·开发研究与工程应用· 文章编号：1000—3428(2013)05—0305—04 文献标识码：A 中图分类号：TP331

FPGA 位流解析及电路还原方法

吴 强[1]，张逸中[2]

(1. 上海交通大学计算机科学与工程系，上海 200240；2. 华东师范大学软件学院，上海 200062)

摘 要：根据 Xilinx 公司 FPGA 芯片的位流配置关系，提出一套完整的自动化解析方法，包括单配置点与位流配置关系，以及基于分布式架构的海量脚本分析。采用电路逆向还原方式，完成位流文件逆向还原私有电路描述(NCD)网表文件工具 Bit2NCD 开发。以 Spartan2 芯片系列为例进行实验验证，结果表明，采用该电路还原方法所获得的位流配置数据正确且完整。

关键词：现场可编程门阵列；位流解析；电路还原；私有电路描述；可配置点；Bit2NCD 工具

Method of Bitstream Resolving and Circuit Recovering of FPGA

WU Qiang[1], ZHANG Yi-zhong[2]

(1. Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240, China;
2. Software Engineering Institute, East China Normal University, Shanghai 200062, China)

【Abstract】This paper presents a complete and automatic method to resolve the relationship between bitstream and circuit configuration of Xilinx Field Programmable Gate Array(FPGA). Including method for analyzing relationship between single Programmable Point(PP) and bitstream data, and method for analyzing mass scripts based on distributed architecture. Using circuit recovering method, a Bit2NCD tool is developed, which can reverse the bitstream file to Native Circuit Description(NCD) netlist file. A demonstration on Spartan2 chip verifies the circuit recovering method is feasible, and the configuration data is correct and complete.

【Key words】Field Programmable Gate Array(FPGA); bitstream resolving; circuit recovering; Native Circuit Description(NCD); Programmable Point(PP); Bit2NCD tool

DOI: 10.3969/j.issn.1000-3428.2013.05.068

1 概述

可重构计算[1]在各种应用场景中占据了越来越重要的地位。作为可重构计算的支撑技术，现场可编程门阵列(Field Programmable Gate Array, FPGA)的可重构特性将会持久地占据可重构研究的热点位置。尽管 FPGA 已经被广泛地应用于不同的关键场合，FPGA 的可重构特性还局限在模块级或系统级可重构。只要在设计中发生一点变化，对应的 FPGA 配置位流文件就要经过 EDA 工具(如 Xilinx ISE Tools)的综合、转换、映射、布局布线等步骤来重新生成。也正是因为如此，很多关于实时可重构及动态部分可重构[2-5]的研究工作只能停留在理论研究阶段。

如果有一种工具支持位级(bit level)的比特流配置修改，同时不需要引入 EDA 工具，将显著节省可重构时间，同时对 FPGA 进行细微修改变得更加容易。这意味着研究人员将具备更高的 FPGA 可重构自制权。尽管已有一些不错的关于比特流文件与实际 FPGA 编程点配置映射关系研究的工作，如 FAT[6]、Debit[7]和 JBits[8]。但这些工作还有不少不足和局限性，远不能认为是这类问题的完全解决方案。

本文将会解决 FPGA 位流解析领域的 2 大问题：(1)如何获得比特流文件中的二进制位值与实际 FPGA 芯片中可编程点具体配置的映射关系。(2)在获得了映射表的前提下，如果将比特流文件逆向还原成 NCD 文件，对于第(1)个问题，提出分布式映射表生成体系结构，结合心跳包和流水线技术来加速比特流映射关系的破解过程。还建立了合适的数学模型来确保映射表的完整性和正确性。同时，开发了 Bit2NCD 工具来解决第(2)个问题。在当前版本中，对于那些使用某些 FPGA 芯片的电路，其通过 Bit2NCD 工具还原得到的 NCD 文件与原始的 NCD 文件是完全一样的。

本文仅针对未加密的位流文件进行分析，而对于加密位流文件的解密研究，文献[9-10]有相关阐述。

2 比特流分析工具结构及可配置点测定办法

本节首先介绍比特流分析工具，然后介绍比特流分析的相应方法。图 1 给出了分布式 FPGA 比特流分析工具架构图，包括服务器端及计算节点端两部分。

作者简介：吴 强(1986－)，男，硕士，主研方向：高性能计算；张逸中，硕士

收稿日期：2012-06-25 **修回日期：**2012-08-21 **E-mail：**cupidwu@sjtu.edu.cn

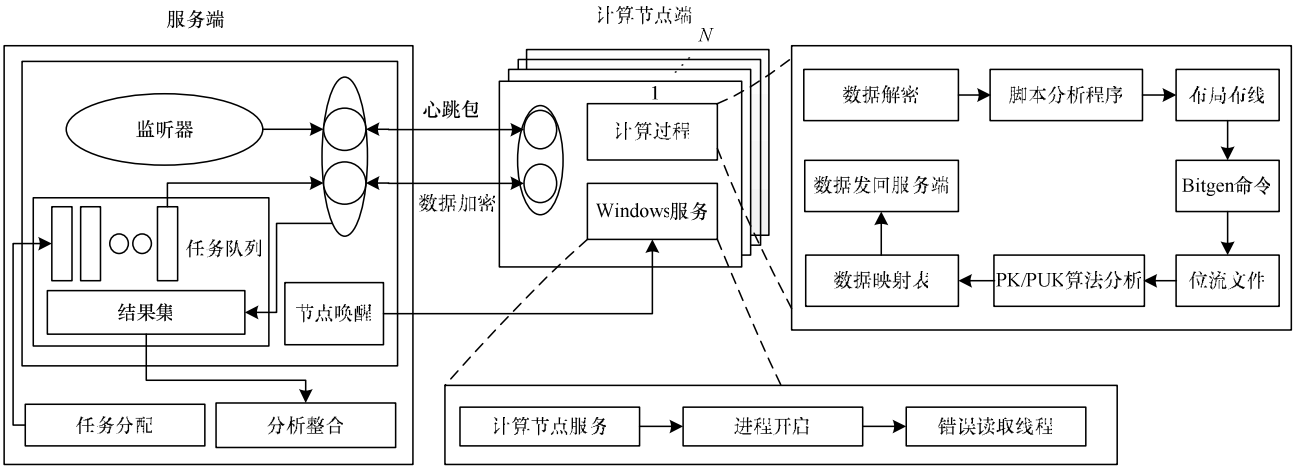


图 1 分布式 FPGA 比特流分析工具架构

2.1 服务器端

服务端和计算节点通过 TCP/IP 协议进行通信。服务端将完成以下工作:

(1)生成并分配分析脚本给计算节点。

(2)追踪计算节点的状态。

(3)分析整合已得到的可编程点映射表。

为处理任务 1,服务端将考虑计算节点间的负载平衡问题。对于任务 2,服务端通过心跳技术来与计算节点保持连接。例如,当计算节点由于断电等原因关机,服务端无法收到心跳包的响应,则会认为计算节点不可用。每个计算节点都被赋予了部分可编程点的映射表分析能力。因此,服务端将对所有计算节点的分析结果进行分析和整合。

为了确保服务端与计算节点数据通信的安全,采用 RSA 加密技术来加密数据包。

2.2 计算节点端

计算节点的任务有:

(1)与服务端保持通信。

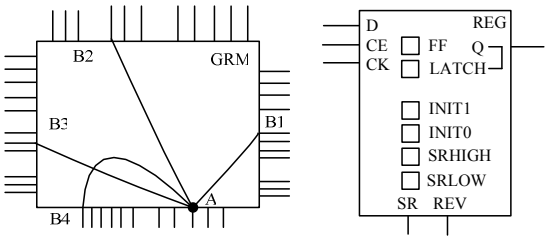
(2)从服务端接收分析脚本,并进行具体的分析过程。

对于任务(1),计算节点将在后台运行系统服务。系统服务将跟踪本节点的运行状态和监测服务端发来的心跳包。当计算节点重新启动后,将自动把自身当前状态发送给服务端,同时在破解网络中注册自己。当计算节点接收到服务端发来的分析脚本后,其将调用 ISE 的相关工具生成对应的比特流文件并进行比较分析。分析结果得到后,将把映射表数据发送回服务端。

2.3 分析脚本

FPGA 内部包含了大量可配置点,不同电路的实现通过可配置点的不同配置来确定,而分析脚本是获得可配置点配置的关键。FPGA 内部的可配置点分为 2 类: PIP (Programmable Interconnection Point)及 PLP(Programmable Logic Point)类型的配置点。实际上,分析脚本是一系列 XDL 文件,每个 XDL 文件中包含了 PIP 及 PLP 的不同配置选项。

图 2(a)给出了在 GRM 中,一个 PIP 类型的配置点配置选项。A 是输出点,其包含了 5 种配置选项: #OFF, B1->A, B2->A, B4->A, B4->A。在分析脚本构建时,需要构建 5 个 XDL 文件,每个 XDL 包含一种连接配置。图 2(b)给出了在 REG 中,一个 PLP 类型的可配置点,这个配置点有 3 种属性: <FFX>, <FFX_INIT_ATTR>, <FFX_SR_ ATTR>。所以在分析脚本中,这个可配置点每个属性都有 3 种可能的配置: {{#OFF,#FF,#LATCH}, {#OFF, #INIT1, #INIT0}, {#OFF, #SRHIGH, #SRLOW}}。



(a)GRM PIP 示例 (b)REG PLP 示例

图 2 可配置点示例

图 3 给出了一配置点在比特流中的控制位及控制值的分析结果形式化表示。其中,Address_x 是配置点在比特流中控制位的地址; value_x 是对应地址的控制值; PFcx 是不同的配置选项。

Address_1	Address_2	…	Address_n	
value_11	value_12	…	value_1n	PFc1
value_21	value_22	…	value_2n	PFc2
…	…		…	…
value_m1	value_m2	…	value_mn	PFcm

图 3 编程点、控制位及控制值映射

2.4 可配置点控制位、值测定办法

通过分析脚本文件,对比获得的不同配置的比特流文件差别,从而获得可配置点的控制位地址,及不同配置选项对应的控制值。以一个 PLP 类型的控制点为例说明。

下面给出了芯片为 Xilinx 的 XC2S15 型号,SLICE R2C7.S0 中的可编程点 FFY 分析脚本的生成过程示例。R2C7.S0 指明了可编程点在 FPGA 芯片上的物理地址。通

过查找 Xdlrc 文件，得到 FFY 编程点有 3 种可配置选项：#FF, #LATCH 及#OFF。加粗部分是可配置选项#FF 和#LATCH 的配置码。通过脚本生成程序，生成仅针对 R2C7.S0 的 FFY 配置项做修改的 XDL 文件，它们的 FFY 选项分别配置成：#FF, #LATCH 和#OFF；再经过 ISE 的 XDL2NCD 和 BITGEN 等相关命令使用后，生成对应的比特流文件，比较位流文件的差异，获得可编程点的各个配置选项对应的控制位地址和值。其中，0x443_1 表示 FFY 的控制位是比特流文件中地址为 0x443 字节的第 1 位。

(1)FFY 配置值为#FF 的 XDL 部分内容：

inst "inst7" "SLICE", placed R2C7 CLB_R2C7.S0

cfg "…FFX::#FF **FFY::#FF** FXMUX::#OFF G::#LUT:D=(~A3)*A1+A3*(~A1) GYMUX::G…"

(2) FFY 配置值为#LATCH 的 XDL 部分内容：

inst "inst7" "SLICE", placed R2C7 CLB_R2C7.S0

cfg"…FFX::#FF **FFY::#LATCH** FXMUX::#OFF G::#LUT:D=(~A3)*A1+A3*(~A1) GYMUX::G…"

(3)比较比特流和 XDL 文件后，得到 FFY 映射表：

FFY	0X443_1
#FF	0
#LATCH	1
#OFF	0

3　Bit2NCD 工具

在本节中，将介绍比特流还原工具 Bit2NCD。Bit2NCD 工具的工作流程如图 4 所示。其主要组件包括：

(1)逻辑配置生成器(LCG)

输入包括比特流文件、映射表及偏移量公式。输出为对应电路的逻辑部分的配置，输出结果可能包含单元多值问题。

(2)互联配置生成器(ICG)

输入包括比特流文件、映射表及偏移量公式。输出为对应电路的互联部分的配置。结果可能出现冗余互联配置信息或部分互联信息丢失情况。

(3)综合分析器(CA)

根据合法电路及 XDL 文件的规则，进一步处理 LCG 及 ICG 的处理结果，消除单元多值问题、冗余、丢失互联问题，并最终生成待还原比特流文件对应的完整合法 XDL 文件。

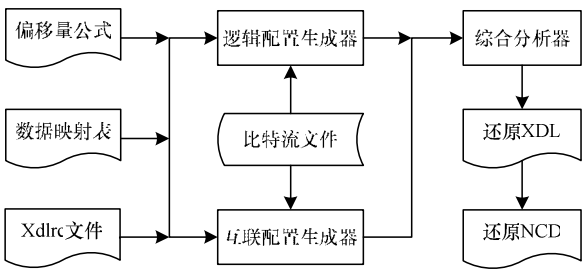


图 4　Bit2NCD 工具结构

以下将给出一个实例说明综合分析器是如何解决内部 PLP 的单元多值问题的。图 5 给出一个 SLICE 内部电路的实例图。当对应比特流通过 LCG 及 ICG 处理后，得到表 1、表 2。

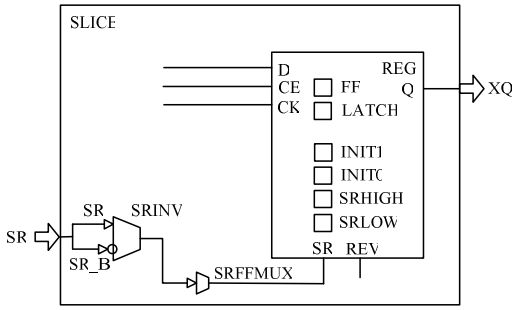


图 5　SLICE 内部结构

表 1　引脚实例表

实例名	类型名	位置	类型
P1	XQ	R2C7.S0	Output
P2	SR	R2C7.S0	Input

表 2　逻辑实例表

实例名	类型	位置	可选配置
I1	SRINV	R2C7.S0	#SR,#OFF
I2	FFX	R2C7.S1	#FF,#OFF
I3	SRFFMUX	R2C7.S2	#0,#OFF

通过搜寻引脚(Pin)实例表，首先发现 outpin P1 和 inpin P2 的使用，表明 R2C7.S0 SLICE 处于使用状态。由于 P1(XQ)端口的使用，表明 I2(FFX 类型的可编程点)处于使用状态，因此剔除 FFX 配置成#OFF 状态的可能，进而唯一确定 I2 配置成#FF。同理，由于 P2(SR)端口的使用，则 I1 唯一确定配置成#SR 选项。当 P2 确定为#SR 后，根据 P2 与 P3 在电路中的物理连接位置关系，I3(SRFFMUX)将唯一确定配置为#0。

经过多次迭代分析，最终得到的 R2C7.S0 SLICE 的 XDL 逻辑配置部分内容为：

inst "DEMO" "SLICE", placed R2C7 CLB_R2C7.S0,

cfg " SRINV::#SR, FFX::#FF, SRFFMUX::#0…";

经过综合分析器后，得到了比特流文件对应的还原 XDL 文件，再通过使用 xdl-xdl2ncd 命令，将 XDL 文件转换为 NCD 文件，最终，完成了比特流的解析及逆向还原成 NCD 工作。

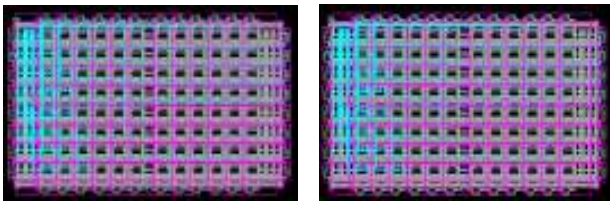
4　实验及性能分析

为验证比特流分析及还原工作的可行性，在 Xilinx 的 Spartan-II 系列的 XC2S15 芯片上做了相关验证实验。图 6 给出了 FIFO 测试实例的还原结果对比图。其中，图 6(a)为实际情况下，原电路 NCD 文件的 FPGA Editor 视图和 XDL

文件部分视图；图 6(b)为经过 BIT2NCD 工具分析后，得到的还原 NCD 文件的 FPGA Editor 视图和还原 XDL 文件部分视图。从 FPGA Editor 视图可以直观得出还原电路与原电路形态相同的结论；而对比 XDL 文件，可得到准确的量化结果。在图 6 顶端，图 6(a)中的网线 NET“_GROUND.0_4”gnd 对应图 6(b)中的 NET“N38”gnd。通过对比，发现原文件与还原文件的差异仅存在于命名，其中，关于 inpin/outpin 和 pip 的使用，都是完全相同一致的。

```
net "_GROUND 0_4" gnd,
  outpin "R1C8_CLE_R1C8 S0" Y ,
  inpin "data_out_1" F4 ,
  inpin "data_out_1" G4 ,
  inpin "data_out_5" G4 ,
  pip R1C7 E22 -> E_F22 ,
  pip R1C7 E_F22 -> S0_F_B4 ,
  pip R1C7 E_F22 -> S0_G_B4 ,
  pip R1C7 N20 -> N_F20 ,
  pip R1C7 N20 == E16 ,
  pip R1C7 N_F20 -> S1_G_B4 ,
  pip R1C8 OUT5 -> W16 ,
  pip R1C8 OUT7 -> W22 ,
  pip R1C8 S0_Y -> OUT5 ,
  pip R1C8 S0_Y -> OUT7 ,
  ;
```

```
net "N38" gnd,
  outpin "inst43" Y ,
  inpin "inst19" G4 ,
  inpin "inst11" F4 ,
  inpin "inst11" G4 ,
  pip R1C8 S0_Y -> OUT5 ,
  pip R1C8 OUT5 -> W16 ,
  pip R1C7 N20 == E16 ,
  pip R1C7 N20 -> N_P20 ,
  pip R1C7 E22 -> E_P22 ,
  pip R1C7 N_P20 -> S1_G_B4 ,
  pip R1C8 S0_Y -> OUT7 ,
  pip R1C8 OUT7 -> W22 ,
  pip R1C7 E_P22 -> S0_F_B4 ,
  pip R1C7 E_P22 -> S0_G_B4 ,
  ;
```



(a)原始部分 XDL 及 NCD (b)还原部分 XDL 及 NCD

图 6 还原结果对比

通过对比原始 XDL 及还原得到的 XDL 文件，可以得到结论：(1)控制位、值及配置选项生成机制是高效正确的。(2)冗余控制位消除机制及偏移量公式理论是正确的、有效的。(3)Bit2NCD 工具的分析还原方法是可行、正确的。

5 结束语

本文展示了将 FPGA 比特流文件逆向还原为 NCD 文件所需的全部工作：(1)采用专用设计的分布式计算架构，使用特有的位流配置分析方法，完成对海量位流配置基础数据的自动化分析和整理，获得完整的配置数据库。(2)基于已获得比特流配置数据，设计和实现了 Bit 转 NCD 文件的整个分析流程和方法；在 Spartan 2 系列芯片上得到正确性验证。

位流解析和电路还原等方法普遍适用于 Xilinx 公司的所有 FPGA 芯片，包括 Spartan-3、Spartan-3E、Virtex-II、Virtex-4、Virtex-5 及目前更新的芯片。下一步将完善相关的位流分析工具和 BIT2NCD 位流还原工具。

参考文献

[1] Compton K, Hauck S. Reconfigurable Computing: A Survey of Systems and Software[J]. ACM Computing Surveys, 2002, 34(2): 171-210.

[2] Donthi S, Haggard R L. A Survey of Dynamically Reconfigurable FPGA Devices[C]//Proc. of the 35th Southeastern Symposium on System Theory. [S. l.]: IEEE Press, 2003.

[3] Sekanina L, Arslan T. Evolvable Components——From Theory to Hardware Implementations[J]. Genetic Programming and Evolvable Machines, 2005, 6(4): 461- 462.

[4] Steige C, Walder H. Operating Systems for Reconfigurable Embedded Platforms: Online Scheduling of Real-time Tasks[J]. IEEE Transactions on Computers, 2004, 53(11): 1393-1407.

[5] Sekanina L. Towards Evolvable IP Cores for FPGAs[C]// Proc. of NASA/DoD Conference on Evolvable Hardware. [S. l.]: IEEE Computer Society, 2003: 145-154.

[6] Kępa K, Morgan F, Braun L, et al. FPGA Analysis Tool: High-level Flows for Low-level Design Analysis in Reconfigurable Computing Reconfigurable Computing[C]// Proc. of the 5th International Workshop on Reconfigurable Computing: Architectures, Tools and Applications. Berlin, Germany: Springer-Verlag, 2009: 62-73.

[7] Note J B, Rannaud E. From the Bitstream to the Netlist[C]// Proc. of the 16th International ACM/SIGDA Symposium on Field Programmable Gate Arrays. New York, USA: ACM Press, 2008.

[8] Xilinx. JBits 3.0 SDK for Virtex-II[EB/OL]. (2004-04-08). http://www. xilinx.com/labs/projects/jbits/.

[9] Moradi A, Kasper M, Paar C. On the Vulnerability of FPGA Bitstream Encryption Against Power Analysis Attacks[EB/OL]. (2011-07-19). http://eprint.iacr.org/2011/390.

[10] Telikepalli A. Is Your FPGA Design Secure?[EB/OL]. (2003-05-16). http://www.xilinx.com/publications/xcellonline.

编辑 顾逸斐