

引用格式:冷明,孙凌宇,郭晨. 一种 XDL 网表电路到有向超图的转换系统[J]. 微电子学与计算机, 2021, 38(2): 66-71.
[LENG M, SUN L Y, GUO C. An effective converter to translate XDL netlist description file into directed hyper-graph[J]. Microelectronics & Computer, 2021, 38(2): 66-71.]

一种 XDL 网表电路到有向超图的转换系统

冷 明, 孙凌宇, 郭 晨

(井冈山大学 电子与信息工程学院计算机科学系, 江西 吉安 343009)

摘 要: FPGA 电路网表的 XDL 描述文件, 不仅能通过解析来抽取 FPGA 设计的 Inst 电路单元和 Net 电路信号, 而且能用于构建 FPGA 电路网表中信号传播的抽象数学模型. 本文通过有向超图来构建 FPGA 电路网表中信号的前向拓扑关系, 其中 FPGA 电路单元的有效管脚表示为超图结点, 管脚间的外部连线、管脚内的电路逻辑功能表示为有向超边. 进而, 本文阐述了 XDL 网表级电路描述文件到有向超图的转换流程, 给出了 XDL 网表级电路描述文件编译所需的 EBNF 表达式、XDL 网表级电路到有向超图的转换算法. 最后, 本文基于 RapidSmith 开源软件和 Java 语言实现了转换系统, 并选用基于 Xilinx 公司的 Virtex-4 型号 FPGA 的测试用例, 实现了 XDL 网表级电路描述文件转换至有向超图, 验证了 XDL 网表级电路描述文件到有向超图转换系统的有效性和正确性.

关键词: 现场可编程门阵列; XDL 网表描述文件; 有向超图; 解析; 转换

中图分类号: TP391

文献标识码: A

文章编号: 1000—7180(2021)02—0066—06

An effective converter to translate XDL netlist description file into directed hyper-graph

LENG Ming, SUN Ling-yu, GUO Chen

(School of Electronics and Information Engineering, Jinggangshan University, Ji'an 343009, Jiangxi, China)

Abstract: The XDL netlist description file of the FPGA is not only used to parse for the extraction of the instance and net inside FPGA design, but also used to translate into the general abstract signal propagation model of FPGA netlist. In this paper, the directed hyper-graph is used to describe the FPGA internal signal forward topology, whose vertex can be considered as the in-use pin of instance and directed hyper-edge can be represented as the net connection or the instance function between the in-use pins of instance. Furthermore, this paper not only presents the workflow of the converter to translate XDL netlist description file into directed hyper-graph, but also gives the EBNF expression of the XDL compiler and the translate algorithm of directed hyper-graph. Finally, this paper implements the converter in Windows OS and JAVA based on the source code of RapidSmith software. The experiment of the test bench of Xilinx's Virtex-4 shows that the converter could translate XDL netlist description file into directed hyper-graph. The correctness and effectiveness of the converter based on the translate algorithm of directed hyper-graph is also proved by the experiment.

Key words: FPGA; XDL netlist description file; directed hyper-graph; parse; translate

1 引言

随着超大规模集成电路的制造工艺进入纳米工艺时代,导致现场可编程门阵列的噪声容限日益减少^[1],对高能粒子辐射和噪声干扰愈发敏感,软错误率(SER)呈指数增长^[2]. XDL(Xilinx Design Language)^[3]是 Xilinx 公司提供的一种描述 FPGA 设计内部结构信息的特征化物理设计语言,使用 BNF 巴科斯范式描述 FPGA 实现特定功能硬件设计的网表级电路,其中的逻辑资源配置信息和布线资源互连信息,可用于准确分析 SRAM 型 FPGA 器件

的单粒子软错误敏感性,提升容错可靠性的计算精确度。

图作为非线性结构,每一个结点可以与多个其它结点相关联,结点之间的关系是“多对多”,可以用于布尔函数操作^[4]等实现。相比图而言,超图为电路描述提供了更为强大的数学模型:每条超边可以连接两个以上的结点,对应于电路连线可以连接两个以上的电路器件管脚。本文基于 SRAM 型 FPGA 器件的单粒子软错误传播机理^[5],构建 XDL 网表级电路的信号传播模型“有向超图^[6,7]”,提出了 XDL 网表级电路到有向超图的转换算法。

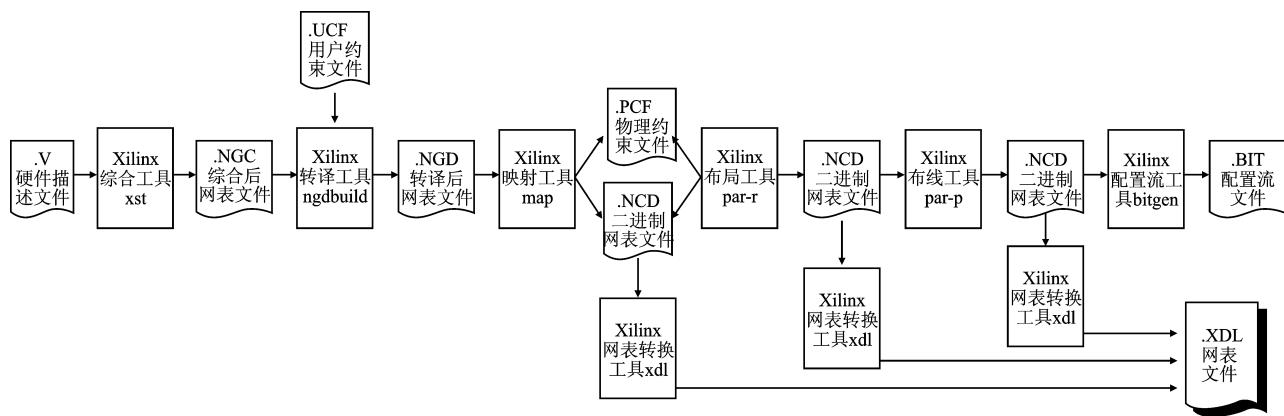


图 1 FPGA 设计映射到 XDL 网表文件的流程图

本文采用了有向超图来构建网表级电路的信号传播模型:电路器件的有效管脚建模成有向超图中的结点,电路管脚间的外部连线、电路管脚内的电路逻辑功能建模成有向超图中的有向超边。相比基于功能模块间近似耦合关系的单粒子软错误传播特性计算方法^[8],本文提出的 XDL 电路网表到有向超图转换算法,精细刻画了信号的前向拓扑关系,体现了单粒子软错误在 SRAM 型 FPGA 器件中的传播特性。

本文阐述了 XDL 网表电路文件到有向超图的转换流程图,给出了遍历电路网表的 Inst 电路单元,抽取 XDL 电路网表的逻辑配置信息以生成超图结点和有向超边;遍历电路网表的 Net 电路信号,抽取 XDL 电路网表的互联配置信息以生成有向超边。最后,本文基于 RapidSmith 开源软件^[9]和 Java 语言设计实现了转换系统,并选用基于 Xilinx 公司的 Virtex-4 型号 FPGA 的测试用例,实现了 XDL 网表级电路描述文件转换至有向超图,从而验证了转换算法的有效性。

2 XDL 网表级电路描述文件的解析

本文通过阐述 FPGA 设计实现的工作流程表

明:XDL 网表级电路描述文件到有向超图的转换,不仅是 FPGA 信号前向拓扑关系建模的需要,更是 SRAM 型 FPGA 器件单粒子软错误敏感性计算的重要环节。本文针对 XDL 网表级电路描述文件到有向超图转换的需要,解析 XDL 网表级电路描述文件,以获得 FPGA 设计的逻辑配置和互联配置信息。

图 1 给出了基于 Xilinx 公司 ISE 套装的综合工具 xst、转译工具 ngdbuild、映射工具 map、布局工具 par(-r)、布线工具 par(-p)、配置流工具 bitgen 和网表转换工具 xdl,实现硬件设计到 FPGA 配置流的开发全流程。其中,FPGA 映射后、布局后和布线后的 NCD 网表文件是面向计算机的二进制格式网表文件,不便于人们理解和分析。网表转换工具 xdl 可实现 NCD 二进制格式网表文件与 XDL 文本格式网表文件的相互转换。

XDL 网表级电路描述文件的架构如图 2 所示,每个 XDL 文件对应一个 FPGA 设计的硬件实现,使用 BNF 范式详细地描述了整个 FPGA 设计的布局布线信息,包含 design,module,instance 和 net 等语句,提供了 FPGA 设计在芯片微观级别(网表级)

逻辑资源的配置信息和布线资源的互连信息. 表 1 给出了 XDL 网表级电路描述语言的 EBNF 表达式语法规则.

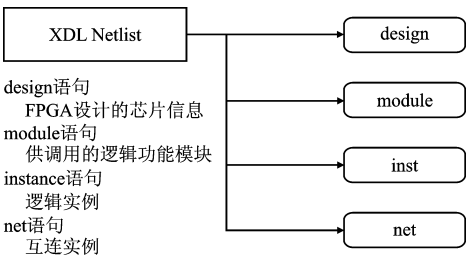


图 2 XDL 网表级电路描述文件的架构图

表 1 XDL 网表级电路描述语言的 EBNF 表达式

序	XDL 网表级电路描述语言的 EBNF 表达式
1	design statement ::= design {<design_name> <part> <ncd version>} {<device> <package> <speed> <ncd_version>}
2	module statement ::= module<module_name>, "left right" cfg <string>; <port list> <inst list> <net list> endmodule <module_name>
3	instance statement ::= inst <inst_name> <sitedef>, {<placed <tile> <site>, cfg <string>} {<unplaced, cfg <string>}
4	net statement ::= net <net_name> <type>, outpin statement {, inpin statement} * {, pip statement} *
5	outpin statement ::= outpin <inst_name> <inst_pin>
6	inpin statement ::= inpin <inst_name> <inst_pin>
7	pip statement ::= pip <tile> <wire0> <dir> < wire1>, # [<rt>]

定义 1: design 语句采用“design”关键字表示, 其生产式规则 $\langle \text{design 语句} \rangle ::= \text{design} \langle \text{design_name} \rangle \langle \text{device} \rangle \langle \text{package} \rangle \langle \text{speed} \rangle \langle \text{ncd_version} \rangle$, 其中 design_name 是 FPGA 设计的名称; device 是使用的 FPGA 芯片型号.

定义 2: module 语句采用“module”关键字表示, 其生产式规则 $\langle \text{module 语句} \rangle ::= \text{module} \langle \text{module_name} \rangle, \text{"left" | "right"} \text{ cfg } \langle \text{string} \rangle; \langle \text{port list} \rangle \langle \text{inst list} \rangle \langle \text{net list} \rangle \text{ endmodule } \langle \text{module_name} \rangle$; 其中 module_name 是 module 的名称, cfg 中的 string 信息是描述 module 在 FPGA 底层电路的配置状态; port 是 module 的 I/O 端口; inst 是 module 内部的实例; net 是 module 内部的信号.

定义 3: instance 语句采用“inst”关键字表示, 其生产式规则 $\langle \text{instance 语句} \rangle ::= \text{inst} \langle \text{name} \rangle \langle \text{sitedef} \rangle, \text{placed } \langle \text{tile} \rangle \langle \text{site} \rangle, \text{cfg } \langle \text{string} \rangle$, 其中 name 是 instance 的名称; sitedef 是 instance 的类型; placed 中的 tile 和 site 信息是描述

instance 在 FPGA 芯片中的位置; cfg 中的 string 信息是描述 instance 在 FPGA 底层电路的配置状态.

定义 4: net 语句采用“net”关键字表示, 其生产式规则 $\langle \text{net 语句} \rangle ::= \text{net } \langle \text{name} \rangle \langle \text{type} \rangle \langle \text{outpin list} \rangle \langle \text{inpin list} \rangle \langle \text{pip list} \rangle$, 其中 name 是 net 的名称; type 是 net 的类型; outpin 是 net 的起点, 把信号由 outpin 传出至 net; inpin 是 net 的终点, 把信号由 net 传入至 inpin; pip 是 net 的内部互联点, 描述了信号传输的路径.

定义 5: 采用有向超图 $H^G = (V^H, E^H)$ 构建描述 XDL 网表电路的前向级联关系, 其中有向超图中的顶点集合 $V^H = \{v_i | 1 \leq i \leq n\}$ 表示各电路单元的有效管脚, 有向超边集合 $E^H = \{e_j^h | e_j^h \subseteq V^H, 1 \leq j \leq m\}$ 表示各电路单元有效管脚间的信号有向传递关系.

3 XDL 网表级电路描述文件到有向超图的转换流程及算法

基于定义 1~定义 5, XDL 网表级电路到有向超图的转换算法的主要思想是: ① 电路器件的有效管脚表示为有向超图中的结点. ② 针对电路管脚间的外部连线、电路管脚内的电路逻辑功能, 都表示为有向超图中的有向超边. ③ 相比有向图而言, 有向超图为 XDL 网表电路提供了更为强大的数学模型: 每条有向超边可以连接两个以上的结点, 对应于电路连线可以连接两个以上的有效电路器件管脚, 或电路逻辑功能可以连接多输入管脚. ④ 在每条有向超边中, 唯一的输出管脚对应于有向超边的尾端结点, 其余输入管脚对应于有向超边源端子集结点.

基于构建 XDL 网表级电路的信号传播模型的需要, 我们设计并实现了 XDL 网表级电路描述文件到有向超图的转换系统, 具体流程如图 3 所示, 通过对 XDL 网表文件的词法分析、语法分析、语义分析和中间代码生成等编译步骤, 进而从 Ins 模块信息抽取 XDL 电路网表的逻辑配置信息, 从 Net 模块信息抽取 XDL 电路网表的互联配置信息后构造完整电路网, 调用信号的前向拓扑关系生成算法, 将电路网转换为有向超图.

```
INPUT: . XDL file
OUTPUT: directed hyper-graph file
01Begin
02 Read <design_name>, <device> <package>,
<speed>, <ncd_version> from . XDL file;
03For Each Ins module from . XDL file according to i-
dentifier 'inst' Do Begin
04 Read <SITEDEF> type and <CFG> con-
```

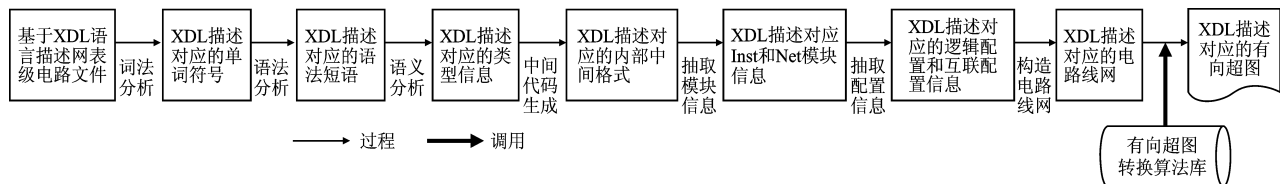


图 3 XDL 网表级电路描述文件到有向超图转换系统的流程图

figuration of Inst module

05 Getin-use pin list by analyzing <CFG> configuration of Inst module

06 Create the vertex for each in-use pin of Inst module

07 Resolvlogic function of each in-use outpin by analyzing <CFG> configuration

08 Create the directed hyper-edge for each in-use outpin of Inst module

09 Add the vertex of outpin as end nod of the directed hyper-edge

10 Addvertices of inpin of logic function to the source subset of directed hyper-edge

11End For

12For Each Net module from . XDL file according to identifier ‘net’ Do Begin

13 Create the directed hyper-edge for Net module

14 Readoutpin and inpin of Net module

15 Add the vertex of outpin as end nod of directed hyper-edge

16 Addvertices of inpin to the source subset of directed hyper-edge

17End For

18 Specified a number to eachvertex and directed hyper-edge of directed hyper-graph

19 Outputtwo entries (# vertices, # hyper-edges) as the first line directed hyper-graph file

20For Each directed hyper-edge of directed hyper-graph Do Begin

21 Output specific number of end nod ofdirected hyper-edge as new line

22 Output specific numbers of the source subset of directed hyper-edge at same line

23End For

24End

图 4 XDL 网表级电路到有向超图的转换算法伪代码

XDL 网表级电路到有向超图的转换算法伪代码如图 4 所示,其详细步骤如下:① 遍历电路网表的 Inst 电路单元模块信息,依次处理每个 Inst 电路单元:抽取 XDL 电路网表的逻辑配置信息,获取有效管脚信息列表,为每个有效管脚生成一个超图结点;根据 Inst 电路单元的不同类型,配合 CFG 配置信息获取的底层电路配置状态,解析每个输出管脚的逻辑功能表达式,为每个输出管脚生成一条有向超边,且该有向超边的尾端结点为该输出管脚对应

的超图结点;依据该输出管脚的逻辑功能表达式,输入管脚对应的超图结点加入到该有向超边的源端子集中.② 遍历电路网表的 Net 电路信号模块信息,依次处理每个 Net 电路信号:为其生成一个超图结点;读取其 INPUT 端点信息,依据信息中指定的 Inst 电路单元和管脚信息找到相应的超图结点,作为该有向超边的尾端结点;读取其每个 OUTPUT 端点信息,依据信息中指定的 Inst 电路单元和管脚信息找到相应的超图结点,加入该有向超边的源端子集.③ 遍历有向超图的超图结点和有向超边,生成有向超图文件:分别为每个超图结点和有向超边单独编号;输出有向超图文件第一行,第一个参数是有向超边数目,第二个参数是超图结点数;遍历有向超边,每条超边输出为一行,先输出尾端结点的编号,再输出源端子集中超图结点的编号.

有向超图在不同的物理存储结构下会产生不同的算法时间复杂度和空间复杂度^[10].因此,本文的转换算法采用压缩存储格式(CSF: Compressed storage format)^[11].CSF 存储格式使用数组 vwgts 存储结点的信息,使用 hewgts 数组存储有向超边的信息,使用 eptr 数组存储每条有向超边对应邻接结点列表的起始位置信息,使用 eind 数组存储每条有向超边的邻接结点信息.假设数组下标和结点编号从零开始,则第 i 条有向超边的邻接结点列表存储在 eind 数组中,位置从 eptr $[i]$ 开始,到 eptr $[i+1]-1$ 结束,即从 eind $[eptr[i]]$ 到 eind $[eptr[i+1]-1]$,且有向超边的尾端结点存储在 eind $[eptr[i]]$ 数组元素中.

设 XDL 网表电路中 # Instances 数为 m , # Pins 数为 n , # Nets 数为 t .该转换算法中第 3 行到第 11 行为每个 Inst 电路单元建立超图结点和有向超边,其循环次数为 m ,第 6 行为每个管脚建立超图结点,其循环次数为 n ;第 12 行到第 17 行为每个 Net 电路信号模块生成有向超边,其循环次数为 t ,第 15 行到第 16 行为每个管脚建立超图结点,其循环次数为 $t \times n$;第 20 行到第 23 行输出每条有向超边的超图结点信息,其循环次数为 $t \times n$,因此算法

时间复杂度为 $\Theta(n+2\times t\times n)$.

该转换算法的空间复杂度取决于有向超图的 CSF 存储空间. 基于 CSF 存储结构, 包含大小为 m 的 vwgts 数组, 大小为 n 的 eind 数组, 大小为 t 的 hewgts 数组和 eptr 数组, 因此空间复杂度为 $\Omega(m+n+2\times t)$.

4 XDL 网表级电路描述文件到有向超图的转换实验及结果

针对 XDL 网表级电路描述文件到有向超图的转换实验, 我们基于 Xilinx 公司 Virtex-4 型号 XC4VSX55 FPGA 芯片对测试用例进行综合、转译、映射、布局、布线、配置流生成和网表转换, 实现硬件设计到 FPGA 配置流的全步骤, 并通过网表转换工具 xdl 将布局布线后的 NCD 二进制格式网表文件, 转换为 XDL 文本格式网表文件. 进而, 我们设计实现了 XDL 网表级电路描述文件到有向超图转换系统, 将 XDL 网表文件转换为有向超图, 并保存为有向超图文件.

读取 XDL 电路网表的 10 个 Inst 电路单元, 抽取逻辑配置信息进行分析, 获取 29 个有效管脚生成 29 个超图结点, 并为其中的 11 个输出管脚生成第 1~11 条有向超边. 读取 XDL 电路网表的 9 个 Net 电路信号, 经抽取互联配置信息进行分析, 生成第 12~20 条有向超边.

```
01 20 29
02 2 4 5 6 3
03 1 7 8 9 3
04 11 10
05 13 12 14
06 15 16
07 17 18
08 19 20
09 22 21 23
10 25 24
11 26 27
12 28 29
13 2 24
14 27 23
15 17 26
16 19 4 7
17 13 12 3
18 28 6 9
19 22 14 21
20 1 10
21 15 5 8
```

图 5 面向 Virtex-4 型号 FPGA 的测试用例生成的有向超图文件

该实验得到如图 5 所示的有向超图文件, 并通

过了超图优化划分软件包 hMeTiS 对有向超图的检测^[10]. 有向超图文件第 1 行显示超图结点数是 29, 有向超边数是 20. 第 2~21 行输出有向超边的结点信息, 其中每行第 1 个编号为该有向超边的尾端结点, 后续编号为其源端子集的结点.

SLICE 电路结点生成第 1 条和第 2 条有向超边, 有向超边 {2 4 5 6 3} 中第一个尾端结点编号 2 映射至输出信号管脚 YQ, 其余源端子集编号 {4 5 6 3} 映射至输入信号管脚 G1~G3 和时钟信号管脚 CLK; 有向超边 {1 7 8 9 3} 中第一个尾端结点编号 1 映射至输出信号管脚 XQ, 其余源端子集编号 {7 8 9 3} 映射至输入信号管脚 F1~F3 和时钟信号管脚 CLK. 两条有向超边共用超图结点 3, 表明共用时钟信号 CLK.

5 结束语

本文采用有向超图的数学模型来构建 XDL 网表级电路, 描述 Inst 电路单元和 Net 电路信号, 其中电路器件的有效管脚表示为有向超图中的结点, 电路管脚间的外部连线、电路管脚内的电路逻辑功能表示为有向超图中的有向超边. 本文解析 XDL 网表电路文件以获得 FPGA 设计的逻辑配置和互联配置信息: 遍历电路网表的 Inst 电路单元, 抽取 XDL 电路网表的逻辑配置信息以生成超图结点和有向超边; 遍历电路网表的 Net 电路信号, 抽取 XDL 电路网表的互联配置信息以生成有向超边. 进而, 本文探讨了 XDL 网表级电路描述文件转换至有向超图系统的实现步骤, 不仅给出了编译 XDL 网表级电路描述文件所需的 EBNF 表达式, 而且给出了 XDL 网表级电路到有向超图的转换算法, 并进行了相应的时间复杂度分析. 最后, 本文基于 Rapid-Smith 开源软件和 Java 语言实现了转换系统, 并选用基于 Xilinx 公司的 Virtex-4 型号 FPGA 的测试用例, 实现了 XDL 网表级电路描述文件转换至有向超图, 验证了 XDL 网表级电路描述文件到有向超图转换系统的有效性和正确性.

参考文献:

[1] ITRS. International Technology Roadmap for Semiconductors (ITRS) Update Overview [R]. Washington, DC: Semiconductor Industry Association, 2019, Available [http:// www. itrs2. net/](http://www.itrs2.net/).
[2] WANG B. Investigation of single event effects induced by heavy ions in micro/nano SRAM devices

- [D]: Lanzhou: Institute Of Modern Physics, Chinese Academy of Sciences, 2017.
- [3] BECKHOFF C, KOCH D, TORRESEN J. The Xilinx design language (XDL): tutorial and use cases [C]//Proceedings of the 6th International Workshop on Reconfigurable Communication-centric Systems-on-Chip. Montpellier, France: IEEE, 2011: 1-8. DOI: 10.1109/ReCoSoC.2011.5981545.
- [4] BRYANT R E. Graph-based algorithms for Boolean function manipulation [J]. IEEE Transactions on Computers, 1986, C-35 (8): 677-691. DOI: 10.1109/TC.1986.1676819.
- [5] LIU J F. Research on the Soft Error Rate Calculation for SRAM-based FPGA[D]: Xi'an: Xidian University, 2017.
- [6] LENG M, SUN L Y, GUO K Q, et al. An effective algorithm for converting ISPD98 netlist into hypergraph[J]. Microelectronics & Computer, 2011, 28 (9): 111-114. DOI: 10.19304/j.cnki.issn1000-7180.2011.09.029.
- [7] LENG M, SUN L Y, GUO K Q, et al. An effective converter to translate VLSI Design into hypergraph [J]. Microelectronics & Computer, 2012, 29(2): 7-12. DOI: 10.19304/j.cnki.issn1000-7180.2012.02.002.
- [8] YUAN J L. Soft error sensitivity analysis of SRAM-based FPGA based on XDL[D]. Changsha: National University of Defense Technology, 2015.
- [9] LAVIN C, PADILLA M, LAMPRECHT J, et al. RapidSmith: do-it-yourself CAD tools for Xilinx FPGAs[C]//Proceedings of the 21st International Conference on Field Programmable Logic and Applications. Chania, Greece: IEEE, 2011: 349-355. DOI: 10.1109/FPL.2011.69.
- [10] YIN R K, TAO Y L. Data structure [D]: Beijing: Tsinghua University Press, 1999: 30-35.
- [11] KARYPIS G, KUMAR V. HMetis 1.5.3: A Hypergraph Partitioning Package [R]. Technical report, Department of Computer Science, University of Minnesota, 1998. Available on the WWW at URL <http://www.cs.umn.edu/Metis>.

作者简介:

冷明男, (1975-), 博士, 教授. 研究方向为 VLSI 算法分析与设计、组合分析与优化. E-Mail: lengming@jgsu.edu.cn

孙凌宇女, (1976-), 博士, 教授. 研究方向为电子设计自动化、组合算法.

郭晨男, (1979-), 博士, 教授. 研究方向为新型互连网络和系统级故障诊断、组合算法.