文章编号：1674-8085(2017)01-0049-07

VPR 的 FPGA 结构描述文件的解析研究

*冷 明[1]，孙凌宇[1]，周 宇[2]，朱 平[1]

(1.井冈山大学电子与信息工程学院，江西，吉安 343009；2. 井冈山大学网络信息中心，江西，吉安 343009)

摘 要：作为描述 FPGA 硬件资源的结构描述文件，不仅是 VPR 布局布线工具的重要组成元素，而且是 FPGA 硬软件工程师相互沟通的桥梁。阐述了 VPR 布局布线的工作流程，对 VPR 的 FPGA 结构描述文件进行解析，以正确地表示 FPGA 芯片内部的通道、开关、逻辑资源、布线资源等结构信息。进而，以可配置逻辑块 CLB 和 Virtex-6 的 Slice 结构为例，分别给出了相应的 FPGA 结构描述文件，重点讨论了<pb_type>逻辑单元块和<interconnect>内部连接的 XML 代码。FPGA 结构描述文件正确地被描述和解析，有助于开展 FPGA 结构在内存物理中的存储和转换等后续研究工作。

关键词：VPR；现场可编程门阵列；结构描述文件；解析

中图分类号：TP391 **文献标识码**：A DOI:10.3969/j.issn.1674-8085.2017.01.010

RESEARCH ON PARSE OF FPGA ARCHITECTURE DESCRIPTION FILE OF VPR

* LENG Ming, SUN Ling-yu, ZHOU Yu, ZHU Ping

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

2. Center of network information, Jinggangshan University, Ji'an 343009, China)

Abstract: As the communication channel between the hardware engineer and software engineer, FPGA architecture description file is crucial component of **V**ersatile **P**lacement and **R**outing tool (VPR) which is used to describe FPGA hardware resources. We present the placement and routing process of VPR. We also parse the FPGA architecture description file of VPR to make sure that we can accurately denote channels, switches, logic resources, routing resources and others architecture information inside the FPGA chip. Furthermore, we give the corresponding the FPGA architecture description files of configurable logic block and Virtex-6 slice. We also discuss the XML codes of physical blocks and intra-block interconnects. It is important to do research on its memory storage format and conversion that we can accurately parse the FPGA architecture description file.

Key words: versatile placement and routing; FPGA; architecture description file; parse

0 引言

自 Xilinx 公司首次上市 FPGA 以来，FPGA 从最初的 1200 个可用门电路规模，到上世纪末的几十万门电路规模，再到最新的 Virtex®-7 系列的千万门电路规模，FPGA 的集成度不断地提高。相比硬接线固定模式的专用集成电路 ASIC，使用可编程

收稿日期：2016-08-03；修改日期：2016-12-13

基金项目：国家自然科学基金项目（61363014）；江西省青年科学家培养对象计划（20153BCB23003）；江西省科技厅科技支撑项目（20132BBE50048）；江西省自然科学基金项目（20161BAB202050）；江西省教育厅科学技术研究项目（GJJ150779,GJJ16079）

作者简介：*冷 明(1975-)，男，教授，博士，主要从事 VLSI 算法分析与设计，组合分析与优化等方面的研究(E-Mail: lengming@jgsu.edu.cn)；
孙凌宇(1976-)，女，教授，硕士，主要从事电子设计自动化，组合算法等方面的研究(E-Mail: lzylmsly@gmail.com)；
周 宇(1974-)，男，高级实验师，硕士，主要从事组合分析与优化方面的研究(E-Mail: zy@jgsu.edu.cn)；
朱 平(1955-)，男，教授，硕士，主要从事算法分析与设计方面的研究((E-Mail: zhuping810@163.com).

模式 FPGA 器件设计的电子产品具有响应时间快、设计成本低、研发周期短、可重复配置、稳定性强等优点，在超高速传输、数字信号处理、汽车电子、医疗和工业控制等领域均得到了广泛应用。

随着超大规模集成电路技术的发展，VLSI 的制造工艺已从深亚微米工艺时代进入纳米工艺时代，国际半导体技术蓝图报告[1]预测 2020 年 VLSI 特征尺寸将减少到 5 nm。现场可编程门阵列 FPGA 的规模急剧增加，极大地扩充了 FPGA 中逻辑功能模块的种类和数量，集成了越来越多的 IP 模块，如 FIFO/RAM/ROM 等片上存储器、PLL/DLL 锁相环、高速串行收发器、嵌入式 CPU 和 DSP 处理器等, 极大地提升了 FPGA 器件应用的灵活性和硬件并行的处理性能。

VPR（Versatile Placement and Routing for FPGAs）是由加拿大多伦多大学 Jason 和 Jonatha 教授等开发的，集 FPGA 布局和布线为一体的开发探索、架构参数、性能评估功能的最新 EDA 工具，获得了 FPGA 学术界和工业界最广泛的认可，为 FPGA 架构研究提供了便利手段[2-3]。研究人员根据 FPGA 结构研究和设计的需要，通过布局布线测试对所提出的 FPGA 结构进行验证，通过测试数据评价该芯片结构并调整优化 FPGA 的结构参数。FPGA 结构描述文件作为 FPGA 硬件平台、FPGA 布局布线工具、FPGA 硬件软件 IP 核、FPGA 开发板之间相互沟通的桥梁，其应用相当广泛，例如：根据 FPGA 结构描述文件构建相应的布线资源图，用于基于 FPGA 互连资源的布线模块；FPGA 结构描述文件被位流文件生成模块用于生成相应的位流文件[4]。

本文通过阐述 VPR 布局布线的工作流程，可以发现使用 FPGA 的结构描述文件正确合理地描述 FPGA 硬件资源，是 VPR 布局布线过程中重要的环节。针对 FPGA 结构研究的需要，对 VPR 的结构描述文件进行解析，从而获得 FPGA 芯片内部的通道、开关、逻辑资源、布线资源等结构信息。进而，以可配置逻辑块 CLB 和 Virtex-6 型号 Slice 结构为例，分别给出了相应的 FPGA 结构描述文件，重点讨论了<pb_type>逻辑单元块和<interconnect>内部连接的 XML 代码。

1 VPR 的 FPGA 结构描述文件的解析

1.1 VPR 的工作流程

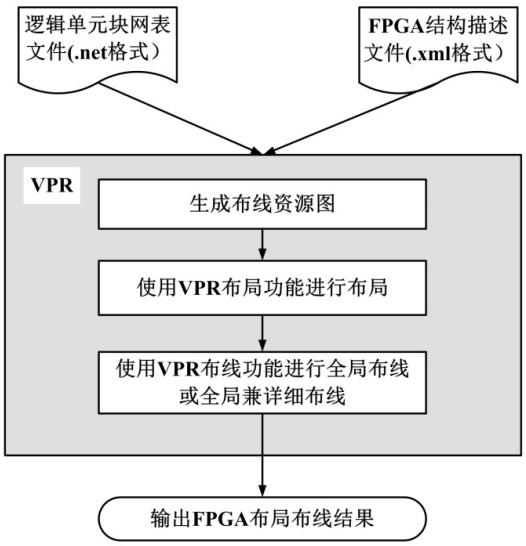

图 1 VPR 的工作流程图[2-3]

Fig.1 The workflow diagram of VPR[2-3]

VPR 的工作流程图如图 1 所示。VPR 输入 FPGA 设计的逻辑单元块网表文件和 FPGA 硬件资源的结构描述文件。其中，FPGA 设计采用“.net”格式的逻辑单元块网表文件来表示，存储所需要布局布线的逻辑网表；FPGA 硬件资源采用“.xml”格式的结构描述文件来表示，存储 FPGA 芯片的版图和布线结构[2]。首先，VPR 对 FPGA 结构描述文件进行词法语法分析和中间代码生成，解析生成相应的布线资源图。然后，VPR 依据网表文件构造 FPGA 设计相应的逻辑网表，实现逻辑网表在物理存储空间中的表示，进而对逻辑网表完成布局；在完成布局后，VPR 在布线资源图上，按照布线算法进行全局布线或者全局兼详细布线[3]。由此可见，正确完整地描述 FPGA 硬件资源并生成相应的 FPGA 结构描述文件，是 VPR 布局布线过程中重要的环节。

1.2 VPR 的 FPGA 结构描述文件的解析

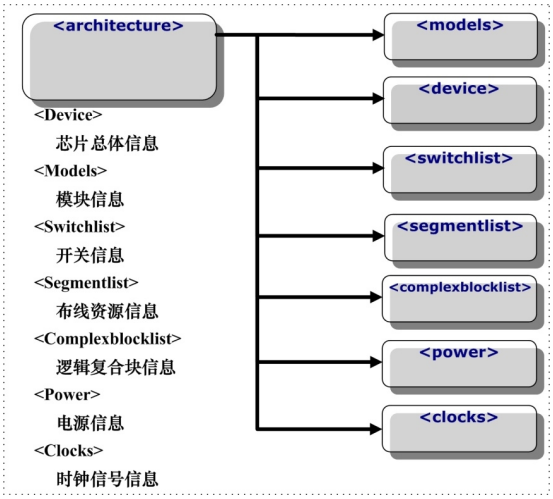

图 2　FPGA 结构描述文件的架构图

Fig.2 The architecture diagram of VPR architecture description file

VPR 的 FPGA 结构描述文件采用 XML 可扩展标记语言来描述 FPGA 芯片内部的通道、开关、逻辑资源、布线资源等结构信息，具有易于掌握和使用、形式与内容分离、解析器开放源代码、良好的扩展性等优点[5]。XML 文件以层次化嵌套的开始标签和结束标签组成，每组标签内有附加说明和可选择的属性值和内容。

FPGA 结构描述文件的架构如图 2 所示，以<architecture>作为根标记，对应一个 FPGA 硬件设计，它包含<device>芯片总体信息、<models>模块信息、<switchlist>开关信息、<segmentlist>布线资源信息、<complexblocklist>逻辑复合块信息、<power>电源信息和<clocks>时钟信号信息[6]。

FPGA 结构描述文件的<complexblocklist>组成如图 3 所示，<complexblocklist>包含若干个<pb_type>逻辑单元块；每个逻辑单元块<pb_type>由 port 和 contents 两部分组成，其中 port 描述外部视图，包含若干个<input>、<output>、<clock>不同类型的管脚，且管脚可以指定 port_class 属性为地址信号(address)、输入数据信号(data_in)、输出数据信号(data_out)、写使能信号(write_en)和时钟信号(clock)；contents 描述内部视图，包含若干个<pb_type>逻辑单元子块和<interconnect>内部连接。<pb_type>逻辑单元块可以指定 class 属性为查找表(lut)、触发器(flipflop)和存储器(memory)。<interconnect>内部连接包含<complete>全互连、<direct>直连、<mux>多路选择连接三种类型。

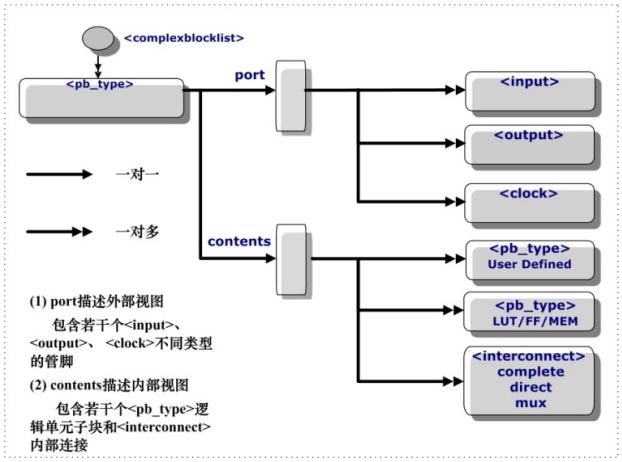

图 3　FPGA 结构描述文件的<complexblocklist>组成图

Fig.3 The <complexblocklist> component diagram of VPR architecture description file

2　可配置逻辑块 CLB 的 FPGA 结构描述文件

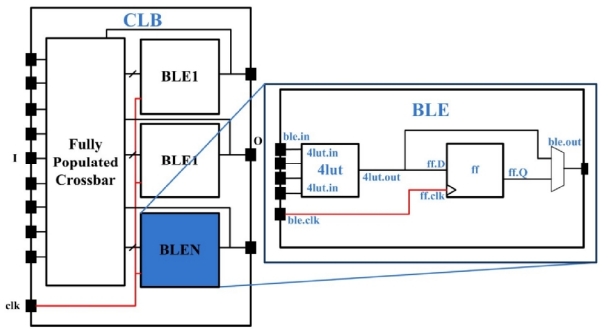

图 4　FPGA 可配置逻辑块 CLB 的模型

Fig.4 The model of FPGA configurable logic block

以图 4 描述的某 FPGA 可配置逻辑块 CLB 为例，其 FPGA 结构描述代码如表 1 所示，构成了<pb_type>逻辑单元块定义的嵌套。在外层的<pb_type>是包含 22 位输入信号 I，10 位输出信号 O，1 位时钟信号 clk 的 CLB；内部包含了 10 个<pb_type name="ble">基本逻辑单元和<interconnect>内部连接；<interconnect>包含了 1 个 32*40 位全互连、1 个 1*10 位全互连和 1 个 10*10 位直连。在内层的<pb_type name="ble">是 4 位输入信号 in，1 位输出信号 out，1 位时钟信号 clk 的 BLE；内部逻辑单元子块包含了 1 个<pb_type name="4lut">4 位查找表、1 个<pb_type name="ff">1 位触发器；<interconnect>内部连接包含了 1 个 1*1 位直连、1 个 4*4 位直连、1 个 2 选 1 多路选择连接和 1 个 1*1 位直连。

表 1 FPGA 可配置逻辑块 CLB 的结构描述代码

Table 1 The architecture description code of FPGA configurable logic block

逻辑模块	逻辑模块内部特性	逻辑模块描述代码
CLB (Configurable Logic Block)	CLB 的管脚特性 22 位输入 10 位输出 1 位时钟信号	<pb_type name="clb"> <input name="clb_I" num_pins="22" equivalent="true"/> <output name="clb_O" num_pins="10" equivalent="true"/> <clock name="clb_clk" equivalent="false"/> <!-- 10 个 BLE--> </pb_type>
	CLB 的内部连接特性 32*40 位全互连 1*10 位全互连 10*10 位直连	<interconnect> <complete output="ble[9:0].ble_in" input="{clb.clb_Ible[9:0].ble_out}" /> <complete output="ble[9:0].ble_clk" input="clb.clb_clk" /> <direct output="clb. clb_O" input="ble[9:0].ble_out" /> </interconnect>
BLE (Basic Logic Element)	BLE 的管脚特性 4 位输入 1 位输出 1 位时钟信号	<pb_type name="ble" num_pb="10"> <input name="ble_in" num_pins="4"/> <output name="ble_out" num_pins="1"/> <clock name="ble_clk"/> <!-- 1 个 ff --> <!-- 1 个 4lut --> </pb_type>
	BLE 的内部连接特性 1*1 位直连 4*4 位直连 2 选 1 多路选择连接 1*1 位直连	<interconnect> <direct output="ff.ff_D" input="4lut.out" /> <direct output="4lut.4lut_in" input="ble.ble_in" /> <mux output="ble.ble_out" input="ff.ff_Q 4lut. 4lut_out" /> <direct output="ff.ff_clk" input="ble.ble_clk" /> </interconnect>
	4 位 LUT	<pb_type name="4lut" blif_model=".names" num_pb="1" class="lut"> <input name="4lut_in" num_pins="4" port_class="lut_in"/> <output name="4lut_out" num_pins="1" port_class="lut_out"/> </pb_type>
	1 位 Flipflop	<pb_type name="ff" blif_model=".latch" num_pb="1" class="flipflop"> <input name="ff_D" num_pins="1" port_class="D"/> <output name="ff_Q" num_pins="1" port_class="Q"/> <clock name="ff_clk" port_class="clock"/> </pb_type>

3 Xilinx 公司 Virtex-6 型号 Slice 的 FPGA 结构描述文件

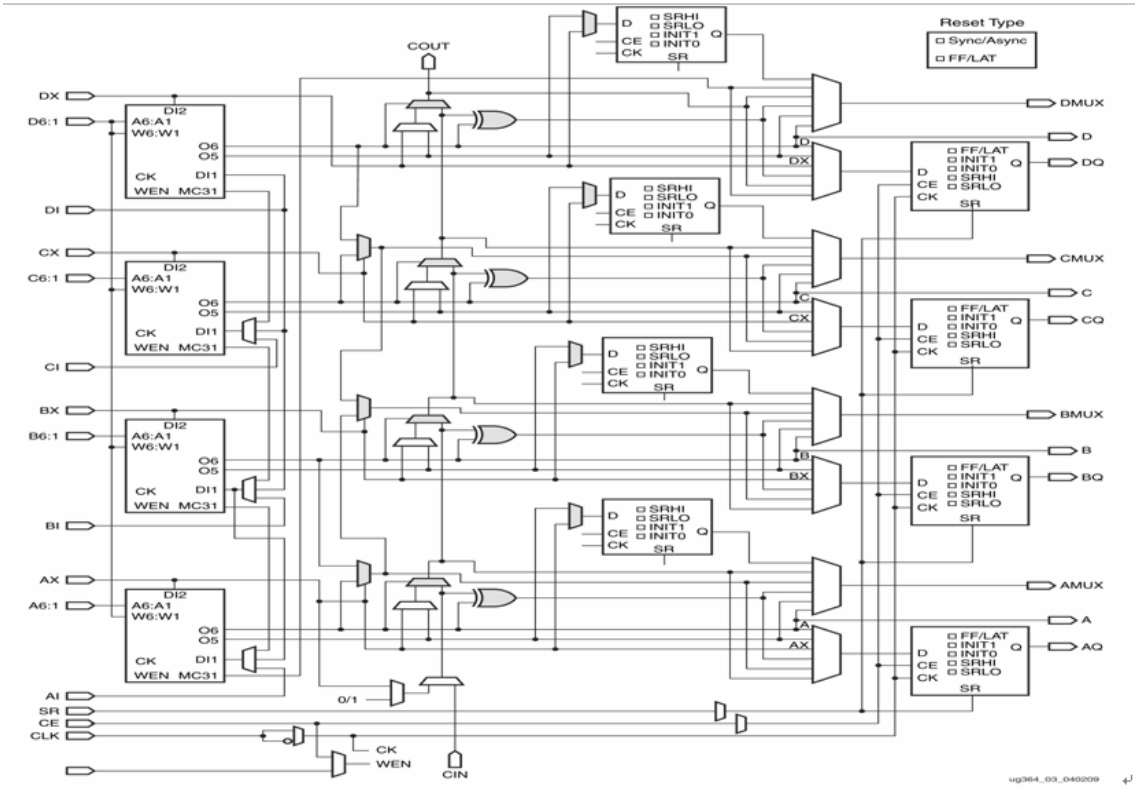

图 5 Xilinx 公司 Virtex-6 型号 slice 的模型[7]

Fig.5 The slice model of Xilinx's Virtex-6[7]

Slice 作为 Xilinx 公司 FPGA 的基本逻辑单位，其 Virtex-6 型号 Slice 的内部结构如图 5 所示。Virtex-6 型号 FPGA 的 Slice 由 4 个六输入的查找表 LUT、4 个大触发器 FF 或锁存器 LATCH、4 个小触发器 FF、多路转换器 MUX、运算进位逻辑 CARRY 构成，其中六输入查找表 LUT 或配置为 64 位 ROM，即具有 1 个六输入的 LUT；或配置为 32 位 ROM，即具有 2 个五输入的 LUT，其不同输出使用相同的地址或逻辑输入。Slice 中的每个 LUT 对应一个触发器，或选择性地配置为锁存器，且其他 4 个触发器保持未使用状态；或选择性地寄存于触发器中。Slice 的算术逻辑包括一个异或门 XORG，用于实现 2bit 全加操作；还包括一个专用与门 MULTAND，用于提高乘法器的效率。Slice 的进位逻辑由函数复用器 MUXC 和专用进位信号组成，用于实现快速的算术加减法操作[7]。

表 2　Xilinx 公司 Virtex-6 型号 slice 的结构描述代码

Table 2 The architecture description code of Xilinx's Virtex-6

逻辑模块内部特性	逻辑模块描述代码
SLICE 的管脚特性 4 组数据输入信号 1 位置位复位信号 SR 1 位使能信号 CE 1 位进位信号 CIN 1 位时钟信号 CLK 4 组数据输出信号 1 位进位信号 COUT	<pb_type name="v6_lslice"> <input name="AX" num_pins="1" equivalent="false"/> <!-- A/AI/BX/B/BI/CX/C/CI/DX/D/DI/SR/CIN --> <input name="CE" num_pins="1" equivalent="false"/> <output name="AMUX" num_pins="1" equivalent="false"/> <!--A/AQ/BMUX/B/BQ/CMUX/C/CQ/DMUX/D/DQ--> <output name="COUT" num_pins="1" equivalent="false"/> <clock name="CLK"/> <pb_type name="lut" num_pb="4" blif_model=".subcktvlut"> <pb_type name="carry" num_pb="4" blif_model=".subckt carry"> <pb_type name="ff_small" num_pb="4" blif_model=".subcktvffs"> <pb_type name="ff_big" num_pb="4" blif_model=".subcktvffb"> </pb_type>
SLICE 内部连接特性 ①3 个 1*8 位全互连 ①2 个 24*24 位直连 ②1 个 4*4 位直连 ③1 个 1*1 位直连 ④4 个 1*1 位直连 ⑤3 个 1*1 位直连 ⑥6 个 1*1 位直连 ⑦3 个 1*1 位直连 ⑧1 个 1*1 位直连 ⑨4 个 1*1 位直连 ⑩4 个 1*1 位直连 ①4 个 2 选 1 多路选择 ②4 个 5 选 1 多路选择 ③4 个 6 选 1 多路选择 ④1 个 3 选 1 多路选择 ⑤1 个 3 选 1 多路选择 ⑤1 个 5 选 1 多路选择 ⑥1 个 2 选 1 多路选择 ⑦4 个 2 选 1 多路选择	<interconnect> <direct name="lutA" input="{v6_lslice.A v6_lslice.B v6_lslice.C v6_lslice.D}" output="lut.A"/> <direct name="lutW" input="{v6_lslice.A v6_lslice.B v6_lslice.C v6_lslice.D}" output="lut.W"/> <direct name="lutDI2" input="{v6_lslice.AX v6_lslice.BX v6_lslice.CX v6_lslice.DX}" output="lut.DI2"/> <direct name="DlutDI1" input="v6_lslice.DI" output="lut[3].DI1"/> <direct name="carryO6" input="lut.O6" output="carry.xor"/> <direct name="carrymuxxor" input="carry[2:0].cmux_out" output="carry[3:1].cmuxxor"/> <direct name="carrymmux" input="{lut[3].O6 lut[2].O6 lut[2].O6 lut[1].O6 lut[1].O6 lut[0].O6}" output="carry[2:0].mmux"/> <direct name="carrymmux_select" input="{v6_lslice.AX v6_lslice.BX v6_lslice.CX}" output="carry[2:0].mmux_select"/> <direct name="cout" input="carry[3].mmux_out" output="v6_lslice.COUT"/> <direct name="ABCD" input="lut[3:0].O6" output="{v6_lslice.D v6_lslice.C v6_lslice.B v6_lslice.A}"/> <direct name="Q" input="ff_big.Q" output="{DQ CQ BQ AQ}"/> <mux name="ff_smallA" input="v6_lslice.AX lut[0].O5" output="ff_small[0].D"/> <mux name="ff_smallB" input="v6_lslice.BX lut[1].O5" output="ff_small[1].D"/> <mux name="ff_smallC" input="v6_lslice.CX lut[2].O5" output="ff_small[2].D"/> <mux name="ff_smallD" input="v6_lslice.DX lut[3].O5" output="ff_small[3].D"/> <mux name="ff_bigA" input="lut[0].O5 lut[0].O6 carry[0].cmux_out carry[0].mmux_out carry[0].xor_out" output="ff_big[0].D"/> <mux name="ff_bigB" input="lut[1].O5 lut[1].O6 carry[1].cmux_out carry[1].mmux_out carry[1].xor_out" output="ff_big[1].D"/> <mux name="ff_bigC" input="lut[2].O5 lut[2].O6 carry[2].cmux_out carry[2].mmux_out carry[2].xor_out" output="ff_big[2].D"/> <mux name="ff_bigD" input="lut[3].O5 lut[3].O6 carry[3].cmux_out carry[3].mmux_out carry[3].xor_out" output="ff_big[3].D"/> <mux name="AMUX" input="lut[0].O5 lut[0].O6 carry[0].cmux_out carry[0].mmux_out carry[0].xor_outff_small[0].Q" output="AMUX"/> <mux name="BMUX" input="lut[1].O5 lut[1].O6 carry[1].cmux_out carry[1].mmux_out carry[1].xor_outff_small[1].Q" output="BMUX"/> <mux name="CMUX" input="lut[2].O5 lut[2].O6 carry[2].cmux_out carry[2].mmux_out carry[2].xor_outff_small[2].Q" output="CMUX"/> <mux name="DMUX" input="lut[3].O5 lut[3].O6 carry[3].cmux_out carry[3].mmux_out carry[3].xor_outff_small[3].Q" output="DMUX"/> <mux name="ClutDI1" input="v6_lslice.CI v6_lslice.DI lut[3].MC31" output="lut[2].DI1"/> <mux name="BlutDI1" input="v6_lslice.BI v6_lslice.DI lut[2].MC31" output="lut[1].DI1"/> <mux name="AlutDI1" input="v6_lslice.AI v6_lslice.BI v6_lslice.DI lut[2].MC31 lut[1].MC31" output="lut[0].DI1"/> <mux name="carrymuxxorA" input="v6_lslice.AX v6_lslice.CIN" output="carry[0].muxxor"/> <mux name="carrymuxA" input="v6_lslice.AX lut[0].O5" output="carry[0].cmux"/> <mux name="carrymuxB" input="v6_lslice.BX lut[1].O5" output="carry[1].cmux"/> <mux name="carrymuxC" input="v6_lslice.CX lut[2].O5" output="carry[2].cmux"/> <mux name="carrymuxD" input="v6_lslice.DX lut[3].O5" output="carry[3].cmux"/> <complete name="clock" input="v6_lslice.CLK" output="{ff_small.CK ff_big.CK}"/> <complete name="ce" input="v6_lslice.CE" output="{ff_small.CEff_big.CE}"/> <complete name="SR" input="v6_lslice.SR" output="{ff_small.SR ff_big.SR}"/> </interconnect>

续表 2 Xilinx 公司 Virtex-6 型号 slice 的结构描述代码

Table 2 The architecture description code of Xilinx's Virtex-6

逻辑模块内部特性	逻辑模块描述代码
4 组 6 位逻辑模块 LUT	<pb_type name="lut" num_pb="4" blif_model=".subcktvlut"> <input name="A" num_pins="5"/> <input name="W" num_pins="5"/> <input name="DI1" num_pins="1"/> <input name="DI2" num_pins="1"/> <output name="MC31" num_pins="1"/> <output name="O6" num_pins="1"/> <output name="O5" num_pins="1"/> </pb_type>
4 组 1 位运算进位逻辑 carry	<pb_type name="carry" num_pb="4" blif_model=".subckt carry"> <input name="xor" num_pins="1"/> <input name="cmuxxor" num_pins="1"/> <input name="cmux" num_pins="1"/> <input name="cmux_select" num_pins="1"/> <input name="mmux" num_pins="2"/> <input name="mmux_select" num_pins="1"/> <output name="xor_out" num_pins="1"/> <output name="cmux_out" num_pins="1"/> <output name="mmux_out" num_pins="1"/> </pb_type>
4 组 1 位小触发器 ff_small	<pb_type name="ff_small" num_pb="4" blif_model=".subcktvffs"> <input name="D" num_pins="1"/> <input name="CE" num_pins="1"/> <input name="SR" num_pins="1"/> <output name="Q" num_pins="1"/> <clock name="CK" num_pins="1"/> </pb_type>
4 组 1 位大触发器 ff_big	<pb_type name="ff_big" num_pb="4" blif_model=".subcktvffb"> <input name="D" num_pins="1"/> <input name="CE" num_pins="1"/> <input name="SR" num_pins="1"/> <output name="Q" num_pins="1"/> <clock name="CK" num_pins="1"/> </pb_type>

以图 5 描述的 Xilinx 公司 Virtex-6 型号 SLICE 为例，其 FPGA 结构描述代码如表 2 所示，构成了<pb_type>逻辑单元块定义的嵌套。在外层的<pb_type>的输入信号包含 4 组信号[1 位 AX，1 位 AI、6 位输入 A]、[1 位 BX，1 位 BI、6 位输入 B]、[1 位 CX，1 位 CI、6 位输入 C]、[1 位 DX，1 位 DI、6 位输入 D] ，1 位进位信号 CIN，1 位置位复位信号 SR，1 位使能信号 CE 和 1 位时钟信号 CLK；输出信号包含 4 组信号[1 位 AMUX，1 位 A、1 位输入 AQ]、[1 位 BMUX，1 位 B、1 位输入 BQ]、[1 位 CMUX，1 位 C、1 位输入 CQ]、[1 位 DMUX，1 位 D、1 位输入 DQ]、1 位进位信号 COUT；内部模块由 4 组 6 位 LUT<pb_type name="lut">、1 位大触发器 FF<pb_type name="ff_big">、1 位小触发器 FF<pb_type name="ff_small">、1 位运算进位逻辑<pb_type name="carry">构成。

<interconnect>全互连的连接如下：

3 个 1*8 位全互连，将置位复位信号 SR，使能信号 CE 和时钟信号 CLK 分别连接 4 个 ff_big[3:0]和 ff_small[3:0].SR/CE/CLK 管脚。

<interconnect>位直接的连接如下：

① 2 个 24*24 位直连，将 6 位输入信号 A/B/C/D 分别连接 lut[3:0].A/W 管脚。

② 1 个 4*4 位直连，将 1 位输入信号 AX/BX/CX/DX 分别连接 lut[3:0].DI2 管脚。

③ 1 个 1*1 位直连，将 1 位输入信号 DI 连接 lut[3].DI2 管脚。

④ 4 个 1*1 位直连，将 lut[3:0].O6 输出信号分别连接 carry[3:0].xor 管脚。

⑤ 3 个 1*1 位直连，将 carry[2:0].cmux_out 输出信号分别连接 carry[3:1].cmuxxor 管脚。

⑥ 6 个 1*1 位直连，将 lut[3:0].O6 输出信号分别连接 carry[2:0].mmux 管脚。

⑦ 3 个 1*1 位直连，将 1 位输入信号 AX/BX/CX 分别连接 carry[2:0].mmux_select 管脚。

⑧ 1 个 1*1 位直连，将 carry[3].mmux_out 输出信号连接到 1 位输出信号 COUT 的管脚。

⑨ 4 个 1*1 位直连，将 lut[3:0].O6 输出信号分别连接到输出信号 A/B/C/D 的管脚。

⑩ 4 个 1*1 位直连，将 ff_big[3:0].Q 输出信号分别连接到输出信号 AQ/BQ/CQ/DQ 的管脚。

<interconnect>多路选择的连接如下：

① 4 个 2 选 1 的多路选择，将输入信号 AX/BX/CX/DX 和 lut[3:0].O5 信号，选择其一分别连接到 ff_small[3:0].D 管脚。

② 4 个 5 选 1 的多路选择，将 lut[3:0].O5/O6 信号和 carry[3:0].cmux_out/mmux_out/xor_out 信号，选择其一分别连接到 ff_big[3:0].D 管脚。

③ 4 个 6 选 1 的多路选择，将 lut[3:0].O5/O6 信号、carry[3:0].cmux_out/mmux_out/xor_out 信号和 ff_small[3:0].D 信号，选择其一分别连接到 AMUX/BMUX/CMUX/DMUX 管脚。

④ 1 个 3 选 1 的多路选择，将 lut[3].MC31 信号、输入信号 CI 和 DI，选择其一连接到 lut[2].DI1 管脚。

⑤ 1 个 3 选 1 的多路选择，将 lut[2].MC31 信号、输入信号 BI 和 DI，选择其一连接到 lut[1].DI1 管脚。

⑥ 1 个 5 选 1 的多路选择，将 lut[2:1].MC31 信号、输入信号 AI、BI 和 DI，选择其一连接到 lut[0].DI1 管脚。

⑦ 1 个 2 选 1 的多路选择，将输入信号 AX 和 CIN，选择其一连接到 carry[0].muxxor 管脚。

⑧ 4 个 2 选 1 的多路选择，将输入信号 AX/BX/CX/DX 和 lut[3:0].O5 信号，选择其一分别连接到 carry[3:0].cmux 管脚。

4 结束语

本文阐述了 VPR 布局布线的工作流程，对 VPR 的 FPGA 结构描述文件进行解析，以正确地表示 FPGA 芯片内部的通道、开关、逻辑资源、布线资源等结构信息。本文以可配置逻辑块 CLB 和 Virtex-6 的 Slice 结构为例，分别给出了相应的 FPGA 结构描述文件，重点讨论了<pb_type>逻辑单元块和<interconnect>内部连接的 XML 代码，特别是查找表(lut)、触发器(flipflop)和存储器(memory)等类型的逻辑单元块<pb_type>，和<complete>全互连、<direct>直连、<mux>多路选择连接等类型的<interconnect>内部连接。今后，我们在 FPGA 结构描述文件正确地被描述和解析的基础上，构造 FPGA 结构在内存物理中的存储格式、进而采用赋权超图描述 FPGA 结构，实现 FPGA 结构设计到赋权超图的转换系统。

参考文献：

[1] ITRS. International Technology Roadmap for Semiconductors (ITRS) Update Overview [EB/OL]. 2015 Edition, Available on the WWW at URL http://www.itrs2.net/.

[2] Betz V, Rose J. VPR: A new packing, placement and routing tool for FPGA research[C]. International Workshop on Field Programmable Logic and Applications. Springer Berlin Heidelberg, 1997: 213-222.

[3] Jason L, Liu T, Vaughn B, Jason A, et al. VPR User's Manual, Version 7.0 [EB/OL].http://www.eecg.utoronto.ca/vpr/vpr.html, 2013.

[4] 胡敏. FPGA 结构描述方法的研究[D]. 上海:复旦大学, 2012.

[5] Betz V, Rose J. Automatic generation of FPGA routing architectures from high-level descriptions[C]. In ACM/ SIGDA International Symposium on Field Programmable Gate Arrays, New York, USA, 2010:175-184.

[6] Luu J, Anderson J H, Rose J S. Architecture Description and Packing for Logic Blocks with Hierarchy, Modes and Complex Interconnect[C]. In ACM/ SIGDA International Symposium on Field Programmable Gate Arrays, California, USA, 2012:227-236.

[7] Xilinx. Xilinx Virtex-6 Family Overview [EB/OL]. http://www.xilinx.com/support/documentation/data_sheets/ds150.pdf,2009.