inspur 浪潮

浪潮 FPGA 加速卡用户手册

F10A

尊敬的浪潮 FPGA 加速卡用户：

衷心感谢您选用浪潮 FPGA 加速卡！

本手册介绍了此款 FPGA 加速卡的技术特性与板卡的安装、使用，有助于您更详细地了解和便捷地使用此款 FPGA 加速卡。

请将我方产品的包装物交废品收购站回收利用，以利于污染预防，造福人类。

本手册中的内容如有变动恕不另行通知。

如果您对本手册有疑问或建议，请向浪潮垂询。

浪潮

2020 年 1 月

版本说明

文档版本：V1.2

日期：2020 年 1 月

文档说明：第三次正式发行

摘 要

手册介绍本 F10A 板卡的规格信息、硬件操作、软件设置、故障诊断等与维护工作密切相关的内容。

本指南认定读者对 FPGA 产品有足够的认识，获得了足够的培训，在操作、维护过程中不会造成个人伤害或产品损坏。

目标受众

本手册主要适用于以下人员：

- FPGA 工程师
- 嵌入式软件工程师

建议由具备 FPGA 知识的专业工程师参考本手册进行 F10A 板卡使用、开发。

目录

1. 安全说明

 警告：以下警告表示存在可能导致财产损失、人身伤害或死亡的潜在危险。

1 请将设备连接到适当的电源，仅可使用额定输入标签上指明的 PCIe 插槽为设备供电，为保护您的设备免受供电不匹配所导致的损坏，F10A 加速卡遵循 PCIe CEM v3.0 规范，因此需要安装于 75W 的 PCIe 插槽，推荐使用 x16 插槽，切勿使用 x24 插槽。

2 请务必使用随机配备的组件，为保证设备及使用者的安全，不要随意更换。

3 为防止系统漏电造成电击危险，务必将系统和外围设备的电源电缆插入已正确接地的电源插座。

4 切勿将任何物体塞入系统的开孔处。如果塞入物体，可能会导致内部组件短路而引起火灾或电击。

5 切勿让食物或液体散落在系统内部或其它组件上，不要在高潮湿、高灰尘的环境中使用产品。

6 在服务器中安装设备之前，请先检查服务器是否工作正常。

 注意： 为了您更好地使用设备，以下注意事项将帮助您避免可能会损坏部件或导致数据丢失等问题的出现：

1 如果出现以下任何情况，请将 FPGA 加速卡从 PCIe 插槽拔下，并与浪潮客户服务部门联系：

1）电源电缆、延长电缆或电源插头已损坏。

2）产品被水淋湿。

3）产品跌落或损坏。

4）物体落入产品内部。

5）按照操作说明进行操作时，产品不能正常工作。

2 如果板卡和系统受潮，请按以下步骤处置：

1）关闭系统和设备电源，断开它们与电源插座的连接，等待 10 至 20 秒钟，然后取下 FPGA 加速卡。

2）将设备移至通风处，使系统至少干燥 24 小时，并确保系统完全干燥。

3）将板卡重新插入 PCIe 插槽，然后开机。

4）如果运行失败或异常，请与浪潮联系，获得技术帮助。

4 取下板卡或接触内部组件之前，应先让设备冷却。

5 为防止静电释放损坏设备内部的电子组件，请注意以下事项：

1）拆装、接触设备内任何电子组件前应先导去身上的静电。您可通过触摸金属接地物（如机箱上未上漆的金属表面）释放身体上的静电，以防止人体静电对敏感组件的损害。

2）对不准备安装使用的静电敏感组件，请不要将其从防静电包装材料中取出。

3）工作中请定期触摸接地导体或机箱上未上漆的金属表面，以便释放身体上可能损坏内部组件的静电。

6 经浪潮同意，拆装系统内部组件时，请注意以下事项：

1）关闭系统电源并断开电缆，包括断开系统的任何连接。

2）取下板卡或接触内部组件之前，应先让产品冷却。

3）拆装、接触设备内任何电子组件前应先通过触摸金属接地物体导去身上的静电。

4）拆装过程中动作幅度不宜过大，以免损坏组件或划伤手臂。

5）拿取插卡或组件时，应抓住板卡的边缘或其金属固定支架。

2. 产品规格介绍

2.1产品概述

FPGA 即现场可编程门阵列，它是可编程器件的基础上进一步发展的产物，作为专用集成电路（ASIC）领域中的一种半定制电路而出现，既解决了定制电路的不足，又克服了原有可编程器件门电路数有限的缺点。传统 FPGA 开发采用 Verilog/VHDL 语言进行硬件编程、编程极度复杂，Altera 率先推出对高级语言 OpenCL 的支持的 FPGA 芯片，为 FPGA 实现更大规模使用提供了基础。随着 Intel 对于 Altera 的收购，对于 FPGA 的生态建设将产生极大正面推动作用。

F10A 是 Inspur 浪潮基于 Intel 最新的 Arria10 系列 FPGA 技术开发的异构计算加速卡，F10A 卡对高性能计算、机器学习、DNN 语音识别、在线识别、CNN 图像识别、Bing 搜索、数据压缩、大数据处理等领域都有非常重要的意义，可覆盖计算加速、存储加速、网络加速等技术。

F10A 卡外观正面如图 2-1 所示。

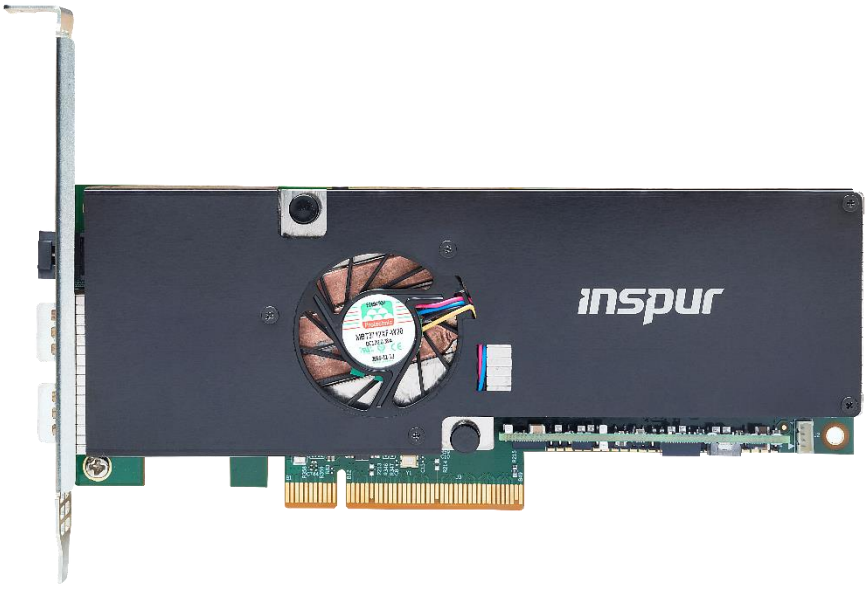

图 2-1 F10A 外观正面图

F10A 系列加速卡包含多个型号，如表 2-1 所示。

表 2-1 F10A 系列加速卡

描述	结构及散热方式	散热要求	出货部件编号
F10A A10GX1150 16GB Active FPGA Board	单 PCIE 槽位主动散热 FPGA 加速卡 (TDP 66W)	Inspur F10A Arria 10 GX1150(1.366 Tflops) 16GB PCIe x8 Active Cooling 1_Slot FPGA Board	SPT0POC00F01
		FPGA 板卡单独出货（单卡使用环境要求：卡的进风温度≤50℃，且进风风速≥1m/s）	
F10A A10GX1150 16GB Passive FPGA Board	双 PCIE 槽位被动散热 FPGA 加速卡 (TDP 66W)	Inspur F10A Arria 10 GX1150(1.366 Tflops) 16GB with 2 SFP+ GE/10GE 接口 PCIe x8 Passive Cooling 1_Slot FPGA Board	SPT0POC00F02
		FPGA 板卡单独出货（单卡使用环境要求：卡的进风温度≤50℃，且进风风速≥1m/s）	

产品功能

- 硬件低耦合

F10A 是标准的半高半长占用一个标准 PCIe 槽位，服务器适配性更强，无需额外机架空间，无需额外结构支持、无需额外散热支持、无需外接电源，对服务器的适应性强，安装简便，易于大规模部署。

- 软件无感知

通过更新系统函数库 Library，或者使用 CAPI（ Coherence Attach Processor Interface）提供的内存访问一致性特性，应用软件只需要调用新函数或启用 CAPI 功能就能实现对原有算法的加速，软件架构无改动。

- 高性能低功耗

相比于软件加速方案，F10A 的加速性能提升数倍到数百倍；F10A 的功耗仅为 30W~45W，每瓦功耗获得的性能优势明显，使得算法应用性价比、性能功耗比相较传统服务器 CPU 有量级倍数的提升。

- 在线升级支持

F10A 系列加速卡支持 PCI Express 在线动态重构加速算法，根据不同应用场景动态加载算法逻辑而无需重启服务器。用户可使用此应用灵活地加载更新不同的加速算法。

- Altera SDK 开发环境

F10A 加速卡支持 Intel SDK 开发环境，支持 OpenCL™开发语言，是真正意义上的 SDA(Software Defined Accelerator)加速器。

- 支持对第三方算法

Inspur 浪潮提供免费的参考设计和参考 IP，供学习和熟悉板卡应用。

2.2参数规格

F10A 卡参数规格如下表 2-2 所示：

表 2-2 F10A 参数规格表

规格		参数
产品		F10A-1150
物理参数	FPGA	10AX115H3F34E2SG
	PCIE	PCI Express Gen3 x8
	内存	两组 DDR4 SODIMM 内存，每组 8GB，位宽 64bit,最大速率 2133Mbps
	Flash	支持 2 片 1Gb@16Bit NOR FLASH 芯片数据级联，用于 FPGA 的 FPP 配置。型号：MT28GU01GAAA1EGC-0SIT 支持 1 片 1Gb QSPI 串行 Norflash，用于 FPGA 的 AS 配置。型号：MT25QU01GBBB8E12-0SIT
	Ethernet 接口	支持 2 路 SFP+ 10GE 光口 （仅 SPT0POC00F02 支持）
	外形（L*D*H）	167mmx21.5mmx68mm
电源参数	电源	整板由 PCIe 插槽供电，无需外接电源，符合 PCIe 标准
	功耗	平均功耗 45W 最大功耗 66W
	散热	在散热器上增加风扇进行主动散热。最大可支持 FPGA 芯片 66W 的散热能力 （仅 SPT0POC00F01 支持单槽位主动散热；仅 SPT0POC00F02 支持双槽位被动散热）
环境要求及标准	工作温度	0°C～50°C
	储藏温度	-40°C～ +70°C

	湿度	93±3%RH
	认证标准	符合EMC国际电磁兼容性标准FCC Class A规范
其他	板卡尺寸	采用PCIe标准半高半长规格：167mm（长） x 68.9mm（高）x 20.32mm（厚）
	加载升级	支持通过Jtag对Flash中的FPGA版本进行在线升级 支持通过pcie对Flash中的FPGA版本进行在线升级

3. 板卡操作

3.1工具介绍

- 安装工具

防静电手套

螺丝刀

- FPGA 烧写工具

USB Blaster II

转接线

- 前提条件

购买 F10A 加速卡之前，请确认服务器 PCIe 接口类型及服务器内尺寸。

PCIE 接口需求：F10A 加速卡使用的 PCIe 模块适用于 PCIe Gen3 x8 的 PCIE 插槽接口。

电源需求：F10A 加速卡遵循 PCIe CEM v3.0 规范，因此需要安装于 75W 的插槽。加速卡采用低功耗设计，无需额外的供电；面板上的红色 LED 灯指示了供电电源是否满足需求，亮则电源满足，灭则不满足。

为了满足 F10A 在 PCIE3.0x4/x8 插槽正常工作，基于《PCI Express® Card Electromechanical Specification Revision 3.0》规定，需要配置主机支持 PCIE 插槽为

高功率卡（high power）模式，最高支持 75W。

Note: 禁止把板卡插在 x24 的 PCI-E 插槽

3.2板卡操作

- 安装板卡步骤

步骤 1 戴好防静电手套。

步骤 2 断开服务器电源，并拔下电源插头，将服务器机箱接地。

步骤 3 依照服务器相关使用说明将服务器机箱打开，将 F10A 插入服务器的 PCIe X8 或者 X16 插槽中。

步骤 4 插好电源插头，给服务器上电。

步骤 5 待服务器启动后，观察 F10A 的电源指示灯。

1) 若常亮，则表示电源正常、板卡运行正常。

2) 若不亮，则表示电源或板卡运行异常。请将服务器安全掉电并且取出板卡，并联系我司技术支持。

步骤 6 执行命令***$ lspci | grep Process*** 查看系统是否检测到板卡。

1) 若显示 Processing accelerations: Inspur Electronic Information Industry Co. , Ltd. Device 2494 (rev 04)，则说明已检测到板卡；

2) 若无显示，则说明板卡损坏或者板卡 Flash 中没有工程，请将服务器安全掉电并且取出板卡，并联系我司技术支持。；

- 拆卸板卡

步骤 1 戴好防静电手套。

步骤 2 关闭机箱电源。

步骤 3 打开机箱盖，两手分别位于加速卡两端，沿垂直 PCIE 卡槽方向慢慢拔出加速。

步骤 4 将拔出的 F10A 加速卡放入静电包装袋。

步骤 5 盖好机箱盖。

- 指示灯说明

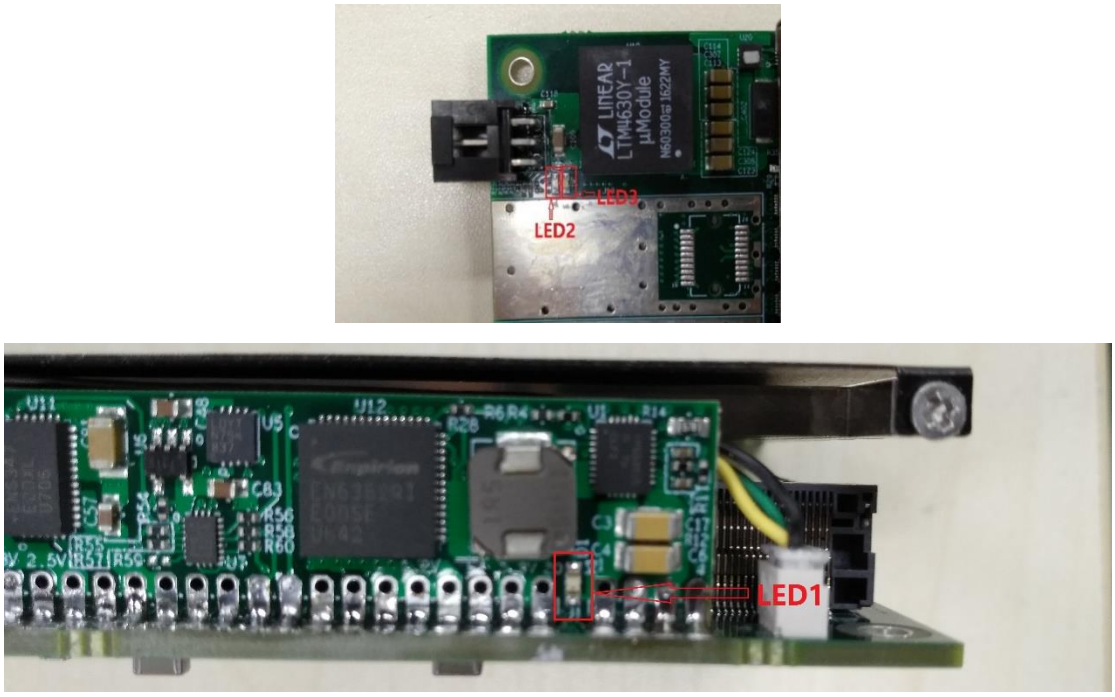

图 3-1 F10A 指示灯标示图

指示灯说明：

LED1：绿灯，电源指示灯

LED2：绿灯，若亮，指示两片 DDR 中的下方 DDR_A 正常工作

LED3：绿灯，若亮，指示两片 DDR 中的上方 DDR_B 正常工作

图中所示的 LED 指示灯都是受 FPGA 控制的，可根据需求设计使用。

4. 板卡硬件电路介绍

4.1板卡系统架构

FPGA 系统框图如下 4-1 所示：

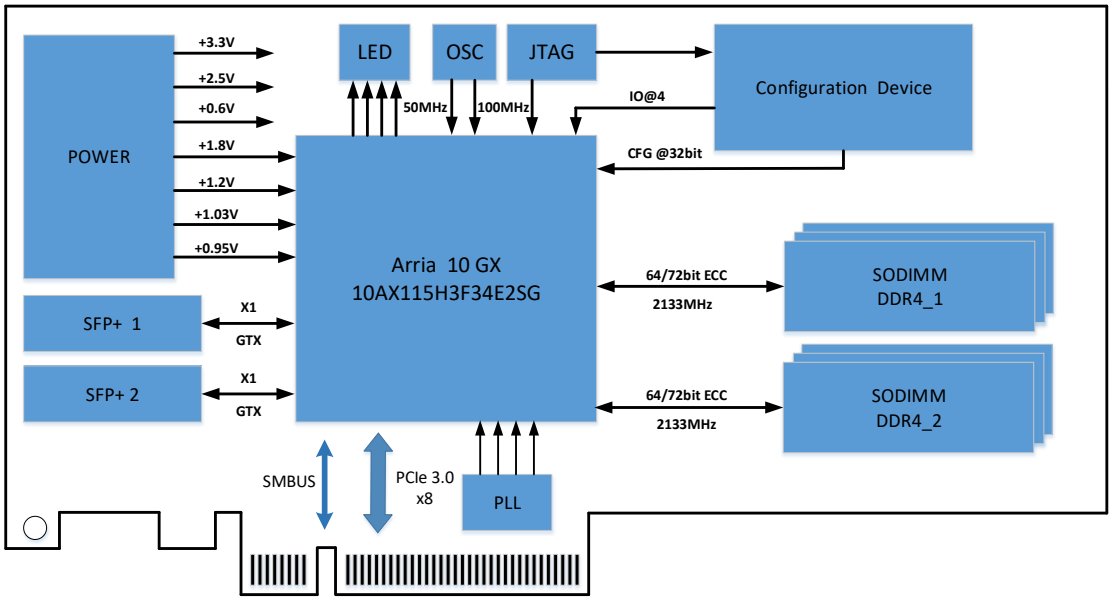

图 4-1 FPGA 系统框图

其主要组成部分：

— 支持 1 颗 Altera Arria 10 GX FPGA 芯片，型号 10AX115H3F34E2SG。

— 支持 1 颗 Altera MAX 10 CPLD 芯片，型号 10M02SCU169，主要用于 FPGA 的并行加载及时钟 PLL 的 I2C 控制等功能。

— 支持 2 个 SODIMM DDR4 内存通道。目前支持的 DDR4 SODIMM 如下：

Part Number	容量	位宽	数据速率	ECC	RANK
Micron					
MTA4ATF51264HZ-2G3B1	4GB	x64	2133MT/s	No	Single
MTA8ATF1G64HZ-2G3B1	8GB	x64	2133MT/s	No	Single
MTA18ASF1G72HZ-2G3B1	8GB	x72	1866MT/s	Yes	Dual
MTA18ASF2G72HZ-2G3B1	16GB	x72	1866MT/s	Yes	Dual
SAMSUNG					
M471A1K43BB0-CPB	8GB	x64	2133MT/s	No	Single

— 支持 PCIe Gen3.0，x8 Lane，访问带宽 128Gbps。

— 支持两路 SFP+，每路速率 10.3125Gb/s。

— 支持 2 片 1Gb@16Bit NOR FLASH 芯片数据级联，用于 FPGA 的 FPP 配置。

型号：MT28GU01GAAA1EGC-0SIT

— 支持 1 片 1Gb QSPI 串行 Norflash，用于 FPGA 的 AS 配置。

型号：MT25QU01GBBB8E12-0SIT

— 支持 JTAG 调试接口。

— 支持 1 片板载 512Kb EEPROM，用于存储板卡信息。

型号：M24512-DFMN6TP

— 支持通过 CPLD 读取 FPGA 芯片温度、板卡温度、风扇转速和整板功耗。

— 支持通过 CPLD 进行风扇调速(V4.2 及以上版本)。

4.2FPGA 电路介绍

4.2.1 时钟电路

FPGA 时钟包含以下几种电路：

1、 1 路单端 50MHz 晶振。

2、 1 路 LVDS 差分 100MHz 晶振，用于 Kernal 运行时钟。

3、 1 个可配置 PLL，可产生 4 路 LVDS 时钟，其中 2 路用于 DDR4 的参考时钟，另外 2 路用于 10G 光网络接口的参考时钟。

4、 1 个差分 100MHz 晶振，用于 PCIe 的参考时钟（预留），通常使用 PCIe 提供参考时钟。

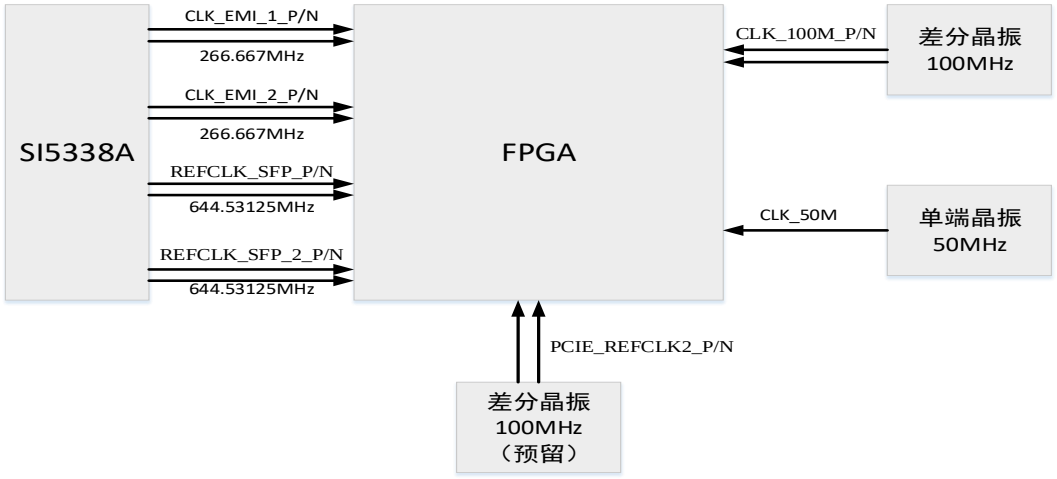

图 4-2 时钟框图

时钟对应 FPGA 管脚定义：

信号名称	频率	电平标准	管脚	说明
CLK_50M	50MHz	1.8V	AJ9	IO 时钟
CLK_100M_P	100MHz	1.8V LVDS	AD10	Kernal 时钟
CLK_100M_N			AD11	
CLK_EMI_2_P	266.667MHz	1.8V LVDS	F6	DDR4_1 参考时钟
CLK_EMI_2_N			F5	
CLK_EMI_1_P	266.667MHz	1.8V LVDS	E23	DDR4_2 参考时钟
CLK_EMI_1_N			E24	
REFCLK_SFP_P	644.53125MHz	1.8V LVDS	P28	SFP+ 1 参考时钟
REFCLK_SFP_N			P27	
REFCLK_SFP_2_P	644.53125MHz	1.8V LVDS	V28	SFP+ 2 参考时钟
REFCLK_SFP_2_N			V27	
PCIE_REFCLK2_P	100MHz	1.8V LVDS	Y28	预留，作为 PCIe 的参考时钟
PCIE_REFCLK2_N		1.8V LVDS	Y27	

4.2.2 PCIe 电路

PCIe 接口电路支持 8 Lane PCIe Gen3.0，支持最高速率 64Gbit/s。

注意：A10 FPGA ES2 系列，支持到 PCIe Gen3.0 时，需要进行 PCIe 完整约束，否则只能支持到 PCIe Gen2.0，详见：https://www.altera.com/support/support-resources/knowledge-base/ip/2017/what-assignments-do-i-need-for-a-pcie-gen1--gen2-or-gen3-design-.html

PCIe 对应 FPGA 管脚定义：

信号名称	电平标准	管脚	说明
PCIE_RX_P0	Current Mode Logic(CML)	AE30	接收 Lane 0
PCIE_RX_N0	Current Mode Logic(CML)	AE29	
PCIE_RX_P1	Current Mode Logic(CML)	AD32	接收 Lane 1
PCIE_RX_N1	Current Mode Logic(CML)	AD31	
PCIE_RX_P2	Current Mode Logic(CML)	AC30	接收 Lane 2

PCIE_RX_N2	Current Mode Logic(CML)	AC29	
PCIE_RX_P3	Current Mode Logic(CML)	AB32	接收 Lane 3
PCIE_RX_N3	Current Mode Logic(CML)	AB31	
PCIE_RX_P4	Current Mode Logic(CML)	AA30	接收 Lane 4
PCIE_RX_N4	Current Mode Logic(CML)	AA29	
PCIE_RX_P5	Current Mode Logic(CML)	Y32	接收 Lane 5
PCIE_RX_N5	Current Mode Logic(CML)	Y31	
PCIE_RX_P6	Current Mode Logic(CML)	W30	接收 Lane 6
PCIE_RX_N6	Current Mode Logic(CML)	W29	
PCIE_RX_P7	Current Mode Logic(CML)	V32	接收 Lane 7
PCIE_RX_N7	Current Mode Logic(CML)	V31	
PCIE_TX_P0	High Speed Differential I/O	AN34	发送 Lane 0
PCIE_TX_N0	High Speed Differential I/O	AN33	
PCIE_TX_P1	High Speed Differential I/O	AL34	发送 Lane 1
PCIE_TX_N1	High Speed Differential I/O	AL33	
PCIE_TX_P2	High Speed Differential I/O	AJ34	发送 Lane 2
PCIE_TX_N2	High Speed Differential I/O	AJ33	
PCIE_TX_P3	High Speed Differential I/O	AG34	发送 Lane 3
PCIE_TX_N3	High Speed Differential I/O	AG33	
PCIE_TX_P4	High Speed Differential I/O	AE34	发送 Lane 4
PCIE_TX_N4	High Speed Differential I/O	AE33	
PCIE_TX_P5	High Speed Differential I/O	AC34	发送 Lane 5
PCIE_TX_N5	High Speed Differential I/O	AC33	
PCIE_TX_P6	High Speed Differential I/O	AA34	发送 Lane 6
PCIE_TX_N6	High Speed Differential I/O	AA33	
PCIE_TX_P7	High Speed Differential I/O	W34	发送 Lane 7
PCIE_TX_N7	High Speed Differential I/O	W33	
PCIE_EDGE_REFCLK_P	HCSL	AD28	母板参考时钟
PCIE_EDGE_REFCLK_N	HCSL	AD27	
PCIE_REFCLK2_P	HCSL	Y28	预留
PCIE_REFCLK2_N	HCSL	Y27	
PCIE_PERSTn	1.8V	AE16	复位信号
PCIE_WAKEN	1.8V	AL18	唤醒信号
PCIE_SMBCLK	1.8V	AL4	SBUS 时钟
PCIE_SMBDAT	1.8V	AL5	SBUS 数据

4.2.3 SFP+ 电路

支持两路 SFP+电路，单路数据传输速率达到 10.3125Gb/s。

SFP+对应 FPGA 管脚定义：

信号名称	电平标准	管脚	说明
SFP+ 1			
SFP_RX_P	PCML	L30	接收通道
SFP_RX_N	PCML	L29	
SFP_TX_P	PCML	E34	发送通道
SFP_TX_N	PCML	E33	
REFCLK_SFP_P	1.8V LVDS	P28	参考时钟
REFCLK_SFP_N	1.8V LVDS	P27	
SFP_RS0	1.8V	AD1	Rate 选择 0
SFP_RS1	1.8V	AD2	Rate 选择 1
SFP_TX_DISABLE	1.8V	AE4	发送 Disable
SFP_RX_LOS	1.8V	AB7	接收指示
SFP_TX_FAULT	1.8V	AB8	发送错误指示
SFP_MOD0_PRSNTn	1.8V	AB5	复位信号
SFP_MOD1_SCL	1.8V	AE9	I2C 时钟
SFP_MOD2_SDA	1.8V	AF9	I2C 数据
SFP+ 2			
SFP_2_RX_P	PCML	U30	接收通道
SFP_2_RX_N	PCML	U29	
SFP_2_TX_P	PCML	U34	发送通道
SFP_2_TX_N	PCML	U33	
REFCLK_2_SFP_P	1.8V LVDS	V28	参考时钟，默认值：644.53125MHz
REFCLK_2_SFP_N	1.8V LVDS	V27	
SFP_2_RS0	1.8V	AH9	Rate 选择 0
SFP_2_RS1	1.8V	AH10	Rate 选择 1
SFP_2_TX_DISABLE	1.8V	AF10	发送 Disable
SFP_2_RX_LOS	1.8V	AG10	接收指示
SFP_2_TX_FAULT	1.8V	AG11	发送错误指示
SFP_2_MOD0_PRSNTn	1.8V	AF11	复位信号
SFP_2_MOD1_SCL	1.8V	AE8	I2C 时钟
SFP_2_MOD2_SDA	1.8V	AF8	I2C 数据

4.2.4 SODIMM DDR4 电路

支持 2 个 SODIMM DDR4 内存通道，每个通道最大支持到 16GB@1860MHz，8GB@2133MHz，4GB@2133MHz。

DDR4 对应 FPGA 管脚定义：

信号名称	电平标准	管脚	说明
DDR4_1			
DDR4_1_RAS_N/A16	SSTL_12	E6	行选通
DDR4_1_CAS_N/A15	SSTL_12	E7	列选通
DDR4_1_WE_N/A14	SSTL_12	D2	写信号
DDR4_1_A13	SSTL_12	E2	地址信号
DDR4_1_A12	SSTL_12	E3	
DDR4_1_A11	SSTL_12	G6	
DDR4_1_A10	SSTL_12	G5	
DDR4_1_A9	SSTL_12	H2	
DDR4_1_A8	SSTL_12	H3	
DDR4_1_A7	SSTL_12	D1	
DDR4_1_A6	SSTL_12	E1	
DDR4_1_A5	SSTL_12	G3	
DDR4_1_A4	SSTL_12	G2	
DDR4_1_A3	SSTL_12	H4	
DDR4_1_A2	SSTL_12	H5	
DDR4_1_A1	SSTL_12	F1	
DDR4_1_A0	SSTL_12	G1	
DDR4_1_ACT_N	SSTL_12	L11	地址命令信号
DDR4_1_ALERT_N	SSTL_12	H13	报警信号
DDR4_1_BA1	SSTL_12	C3	BANK 选择
DDR4_1_BA0	SSTL_12	E4	
DDR4_1_BG1	SSTL_12	N10	GROUP 选择
DDR4_1_BG0	SSTL_12	B3	
DDR4_1_CK1_T	Differential 1.2-V SSTL	K11	时钟 1
DDR4_1_CK1_C	Differential 1.2-V SSTL	J11	
DDR4_1_CK0_T	Differential 1.2-V SSTL	L9	时钟 0
DDR4_1_CK0_C	Differential 1.2-V SSTL	L10	
DDR4_1_CKE1	SSTL_12	H10	时钟 1 使能
DDR4_1_CKE0	SSTL_12	J6	时钟 0 使能
DDR4_1_CS1_N	SSTL_12	H8	片选 1
DDR4_1_CS0_N	SSTL_12	M11	片选 0
DDR4_1_ODT1	SSTL_12	J10	端接 1 信号
DDR4_1_ODT0	SSTL_12	K8	端接 0 信号
DDR4_1_PARITY	SSTL_12	J9	极性控制
DDR4_1_RESET_N	SSTL_12	M10	复位
DDR4_1_RZQIN	SSTL_12	V2	RZQ
DDR4_1_DM8_N	1.2-V POD	T3	数据 MASK
DDR4_1_DM7_N	1.2-V POD	M6	
DDR4_1_DM6_N	1.2-V POD	N9	
DDR4_1_DM5_N	1.2-V POD	V9	

DDR4_1_DM4_N	1.2-V POD	M2	
DDR4_1_DM3_N	1.2-V POD	E11	
DDR4_1_DM2_N	1.2-V POD	A6	
DDR4_1_DM1_N	1.2-V POD	A11	
DDR4_1_DM0_N	1.2-V POD	H12	
DDR4_1_CB7	1.2-V POD	R3	ECC 数据
DDR4_1_CB6	1.2-V POD	P4	
DDR4_1_CB5	1.2-V POD	T4	
DDR4_1_CB4	1.2-V POD	P5	
DDR4_1_CB3	1.2-V POD	V5	
DDR4_1_CB2	1.2-V POD	U5	
DDR4_1_CB1	1.2-V POD	U6	
DDR4_1_CB0	1.2-V POD	R4	
DDR4_1_DQ63	1.2-V POD	L4	数据信号
DDR4_1_DQ62	1.2-V POD	L6	
DDR4_1_DQ61	1.2-V POD	N5	
DDR4_1_DQ60	1.2-V POD	M5	
DDR4_1_DQ59	1.2-V POD	N4	
DDR4_1_DQ58	1.2-V POD	K4	
DDR4_1_DQ57	1.2-V POD	L5	
DDR4_1_DQ56	1.2-V POD	M7	
DDR4_1_DQ55	1.2-V POD	R8	
DDR4_1_DQ54	1.2-V POD	R6	
DDR4_1_DQ53	1.2-V POD	N7	
DDR4_1_DQ52	1.2-V POD	L8	
DDR4_1_DQ51	1.2-V POD	P9	
DDR4_1_DQ50	1.2-V POD	R7	
DDR4_1_DQ49	1.2-V POD	K7	
DDR4_1_DQ48	1.2-V POD	P7	
DDR4_1_DQ47	1.2-V POD	V7	
DDR4_1_DQ46	1.2-V POD	W9	
DDR4_1_DQ45	1.2-V POD	U7	
DDR4_1_DQ44	1.2-V POD	T9	
DDR4_1_DQ43	1.2-V POD	V8	
DDR4_1_DQ42	1.2-V POD	W10	
DDR4_1_DQ41	1.2-V POD	T8	
DDR4_1_DQ40	1.2-V POD	R9	
DDR4_1_DQ39	1.2-V POD	M1	
DDR4_1_DQ38	1.2-V POD	L3	
DDR4_1_DQ37	1.2-V POD	K3	
DDR4_1_DQ36	1.2-V POD	K1	
DDR4_1_DQ35	1.2-V POD	N3	

DDR4_1_DQ34	1.2-V POD	M3	
DDR4_1_DQ33	1.2-V POD	K2	
DDR4_1_DQ32	1.2-V POD	L1	
DDR4_1_DQ31	1.2-V POD	C10	
DDR4_1_DQ30	1.2-V POD	E12	
DDR4_1_DQ29	1.2-V POD	C13	
DDR4_1_DQ28	1.2-V POD	B10	
DDR4_1_DQ27	1.2-V POD	C12	
DDR4_1_DQ26	1.2-V POD	F11	
DDR4_1_DQ25	1.2-V POD	D12	
DDR4_1_DQ24	1.2-V POD	D10	
DDR4_1_DQ23	1.2-V POD	D5	
DDR4_1_DQ22	1.2-V POD	B6	
DDR4_1_DQ21	1.2-V POD	C5	
DDR4_1_DQ20	1.2-V POD	B5	
DDR4_1_DQ19	1.2-V POD	D4	
DDR4_1_DQ18	1.2-V POD	C4	
DDR4_1_DQ17	1.2-V POD	D7	
DDR4_1_DQ16	1.2-V POD	A5	
DDR4_1_DQ15	1.2-V POD	A10	
DDR4_1_DQ14	1.2-V POD	B7	
DDR4_1_DQ13	1.2-V POD	D9	
DDR4_1_DQ12	1.2-V POD	B8	
DDR4_1_DQ11	1.2-V POD	A9	
DDR4_1_DQ10	1.2-V POD	C8	
DDR4_1_DQ9	1.2-V POD	C9	
DDR4_1_DQ8	1.2-V POD	C7	
DDR4_1_DQ7	1.2-V POD	G13	
DDR4_1_DQ6	1.2-V POD	J12	
DDR4_1_DQ5	1.2-V POD	F13	
DDR4_1_DQ4	1.2-V POD	G12	
DDR4_1_DQ3	1.2-V POD	M13	
DDR4_1_DQ2	1.2-V POD	L13	
DDR4_1_DQ1	1.2-V POD	E13	
DDR4_1_DQ0	1.2-V POD	G11	
DDR4_1_DQS8_T	Differential 1.2-V POD	T5	数据 Probe 信号
DDR4_1_DQS8_C	Differential 1.2-V POD	T6	
DDR4_1_DQS7_T	Differential 1.2-V POD	J4	
DDR4_1_DQS7_C	Differential 1.2-V POD	J5	
DDR4_1_DQS6_T	Differential 1.2-V POD	N8	
DDR4_1_DQS6_C	Differential 1.2-V POD	M8	
DDR4_1_DQS5_T	Differential 1.2-V POD	U10	

DDR4_1_DQS5_C	Differential 1.2-V POD	T10	
DDR4_1_DQS4_T	Differential 1.2-V POD	J1	
DDR4_1_DQS4_C	Differential 1.2-V POD	J2	
DDR4_1_DQS3_T	Differential 1.2-V POD	B11	
DDR4_1_DQS3_C	Differential 1.2-V POD	B12	
DDR4_1_DQS2_T	Differential 1.2-V POD	A3	
DDR4_1_DQS2_C	Differential 1.2-V POD	A4	
DDR4_1_DQS1_T	Differential 1.2-V POD	E8	
DDR4_1_DQS1_C	Differential 1.2-V POD	E9	
DDR4_1_DQS0_T	Differential 1.2-V POD	K12	
DDR4_1_DQS0_C	Differential 1.2-V POD	K13	
CLK_EMI_2_P	LVDS	F6	参考时钟，默认值：266.667MHz
CLK_EMI_2_N	LVDS	F5	
DDR4_2			
DDR4_2_RAS_N/A16	SSTL_12	F23	行选通
DDR4_2_CAS_N/A15	SSTL_12	E22	列选通
DDR4_2_WE_N/A14	SSTL_12	D27	写信号
DDR4_2_A13	SSTL_12	E27	地址信号
DDR4_2_A12	SSTL_12	F24	
DDR4_2_A11	SSTL_12	F26	
DDR4_2_A10	SSTL_12	G26	
DDR4_2_A9	SSTL_12	J22	
DDR4_2_A8	SSTL_12	H22	
DDR4_2_A7	SSTL_12	H23	
DDR4_2_A6	SSTL_12	H24	
DDR4_2_A5	SSTL_12	G25	
DDR4_2_A4	SSTL_12	H25	
DDR4_2_A3	SSTL_12	G22	
DDR4_2_A2	SSTL_12	G23	
DDR4_2_A1	SSTL_12	G27	
DDR4_2_A0	SSTL_12	H27	
DDR4_2_ACT_N	SSTL_12	J24	地址命令信号
DDR4_2_ALERT_N	SSTL_12	B20	报警信号
DDR4_2_BA1	SSTL_12	D25	BANK 选择
DDR4_2_BA0	SSTL_12	D26	
DDR4_2_BG1	SSTL_12	M24	GROUP 选择
DDR4_2_BG0	SSTL_12	C25	
DDR4_2_CK1_T	Differential 1.2-V SSTL	B23	时钟 1
DDR4_2_CK1_C	Differential 1.2-V SSTL	C23	
DDR4_2_CK0_T	Differential 1.2-V SSTL	L23	时钟 0
DDR4_2_CK0_C	Differential 1.2-V SSTL	M23	

DDR4_2_CKE1	SSTL_12	D24	时钟 1 使能
DDR4_2_CKE0	SSTL_12	J27	时钟 0 使能
DDR4_2_CS1_N	SSTL_12	A25	片选 1
DDR4_2_CS0_N	SSTL_12	K24	片选 0
DDR4_2_ODT1	SSTL_12	C24	端接 1 信号
DDR4_2_ODT0	SSTL_12	K25	端接 0 信号
DDR4_2_PARITY	SSTL_12	K22	极性控制
DDR4_2_RESET_N	1.2	L24	复位
DDR4_2_RZQIN	1.2	AL20	RZQ
DDR4_2_DM8_N	1.2-V POD	AF20	数据 MASK
DDR4_2_DM7_N	1.2-V POD	AN23	
DDR4_2_DM6_N	1.2-V POD	AN27	
DDR4_2_DM5_N	1.2-V POD	AH26	
DDR4_2_DM4_N	1.2-V POD	AE23	
DDR4_2_DM3_N	1.2-V POD	F20	
DDR4_2_DM2_N	1.2-V POD	K18	
DDR4_2_DM1_N	1.2-V POD	H19	
DDR4_2_DM0_N	1.2-V POD	C19	
DDR4_2_CB7	1.2-V POD	AD20	ECC 数据
DDR4_2_CB6	1.2-V POD	AG21	
DDR4_2_CB5	1.2-V POD	AE22	
DDR4_2_CB4	1.2-V POD	AH22	
DDR4_2_CB3	1.2-V POD	AC22	
DDR4_2_CB2	1.2-V POD	AD22	
DDR4_2_CB1	1.2-V POD	AC20	
DDR4_2_CB0	1.2-V POD	AF21	
DDR4_2_DQ63	1.2-V POD	AP22	数据信号
DDR4_2_DQ62	1.2-V POD	AL24	
DDR4_2_DQ61	1.2-V POD	AP21	
DDR4_2_DQ60	1.2-V POD	AP20	
DDR4_2_DQ59	1.2-V POD	AP24	
DDR4_2_DQ58	1.2-V POD	AL25	
DDR4_2_DQ57	1.2-V POD	AN20	
DDR4_2_DQ56	1.2-V POD	AM23	
DDR4_2_DQ55	1.2-V POD	AL27	
DDR4_2_DQ54	1.2-V POD	AP26	
DDR4_2_DQ53	1.2-V POD	AN25	
DDR4_2_DQ52	1.2-V POD	AP25	
DDR4_2_DQ51	1.2-V POD	AM27	
DDR4_2_DQ50	1.2-V POD	AP27	
DDR4_2_DQ49	1.2-V POD	AK26	

DDR4_2_DQ48	1.2-V POD	AK27	
DDR4_2_DQ47	1.2-V POD	AD25	
DDR4_2_DQ46	1.2-V POD	AJ26	
DDR4_2_DQ45	1.2-V POD	AE24	
DDR4_2_DQ44	1.2-V POD	AH25	
DDR4_2_DQ43	1.2-V POD	AC24	
DDR4_2_DQ42	1.2-V POD	AJ25	
DDR4_2_DQ41	1.2-V POD	AD24	
DDR4_2_DQ40	1.2-V POD	AH27	
DDR4_2_DQ39	1.2-V POD	AG23	
DDR4_2_DQ38	1.2-V POD	AK24	
DDR4_2_DQ37	1.2-V POD	AK23	
DDR4_2_DQ36	1.2-V POD	AK22	
DDR4_2_DQ35	1.2-V POD	AJ24	
DDR4_2_DQ34	1.2-V POD	AL23	
DDR4_2_DQ33	1.2-V POD	AF24	
DDR4_2_DQ32	1.2-V POD	AJ22	
DDR4_2_DQ31	1.2-V POD	C20	
DDR4_2_DQ30	1.2-V POD	E19	
DDR4_2_DQ29	1.2-V POD	F19	
DDR4_2_DQ28	1.2-V POD	D22	
DDR4_2_DQ27	1.2-V POD	D20	
DDR4_2_DQ26	1.2-V POD	E18	
DDR4_2_DQ25	1.2-V POD	G20	
DDR4_2_DQ24	1.2-V POD	C22	
DDR4_2_DQ23	1.2-V POD	M18	
DDR4_2_DQ22	1.2-V POD	L19	
DDR4_2_DQ21	1.2-V POD	J21	
DDR4_2_DQ20	1.2-V POD	L20	
DDR4_2_DQ19	1.2-V POD	M17	
DDR4_2_DQ18	1.2-V POD	L18	
DDR4_2_DQ17	1.2-V POD	K21	
DDR4_2_DQ16	1.2-V POD	M20	
DDR4_2_DQ15	1.2-V POD	H17	
DDR4_2_DQ14	1.2-V POD	F18	
DDR4_2_DQ13	1.2-V POD	H18	
DDR4_2_DQ12	1.2-V POD	G18	
DDR4_2_DQ11	1.2-V POD	J17	
DDR4_2_DQ10	1.2-V POD	G17	
DDR4_2_DQ9	1.2-V POD	J20	
DDR4_2_DQ8	1.2-V POD	H20	
DDR4_2_DQ7	1.2-V POD	B22	
DDR4_2_DQ6	1.2-V POD	C18	

DDR4_2_DQ5	1.2-V POD	B21	
DDR4_2_DQ4	1.2-V POD	A18	
DDR4_2_DQ3	1.2-V POD	D17	
DDR4_2_DQ2	1.2-V POD	D19	
DDR4_2_DQ1	1.2-V POD	A21	
DDR4_2_DQ0	1.2-V POD	B18	
DDR4_2_DQS8_T	Differential 1.2-V POD	AG20	数据 Probe 信号
DDR4_2_DQS8_C	Differential 1.2-V POD	AH20	
DDR4_2_DQS7_T	Differential 1.2-V POD	AM22	
DDR4_2_DQS7_C	Differential 1.2-V POD	AN22	
DDR4_2_DQS6_T	Differential 1.2-V POD	AM25	
DDR4_2_DQS6_C	Differential 1.2-V POD	AM26	
DDR4_2_DQS5_T	Differential 1.2-V POD	AG25	
DDR4_2_DQS5_C	Differential 1.2-V POD	AF25	
DDR4_2_DQS4_T	Differential 1.2-V POD	AH23	
DDR4_2_DQS4_C	Differential 1.2-V POD	AH24	
DDR4_2_DQS3_T	Differential 1.2-V POD	E21	
DDR4_2_DQS3_C	Differential 1.2-V POD	D21	
DDR4_2_DQS2_T	Differential 1.2-V POD	M21	
DDR4_2_DQS2_C	Differential 1.2-V POD	L21	
DDR4_2_DQS1_T	Differential 1.2-V POD	G21	
DDR4_2_DQS1_C	Differential 1.2-V POD	F21	
DDR4_2_DQS0_T	Differential 1.2-V POD	A19	
DDR4_2_DQS0_C	Differential 1.2-V POD	A20	
CLK_EMI_1_P	LVDS	E23	参考时钟，默认值：266.667MHz
CLK_EMI_1_N	LVDS	E24	

4.2.5 FPGA 调试、配置

采用 JTAG 接口调试，通过菊花链方式连接 CPLD 和 FPGA，需外接 Blaster II。

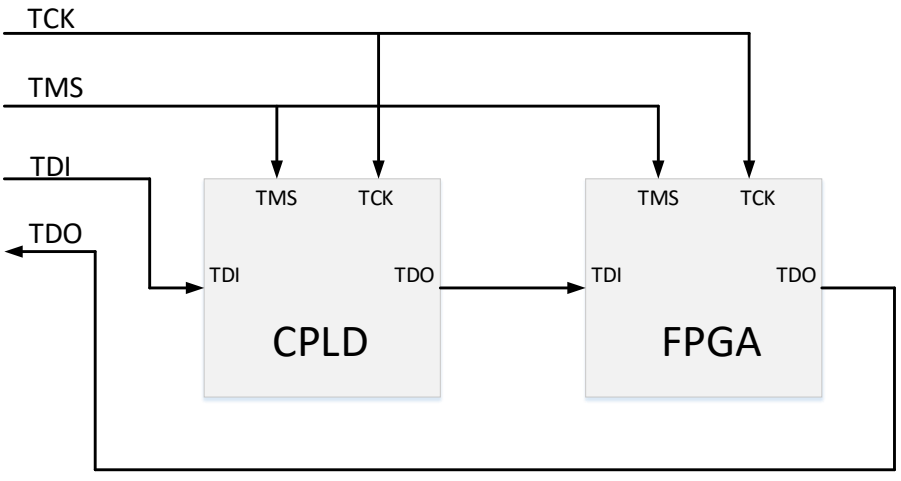

图 4-3　JTAG 菊花链结构图

板卡 JTAG 接口线序如下表所示：

1	2	3	4	5	6
TCK	GND	TDO	VCC	TMS	TDI

对应板卡管脚位置：

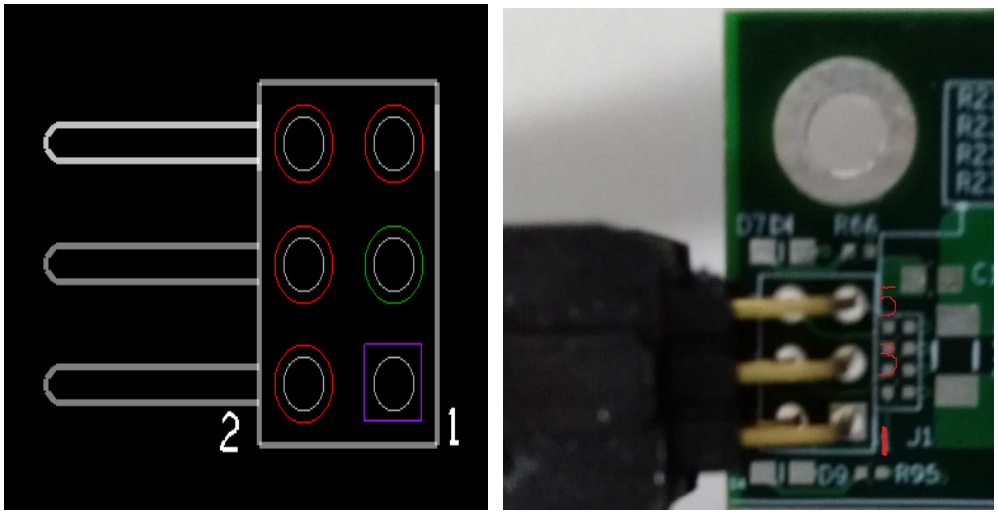

图 4-4　6 PIN JTAG 管脚位置及实物图

FPGA 加载方式默认为 AS 模式，其加载结构示意图如下图 4-5 所示。

AS 加载：CFI FLASH 规格 1Gbit. QSPI (EPCQ_L)

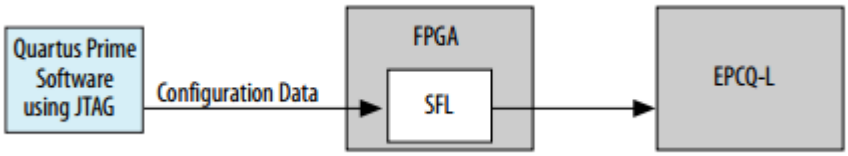

图 4-5 AS 加载结构示意图

4.2.6　LED 电路

板卡预留 4 个 LED 灯，用于信号状态指示。

LED 灯对应 FPGA 管脚：

信号名称	管脚	电平标准	位号	亮/灭
PCIE_LED_X1	AB11	1.8V	D3	逻辑 0 /逻辑 1
PCIE_LED_X4	AB10	1.8V	D4	逻辑 0 /逻辑 1
LED_1	AC4	1.8V	D7	逻辑 0 /逻辑 1
LED_2	AC7	1.8V	D9	逻辑 0 /逻辑 1

4.2.7 FPGA 与 CPLD 通信接口

为方便 FPGA 与 CPLD 进行通信，硬件设计上预留了 7 根信号，可通过 UART 或 SPI 协议传输命令和数据。

对应 FPGA 管脚：

信号名称	FPGA 管脚	IO 标准	说明
CFG_S1	AE1	1.8V	用户自定义
CFG_S2	AF1	1.8V	
CFG_S3	AL1	1.8V	
CFG_S5_FPGA	AG3	1.8V	
FPGA_PR_DONE	AF16	1.8V	
FPGA_PR_REQUEST	AJ16	1.8V	
MAX_RESET_n	AK1	1.8V	

4.2.8 FPGA 与 HOST 管理接口 SMBUS

为方便 HOST 端对板卡的统一管理，PCIe 插槽上的 SMBUS 总线连接到 FPGA 管脚和 CPLD 管脚上，请注意编程时器件地址要区分开。

对应 FPGA 管脚：

信号名称	管脚	IO 标准	说明
PCIE_SMBCLK	AL4	1.8V	串行时钟
PCIE_SMBDAT	AL5	1.8V	串行数据

4.3CPLD 电路介绍

CPLD 采用芯片 MAX10 10M02SCU169，主要完成 FPGA 的并行加载，与 FPGA 的通信，I2C 管理系统（设置时钟 PLL、读取 FPGA 温度和板卡温度、读取板卡功耗、读取 EEPROM）等功能。

4.3.1 时钟电路

采用一片单端 100M 晶振为 CPLD 提供工作时钟。

CPLD 时钟输入管脚：

信号名称	电平标准	管脚	说明
CLK_CONFIG	1.8V	H6	100MHz 时钟

4.3.2 FLASH 电路

在 FPP 配置方式下，板上采用两片 CFI FLASH 级联成 32bit 数据，每片 FLASH 容量 1Gb，工作频率 100MHz

FLASH 接口信号对应 CPLD 管脚：

信号名	电平标准	管脚	说明
FM_A1	1.8V	D11	FLASH 地址线
FM_A2	1.8V	E4	
FM_A3	1.8V	C2	
FM_A4	1.8V	E3	
FM_A5	1.8V	B4	
FM_A6	1.8V	A10	
FM_A7	1.8V	A2	
FM_A8	1.8V	C9	
FM_A9	1.8V	G4	
FM_A10	1.8V	B2	
FM_A11	1.8V	A5	
FM_A12	1.8V	A3	
FM_A13	1.8V	B10	
FM_A14	1.8V	A6	
FM_A15	1.8V	E6	
FM_A16	1.8V	A4	
FM_A17	1.8V	B3	
FM_A18	1.8V	A9	
FM_A19	1.8V	A7	
FM_A20	1.8V	D9	
FM_A21	1.8V	F4	
FM_A22	1.8V	A8	
FM_A23	1.8V	F1	
FM_A24	1.8V	K2	
FM_A25	1.8V	B6	
FM_A26	1.8V	E8	
FLASH_CLK	1.8V	B1	时钟
FLASH_RESETn	1.8V	B5	复位
FLASH_CEn0	1.8V	D7	FLASH0 片选
FLASH_CEn1	1.8V	D6	FLASH1 片选
FLASH_RDYBSYn0	1.8V	C5	FLASH 0 BYSY
FLASH_RDYBSYn1	1.8V	C4	FLASH 1 BYSY
FLASH_OEn	1.8V	C1	输出使能
FLASH_WEn	1.8V	E1	写信号
FLASH_ADVn	1.8V	D1	地址有效信号
FLASH_WPn	1.8V	-------	写保护
FM_D0	1.8V	F12	FLASH 数据线
FM_D1	1.8V	F13	

FM_D2	1.8V	A11	
FM_D3	1.8V	E12	
FM_D4	1.8V	D12	
FM_D5	1.8V	A12	
FM_D6	1.8V	B12	
FM_D7	1.8V	B11	
FM_D8	1.8V	G10	
FM_D9	1.8V	E13	
FM_D10	1.8V	D13	
FM_D11	1.8V	C13	
FM_D12	1.8V	C12	
FM_D13	1.8V	C10	
FM_D14	1.8V	C11	
FM_D15	1.8V	B13	
FM_D16	1.8V	F9	
FM_D17	1.8V	E10	
FM_D18	1.8V	L1	
FM_D19	1.8V	E9	
FM_D20	1.8V	F8	
FM_D21	1.8V	H2	
FM_D22	1.8V	H3	
FM_D23	1.8V	N3	
FM_D24	1.8V	N11	
FM_D25	1.8V	F10	
FM_D26	1.8V	H1	
FM_D27	1.8V	G5	
FM_D28	1.8V	G9	
FM_D29	1.8V	N2	
FM_D30	1.8V	H4	
FM_D31	1.8V	B7	

4.3.3 CPLD JTAG 接口

JTAG 接口对应 CPLD 管脚：

信号名称	电平标准	管脚
JTAG_TCK	1.8V	G2
JTAG_TDI	1.8V	F5
JTAG_TDO	1.8V	F6
JTAG_TMS	1.8V	G1

4.3.4 CPLD 配置 FPGA 电路

系统上电后，CPLD 读取 FLASH 中的数据，通过并行总线配置 FPGA。

CPLD 配置 FPGA 管脚定义：

信号名称	电平标准	管脚	应用说明
FPGA_CONFIG_D0	1.8V	M3	配置数据
FPGA_CONFIG_D1	1.8V	M4	
FPGA_CONFIG_D2	1.8V	N10	
FPGA_CONFIG_D3	1.8V	N9	
FPGA_CONFIG_D4	1.8V	M8	
FPGA_CONFIG_D5	1.8V	J7	
FPGA_CONFIG_D6	1.8V	N4	
FPGA_CONFIG_D7	1.8V	N5	
FPGA_CONFIG_D8	1.8V	N6	
FPGA_CONFIG_D9	1.8V	M5	
FPGA_CONFIG_D10	1.8V	M7	
FPGA_CONFIG_D11	1.8V	N7	
FPGA_CONFIG_D12	1.8V	J8	
FPGA_CONFIG_D13	1.8V	K8	
FPGA_CONFIG_D14	1.8V	M10	
FPGA_CONFIG_D15	1.8V	M12	
FPGA_CONFIG_D16	1.8V	K6	
FPGA_CONFIG_D17	1.8V	K7	
FPGA_CONFIG_D18	1.8V	M11	
FPGA_CONFIG_D19	1.8V	N12	
FPGA_CONFIG_D20	1.8V	M9	
FPGA_CONFIG_D21	1.8V	N8	
FPGA_CONFIG_D22	1.8V	J6	
FPGA_CONFIG_D23	1.8V	J1	
FPGA_CONFIG_D24	1.8V	L4	
FPGA_CONFIG_D25	1.8V	L5	
FPGA_CONFIG_D26	1.8V	L11	
FPGA_CONFIG_D27	1.8V	J5	
FPGA_CONFIG_D28	1.8V	M13	
FPGA_CONFIG_D29	1.8V	L10	
FPGA_CONFIG_D30	1.8V	K5	
FPGA_CONFIG_D31	1.8V	L3	
FPGA_CLK	1.8V	L2	配置时钟
FPGA_CONF_DONE	1.8V	M2	配置完成
FPGA_nCONFIG	1.8V	J2	配置起始信号
FPGA_nSTATUS	1.8V	M1	配置状态

4.3.5 SMBus 系统管理

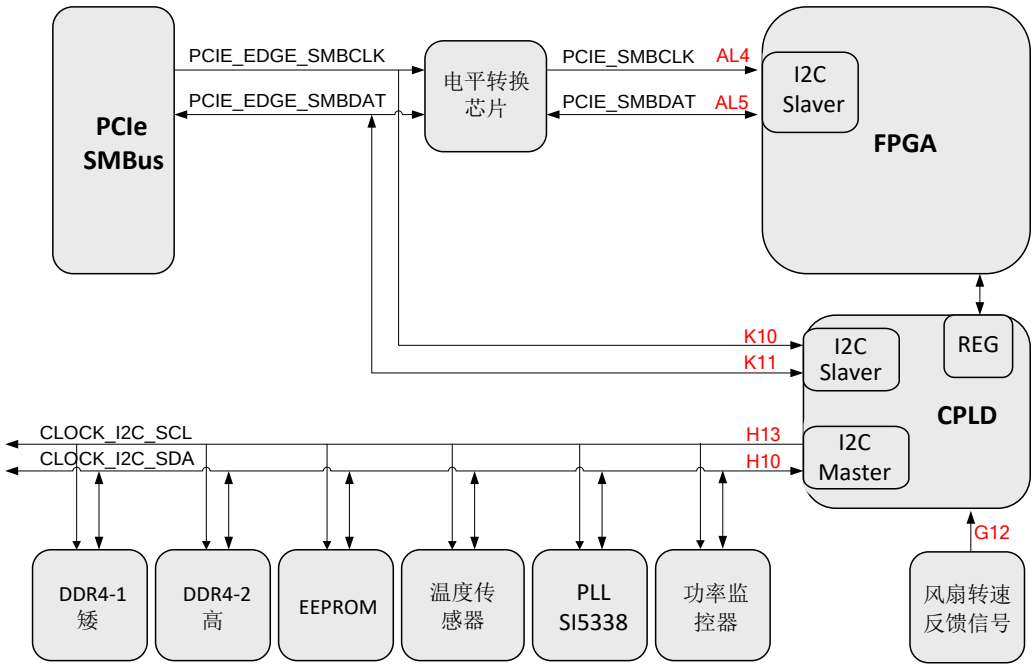

图 4-6 I2C 配置电路框图

PCIe SMBUS（PCIE_EDGE_SMBCLK、PCIE_EDGE_SMBDAT）连接 CPLD 和 FPGA，CPLD 和 FPGA 作为 I2C SLAVE，地址自定义，注意不要冲突。

CPLD 作为 MASTER（CLOCK_I2C_SCL、CLOCK_I2C_SDA），连接各 I2C SLAVE，地址如下：

I2C SLAVE DEVICE		ADDR(A7-A0)	
矮处内存	DDR4_1	SPD	1010 000x
		TEMP	0011 000x
高处内存	DDR4_2	SPD	1010 001x
		TEMP	0011 001x
EEPROM	M24512	1010 111x	
温度传感器	MAX1619	1001 100x	
时钟发生器	SI5338	1110 000x	
功率监控器	INA219	1000 000x	

PLL SI5338，产生 DDR4 的参考时钟和 SFP+的参考时钟。

图 4-7 PLL 时钟发生器框图

风扇转速反馈信号（FAN_SPEED）和风扇调速信号 FAN_PWM，是连接 CPLD 的 3.3V 脉冲信号，频率和风扇的转速成正比。

信号名称	电平标准	管脚	说明
FAN_SPEED	3.3V	G12	风扇转速反馈信号
FAN_PWM/CPLD_LED	3.3V	H9	风扇 PWM 控制信号

注意：FAN_PWM/CPLD_LED 为复用管脚，V4.2 版本上作为 FAN_PWM，V4.1 及以下版本仍作为 CPLD_LED 内部状态指示灯信号（高电平点灯）。

I2C 对应 CPLD 管脚：

信号名称	电平标准	管脚	说明
PCIE_EDGE_SMBCLK	3.3V	K10	I2C Slaver 时钟
PCIE_EDGE_SMBDAT	3.3V	K11	I2C Slaver 数据
CLOCK_I2C_SCL	3.3V	H13	I2C Master 时钟
CLOCK_I2C_SDA	3.3V	H10	I2C Master 数据

4.3.6 拨码开关及状态指示灯

板上留有 4 位拨码开关和两个状态指示灯。拨码开关 3 位连接 CPLD，用于作为预留，1 位连接 FPGA。

两个状态指示灯，1 个连接 CPLD 作为状态预留（位号 D6），一个有 3.3V 电源直接点灯，指示上电状态（位号 D8）。

拨码开关及状态指示灯对应 CPLD&FPGA 管脚：

信号名称	电平标准	管脚	初始状态 or 位号	说明
SW1	3.3V	K12	ON，逻辑 0	用户定义
SW2	3.3V	L13	ON，逻辑 0	
SW3	3.3V	J12	ON，逻辑 0	
MSEL1	1.8V	G12（FPGA）	ON，逻辑 0	连接 FPGA 的模式选择管脚
CPLD_LED	3.3V	H9	D6	On，逻辑 1

4.3.7 CPLD 与 FPGA 通信

为方便 CPLD 与 FPGA 进行通信，硬件设计上预留了 7 根信号，可通过 UART 或 SPI 协议传输命令和数据。

对应 CPLD 管脚：

信号名称	电平标准	管脚	说明
CFG_S1	1.8V	B9	用户自定义
CFG_S2	1.8V	D8	
CFG_S3	1.8V	K1	
CFG_S5_CPLD	3.3V	J13	
CPLD_PR_DONE	3.3V	H8	
CPLD_PR_REQUEST	3.3V	G13	
MAX_RESET_n	1.8V	H5	

5. F10A 板卡环境搭建

5.1板卡使用环境

➢ 硬件：

- 带有 x16 或 x8 PCIe 插槽的服务器或工作站（e.g. NF5280M4, NF5280M5，NF5212M4）
- 至少要有 150GB 的可用内存，用于 OpenCL kernel 编译
- F10A 板卡
- 板卡工程烧写器件 Blaster II 及对应的转接线

➢ 软件：

- 操作系统（Redhat[6.x ,7.x]　CentOS[6.x,7.x]），要求以开发模式安装(包含 GCC、Make 等工具以及 kernel 源代码，用于驱动编译)
- Inspur RTE Package for F10A
- Inspur DEV Package for F10A

Note: 禁止把板卡插在 x24 的 PCI-E 插槽

F10A 板卡可用于两种不同模式下的开发：RTL 开发和 OpenCL 开发。现分别介绍两种不同模式下的开发流程以及相关文件。

5.2RTL 开发简介

F10A 板卡可用于加速某一应用，比如图像转换，视频解压缩等。RTL 开发流程，需关注三个层面：应用层，驱动层和板卡逻辑层，如下图 5-1 所示。

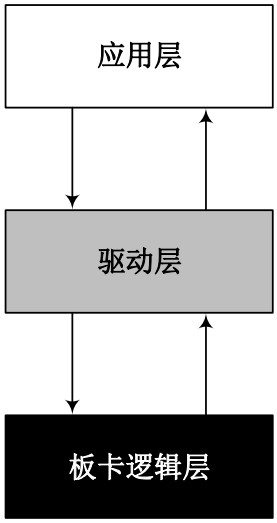

图 5-1 RTL 开发三个层面图

- **板卡逻辑层：** 由 Verilog 或 VHDL 语言编写运算量大或时效性要求高的业务逻辑，综合编译得到 FPGA 芯片的配置文件，烧写到板卡芯片中运行。
- **应用层：** 由高级语言编写（C，C++，Java 等），主要用于数据预处理和控制作用的业务逻辑，运行在主机系统上。
- **驱动层：** 连接应用层和板卡逻辑层的桥梁，实现寄存器的访问配置和 DMA 传输。

➢ 相关文件

```
[root@localhost F10A]# tree -L 3
.
└── RTL
    ├── application_demo
    │   ├── readme.pdf
    │   └── src
    ├── document
    │   └── ug_a10_pcie_avmm_dma.pdf
    ├── driver
    │   └── src
    └── reference_design
        ├── ddr4_parameter_set.zip
        ├── FPGA_configuration_file
        └── pcie_ddr4_8gx2_example_v1.tar.gz

8 directories, 4 files
```

说明：

- **reference_design:**

- pcie_ddr4_8gx2_example_v1.tar.gz：针对 F10A 板卡的一个参考设计，具备 MSI 中断和 DMA 传输功能。
- FPGA_configuration_file：上述设计综合得到的 top.sof 配置文件，可烧写到板卡芯片中用于验证。建议此参考设计用 Quartus Prime Pro 16.1 打开。
- ddr4_parameter_set.zip：板卡搭载的两个 DDR，相关参数配置示例图。

– **driver**: 包含驱动源代码文件

– **application_demo**：包含一个简单的上层用代码示例，展示 DMA 传输功能。

– **document:**包含一个介绍 DMA 实现原理的手册，有助于理解驱动中的 DMA 实现代码

5.3OpenCL 开发简介

使用 OpenCL 开发加速某一应用，只需关注两个层面：主机端（Host Code）和设备端(Kernel Code)，如下图 5-2 所示：

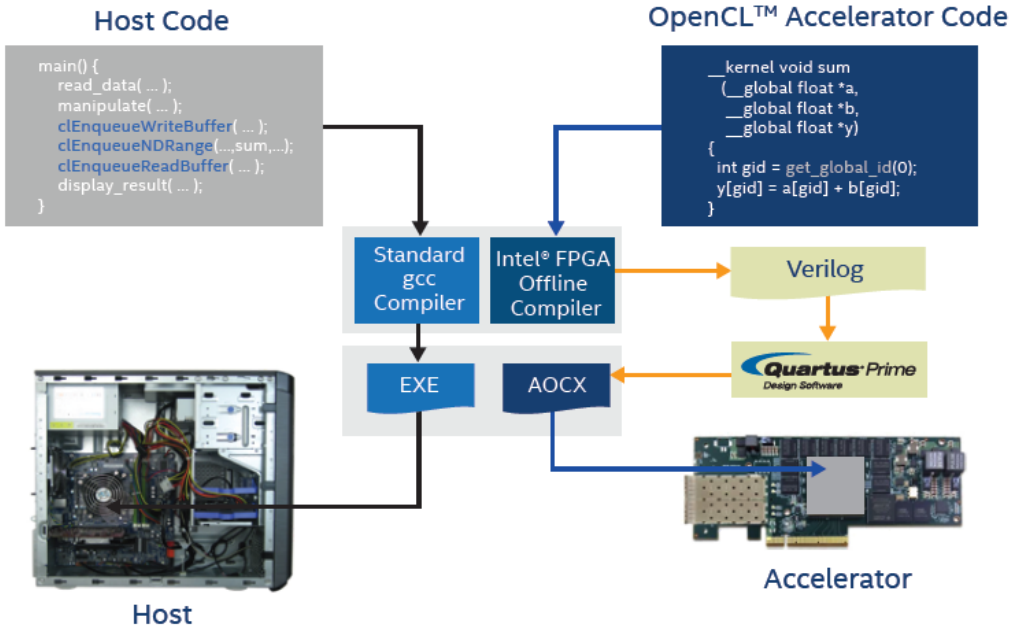

图 5-2 OpenCL 开发框架图

- **主机端：**由 C/C++语言编写，通过调用 OpenCL 定义的 API，实现数据预处理和主机板卡间数据传输以及通信的功能。
- **设备端：**由类 C 的一套语法规范，编写实现具有不同功能的 kernel 函数，经由 Altera 提供的编译器 AOC（Altera Offline Compiler）生成对应的 Verilog 代码，后通过 Quartus 综合编译得到配置文件(*.sof),最后再封装成可执行文件（*.aocx），下载到 FPGA 芯片中去执行。

➢ OpenCL 开发相关文档

在此列举了 OpenCL 开发入门的相关文档，请参见：

➢ Books

- Heterogeneous Computing with OpenCL
- OpenCL Programming Guide

➢ Altera OpenCL collateral

- www.altera.com/OpenCL
- Demos and Design Examples
- Altera SDK for OpenCL Getting Started Guide
- Altera SDK for OpenCL Programming Guide
- Altera SDK for OpenCL Best Practices Guide
- ug_aocl_custom_platform_toolkit.pdf
- ug-aocl-altera-a10pciedk-platform.pdf

可通过阅读了解 OpenCL 开发的编程框架以及相关概念（e.g. Platform，Device, Context, Compute Unit），随后阅读 OpenCL 的规范，了解相关的 API，最后阅读 Altera 提供的指导手册，其中 Getting Started Guide, Programming Guide 在使用板卡前必须要读，其中 ug-aocl-altera-a10pciedk-platform 可加深对 Altera SDK 的理解。

➢ OpenCL 开发流程

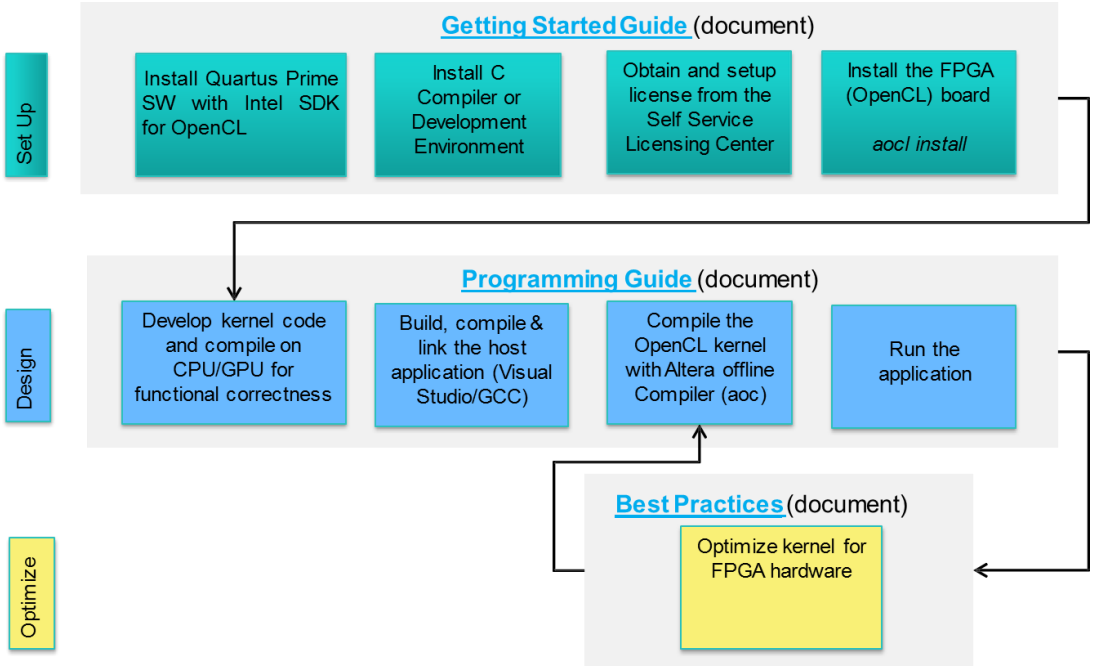

图 5-3 OpenCL 开发流程图

上图 5-3 是 OpenCL 开发的简化流程，主要分为三个部分：环境搭建（Set_Up），功能实现（Design），优化（Optimize）。本用户手册中提到的 F10A DEV 和 RTE Package 都支持 OpenCL 开发，下面将分别介绍 DEV 和 RTE Package 的安装和环境搭建流程。

5.4DEV Package 使用说明

DEV Packge 为 F10A 板卡开发环境包。安装 DEV Package 并设置环境变量即可完成 F10A 板卡 OpenCL 开发环境搭建。

5.4.1 DEV Package 安装

获取浪潮提供的 DEV 安装包<inspur_f10a_dev_package>.tar.gz，对安装包进行解压，然后运行 setup.sh 安装脚本：***$./setup.sh***，在安装过程中根据提示输入 DEV 安装路径（需确认安装路径有普通用户权限），也可安装到默认路径，如下图 5-4 所示。

```
[chris@localhost inspur_f10a_dev_package]$ ./setup.sh

------------------------------------------------------------------------------
- Beginning installing Intel Acceleration Stack Development Package
------------------------------------------------------------------------------

Enter the path you want to extract the Inpsur F10A with Intel Arria10 GX FPGA release package [default: /home/chris/inspur_f10a_dev_stack]:

INSTALLDIR=/home/chris/inspur_f10a_dev_stack

------------------------------------------------------------------------------
- Copying components.tar.gz to /home/chris/inspur_f10a_dev_stack
------------------------------------------------------------------------------
>>> Running cmd:
      cp -pf /home/chris/work2019/f10a/dev/inspur_f10a_dev_package/components.tar.gz /home/chris/inspur_f10a_dev_stack

mkdir /home/chris/inspur_f10a_dev_stack/components

------------------------------------------------------------------------------
- Untar components.tar.gz
------------------------------------------------------------------------------
DCP_LOC=/home/chris/inspur_f10a_dev_stack/components
>>> Running cmd:
      tar -xzf /home/chris/work2019/f10a/dev/inspur_f10a_dev_package/components.tar.gz -C ../


------------------------------------------------------------------------------
- Installing the Intel FPGA Quartus Prime Pro Edition Software
------------------------------------------------------------------------------
```

图 5-4 DEV 安装图示

5.4.2 DEV Package 环境搭建

DEV Package 安装完成之后，根据提示设置环境变量，执行指令：***$source <install_path>/inspur_f10a_dev_stack/init_env.sh***，如下图 5-5 示例。

```
[chris@localhost inspur_f10a_dev_package]$ source /opt/inspur/inspur_f10a_dev_stack/init_env.sh
-------------------------------------------------------------------------------
- ====================Setup Environment====================
-------------------------------------------------------------------------------
make -C host TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[1]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/host"
make -C mmd TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[2]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/host/mmd"
rm -rf /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/lib/libaltera_f10a_sr2x8g_mmd.so ./acl_pcie_debug.
make[2]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/host/mmd"
make[1]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/host"
make -C util TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[1]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util"
make -C diagnostic TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[2]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/diagnostic"
rm -rf /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/libexec/diagnose ./ocl.o ./main.o ./acl_aligned.o
make[2]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/diagnostic"
make -C reprogram TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[2]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/reprogram"
rm -rf /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/libexec/program ./reprogram.o
make[2]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/reprogram"
make -C download TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[2]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/download"
rm -rf /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/libexec/download ./download.o ./acl_aligned.o
make[2]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/download"
make -C flash_pcie TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[2]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/flash_pcie"
rm -rf /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/libexec/flash_pcie ./acl_pcie_flash.o ./common.o .
make[2]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/flash_pcie"
make -C remote_update TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
make[2]: 进入目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/remote_update"
rm -rf /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/libexec/remote_update ./remote_update.o
make[2]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/remote_update"
make -C reg_ctl TOP_DEST_DIR=/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 clean
```

图 5-5 DEV 环境变量配置图

如果是首次设置搭建环境，执行指令之后会提示安装驱动，如下图 5-6 所示。

```
make[2]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util/fru_readwrite"
make[1]: 离开目录"/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/source/util"
-------------------------------------------------------------------------------
- ====================Environment Set Finish====================
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
- ====================Install Driver====================
-------------------------------------------------------------------------------
lsmod: /opt/inspur/inspur_f10a_dev_stack/intelFPGA_pro/quartus/linux64/liblzma.so.5: no version information available (required by lsmod)
-------------------------------------------------------------------------
- Please run the driver install command using the root user:aocl install
-------------------------------------------------------------------------------
```

图 5-6 驱动安装指令示例图

此时需要切换到 root 用户，并执行驱动安装指令：***$aocl install***，如下图 5-7 所示。如果提示驱动更新，则需切换到 root 用户下先执行指令***$aocl uninstall*** 卸载指令，然后执行指令：***$sudo rm –rf /opt/Intel*** 再执行驱动安装指令。注意：此步骤必须要切换到 root 用户！

```
bash-4.2# aocl install
Do you want to setup the FCD at directory /opt/Intel/OpenCL/Boards [y/n] y
aocl install: Adding the board package /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 to the list of installed packages
aocl install: Setting up the FPGA Client Driver (FCD) to the system.
Install the FCD file to /opt/Intel/OpenCL/Boards
Installing the board package driver to the system.
aocl install: Running install from /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/libexec
Looking for kernel source files in /lib/modules/3.10.0-862.el7.x86_64/build
Using kernel source files from  /lib/modules/3.10.0-862.el7.x86_64/build
Building driver for BSP with name f10a_sr2x8g
make: 进入目录"/usr/src/kernels/3.10.0-862.el7.x86_64"
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_queue.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_fileio.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_dma.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_pr.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_cmd.o
  LD [M]  /tmp/opencl_driver_cTYJna/aclpci_f10a_sr2x8g_drv.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /tmp/opencl_driver_cTYJna/aclpci_f10a_sr2x8g_drv.mod.o
  LD [M]  /tmp/opencl_driver_cTYJna/aclpci_f10a_sr2x8g_drv.ko
make: 离开目录"/usr/src/kernels/3.10.0-862.el7.x86_64"
```

图 5-7 驱动安装图

驱动安装完成之后重新执行环境变量设置指令：*$source <install_path>/inspur_f10a_dev_stack/init_env.sh*，提示执行指令*$aocl diagnose* 查看板卡运行状态，如下图 5-8 所示。

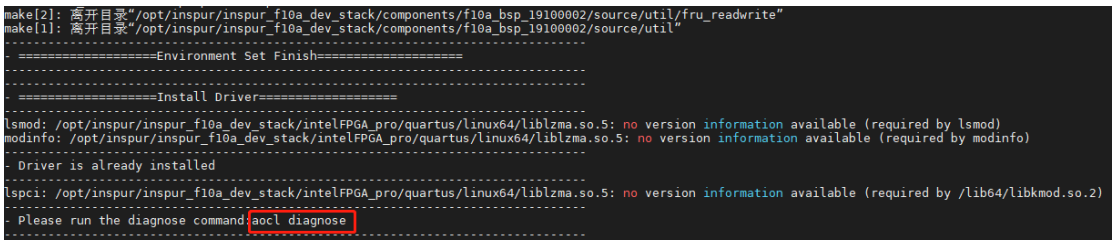

图 5-8 驱动安装完成图

至此，板卡 DEV 开发环境搭建完成，可执行指令*$aocl diagnose* 查看板卡运行状态，如下图 5-9 所示。如果显示 DIAGNOSTIC_PASSED，则说明环境搭建完成，板卡运行正常。

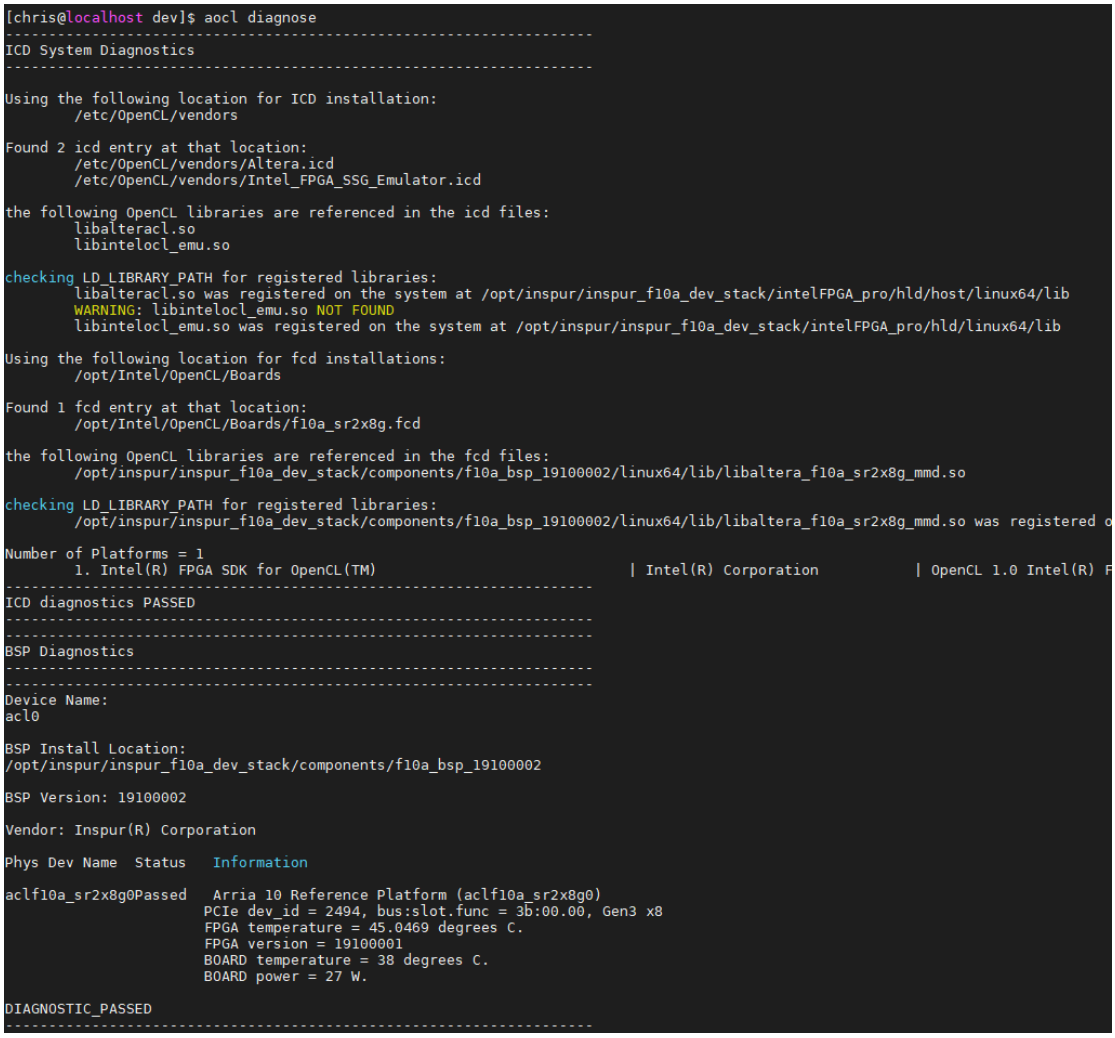

图 5-9 板卡诊断指令示例图

注意：每次打开终端都需要重新设置环境变量，即执行指令：***$source***

<install_path>/inspur_f10a_dev_stack/init_env.sh

5.5RTE Package 使用说明

获取浪潮提供的 F10A 板卡 RTE 包<inspur_f10a_rte_package>.tar.gz，解压之后得到 F10A RTE 包，如下图 5-10 所示。

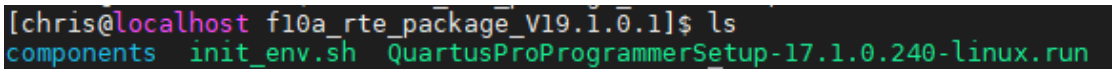

图 5-10 RTE Package 图

RTE Package 不需要安装，执行指令***$source init_env.sh*** 即可完成 F10A 板卡运行环境变量设置，如下图 5-11 所示。

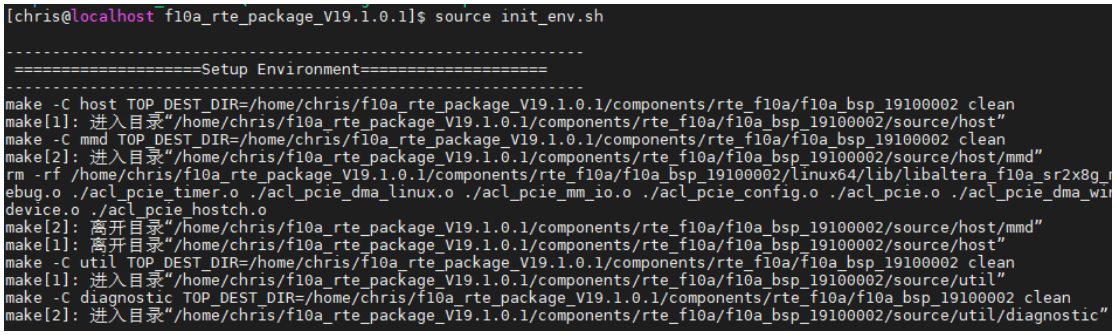

图 5-11 RTE 环境变量配置图

如果是首次设置搭建环境，执行指令之后会提示安装驱动，如下图 5-12 所示。

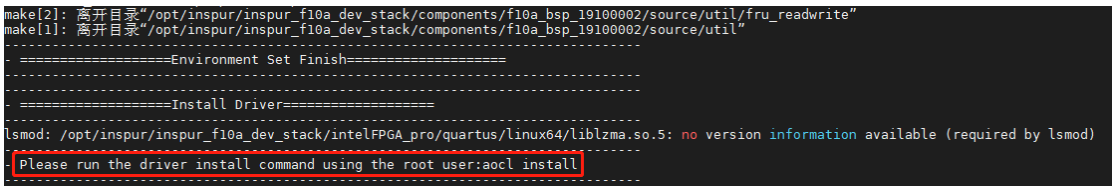

图 5-12 驱动安装指令示例图

此时需要切换到 root 用户，并执行驱动安装指令：*$aocl install*，如下图 5-13 所示。如果提示驱动更新，则需切换到 root 用户下先执行指令*$aocl uninstall* 卸载指令，然后再执行驱动安装指令。注意：此步骤必须要切换到 root 用户！

```
bash-4.2# aocl install
Do you want to setup the FCD at directory /opt/Intel/OpenCL/Boards [y/n] y
aocl install: Adding the board package /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002 to the list of installed packages
aocl install: Setting up the FPGA Client Driver (FCD) to the system.
Install the FCD file to /opt/Intel/OpenCL/Boards
Installing the board package driver to the system.
aocl install: Running install from /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/libexec
Looking for kernel source files in /lib/modules/3.10.0-862.el7.x86_64/build
Using kernel source files from  /lib/modules/3.10.0-862.el7.x86_64/build
Building driver for BSP with name f10a_sr2x8g
make: 进入目录"/usr/src/kernels/3.10.0-862.el7.x86_64"
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_queue.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_fileio.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_dma.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_pr.o
  CC [M]  /tmp/opencl_driver_cTYJna/aclpci_cmd.o
  LD [M]  /tmp/opencl_driver_cTYJna/aclpci_f10a_sr2x8g_drv.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /tmp/opencl_driver_cTYJna/aclpci_f10a_sr2x8g_drv.mod.o
  LD [M]  /tmp/opencl_driver_cTYJna/aclpci_f10a_sr2x8g_drv.ko
make: 离开目录"/usr/src/kernels/3.10.0-862.el7.x86_64"
```

图 5-13 驱动安装图

驱动安装完成之后重新执行环境变量设置指令*$source init_env.sh*，此时会提示执行指令*$aocl diagnose* 查看板卡运行状态，如下图 5-14 所示。如果显示 DIAGNOSTIC_PASSED，则说明环境搭建完成，板卡运行正常。

```
[chris@localhost f10a_rte_package_V19.1.0.1]$ aocl diagnose
-----------------------------------------------------------------------
ICD System Diagnostics
-----------------------------------------------------------------------

Using the following location for ICD installation:
       /etc/OpenCL/vendors

Found 2 icd entry at that location:
       /etc/OpenCL/vendors/Altera.icd
       /etc/OpenCL/vendors/Intel_FPGA_SSG_Emulator.icd

the following OpenCL libraries are referenced in the icd files:
       libalteracl.so
       libintelocl_emu.so

checking LD_LIBRARY_PATH for registered libraries:
       libalteracl.so was registered on the system at /home/chris/work2019/f10a/rte/f10a_rte_package_V19.1.0.1/components/rte_f10a/host/linux64/lib
       libintelocl_emu.so was registered on the system at /opt/inspur/inspur_f10a_dev_stack/intelFPGA_pro/hld/linux64/lib

Using the following location for fcd installations:
       /opt/Intel/OpenCL/Boards

Found 1 fcd entry at that location:
       /opt/Intel/OpenCL/Boards/f10a_sr2x8g.fcd

the following OpenCL libraries are referenced in the fcd files:
       /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/lib/libaltera_f10a_sr2x8g_mmd.so

checking LD_LIBRARY_PATH for registered libraries:
       /opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002/linux64/lib/libaltera_f10a_sr2x8g_mmd.so was registered on the system.

Number of Platforms = 1
       1. Intel(R) FPGA SDK for OpenCL(TM)                          | Intel(R) Corporation          | OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), Version 19.1
-----------------------------------------------------------------------
ICD diagnostics PASSED
-----------------------------------------------------------------------
-----------------------------------------------------------------------
BSP Diagnostics
-----------------------------------------------------------------------
-----------------------------------------------------------------------
Device Name:
acl0

BSP Install Location:
/opt/inspur/inspur_f10a_dev_stack/components/f10a_bsp_19100002

BSP Version: 19100002

Vendor: Inspur(R) Corporation

Phys Dev Name  Status   Information

aclf10a_sr2x8g0Passed   Arria 10 Reference Platform (aclf10a_sr2x8g0)
                      PCIe dev_id = 2494, bus:slot.func = 3b:00.00, Gen3 x8
                      FPGA temperature = 43.6641 degrees C.
                      FPGA version = 19100001
                      BOARD temperature = 37 degrees C.
                      BOARD power = 27 W.

DIAGNOSTIC_PASSED
-----------------------------------------------------------------------
```

图 5-14 板卡诊断指令示例图

注意：每次打开终端都需要设置环境变量，即执行指令：***$source init_env.sh***

5.6init_env.sh 说明

init_env.sh 脚本中定义了 Quartus 编译工程的规则，用户可根据需求进行更改，其中相关的配置脚本如下图 5-15 所示。

```
#export CL_CONTEXT_EMULATOR_DEVICE_INTELFPGA=1
unset CL_CONTEXT_EMULATOR_DEVICE_INTELFPGA
unset CL_CONTEXT_EMULATOR_DEVICE_ALTERA

export ACL_DEFAULT_FLOW=top
export ACL_PCIE_JTAG_DEVICE_INDEX=2


export CL_CONTEXT_COMPILER_MODE_INTELFPGA=3
#export CL_CONTEXT_MPSIM_DEVICE_INTELFPGA=1
#unset CL_CONTEXT_COMPILER_MODE_INTELFPGA
```

图 5-15 init_env.sh 脚本编译选项配置图

（一）如果确认当前板卡烧写的工程即为要执行的工程或者已通过 aocl program 把工程烧写到板卡，不想 host 代码执行时重新执行 PR 烧写工程，可设置环境变量：

export CL_CONTEXT_COMPILER_MODE_INTELFPGA=3

如果想重新具备 PR 烧写工程的功能，需执行如下环境变量：

unset CL_CONTEXT_COMPILER_MODE_INTELFPGA

（二）如果通过 program 烧写的 aocx 文件与板卡 FPGA 芯片运行工程是基于同一个 BSP 编译生成，则执行 PR，否则会退化到直接烧写模式，称为 full-chip programming。需设置环境变量：

export ACL_PCIE_JTAG_DEVICE_INDEX=2

（三）默认设置都是以支持 PR 模式进行编译的，即：

export ACL_DEFAULT_FLOW=top

如果想以 flat 模式进行编译，需更改为如下环境变量：

export ACL_DEFAULT_FLOW=flat

6. 软件工具使用说明

BSP 软件部分提供了包括板卡温度测试、功耗测试、板卡 FRU 信息读取等一系列工具，下面对 BSP 提供的工具使用方法进行说明。在使用 BSP 附带各种工具之前需按上文所述安装 DEV 或 RTE Package，并完成环境变量设置。

6.1 诊断工具

BSP 提供的诊断测试工具 diagnose，用于对板卡工作是否处于正常状态进行诊断测试，返回板卡基本信息、DDR 工作状态等信息。

使用方法

执行时通过参数设置分为两种执行命令：

-aocl diagnose

-aocl diagnose <dev_name>

由 SDK 提供的 aocl 工具会调用 BSP 中编译生成的 diagnose 主机端程序，对板卡工作状态进行诊断测试。

aocl diagnose 用于测试并返回板卡工作状态信息；

aocl diagnose <dev_name>测试并返回 DDR 工作状态信息。<dev_name>是驱动识别到的 device_name 的简写，device_name 可以通过 ls /dev 命令查看。例如：device_name 为 aclf10a_sr2x8g0 简写为 acl0。

示例如下：

$aocl diagnose

$aocl diagnose acl0

6.2 下载工具

➢ program

program 工具用于加载 OpenCL 工程到 FPGA 芯片，有 pr 和 JTAG full-chip 两种模式。

两种模式区分依据芯片运行工程与加载到芯片工程是否为基于同一版本BSP编译自动区分。若基于同一 BSP 采用 pr 模式编译生成，执行 pr 功能，通过 PCIe 接口将工程加载到 FPGA 芯片运行，执行速度较快；若基于不同版本 BSP 或同一 BSP flat 模式编译生成，执行 JTAG full-chip 功能，使用外接于服务器的硬件烧写工具 blasterII 通过 jtag 接口将工程加载到 FPGA 芯片运行，执行速度慢。

使用方法

$aocl program <dev_name> <file.aocx>

<file.aocx>为 OpenCL 工程编译生成的后缀为 aocx 的设备端工程文件。

执行示例如下：

$aocl program acl0 boardtest.aocx

➢ download

通过 blasterII 加载 sof 工程文件到 FPGA 芯片的工具。直接调用 BSP 中编译生成的 download 主机端程序，利用 blasterII 将后缀为 sof 的文件通过 jtag 加载到名为<device_name>的设备，即 FPGA 芯片，从而避免 FPGA 芯片中正在运行工程时重新加载 sof 文件而导致的服务器重启问题。

使用方法

$download <device_name> <file.sof>

<file.sof>为 OpenCL 工程编译生成的后缀为 sof 的设备端工程文件

使用举例

$download aclf10a_sr2x8g0 jpeg_decoder.sof

注：program 和 dowanload 工具会首先检测 FPGA 芯片相应寄存器，若满足匹配条件：FPGA 运行基于浪潮 F10A 板卡 BSP 编译的工程才满足执行条件，否则执行失败。

➢ flash_program

通过 flash_program_linux.sh 脚本可以将 FPGA 固件和 CPLD 固件恢复到出厂设置。flash_program_linux.sh 脚本位于 $AOCL_BOARD_PACKAGE_ROOT/../flash_program 路径下。该路径下 boardtest.jic 即为 FPGA 固件，f10a_bmc.pof 即为 CPLD 固件。

使用举例

$cd $AOCL_BOARD_PACKAGE_ROOT/../../flash_program

$./flash_program_linux.sh

注意：因 QUARTUS19.1 不支持 Linux 系统下的 JIC 文件下载，若使用 flash_program 工具需首先安装 DEV 或 RTE Package 下的 QuartusProProgrammerSetup-17.1.0.240-linux.run，并将安装路径添加到 PATH 环境变量中。

6.3远程更新工具

为摆脱下载器的束缚，推荐使用远程更新的方法更新 FPGA 固件。远程更新可通过 PCIE 接口直接将 RPD 格式文件更新到 FPGA Flash。

6.3.1 使用主要步骤

用户使用流程如下图 6-1 所示：

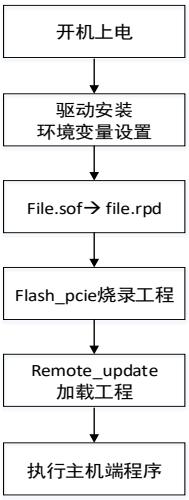

图 6-1 远程更新 flash 工具使用步骤

使用步骤具体如下（以更新 application，一机一卡为例）：

1） 完成上文所述环境搭建；

2） 生成用于更新的 application image <file.rpd>文件，参照“ 6.4.3 SOF 转 RPD 文件”；

3） 执行***$remote_update <dev_name> –l f***；

4） 执行***$flash_pcie <dev_name> <application_image>.rpd –a***，烧写时间大概为三分钟；

5） 烧写成功后，执行***$remote_update <dev_name> –l a***，加载新烧写的 application image

到 FPGA 芯片；

6） 执行主机端程序。

注意：

1） 驱动安装前确认驱动安装脚本（install）可执行权限；

2） 执行 flash_pcie 工具之前，先将 factory image 加载到 FPGA 芯片，即：执行步骤 3；

3） 默认的驱动兼容识别 application 和 factory，若 application 有单独驱动需重新安装；

4） 通过以下命令 rescan PCIE：

$echo 1 > /sys/bus/pci/device/0000\:<bus>.<device >.<function>/remove

$echo 1 > /sys/bus/pci/device/rescan

6.3.2 PCIE 远程更新工具说明

1、flash_pcie 工具

flash_pcie utility 将待烧写的文件 flash_ru.rpd 进行解析，并将其读取到内存，然后调用 MMD 的 FLASH 接口及驱动的 DMA 接口，将数据写入 FLASH。

使用方法：

$flash_pcie <dev_name> <rpd_file_name> [-a | f]

dev_name 为 f10a 板卡对应的 device name, 可以在 /dev 下查看到，以默认出厂驱动和一块<device_name>名称为aclf10a_sr2x8g0的板卡为例，可以使用acl0指代aclf10a_sr2x8g0。

rpd_file_name 为所要烧写的 application image 所在的文件，如 flash_ru.rpd，其为 flash 的一个镜像。application image 的起始地址为0x04000000。factory image 的起始地址为0x20。如果 rpd 文件不在当前目录，参数要指明其路径。

-a 更新 application image

-f 更新 factory image

使用举例：（flash_ru.rpd 为测试用例）

- 更新 application image：

```
$ flash_pcie acl0 flash_ru.rpd –a
INFO : Ver    : 1.1
INFO : Date : 19/08/13 16:00:00
INFO : Compile date Aug 13 2019 time 18:53:49.
INFO : Get memory and load app image:    0x7fcdaa041040.
```

INFO : Open PCIE DEVICE aclf10a_sr2x8g0.

INFO : Creact PCIE Flash object.

INFO : Start to flash program ...

INFO : Opening FLASH...

INFO : Unlocked flash sectors.

INFO : Image size is 64 MB, program to addr 0x4000000.Sector size 64 KB,page size 1024 Byte.

INFO : Erase sector num 1024.

INFO : Erasing sector[==========]100.0% [-]

INFO : Erasing flash complete. Time 82 s.

INFO : Swap img addr 0x7fcdaa041040 page_num 65536.

INFO : Programing application image ,The mode is IRQ.

INFO : Write page num 65536.

INFO : Write Page[==========]100.0% [-]

INFO : Programing flash complete. Time 66 s.

INFO : Closing FLASH...

INFO : Delete PCIE Flash object.

INFO : Close mmd handle.

INFO : Free app memory 0x7fcdaa041040 .

INFO : Flashing succeeded. You should run remote update tool to re-configure the FPGA from the new Flash image.

- 更新 factory image：

$ flash_pcie acl0 flash_ru.rpd –f

INFO : Ver : 1.1

INFO : Date : 19/08/13 16:00:00

INFO : Compile date Sep 10 2019 time 09:21:25.

INFO : Get memory and load app image: 0x7f1906332040.

INFO : Open PCIE DEVICE aclf10a_sr2x8g0.

INFO : Creact PCIE Flash object.

INFO : Start to flash program ...

INFO : Opening FLASH...

INFO : Unlocked flash sectors.

INFO : Image size is 64 MB, program to addr 0x0.Sector size 64 KB,page size 1024 Byte.

INFO : Erase sector num 1024.

INFO : Erasing sector[==========]100.0% [-]

INFO : Erasing flash complete. Time 78 s.

INFO : Swap img addr 0x7fcdaa041040 page_num 65536.

INFO : Programing application image ,The mode is IRQ.

INFO : Write page num 65536.

INFO : Write Page[==========]100.0% [-]

INFO : Programing flash complete. Time 56 s.

INFO : Closing FLASH...

INFO : Delete PCIE Flash object.

INFO : Close mmd handle.

INFO : Free app memory 0x7f1906332040 .

INFO : Flashing succeeded. You should run remote update tool to re-configure the FPGA from

2、remote_update 工具

remote_update 功能为将 FLASH 内指定的 image 加载到 FPGA 内。FLASH 内默认有两个 image。Factory image 起始地址为 0x20。Application image 起始地址为 0x04000000。在加载完成后（重启），此工具也可以查询加载（reconfiguration）的状态，判断是否加载成功。

使用方法：

$remote_update dev_name [-c | -l] [a|f]

dev_name 为 f10a 板卡对应的 device name，可以在 /dev 下查看到。

-c 选项为 check reconfiguration status。

-l a 选项为加载 application image

-l f 选项为加载 factory image

-c –l 二选项有且只有一个

使用举例：

开机首次查看状态：

$remote_update aclf10a_sr2x8g0(acl0) –c

INFO : Ver :1.0

INFO : Date: 19/07/05 16:39:00

INFO : Complile date Sep 1 2019 time 10:19:01.

checking RU reconfigure status...

INFO : the reconfiguration trigger conditions is 0.

MMD INFO : No reconfiguration status.

加载 application image：

$remote_update aclf10a_sr2x8g0(acl0) –l a

INFO : Ver :1.0

INFO : Date: 19/07/05 16:39:00

INFO : Complile date Sep 1 2019 time 10:19:01.

Start to load the application image at 0x4000000 ...

INFO : the watchdog timeout value is fff.

INFO : the reconfiguration trigger conditions is 4.

INFO : The boot address is 4000000

MMD INFO : Loading image ...

New image loaded.

加载 factory image：

$remote_udpate aclf10a_sr2x8g0(acl0) –l f

INFO : Ver :1.0

INFO : Date: 19/07/05 16:39:00

INFO : Complile date Sep 1 2019 time 10:19:01.

Start to load the application image at 0x20 ...

INFO : the watchdog timeout value is fff.

INFO : the reconfiguration trigger conditions is 0.

INFO : The boot address is 20

MMD INFO : Loading image ...

New image loaded.

完成 remote_update 加载后查看状态：

$remote_update aclf10a_sr2x8g0 –c

INFO : Ver :1.0

INFO : Date: 19/07/05 16:39:00

INFO : Complile date Sep 10 2019 time 09:21:25.

checking RU reconfigure status...

MMD INFO : the reconfiguration trigger conditions is 4.

MMD INFO : Configuration reset triggered from logic array.

如果加载 application image 时发生错误（CRC ERROR），将会尝试加载三次，如果还错误，将自动加载 factory image。

6.3.3 远程更新 SOF 转 RPD 文件说明

1、flash 空间分配、rpd 文件介绍

RPD 文件是 Intel 推荐的第三方烧写工具使用的文件，由 sof 文件利用 Quartus 软件转化得到。其格式为二进制文件，内容为 FLASH 的全部镜像。其大小等同于板卡 FLASH 的大小（128MB）。

板卡 FLASH 地址空间的分配如下图 6-2 所示：

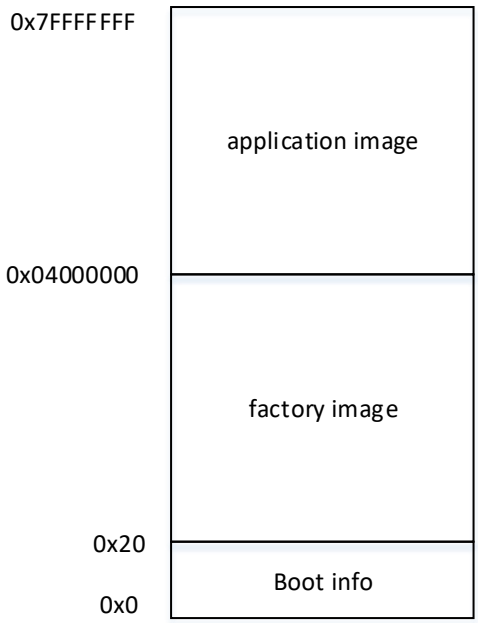

图 6-2 RPD 文件地址空间分配图

0x0-0x1f 为 boot info，其包括默认的加载镜像地址。现配置的默认镜像地址为 application image 地址 0x04000000。

我们将剩余部分划分为两块 image，factory image 和 application image，分别独立存储两个工程：

- 0x20-0x03ffffff 为 factory image。Factory image 出厂时已经烧好，其标志为 PCIE 设备的 revision id 为 1。其用于在 application image 出现错误时，将 FPGA 恢复到出厂的默认配置。在 Factory image 配置下，用户可以烧写 application image 和加载新的 image。在 application image 下也可以直接切换到 factory image。Factory image 默认永久保留在 flash 内，不应被擦除。
- 0x04000000-0x07ffffff 为 application image。在出厂时，application image 也已经烧好，其标志为 PCIE 设备的 revision id 为 4。每次断电重启后，FPGA 板卡默认加载 application image。用户可以烧写新的 application image。

在使用 Quartus 工具将 sof 文件转化为 rpd 文件时，将起始地址手动设定为 0x04000000，flash_pcie 工具针对 file.rpd 待烧录文件只读区 0x04000000—0x07FFFFFF 地址的数据，将该数据烧录到板卡 FLASH application image 中对应地址。RPD 文地址空间分配如下图 6-3：

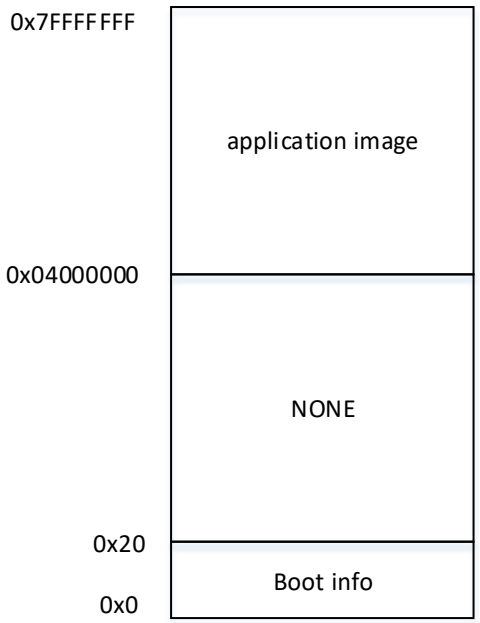

图 6-3 RPD 文件地址空间分配图

2、SOF 转 RPD 文件步骤

SOF 文件转 RPD 文件借助于 Quartus 工具，分为两步进行：

1） 首先根据 sof 文件生成 pof 文件

从“file”中打开 Quartus 的“convert programming file”工具；首先设置“Configuration device”为“EPCQL 1024”，设置 Mode 为 “Active Serial x4” ；然后点击“Add file”按钮加载相应 sof 工程文件到 SOF Data 后，按如下图 6-4 所示进行配置。

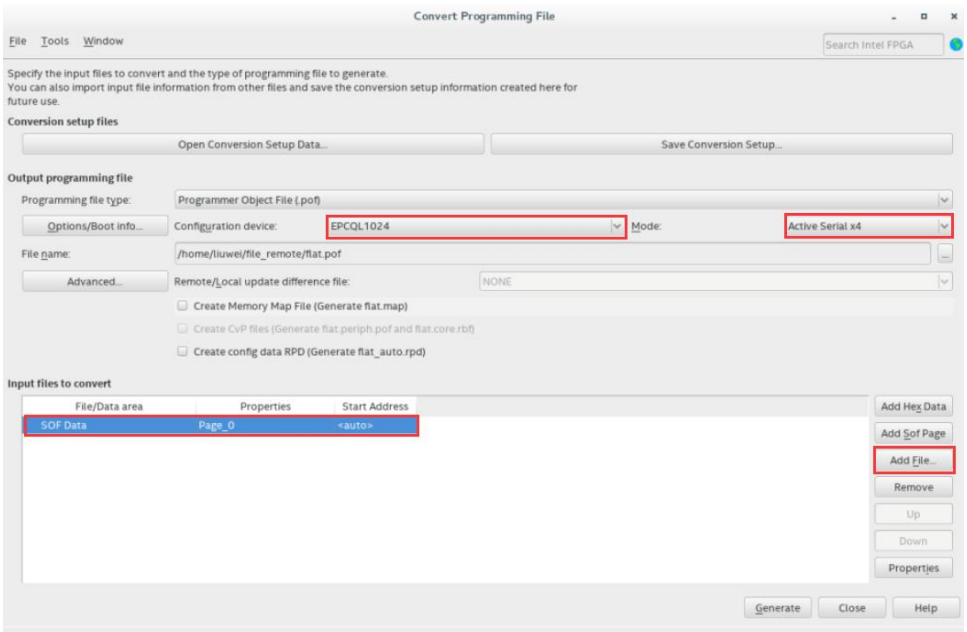

图 6-4 Quartus 配置图

选中需要加载的 flat.sof 文件，单击“open”，如下图 6-5 所示。

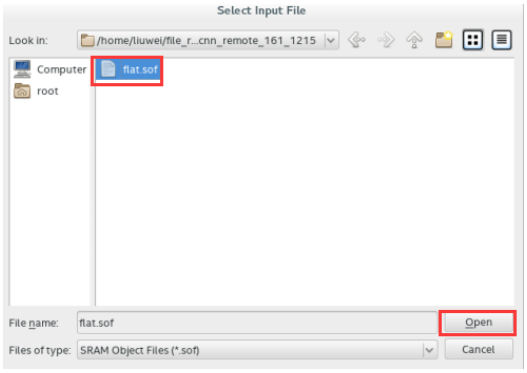

图 6-5 sof 文件选择

打开 flat.sof 之后界面如下，选择 SOF Data 一栏，然后单击“Properties”按钮，对其进行设置 Properties；最后单击“Options/Boot info”、“Advanced…”设置 “Options/Boot info”、“Advanced option”，如下图 6-6 所示。

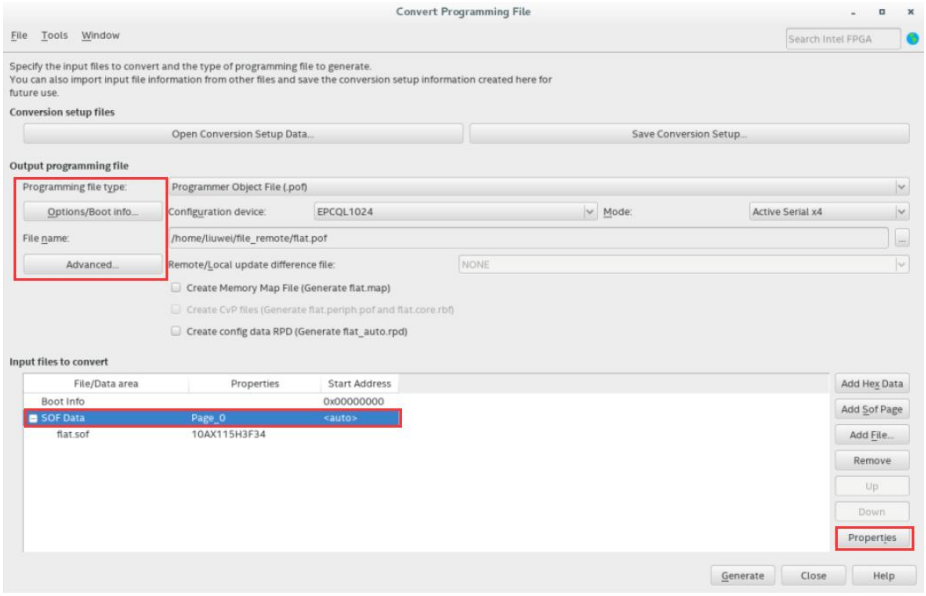

图 6-6 文件转换配置图

Properties 设置如下，将左图默认设置修改为右图设置，如下图 6-7 所示。

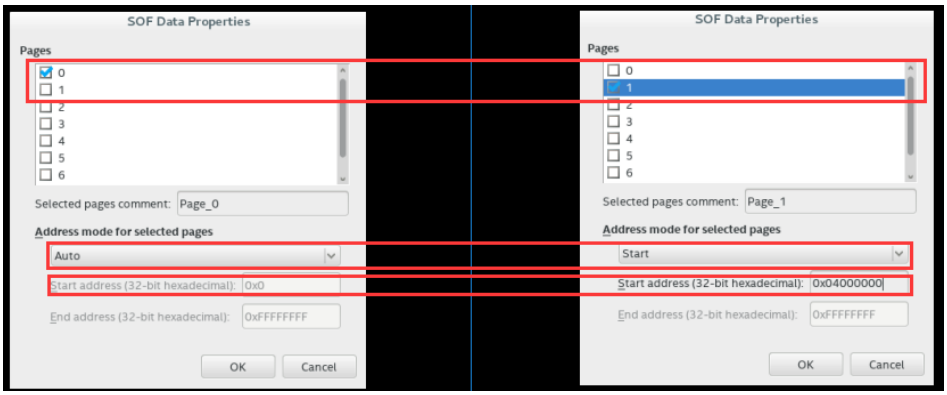

图 6-7 Properties 配置图

“Options/Boot info”、“Advanced option” 设置如下图 6-8 所示。

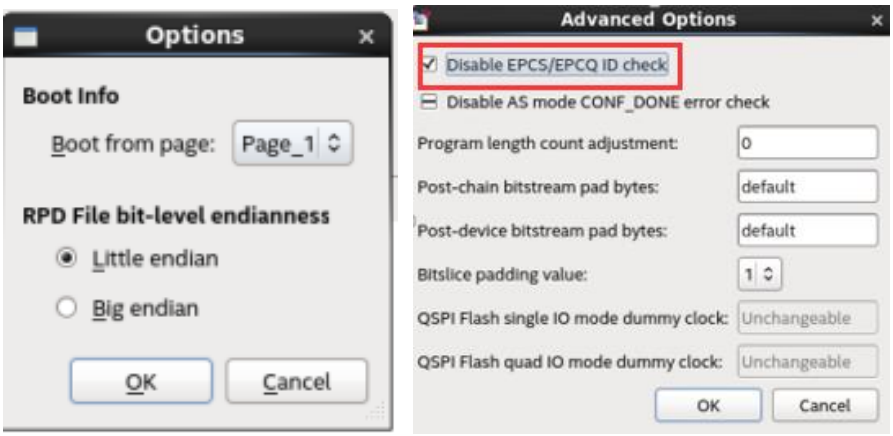

图 6-8 Advanced option 配置图

设置完成后界面显示如下，点击“Generate”按钮生成 pof 文件，如下图 6-9 所示。

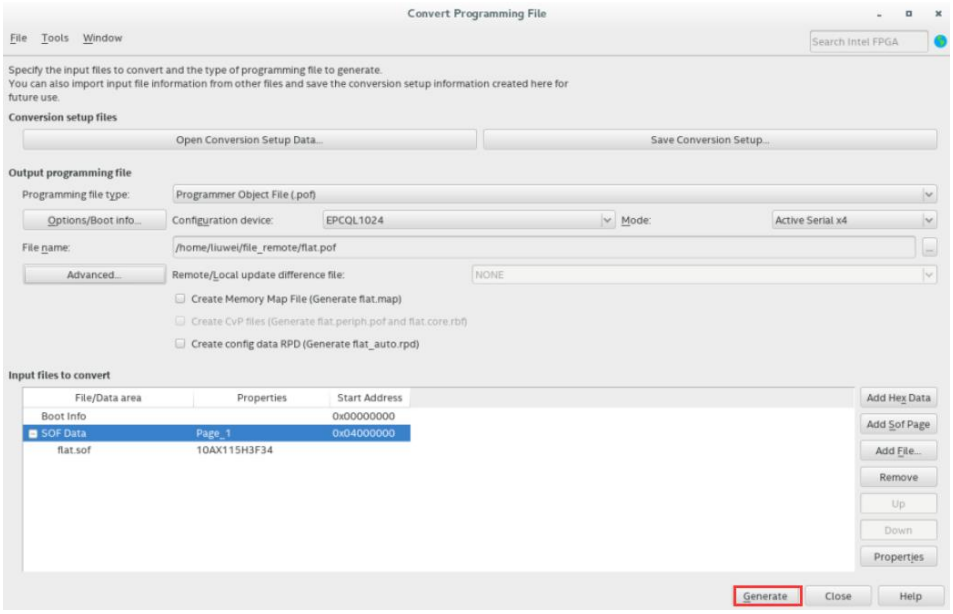

图 6-9 pof 文件生成图

注：以上为 application image 更新文件 sof 转 pof 转换过程，factory image 更新与 application 更新略有不同，区别仅存在于 sof 转 pof 文件时 Quartus 图形界面参数设置不同，具体参数设置如下图 6-10 所示：

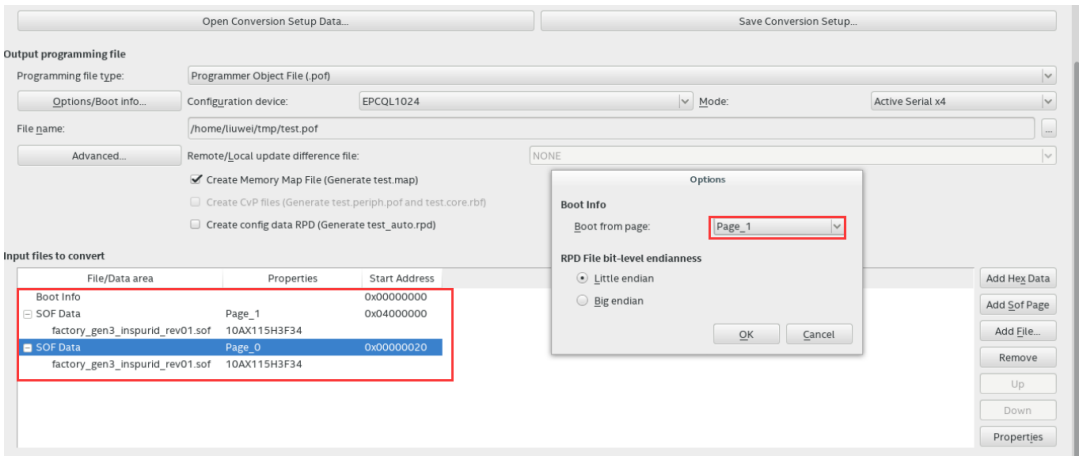

图 6-10 factory image 文件转换配置图

2）将 pof 文件转换成 rpd 文件。

首先在“convert programming file”界面中从“Programming file type”中选择 rpd，如下图 6-11 所示。

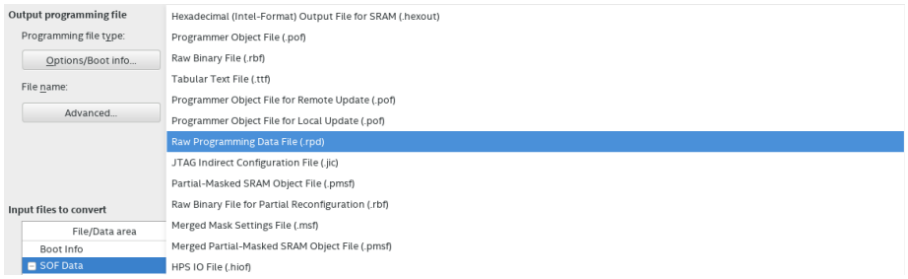

图 6-11 rpd 文件转化配置图

然后单击“Add file”按钮将上一步生成的“flat.pof”加载进来，完成后界面如下图 6-12 所示。之后在界面中依次设置 Mode 为 “Active Serial x4”；单击“Options/Boot info”、“Advanceded…”设置 “Options/Boot info”、“Advanced option”。

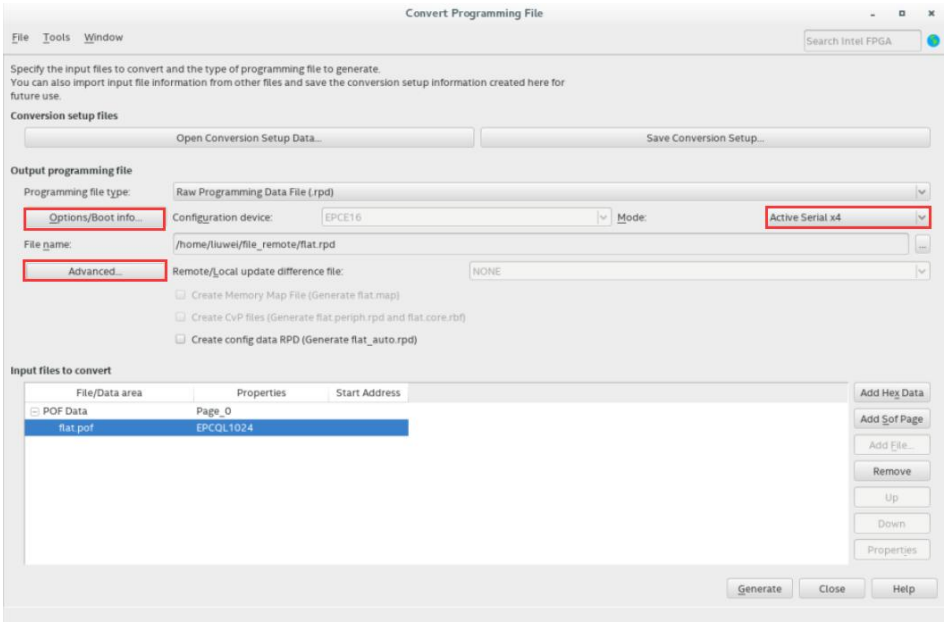

图 6-12 rpd 文件转换配置图

“Options/Boot info”、“Advanced option” 设置如下图 6-13 所示。

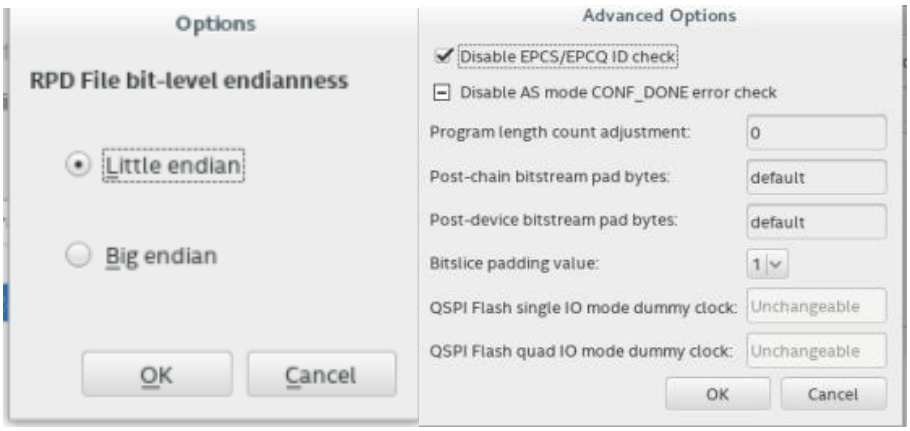

图 6-13 Advanced Option 配置转换图

设置完成后点击“Generate”按钮生成 RPD 文件，成功后弹出对话框。

6.4板卡 FRU 信息读取

该版本 DEV 和 RTE Package 支持对 FLASH、EEPROM 中板卡 FRU 信息的读取，其中 FLASH 中存放了 PN、SN 两种 FRU 信息，EEPROM 中存放 MAC1、MAC2、PN、SN 四种 FRU 信息。目前可支持同时读取三张板卡的 FRU 信息。

6.4.1 读取 FLASH 中 FRU 信息

首先切换到读取 FRU 信息脚本所在路径，然后执行该路径下的脚本，具体 FLASH 中的 FRU 信息读取指令如下：

$cd $AOCL_BOARD_PACKAGE_ROOT/../diag

$./fru_f10a.py f r 【参数一】

参数一：选择板卡，最多可支持服务器同时插三张板卡：第一张：0x00；第二张：0x01；第三张：0x02；

读取 0x00 板卡 FLASH 中 FRU 信息操作示例如下图 6-14 所示：

```
[chris@localhost diag]$ ./fru_f10a.py f r 0x00
INFO : Ver  : 0.1
INFO : Date : 19/10/21 16:00:00
INFO : Compile date Dec 24 2019 time 19:30:19.
INFO : Open PCIE DEVICE  aclf10a_sr2x8g0.
INFO : Creact PCIE Flash object.
INFO : Start to read fru information ...
INFO : Opening FLASH...
INFO : Swap img addr 0x1a82c00 page_num 1.
INFO : CRC check success!
INFO : The Product Part Number is: YZPC-00911-101.
INFO : The Product Serial Number is: QTJ827EC0134A10.
INFO : Delete PCIE Flash object.
```

图 6-14 读取 FLASH 中 FRU 信息示例图

6.4.2 读取 EEPROM 中 FRU 信息

目前仅在浪潮 NF5280M5 服务器上支持读取 EEPROM 中板卡 FRU 信息。在读取之前首先要到*$AOCL_BOARD_PACKAGE_ROOT/../BMC_firmware* 路径下获取 BMC 固件，并将服务器 BMC 更新到 4.26.0 版本。

具体读取操作如下，首先切换到读取板卡 FRU 信息脚本所在路径，然后执行该路径下的脚本，具体 EEPROM 中的 FRU 信息读取指令如下：

$cd $AOCL_BOARD_PACKAGE_ROOT/../diag

$sudo ./fru_f10a.py e r 【参数一】

参数一：选择板卡，最多可支持服务器同时插三张板卡：第一张：0x00；第二张：0x01；第三张：0x02；

读取 0x00 板卡 EEPEOM 中 FRU 信息操作示例，如下图 6-15：

```
[chris@localhost diag]$ sudo ./fru_f10a.py e r 0x00
[sudo] password for chris:
MAC_ADDR1:6C92BF111111
MAC_ADDR2:6C92BF111112
Product Part Number:YZQT-00911-101
Product Serial Number:AAAAADDDDDCCCCC
```

图 6-15 读取 EEPROM 中 FRU 信息示例图

6.5板卡温度、功耗获取

板卡环境搭建完成之后，可以通过内核获取板卡的温度和功耗信息，且支持同时读取三张板卡信息。

6.5.1 读取板卡功耗信息

读取功耗信息指令：***$power_temper_get power***，如下图 6-16 所示。

```
[chris@localhost f10a_rte_package_V19.1.0.1]$ power_temper_get power
--------------------------------

The Board0 Power:28 W.

--------------------------------
```

图 6-16 功耗读取示例图

6.5.2 读取板卡温度信息

读取功耗指令：***$power_temper_get temper***，如下图 6-17 所示。

```
[chris@localhost f10a_rte_package_V19.1.0.1]$ power_temper_get temper
-----------------------------------------------------

The Board0 Temperature:37 degrees C.

-----------------------------------------------------
```

图 6-17 温度读取示例图

6.6板卡功能测试

6.6.1 板卡基本功能测试

板卡出厂自带 boardtest 工程，运行$AOCL_BOARD_PACKAGE_ROOT/../demo/boardtest/下的 boardtest_host 可进行 boardtest 测试，如内存读写、带宽等等（也可更新板卡工程然后再运行主机端工程），具体操作指令如下。

$cd $AOCL_BOARD_PACKAGE_ROOT/../demo/boardtest

$aocl program <dev_name> boardtest.aocx

$./boardtest_host

具体操作示例如下图 6-18 所示。

```
[chris@localhost boardtest]$ ./boardtest_host
Boardtest usage information
Usage: boardtest_host [--device d] [--test t]
  --device d: device number (0 - NUM_DEVICES-1)
  --test t: test number (0 - 7)
  (default is running all tests on all devices)
Total number of devices = 1.
Running all tests.
Running on all devices.
Program object created for all devices.
Program built for all devices.

*********************************************************************
********************** TEST FOR DEVICE 0 ***********************
*********************************************************************


*********************************************************************
********************** Host Speed Test ************************
*********************************************************************

clGetDeviceInfo CL_DEVICE_GLOBAL_MEM_SIZE = 17179868160
clGetDeviceInfo CL_DEVICE_MAX_MEM_ALLOC_SIZE = 17179868160
Actual maximum buffer size = 17179868160 bytes
Writing 16383 MB to global memory ... 6007.319858 MB/s
Reading 17179868160 bytes from global memory ... 6129.093073 MB/s
Verifying data ...
Successfully wrote and readback 16383 MB buffer

Transferring 8192 KBs in 256 32 KB blocks ...
Transferring 8192 KBs in 128 64 KB blocks ...
Transferring 8192 KBs in 64 128 KB blocks ...
Transferring 8192 KBs in 32 256 KB blocks ...
Transferring 8192 KBs in 16 512 KB blocks ...
Transferring 8192 KBs in 8 1024 KB blocks ...
Transferring 8192 KBs in 4 2048 KB blocks ...
Transferring 8192 KBs in 2 4096 KB blocks ...
Transferring 8192 KBs in 1 8192 KB blocks ...

PCIe Gen2.0 peak speed: 500MB/s/lane
```

图 6-18 boardtest 测试示例图

如果在执行主机端工程时出现如下图 6-19 错误，说明是环境搭建过程中影响了之前编译的主机端工程，只需重新执行***$make –f Makefile.linux*** 指令编译主机端工程即可。

```
[chris@localhost boardtest]$ ./boardtest_host
Boardtest usage information
Usage: boardtest_host [--device d] [--test t]
  --device d: device number (0 - NUM_DEVICES-1)
  --test t: test number (0 - 7)
  (default is running all tests on all devices)
Warning: Cannot find any Intel(R) FPGA Board libraries. No Intel(R) FPGA devices will be loaded. Please contact yo
Total number of devices = 1.
Running all tests.
Running on all devices.
  OpenCL Notification Callback: clCreateProgramWithBinary failed
Failed clCreateProgramWithBinary.
Error code: -42
```

图 6-19 主机端工程错误示例

6.6.2 PR 测试

利用 program 工具，将 matrix_mult.aocx 工程更新到板卡中并运行主机端工程进行测试，完成 PR 功能验证。具体操作步骤如下：

$cd $AOCL_BOARD_PACKAGE_ROOT/../demo/matrix_mult/bin

$aocl program <dev_name> matrix_mult.aocx

$cd $AOCL_BOARD_PACKAGE_ROOT/../demo/matrix_mult

$./host

示例如下图 6-20 所示：

```
[chris@localhost matrix_mult]$ ./host
./host Starting...


===== Host-CPU checking the systolic array matrix multiplication parameters ======

HA:             256
WA:             4096

HB:             4096
WB:             256

HC:             256
WC:             256

PE_ROWS:                2
PE_COLS:                2
DOT_PROD_VECTOR_SIZE:           8

ACCUM_SHIFT_REG_SIZE:           1024

ROWS_INTERLEAVED:               32
COLUMNS_INTERLEAVED:            32
```

图 6-20 PR 功能测试示例图

注意：如果主机端功能运行失败，需要在$AOCL_BOARD_PACKAGE_ROOT/../demo/matrix_mult/路径下重新执行 make 指令对主机端工程进行编译即可！

7. 板卡 BMC 监控管理

7.1板卡 BMC 主要特点

- 支持PWM控制风扇转速
- 支持读取EEPROM的信息
- 支持读取FPGA温度信息
- 支持读取FPGA功耗信息
- 支持FPGA的reconfigure（热重配，从flash重新加载）
- 支持通过BMC slave设备去读取其他设备信息

7.2板卡 BMC 工作原理

BMC 系统的总体结构如图 7-1 所示，主机端的 BMC 为主设备，F10A 加速卡上的 BMC 为从设备，两者通过 PCIe SMBUS 协议进行通信。加速卡上的 BMC 在接收到主机端所发送的命令后，对接收到的命令进行解析，并返回主机端所需的信息。

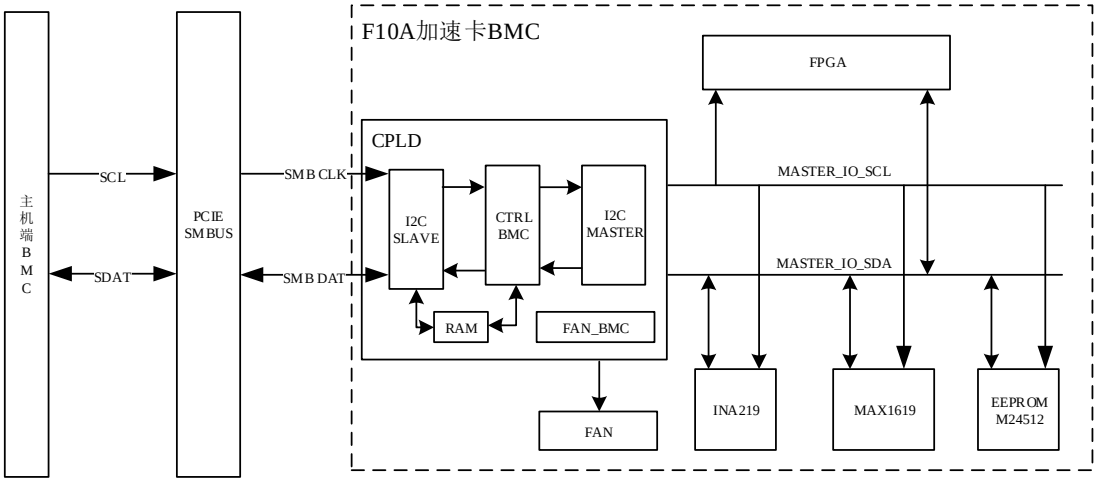

图 7-1 BMC 系统结构

7.3FPGA 加速卡设备地址

主机端访问 FPGA 加速卡时的设备地址（板卡 ID）是 0x55 或 0x54（由 SW2 拨码开关决定，“0”对应 ID 0x55，“1”对应 ID 0x54），卡上其他设备地址（ID）如表 1。板卡 BMC 通过 I2C 总线连接板卡上其他设备，作为板卡上其他设备的 MASTER，BMC 收到并解析主机端发来的命令后，然后去访问这些设备，通过设备地址（ID）和寄存器地址来确定具体访问对象。

表 7-1 FPGA 加速卡各设备地址

I2C DEVICE	ADDR (DEVICE ID)
FPGA加速卡	0x55或0x54（由SW2拨码开关决定）
EEPROM	0x57
MAX1619	0x4C
INA219	0x40

7.4板卡 BMC 命令使用

通过 ipmitool 工具配合主机端 BMC 可获取板卡温度、功耗、芯片温度等信息。本功能只能在浪潮的 5280M4、5280M5 机型的服务器上才能使用，其他厂商的服务器或浪潮的其他机型服务器暂不支持此项功能。使用之前需要更新服务器的 BMC 到指定版本 4.26.0。

主机端读 FPGA 温度、功耗、板卡温度

通过 ipmitool 指令可以读取板卡的功耗和温度，且同时可以支持读取三个板卡的功耗和温度值。具体指令如下：***$sudo ipmitool sdr | grep -i FPGA***

读取示例参见下图 7-2 所示。

```
[chris@localhost dev]$ sudo ipmitool sdr | grep -i FPGA
[sudo] password for chris:
FPGA0_Chip_Temp  | 47 degrees C      | ok
FPGA1_Chip_Temp  | no reading        | ns
FPGA2_Chip_Temp  | no reading        | ns
FPGA0_Card_Temp  | 37 degrees C      | ok
FPGA1_Card_Temp  | no reading        | ns
FPGA2_Card_Temp  | no reading        | ns
FPGA0_Total_Pow  | 27 Watts          | ok
FPGA1_Total_Pow  | no reading        | ns
FPGA2_Total_Pow  | no reading        | ns
[chris@localhost dev]$
```

图 7-2 BMC 读取板卡信息示例图

8. 常见故障分析

8.1板卡识别故障分析

如果加速卡安装后系统识别失败，可能原因及解决办法如下表所示：

可能原因	解决办法
FPGA内没有工程	通过BlasterII由JTAG烧写工程文件到Flash
加速卡与主板PCIE插槽接触不良	对服务器掉电之后，重新拔插加速卡，并进行有效的固定
主板PCIe扫描时间小于100ms	reboot热重启机器，更换搭载机型
Ubuntu系统非root用户	重启后以root用户登录
加速卡在运输或使用过程中有损坏	联系浪潮售后服务人员

8.2Blaster 识别故障分析

可能原因	解决办法
驱动、工具安装问题	Windows下请检查驱动是否安装 推荐使用软件版本为17.1版本
加速卡与主板PCIE插槽接触不良	对服务器掉电之后，重新拔插加速卡，并进行有效的固定
blaster II信号不稳定情况	使用blaster II下载器
非root用户登录	重启后以root用户登录 or添加文件51-usbblaster.rules文件到/etc/udev/rules.d目录下
加速卡在运输或使用过程中有损坏	联系浪潮售后服务人员

8.3诊断工具 diagnose 运行故障分析

可能原因	解决办法
驱动未安装	aocl install命令重新安装驱动
存在同类型其他驱动相互覆盖	卸载其他同类型驱动

8.4环境配置、驱动安装故障分析

可能原因	解决办法
拷贝丢失执行权限	重新增加执行权限，参考命令chmod +x *.sh
Quartus工具未安装	安装Quartus工具
存在同类型其他驱动相互覆盖	卸载其他同类型驱动

8.5如何获得技术支持服务

1）如需要保修服务，您可直接与产品销售商或拨打浪潮服务热线电话 400－860－0011 与我们联系。

2）登录浪潮网站 www.inspur.com 获取驱动与产品手册相关支持。

3）发送邮件到 FPGA@inspur.com 咨询获取支持。