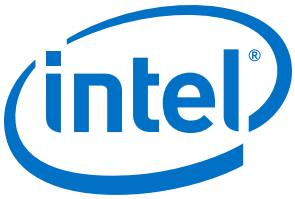


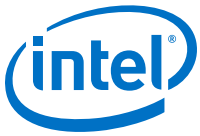
Intel® Stratix® 10 Configuration via Protocol (CvP) Implementation User Guide

Updated for Intel® Quartus® Prime Design Suite: **Quartus Prime Pro 17.1**

Subscribe
Send Feedback

UG-20045 | 2017.12.18
Latest document on the web: PDF | HTML

Contents



1 Overview

Configuration via Protocol (CvP) is a configuration scheme supported in Arria® V, Cyclone® V, Stratix® V, Intel® Arria 10, Intel Stratix 10, and Intel Cyclone 10 GX 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 FPGA fabric through the PCI Express* (PCIe*) link and is available for Endpoint variants only.

Related Links

- Arria 10 CvP Initialization and Partial Reconfiguration over PCI Express User Guide
  Provides more information about the CvP implementation in Arria 10 devices.
- Configuration via Protocol (CvP) Implementation in V-series FPGA Devices User Guide
  Provides more information about the CvP implementation in V-series FPGA devices.

1.1 Benefits of Using CvP

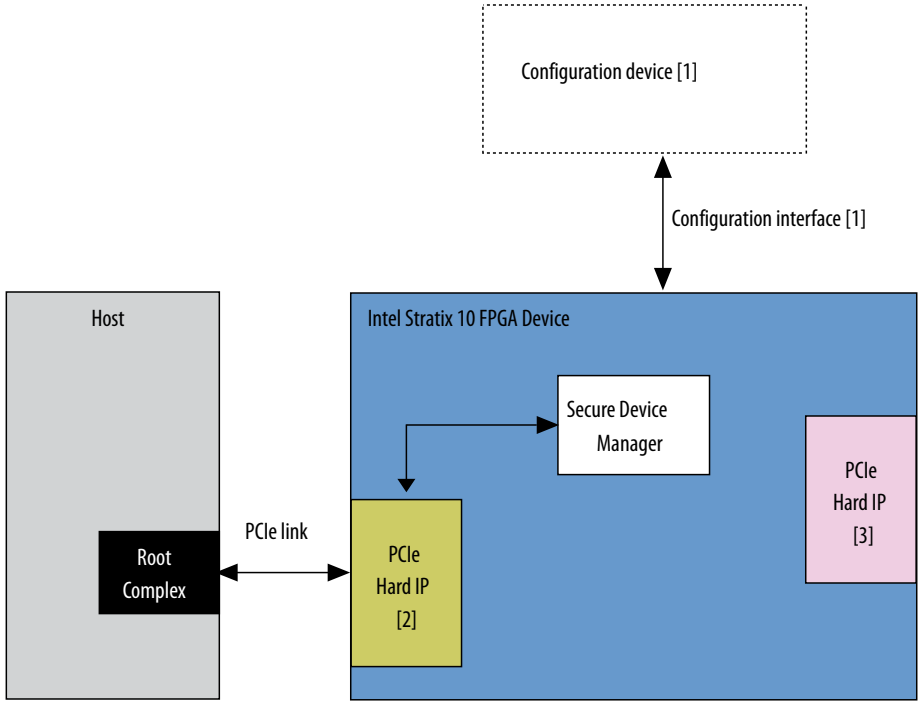
The CvP configuration scheme has the following advantages:

- Reduces system costs by reducing the size of the local flash device used to store the configuration data. The smallest EPCQ device is large enough for all Intel Stratix 10 periphery images.
- Allows update of the FPGA without reprogramming the flash.
- Enables dynamic core updates without requiring a system power down. CvP allows the FPGA fabric to be updated through the PCIe link without a host reboot 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 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.

Figure 1. CvP Block Diagram



1. The configuration device is connected to the FPGA using the fast active serial (ASx4) configuration scheme.
2. PCIe Hard IP block (bottom left) is use for CvP and other PCIe applications.
   - Most Intel Stratix 10 FPGAs include more than one Hard IP block for PCI Express. The CvP configuration scheme can only utilize the bottom left PCIe Hard IP block on each device. It must be configured as an Endpoint.
3. Other PCIe Hard IP blocks can only be used for PCIe applications and cannot be used for CvP.

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 and any PCIe cores (peripheral image) of the FPGA through the PCIe link upon system power up.

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

This mode assumes that you have configured the FPGA with the full configuration image (both periphery and core) after the initial system power up. The PCIe link is used for subsequent core image updates (only core, the periphery must remain unchanged during CvP update).

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

- To change core algorithms
- 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 works 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 Stratix 10 Device Family

PCIe Version	Supported CvP Modes
Gen 1 / Gen 2 / Gen 3	CvP Initialization, CvP Update(1)

(1) For more information or questions about the availability of the CvP update flow, please contact mySupport .

2 CvP Description

2.1 Configuration Images

In CvP, you partition your design into two images: core image and periphery image.

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

- Periphery image (`*.periph.jic`)— contains bottom left transceiver tiles I/O settings. The entire periphery image is static and cannot be reconfigured.
- Core image (`*.core.rbf`)—contains logic that is programmed by the configuration RAM (CRAM). This image includes everything other than bottom left transceiver tiles I/O settings.

2.2 CvP Modes

2.2.1 CvP Initialization Mode

In this mode, the periphery image is stored in an external configuration device and is loaded into the FPGA through the Active Serial x4 (Fast mode) configuration scheme. The core image is stored in a host memory and is loaded into the FPGA through the PCIe link.

After the periphery image configuration is complete, the `CONF_DONE` signal goes high and allows the FPGA to start PCIe link training. When PCIe link training is complete, the PCIe link transitions to `L0` state and then through PCIe enumeration. The PCIe host then initiates the core image configuration through the PCIe link.(2)

After the core image configuration is complete, the `CvP_CONFDONE` 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 has entered user mode.

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

2.2.2 CvP Update Mode

In this mode, the FPGA device is initialized after initial system power up by loading the full configuration image from the external local configuration device to the FPGA or after the CvP initialization.

After the full FPGA configuration image is complete, the `CONF_DONE` signal goes high.

(2) PCIe REFCLK needs to be running for the link to be trained.

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

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

2.3 Compression and Encryption Features

Data Compression

The Intel Quartus Prime Pro Edition software compresses all Intel Stratix 10 bitstreams to reduce the storage requirement and increase bitstream processing speed. Compressing the periphery and core images.

Data Authentication and Encryption

Secure Device Manager (SDM) supports various enhanced security features which are also supported in CvP. You can choose to encrypt the core and peripheral images. To configure the FPGA with an encrypted core image, you must pre-program the FPGA with authentication and encryption keys. These keys are then used to authenticate and decrypt the incoming configuration bitstream.

A key-programmed FPGA can only accept signed and optionally encrypted bitstreams. Use the same key to encrypt all revisions of the periphery and core image.

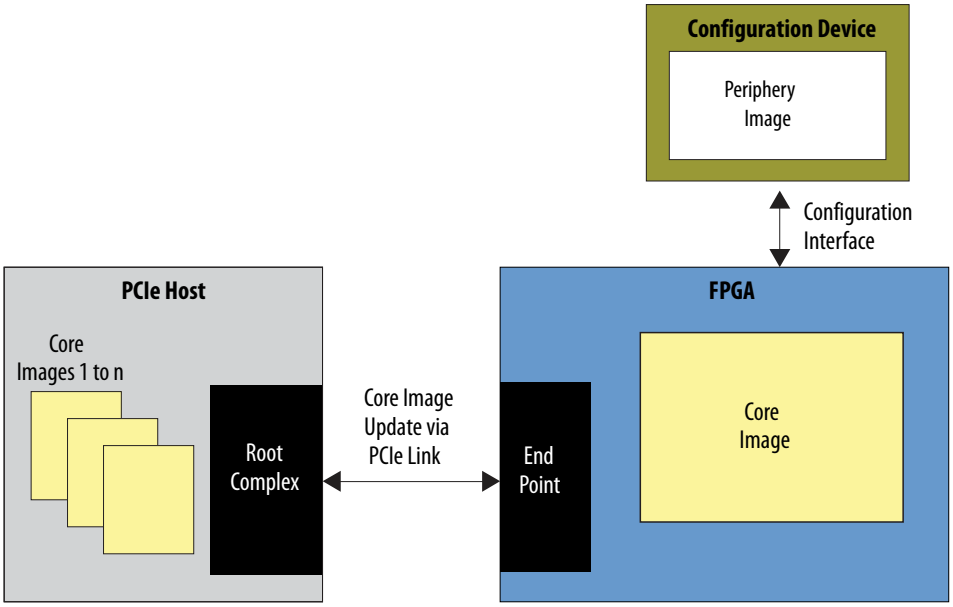
2.4 Core Image Update

After the FPGA enters user mode, the PCIe host can trigger an FPGA core image update through the PCIe link. Both CvP Initialization mode and CvP update mode support core images updates.

You must choose the same bitstream settings for all core images. For example, if you have selected either encryption, compression, or both encryption and compression features for the first core image, you must ensure you turned on the same features for the other core images that you will use for core image update using CvP.

Figure 2. Periphery and Core Images 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.



You can use CvP revision design flow to create multiple reconfigurable core images that connect to the same periphery image.

2.5 Pin Description

The following table lists the CvP pin descriptions and connection guidelines.

Table 2. CvP Pin Descriptions and Connection Guidelines

Pin Name	Pin Type	Pin Description	Pin Connection
CvP_CONFDONE	Output	The CvP_CONFDONE pin is driven low during configuration. When configuration via PCIe is complete, this signal is actively driven high. During FPGA configuration in CvP Initialization mode, you may observe this pin after the CONF_DONE pin goes high to determine if the FPGA is successfully configured.	If this pin is set as dedicated output, the VCCIO_SDM power supply must meet the input voltage specification of the receiving side. You can assign SDM_IO0, SDM_IO10, SDM_IO11, SDM_IO12, SDM_IO13, SDM_IO14, SDM_IO15 or SDM_IO16 as CvP_CONFDONE in Intel Quartus Prime Pro Edition software If you are not using the CvP modes, you can use this pin as a user I/O pin.
INIT_DONE	Output	The INIT_DONE pin may goes high indicating the device has enter user mode upon completion of configuration.	Intel recommends to use SDM_IO0 pin for implementing the INIT_DONE function, provided that this function is enabled in the Intel Quartus Prime Pro Edition software. This pin has a weak pull-down for the correct function during power up.

continued...

Pin Name	Pin Type	Pin Description	Pin Connection
			The INIT_DONE function can also be implemented using other unused SDM I/O pins (with a weak pull-down).
CONF_DONE	Output	The CONF_DONE pin drives low before and during configuration. After all configuration data is received without error and the initialization cycle starts, CONF_DONE is driven high.	Intel recommends to use SDM_IO16 pin implementing the CONF_DONE function, provided that this function is enabled in the Intel Quartus Prime Pro Edition software.
nPERST[L,R][0:2]	Input	Dual-purpose fundamental reset pin is only available when you use PCI Express hard IP. When the PCIe hard IP on a side (left or right) is enabled, then nPERST pins on that side cannot be used as general-purpose I/Os (GPIOs). In this case, connect the nPERST pin to the system PCIe nPERST signal to ensure that both ends of the link start link-training at the same time. The nPERST pins on a side are available as GPIOs only when the PCIe hard IP on that side is not enabled. When this pin is low, the transceivers are in reset. When this pin is high, the transceivers are out of reset. When you do not use this pin as the fundamental reset, you can use this pin as a user I/O pin	Connect this pin as defined in the Intel Quartus Prime Pro Edition software. For more details, refer to *Intel Stratix 10 Avalon®-MM/ST Interface for PCIe Solutions User Guide*. This pin is powered by the VCCIO3V supply. When VCCIO3V is connected to a 3.0-V supply, you must use a diode to clamp the 3.3V LVTTL PCIe input signal to the VCCIO3V power of the device. When VCCIO3V is connected to any voltage other than 3.0V, you must use a level translator to shift down the voltage from 3.3V LVTTL to the corresponding voltage level powering the VCCIO3V pin. Only one nPERST pin is used per PCIe hard IP. The Intel Stratix 10 device components may have all six pins listed even when the specific component might only have 1 or 2 PCIe hard IPs. • nPERSTL0 = Bottom Left PCIe hard IP & CvP • nPERSTL1 = Middle Left PCIe hard IP (When available) • nPERSTL2 Top Left PCIe hard IP (When available) • nPERSTR0 = Bottom Right PCIe hard IP (When available) • nPERSTR1 = Middle Right PCIe hard IP (When available) • nPERSTR2 = Top Right PCIe hard IP (When available) *Note:* For maximum compatibility, always use the bottom left PCIe Hard IP first, as this is the only location that supports Configuration via Protocol (CvP) using the PCIe link.

Related Links

- Intel Stratix 10 Avalon-MM Interface for PCIe Solutions User Guide
- Intel Stratix 10 Avalon-ST and Single Root I/O Virtualization (SR-IOV) Interface for PCIe Solutions User Guide
- Intel Stratix 10 GX, MX, and SX Device Family Pin Connection Guidelines

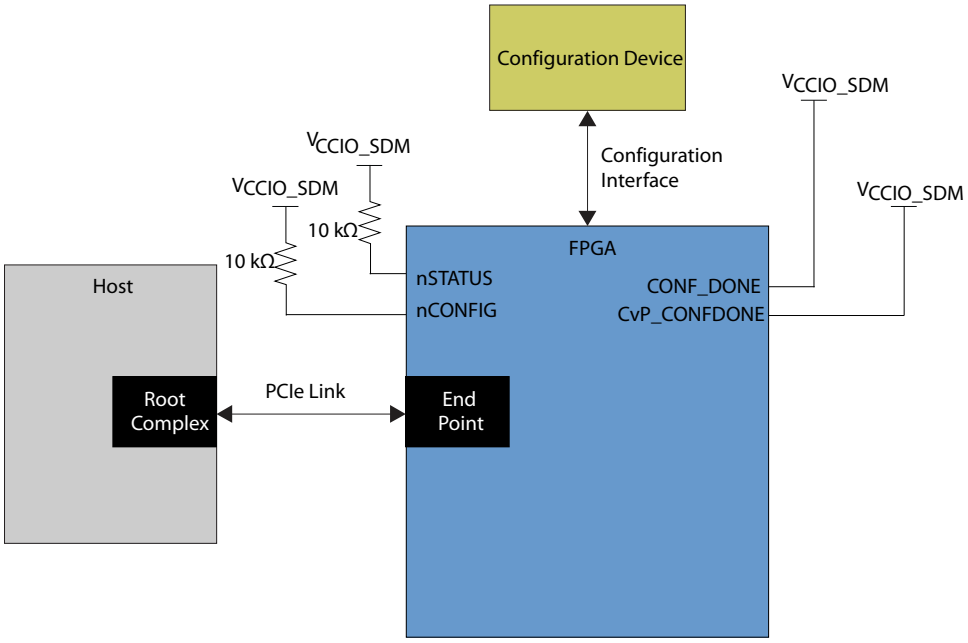
3 CvP Topologies

CvP supports two types of topologies that allow you to configure single or multiple FPGAs.

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.

Figure 3. Single Endpoint Topology

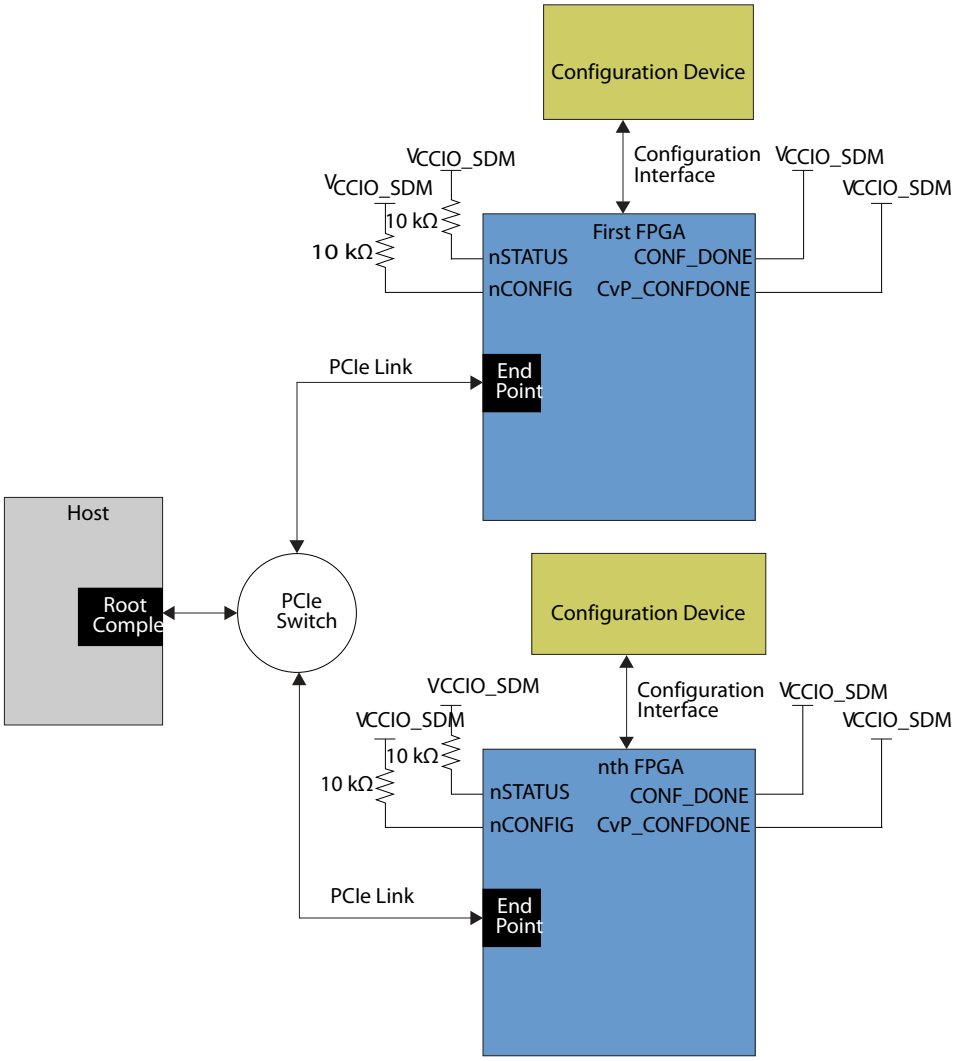


3.2 Multiple Endpoints

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 configure 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



4 Design Considerations

4.1 Designing CvP for an Open System

While designing a CvP system for an open system where you don't control both ends of the PCIe link completely, ensure that you observe the guidelines provided in this section.

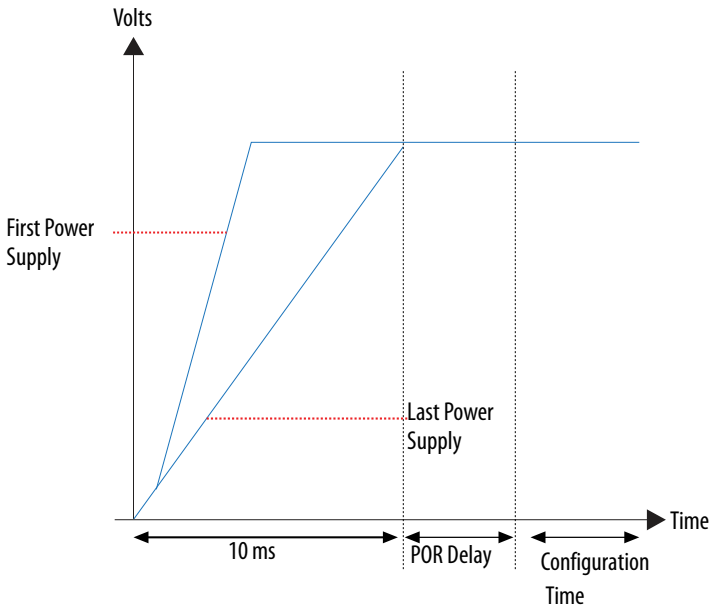
4.1.1 FPGA Power Supplies Ramp Time Requirement

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_{RAMP}. If t_{RAMP} is not met, the device I/O pins and programming registers remain tri-stated, during which device configuration could fail.

To meet the PCIe link up time for CvP, the total t_{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. Power Supplies Ramp-Up Time and POR



Related Links

Intel Stratix 10 Power Management User Guide

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 Links

PCI Express Card Electromechanical 3.0 Specification

4.1.2.1 For CvP Initialization Mode

For CvP Initialization mode, the Hard IP for PCI Express IP core is guaranteed to meet the 120 ms requirement because the periphery image configuration time is significantly less than the full FPGA configuration time. You can 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[3]. 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 PCIe applications in user mode.

Note: For Gen 2/Gen3 capable Endpoints, after loading the core SRAM Object File (.sof), Intel recommends to verify that the link has been trained to the expected Gen 2/Gen3 rate. If the link is not operating at Gen 2/Gen3, software can trigger the Endpoint to retrain.

4.1.2.2 For CvP Update Mode

Before you peform CvP update mode, the device must be in user mode either through CvP initialization or full image configuration (Active Serial x4 fast mode).

Note: For Gen 2/Gen3 capable Endpoints, after loading the core `.sof`, Intel recommends to verify that the link has been trained to the expected Gen 2/Gen3 rate. If the link is not operating at Gen 2/Gen3, 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.

---

[3] REFCLK is stable 80 ms after the power supplies are stable in order to hit the 145 ms link training complete time

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.

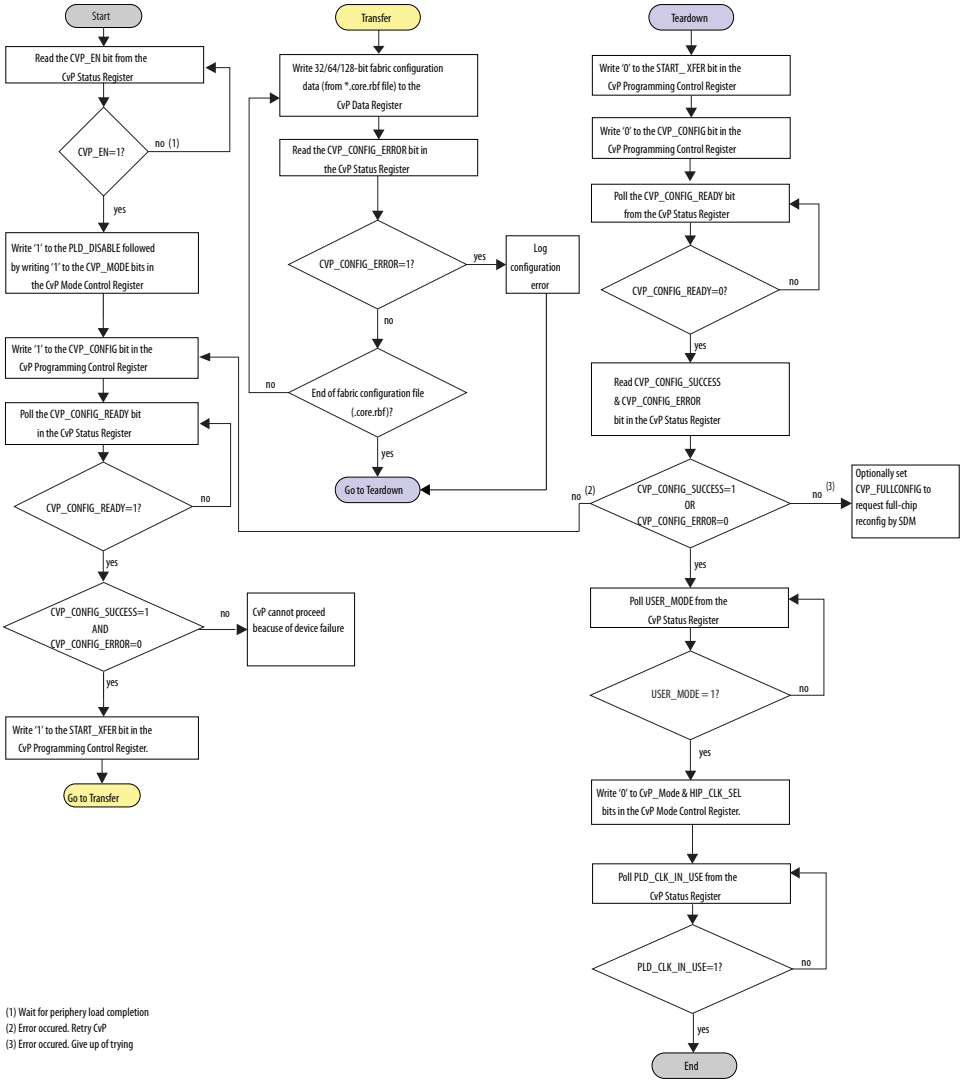
Related Links

Download the OpenSource Linux CvP Driver

5.2 CvP Driver Flow

The following figure shows the flow of the provided CvP driver. The 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. The dummy writes cause a 2 ms delay, allowing the SDM to complete required operations.

Figure 8. CvP Driver Flow



5.3 VSEC Registers for CvP

The Vendor Specific Extended Capability (VSEC) registers occupy byte offsets 0xB80 to 0xBC0 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

Byte Offset	Register Name
0xB80	Vendor Specific Capability Header
0xB84	Vendor Specific Header
0xB88	Intel Marker
	continued...

Byte Offset	Register Name
0xB8C:0xB98	Reserved
0xB9C	User Configurable Device/Board ID
0xB9E	CvP Status
0xBA0	CvP Mode Control
0xBA4	CvP Data 2
0xBA8	CvP Data
0xBAC	CvP Programming Control
0xBB0	General Purpose Control and Status Register
0xBB4	Uncorrectable Internal Error Status Register
0xBB8	Uncorrectable Internal Error Mask Register
0xBCC	Correctable Internal Error Status Register
0xBC0	Correctable Internal Error Mask Register

5.3.1 Vendor Specific Capability Header Register

Table 6. Vendor Specific Capability Header Register (Byte Offset: 0xB80)

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

5.3.2 Vendor Specific Header Register

Table 7. Vendor Specific Header Register (Byte Offset: 0xB84)

Bits	Name	Reset Value	Access	Description
[15:0]	VSEC ID	0x1172	RO	A user configurable VSEC ID.
[19:16]	VSEC Revision	0	RO	A user configurable VSEC revision.
[31:20]	VSEC Length	0x05C	RO	Total length of this structure in bytes.

5.3.3 Intel Marker Register

Table 8. Intel Marker Register (Byte Offset: 0xB88)

Bits	Name	Reset Value	Access	Description
[31:0]	Intel Marker	0x41721172	RO	An additional marker. If you use the standard Intel Programmer software to configure the device with CvP, this marker provides a value that the programming software reads to ensure that it is operating with the correct VSEC.

5.3.4 User Configurable Device/Board ID Register

Table 9. User Configurable Device/Board ID Register (Byte Offset: 0xB9C)

Bits	Name	Reset Value	Access	Description
[15:0]	User Configurable Device/ Board ID	0x00	RO	Helps user to select the correct programming file.

5.3.5 CvP Status Register

Table 10. CvP Status Register (Byte Offset: 0xB9E)

Bits	Name	Reset Value	Access	Description
[15]	CVP_AVMM_XFER_PENDING	0x00	RO	Deasserts before starting tear-down procedures
[14:11]	—	Variable	RO	Reserved.
[10]	CVP_CONFIG_SUCCESS	Variable	RO	Status bit set by the SDM to indicate that the core image configuration was successful.
[9]	—	Variable	RO	Reserved.
[8]	PLD_CLK_IN_USE	Variable	RO	From clock switch module to fabric. This status bit is provided for debug.
[7]	CVP_CONFIG_DONE	Variable	RO	Indicates that the SDM has completed the device configuration via CvP and there were no errors.
[6]	—	Variable	RO	Reserved.
[5]	USERMODE	Variable	RO	Indicates if the configurable FPGA fabric is in user mode.
[4]	CVP_EN	Variable	RO	Indicates if the SDM has enabled CvP mode.
[3]	CVP_CONFIG_ERROR	Variable	RO	Reflects the value of this signal from the SDM, checked by software to determine if there was an error during configuration.
[2]	CVP_CONFIG_READY	0x0	RO	Reflects the value of this signal from the SDM, checked by software during programming algorithm.
[1]	CvP Data Compressed	0x0	RO	Indicates to the host driver that CVP data should be compressed.
[0]	CvP Data Encrypted	Variable	RO	Indicates to the host driver that CVP data should be encrypted.

5.3.6 CvP Mode Control Register

Table 11. CvP Mode Control Register (Byte Offset: 0xBA0)

Bits	Name	Reset Value	Access	Description
[31:3]	—	0x0000	RO	Reserved.
[2]	CVP_FULLCONFIG	1'b0	RW	A value of 1 indicates a request to the control block to reconfigure the entire FPGA including the Hard IP for PCI Express and bring the PCIe link down.
[1]	PLD_DISABLE	1'b0	RW/RO	Enables/Disables the PLD interface. This allows Host driver to switch the PLD interface out before USER MODE deasserts, 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. The value of this signal should only be changed 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 is changed. When entering CVP, this bit should be set before CVP_MODE is set. When exiting CVP, it should be cleared after CVP_MODE is cleared. This ensures that there is no PLD switching during CVP. This field is RW when cvp_en=1, and RO when cvp_en=0.
[0]	CVP_MODE	1'b0	RW	Controls whether the Hard IP for PCI Express is in CVP_MODE or normal mode. The following encodings are defined: • 1: CVP_MODE is active. Signals to the SDM active and all TLPs are routed to the Configuration Space. This CVP_MODE cannot be enabled if CVP_EN = 0. • 0: The IP core is in normal mode and TLPs are route to the FPGA fabric.

5.3.7 CvP Data Registers

Table 12. CvP Data Register (Byte Offsets: 0xBA4 - 0xBA8)

Bits	Name	Reset Value	Access	Description
[31:0]	CVP_DATA2	0x00000000	RW	Contains the upper 32 bits of a 64-bit configuration data. Software must ensure that all Bytes in both dwords are enabled. Use of 64-bit configuration data is optional.
[31:0]	CVP_DATA	0x00000000	RW	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

continued...

Bits	Name	Reset Value	Access	Description
				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.

5.3.8 CvP Programming Control Register

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

Bits	Name	Reset Value	Access	Description
[31:2]	—	0x0000	RO	Reserved.
[1]	START_XFER	1'b0	RW	When asserted, indicates that the host started configuration data transfer. When deasserted, indicates that the configuration data transfer has been completed.
[0]	CVP_CONFIG	1'b0	RW	When set to 1, indicates that the host request for performing CvP.

5.3.9 General Purpose Control and Status Register

Table 14. General Purpose Control and Status Register (Byte Offset: 0xBB0)

Bits	Name	Reset Value	Access	Description
[31:16]	—	0x0000	RO	Reserved.
[15:8]	General Purpose Status Register	1'b0	RO	It wires directly to the HIP INPUT port.
[7:0]	General Purpose Control Register	1'b0	RW	It wires directly to the HIP OUTPUT port.

5.3.10 Uncorrectable Internal Error Status Register

This register reports the status of the internally checked errors that are uncorrectable. When specific errors are enabled by the `Uncorrectable Internal Error Mask` register, they are handled as Uncorrectable Internal Errors as defined in the *PCI Express Base Specification 3.0*. This register is for debug only. Use this register to observe behavior, not to drive custom logic.

Table 15. Uncorrectable Internal Error Status Register (Byte Offset: 0xBB4)

Bits	Reset Value	Access	Description
[31:12]	0x00	RO	Reserved.
[11]	1'b0	RW1CS	A value of 1 indicates the ECC error from Configuration RAM block.
[10]	1'b0	RW1CS	A value of 1 indicates the uncorrectable ECC error status for Retry Buffer.
[9]	1'b0	RW1CS	A value of 1 indicates the uncorrectable ECC error status for Retry Start of TLP RAM.
[8]	1'b0	RW1CS	A value of 1 indicates a parity error was detected on the RX TLP.

continued...

Bits	Reset Value	Access	Description
[7]	1'b0	RW1CS	A value of 1 indicates a parity error was detected in a TX TLP and the TLP is not sent.
[6]	1'b0	RW1CS	A value of 1 indicates that the Application Layer has detected an uncorrectable internal error.
[5]	1'b0	RW1CS	A value of 1 indicates a configuration error has been detected in CvP mode which is reported as uncorrectable. This CVP_CONFIG_ERROR_LATCHED bit is set whenever a CVP_CONFIG_ERROR is asserted while in CVP_MODE.
[4]	0x00	RO	Reserved.
[3]	1'b0	RW1CS	A value of 1 indicates the uncorrectable ECC error status for RX Buffer Header 2.
[2]	1'b0	RW1CS	A value of 1 indicates the uncorrectable ECC error status for RX Buffer Header 1.
[1]	1'b0	RW1CS	A value of 1 indicatesthe uncorrectable ECC error status for RX Buffer Data 2.
[0]	1'b0	RW1CS	A value of 1 indicates the uncorrectable ECC error status for RX Buffer Data 1.

5.3.11 Uncorrectable Internal Error Mask Register

This register controls which errors are forwarded as internal uncorrectable errors. With the exception of the configuration errors detected in CvP mode, all of the errors are severe and may place the device or PCIe link in an inconsistent state. The configuration error detected in CvP mode may be correctable depending on the design of the programming software.

Table 16. Uncorrectable Internal Error Mask Register (Byte Offset: 0xBB8)

Bits	Reset Value	Access	Description
[31:12]	0x00	RO	Reserved.
[11]	1'b1	RWS	Mask for ECC error from Configuration RAM block.
[10]	1'b1	RWS	Mask for uncorrectable ECC error status for Retry Buffer.
[9]	1'b1	RWS	Mask for uncorrectable ECC error status for Retry Start of TLP RAM.
[8]	1'b1	RWS	Mask for parity error on the RX TLP.
[7]	1'b1	RWS	Mask for parity error in the transaction layer packet.
[6]	1'b1	RWS	Mask for parity error in the application layer.
[5]	1'b0	RWS	Mask for configuration error in CvP mode.
[4]	0x00	RO	Reserved.
[3]	1'b1	RWS	Mask for uncorrectable ECC error status for RX Buffer Header 2.
[2]	1'b1	RWS	Mask for uncorrectable ECC error status for RX Buffer Header 1.
[1]	1'b1	RWS	Mask for uncorrectable ECC error status for RX Buffer Data RAM 2.
[0]	1'b1	RWS	Mask for uncorrectable ECC error status for RX Buffer Data RAM 1.

5.3.12 Correctable Internal Error Status Register

This register reports the status of the internally checked errors that are correctable. When these specific errors are enabled by the `Correctable Internal Error Mask` register, they are forwarded as Correctable Internal Errors as defined in the *PCI Express Base Specification 3.0*. This register is for debug only. Use this register to observe behavior, not to drive custom logic.

Table 17. Correctable Internal Error Status Register (Byte Offset: 0xBBC)

Bits	Reset Value	Access	Description
[31:12]	0x000	RO	Reserved.
[11]	1'b0	RW1CS	A value of 1 indicates the correctable ECC error status for Configuration RAM.
[10]	1'b0	RW1CS	A value of 1 indicates the correctable ECC error status for Retry Buffer.
[9]	1'b0	RW1CS	A value of 1 indicates the correctable ECC error status for Retry Start of TLP RAM.
[8:7]	0x000	RO	Reserved.
[6]	1'b0	RW1CS	A value of 1 indicates that the Application Layer has detected a correctable internal error.
[5]	1'b0	RW1CS	A value of 1 indicates a configuration error has been detected in CvP mode, which is reported as correctable. This bit is set whenever a CVP_CONFIG_ERROR occurs while in CVP_MODE.
[4]	0x0	RO	Reserved.
[3]	1'b0	RW1CS	A value of 1 indicates an RX Buffer Header RAM 2 correctable ECC error status.
[2]	1'b0	RW1CS	A value of 1 indicates an RX Buffer Header RAM 1 correctable ECC error status.
[1]	1'b0	RW1CS	A value of 1 indicates an RX buffer Data RAM 2 correctable ECC error status.
[0]	1'b0	RW1CS	A value of 1 indicates an RX Buffer Data RAM 1 correctable ECC error status.

5.3.13 Correctable Internal Error Mask Register

This register controls which errors are forwarded as Internal Correctable Errors. This register is for debug only.

Table 18. Correctable Internal Error Mask Register (Byte Offset: 0xBC0)

Bits	Reset Value	Access	Description
[31:12]	0x000	RO	Reserved.
[11]	1'b0	RWS	Mask for correctable ECC error from Configuration RAM block.
[10]	1'b1	RWS	Mask for correctable ECC error status for Retry Buffer.
[9]	1'b1	RWS	Mask for correctable ECC error status for Retry Start of TLP RAM.
[8:7]	0x000	RO	Reserved.
[6]	1'b0	RWS	Mask for corrected internal error reported by the Application Layer.

continued...

Bits	Reset Value	Access	Description
[5]	1'b0	RWS	Mask for configuration error detected in CvP mode.
[4]	0x0	RO	Reserved.
[3]	1'b1	RWS	Mask for correctable ECC error status for RX Buffer Header 2.
[2]	1'b1	RWS	Mask for correctable ECC error status for RX Buffer Header 1.
[1]	1'b1	RWS	Mask for correctable ECC error status for RX Buffer Data RAM 2.
[0]	1'b1	RWS	Mask for correctable ECC error status for RX Buffer Data RAM 1.

6 Understanding the Design Steps for CvP Initialization and Update Mode in Intel Stratix 10

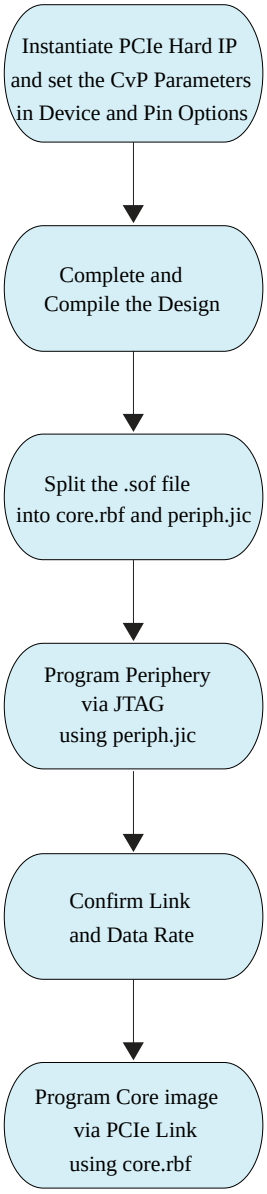
6.1 Implementation of CvP Initialization Mode

CvP Initialization mode divides the design 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 Power up and subsequent core configuration and also instantiate the **Avalon-ST Stratix 10 Hard IP for PCI Express**(4). You might choose CvP Initialization to prevent unauthorized access to the core image as well as save cost by storing the core image in the host memory.

(4) CvP is also supported in the Avalon-MM.

Figure 9. Design Flow for CvP Initialization



The CvP Initialization demonstration walkthrough includes the step mentioned in the following sections:

6.1.1 Generating the Synthesis HDL files for Intel Stratix 10 Hard IP for PCI Express IP Core

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, select **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 **Avalon-ST Stratix 10 Hard IP for PCI Express**. The new window appears.
6. On the **IP Settings** tab, specify the parameters and options for your design variation.
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. For **Target Development Kit**, select the board of your choice.
10. Click the **Generate Example Design** button. The **Select Example Design Directory** window appears. Click **OK**. The software generates Quartus project files for PCI Express reference design. Click **Close** when generation completes. An example design `pcie_s10_hip_ast_0_example_design` is created in your project directory.
11. Click **Finish**. Close your current project and open the generated PCI Express example design (`pcie_example_design.qpf`).
12. 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.

Alternatively, you can download the complete Intel Stratix 10 CvP Initialization reference design from the link below.

Related Links

- Intel Stratix 10 Avalon-MM Interface for PCIe Solutions User Guide
- Intel Stratix 10 Avalon-ST and Single Root I/O Virtualization (SR-IOV) Interface for PCIe Solutions User Guide
- CvP Reference Design Example
- 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:

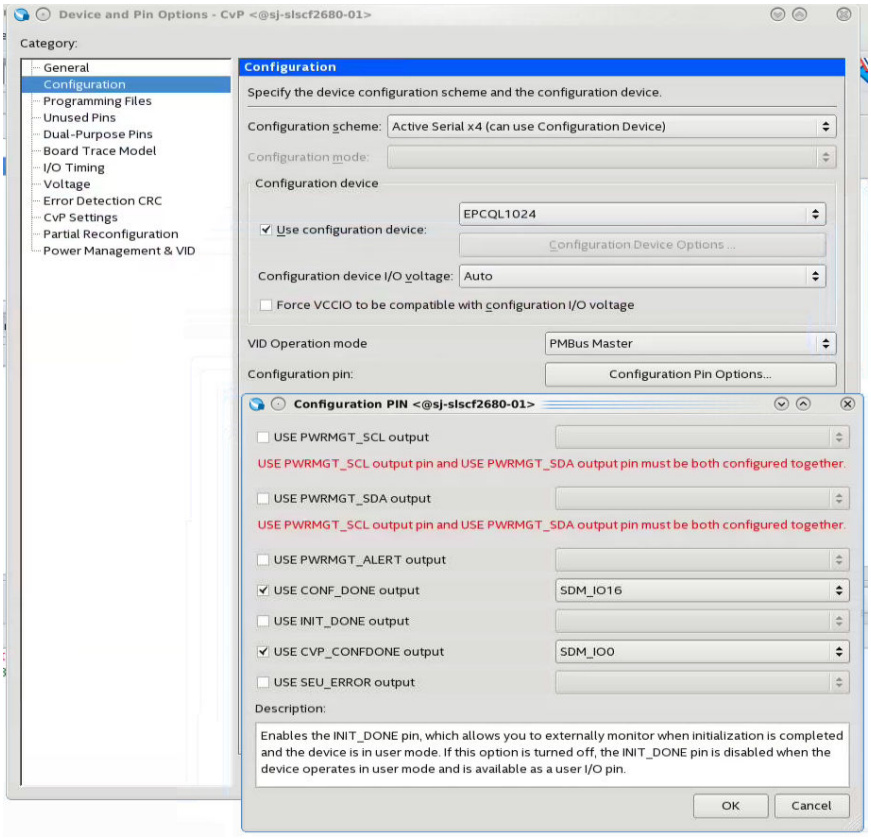
1. On the Intel Quartus Prime **Assignment** menu, select **Device**, and then click **Device and Pin Options**.
2. Under **Category**, select **Configuration** and then enable the following options:

a. For **Configuration scheme**, select **Active Serial x4(can use Configuration Device)**.

b. For **Use configuration device**, select **EPCQL1024**.

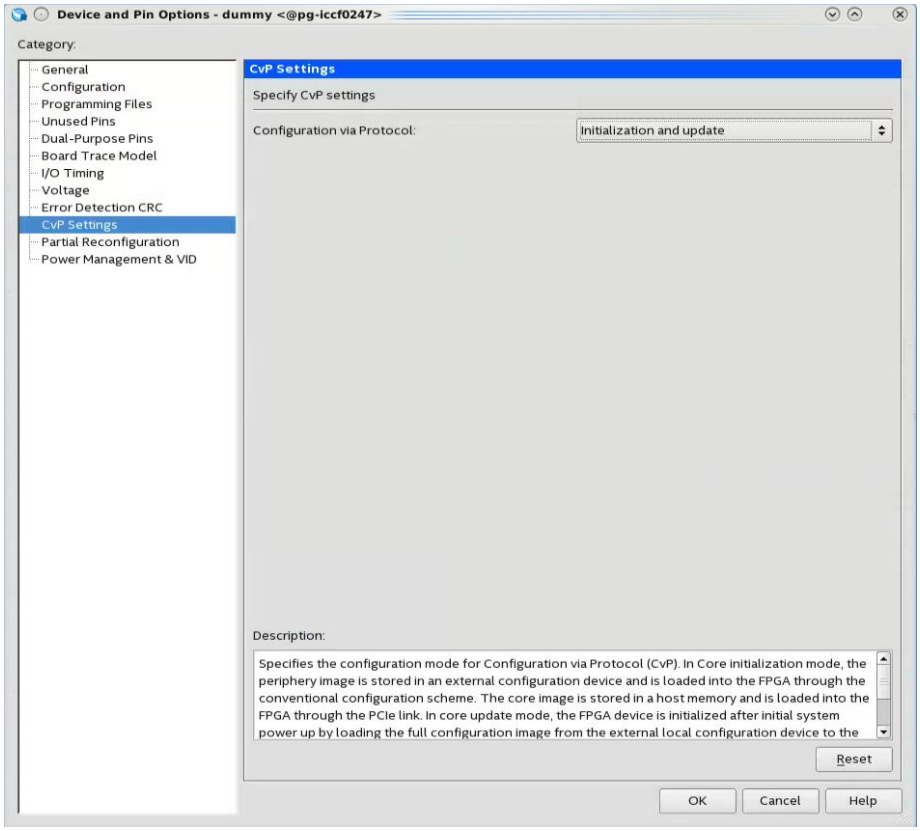
c. For **Configuration pin**, click on **Configuration Pin Options** and then turn on **USE CONF_DONE output** and **USE CVP_CONFDONE output**. Click **OK**.

Figure 10. CvP Parameters in Configuration Tab



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

Figure 11. CvP Parameters in CvP Settings Tab



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 Splitting the SOF File

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

1. After the `.sof` file is generated, under **File** menu, select **Convert Programming Files**. The new window appears.
2. Under **Output programming file** section, specify the following parameters:

Table 19. Parameters: Output Programming File Tab

Parameter	Value
Programming file type	**JTAG Indirect Configuration File (.jic)**
Configuration device	**EPCQL1024**
Mode	**Active Serial x4**
	continued...

Parameter	Value
File name	**output_file.jic**
Create Memory Map File (Generate output_file.map)	Turn this option on.
Create CvP files (Generate cvp_init.periph.jic and cvp_init.core.rbf)	Turn this option on. This option is only available when you specify the **SOF Data** file under **Input files to convert**.

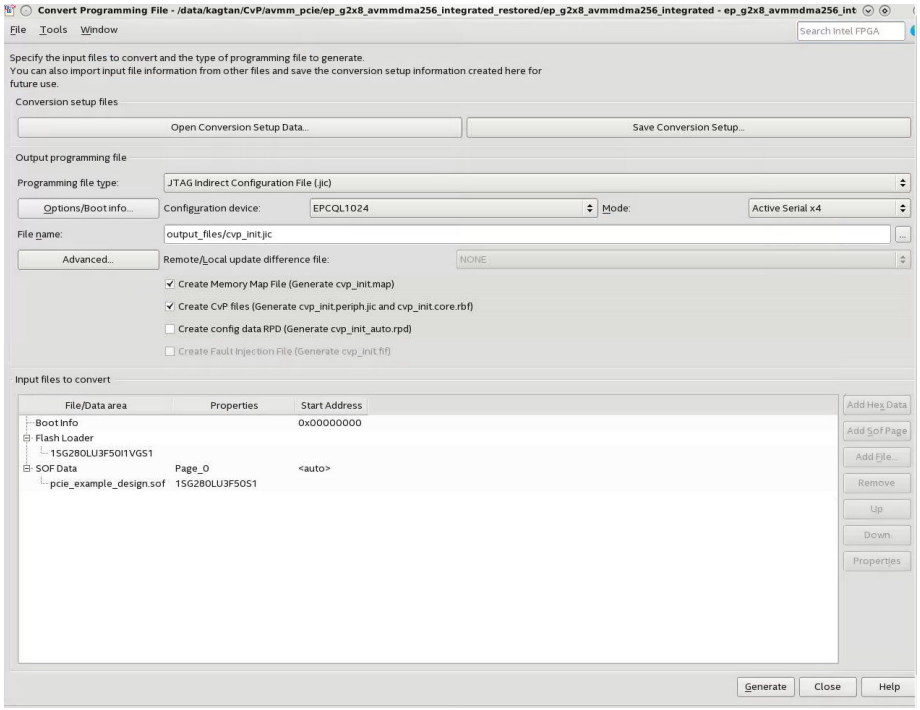
Note: Make sure to turn on the Create CvP files option. If you do not select this option, the Intel Quartus Prime software does not create separate files for the periphery and core images.

3. Under **Input files to convert**, specify the following parameters:

Table 20. Parameters: Input Files to Convert Tab

Parameter	Value
Flash Loader	First click on **Flash Loader**. Click **Add Device**, under **Device family**, select **Stratix 10** and then for **Device name** select **1SG280LU3F50E3VGS1**. Click **OK**.
SOF Data	First click on **SOF Data**. Click **Add File** and then select `*.sof`.

Figure 12. Illustrating the above Specified Options in the Convert Programming File GUI



4. Click **Generate** to create `*.periph.jic` and `*.core.rbf` files.

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.

The test setup includes the following components:

1. Intel Stratix 10 FPGA Development Kit
2. Intel FPGA Download Cable
3. A DUT PC with PCI Express slot to plug in the FPGA Development Kit
4. A host PC running the Intel Quartus Prime software to program the periphery image, `.sof` or `.pof` file.

Although a separate host PC is not strictly necessary, it makes testing less cumbersome.

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 -zxvf <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. Use `*.core.rbf` file to perform the CvP configuration by entering the following command:

```
cp *.core.rbf /dev/altera_cvp
```

7. You will see your logic running once the CvP is successfully configured. Alternatively, you may read out the kernel activity through `dmesg` to ensure the CvP is completed successfully.

6.1.5.2 Modifying MSEL/DIP switch on Intel Stratix 10 FPGA Development Kit

The MSEL/DIP switch labeled SW1 at the front part of the Intel Stratix 10 FPGA Development Kit. Select Active Serial x4 (Fast mode) for CvP operation.

Table 21. MSEL Pin Settings for Each Configuration Scheme of Intel Stratix 10 Devices

Configuration Scheme	MSEL[2:0]
Avalon-ST (x32)	000
Avalon-ST (x16)	101
Avalon-ST (x8)	110
AS (Fast mode - for CvP)(5)	001
AS (Normal mode)(6)	011
NAND x8	010
SD/MMC x4/x8	100
JTAG only(7)	111

Related Links

Intel Stratix 10 GX, MX, and SX Device Family Pin Connection Guidelines

6.1.5.3 Programming CvP Images

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 Stratix 10 device.

After loading the periphery image, the Intel Stratix 10 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 Stratix 10 FPGA Development Kit into the PCI Express slot of the DUT PC and power it ON. It is recommended to use the ATX power supply that the development kit includes.
2. On the host PC, 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 Stratix 10 FPGA.
4. Follow these steps to program the periphery image:
   a. Select **Stratix 10** device, and then right click **None** under **File** column and select **Change File**.

(5) To support AS fast mode, the V_{CCIO_SDM} of Intel Stratix 10 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.

(6) To support AS normal mode, V_{CCIO_SDM} of the Intel Stratix 10 device must be fully ramped-up within 10ms to the recommended operating condition.

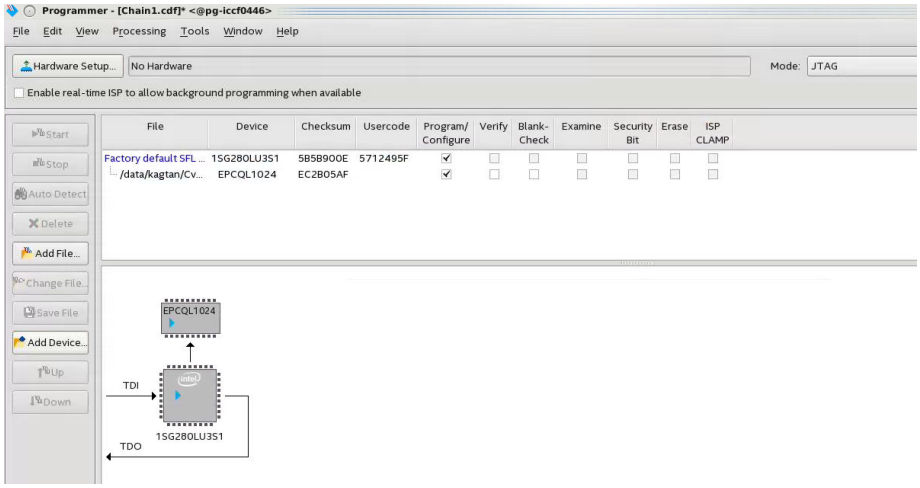
(7) JTAG configuration also works with MSEL settings for other configuration schemes, unless disabled for security

b. Navigate to `.periph.jic` file and click **Open**.

c. Under **Program/Configure** column, select the respective devices. For example, **1SG280LU3S1** and **EPCQL1024**.

d. Click **Start** to program the periphery image into **EPCQL1024** flash.

Figure 13. Illustrating the Specified Options to the Program Periphery Image



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 host PC to re-enumerate the link with the new image, power cycle the DUT PC and the Intel Stratix 10 FPGA Development Kit.
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.
   c. Program the core image by typing the following command: `cp *.core.rbf /dev/altera_cvp`
8. You will see your core image running on the Intel Stratix 10 FPGA Development Kit. Alternatively, print out the kernel message using the `dmesg` to ensure the CvP is completed successfully.

6.2 Implementation of CvP Update Mode

For more information or questions about the availability of the CvP update flow, please contact mySupport .

A Document Revision History for Intel Stratix 10 Configuration via Protocol Implementation User Guide

Date	Version	Changes
December 2017	2017.12.18	Initial release.