Altera’s new device configuration mode, Configuration via PCI Express® (CvPCIe), is a major feature enhancement in Stratix® V FPGAs that will benefit most PCIe-based customer applications. CvPCIe can reduce the product cost, lower the board size, simplify the software usage model, and provide a robust in-field system upgrade capability to customers. In addition, the Stratix V device’s autonomous embedded PCIe core ensures designers that their FPGA will meet the PCIe power-up time requirements, irrespective of the size and link-operating mode, guaranteeing a wide range of interoperability with various PCIe-based computer platforms.

Introduction

PCI Express® (PCIe) technology has replaced the venerable PCI as the de facto control plane interface between processors and the devices that they monitor. Since its introduction in 2005, FPGA designers have made PCIe one of the most widely used interfaces between FPGAs and processors, ASICs, and ASSP devices.

Today’s FPGAs include embedded PCIe cores that can serve as endpoints or root ports. However, prior to Altera® Stratix® V FPGAs, all PCIe cores embedded in FPGAs could not act as endpoints or root ports until the FPGA fabric was fully programmed. Full FPGA programming from an external source was required to create an endpoint that can respond to host CPU messages. In addition, due to shrinking device geometries, FPGA fabric sizes have rapidly grown to the extent that designers are concerned about device programming times that in some instances do not meet PCIe device (or endpoint) initialization time requirements.

Figure 1 provides a simplified representation of the PCIe power-up timing sequence as described in the PCIe Base and PCIe Card Electromechanical (CEM) Specifications. The minimum time allocated to device initialization and device training is 200 ms (equivalent to the difference between point 5 and point 1 in the diagram). The minimum amount of time allocated to device initialization is depicted by the time difference between point 3 and point 2 in the diagram, or about 95 ms.

Figure 1. PCIe Power-Up Timing Waveform
Since FPGA devices continue to pack more logic at smaller geometries, it takes a longer time to program large FPGA fabrics with their application-specific content. This time can exceed 95 ms in large devices. When an endpoint device does not reach L0 within the time allocated to it by the PCIe specifications, it may not respond to the software configuration access transactions (point 5) and the host CPU may fail to recognize this endpoint. In that case, the host CPU may ignore the endpoint and the system will operate without it.

In order to circumvent this failure discovery mechanism, Stratix V FPGA-embedded PCIe cores can be operational while the FPGA fabric is programmed from an external device. The Stratix V FPGA-embedded PCIe core is designed to always meet the PCIe power-up timing requirements by initializing the embedded PCIe cores and the device I/O ring in less than 95 ms.

During this initialization period, the embedded PCIe core is held in reset by PERST# and is released shortly before point 3 to start PCIe link discovery and training. The rest of the FPGA fabric begins programming after the PCIe link completes the training phase and reaches the L0 state. After the embedded PCIe endpoint core reaches the L0 state, the host operating system (OS) starts accessing the PCIe core’s configuration space registers (CSR) to perform configuration write access cycles that are part of the system initialization and discovery process (point 5).

In case the FPGA fabric is not fully programmed with the designer’s application content (in other words, it has not reached “user mode” yet), the autonomous PCIe core responds with configuration retry status (CRS) transactions until the FPGA fabric is fully loaded. The OS will correctly identify this endpoint and will attempt to poll it again until it becomes fully functional. The endpoint is allowed to respond with CRS for one second before the OS determines that the endpoint is faulty. In other words, the time allocated for the FPGA fabric programming cannot exceed one second in this device initialization mode.

FPGA Configuration via PCIe

Altera added a new device configuration mode, Configuration via PCI Express (CvPCIe), in Stratix V FPGAs. CvPCIe can leverage the autonomous PCIe core capability described in the previous section. After the PCIe core and their respective transceiver I/Os are programmed, the link trains to the designer-defined PCIe operating mode. When CvPCIe is enabled, designers can load the initial FPGA fabric image via PCIe, and then later modify the fabric image content at run time based on their application needs.

After the PCIe link finishes training and reaches the L0 state, the host CPU can program the FPGA fabric image via PCIe. (One of the embedded PCIe cores in each Stratix V FPGA is capable of performing CvPCIe when used as an endpoint.) During the FPGA fabric configuration via PCIe, all non-serializer/deserializer (SERDES) I/O pins can be held high by internal weak pull-up resistors. All other high-speed SERDES pins are essentially held in reset during CvPCIe fabric image loading events. These I/O assignments are designed to freeze I/O operations while the FPGA content is updated. When CvPCIe is enabled, the autonomous PCIe core will not respond with CRS transactions but rather accept and respond to PCIe configuration and data transactions during FPGA fabric configuration.

Table 1 below summarizes the Stratix V FPGA configuration modes.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Status</th>
<th>Data Widths (bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active serial (AS)</td>
<td>Existing</td>
<td>1</td>
</tr>
<tr>
<td>Active quad (AQ)</td>
<td>New</td>
<td>4</td>
</tr>
<tr>
<td>Passive Serial (PS)</td>
<td>Existing</td>
<td>1</td>
</tr>
<tr>
<td>8-bit fast passive parallel (FPP)</td>
<td>Existing</td>
<td>8</td>
</tr>
<tr>
<td>4-, 16-, or 32-bit FPP</td>
<td>New</td>
<td>4, 16, 32</td>
</tr>
<tr>
<td>JTAG based</td>
<td>Existing</td>
<td>Dedicated JTAG port</td>
</tr>
<tr>
<td>CvPCIe</td>
<td>New</td>
<td>1, 2, 4, 8 (1)</td>
</tr>
</tbody>
</table>

Notes:
(1) Number of lanes in the PCIe link (Gen1 x1, x2, x4, or x8; Gen2 x1, x2, x4, or x8; Gen3 x1, x2, x4, or x8)
Figure 2 provides a high-level pictorial representation of Stratix V configuration modes and flash programming methods. To simplify the device configuration modes, all flash modes are combined in the block diagram.

Figure 2. Stratix V Device Configuration and Flash Programming Modes

Notes:

1. Four possible FPGA configuration modes: (a) Active serial (x1, x4). (b) Passive serial/parallel (x1, x4, x8, x32) using an Altera MAX® CPLD or other logic to read from flash memory and configure the FPGA. (c) JTAG configuration for debug purposes (no need for external flash memory to configure the FPGA). (d) CvPCIe of the FPGA core fabric only. The PCIe hard IP (HIP) and I/O ring are first configured through another method.

2. Altera EPCS (serial) or EPCQ (quad) can be directly programmed via the download cable.

3. In cases where a MAX CPLD is used to program the flash memory, the MAX CPLD reads from the flash memory and configures the FPGA.

When using CvPCIe, the various PCIe core functional parameters as well as the functionality of the respective high-speed transceivers (SERDES) are initially programmed through one of the existing device initialization modes shown in Table 1. Cheaper production solutions are possible (but not mandated) through the use of an Altera serial- or quad-flash device that stores only the initialization bits associated with the I/O ring and the PCIe cores. Table 2 shows how two bits loaded via one of the device configuration modes (listed in Table 1) determine CvPCIe functionality. The bits are “CvPCIe_enabled” and “full chip initialization.”

Table 2. CvPCIe Operating Modes

<table>
<thead>
<tr>
<th>CvPCIe Mode Number</th>
<th>CvPCIe Mode Bits</th>
<th>FPGA Configuration Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CvPCIe Enabled</td>
<td>Full Chip Initialization</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>CvPCIe is off. Full chip initialization through a standard configuration mode. CvPCIe can not be used to update the FPGA fabric image.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>CvPCIe is on. Only PCIe cores, FPGA I/Os, and transceivers are initialized through a standard configuration mode. CvPCIe initially configures the FPGA fabric. CvPCIe may also be used to update the FPGA fabric image.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>CvPCIe is on. Full chip initialization through a standard configuration mode. CvPCIe can be used to update the FPGA fabric image.</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3 depicts a possible CvPCIe operating mode with a low-cost Altera flash device, and illustrates Mode 2 of Table 2.
Regardless of the CvPCie mode bit settings, the embedded PCIe core will always be autonomous. In other words, it will wake up and start link training before the FPGA fabric is configured. When CvPCie_enabled is on, CvPCie is enabled after the embedded PCIe core reaches the L0 state. The host CPU then can configure the FPGA fabric image whenever the software application instructs the host CPU to do so.

Figure 4 depicts Mode 3 in Table 2.

The PCIe endpoint link operating mode used for device programming is the same mode used by the FPGA target application after CvPCie is done. For example, designers can program their CvPCie PCIe core to operate at Gen3 x4 mode. In that case the fabric image is loaded through a Gen3 x4 link. The user application logic that operates in the FPGA after the completion of CvPCie also uses a Gen3 x4 link. The I/O ring configuration (including SERDES) and the embedded PCIe CSRs content remain unchanged during and after the CvPCie FPGA image updates. Designs modified to replace the initial fabric image must maintain the same I/O and PCIe core parameters and functionality used in the initial design image.
The CvPCIe topology is not limited to the basic operating modes depicted in Figure 3 and Figure 4; Figure 5 describes a mixed FPGA programming mode that is also feasible in Stratix V FPGAs. FPGA #1 is programmed via CvPCIe. The embedded PCIe endpoint core and the I/O ring of FPGA #1 are programmed via an Altera serial configuration devices (EPCS) or quad configuration devices (EPCQ). The FPGA fabric is programmed via CvPCIe, similar to Figure 3. Subsequent FPGA devices are programmed through the fast passive parallel mode described in Table 1. A user-designed intellectual property (IP) in FPGA #1 controls the programming of the other cascaded FPGAs (FPGA #2 – FPGA #N).

Figure 5. Mixed-Mode Programming with CvPCIe

Another possible CvPCIe topology is shown in Figure 6. Multiple FPGAs are programmed via CvPCIe. All FPGAs interface with the root port behind a PCIe switch, thereby using the PCIe topology to program all the FPGA devices that are attached to the switch.
In addition, CvPCIe support is feasible in other PCIe topologies, such as the daisy-chain FPGA topology shown in Figure 7. In this mode, all embedded PCIe cores that reside in the FPGAs are initially programmed by their respective Altera EPCS or EPCQ. The FPGA fabrics of the daisy-chained devices are programmed via CvPCIe, and all endpoints and root ports come up in parallel. Each FPGA (except FPGA #N) has a designer-developed IP in its fabric core that controls the programming of the next FPGA.
The FPGAs shown in Figure 5, Figure 6, and Figure 7 may have different user application contents and hence different configuration file images. In all cases, the programming files are initially retrieved from the host CPU memory or from its file system.

**CvPCIe Benefits**

FPGA fabric programming via PCIe leverages the autonomous PCIe core capability. Combined together, these features can deliver any or all of the following key benefits to Stratix V designers:

- **Lower system cost**—CvPCIe can eliminate one or more parallel flash devices and possibly an external programming controller device. In addition, it allows designers to have FPGA programming files stored in a CPU memory system attached to the FPGA via a PCIe link. By doing so, only FPGA I/O programming and PCIe core parameters are required to be stored in a flash device, hence a smaller and cheaper flash device can be used.
- **Smaller board space**—Parallel flash devices can be replaced by a single Altera EPCS or EPCQ flash device.
- **Reduction of dedicated FPGA configuration pins**—Stratix series devices typically require one or multiple wide data-path flash devices to store the FPGA programming file. In contrast, EPCS and EPCQ devices require fewer dedicated pins.
- **No host-CPU downtime during fabric updates**—No need for a host-CPU stall or reboot following fabric image updates when the FPGA operates in the user mode. CvPCIe is just another software application that the CPU can execute.
- **User application image protection**—Fabric image copies are accessible only to the host CPU and can be encrypted and/or compressed.
Simple user software model—This model uses the PCIe protocol and the user application PCIe topology to initialize single or multiple FPGAs.

Power saving—Low power, temporary images can be loaded via software control based on the user application profile. This feature can be useful in portable computers that are battery powered.

**CvPCIe Operation**

Figure 8 provides an overview of the main building blocks in Stratix V FPGAs that participate in CvPCIe and the respective interfaces.

Figure 8. CvPCIe Main Building Blocks and Interfaces in a Stratix V FPGA

I/O pin configurations, including transceiver-block electrical and logical parameters and the embedded PCIe core functionality, are programmed by the FPGA control block by reading programming data from the serial, quad, or parallel flash devices. Next, the FPGA fabric is programmed through CvPCIe. The embedded PCIe endpoint buffers the data and sends it to the control block to program the FPGA fabric.

The host CPU views the CvPCIe system as a collection of PCIe CSRs and data registers. The host CPU transmits FPGA fabric programming data to the embedded PCIe endpoint and the PCIe device passes these data onto the control block, which in turn programs the FPGA fabric. CvPCIe software monitors the CvPCIe status register to determine if the control block detects any errors, and reacts accordingly.

After CvPCIe completion, the PCIe core switches to the functionality assigned to it by the application logic that resides in the FPGA. As the FPGA operates in user mode, the host CPU software can switch the PCIe core back to the CvPCIe mode via CSR write transactions.
In PCIe terms, CvPCIe support is added as a Vendor-Specific Extended Capability (VSEC). There are new Altera-added registers that reside in the CSR. CvPCIe writes in these registers and poll status bits to communicate with the FPGA control block. The new set of VSEC registers includes:

- VSEC capability header
- VSEC length, revision, and ID
- 16-bit CvPCIe status register—The host CPU monitors this register to know when to start/stop sending data, or when there has been a programming error that should be treated as an uncorrectable error. There are also bits to reflect whether Encryption (AES) and/or Compression (DC) are enabled.
- 32-bit CvPCIe control register—This register provides mode and programming control. The host CPU software driver can set these bits to initiate CvPCIe events.
- 32-bit CvPCIe data register—This register holds the programming data coming from the embedded PCIe core receiver buffer before sending it on the control block.
- 32-bit JTAG Silicon ID—This read-only register returns the FPGA Silicon ID, which can be used by an Altera programming software to make sure it is using the correct programming file.
- 16-bit user device/board type ID—Provides a user-settable value to distinguish between the different FPGAs that need to be programmed in the PCIe topologies, such as those shown in Figure 5, Figure 6, and Figure 7.

The host CPU programs the fabric image using the PCIe technology’s standard 32-bit memory-mapped I/O (MMIO) or configuration write transactions. As mentioned in a previous section, the I/O ring configuration (including SERDES) and the embedded PCIe CSR content remain unchanged during and after the FPGA image updates.

During CvPCIe events, all PCIe base address registers (BARs) are intercepted by the CvPCIe. In normal operating mode, all BARs are available for application use.

PCIe core cold-reset events bring down the PCIe link, but they do not start CvPCIe image-loading events and they do not alter the FPGA fabric image.

Software Support
The CvPCIe functionality is complemented by Altera’s Quartus® II development software, CvPCIe design flow, and design examples that demonstrate CvPCIe operation.

Quartus II Software
Altera Quartus II development software tools provide CvPCIe support in Windows and Linux across all platforms and OS already supported by Quartus® II today. Quartus II software generates the respective programming files needed to initialize the embedded PCIe cores and/or program the external flash devices used for the CvPCIe system initialization of the FPGAs that reside in the designer’s PCIe topology. The content of the files will vary based on the CvPCIe operating modes (Table 2). The Quartus II software also generates one or more separate CvPCIe FPGA-fabric programming files for each of the FPGAs that participate in the PCIe CvPCIe hierarchy.

For example, assume that customers use CvPCIe to configure four FPGAs populated below a PCIe switch, similar to the topology depicted in Figure 6. The Quartus II software generates two types of configuration files: raw binary files and FPGA core images.

Four raw binary files are created to program the four EPCS or EPCQ devices of this topology. Each file contains the programming information necessary to configure the embedded PCIe cores and I/O rings of its respective FPGA. The EPCS or EPCQ devices are programmed via one of the regular Stratix V fabric programming methods.

The Quartus software also creates one or more FPGA fabric core images per device, depending on the number of different user application variants per FPGA. In total, there are at least four such files (one per FPGA). In theory, there is no upper limit to the number of FPGA core images per device that the Quartus II software can handle. The FPGA core image files can use raw binary, encrypted, or compressed format. One of the images (called initial image) is used to initialize the FPGA upon power-up. The initial image can be loaded via CvPCIe or through one of the
regular Stratix V fabric programming methods depicted in Table 1. All other FPGA core images can be used to update the FPGA fabric through CvPCIe. Each of the FPGAs in the PCIe topology has a unique 16-bit user device/board type ID value to help direct the host CPU to program the right device through CvPCIe.

**Design Flow Considerations**

Designers who want to update the FPGA content through CvPCIe must keep the I/O ring and the embedded PCIe core parameters and functionality unchanged. The FPGA image content update through CvPCIe can be viewed as a simple partial reconfiguration case with two partitioned regions where one region (the embedded PCIe cores and the I/O ring) remains unchanged, while the other region (FPGA fabric) can be updated multiple times. This entails keeping the embedded PCIe core parameters fixed (including the PCIe cores that are not used for CvPCIe) as well as maintaining the same I/O functionality.

Some timing and clocking constraints are introduced in the multiple designs that target a single FPGA to ensure migration from the initial design to the other designs. Additionally, in multi-PCIe core applications, only the CvPCIe-capable PCIe core is guaranteed to remain operational during and after the CvPCIe fabric image update events.

By combining partial reconfiguration with CvPCIe, designers soon will be able to keep all other PCIe cores operational during and after CvPCIe partial fabric image update events. Partial reconfiguration via CvPCIe is feasible in Stratix V FPGAs, but it is outside the scope of this white paper.

Altera provides a design flow that logically locks the I/O ring and the embedded PCIe cores and lets designers attach them to one or multiple FPGA fabric design images that may be later loaded via the CvPCIe. Figure 9 depicts the partition between the building blocks, which are configured just once following device power-up (I/O ring including SERDES and PCIe cores), and the FPGA fabric, which may be updated by CvPCIe as many times as required based on the user application code that runs on the host CPU.

**Figure 9. FPGA Design Partitioning with CvPCIe**

![FPGA Design Partitioning with CvPCIe](image)

**Design Example**

The ultimate goal of CvPCIe designers is to initialize and periodically update the FPGA fabric image in the target systems. From an embedded system designer’s viewpoint, CvPCIe is a software application that runs on top of a PCIe device driver that accesses CvPCIe-capable endpoints in their native OS environment.

Altera will provide a clear-text C application program design example that targets Stratix V PCIe development boards under Windows 7 and Linux. The C program will initialize the FPGA fabric through CvPCIe. Designers will be able to use this C program as a starting point for their own code development.

The C program will demonstrate the steps needed to implement the CvPCIe algorithm. The C program design example will be applicable in both CvPCIe operating modes (Table 2, Modes 2 and 3), and can be used as a software design example for FPGA fabric image initialization as well as subsequent fabric image updates.
Conclusion

The Stratix V device’s autonomous embedded PCIe core ensures designers that their FPGA will meet the power-up time requirements of the PCIe Base as well as the PCIe CEM specifications irrespective of the FPGA fabric size and link-operating mode. This feature guarantees wide-range interoperability with various PCIe-based computer platforms.

The Stratix V FPGA CvPCIe is a major feature enhancement that will benefit most PCIe-based customer applications. CvPCIe can reduce the product cost, lower the board size, simplify the software usage model, and provide a robust in-field system upgrade capability to customers. CvPCIe enables designers to initialize the FPGA fabric core image and later update it in run time as many times as needed by their applications.

Further Information

- PCIe Base Specification and PCIe Card Electromechanical Specification:  
  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 FPGAs: Built for Bandwidth:  
  www.altera.com/products/devices/stratix-fpgas/stratix-v/stxv-index.jsp
- Literature: Stratix V Devices:  
  www.altera.com/literature/lit-stratix-v.jsp

Acknowledgements

- Arye Ziklik, Senior Manager, Product Planning, Altera Corporation
- Mario Khalaf, Principal Investigator, Office of the CTO, Altera Corporation