(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 107357948 B
(45)授权公告日 2023.04.07

(21)申请号 201710171544.8
(22)申请日 2017.03.22
(65)同一申请的已公布的文献号
申请公布号 CN 107357948 A
(43)申请公布日 2017.11.17
(30)优先权数据
16168899.9 2016.05.10 EP
(73)专利权人 德斯拜思有限公司
地址 德国帕德博恩
(72)发明人 H•卡尔特 D•卢贝雷
(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
专利代理师 刘盈
(51)Int.Cl.
G06F 30/327(2020.01)
G06F 30/331(2020.01)
(56)对比文件
CN 101933098 A,2010.12.29
US 2002162084 A1,2002.10.31
US 2011184713 A1,2011.07.28
US 2015347669 A1,2015.12.03
匡春雨.基于GK803的SoC设计与验证平台实现.《中国优秀硕士学位论文全文数据库 信息科技》.2014,第1-72页.
Eddie Hung,et al.Delay-Bounded Routing for Shadow Registers.《FPGA ″15: Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays》.2015,Pages 56-65.
Jason Xin Zheng,et al.Securing netlist-level FPGA design through exploiting process variation and degradation.《FPGA ″12: Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays》.2012,Pages 129-138.
审查员 舒志勇

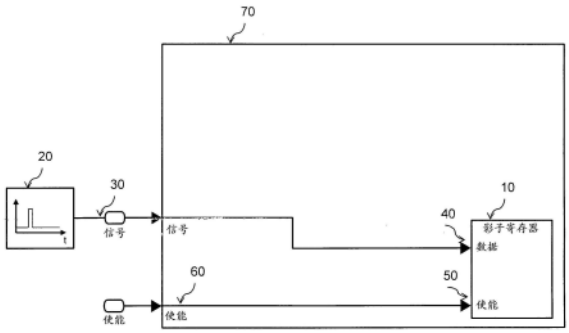
权利要求书2页 说明书8页 附图9页

(54)发明名称

用于建立FPGA网表的方法

(57)摘要

本发明涉及一种用于建立FPGA网表的方法，其中，网表从FPGA源代码和至少一个影子寄存器中产生，其中，FPGA源代码定义至少一个功能和至少一个信号，其中，将影子寄存器配设给所述至少一个信号并且将影子寄存器设立和设置用于在运行时存储所配设的信号的值，其中，设置并设立用于在运行时读取所存储的信号值的器件，其中，在FPGA源代码中所定义的功能不通过影子寄存器改变，其中，将网表设置用于加载到FPGA上并且由FPGA执行，其中，所述由FPGA源代码所描述的功能由FPGA执行，其中，设置和设立所述影子寄存器与在FPGA源代码中所描述的功能的在功能上的解耦，其中，在执行所述在FPGA源代码中所描述的功能期间，影子寄存器通过解耦来保留在解耦的时间点上所存储的信号值。



CN 107357948 B

1.用于建立FPGA网表的方法，其中，网表从FPGA源代码(20)和至少一个影子寄存器(10)中产生，其中，FPGA源代码(20)定义至少一个功能和至少一个信号(30)，其中，将影子寄存器(10)配设给所述至少一个信号(30)并且将影子寄存器设置用于在运行时存储所配设的信号(30)的值，其中，设置用于在运行时读取所存储的信号值的器件，其中，在FPGA源代码(20)中所定义的功能不通过影子寄存器(10)改变，其中，将网表设置用于加载到FPGA上并且由FPGA执行，其中，所述由FPGA源代码(20)所描述的功能由FPGA执行，其特征在于，设置所述影子寄存器(10)与在FPGA源代码(20)中所描述的功能的在功能上的解耦，其中，在执行所述在FPGA源代码(20)中所描述的功能期间，影子寄存器(10)通过解耦来保留在解耦的时间点上所存储的信号值。

2.根据权利要求1所述的方法，其特征在于，所述FPGA源代码(20)定义多个信号(30)，其中，多个影子寄存器(10)分别配设给一个信号(30)，其中，在功能上的解耦设置用于同步地解耦所述多个影子寄存器(10)。

3.根据权利要求1或2所述的方法，其特征在于，为了解耦，将影子寄存器(10)的使能信号(60)或影子寄存器(10)的时钟信号(700)中断。

4.根据权利要求1或2所述的方法，其特征在于，所述FPGA源代码(20)作为图形化模型或作为文本代码存在。

5.根据权利要求1或2所述的方法，其特征在于，将影子寄存器(10)加入到FPGA源代码(20)或FPGA源代码(20)的副本中。

6.根据权利要求1或2所述的方法，其特征在于，从FPGA源代码(20)中生成网表并且将影子寄存器(10)加入到该网表中。

7.根据权利要求1或2所述的方法，其特征在于，将影子寄存器(10)自动化地加入并且配设给所述信号(30)。

8.根据权利要求1或2所述的方法，其特征在于，自动化地检验，所述信号(30)是否已经在FPGA源代码(20)的其他位置上配设给一个影子寄存器(10)，并且如果是，就不给该信号(30)配设另外的影子寄存器。

9.根据权利要求1或2所述的方法，其特征在于，加入至少两个影子寄存器(10)并且将所述至少两个影子寄存器配设给信号(30)，其中，将第一影子寄存器设置用于在运行时在第二影子寄存器解耦期间存储当前信号值。

10.根据权利要求1或2所述的方法，其特征在于，在建立网表时，设置通过FPGA的外部和/或内部的回读接口来对影子寄存器(10)的读取。

11.根据权利要求1或2所述的方法，其特征在于，加入多个影子寄存器(10)，其中，将所述多个影子寄存器(10)连接成一个移位寄存器链并且所述多个影子寄存器设置用于通过FPGA的外部接口来读取。

12.根据权利要求1或2所述的方法，其特征在于，加入多个影子寄存器(10)，其中，将地址译码器设置用于通过FPGA的外部接口来读取所述多个影子寄存器(10)。

13.根据权利要求1或2所述的方法，其特征在于，附加于影子寄存器加入逻辑电路(110、140)，其中，将逻辑电路(110、140)设置用于在运行时在信号值变化时输出触发信号(130)，触发信号(130)引起影子寄存器的解耦。

14.根据权利要求1或2所述的方法，其特征在于，在加入影子寄存器(10)之前实施以下

步骤：

-确定在FPGA源代码(20)中的所有如下常数，第一信号值与这些常数有关；

-为这些常数的找到的值确定最小必需的位宽；

-将这些常数重新配置到相应确定的最小所需的位宽上或者将这些常数事后构成到相应确定的最小所需的位宽上；

-通过源代码传播所述位宽。

15.根据权利要求1或2所述的方法，其特征在于，所述影子寄存器(10)在建立和/或处理网表时被保护免于路径优化。

16.用于建立FPGA网表的数据处理装置，其中，所述数据处理装置具有处理器单元，所述数据处理装置实施用于执行如下方法步骤，其中，所述网表从FPGA源代码(20)和至少一个影子寄存器(10)中产生，其中，FPGA源代码(20)定义至少一个功能和至少一个信号(30)，其中，影子寄存器(10)配设给所述至少一个信号(30)，并且影子寄存器设置用于在运行时存储所配设的信号(30)的值，其中，设置有用于在运行时读取所存储的信号值的器件，其中，在FPGA源代码(20)中所定义的功能不通过影子寄存器(10)改变，其中，网表设置用于加载到FPGA上并且由FPGA执行，其中，所述由FPGA源代码(20)所描述的功能由FPGA执行，其特征在于，设置所述影子寄存器(10)与在FPGA源代码(20)中所描述的功能的在功能上的解耦，其中，在执行所述在FPGA源代码(20)中所描述的功能期间，影子寄存器(10)通过解耦来保留在解耦的时间点上所存储的信号值。

17.根据权利要求16所述的数据处理装置，其特征在于，所述FPGA源代码(20)定义多个信号(30)，其中，多个影子寄存器(10)分别配设给一个信号(30)，其中，功能上的解耦设置用于同步地解耦多个影子寄存器(10)。

18.根据权利要求16或17所述的数据处理装置，其特征在于，为了解耦，将影子寄存器(10)的使能信号(60)或影子寄存器(10)的时钟信号(700)中断。

19.根据权利要求16或17所述的数据处理装置，其特征在于，所述FPGA源代码(20)作为图形化模型或作为文本代码存在。

20.带有电子可读取的控制信号的数字存储介质，其中，所述控制信号能够与可编程的按照权利要求16至19之一所述的数据处理装置协同作用，使得按照权利要求1至15之一所述的方法在所述数据处理装置上执行。

用于建立FPGA网表的方法

技术领域

[0001] 本发明涉及一种用于为FPGA建立网表的方法。

背景技术

[0002] 复杂、动态的模型的实时仿真由于受限制的时间上的边界条件自身对现代的计算节点提出高的要求。在自动化的硬件在环仿真(HiL)中，这样的模型特别是在快速控制环必须闭合的场合使用。这在气缸内压力传感器的仿真时大致是这种情况，气缸内压力传感器在消耗或废气减少中起着越来越大的作用。但即使在具有高的动态性的受控系统中、例如在电动机中短周期时间和低时延也是绝对必要的。这几点利用基于CPU的仿真实际上几乎不再能实现。

[0003] 现场可编程门阵列(FPGA)可以在实时仿真中通过其承担模型的动态部分的计算来支持计算节点。由于对信号的并行处理的高灵活性和可能性通过使用FPGA也能容易地满足硬实时要求。FPGA可以作为用于计算节点的CPUs的硬件加速器而使用。相应地，例如将环境模型的非常动态的部分转移到FPGA中，从而保证控制器的足够精确且快速的反应时间。FPGA网表通常基于以硬件描述语言的FPGA模型在建立过程中产生。

[0004] 受控系统的模型由于对精确度不断增加的要求变得越来越复杂并且因此也变得难以管理。在自动化的HIL环境中这样的模型通常利用Math Works公司的Matlab/Simulink工具箱来建立。Simulink以框图的形式给这样的模型提供基于模块的视图。各模型部分可以在框图中组合为子系统并且各模型部分可以利用信号互相连接起来。在此，在这些模块之间的数据流通过信号线来表示。

[0005] 基于FPGA的仿真可以在借助dSPACE公司的Xilinx系统生成器(XSG)和FPGA编程模块库(PGA-Programming Blocksets)的情况下类似于基于CPU的仿真，在框图中利用Simulink来建模。

[0006] 然而，相比于CPU仿真，该模型不被翻译为一种迭代式编程语言，而是被翻译为一种FPGA网表，FPGA网表描述一种用户特定的数字电路。FPGA网表可以被翻译为一种FPGA配置数据流。

[0007] 从专利文献EP2765528A1中已知一种用于在运行时从FPGA中读取变量的方法。

[0008] 在一些读取技术中直接从工作寄存器中读取数据。这点在FPGA运行时进行读取时是不利的，因为通常不能在特殊时间点上读取寄存器。因为工作寄存器在运行时持续地以当前值覆盖，所以通常不能检测到在特定时间点上所存在的值。

发明内容

[0009] 本发明的任务是进一步扩展现有技术。

[0010] 本发明涉及一种用于建立FPGA网表的方法，其中，网表从FPGA源代码和至少一个影子寄存器中产生，其中，FPGA源代码定义至少一个功能和至少一个信号，其中，将影子寄存器配设给所述至少一个信号并且将影子寄存器设立和设置用于在运行时存储所配设的

信号的值，其中，设置并设立用于在运行时读取所存储的信号值的器件，其中，在FPGA源代码中所定义的功能不通过影子寄存器改变，其中，将网表设置用于加载到FPGA上并且由FPGA执行，其中，所述由FPGA源代码所描述的功能由FPGA执行，其中，设置和设立所述影子寄存器与在FPGA源代码中所描述的功能的功能上的解耦，其中，在执行所述在FPGA源代码中所描述的功能期间，影子寄存器通过解耦来保留在解耦的时间点上所存储的信号值。

[0011] 按照本发明，还给出一种数据处理装置，其中，所述数据处理装置具有处理器单元，其中，所述数据处理装置实施用于执行上述方法。按照本发明，用于建立FPGA网表的数据处理装置，其中，所述数据处理装置具有处理器单元，所述数据处理装置实施用于执行如下方法步骤，其中，网表从FPGA源代码和至少一个影子寄存器中产生，其中，FPGA源代码定义至少一个功能和至少一个信号，其中，影子寄存器配设给所述至少一个信号，并且影子寄存器设立和设置用于在运行时存储所配设的信号的值，其中，设置并设立有用于在运行时读取所存储的信号值的器件，其中，在FPGA源代码中所定义的功能不通过影子寄存器改变，其中，网表设置用于加载到FPGA上并且由FPGA执行，其中，所述由FPGA源代码所描述的功能由FPGA执行，其特征在于，设置和设立所述影子寄存器与在FPGA源代码中所描述的功能的在功能上的解耦，其中，在执行所述在FPGA源代码中所描述的功能执行期间，影子寄存器通过解耦来保留在解耦的时间点上所存储的信号值。

[0012] 按照本发明，也给出一种带有计算机执行指令的计算机程序产品，该计算机程序产品在合适的数据处理装置中加载和执行之后执行以上方法的步骤。

[0013] 此外，按照本发明，给出一种带有电子可读取的控制信号的数字存储介质，这些控制信号可以这样与可编程的数据处理装置协同作用，使得上述方法在所述数据处理装置上执行。

[0014] 网表可以转换为比特流并且可以在FPGA上加载。于是，FPGA可以执行所述在FPGA源代码中定义的功能。通过解耦能精确地读取在解耦时间点上的当前值，即使对信号值的读取延迟地进行也进行所述读取。这里，在运行时应这样理解，即，所述在FPGA源代码中定义的功能继续执行并且信号值继续更新，在此期间将影子寄存器解耦并且保留所述在解耦的时间点上的当前值。解耦优选通过触发信号来触发。为此，在FPGA网表中解耦机构与触发信号连接。备选地，影子寄存器可以按照标准地解耦并且根据触发信号短时间地与该功能连接。于是，在影子寄存器中存储所述在触发信号的时间点上的当前信号值并且影子寄存器再次与该功能解耦。

[0015] 因此，按照本发明的方法的结果是一个FPGA网表，其中，在网表中定义有至少一个影子寄存器，其中，在FPGA网表中定义有至少一个功能和至少一个信号，其中，影子寄存器与所述至少一个信号相连接并且设立和设置用于在运行时存储信号的值，其中，网表设置用于在FPGA上加载并且由FPGA执行，其中，在网表中设置和设立有一种可能性以用于在FPGA运行时读取在影子寄存器中所存储的信号值，其中，影子寄存器的在功能上的解耦由信号来设置和设立，其中，在FPGA执行功能期间，影子寄存器通过解耦来保留在解耦的时间点上所存储的信号值。

[0016] 在一种优选的设计方案中，FPGA源代码定义多个信号，其中，将多个影子寄存器分别配设给一个信号，其中，在功能上的解耦设置用于同步地解耦所述多个影子寄存器。

[0017] 通过同步的解耦能在运行时从FPGA中读取多个同时存在的信号值。从FPGA中同步

地读取多个信号值通常是不可能的。通过同步可解耦的影子寄存器可以存储和相继读取同时存在的信号值。因此可以把同时存在的变量的一致数据记录下来。

[0018] 优选的设计方案的结果是一种FPGA网表，其中，在FPGA网表中定义有多个信号，其中，多个影子寄存器分别配设给一个信号，其中，在功能上的解耦设置用于同步地解耦所述多个影子寄存器。

[0019] 在一种设计方案中，为了解耦，将影子寄存器的使能信号或影子寄存器的时钟信号中断。

[0020] 通过影子寄存器的使能信号的中断，影子寄存器在运行时不再更新。这构成一种用于将一个或多个影子寄存器解耦的一种非常简单的可能性。

[0021] 通过影子寄存器的时钟信号的中断，影子寄存器不再接收变化量，从而寄存器的状态近似冻结。尽管如此，所述在影子寄存器中所存储的值可以按不同的方式来读取。

[0022] 优选的设计方案的结果是一种FPGA网表，其中，在网表中规定，为了解耦，将影子寄存器的使能信号或影子寄存器的时钟信号中断

[0023] 在一种设计方案中，所述FPGA源代码作为图形化模型或作为文本代码存在。

[0024] FPGA源代码通常以图形化模型的形式存在，例如作为在开发环境中的框图存在。这样的框图的例子是Math Works公司的Simulink。在图形化的源代码中可以将影子寄存器简单地作为附加的寄存器加入，其中，附加的寄存器与所配设的信号连接并且可以通过同样加入的解耦机构在运行时在功能上与信号解耦。FPGA源代码的一种备选方式是一种文本代码、例如VHDL或Verilog。在文本源代码中也可以将影子寄存器简单地作为附加的寄存器加入，其中，附加的寄存器与所配设的信号连接并且可以通过同样加入的解耦机构在运行时在功能上与信号解耦。

[0025] 在一种优选的设计方案中，将影子寄存器加入到FPGA源代码或FPGA源代码的副本中。

[0026] 通过将影子寄存器直接加入到FPGA源代码中，网表的建立是特别简单的，因为仅需将FPGA源代码翻译为网表。通过将影子寄存器加入到FPGA源代码的副本中保持原始的FPGA源代码未改动。如果该方法对于用户应透明，那么这点是有利的。因此，用户不参与将影子寄存器加入，尽管如此可以利用在运行时读取信号值的优点。

[0027] 在一种备选的设计方案中，从源代码中生成网表并且将影子寄存器加入到网表中。

[0028] 通过将影子寄存器加入到网表中保持FPGA源代码不改变，而不必创建副本。于是，具有影子寄存器的网表可以如每个其他的网表那样被继续使用。例如可以将网表转换为比特流并且紧接着加载到FPGA上。在此，网表(影子寄存器加入到网表中)可以作为简单的网表或者具有关于映射、布局和/或布线的附加信息而存在。

[0029] 在一种优选的设计方案中，将影子寄存器自动化地加入并且配设给所述信号。

[0030] 通过自动化，用户减轻了如下任务的负担，即加入影子寄存器和将其配设给信号。特别是在多个信号的情况下(所述多个信号应分别配设有一个影子寄存器)自动化是具有很大优点的。

[0031] 在另一设计方案中，自动化地检验，所述信号是否已经在其他位置上配设给一个影子寄存器，并且如果是，就不给该信号配设另外的影子寄存器。

[0032] 通过自动化的检验来防止对于一个信号加入多个影子寄存器并且将它们配设给该信号。特别是在FPGA源代码很大时、即包括许多行文本或作为图形化模型包括多个层级，人工检验是耗费的并且易出错误的并且因此自动化检验是有利的。

[0033] 在另一设计方案中，加入至少两个影子寄存器并且将它们配设给所述信号，其中，第一影子寄存器设置和设立用于在运行时在第二影子寄存器解耦期间存储当前信号值。

[0034] 利用两个影子寄存器能将旧的信号值记录在第一影子寄存器中并且同时将当前的信号值记录在第二影子寄存器中。这例如可以当影子寄存器的读取过程可以持续多个FPGA时钟或不立即实施时使用。于是，在读取第一影子寄存器期间可以将当前值存储在第二影子寄存器中。

[0035] 多个影子寄存器也可以构成一个存储窗口。在该情况下，在一个影子寄存器中存储当前的信号值并且在另一影子寄存器中存储旧的信号值。根据触发信号可以同时解耦各影子寄存器。由此能在触发信号之前读取信号值。

[0036] 在另一备选的实施方式中，可以使用存储窗口，以便记录紧跟触发信号的信号值。在该情况下，影子寄存器相继地在预先确定的持续时间之后解耦。

[0037] 通过三个或更多个影子寄存器可以同时使用两种变型方案。因此能在触发信号之前和之后记录并读取信号值。

[0038] 优选的设计方案的结果是一种FPGA网表，其中在FPGA网表中定义有至少两个影子寄存器并且将所述影子寄存器配设给一个信号，其中，第一影子寄存器设置和设立用于在运行时在第二影子寄存器解耦期间存储当前信号值。

[0039] 在一种特别优选的设计方案中，在建立网表时，设置和设立通过FPGA的外部和/或内部的回读接口对影子寄存器的读取。

[0040] 通过内部或外部的回读接口对影子寄存器的读取构成读取的一种轻松的可能性。在该实施方式中，特别是需要非常少的FPGA的逻辑电路和布线资源来用于读取。影子寄存器可以通过执行工具直接布置在所配设的信号附近。由此相对于没有加入的影子寄存器的网表而言变化是最小的。

[0041] 在另一设计方案中，加入多个影子寄存器，其中，将所述多个影子寄存器连接成一个移位寄存器链并且所述多个影子寄存器设置和设立用于通过FPGA的外部接口来读取。

[0042] 在一种备选的设计方案中，加入多个影子寄存器，其中，将地址译码器设置和设立用于通过FPGA的外部接口来读取所述多个影子寄存器。

[0043] 相比于回读接口，通过外部的接口的读取典型地能实现更高的数据传输率。因此，当在运行时应频繁地读取许多信号值时，这种读取类型是有利的。

[0044] 在另一设计方案中，附加于影子寄存器加入逻辑电路，其中，逻辑电路设置和设立用于在运行时在信号值变化时输出触发信号，其中，该触发信号引起影子寄存器的解耦。

[0045] 通过安装的逻辑电路可以在出现预先确定的事件时非常快地触发解耦。于是，触发信号可以立即或以预先确定的延迟来触发一个或多个影子寄存器的解耦。一种这样的逻辑电路可以按不同的类型来实施。对此的例子在各实施例中可找到。

[0046] 在一种优选的设计方案中，在加入影子寄存器之前实施以下步骤：

[0047] -确定在FPGA源代码中的所有如下常数，第一信号值与这些常数有关；

[0048] -为这些常数的找到的值确定最小必需的位宽；

[0049] -将这些常数重新配置到相应确定的最小所需的位宽上或者将这些常数事后构成到相应确定的最小所需的位宽上；

[0050] -通过整个FPGA模型传播所述位宽。

[0051] 在图形化FPGA模型中的常数模块或在VHDL模型中的VHDL信号通常以固定位宽(例如32位)实例化。然而，在紧接着确定各值时通常使用没有用尽给定的位宽的值域的值。所述从图形化模型中生成的VHDL代码同样包含整个位宽，而与实际所需的位宽无关。在手动编程VHDL代码时技术人员一般同样地处理。

[0052] 这通常没有问题，因为不需要的位通过综合和执行工具进行路径优化，以便节约逻辑和布线资源。

[0053] 然而，所加入的设立用于通过外部的接口来读取的影子寄存器没有被综合和执行工具优化，因为这些工具没有优化外部通道。为了通过加入的影子寄存器和读取逻辑电路保持资源消耗量为低的，因此有利的是确定各常数的最小所需的位宽以及所有与这些常数有关的信号。于是，所述影子寄存器能以配设的信号的最小所需的位宽来执行。

[0054] 在另一设计方案中，所述影子寄存器在网表的建立和/或处理时被保护免于路径优化。

[0055] 不具有输出端的寄存器通常通过用于建立和处理网表的自动化工具来路径优化、即删除。这通常是有意义的，因为否则这些寄存器占用FPGA中的资源而没有为FPGA的功能作贡献。所述在按照本发明的方法中加入的影子寄存器(它们应通过回读接口来读取)不具有输出端并且因此被工具删除。然而，需要这些影子寄存器，以便在运行时能从FPGA中读取信号值。因此有利的是，这些影子寄存器被保护免于路径优化。一种能实现这点的可能性是，将各影子寄存器和在其中包含的各信号给予属性，其中，这些属性由各工具识别出并且这些工具引起不采取对寄存器的优化措施。

[0056] 所述被保护免于路径优化的影子寄存器与上述用于确定最小所需的位宽的步骤的组合是特别有利的。

附图说明

[0057] 以下参考附图进一步阐述本发明。在此，相同类型的部件标明相同的附图标记。所示出的实施方式是极其示意性的，也就是说，距离以及横向和垂直的延伸尺寸不是按比例的并且只要没有另外说明也就互相不具有可推导的几何关系。

[0058] 其中：

[0059] 图1示出按照本发明的第一实施方式的示意图；

[0060] 图2示出按照本发明的第二实施方式的示意图；

[0061] 图3示出按照本发明的第三实施方式的示意图；

[0062] 图4示出按照本发明的第四实施方式的示意图；

[0063] 图5示出第一、第二和第三实施方式的行为的示意图；

[0064] 图6示出网表的示意图，连同用于通过回读接口读取寄存器的准备；

[0065] 图7示出利用时钟线的中断的示意性解耦电路；

[0066] 图8示出利用使能线的中断的示意性解耦电路；

[0067] 图9示出按照本发明的方法的步骤。

具体实施方式

[0068] 在图1至4中，借助示意性的图形化的FPGA源代码来阐明所述方法。不言而喻地，方法可以类似地以文本FPGA源代码地实施。从FPGA源代码中相应可以产生一个能加载到FPGA上的网表，由此FPGA可以执行在FPGA源代码中所定义的功能。

[0069] 图1的图示示出第一实施方式的视图，包括：在自由运行(Freilauf)模式中的影子寄存器10。在FPGA源代码20中定义了具有信号30的功能。信号30配设给影子寄存器10。影子寄存器10具有两个输入端40、50。第一输入端40用于信号30以及第二输入端50用于使能信号60。利用使能信号60可以激活影子寄存器10。影子寄存器10仅在激活状态下接收所配设的信号30的当前信号值。使能信号60可以持续地施加并且仅在期望的时间点上中断，以便持续地在影子寄存器中存储当前值。备选地，可以仅短时间地施加使能信号60，以便持续地在影子寄存器中存储信号30在激活的使能信号的时间点上的当前信号值。在该例子中，所述在方法期间所加入的部分70包括影子寄存器10以及用于信号30和使能信号60的线路。使能信号60优选从外部施加到FPGA上。不过也能让使能信号60由FPGA来控制。

[0070] 在图2的图示中示出第二实施方式。以下仅阐述相对于图1的图示的区别。中间寄存器100、也可以称为第二影子寄存器与信号30和影子寄存器10相连接。在影子寄存器解耦期间，在中间寄存器100中可以存储当前信号值。两件式的逻辑电路110、140与信号30和中间寄存器100连接。逻辑电路110的第一部分在运行时识别出已经发生信号值的变化。逻辑电路140的第二部分在通过逻辑电路110的第一部分识别出的信号值变化的情况下产生一个触发信号130，该触发信号用作用于第二寄存器的使能信号。逻辑电路110、140这样构建，使得在信号值变化时仅在一个时钟上产生触发信号130。由此在中间寄存器100中仅存储变化的第一信号值。使能信号60能实现在中间寄存器100中所存储的信号值接纳到影子寄存器10中。于是，从影子寄存器100中可以——例如通过回读接口——读取值。使能信号60也用作用于逻辑电路110、140和中间寄存器100的复位信号。逻辑电路110、140这样设计，使得它们在接收到复位信号之后才产生新的触发信号130。中间寄存器100的复位能实现，在读取到的数据中清楚地区别出，是读取了当前值还仅是在复位后的寄存器的初始值。

[0071] 不言而喻地，触发信号130也可以转发到其他加入的电路上。因此，触发信号可以触发对多个信号值的存储。如果应检测同时存在于FPGA中的信号值的一致记录，那么这点是有利的

[0072] 在该例子中，所述在方法期间所加入的部分70包括影子寄存器10、中间寄存器100和两件式的逻辑电路110、140。

[0073] 在图3的图示中示出第三实施方式。以下仅阐述相对于图2的图示的区别。使能信号60在这里不用作用于第二寄存器100和两件式的逻辑电路110、140的复位信号。使能信号60仅用于将信号值接纳到影子寄存器10中。用于逻辑电路110、140和中间寄存器100的复位信号70与使能信号60相分开并且可以根据在FPGA之外或之内的执行来驱控。

[0074] 在图4的图示中示出第四实施方式。以下仅阐述相对于图3的图示的区别。加入了模式切换逻辑电路200。模式切换逻辑电路200包含作为输入的复位信号70、使能信号60和模式信号210。根据模式信号210的值可以控制两件式的逻辑电路110、140，从而第四实施方式的电路如图1中的第一实施方式的电路那样地行为。附加地根据模式信号210，使能信号60用作用于中间寄存器100和两件式的逻辑电路110、140的复位信号，从而电路如图2中的

第二实施方式的电路那样行为。如果没有施加模式信号210，那么电路如图3中的第三实施方式的电路那样行为。第四实施方式即是前三个实施方式的组合，可以通过模式信号210在运行时选择行为。如果在建立网表时还没有确定期望哪种行为，那么这点是有利的。

[0075] 在图5中的图示示出第一、第二和第三实施方式的行为的示意图。

[0076] 最上面一行500示出信号30的一种示例性的信号曲线。信号值随着时间多次地在0和1之间更换。

[0077] 第二行510示出以图1中的第一实施例为例的影子寄存器10的值。使能信号60在四个不同的时间550、560、570、580上分别短暂地激活。因为影子寄存器的值仅在使能信号60激活时更新，所以影子寄存器10的值一直保持为0，直到信号值30为1且同时使能信号激活为止。于是，可以从影子寄存器中读取该值。

[0078] 第三行520示出以图3中的第三实施例为例的中间寄存器100的值。在信号30的信号值的第一个变化时，两件式的逻辑电路110、140对触发信号130进行触发并且当前值被接纳到中间寄存器100中。复位信号70在第五时间点590激活。复位信号70使中间寄存器100的值置为0并且使两件式的逻辑电路110、140重置，从而在信号30的信号值的下一个变化时又产生一个触发信号130。每次当使能信号60在四个所示出的时间点550、560、570、580之一上激活时，所述中间寄存器100在该时间上的当前值被接纳到影子寄存器10中并且可以从那里读取。

[0079] 第四行530示出以图2中的第二实施例为例的中间寄存器100的值。在信号30的信号值的第一个变化时，两件式的逻辑电路110、140对触发信号130进行触发并且当前值被接纳到中间寄存器100中。每次当使能信号60在四个所示出的时间点550、560、570、580之一上激活时，所述中间寄存器100在该时间上的当前值被接纳到影子寄存器10中并且可以从那里读取。同时使能信号60使中间寄存器100的值置为0并且使两件式的逻辑电路110、140重置，从而在信号30的信号值的下一个变化时又产生一个触发信号130。

[0080] 图6示出具有网表的示意图连同通过回读接口读取寄存器的准备。网表由三个逻辑模块MUX、ADD、MULT和多条线路组成。网表的两个输入端610、620引导到第一逻辑模块MUX的两个输入端中。第一逻辑模块MUX的输出端与第二逻辑模块ADD的第一输入端连接。网表的第三输入端630与第二逻辑模块ADD的第二输入端连接。第二逻辑模块ADD的输出端与第三逻辑模块MULT的第一输入端连接。网表的第四输入端640与第三逻辑模块MULT的第二输入端连接。第三逻辑模块的输出端构成网表的输出端。三个影子寄存器REG加入到该网表中。每个影子寄存器REG分别配设给一个逻辑模块的一个输出端信号。影子寄存器REG设置并设立用于通过回读接口读取，因此在网表中没有加入读取逻辑。影子寄存器可以通过解耦电路在相同的时钟步长上与逻辑模块在功能上分离。由此可以在各影子寄存器中保持一致的数据记录。

[0081] 图7示出利用时钟信号中断的示意性解耦电路。各影子寄存器的时钟信号700通过加入的解耦机构710与配设给影子寄存器的信号的时钟网CLK连接。解耦机构可以在运行时从FPGA之内和/或之外起作用并且中断时钟信号。所述FPGA源代码20中的配设给影子寄存器10的信号30固定地与影子寄存器10的信号输入端40连接。影子寄存器10仅在解耦机构700将影子寄存器10的时钟信号与时钟网CLK连接时才接收当前信号值。

[0082] 图8示出利用使能信号中断的示意性解耦电路。在该实施方式中，影子寄存器10的

时钟信号700直接与时钟网CLK连接。为了解耦影子寄存器10加入解耦机构并且将其与影子寄存器10的使能输入端50连接。解耦机构可以在运行时从FPGA之内和/或之外起作用并且中断使能信号。

[0083] 具有多个时钟域的FPGA程序也可以按上述方式可靠地、即无亚稳定性地运行。对此，在利用时钟信号中断的解耦时，影子寄存器连接到时钟网上，所配设的信号也利用该时钟网来运行。在配设给不同的信号的多个影子寄存器中，各影子寄存器的时钟线路连接到所配设的信号的相应的时钟网上。在利用使能线的中断的解耦中，对于每个时钟域在FPGA网表中加入一个带有正确的时钟域通道的自身的使能信号并且连接到相应的时钟域的影子寄存器的使能端口上。时钟域通道例如可以由双寄存器正确地越过(überschreiten)。

[0084] 图9示出按照本发明的方法的步骤。在此，一些步骤是强制的，另外的步骤是可选的。在可选的第一步骤S110中，创建源代码的副本。在可选的第二步骤S120中，确定源代码或源代码的副本中常数的最小所需的位宽。在可选的第三步骤S130中，通过源代码来传播所述在第二步骤S120中所确定的最小所需的位宽。在强制的第四步骤S140中，确定应配设有影子寄存器10的信号30。该步骤可以自动化地通过算法或通过用户输入发生。在可选的第五步骤S150中，检验所述在第四步骤S140中确定的信号的值是否可以从已经加入的各影子寄存器的值中确定。在强制的第六步骤S160中，为所述在第四步骤S140中所确定的信号30加入一个影子寄存器10。附加地，为该影子寄存器10加入和设立一个解耦机构，或者该影子寄存器10与一个已经加入的解耦机构连接。如果在第五步骤S150中确定，所述在第四步骤S140中确定的信号30的值可以从已经加入的各影子寄存器的值中确定，则在第六步骤S160中不加入影子寄存器，而是关联已经加入的影子寄存器，从这些影子寄存器中可以确定信号的值。在强制的第七步骤S170中，为所述在第六步骤S160中加入的影子寄存器10加入并设立一个读取机构。读取机构的加入和设立例如可以在于，对执行工具的指令加入到源代码中，即在FPGA上(网表应在FPGA上加载)应能实现通过回读接口读取。第四至第七步骤可以重复多次，以便给多个信号配设影子寄存器。优选地，第四至第七步骤自动化地通过算法重复地执行，其中，算法系统地执行所有在源代码中定义的信号。

[0085] 在强制的第八步骤S180中，从源代码中综合出一个网表。应注意，第八步骤S180也可以在第四步骤S140之前实施。在任何情况下，网表不仅包括原始的源代码的信息而且包括加入的具有设立的解耦机构和设立的读取机构的影子寄存器。在可选的第九步骤S190中，所述在第八步骤S180中产生的网表转换为比特流，于是该比特流可以在可选的第十步骤S200中加载到FPGA上并且可以在那里执行。于是，在FPGA运行时在可选的第11步骤S210中，所述在第六步骤S160中所加入的影子寄存器10可以通过解耦机构与原始的源代码的功能解耦并且通过所述在第七步骤S170中所设立的读取机构来读取。方法的各强制的步骤能实现在FPGA运行时可靠地读取在第四步骤S140中所确定的信号30的值。

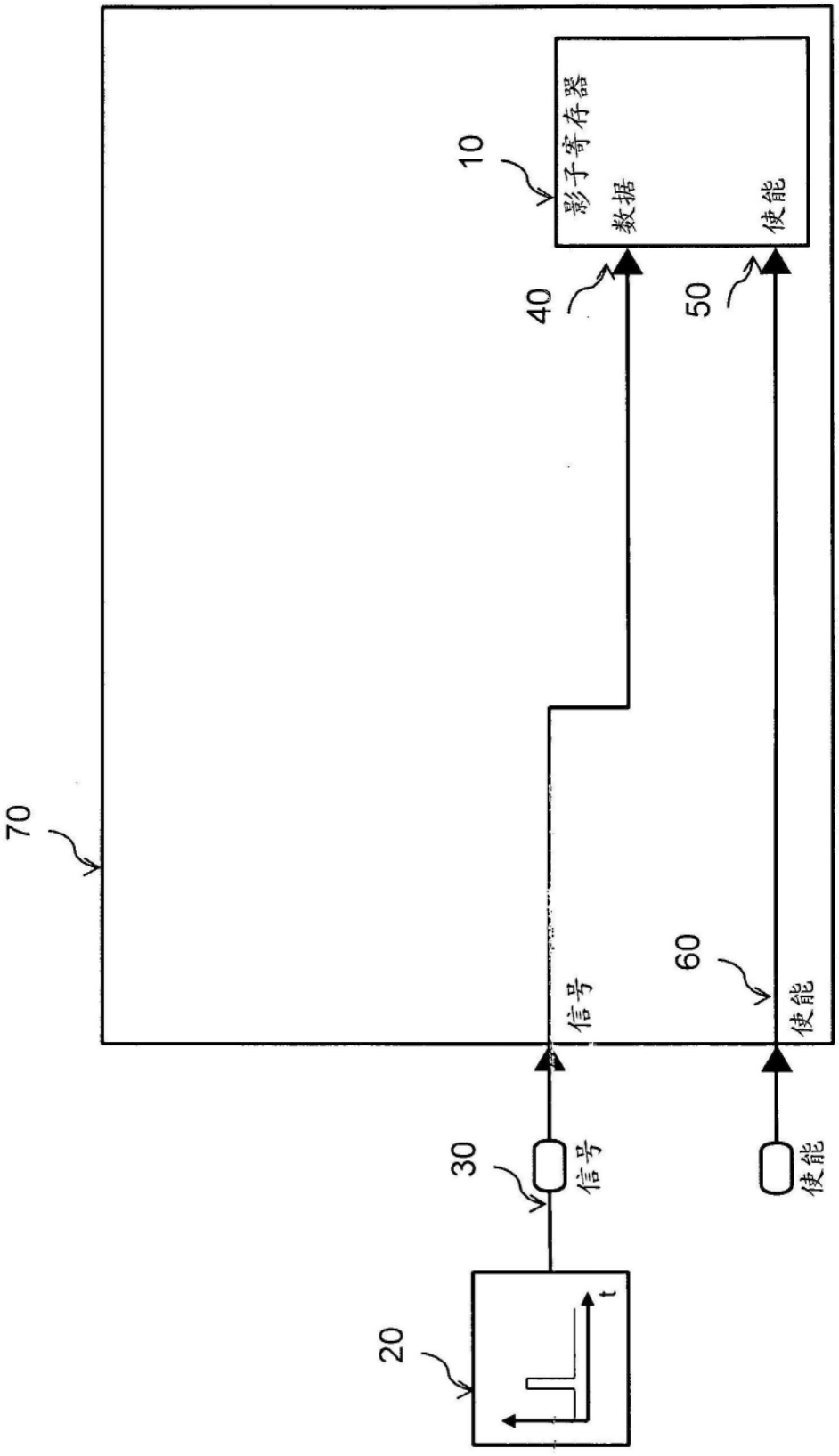


图1

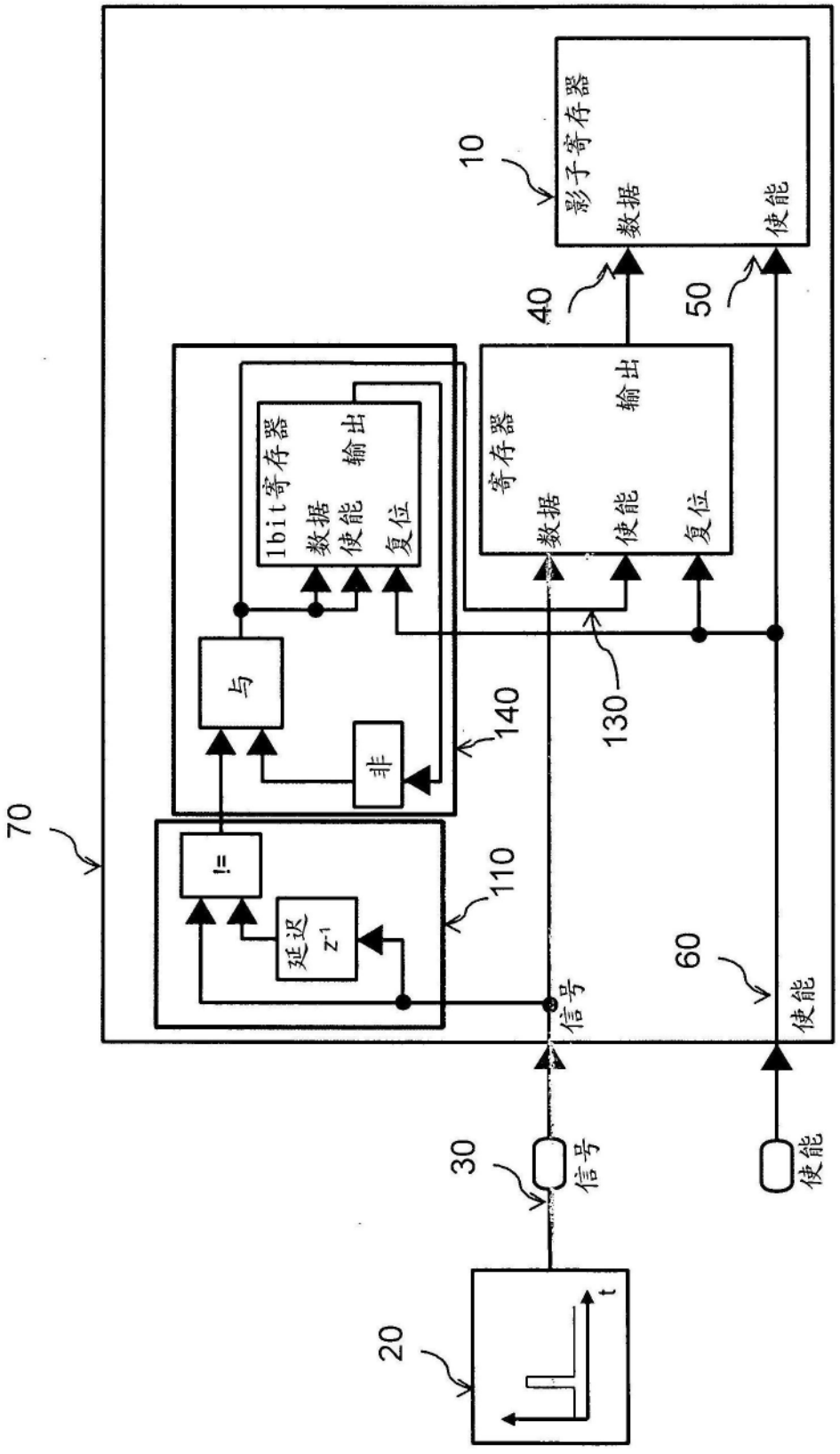

10
影子寄存器
数据
使能
40
50
1bit寄存器
数据
使能
复位
输出
寄存器
数据
使能
复位
输出
与
非
140
130
70
!=
延迟
z^{-1}
110
信号
60
使能
30
信号
使能
20
t

图2

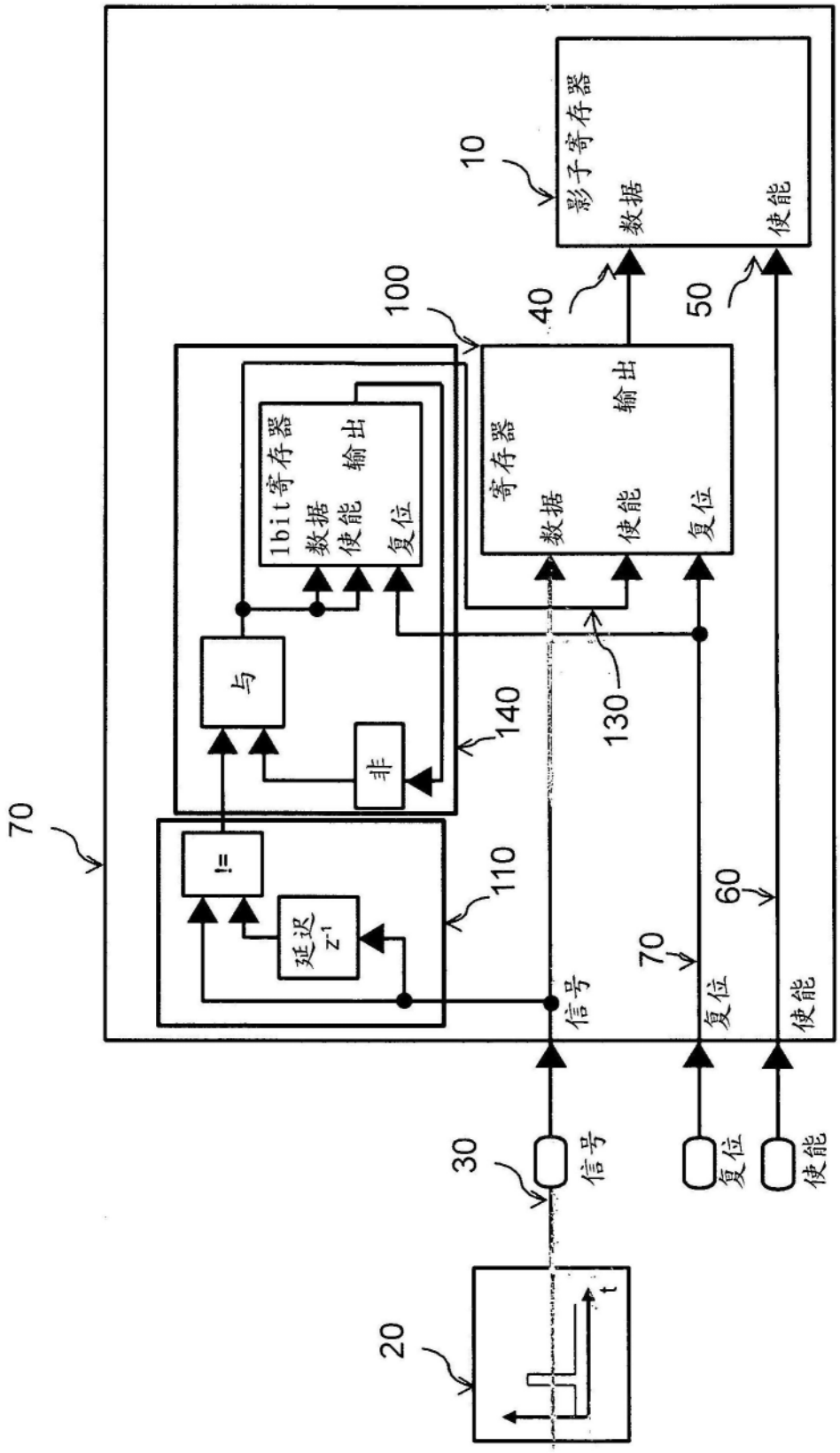

70
10
影子寄存器
数据
使能
100
40
50
寄存器
数据
使能
复位
输出
1bit寄存器
数据
使能
复位
输出
与
非
140
130
!=
延迟
z^{-1}
110
60
70
信号
复位
使能
30
信号
复位
使能
20
t

图3

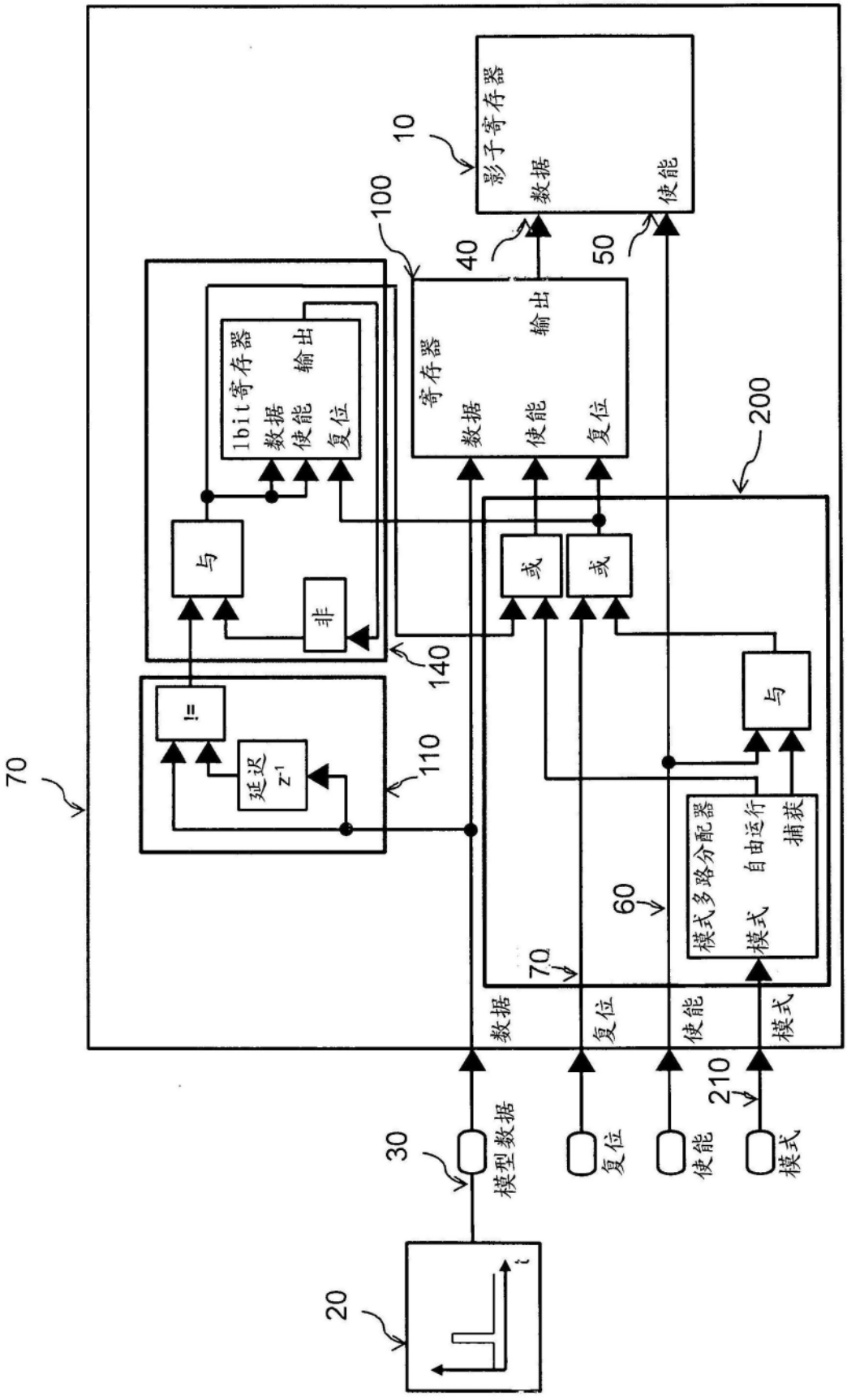

70
10
影子寄存器
数据
使能
100
40
50
寄存器
数据
使能
复位
输出
200
1bit寄存器
数据
使能
复位
输出
与
非
或
或
与
!=
延迟
z^{-1}
110
140
模式多路分配器
模式
自由运行
捕获
60
70
数据
复位
使能
模式
210
30
模型数据
复位
使能
模式
20
t

图4

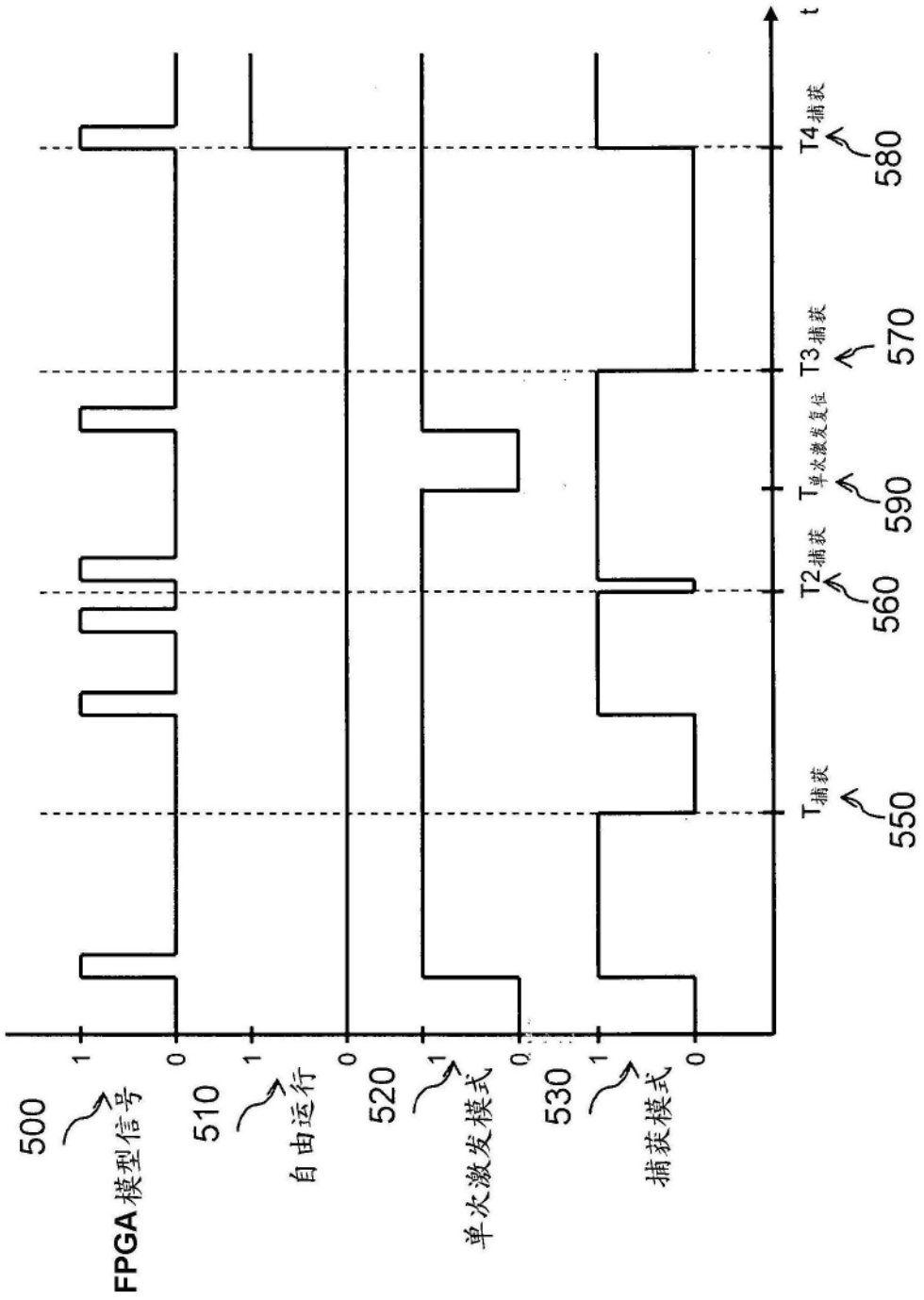

500
FPGA模型信号
510
自由运行
520
单次激发模式
530
捕获模式
1
0
t
T捕获
550
T2捕获
560
T单次激发复位
590
T3捕获
570
T4捕获
580

图5

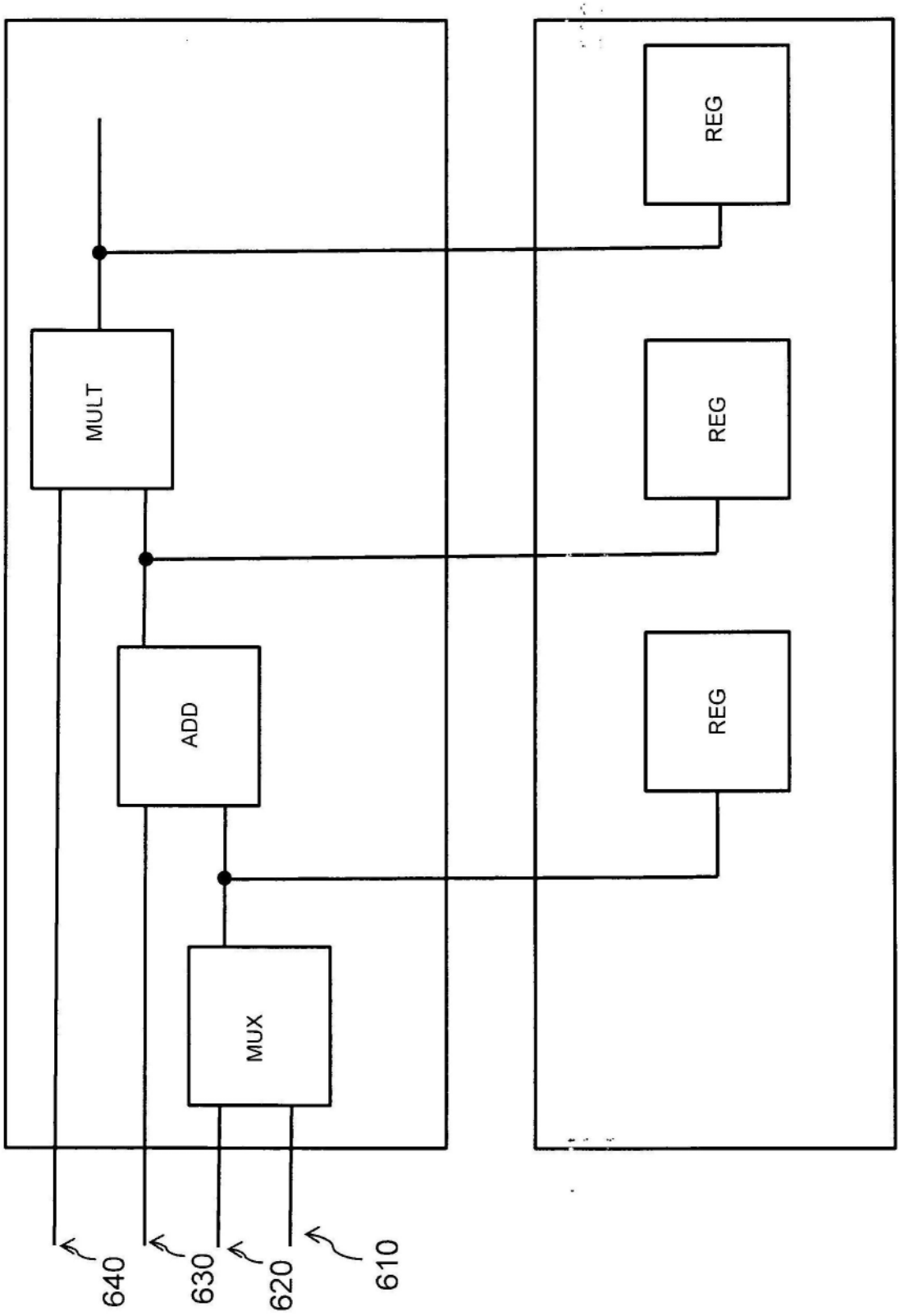


图6

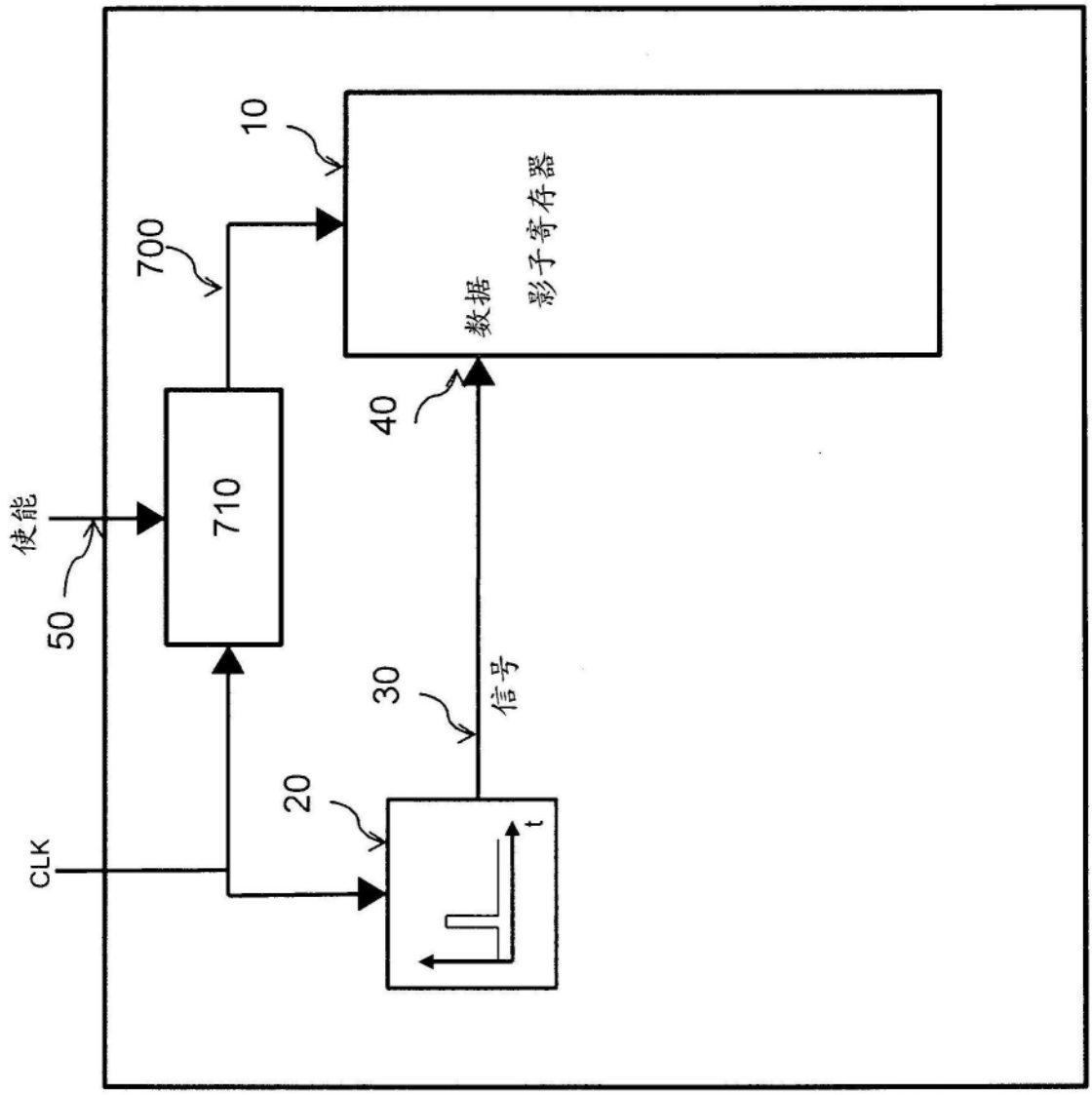


图7

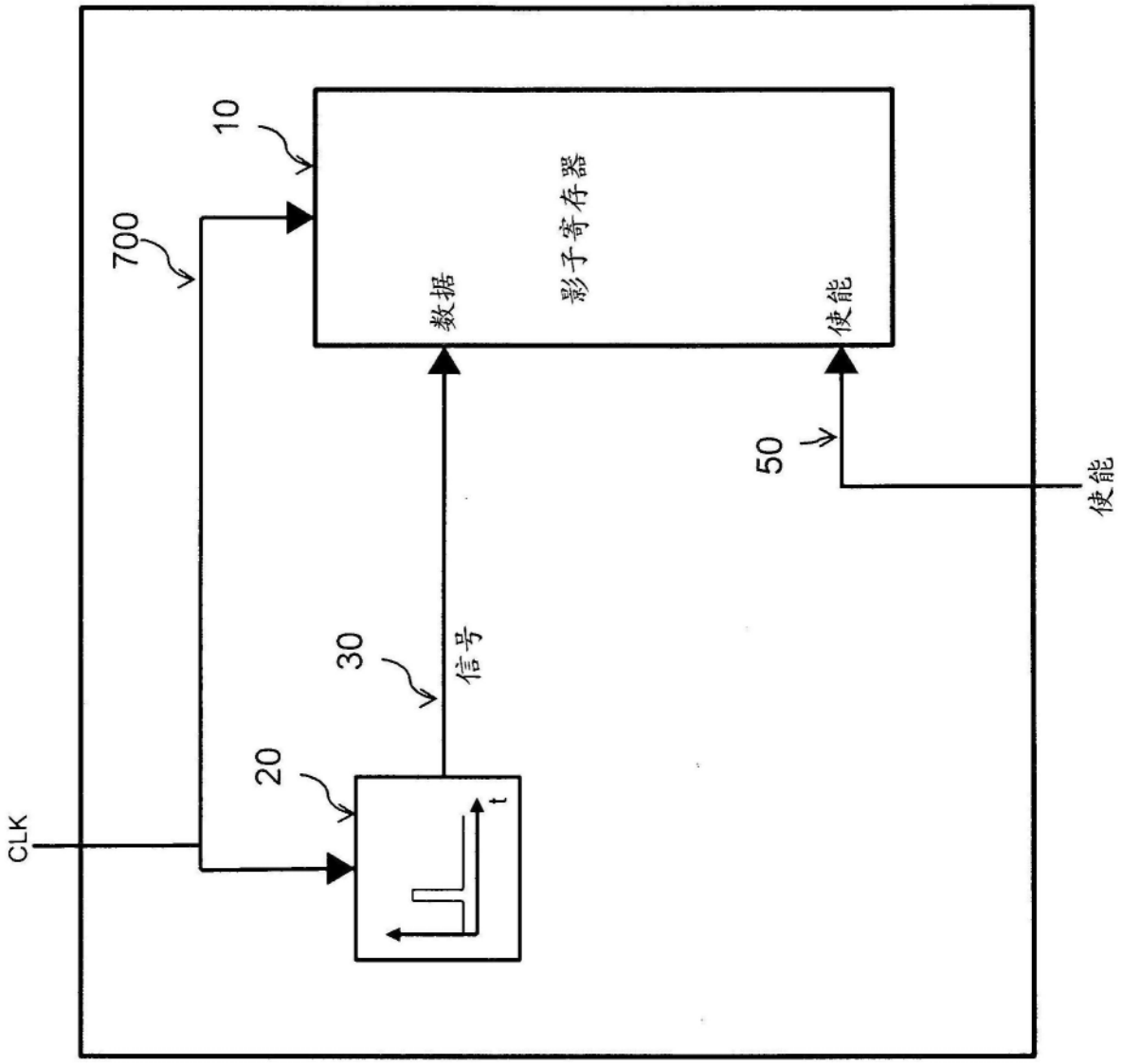


图8

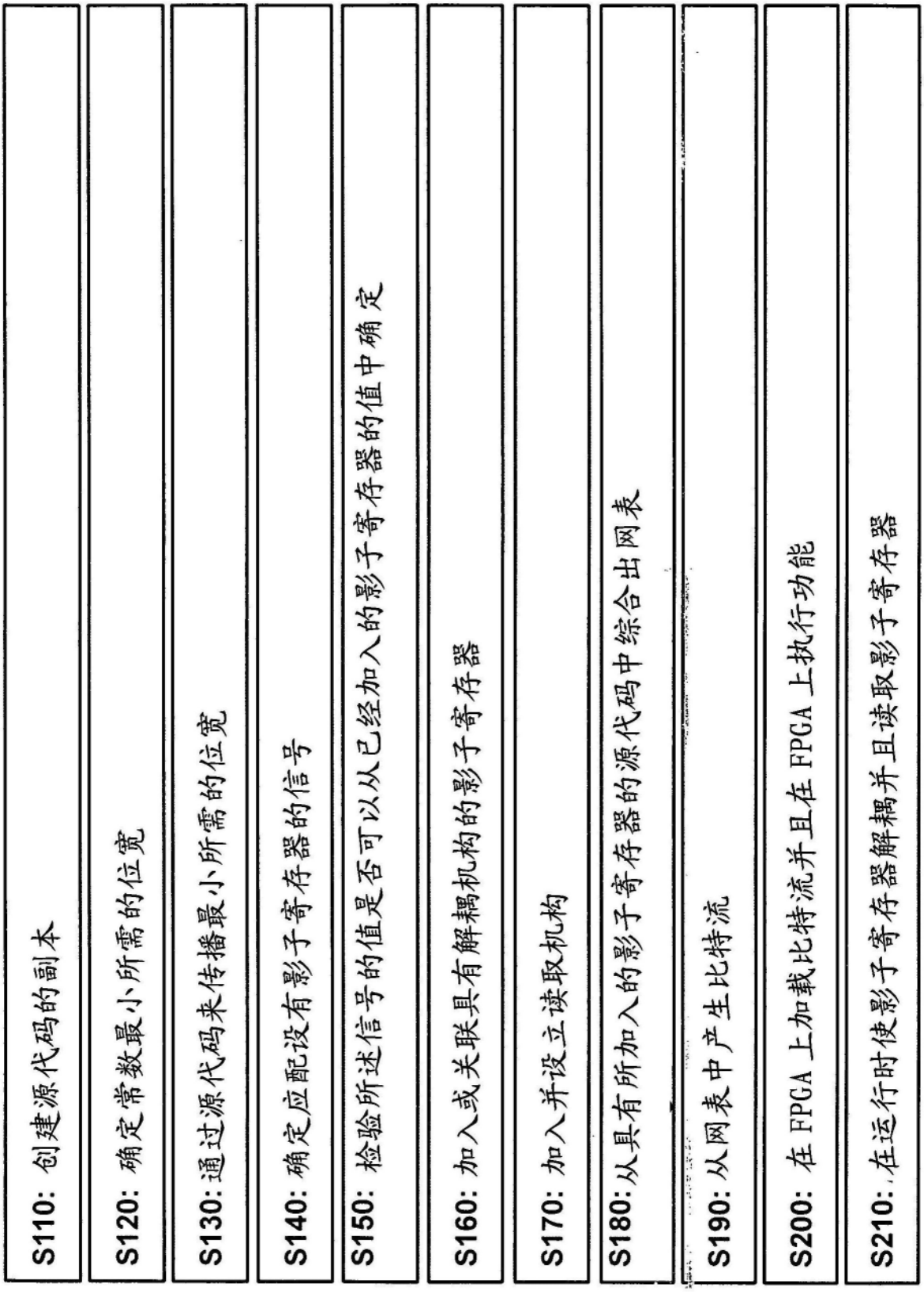

S110: 创建源代码的副本
S120: 确定常数最小所需的位宽
S130: 通过源代码来传播最小所需的位宽
S140: 确定应配设有影子寄存器的信号
S150: 检验所述信号的值是否可以从已经加入的影子寄存器的值中确定
S160: 加入或关联具有解耦机构的影子寄存器
S170: 加入并设立读取机构
S180: 从具有所加入的影子寄存器的源代码中综合出网表
S190: 从网表中产生比特流
S200: 在 FPGA 上加载比特流并且在 FPGA 上执行功能
S210: 在运行时使影子寄存器解耦并且读取影子寄存器

图9