Intel® Agilex™ Device Configuration via Protocol (CvP) Implementation User Guide

Updated for Intel® Quartus® Prime Design Suite: 19.4
# Contents

## 1. Overview
1.1. Benefits of Using CvP ................................................................. 4
1.2. CvP System .................................................................................. 4
1.3. CvP Modes ................................................................................... 5
  1.3.1. CvP Limitations .................................................................... 7

## 2. CvP Description
2.1. Configuration Images ................................................................. 8
2.2. CvP Modes ................................................................................... 8
  2.2.1. CvP Initialization Mode ............................................................ 8
  2.2.2. CvP Update Mode ................................................................. 8
2.3. Compression Features .................................................................. 9
  Data Compression ......................................................................... 9
  2.4. Pin Description ........................................................................ 9

## 3. CvP Topologies
3.1. Single Endpoint .......................................................................... 12
3.2. Multiple Endpoints ..................................................................... 12

## 4. Design Considerations
4.1. Designing CvP for an Open System ........................................... 14
  4.1.1. FPGA Power Supplies Ramp Time Requirement ...................... 14
  4.1.2. PCIe Wake-Up Time Requirement ......................................... 15
4.2. Designing CvP for a Closed System ........................................... 16

## 5. CvP Driver and Registers
5.1. CvP Driver Support ..................................................................... 17
5.2. CvP Driver Flow .......................................................................... 17
5.3. VSEC Registers for CvP ............................................................. 18
  5.3.1. Vendor Specific Capability Header Register ............................... 19
  5.3.2. Vendor Specific Header Register ............................................ 19
  5.3.3. Intel Marker Register ............................................................ 19
  5.3.4. User Configurable Device/Board ID Register ......................... 20
  5.3.5. CvP Status Register ............................................................... 20
  5.3.6. CvP Mode Control Register .................................................. 20
  5.3.7. CvP Data Registers ............................................................... 21
  5.3.8. CvP Programming Control Register ....................................... 21
  5.3.9. CvP Credit Register ............................................................. 22

## 6. Understanding the Design Steps for CvP Initialization using the P-tile in Intel Agilex Devices
6.1. Implementation of CvP Initialization Mode .................................. 23
  6.1.1. Generating the Synthesis HDL files for Intel FPGA P-Tile Avalon Streaming (Avalon-ST) for PCIe Express ............................................. 25
  6.1.2. Setting up the CvP Parameters in Device and Pin Options ........ 26
  6.1.3. Compiling the Design ......................................................... 27
  6.1.4. Converting the SOF File ...................................................... 27
  6.1.5. Bringing up the Hardware ................................................... 29
1. Overview

Configuration via Protocol (CvP) is a configuration scheme supported in Arria® V, Cyclone® V, Stratix® V, Intel® Arria 10, Intel Stratix 10, Intel Cyclone 10 GX, and Intel Agilex™ device families. The CvP configuration scheme creates separate images for the periphery and core logic. You can store the periphery image in a local configuration device and the core image in host memory, reducing system costs and increasing the security for the proprietary core image. CvP configures the Intel FPGA fabric through the PCI Express* (PCIe*) link, and is available for Endpoint variants only. This document describes the CvP configuration scheme for Intel Agilex device family.

Related Information

- Intel Stratix 10 Configuration via Protocol (CvP) Implementation User Guide
- Intel Arria 10 CvP Initialization and Partial Reconfiguration over PCIe Express User Guide
- Additional Clock Requirements for Transceivers, HPS, PCIe, and HBM2

1.1. Benefits of Using CvP

The CvP configuration scheme has the following advantages:

- Reduces system costs by reducing the size of the local flash device that stores the configuration data.
- Allows update of the FPGA without reprogramming the flash.
- Enables dynamic core updates without requiring a system power down. CvP allows you to update the FPGA core fabric through the PCIe link without a host restart or FPGA full chip reinitialization.
- Provides a simpler software model for configuration. A smart host can use the PCIe protocol and the application topology to initialize and update the FPGA core fabric.
- Allows quick update of your design for changing application loads.

1.2. CvP System

A CvP system typically consists of an FPGA, a PCIe host, and a configuration device.
1. The FPGA connects to the configuration device using the Active Serial x4 (fast mode) configuration scheme.

2. CvP applications use the PCIe Hard IP block on the left side of the device only.

3. You can use other PCIe Hard IP blocks for PCIe applications. You can select only one of the PCIe Hard IP blocks for CvP, and it must be on the left side of the device. Once you made this selection, you can't use the other PCIe Hard IP blocks for CvP.

Note: To avoid configuration failure, you must provide a free running and stable reference clock source to PCIe IP core before you start the configuration.

1.3. CvP Modes

The CvP configuration scheme supports the following modes:

- CvP Initialization mode
- CvP Update mode

CvP Initialization Mode

This mode configures the CvP PCIe core using the peripheral image of the FPGA through the on-board configuration device. Subsequently, configures the core fabric and all GPIOs through PCIe link.

Benefits of using CvP Initialization mode include:

- Satisfying the PCIe wake-up time requirement
- Saving cost by storing the core image in the host memory
**CvP Update Mode**

In the CvP update mode, you reconfigure the entire device except the CvP PCIe core after the device enters the user mode through full chip configuration or CvP initialization. The subsequent core image updates use the PCIe link (the periphery must not change during CvP update).

The CvP update mode uses the same process as root partition reuse in block-based design, which allows you to reuse the device periphery.

Choose this mode if you want to update the core image for any of the following reasons:

- To change core algorithms logic blocks
- To perform standard updates as part of a release process
- To customize core processing for different components that are part of a complex system

*Note:* The CvP update mode is available after the FPGA enters user mode. In user mode, the PCIe link is available for normal PCIe applications as well as to perform an FPGA core image update.

**Table 1. CvP Support for Intel Agilex Device Family**

<table>
<thead>
<tr>
<th>PCIe Version</th>
<th>Supported CvP Modes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gen 3 x16</td>
<td>Cvp Initialization</td>
</tr>
<tr>
<td>Gen 4 x 16(1)</td>
<td></td>
</tr>
<tr>
<td>Gen 4 x 8</td>
<td></td>
</tr>
<tr>
<td>Gen 3 x 8</td>
<td></td>
</tr>
</tbody>
</table>

(1) Only P-tile transceiver supports Gen 4 x 16
1.3.1. CvP Limitations

The Intel Agilex device CvP implementation has the following limitations and restrictions in the current version of the Intel Quartus® Prime software:

- Only MemWR transactions can be used to write fabric configuration data to the CvP data register. ConfigWR transactions are not supported.
- When you poll the CVP_CREDIT bits from the CvP credit register, you must write the next 4KB of fabric configuration data to the CvP data register within 50 ms of receiving an additional credit. Failure to send the data results in configuration failure.
- The CvP response time is variable and depends on different conditions. The typical delay time is 5 sec and it is safe to wait till 1 min. So the driver should poll status in credit register to decide on driver timeout.
- In CvP initialization and update mode, when FPGA fabric is not programmed, the PCIe features that uses FPGA fabric are not accessible.
- To generate the update image in the CvP update mode, you must use the same version of the Intel Quartus Prime software that you use to generate the base image.

1.3.1.1. CvP Error Recovery

This section describes expected behavior during different error situations.

Table 2. Intel Agilex Device CvP Error Events and Suggested Recovery Methods

<table>
<thead>
<tr>
<th>Error Events</th>
<th>Suggested Recovery Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCIe bus error during CvP</td>
<td>System is unrecoverable and you must power-cycle the system.</td>
</tr>
<tr>
<td>PCIe bus error results in PERST assert.</td>
<td>System is unrecoverable and you must power-cycle the system.</td>
</tr>
<tr>
<td>CvP operation requests to stop</td>
<td>Unsupported. Aborting configuration after requesting CvP operation is not supported. Intel recommends to power-cycle the system.</td>
</tr>
<tr>
<td>A bitstream is provided from a Intel Quartus Prime version other than the one used to generate configuration firmware currently running in the device.</td>
<td>The CVP_CONFIG_ERROR bit in the CvP status register goes high. Go through the Teardown sequence prior to sending another bitstream. Refer to CvP Drive Flow section for more information on Teardown sequence. Note: Mixing bitstreams from different Quartus versions is not supported.</td>
</tr>
</tbody>
</table>
2. CvP Description

2.1. Configuration Images

In CvP, you split your bitstream into two images: periphery image and core image.

You use the Intel Quartus Prime Pro Edition software to generate the images:

- Periphery image (*.periph.jic) — contains all of the periphery. The entire periphery image is static and cannot be reconfigured.
- Core image (*.core.rbf) — contains all of the core components of the design.

Related Information
Intel Agilex Configuration User Guide

2.2. CvP Modes

2.2.1. CvP Initialization Mode

In this mode, an external configuration device stores the periphery image and it loads into the FPGA through the Active Serial x4 (Fast mode) configuration scheme. The host memory stores the core image and it loads into the FPGA through the PCIe link.

After the periphery image configuration is complete, the CONF_DONE signal goes high and the FPGA starts PCIe link training. When PCIe link training is complete, the PCIe link transitions to L0 state and then allows the host to complete PCIe enumeration of the link. The PCIe host then initiates the core image configuration through the PCIe link. The PCIe REFCLK needs to be running prior to sending the periphery image.

After the core image configuration is complete, the CVP_CONF_DONE pin (if enabled) goes high, indicating the FPGA is fully configured.

After the FPGA is fully configured, the FPGA enters user mode. If the INIT_DONE signal is enabled, the INIT_DONE signal goes high after initialization is complete and the FPGA enters the user mode.

In user mode, the PCIe links are available for normal PCIe applications.

2.2.2. CvP Update Mode

CvP update mode is a reconfiguration scheme that allows a host device to deliver an updated bitstream to a target FPGA device after the device enters user mode. In this mode, the FPGA device initializes by loading the full configuration image from the external local configuration device to the FPGA or after CvP initialization.
You can perform CvP update on a device that you originally configure using CvP initialization or any other configuration scheme. CvP initialization is not a prerequisite for performing CvP update.

In user mode, the PCIe links are available for normal PCIe applications. You can use the CvP PCIe link to perform an FPGA core image update. To perform the FPGA core image update, you can create one or more FPGA core images in the Intel Quartus Prime Pro Edition software that have identical connections to the periphery image.

**Figure 2. Periphery and Core Image Storage Arrangement for CvP Core Image Update**

The periphery image remains the same for different core image updates. If you change the periphery image, you must reprogram the local configuration device with the new periphery image.

### 2.3. Compression Features

**Data Compression**

The Intel Quartus Prime Pro Edition software compresses all Intel Agilex device bitstreams to reduce the storage requirement and increase bitstream processing speed. The periphery and core images are both compressed.

### 2.4. Pin Description

The following table lists the CvP pin descriptions and connection guidelines:
<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Pin Type</th>
<th>Pin Description</th>
<th>Pin Connection</th>
<th>Configuration Schemes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVP_CONF_DONE</td>
<td>Output</td>
<td>The CVP_CONF_DONE pin indicates the device has received the complete bitstream during configuration via protocol (CvP) core image configuration. When used for this purpose, enable this pin using the Intel Quartus Prime software. Connect this output pin to an external logic device that monitors the CvP operation. The VCCIO_SDM power supply must meet the input voltage specification of the receiving side.</td>
<td>SDM_IO0 SDM_IO10 SDM_IO11 SDM_IO12 SDM_IO13 SDM_IO14 SDM_IO15 SDM_IO16</td>
<td></td>
</tr>
<tr>
<td>INIT_DONE</td>
<td>Output</td>
<td>The INIT_DONE pin indicates the device has entered user mode upon completion of configuration. To use INIT_DONE to indicate user mode entry, you must enable it in the Intel Quartus Prime software. When the INIT_DONE function is enabled, this pin drives high when configuration is completed and the device goes into user mode. Intel recommends you to use SDM_IO0 or SDM_IO16 to implement the INIT_DONE function when available as it has an internal weak pull-down for the correct function of INIT_DONE during power up. If SDM_IO0 and SDM_IO16 are unavailable, the INIT_DONE function can also be implemented using any unused SDM_IO pins provided that an external 4.7–kΩ pull-down resistor is provided for the INIT_DONE signal.</td>
<td>SDM_IO0 SDM_IO10 SDM_IO11 SDM_IO12 SDM_IO13 SDM_IO14 SDM_IO15 SDM_IO16</td>
<td></td>
</tr>
<tr>
<td>CONF_DONE</td>
<td>Output</td>
<td>The CONF_DONE pin indicates all configuration data has been received. By default, Intel recommends using the SDM_IO16 pin to implement the CONF_DONE function. If SDM_IO16 is unavailable, the CONF_DONE function can also be implemented using any unused SDM_IO pins. Except for SDM_IO0 and SDM_IO16, other SDM_IO pins are required to connect to an external 4.7-kΩ pull-down resistor for the CONF_DONE signal. Connect the CONF_DONE pin to the external configuration controller when configuring using the Avalon-ST (AVST) interface. You have an option to monitor this signal with an external component if you are using the active serial (AS) x4 configuration scheme.</td>
<td>SDM_IO0 SDM_IO10 SDM_IO11 SDM_IO12 SDM_IO13 SDM_IO14 SDM_IO15 SDM_IO16</td>
<td></td>
</tr>
<tr>
<td>I_PIN_PERST_N_U[10,20]_P</td>
<td>Input</td>
<td>PCIe Platform reset pin In a PCIe adapter card implementation, connect the PCIe nPERST signal from the PCIe edge connector to each P-tile transceiver bank I_PIN_PERST_N input. Use a level translator to fan out and change the 3.3V open-drain nPERST signal from the PCIe connector to the 1.8V I_PIN_PERST_N input of each P-tile transceiver that is used on the board.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Pin Name</td>
<td>Pin Type</td>
<td>Pin Description</td>
<td>Pin Connection</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
<td>-----------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Provide a 1.8V pull-up resistor to the \texttt{I_PIN_PERST_N} input as the \texttt{nPERST} signal from the PCIe connector is an open-draw signal. You must pull up the 3.3V PCIe \texttt{nPERST} signal on the adapter card. For non-PCIe systems, connect the system's master reset signal to the \texttt{I_PIN_PERST_N} input pin. If the master reset is not 1.8V, use a level shifter to meet the 1.8V \texttt{I_PIN_PERST_N} input requirement. For open-drain master reset driving the \texttt{I_PIN_PRST_N} input, provide a 1.8V pull-up resistor. Ensure all power to the device as well as the PCIe clock is stable prior to releasing the reset to the \texttt{I_PIN_PRST_N} pin. This input pin does not have an internal pull-up resistor, you need to add an external 5kΩ – 10kΩ pull-up resistor if the voltage translator does not provide an active driver. If the tile is unused, tie to GND.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Related Information**

- Intel Agilex Device Family Pin Connection Guidelines
- Intel FPGA P-Tile Avalon Streaming (Avalon-ST) for PCI Express User Guide
3. CvP Topologies

3.1. Single Endpoint

Use the single endpoint topology to configure a single FPGA. In this topology, the PCIe link connects one PCIe endpoint in the FPGA device to one PCIe root port in the host.

![Single Endpoint Topology](image)

3.2. Multiple Endpoints

Use the multiple endpoints topology to configure multiple FPGAs through a PCIe switch. This topology provides you with the flexibility to select the device to be configured or update through the PCIe link. You can connect any number of FPGAs to the host in this topology.

The PCIe switch controls the core image configuration through the PCIe link to the targeted PCIe endpoint in the FPGA. You must ensure that the root port can respond to the PCIe switch and direct the configuration transaction to the designated endpoint based on the bus/device/function address of the endpoint specified by the PCIe switch.
Figure 4. Multiple Endpoints Topology

Intel® Agilex™ Device

Configuration Control Signals & CvP Pins

Optional Monitoring

External Clock Source

PCle Host

Root Complex

Core Image

Update via PCIe Link

PCIe Link

PCIe Switch

MSEL

Configuration Control Signals

End Point

PCle Hard IP (HIP)

Secure Device Manager

FPGA Fabric

Core Image

Configuration Control Signals & CvP Pins

QSPI Flash Memory

DATA[3:0]

DCLK

nCS0

Peripheral Image (.jic)

Configuration Control Signals & CvP Pins

Optional Monitoring

External Clock Source

Intel® Agilex™ Device

Configuration Control Signals & CvP Pins

MSEL

Configuration Control Signals

End Point

PCle Hard IP (HIP)

Secure Device Manager

FPGA Fabric

Core Image

Configuration Control Signals & CvP Pins

QSPI Flash Memory

DATA[3:0]

DCLK

nCS0

Peripheral Image (.jic)
4. Design Considerations

4.1. Designing CvP for an Open System

Follow these guidelines when designing an open CvP system where you do not have complete control of both ends of the PCIe link.

4.1.1. FPGA Power Supplies Ramp Time Requirement

For an open system, you must ensure that your design adheres to the FPGA power supplies ramp-up time requirement.

The power-on reset (POR) circuitry keeps the FPGA in the reset state until the power supply outputs are in the recommended operating range. A POR event occurs from when you power up the FPGA until the power supplies reach the recommended operating range within the maximum power supply ramp time, $t_{\text{RAMP}}$. If $t_{\text{RAMP}}$ is not met, the device I/O pins and programming registers remain tri-stated, during which device configuration can fail.

To meet the PCIe link up time for CvP, the total $t_{\text{RAMP}}$ must be less than 10 ms, from the first power supply ramp-up to the last power supply ramp-up. You must select ASx4 fast mode for MSEL settings to make sure the shortest POR delay.

Figure 5. FPGA Power Supplies Ramp-Up Time and POR
4.1.2. PCIe Wake-Up Time Requirement

For an open system, you must ensure that the PCIe link meets the PCIe wake-up time requirement as defined in the PCI Express CARD Electromechanical Specification. The transition from power-on to the link active (L0) state for the PCIe wake-up timing specification must be within 200 ms. The timing from FPGA power-up until the Hard IP for PCI Express IP Core in the FPGA is ready for link training must be within 120 ms.

Related Information
PCI Express Card Electromechanical 3.0 Specification

4.1.2.1. For CvP Initialization Mode

To meet the 120 ms wake-up time requirement for the PCIe Hard IP in CvP initialization mode, you need to use periphery image because the configuration time for periphery image is significantly less than the full FPGA configuration time. You must use the Active Serial x4 (fast mode) configuration scheme for the periphery image configuration.

To ensure successful configuration, all POR-monitored power supplies must ramp up monotonically to the operating range within the 10 ms ramp-up time. The PERST# signal indicates when the FPGA power supplies are within their specified voltage tolerances and the REFCLK is stable. The embedded hard reset controller triggers after the internal status signal indicates that the periphery image has been loaded. This reset does not trigger off of PERST#. For CvP Initialization mode, the PCIe link supports the FPGA core image configuration and subsequent PCIe applications in user mode.

Note: For Gen 3/Gen 4 capable Endpoints, after loading the core bitstream (core.rbf), Intel recommends to verify that the link has been trained to the expected Gen 3/Gen 4 rate. If the link is not operating at Gen 3/Gen 4, software can trigger the Endpoint to retrain.

Figure 6. PCIe Timing Sequence in CvP Initialization Mode

(2) REFCLK must be stable 80 ms after the power supplies are stable in order to achieve the 145 ms link training complete time.
Table 4. Power-Up Sequence Timing in CvP Initialization Mode

<table>
<thead>
<tr>
<th>Timing Sequence</th>
<th>Timing Range (ms)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>2-6.5</td>
<td>FPGA POR delay time (AS Fast Mode)</td>
</tr>
<tr>
<td>b</td>
<td>80</td>
<td>Maximum time from the FPGA power up to the end of periphery configuration in CvP initialization mode (before transceiver calibration)</td>
</tr>
<tr>
<td>c</td>
<td>20</td>
<td>Minimum calibration time before PERST# is deasserted</td>
</tr>
<tr>
<td>d</td>
<td>60</td>
<td>Minimum transceiver calibration window</td>
</tr>
<tr>
<td>e</td>
<td>80</td>
<td>Typical transceiver calibration window</td>
</tr>
<tr>
<td>f</td>
<td>100</td>
<td>Minimum PERST# signal active from the host</td>
</tr>
<tr>
<td>g</td>
<td>120</td>
<td>Maximum time from the FPGA power up to the end of periphery configuration in CvP initialization mode (include transceiver calibration)</td>
</tr>
<tr>
<td>h</td>
<td>20</td>
<td>Maximum PERST# signal inactive time from the host before the PCIe link enters training state</td>
</tr>
<tr>
<td>i</td>
<td>100</td>
<td>Maximum time PCIe device must enter L0 after PERST# is deasserted</td>
</tr>
<tr>
<td>j</td>
<td>10</td>
<td>Maximum ramp-up time requirement for all POR-monitored power supplies in the FPGA to reach their respective operating range</td>
</tr>
</tbody>
</table>

Note: 100 ms timing range is only applicable to PCIe Gen1/Gen2. PCIe Gen 3 does not need to meet 100 ms timing requirement.

4.1.2.2. For CvP Update Mode

Before you perform CvP update mode, the device must be in user mode.

Note: For Gen 3/Gen 4 capable Endpoints, in user mode, Intel recommends to verify that the link has been trains to the expected Gen 3/Gen 4 rate. If the link is not operating at Gen 3/Gen 4, software can trigger the Endpoint to retrain.

4.2. Designing CvP for a Closed System

While designing CvP for a closed system where you control both ends of the PCIe link, estimate the periphery configuration time for CvP Initialization mode or full FPGA configuration time for CvP update mode. You must ensure that the estimated configuration time is within the time allowed by the PCIe host. Your driver can poll the USERMODE bit of the CvP Status Register to determine if the FPGA enters the user mode.
5. CvP Driver and Registers

5.1. CvP Driver Support

You can develop your own custom CvP driver for Linux using the sample Linux driver source code provided by Intel.

Note: The Linux driver provided by Intel is not a production driver. You must adapt this driver to your design’s strategy.

Related Information
Download the OpenSource Linux CvP Driver

5.2. CvP Driver Flow

The CvP driver flow assumes that the FPGA is powered up and the SDM control block has already configured the FPGA with the periphery image, which is indicated by the CVP_EN bit in the CvP status register.
5.3. VSEC Registers for CvP

The Vendor Specific Extended Capability (VSEC) registers occupy byte offsets 0xD00 to 0xD4C in the PCIe Configuration Space. The PCIe host uses these registers to communicate with the FPGA control block. The following table shows the VSEC register map. Subsequent tables provide the fields and descriptions of each register.

Table 5. VSEC Registers for CvP

<table>
<thead>
<tr>
<th>Byte Offset</th>
<th>Register Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xD00</td>
<td>Vendor Specific Capability Header</td>
</tr>
<tr>
<td>0xD04</td>
<td>Vendor Specific Header</td>
</tr>
<tr>
<td>0xD08</td>
<td>Intel Marker</td>
</tr>
</tbody>
</table>

continued...
### 5.3.1. Vendor Specific Capability Header Register

**Table 6. Vendor Specific Capability Header Register (Byte Offset: 0xD00)**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[15:0]</td>
<td>PCI Express Extended Capability ID</td>
<td>0x000B</td>
<td>RO</td>
<td>PCIe specification defined value for VSEC Capability ID.</td>
</tr>
<tr>
<td>[19:16]</td>
<td>Version</td>
<td>0x1</td>
<td>RO</td>
<td>PCIe specification defined value for VSEC version.</td>
</tr>
<tr>
<td>[31:20]</td>
<td>Next Capability Offset</td>
<td>Variable</td>
<td>RO</td>
<td>Starting address of the next Capability Structure implemented, if any.</td>
</tr>
</tbody>
</table>

### 5.3.2. Vendor Specific Header Register

**Table 7. Vendor Specific Header Register (Byte Offset: 0xD04)**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[15:0]</td>
<td>VSEC ID</td>
<td>0x1172</td>
<td>RO</td>
<td>A user configurable VSEC ID.</td>
</tr>
<tr>
<td>[19:16]</td>
<td>VSEC Revision</td>
<td>0</td>
<td>RO</td>
<td>A user configurable VSEC revision.</td>
</tr>
<tr>
<td>[31:20]</td>
<td>VSEC Length</td>
<td>0x05C</td>
<td>RO</td>
<td>Total length of this structure in bytes.</td>
</tr>
</tbody>
</table>

### 5.3.3. Intel Marker Register

**Table 8. Intel Marker Register (Byte Offset: 0xD08)**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[31:0]</td>
<td>Intel Marker</td>
<td>0x41721172</td>
<td>RO</td>
<td>An additional marker.</td>
</tr>
</tbody>
</table>

---

(3) This register is no longer functional in Intel Agilex devices.
5.3.4. User Configurable Device/Board ID Register

Table 9. User Configurable Device/Board ID Register (Byte Offset: 0xD1C)

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[15:0]</td>
<td>User Configurable Device/Board ID</td>
<td>0x00</td>
<td>RO</td>
<td>Helps user to select the correct programming file.</td>
</tr>
</tbody>
</table>

5.3.5. CvP Status Register

Table 10. CvP Status Register (Byte Offset: 0xD1E)

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[10]</td>
<td>CVP_CONFIG_SUCCESS</td>
<td>Variable</td>
<td>RO</td>
<td>Status bit set by the device to indicate that the core image configuration was successful.</td>
</tr>
<tr>
<td>[8]</td>
<td>PLD_CLK_IN_USE</td>
<td>Variable</td>
<td>RO</td>
<td>From clock switch module to fabric. You can use this bit for debug.</td>
</tr>
<tr>
<td>[7]</td>
<td>CVP_CONFIGDONE</td>
<td>Variable</td>
<td>RO</td>
<td>Indicates that the device has completed the device configuration via CvP and there were no errors.</td>
</tr>
<tr>
<td>[5]</td>
<td>USERMODE</td>
<td>Variable</td>
<td>RO</td>
<td>Indicates if the configurable FPGA fabric is in user mode.</td>
</tr>
<tr>
<td>[4]</td>
<td>CVP_EN</td>
<td>Variable</td>
<td>RO</td>
<td>Indicates if the device has enabled CvP mode.</td>
</tr>
<tr>
<td>[3]</td>
<td>CVP_CONFIG_ERROR</td>
<td>Variable</td>
<td>RO</td>
<td>Reflects the value of this signal from the device, checked by software to determine if there was an error during configuration.</td>
</tr>
<tr>
<td>[2]</td>
<td>CVP_CONFIGREADY</td>
<td>0x00</td>
<td>RO</td>
<td>Reflects the value of this signal from the device, checked by software during programming algorithm to determine the device is ready for configuration.</td>
</tr>
<tr>
<td>[1:0]</td>
<td>—</td>
<td>Variable</td>
<td>RO</td>
<td>Reserved.</td>
</tr>
</tbody>
</table>

5.3.6. CvP Mode Control Register

Table 11. CvP Mode Control Register (Byte Offset: 0xD20)

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[31:3]</td>
<td>—</td>
<td>0x0000</td>
<td>RO</td>
<td>Reserved.</td>
</tr>
<tr>
<td>[2]</td>
<td>—</td>
<td>0x0000</td>
<td>RW</td>
<td>Reserved&lt;sup&gt;(4)&lt;/sup&gt;.</td>
</tr>
<tr>
<td>[1]</td>
<td>PLD_DISABLE</td>
<td>1'b0</td>
<td>RW/RO</td>
<td>Enables/disables the PLD interface. This allows Host driver to switch the PLD interface out before USER MODE deasserts, continued...</td>
</tr>
</tbody>
</table>

<sup>(4)</sup> Intel recommends to set the reserved bit to 0 for write operation. For read operations, the PCIe IP always generates 0 as the output.
and to switch the PLD interface back in only after USER MODE has been asserted. This helps to prevent any glitches or race conditions during the USER MODE switching.

- 1: Disable the application layer interface.
- 0: Enable the application layer interface.

Only change the value of this signal when there has been no other TLP's to or from the HIP for 10 us. There should be no TLP's issued to the HIP for 10 us after this value changes. When entering CVP, this bit should be set before CVP_MODE is set. When exiting CVP, it should be cleared after CVP_MODE is clears. This ensures that there is no PLD switching during CVP. This field is RW when cvp_en=1, and RO when cvp_en=0.

5.3.7. CvP Data Registers

Table 12. CvP Data Register (Byte Offsets: 0xD24 - 0xD28)

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[31:0]</td>
<td>CVP_DATA</td>
<td>0x00000000</td>
<td>RW</td>
<td>Write the configuration data to this register. The data is transferred to the SDM to configure the device. Software must ensure that all bytes in the memory write dword are enabled. You can access this register using configuration writes. Alternatively, when in CvP mode, this register can also be written by a memory write to any address defined by a memory space BAR for this device. Using memory writes are higher throughput than configuration writes.</td>
</tr>
</tbody>
</table>

5.3.8. CvP Programming Control Register

Table 13. CvP Programming Control Register (Byte Offset: 0xD2C)

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[31:2]</td>
<td>—</td>
<td>0x0000</td>
<td>RO</td>
<td>Reserved.</td>
</tr>
<tr>
<td>[1]</td>
<td>START_XFER</td>
<td>1'b0</td>
<td>RW</td>
<td>Sets the CvP output to the FPGA control block indicating the start of a transfer.</td>
</tr>
<tr>
<td>[0]</td>
<td>CVP_CONFIG</td>
<td>1'b0</td>
<td>RW</td>
<td>When set to 1, the FPGA control block begins a transfer via CvP.</td>
</tr>
</tbody>
</table>
5.3.9. CvP Credit Register

The credit registers slow down the transmission of the CvP data to handle back pressure when there is no buffer space available within the configuration system. The crediting mechanism handles the back pressure from the configuration system. The total credits register increments each time an additional 4k buffer is available.

Table 14.  CvP Credits Register (Byte Offset: 0xD48)

<table>
<thead>
<tr>
<th>Bits</th>
<th>Reset Value</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[31:16]</td>
<td>0x00</td>
<td>RO</td>
<td>Reserved.</td>
</tr>
<tr>
<td>[15:8]</td>
<td>0x00</td>
<td>RO</td>
<td>Least significant 8 bits of the total number of 4k credits granted.</td>
</tr>
<tr>
<td>[7:0]</td>
<td>0x00</td>
<td>RO</td>
<td>Reserved.</td>
</tr>
</tbody>
</table>
6. Understanding the Design Steps for CvP Initialization using the P-tile in Intel Agilex Devices

6.1. Implementation of CvP Initialization Mode

CvP Initialization mode splits the bitstream into periphery and core images. The periphery image is stored in a local flash device on the PCB. The core image is stored in host memory. You must download the core image to the FPGA using the PCI Express link.

You must specify CvP Initialization mode in the Intel Quartus Prime Pro Edition software by selecting the CvP Settings Initialization and Update and you must also instantiate the Intel P-Tile Avalon-ST for PCI Express.
Figure 8. Example Implementation Flow for CvP Initialization

The CvP Initialization demonstration walkthrough includes the following steps:

- **Generating the Synthesis HDL files for Intel FPGA P-Tile Avalon Streaming (Avalon-ST) for PCIe Express** on page 25
- **Setting up the CvP Parameters in Device and Pin Options** on page 26
- **Compiling the Design** on page 27
- **Converting the SOF File** on page 27
- **Bringing up the Hardware** on page 29
Related Information
Intel FPGA P-Tile Avalon Streaming (Avalon-ST) for PCI Express User Guide

6.1.1. Generating the Synthesis HDL files for Intel FPGA P-Tile Avalon Streaming (Avalon-ST) for PCIe Express

Follow these steps to generate the synthesis HDL files with CvP enabled:

1. Open the Intel Quartus Prime Pro Edition software.
2. On the Tools menu, click Platform Designer. The Open System window appears.
3. For System, click + and specify a File Name to create a new platform designer system. Click Create.
4. On the System Contents tab, delete the clock_in and reset_in components that appear by default.
5. In the IP Catalog locate and double-click Intel P-tile Avalon-ST for PCI Express. The new window appears.
6. On the IP Settings tab, specify the parameters and options for your design variation.
   
   Note: In case of Gen 3 and Gen 4 x8 variants, only Port 0 implements CvP.

7. On the Example Designs tab, select the Simulation option to generate the testbench, and select the Synthesis option to generate the hardware design example.

8. For Generated file format, only Verilog is available.

9. Click the Generate Example Design button. The Select Example Design Directory dialog box appears. Click OK. The software generates Intel Quartus Prime project files for PCI Express reference design. Click Close when generation completes. An example design intelpcie_ptile_ast_0_example_design is created in your project directory.

10. Click Finish. Close your current project and open the generated PCI Express example design (pcie_ed.qpf).

11. Complete your CvP design by adding any desired top-level design and any other required modules. Pin assignments already being assigned properly based on the target development kit that user specified earlier.

   Note: Reference design for CvP initialization and update is not available in the current version of the Intel Quartus Prime software.

Related Information
• Intel FPGA P-Tile Avalon Streaming (Avalon-ST) for PCI Express User Guide
• Download the OpenSource Linux CvP Driver
6.1.2. Setting up the CvP Parameters in Device and Pin Options

Follow these steps to specify CvP parameters:

1. On the Intel Quartus Prime Assignment menu, select **Device**, and then click **Device and Pin Options**.

2. Under **Category**, select **Configuration** and then enable the following options:
   a. For **Configuration scheme**, select **Active Serial x4 (can use Configuration Device)**.
   b. For **Configuration pin**, click **Configuration Pin Options** and then turn on **USE CONF_DONE output** and **USE CVP_CONFDONE output**. Click **OK**.

3. Under **Category**, select **CvP Settings** to specify CvP settings. For **Configuration via Protocol**, select **Initialization and update** option. Click **OK**.
4. Click OK.

6.1.3. Compiling the Design

To compile the design, on the Processing menu, select Start Compilation to create the .sof file.

6.1.4. Converting the SOF File

Follow these steps to convert your .sof file into separate images for the periphery and core logic.

1. After the .sof file is generated, under File menu, select Programming File Generator. The new window appears.
2. In the Device family, select Agilex.
3. For the Configuration mode, select Active Serial x4.
4. Under Output files tab, specify the following parameters:
   a. Specify the Output directory and Name for the output file.
      Note: The output directory you specify must already exist in the file system.
   b. Select Raw Binary File for CvP Core Configuration (.rbf).
   c. Select JTAG Indirect Configuration File for Periphery Configuration (.jic) if you want to use Active Serial configuration mode.
   d. Select Memory Map File (.map) or Raw Programming Data File (.rpd) if you plan to use third party programmer for flash programming.
5. Under the **Input Files** tab, click **Add Bitstream**. Navigate your file system, and select the .sof file, and click **Open**.

6. Under the **Configuration device** tab:
   a. Click **Add Device**.
   b. Under the **Configuration Device** tab, click to select your configuration device and click **OK**.
   c. Click to select the configuration device in the list and click **Add Partition**.
   d. In the **Add Partition** window, select the file in the **Input file** box, select **Start** in the **Address Mode** box, and then click **OK**.
   e. Click **Select**.
   f. In the **Select Devices** window, click **Agilex** in the device family list, select your flash loader device in the **Device name** list, and then click **OK**.
6. Understanding the Design Steps for CvP Initialization using the P-tile in Intel Agilex Devices

Figure 13. Programming File Generator- Configuration Device Tab

7. Click **Generate**.

6.1.5. Bringing up the Hardware

Before testing the design in hardware, you must install the CvP driver in your DUT system. You can also install RW Utilities or other system verification tools to monitor the link status of the Endpoint and to observe traffic on the link. You can download these utilities for free from many web sites.

*Note:* You can develop your own custom CvP driver for Linux using the sample Linux driver source code provided by Intel.

6.1.5.1. Installing Open Source CvP Driver in Linux Systems

1. Download the open source Linux CvP driver from the [CvP Driver](#).
2. Navigate to the driver directory.
3. Unzip the drive by typing the following command:
   
   ```
   tar -xjvf <driver>.gz
   ```
4. Run the installation by typing the following command:
   
   ```
   sudo make
   sudo make install
   ```
5. Once the installation completed successfully, it generates the `altera_cvp` file under directory `/dev/altera_cvp`.

6.1.5.2. Setting up the Correct MSEL Switch State

Select Active Serial x4 (Fast mode) for CvP operation.
Table 15. MSEL Pin Settings for Active Serial x4 (Fast mode) Scheme of Intel Agilex Devices

<table>
<thead>
<tr>
<th>Configuration Scheme</th>
<th>MSEL[2:0]</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS (Fast mode - for CvP)</td>
<td>001</td>
</tr>
</tbody>
</table>

Related Information

Intel Agilex Device Family Pin Connection Guidelines

6.1.5.3. Programming CvP Images

In Active Serial configuration mode, you must program the periphery image (.periph.jic) into your AS configuration device and then download the core image (.core.rbf) using the PCIe Link. You can use Active Serial x4 (Fast mode) to load .periph.jic into your selected CvP initialization enabled Intel Agilex device.

After loading the periphery image, the Intel Agilex device is triggered to reconfigure from AS to load it. The link should reach the expected data rate and link width. You can confirm the PCIe link status using the RW Utilities. Follow these steps to program and test the CvP functionality:

1. Plug the Intel Agilex device PCIe card into the PCI Express slot of the DUT PC and power it ON.
2. Open the Intel Quartus Prime Tools menu and select Programmer.
3. Click Auto Detect to verify that the Intel FPGA Download Cable recognizes the Intel Agilex device.
4. Follow these steps to program the periphery image:
   a. Select Intel Agilex device, and then right click None under File column and select Change File.
   b. Navigate to .periph.jic file and click Open.
   c. Under Program/Configure column, select the respective devices.
   d. Click Start to program the periphery image into flash.
5. After the .periph.jic is programmed, the FPGA must be powered cycle to allow the new peripheral image to load from the on-board flash into the FPGA. To force the DUT PC to re-enumerate the link with the new image, power cycle the DUT PC and the Intel Agilex device PCIe card.
6. You can use RW Utilities or another system software driver to verify the link status. You can also confirm expected link speed and width.
7. Follow these steps to program the core image:
   a. Copy the .core.rbf file to your working directory.
   b. Open a console in Linux. Change the directory to the same mentioned above where the file is copied.

(5) To support AS fast mode, the VCCIO_SDM of Intel Agilex device must be fully ramped-up within 10ms to the recommended operating conditions. The delay between the device exiting POR and the SDM Boot-up is shorter for the fast mode compared to the normal mode. Therefore, AS fast mode is the recommended configuration scheme for CvP because the device can conform to the PCIe 100ms power-up-to-active time requirement.
c. Program the core image by typing the following command:

```
 dd if=<filename>.core.rbf of=/dev/altera_cvp bs=4K
```

8. You can see your core image running on the Intel Agilex device PCIe card. Alternatively, print out the kernel message using the `dmesg` to ensure the CvP is completed successfully.

<table>
<thead>
<tr>
<th>Document Version</th>
<th>Intel Quartus Prime Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020.01.07</td>
<td>19.4</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>