可以理解的是, 每个模型视图包含可以提供对模块的操作和性能的洞察的机密仿真信息。

[0033] 图 2 是平面门级 SPEF 的例子。网表 200 包含名称映射 202、端口寄生信息 204 以及网连接和寄生信息 206。

[0034] 图 3 示出了 Verilog 视图中的模块或核心设计的例子。平面门级网表 300 定义了每个“实例名”、每个“网名”以及在实例名与网名之间的等级关系。接口 302 识别连接到模块顶层的输入和输出的网。网名 304 识别未连接到模块顶层的输入或输出的网的每个实例。实例名 306 识别视图中的接口单元的每个实例。图 4 示出了通过实例名 402、多个输入 404, 406 (或更多) 以及输出 408 识别的实例名 306 的格式。

[0035] 返回参照图 1, 设计保护系统 110 对外部设计者屏蔽仿真模型、核心和 / 或组件的专有部分的公开, 但是仍然为设计者提供了有关组件、核心或模型是否符合设计目标的理解。根据优选实施例, 设计保护系统 110 重新定义了实例名和网名以及 IC 模型中可能包含的相关联的等级信息, 从而保护机密信息。设计保护系统 110 采用一致的方式重新定义贯穿所有设计视图的特定数据。通过使用密码来对设计的第一视图进行编码, 以重新定义视图中的某些实例名和网名。当平面网表中的实例名、网名或等级关系识别完整的逻辑路径时, 用关于该实例或网的单个唯一标识符来代替它们。然后, 采用完全相同的方式来对相关联的互补视图进行编码, 以便所有视图保持一致。

[0036] 在优选实施例中, 设计保护系统 110 中的脚本生成剔除某些或所有敏感或机密信息的受保护的 SPEF ILM 112 以及受保护的 Verilog ILM 114, 但是仍然允许第三方使用者与模型进行完整的交互。受保护的 SPEF ILM 112 以及受保护的 Verilog ILM 114 导致以与标准 EDA 工具完全兼容的格式来模拟核心。

[0037] 图 5 示出了依据本发明的优选实施例的由设计保护系统 110 中的脚本执行的代替方案的映射, 用于对每个视图 102、104 或 ILM 106、108 中发现的实例名、网名和等级关系进行编码。为了描述的目的, 设计保护系统 110 被描述为用于保护 ILM 106、108 (为了说明起见, 被认为是从示范性视图 200、300 取得的), 然而, 设计保护系统 110 可用于保护大多数的任何设计视图, 例如包括视图 102、104。映射密钥 500 的每一行具有一个指定 SPEF 104 中的实例名的标识符 502 (如在前两个“!”之间定义的) 和一个代替标识符 502 的编码标识符 504 (如在第二和第三个“!”之间定义的)。例如, 用在行 506 中所示的唯一标识符 504 (即, “JWW12JWW”) 来代替行 506 中的实例名 502 (即, 在名称映射 202 的行“*12”发现的“and_gate_3”)。在这个例子中, 已经通过识别其中识别实例名的 SPEF ILM 106 中的行 (从 SPEF 104 转换之后) 并且用预先选择的字母预先计划和添加它来创建编码标识符 504。本领域技术人员应理解, 可以使用任何命名或编码方案, 只要为网表中的每个实例名、网名和等级关系给予一个唯一并非共享的标识符即可。映射密钥 500 也使用编码的实例名和未编码的实例名两者来提供对网名的编码。例如, 如图 5 的行 508 中所示的, 创建唯一标识符“JWW17JWW”, 以识别连接以前重新命名的实例“JWW12JWW”和单元“nand_gate”的网。

[0038] 在优选实施例中, 通过为名称映射 (从 SPEF 200 中的名称映射 202 取得的) 扫描 SPEF ILM 106 的脚本来实现设计保护系统 110, 以便识别要编码的实例名和网名。通过扫

描,设计保护系统 110 生成编码映射 500,以便使用与网表中包含的名称或其它机密信息不相关的命名法来唯一识别每个所识别的名称。设计保护系统 110 然后生成受保护的 SPEF ILM 112 作为完整功能的 ILM 设计视图,但是从网表中剔除了所选择的敏感或机密信息。

[0039] 图 6 示出了依据本发明的优选实施例的受保护的 SPEF 的例子。在该例子中,图 2 的例子中的名称映射 202 的行“*12”-“*17”已经被确定为是机密信息,并且已经使用映射密钥 500 将其编码成唯一代码 602,以生成受保护的 SPEF 600。

[0040] 设计保护系统 110 然后继续使用同一映射密钥 500 来保护核心设计的所有其它相关的设计视图。在优选实施例中,设计保护系统 110 使用映射密钥 500 从 Verilog ILM 108 生成受保护的 Verilog ILM 114。

[0041] 图 7 中示出了通过将映射密钥 500 应用于 Verilog 300 而生成的受保护的 Verilog 文件(被实现为 HDL 网表 700)的例子。Verilog 300 中的网 304 已经被编码为如受保护的网名 702 所示的唯一命名的连接。类似地,实例名 306 已经根据映射密钥 500 被编码为受保护的实例名 704。例如,已使用不相关联的实例名“JWW17JWW”和唯一识别的输出“JWW30JWW”将实例名 306 的第一个“NAND3x1”编码为第一个受保护的实例名(704 的)。

[0042] 现在参照图 8,示出了依据本发明的优选实施例的用于保护 EDA 工具的设计视图中的机密信息的处理的流程图。处理 800 在步骤 802 开始,在该步骤,描述 IC 设计的 HDL 代码被平面化以便删除分级结构。例如,当保护为了模拟的目的而要转移到第三方的文件时,SPEF 和 Verilog HDL 文件被平面化。处理然后进行到步骤 804,在该步骤生成 ILM 模型。处理然后进行到步骤 806,在该步骤识别要受保护的信号名。在这里,处理将解析 SPEF 文件以确定 SPEF 文件中要保留或被重命名的信号名的数量和位置。例如,解析 SPEF 文件以确定模块端口的数量和位置,以便以其原始的形式保留他们。

[0043] 此后,处理进行到步骤 808,在该步骤,通过将所选择的信号名映射到唯一标识符来创建用于将信号名与受保护的信号名相关联的映射文件,所述唯一标识符与原始信号名不相关但是唯一地识别要彼此区分的每个信号名。在优选实施例中,不映射或代替在端口名中包含的信号名。处理然后进行到步骤 810,在该步骤,通过扫描关于原始信号名的实例的文件并且用他们所映射的受保护名来代替他们,来利用受保护的信号名对 SPEF 文件进行编码。此后,处理进行到步骤 812,在该步骤,生成包含代码的脚本文件,所述代码用于将信号名映射到他们的唯一受保护的信号名。此后,处理进行到步骤 814,在该步骤,对包含与第一文件相同的信号名的 IC 的 Verilog 设计文件和其它支持设计视图执行脚本文件。当运行时,脚本对所涉及的设计视图文件执行步骤 806,808 和 810,以便通过用他们唯一映射的受保护信号名代替他们来保护所选信号名的每个实例。通过以受保护的格式输出所编码的 SPEF、Verilog 和其它支持的设计视图,在步骤 816 结束处理,所述格式隐藏模块名和结构,但是仍然保留每个设计视图之间的一致性并且通过设计视图的第三方接收器来提供在 EDA 工具中的完全可用性。

[0044] 如应理解的,本发明的优选实施例允许以有效的方式过滤多个 IC 设计视图中的机密信息,以便呈现受保护的设计视图而不丢失必要信息,以允许标准 EDA 工具对整个多个视图的利用。因为使用映射密钥来执行对受保护的设计视图的编码,所以原始设计者可以容易地翻译来自受保护文件设计视图的用户的注释或技术问题。为了支持用户与受保护的设计视图交互关于技术设计的问题或疑问,这是重要的。

[0045] 如应理解的,可以使用计算机程序软件、固件或硬件的任意组合来实现本发明优选实施例中的处理。作为用软件来实践本发明的预备步骤,根据优选实施例的计算机程序代码(软件或固件)典型地被存储到一个或多个诸如固定(硬)驱动器、磁盘、光盘、磁带的机器可读存储介质中、诸如 ROM、PROM 的半导体存储器等中,从而能够制造根据本发明的物品。通过执行直接来自存储设备的代码、通过将存储设备的代码复制到诸如硬盘、RAM 等的另一个存储设备中、或者通过发送代码来远程执行,来使用包含计算机程序代码的制造物品。通过将一個或多个包含根据本发明的代码的机器可读存储设备与适当的标准计算机硬件组合来执行其中所包含的代码,可以实践本发明的方法形式。用于实践本发明的装置可以是包含或具有访问根据本发明编码的计算机程序的网络的一个或多个计算机和存储系统。

[0046] 虽然已经参照优选实施例具体示出并描述了本发明,但是本领域技术人员应理解,在不背离本发明的精神和范围的情况下,可以对其中的形式和细节上做出各种改变。对于所描述的实施例的任何变化、修改、增加以及改进都是可能的并且可以落入如以下权利要求中详细描述的本发明的范围内。

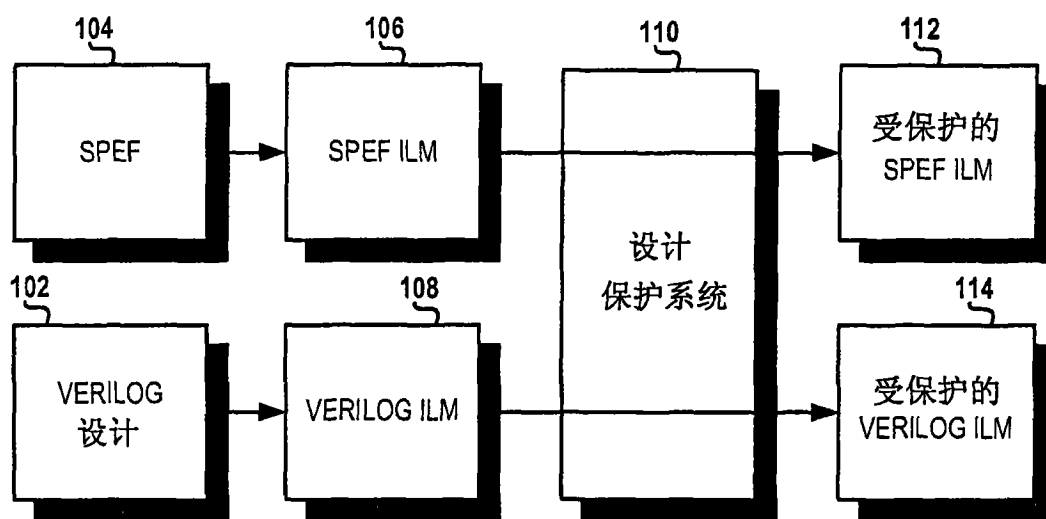


图 1

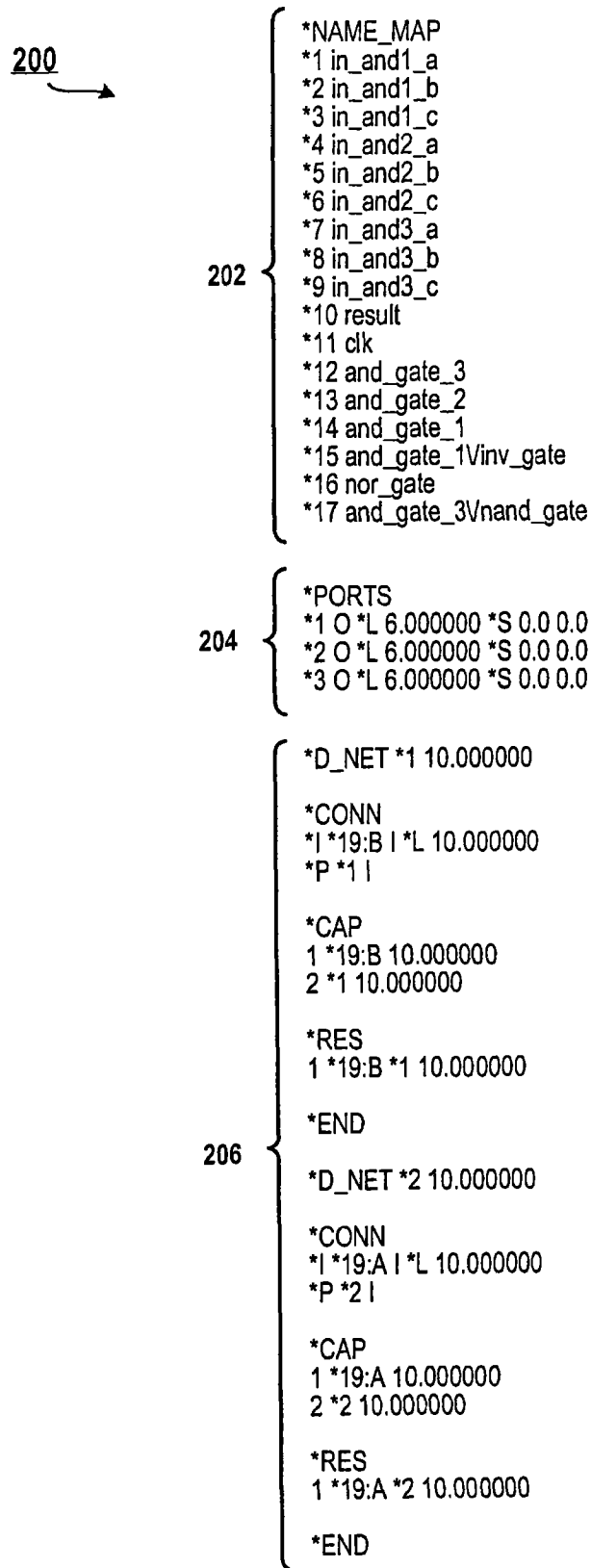


图 2

300

```

302 { module top_example ( in_and1_a, in_and1_b, in_and1_c, in_and2_a, in_and2_b,
        in_and2_c, in_and3_a, in_and3_b, in_and3_c, result, clk );
    input in_and1_a, in_and1_b, in_and1_c, in_and2_a,
        in_and2_b, in_and2_c, in_and3_a, in_and3_b, in_and3_c, clk;
    output result;

304 { wire \and_gate_3/and_z_reg_reg ;
        wire \and_gate_3/nand_z ;
        wire \and_gate_3/nand_z_reg ;
        wire \and_gate_3/and_z_reg ;
        wire \and_gate_2/and_z_reg_reg ;
        wire \and_gate_2/nand_z ;
        wire \and_gate_2/nand_z_reg ;
        wire \and_gate_2/and_z_reg ;
        wire \and_gate_1/and_z_reg_reg ;
        wire \and_gate_1/nand_z ;
        wire \and_gate_1/inv_gate/and_z_reg ;
        wire \and_gate_1/nand_z_reg ;

306 { NAND3x1 \and_gate_3/nand_gate ( .A(in_and3_b), .B(in_and3_a), .C(in_and3_c), .Z(\and_gate_3/
        nand_z ));
        NAND3x2 \and_gate_2/nand_gate ( .A(in_and2_b), .B(in_and2_a), .C(in_and2_c), .Z(\and_gate_2/
        nand_z ));
        NAND3x1 \and_gate_1/nand_gate ( .A(in_and1_b), .B(in_and1_a), .C(in_and1_c), .Z(\and_gate_1/
        nand_z ));
        DFFx1 \and_gate_3/delay_1 ( .D(\and_gate_3/nand_z ), .CK(clk), .Q(\and_gate_3/nand_z_reg ));
        INVx8 \and_gate_3/inv_gate ( .A(\and_gate_3/nand_z_reg ), .Z(\and_gate_3/and_z_reg ));
        DFFx1 \and_gate_3/delay_2 ( .D(\and_gate_3/and_z_reg ), .CK(clk), .Q(\and_gate_3/
        and_z_reg_reg ));
        DFFx1 \and_gate_2/delay_1 ( .D(\and_gate_2/nand_z ), .CK(clk), .Q(\and_gate_2/nand_z_reg ));
        INVx1 \and_gate_2/inv_gate ( .A(\and_gate_2/nand_z_reg ), .Z(\and_gate_2/and_z_reg ));
        DFFx1 \and_gate_2/delay_2 ( .D(\and_gate_2/and_z_reg ), .CK(clk), .Q(\and_gate_2/
        and_z_reg_reg ));
        DFFx1 \and_gate_1/delay_1 ( .D(\and_gate_1/nand_z ), .CK(clk), .Q(\and_gate_1/nand_z_reg ));
        INVx1 \and_gate_1/inv_gate/inv_gate ( .A(\and_gate_1/nand_z_reg ), .Z(\and_gate_1/inv_gate
        and_z_reg ));
        DFFx1 \and_gate_1/delay_2 ( .D(\and_gate_1/inv_gate/and_z_reg ), .CK(clk),
        .Q(\and_gate_1/and_z_reg_reg ));
        NOR3x1 nor_gate ( .A(\and_gate_2/and_z_reg_reg ), .B(\and_gate_3/and_z_reg_reg ), .C(\
        and_gate_1/and_z_reg ), .Z(result) );
    endmodule

```

图 3

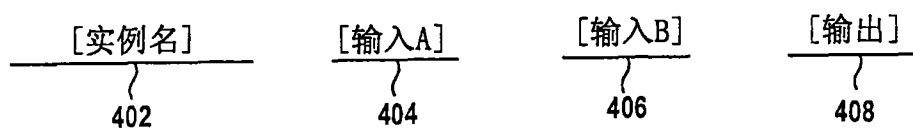


图 4

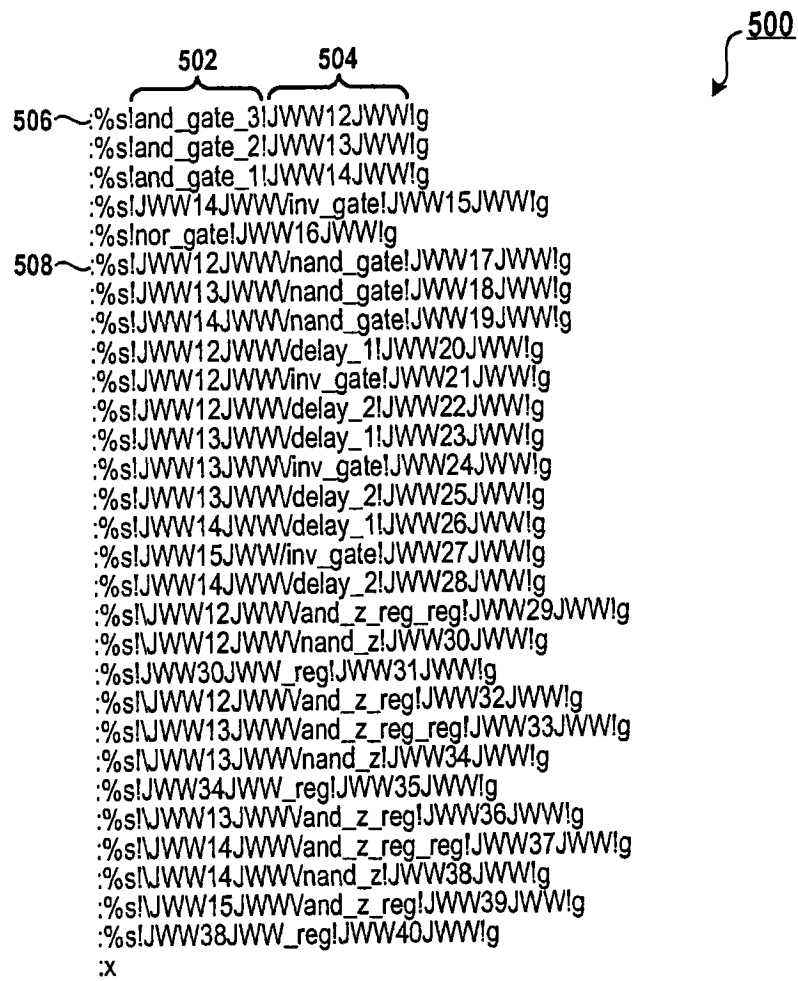


图 5

```
*NAME_MAP
*1 in_and1_a
*2 in_and1_b
*3 in_and1_c
*4 in_and2_a
*5 in_and2_b
*6 in_and2_c
*7 in_and3_a
*8 in_and3_b
*9 in_and3_c
*10 result
*11 clk
602 { *12 JWW12JWW
      *13 JWW13JWW
      *14 JWW14JWW
      *15 JWW15JWW
      *16 JWW16JWW
      *17 JWW17JWW

*PORTS
*1 O *L 6.000000 *S 0.0 0.0
*2 O *L 6.000000 *S 0.0 0.0
*3 O *L 6.000000 *S 0.0 0.0

*D_NET *1 10.000000
*CONN
*I *19:B I *L 10.000000
*P *1 I
*CAP
1 *19:B 10.000000
2 *1 10.000000
*RES
1 *19:B *1 10.000000
*END
*D_NET *2 10.000000
*CONN
*I *19:A I *L 10.000000
*P *2 I
*CAP
1 *19:A 10.000000
2 *2 10.000000
*RES
1 *19:A *2 10.000000
*END
```

图 6

```

module top_example ( in_and1_a, in_and1_b, in_and1_c, in_and2_a, in_and2_b, 700
                    in_and2_c, in_and3_a, in_and3_b, in_and3_c, result, clk );
input in_and1_a, in_and1_b, in_and1_c, in_and2_a, in_and2_b, in_and2_c,
      in_and3_a, in_and3_b, in_and3_c, clk;
output result;

702 { wire \JWW29JWW;
      wire \JWW30JWW;
      wire \JWW31JWW;
      wire \JWW32JWW;
      wire \JWW33JWW;
      wire \JWW34JWW;
      wire \JWW35JWW;
      wire \JWW36JWW;
      wire \JWW37JWW;
      wire \JWW38JWW;
      wire \JWW39JWW;
      wire \JWW40JWW;

704 { NAND3x1 \JWW17JWW ( .A(in_and3_b), .B(in_and3_a), .C(in_and3_c), .Z(\JWW30JWW) );
      NAND3x2 \JWW18JWW ( .A(in_and2_b), .B(in_and2_a), .C(in_and2_c), .Z(\JWW34JWW) );
      NAND3x1 \JWW19JWW ( .A(in_and1_b), .B(in_and1_a), .C(in_and1_c), .Z(\JWW38JWW) );
      DFFx1 \JWW20JWW ( .D(\JWW30JWW), .CK(clk), .Q(\JWW31JWW) );
      INVx8 \JWW21JWW ( .A(\JWW31JWW), .Z(\JWW32JWW) );
      DFFx1 \JWW22JWW ( .D(\JWW32JWW), .CK(clk), .Q(\JWW29JWW) );
      DFFx1 \JWW23JWW ( .D(\JWW34JWW), .CK(clk), .Q(\JWW35JWW) );
      INVx1 \JWW24JWW ( .A(\JWW35JWW), .Z(\JWW36JWW) );
      DFFx1 \JWW25JWW ( .D(\JWW36JWW), .CK(clk), .Q(\JWW33JWW) );
      DFFx1 \JWW26JWW ( .D(\JWW38JWW), .CK(clk), .Q(\JWW40JWW) );
      INVx1 \JWW27JWW ( .A(\JWW40JWW), .Z(\JWW39JWW) );
      DFFx1 \JWW28JWW ( .D(\JWW39JWW), .CK(clk), .Q(\JWW37JWW) );
      NOR3x1 \JWW16JWW ( .A(\JWW33JWW), .B(\JWW29JWW), .C(\JWW37JWW), .Z(result) );
endmodule

```

图 7

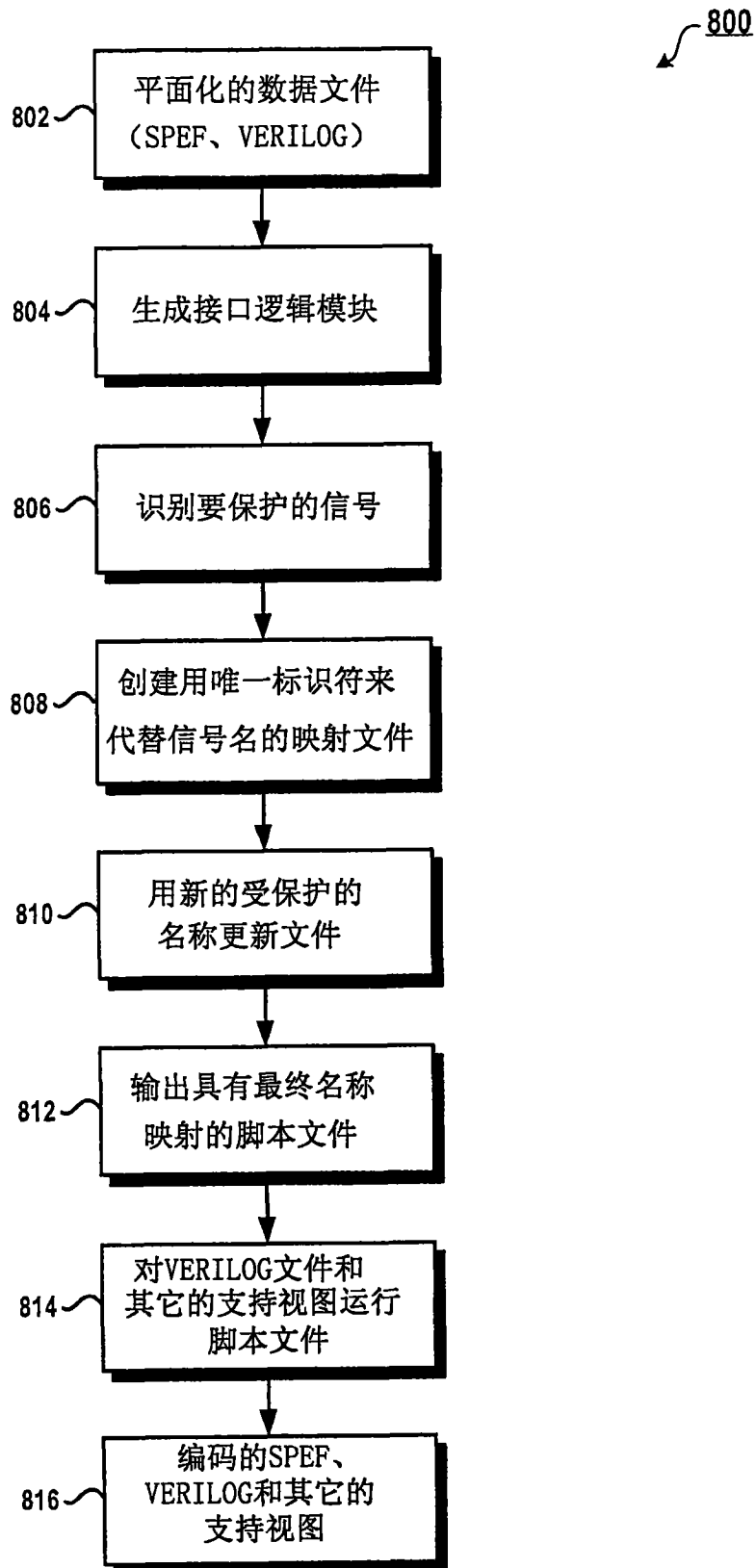


图 8