

Altera 新的器件配置模式——协议实现配置 (CvP)，通过结合 PCI Express®来配置 Altera 28-nm Arria® V、Cyclone® V 和 Stratix® V FPGA 的内核架构。CvP 能够降低产品成本，减小电路板面积，同时简化了软件应用模型，具备可靠的现场系统更新功能。此外，嵌入式自治 PCIe IP 内核有助于确保设计满足 PCIe 上电时序要求，FPGA 内核架构配置时间对其没有影响，保证了各种基于 PCIe 计算机平台上广泛的互操作性。

引言

PCIe 技术替代了 PCI 成为处理器和被监控设备之间的标准控制平面接口。自从 2005 年推出以来，FPGA 设计人员在 FPGA 和处理器之间已经广泛使用了 PCIe 接口。现在的 FPGA 包括嵌入式 PCIe 内核，它用作端点或者根端口。

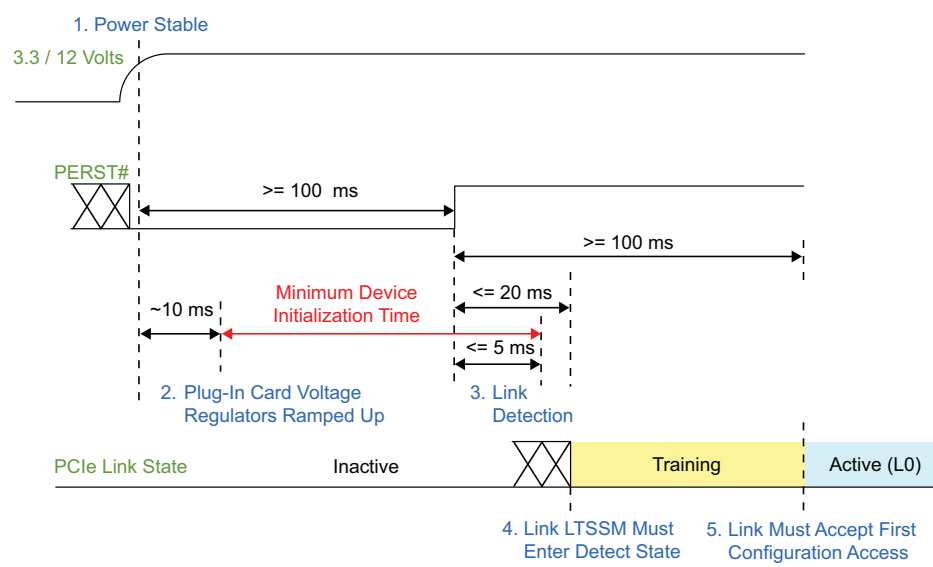
直到最近，在对 FPGA 进行全面配置之前，嵌入式 PCIe 内核还不支持链路训练和总线枚举功能。随着器件密度的提高，FPGA 配置时间也越来越长，很难在 PCIe 规范要求的初始化时间之内完成 FPGA 的全面配置。

在 28-nm 系列器件推出后，Altera 支持从 FPGA 内核逻辑中单独配置 PCIe 硬核 IP，从而解决了这一问题。该技术支持设计人员通过 PCIe 来配置 Altera Arria V、Cyclone V 和 Stratix V FPGA 内核架构。CvP 器件编程新方法能够降低产品成本，减小电路板面积，同时简化了软件应用模型，具备可靠的现场系统更新功能，如下所述：

- 降低系统成本——CvP 避免了采用一个或者多个并行闪存器件，甚至是外部编程控制器件。而且，CvP 支持设计人员将 FPGA 编程文件存储在通过 PCIe 链路与 FPGA 连接的 CPU 存储器系统中。使用这一技术，只有 FPGA I/O 编程和 PCIe 内核参数存储在闪存器件中，只需要更小更便宜的闪存器件。
- 减少了对 FPGA 资源的占用——Stratix 系列器件一般需要数据通路较宽的闪存器件来存储 FPGA 编程文件。相反，CvP 支持的 EPCS 和 EPCQ 器件需要较少的专用引脚。
- 节省能源——可以根据用户应用参数，通过软件控制来装入低功耗临时 FPGA 镜像。这一特性适用于电池供电的计算机系统。

图 1 简要介绍了 *PCI Express 基本规范 Gen1 1.0a 和 1.1* 以及 *PCI Express 基本规范 Gen2 2.0* 的 PCIe 上电排序。分配给器件初始化和器件训练的最短时间是 200 ms（等于图 1 中第 5 点和第 1 点之间的时间差）。分配给器件初始化的最短时间由图 1 中第 3 点和第 2 点之间的时间差来描述，大约是 95 ms。

图 1. PCIe 上电时序波形



FPGA 器件在很小的体积中封装了更多的逻辑，因此，需要更多的时间采用专用内容对较大的 FPGA 内核架构进行编程。在大型器件中，总配置时间会超过 95 ms。当端点器件没有达到 PCIe 范分配给它的 L0 时间时，端点不会响应软件配置访问操作（第 5 点），主机 CPU 将不能识别这一端点。在这种情况下，主机 CPU 可能会忽略端点，系统工作在没有端点的状态下。

自治 PCIe 硬核 IP

为克服这一失败发现机制，Altera 28-nm FPGA 支持在全面配置 FPGA 内核架构之前进行嵌入式 PCIe 内核操作。通过在不到 95 ms 内初始化嵌入式 PCIe 内核以及器件 I/O 环，28-nm FPGA 嵌入式 PCIe 内核总是能够满足 PCIe 上电时序要求。这种单独的可配置嵌入式 PCIe IP 内核被称为“自治”。


以下序列定义了 PCIe 的 CvP 初始化周期：

1. 通过 PERST# 使嵌入式 PCIe 内核保持复位，在第 3 点开始 PCIe 链路发现和训练之前释放。
2. 在 PCIe 链路完成训练阶段，达到 L0 状态后，FPGA 内核架构的其他部分开始编程。
3. 嵌入式 PCIe 端点内核到达 L0 状态后，主机操作系统 (OS) 开始访问 PCIe 内核的配置空间寄存器 (CSR)，完成配置写访问周期，它是系统初始化和发现过程的一部分（第 5 点）。
4. 对于 FPGA 内核架构没有完全按照设计人员应用内容进行编程的情况（换言之，还没有达到“用户模式”），自治 PCIe 内核以配置重试状态 (CRS) 操作进行响应，直至 FPGA 内核架构被全部装入。
5. OS 正确的找到这一端点，尝试再次查询它直到正常工作为止。
6. 在 OS 确定端点失效之前，端点可以响应 CRS 一秒钟。换句话说，在这一器件初始化模式中，分配给 FPGA 内核架构编程的时间不能超过一秒钟。

PCIe 总线上的 FPGA 内核架构编程

以下章节详细介绍了 CvP 和 PCIe 的使用。设计人员可以使用 CvP 通过 PCIe 装入最初的 FPGA 内核架构镜像，然后，在运行时修改这一内核架构镜像，以满足应用需求。对 PCIe IP 内核外设进行编程后，链路训练进入相应的 PCIe 工作模式。

PCIe 链路完成训练并到达 L0 状态后，主机 CPU 可以通过 PCIe 对 FPGA 内核架构镜像进行编程。

 每片 28-nm FPGA 只有一个嵌入式 PCIe 内核能够进行 CvP，也只适用于用作端点的情况。

通过 PCIe 配置 FPGA 内核架构期间，所有非串化器 / 解串器 (SERDES) I/O 引脚被内部弱上拉电阻置于高电平状态。在装入 CvP 内核架构镜像期间，所有其他高速 SERDES 引脚实际上处于复位状态。之所以这样设计这些 I/O 分配，是为了在更新 FPGA 内核架构配置期间保持 I/O 工作状态不变。使能 CvP 后，自治 PCIe 内核并不以 CRS 操作进行响应，而是接受并响应 PCIe 配置和数据操作，以完成 FPGA 内核架构配置。

表 1. 28-nm FPGA 配置模式

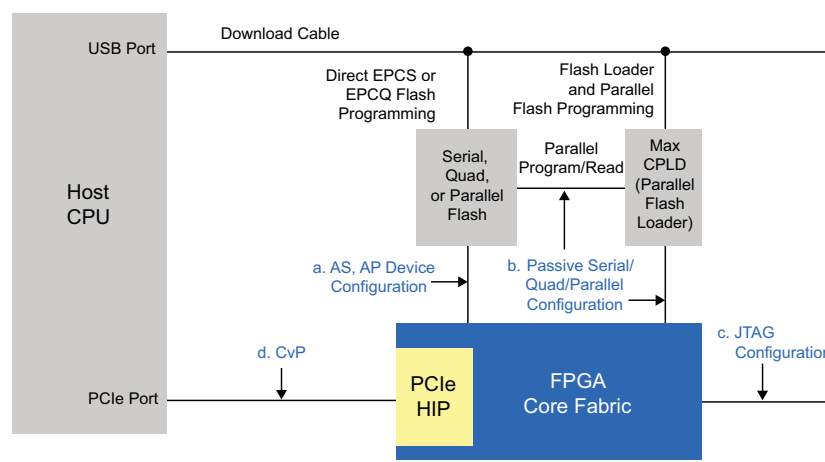
模式	状态	数据宽度 (位)
主动串行 (AS)	已有	1
主动四路 (AQ)	新	4
被动串行 (PS)	已有	1
8 位快速被动并行 (FPP)，使用闪存装入器。	已有	8
4、16 或者 32 位 FPP，使用闪存装入器。	新	4, 16, 32 ⁽¹⁾
基于 JTAG	已有	专用 JTAG 端口
CvP 使用 PCIe	新	1, 2, 4, 8 ⁽²⁾

注释:

- (1) 仅 Stratix V 器件。
- (2) PCIe 链路中的通路数量 (Gen1 x1、x2、x4 或者 x8； Gen2 x1、x2、x4 或者 x8；)

图 2 显示了 Stratix V 配置模式和闪存编程方法的高级表示。为简化这一图形表示，结构图中组合了所有闪存模式。

图 2. 器件配置和闪存编程模式



注释:

- (1) 四种可能的 FPGA 配置模式 (a) 主动串行 (x1, x4)。 (b) 被动串行 / 并行 (x1, x4, x8, x32), 使用 Altera MAX[®] CPLD 或者其他逻辑来读取闪存, 配置 FPGA。 (c) 用于调试目的的 JTAG 配置 (不需要外部闪存来配置 FPGA)。 (d) 仅 FPGA 内核架构的 CvP。 通过其他方法首先配置 PCIe 硬核 IP (HIP) 和 I/O 环。
- (2) 可以通过下载电缆直接编程 Altera EPCS (串行) 或者 EPCQ (四路)。
- (3) 对于使用 MAX CPLD 对闪存进行编程的情况, MAX CPLD 读取闪存, 配置 FPGA。

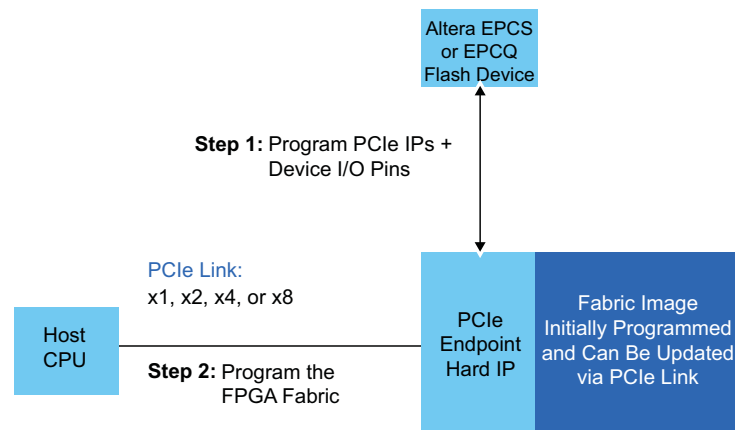
使用 CvP 和 PCIe 时, 通过现有器件初始化模式对各种 PCIe IP 内核参数以及每一高速收发器 (SERDES) 功能进行编程, 如表 1 所示。可以使用 Altera 串行或者四路闪存器件实现低成本产品解决方案 (不是强制使用), 只存储与 I/O 环和 PCIe 硬核 IP 相关的初始化比特。表 2 显示了怎样通过一种器件配置模式来装入两个比特 (表 1 所列), 确定 CvP 功能。这两个比特是 `cvp_enabled` 和 `full_chip_initialization`。

表 2. CvP 工作模式

CvP 模式编号	CvP 模式比特		FPGA 配置方法
	CvP 使能	全芯片初始化	
1	0	1	关掉 CvP。通过标准配置模式进行全芯片初始化。CvP 不能用于更新 FPGA 内核架构镜像。
2	1	0	打开 CvP。通过标准配置模式, 仅初始化 PCIe 硬核 IP、FPGA I/O 和收发器。CvP 先配置 FPGA 内核架构。CvP 也可以用于更新 FPGA 内核架构镜像。
3	1	1	打开 CvP。通过标准配置模式进行全芯片初始化。CvP 可以用于更新 FPGA 内核架构镜像。

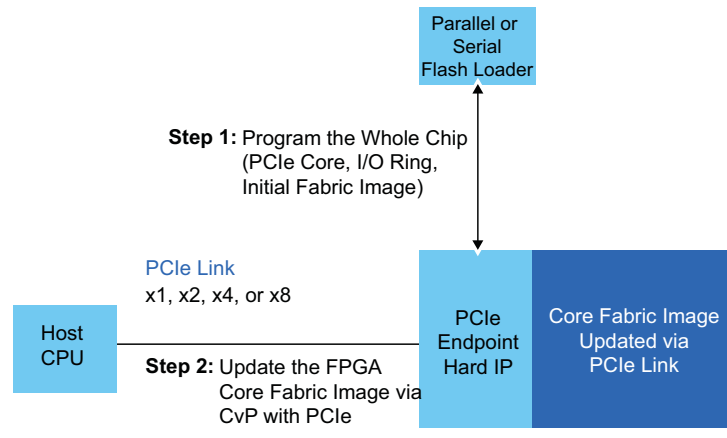
图 3 描述了一种可能的 CvP 工作模式, 它采用了低成本 Altera 闪存, 还描述了表 2 中的模式 2。

图 3. 采用了 Altera 闪存的自治 PCIe 硬核 IP 内核和 CvP



无论比特设置如何，嵌入式 PCIe 内核总是保持自治。换句话说，在配置 FPGA 内核架构之前，它唤醒并开始链路训练。打开 `cvp_enabled`，嵌入式 PCIe 内核到达 L0 状态后，使能 CvP。然后，应用软件发出指令后，主机 CPU 配置 FPGA 内核架构镜像。图 4 介绍了表 2 中的模式 3。

图 4. CvP 模式，采用了 Altera 闪存装入器



器件编程所使用的 PCIe 端点链路工作模式与完成 CvP 后 FPGA 目标应用程序所使用的模式相同。例如，设计人员可以将 CvP PCIe 内核设置为工作在 Gen1 x4 模式。在这种情况下，通过 Gen1 x4 链路装入内核架构镜像。FPGA 中 CvP 之后的用户应用逻辑也使用了 Gen1 x4 链路。在 CvP FPGA 镜像更新期间以及更新之后，I/O 环配置（包括 SERDES）和嵌入式 PCIe CSR 内容保持不变。设计人员进行修改，替换最初的内核架构镜像，必须与最初设计镜像所使用的 I/O 和 PCIe 内核参数以及功能相一致。

CvP 拓扑不限于图 3 和图 4 所示的基本工作模式。图 5 介绍了 Stratix V FPGA 也支持的混合 FPGA 编程模式。通过 CvP 对第一个 FPGA 进行编程。通过 Altera 串行配置器件 (EPCS) 或者四路配置器件 (EPCQ) 对第一个 FPGA 的嵌入式 PCIe 端点内核和 I/O 环进行编程。与图 3 相似，通过 CvP 对 FPGA 内核架构进行编程。通过快速被动并行模式对后面的 FPGA 器件进行编程，如表 1 所示。第一个 FPGA 中用户设计的 IP 内核控制其他级联 FPGA 的编程（第二个 FPGA——第 N 个 FPGA）。

图 5. 混合模式 CvP 编程

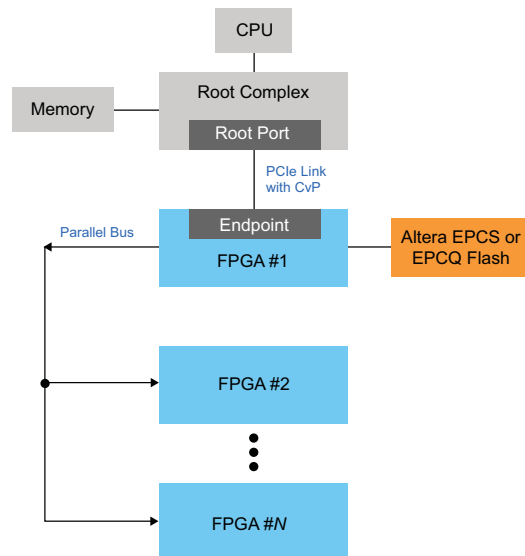
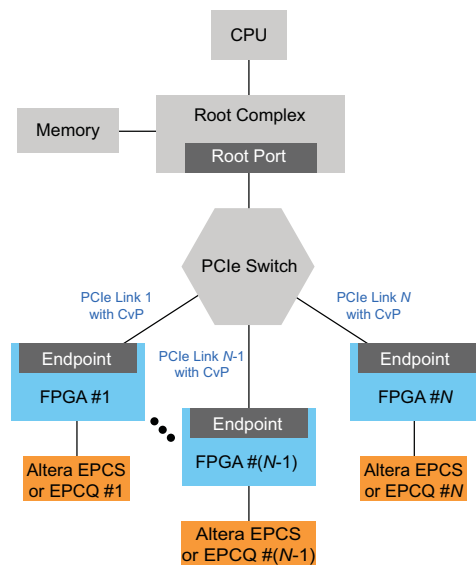


图 6 显示了另一种可能的 CvP 拓扑。通过 CvP 对多个 FPGA 进行编程。所有 FPGA 连接 PCIe 开关之后的根端口，因此，使用 PCIe 拓扑对连接这一开关的所有 FPGA 器件进行编程。

图 6. CvP 和 PCIe 开关



其他 PCIe 拓扑也支持 CvP，例如图 7 所示的菊花链 FPGA 拓扑。在这一模式中，由各自的 Altera EPCS 或者 EPCQ 器件对 FPGA 中的所有嵌入式 PCIe 内核进行初始化编程。通过 CvP 对菊花链器件的 FPGA 内核架构进行编程，所有端点和根端口并行连接。每一 FPGA（第 N 个 FPGA 除外）在其内核架构中都有设计人员开发的 IP 内核，用于控制下一 FPGA 的编程。

图 7. 菊花链格式 FPGA 编程和 CvP

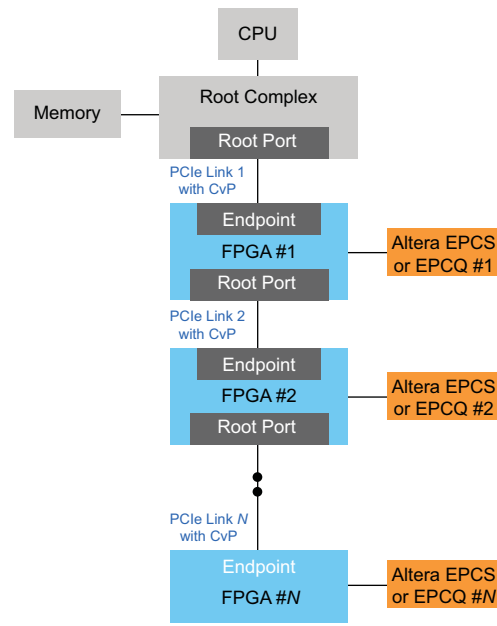


图 5、图 6 和图 7 所示的 FPGA 设计有不同的用户应用程序内容，因此有不同的配置文件镜像。在所有情况下，首先从主机 CPU 存储器或者其文件系统中恢复编程文件。

CvP 的优点

通过 PCIe 对 FPGA 内核架构进行编程利用了自治 PCIe 内核的功能。将这些特性结合起来，28-nm 设计人员可以充分发挥以下优势：

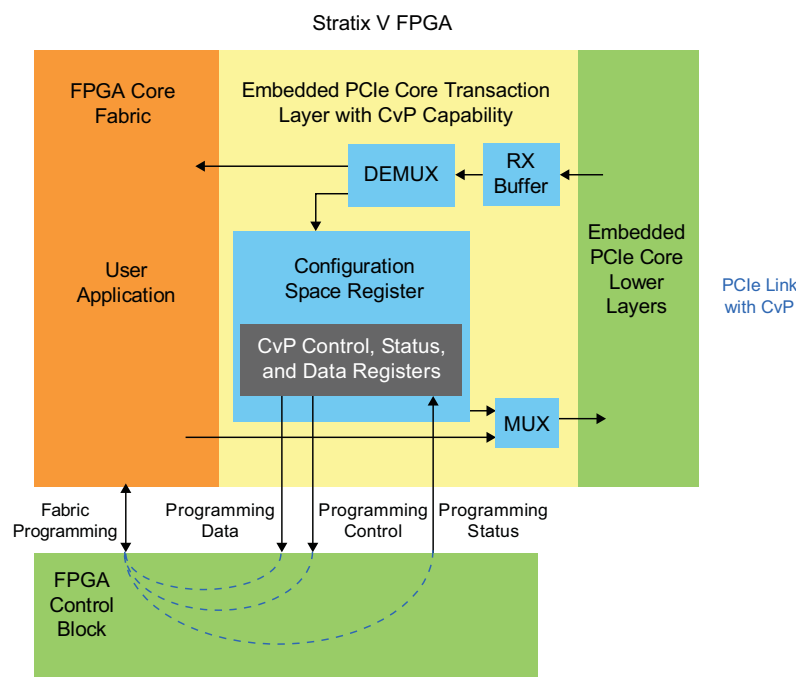
- 降低系统成本——CvP 避免了采用一个或者多个并行闪存器件，甚至是外部编程控制器件。而且，CvP 支持设计人员将 FPGA 编程文件存储在通过 PCIe 链路和 FPGA 链接的 CPU 存储器系统中。使用这一技术，只有 FPGA I/O 编程和 PCIe 内核参数存储在闪存器件中，只需要更小更便宜的闪存器件。
- 面积更小的电路板——一片 Altera EPCS 或者 EPCQ 闪存器件可以替代并行闪存器件。
- 减少了专用 FPGA 配置引脚数量——Stratix 系列器件一般需要数据通路较宽的闪存器件来存储 FPGA 编程文件。相反，EPCS 和 EPCQ 器件需要较少的专用引脚。
- 内核架构更新时，主机 CPU 不会停止工作——当 FPGA 工作在用户模式时，内核架构镜像更新后，主机 CPU 不用停机或者重新启动。CvP 只是 CPU 能够执行的一种软件应用程序。
- 用户应用程序镜像保护——只有主机 CPU 才能访问内核架构镜像，并且进行了加密或者压缩。
- 简单用户软件模型——这一模型使用了 PCIe 协议和用户应用程序 PCIe 拓扑来初始化一个或者多个 FPGA。

- 节省能源——可以根据用户应用程序参数，通过软件控制来装入低功耗临时镜像。这一特性适用于电池供电的便携式计算机系统。

CvP 工作

图 8 简要介绍了 Altera 28-nm FPGA 中支持 CvP 的主要构建模块以及相关接口。

图 8. 28-nm FPGA 中的 CvP 主要构建模块以及接口



CvP 按以下顺序工作：

1. 由 FPGA 控制模块对 I/O 引脚配置进行编程，包括收发器模块电信号和逻辑参数，以及嵌入式 PCIe 内核功能等。
2. FPGA 控制模块读取串行、四路或者并行闪存器件中的编程数据。
3. 通过 CvP 对 FPGA 内核架构进行编程。
4. 嵌入式 PCIe 端点对数据进行缓冲，将其发送至控制模块，对 FPGA 内核架构进行编程。
5. 主机 CPU 把 CvP 系统作为多个 PCIe CSR 和数据寄存器来看待。
6. 主机 CPU 将 FPGA 内核架构编程数据传送到嵌入式 PCIe 端点，PCIe 器件把这些数据传送到控制模块，由其对 FPGA 内核架构进行编程。
7. CvP 软件监视 CvP 状态寄存器，确定控制模块是否探测到任何错误，并采取相应的措施。
8. CvP 完成后，PCIe 内核切换到 FPGA 中应用逻辑分配给它的功能。FPGA 工作在用户模式时，主机 CPU 软件能够通过 CSR 写操作将 PCIe 内核切换回 CvP 模式。

对于 PCIe, CvP 支持是指供应商专用扩展功能 (VSEC)。Altera 增加的新寄存器是在 CSR 中。CvP 写入这些寄存器, 查询状态比特, 与 FPGA 控制模块进行通信。VSEC 一组新寄存器包括:

- VSEC 功能前缀。
- VSEC 长度、版本和 ID。
- 16 位 CvP 状态寄存器——主机 CPU 监视这一寄存器, 以便知道什么时候开始或者停止发送数据, 什么时候出现了无法纠正的编程错误。它还包括反应是否使能了加密 (AES) 和压缩 (DC) 功能的比特。
- 32 位 CvP 控制寄存器——这一寄存器提供模式和编程控制功能。主机 CPU 软件驱动器可以设置这些比特来初始化 CvP。
- 32 位 CvP 数据寄存器——这一寄存器保存来自嵌入式 PCIe 内核接收器缓冲的编程数据, 然后将其发送至控制模块。
- 32 位 JTAG 硅片 ID——这一只读寄存器返回 FPGA 硅片 ID, Altera 编程软件可以利用它来确定是否使用了正确的编程文件。
- 16 位用户器件 / 电路板类型 ID——提供用户可设置的数值, 以区分需要在 PCIe 拓扑中进行编程的不同 FPGA, 如图 5、图 6 和图 7 所示。

主机 CPU 使用 PCIe 技术的标准 32 位存储器映射 I/O (MMIO) 或者配置写操作对内核架构镜像进行编程。如前所述, 在 FPGA 镜像更新期间以及更新之后, I/O 环配置 (包括 SERDES) 和嵌入式 PCIe CSR 内容保持不变。在 CvP 期间, 由 CvP 监视所有 PCIe 基本地址寄存器 (BAR)。在正常工作模式中, 应用程序可以使用所有 BAR。



PCIe 内核冷复位会中断 PCIe 链路, 但不会启动 CvP 镜像装入, 也不会改变 FPGA 内核架构镜像。

软件支持

由 Altera 的 Quartus® II 开发软件、CvP 设计流程以及演示 CvP 工作的设计实例为 CvP 功能提供支持。

Quartus II 软件支持

Quartus II 软件在所有支持的平台以及操作系统上提供 CvP 支持。Quartus II 软件产生初始化嵌入式 PCIe 内核所需要的相应编程文件, 以及对外部闪存器件进行编程的文件, 用于对设计人员 PCIe 拓扑中 FPGA 的 CvP 系统进行初始化。不同的 CvP 工作模式有不同的文件内容, 如表 2 所示。Quartus II 软件还为 PCIe CvP 分层结构中的每一 FPGA 生成一个或者多个单独的 FPGA 内核架构编程文件。

例如, 如果设计人员使用 CvP 来配置四个位于 PCIe 开关下面的 FPGA, 与图 6 所示的拓扑结构相似, Quartus II 软件会产生两类配置文件: 原始二进制文件和 FPGA 内核镜像。

产生四个原始二进制文件, 对这一拓扑中的四个 EPCS 或者 EPCQ 器件进行编程。每一文件都含有配置相应 FPGA 的嵌入式 PCIe 硬核 IP 和 I/O 环所需要的编程信息。采用一种常用的 FPGA 编程方法对 EPCS 或者 EPCQ 器件进行编程。

根据每一 FPGA 不同用户应用程序版本数量，Quartus II 软件还为每一器件生成一个或者多个 FPGA 架构内核镜像。总之，至少有四个这类文件（每个 FPGA 一个）。理论上，每个器件的 FPGA 内核镜像数量没有上限。FPGA 内核镜像文件可以使用原始二进制、加密或者压缩格式。其中一个镜像（名为“初始”镜像）用于在上电时初始化 FPGA。可以通过 CvP 或者通过表 1 中介绍的其他 FPGA 内核架构编程方法装入最初的镜像。可以通过 CvP 使用所有其他 FPGA 内核镜像来更新 FPGA 内核架构。在这一 PCIe 拓扑中的每一 FPGA 都有自己的 16 位用户器件 / 电路板类型 ID 值，以便主机 CPU 通过 CvP 对相应的器件进行编程。

CvP 设计流程支持

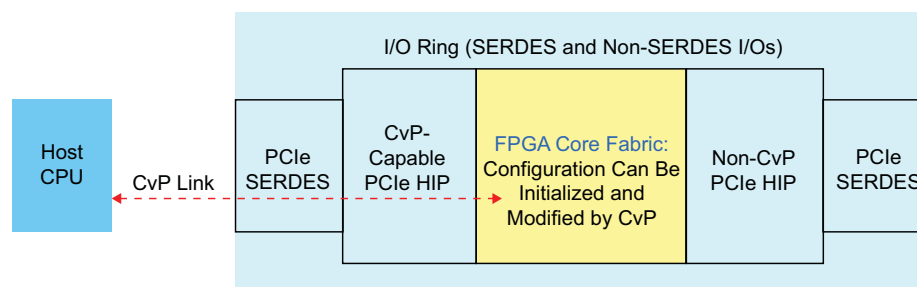
希望通过 CvP 来更新 FPGA 内容的设计人员必须确保 I/O 环和嵌入式 PCIe 内核参数以及功能保持不变。可以把通过 CvP 进行的 FPGA 镜像更新看成是简单部分重新配置，它有两个分区，一个分区（嵌入式 PCIe 硬核 IP 和 I/O 环）保持不变，而另一分区（FPGA 内核架构）能够多次更新。这样，可以保证嵌入式 PCIe 参数固定不变（包括 CvP 不使用的 PCIe 硬核 IP），并维持相同的 I/O 功能。

在面向一个 FPGA 的多个设计中引入了一些时序和时钟约束，以实现从最初设计到其他设计的移植。此外，在多 PCIe 内核应用中，在 CvP 内核架构镜像更新期间以及更新之后，只保证 CvP 功能 PCIe 内核正常工作。

结合部分重新配置功能和 CvP，在 CvP 部分内核架构镜像更新期间以及更新之后，设计人员可以保持所有其他 PCIe 链路正常工作。Altera FPGA 支持通过 CvP 进行部分重新配置，但这已经超出了本白皮书的讨论范围。

Altera 推荐了一种逻辑锁定 I/O 环以及嵌入式 PCIe 硬核 IP 内核的设计流程，支持设计人员通过 CvP 顺序装入一个或者多个 FPGA 内核架构设计镜像。图 9 介绍了各种构建模块之间的分区，器件上电后只需要进行一次配置即可（I/O 环包括 SERDES 和 PCIe 内核），根据主机 CPU 中运行的用户应用程序代码，通过 CvP 对 FPGA 内核架构进行多次更新。

图 9. FPGA 设计划分和 CvP



设计实例支持

CvP 设计人员的最终目标是初始化目标系统中的 FPGA 内核架构镜像，并周期性的进行更新。从嵌入式系统设计人员的角度看，CvP 是运行在 PCIe 器件驱动器之上的应用软件，可以在自己的 OS 环境中访问 CvP 功能端点。

Altera 提供清晰文本 C 应用程序设计实例，应用于 Windows 7 和 Linux 的 PCIe 开发板。C 程序通过 CvP 初始化 FPGA 内核架构。设计人员可以使用这一 C 程序作为自己程序开发的起点。

C 程序演示了实现 CvP 算法所需的步骤。C 程序设计实例适用于所有 CvP 工作模式（表 2，模式 2 和模式 3），可以用作 FPGA 内核架构镜像初始化的软件设计实例，也可以用于以后的内核架构更新。

结论

28-nm 系列器件的自治嵌入式 PCIe 内核保证了设计人员的 FPGA 能够满足 PCIe Base 的上电时间要求，以及 FPGA 内核架构容量和链路工作模式的 PCIe CEM 规范要求。这一特性实现了各种基于 PCIe 的计算机平台上较宽范围的互操作性。

Altera 的 28-nm FPGA 系列包括主要 CvP 增强功能，适用于大部分基于 PCIe 的定制应用。CvP 能够降低产品成本，减小电路板面积，同时简化了软件应用模型，具备可靠的现场系统更新功能。利用 CvP，设计人员可以初始化 FPGA 内核架构镜像，以后在运行时根据应用需要进行多次更新。

详细信息

- *PCIe 基本规范和 PCIe 卡电气机械规范*:
www.pcisig.com/specifications/pciexpress/base
- “PCI Express bridging options enable FPGA-based configurable computing,” *Programmable Logic Designline*, Mike Alford, Gennum Corp., September 8, 2008:
www.pldesignline.com/howto/210300269
- Stratix V FPGA: 为带宽而打造:
www.altera.com/products/devices/stratix-fpgas/stratix-v/stxv-index.jsp
- 资料: Stratix V 器件:
www.altera.com/literature/lit-stratix-v.jsp

致谢

- Arye Ziklik, 产品规划高级经理, Altera 公司。
- Mario Khalaf, CTO 办公室首席调查员, Altera 公司。