Partial Reconfiguration over PCI Express Reference Design for Arria 10 Devices

AN-784
2016.10.31

Subscribe

Send Feedback

Contents

1 Partial Reconfiguration over PCI Express Reference Design for Arria 10 Devices

The Partial Reconfiguration (PR) over PCI Express® (PCIe®) reference design demonstrates reconfiguring the FPGA fabric through the PCIe link in Arria® 10 devices. This reference design runs on a Linux system with the Arria 10 GX FPGA development board. Adapt this reference design to your requirements by implementing the PR region logic using the given template. Run your custom design on this fully functional system that enables communication over PCIe.

Arria 10 devices use the PR over PCIe solution to reconfigure the device, rather than Configuration via Protocol (CvP) update. Partial reconfiguration allows you to reconfigure a portion of the FPGA dynamically, while the remaining FPGA design continues to function. Create multiple personas for a particular region in your design, without impacting operation in areas outside this region. Partial reconfiguration enables the implementation of more complex FPGA systems.

Partial reconfiguration provides the following advancements to a flat design:

- Allows run-time design reconfiguration

- Increases scalability of the design through time-multiplexing

- Lowers cost and power consumption through efficient use of board space

- Supports dynamic time-multiplexing functions in the design

- Improves initial programming time through smaller bitstreams

- Reduces system down-time through line upgrades

- Enables easy system update by allowing remote hardware change

The Quartus® Prime Pro Edition software supports the partial reconfiguration feature for the Arria 10 device family.

Related Links

- Creating a Partial Reconfiguration Design
  For complete information on the partial reconfiguration design flow.

- Partially Reconfiguring a Design on Arria 10 SoC Development Board
  For a step-by-step tutorial on creating a partial reconfiguration design.

- Arria 10 CvP Initialization and Partial Reconfiguration via Protocol User Guide
  For complete information on the Configuration via Protocol (CvP) configuration scheme.

- Arria 10 FPGA Development Kit User Guide
  For information on Arria 10 GX FPGA development board overview and setup procedure.

ISO 9001:2008 Registered

1.1 Reference Design Overview

The reference design consists of the following components:

- PCIe subsystem—contains the Arria 10 Hard IP for PCI Express IP core and the Partial Reconfiguration IP core.

- Freeze wrapper—prevents random toggling of the PR region outputs. The freeze wrapper resides in the static region. The freeze wrapper contains the PR logic.

- PR logic—encloses the PR personas logic. This design contains a single PR region, and three personas. Extend this reference design by reconfiguring this PR logic with your own personas.

- Memory controller subsystem—contains the Arria 10 External Memory Interfaces IP core that provides access to the on-board DDR4 external memory. This external memory connects to the PR region.

Figure 1.   Arria 10 PCIe Reference Design Block Diagram
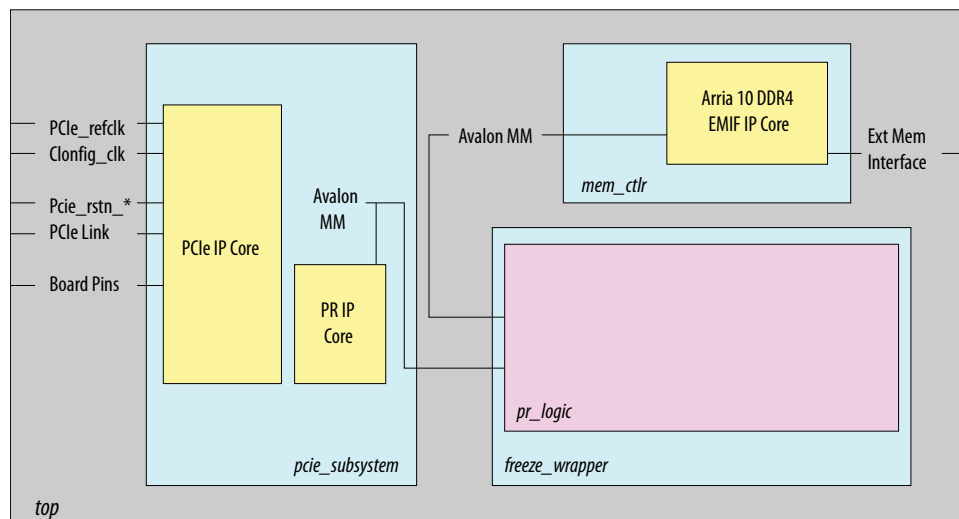
1.1.1 Clocking Scheme

The reference design creates a separate Altera IOPLL IP core-generated clock. This clock creation decouples the PR logic clocking from both the PCIe clocking domain that runs at 250 MHz, and the EMIF clocking domain that runs at 300 MHz. The clock for PR Logic is set at 250 MHz. To ensure timing closure, modify the parameterization of the IOPLL IP core to a lower clock frequency.

Figure 2.   Timing Closure
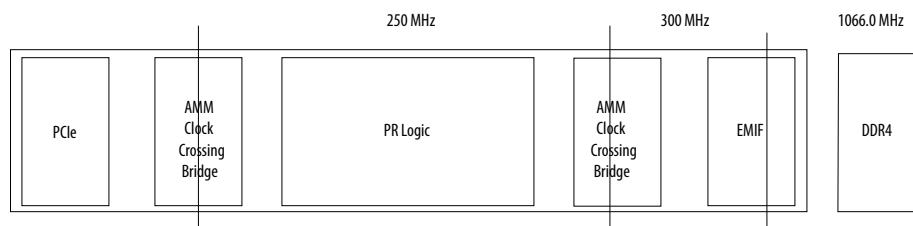
1.1.2 Memory Address Mapping

The PCIe IP core connects to the PR logic through an Avalon-MM interface. The following table lists the memory address mapping from the PCIe IP core, to the PR logic:

Table 1.   PCIe Memory Address Map

Address Map	Base	End
PR Logic	0x0000_0000	0x0000_3fff
PR_IP	0x0000_cfb0	0x0000_cfb7

The 2 GB DDR4 memory space is available to PR logic through an Avalon-MM interface. The following table lists the memory address mapping from the External Memory Interfaces IP core, to the PR logic:

**Table 2.   DDR4 External Memory Interfaces (EMIF) Memory Address Map**

Address Map	Base	End
DDR	0x0000_0000	0x7fff_ffff

1.1.3 Floorplanning

The floorplan constraints in your partial reconfiguration design physically partitions the device. This partitioning ensures that the resources available to the PR region are the same for any persona that you implement.

To maximize the fabric amount available for the PR region, the reference design constrains the static region to a very small area.

Figure 3.   **Reference Design Floorplan**

LogicLock Plus Region (Fabric)
Available to the PR Region
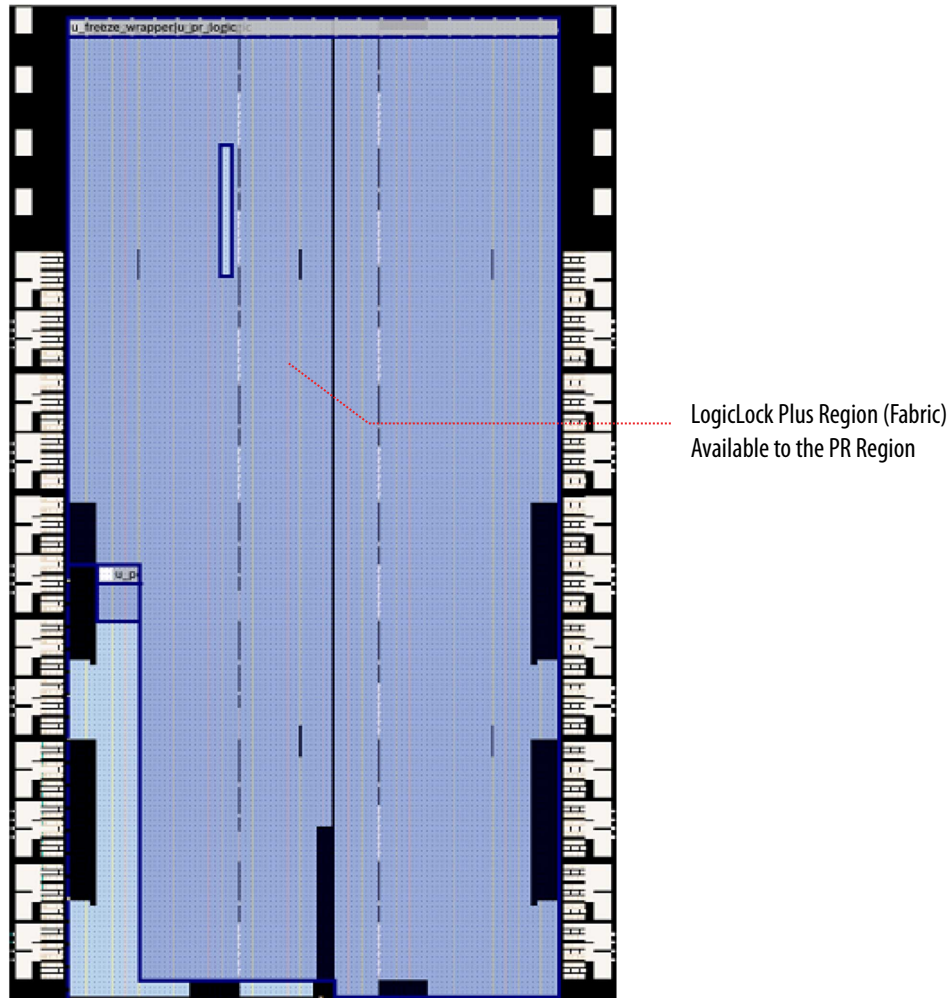
For more information on floorplanning, refer to *Floorplan the Partial Reconfiguration Design* section in Volume 1 of the *Quartus Prime Pro Edition Handbook*.

Related Links

Floorplan the Partial Reconfiguration Design

1.2 Getting Started

This section describes the requirements and the procedure to run the reference design.

1.2.1 Hardware and Software Requirements

The reference design requires and uses the following hardware and software tools:

- Arria 10 GX FPGA development board
- Linux Operating System with a PCI Express slot to plug-in the Arria 10 FPGA development board
- Linux driver for the Arria 10 PR over PCIe reference design
- Quartus Prime Pro Edition software v. 16.1

1.2.2 Installing the Arria 10 GX FPGA Development Kit

For complete instructions on installing and powering the Arria 10 GX FPGA development board in your Linux system, refer to *Arria 10 FPGA Development Kit User Guide*.

Note: Before powering the board, set the switch 4 (FACTORY) of the DIP switch bank (SW6) to **ON**. Setting this switch to **ON** loads the factory image area of the flash memory at boot time. Program the reference design into this factory image area. For complete instructions on flashing the reference design onto the board, refer to *Bringing Up the Reference Design*.

Related Links

- Arria 10 FPGA Development Kit User Guide
- Bringing Up the Reference Design on page 7

1.2.3 Installing the Linux Driver

The reference design includes the complete source code for the Linux driver for the Arria 10 PR over PCIe reference design.

To compile and load the Linux driver:

1. Navigate to the source/driver directory in the reference design.
2. Run the install.sh script.

1.2.4 Bringing Up the Reference Design

The reference design is available in the following location:

https://github.com/01org/fpga-partial-reconfig

To access the reference design, navigate to the ref_designs sub-folder. Copy the a10_pcie_devkit_cvp folder to the home directory in your Linux system.

To bring up the reference design on the board:

1. Plug-in the Arria 10 GX FPGA development board to an available PCIe slot in your host machine.
2. Connect the host machine's ATX auxiliary power connector to the 12 V ATX input J4 of the development board.
3. Power-up the host machine.
4. Verify the micro-USB cable connection to the FPGA development board. Ensure that no other applications that use the JTAG chain are running.

セグ

5. Navigate to the `a10_pcie_devkit_cvp/software/installation` folder in your system.

6. To overwrite the existing factory image on the board with the reference design, execute the `run_program.sh` script.

   Running this script configures the device with the contents of the `flash.pof` file. This parallel object file acts as the base image for the reference design.

7. Power-cycle the host machine.

1.3 Reference Design Components

The reference design contains the following design components.

1.3.1 PCIe Subsystem

In the `pcie_subsystem` logic, the reference design implements the Arria 10 Hard IP for PCI Express with driver compatible configuration.

The Arria 10 Hard IP for PCI Express IP core is a Gen3x8, with a 256-bit interface and running at 250 MHz.

The following table provides information on the configuration fields of the PCI Express IP core that the reference design uses that are different from the default settings:

Table 3.   PCI Express IP Core Configuration

Setting	Parameter	Value
System Settings	Application interface type	Avalon-MM with DMA
	Hard IP mode	Gen3:x8, Interface: 256-bit, 250 MHz
	Port type	Native endpoint
	RX buffer credit allocation for received requests vs completions	Low
Avalon-MM Settings	Enable control register access (CRA) Avalon-MM slave port	Disable
Base Address Registers - BAR2	Type	Disabled
Base Address Registers - BAR4	Type	32-bit non-prefetchable memory
Device Identification Registers	Vendor ID	0x00001172
	Device ID	0x00005052
	Revision ID	0x00000001
	Class code	0x00ea0001
	Subsystem Vendor ID	0x00001172
	Subsystem Device ID	0x00000001
PCI Express/PCI Capabilities - Device	Maximum payload size	256 Bytes
Configuration, Debug, and Extension Options	Enable Arria 10 GX FPGA Development Kit Connection	Enable
PHY Characteristics	Requested equalization far-end TX preset	Preset 9
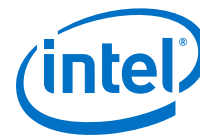
Note: Instantiate the PCI Express IP core as part of a Qsys Pro system.

1.3.2 Partial Reconfiguration IP Core

The reference design configures the Partial Reconfiguration IP core to operate as an internal host. The design connects the PR IP core to the PCI Express IP core, via an instance of the Avalon-MM interface. The PR IP core has a clock-to-data ratio of 1. Therefore, the PR IP core is not capable of handling encrypted or compressed PR data.

The following table lists the configuration fields of the Arria 10 Hard IP for PCI Express IP core that are different from the preset settings:

Parameters	Value
Enable JTAG debug mode	Disable
Enable Avalon-MM slave interface	Enable
Input data width	32

1.3.3 External Memory Controller Subsystem

The `mem_ctlr` logic includes the Arria 10 External Memory Interfaces IP core. This IP core interfaces to the DDR4 external memory, with a 64-bit interface that runs at 1066.0 MHz. Also, the IP core provides 2 GB of DDR4 SDRAM memory space. The EMIF Avalon-MM slave runs at 300 MHz clock.

The following table lists the configuration fields of the Arria 10 External Memory Interfaces IP core that are different from the Arria 10 GX FPGA Development Kit with DDR4 HILO preset settings:

Table 4.   Arria 10 DDR4 External Memory Interfaces IP Configuration

Setting	Parameter	Value
Memory - Topology	**DQ width**	64
	DQ pins per DQS group	8
	Number of DQS groups	8
	Alert# pin placement	I/O Lane with Address/Command Pins
	Address/Command I/O lane of ALERT#	3
	Pin index of ALERT#	0

1.3.4 Partial Reconfiguration Logic

The reference design provides the following personas:

Table 5.   **Reference Design Personas**

Persona	Description
DDR4 access	Communicates with the Arria 10 External Memory Interfaces IP core, instantiated in the static region. This communication ensures that the persona accesses the DDR4 memory correctly.
Register file	Provides access to a set of registers that read and write over PCIe.
Basic arithmetic	Basic adder persona that functions as a hardware accelerator.

Each persona has an 8-bit `persona_id` field in the `pr_data` register to indicate a unique identification number.

Additionally, the reference design provides a template to implement your custom persona. This template persona allows you modify the RTL, create a wrapper to interface with the register file, compile, and run your design.

1.4 Testing the Reference Design

The reference design provides the following software modules for programming the FPGA board:

- `program_over_jtag`
- `program_over_pcie`
- `example_host`

These modules are examples for host side applications that communicate with the FPGA, over PCIe.

1.4.1 `program_over_jtag`

Use `program_over_jtag` module to program the entire device (full-chip programming) without any requirement for reboot, and for overwriting the flash memory using JTAG.

`program_over_jtag` performs the following functions:

- Uses the Quartus Prime Programmer to program the device.
- Accepts an SRAM Object File (`.sof`) and configures the target device over a JTAG interface.
- Saves the PCIe control registers before programming the device with a new `.sof` file.
- Restores the PCIe control registers after completing the programming.

Table 6. `program_over_jtag` **Command-Line Options**

Option	Description
`-f [<filename>]`	Specifies the `.sof` file name.
`-c [<cable number>]`	Specifies the programmer cable. Default value is 1.
`-d [<device index>]`	Specifies the device index. Default value is 1.
`-h`	Provides help for `program_over_jtag` application.

1.4.2 `program_over_pcie`

The `program_over_pcie` module accepts a Raw Binary File (`.rbf`) for a given persona. The application updates the PR region with the specified `.rbf` file, using the Avalon-MM interface over PCIe. On PR completion, the application reports the PR status, and releases the device.

Table 7. `program_over_pcie` **Command-Line Options**

Option	Description
`-f [<filename>]`	Specifies the `.rbf` file name.
`-h`	Provides help for `program_over_pcie` application.

1.4.3 `example_host`

The `example_host` module demonstrates the FPGA device access. This application interacts with each persona, verifying the contents and functionality of the personas.

1.4.3.1 Compiling the Example Applications

The reference design software applications are available in the `source/util` directory. Each application has a respective sub-directory structure, with a corresponding `Makefile`.

To build the example applications:

1. To compile the `program_over_pcie` module, type the following from the Linux shell:

```
cd source/util/program_over_pcie
make
```

2. To compile the `program_over_jtag` module, type the following from the Linux shell:

```
cd source/util/program_over_jtag
make
```

3. To compile the `example_host` module, type the following from the Linux shell:

```
cd source/util/example_host
make
```

These commands generate the executables in the respective sub-directories.

1.4.3.2 Programming the Design Using Example Applications

The following steps describe programming your design using the example applications:

1. To program the DDR access persona's `.sof` file after design compilation, type the following from the Linux shell:

   ```
   program_over_jtag -f
   a10_pcie_devkit_cvp_pr_logic.sof
   ```

2. To verify the functionality of the design, type the following from the Linux shell:

   ```
   source/util/example_host/sample_host
   ```

3. To program the register file persona's `.rbf` file after design compilation, type the following from the Linux shell:

   ```
   program_over_pcie -f
   a10_pcie_devkit_cvp_register_file.pr_partition.rbf
   ```

4. To verify the functionality of the design, type the following from the Linux shell:

   ```
   source/util/example_host/sample_host
   ```

5. To program the basic arithmetic persona's `.rbf` file after design compilation, type the following from the Linux shell:

   ```
   program_over_pcie -f
   a10_pcie_devkit_cvp_basic_arithmetic.pr_partition.rbf
   ```

6. To verify the functionality of the design, type the following from the Linux shell:

   ```
   source/util/example_host/sample_host
   ```

1.5 Extending the Reference Design with Custom Persona

This reference design provides an example template to create your own personas for PR over PCIe. To extend the reference design with your custom persona:

1. Navigate to the `a10_pcie_devkit_cvp` folder:

   ```
   cd ~/a10_pcie_devkit_cvp
   ```

2. Create a copy of the `pr_logic_impl_template.qsf.template` implementation revision file, and the `pr_logic_synth_template.qsf.template` synthesis revision file:

   ```
   cp pr_logic_impl_template.qsf.template <persona_impl_revision_name>.qsf
   cp pr_logic_synth_template.qsf.template <persona_synth_revision_name>.qsf
   ```

3. Create a folder and copy your persona-specific RTL to this folder:

   ```
   mkdir <persona_name>
   cp <custom_persona>.sv <persona_name>/
   ```

4. Your custom top-level entity must match the ports for the `custom_persona` module, defined in the `source/templates/pr_logic_template.sv` file. The following example shows interfacing your design with the Avalon-MM interface, controlled over PCIe register file:

   ```
   module custom_persona #(
   parameter REG_FILE_IO_SIZE = 8
   )(
   //clock
   input wire clk,

   //active low reset, defined by hardware
   input wire rst_n,

   //Persona identification register, used by host in host program
   output wire [31:0] persona_id,

   //Host control register, used for control signals.
   input wire [31:0] host_cntrl_register,

   // 8 registers for host -> PR logic communication
   input wire [31:0] host_pr [0:REG_FILE_IO_SIZE-1],

   // 8 Registers for PR logic -> host communication
   output wire [31:0] pr_host [0:REG_FILE_IO_SIZE-1]
   );
   ```

   Utilize any of the parallel I/O port (PIO) register files for customization. The `host_pr` register sends the data from the persona to the host machine. The `pr_host` register sends the data from the host machine to the persona.

5. In your top-level entity file, specify the persona ID as any 32-bit value:

   ```
   assign persona_id  = 32'h0000_aeed;
   ```

   Note: The example template uses only 8 bits, but you can specify any value, up to 32 bits.

6. Set the unused output ports of the `pr_host` register to 0:

   ```
   generate
   genvar i;
   //Tying unused output ports to zero.
   for (i = 2; i < REG_FILE_IO_SIZE; i = i + 1) begin
   ```

```
    assign pr_host [i] = 32'b0;
end
endgenerate
```

7. Modify your `persona_synth_revision_name.qsf` to include the following assignments:

```
set_global_assignment -name TOP_LEVEL_ENTITY <custom_persona>
set_global_assignment -name SYSTEMVERILOG_FILE <persona_name>/
<custom_persona>.sv
```

8. Update the partial reconfiguration flow script to define your new PR implementation.

9. Update the `a10_pcie_devkit_cvp.qpf` project file to include your synthesis and implementation revisions:

```
PROJECT_REVISION = "<persona_synth_revision_name>"
PROJECT_REVISION = "<persona_impl_revision_name>"
```

10. Compile the revision.

   For complete information on adding a custom persona to a PR design, refer to *Adding a New Persona to the Design* section in the *Partially Reconfiguring a Design on Arria 10 SoC Development Board* application note.

Related Links

Adding a New Persona to the Design

1.6 Document Revision History

This document has the following revision history.

Table 8.   Document Revision History

Date	Version	Changes
2016.10.31	16.1.0	Initial Release

A Reference Design Files

Table 9. Reference Design Files List

Type	File/Folder	Description
IP files	source └ static_region └ iopll.ip	Contains all the IOPLL IP core settings.

	source
	└ static_region
	  └ ip
	    └ ddr4_ctlr
		Contains the Arria 10 External Memory Interfaces IP core the corresponding clock-domain-crossing bridge settings.

	source
	└ static_region
	  └ ip
	    └ pcie_subsystem
		Contains the Arria 10 Hard IP for PCI Express IP core, the Partial Reconfiguration IP core, and the corresponding clock-domain-crossing bridge settings.

Qsys Pro System Files	source
	└ static_region
	  ├ ddr4_ctlr.qsys
	  └ pcie_subsystem.qsys
		Contains the PCIe and memory controller subsystems, the IP components, and the respective Qsys Pro interconnect.

SystemVerilog design files	source └ ddr4_access_persona	DDR4 access persona that calibrates, and generates data and address. These files perform the following functions: • Write the generated data to the generated address • Read back the data to verify successful save operation • Calibrate the DDR4 access persona • Return the read or write error value to the host, via the PCIe interface • Provide the host with access to a series of registers that function as control interface

Type	File/Folder	Description
	📁 source 　　📁 register_file_persona	Register file persona that provides the register block, and 256x32-bit set of readable-writable registers.
	📁 source 　　📁 basic_arithmetic_persona	Basic arithmetic persona that performs the adder operation. The host side application controls the inputs to the function, and reads the result in a separate register.
	📁 source 　　📁 templates	Template persona that you modify to write your custom top-level entity.
	📁 template_example	Example personas that use the template for persona configuration. These examples demonstrate integrating a custom persona RTL into the reference design.
Resource files	📁 source 　　📁 common 　　　　📁 reg_file	Contains the common resources that you use with the template.
Example utilities	📁 software 　　📁 util 　　　　📁 example_host	Contains the example application to access the programmed FPGA and run applications.
	📁 software 　　📁 util 　　　　📁 program_over_jtag	Contains the example application that accepts a `.sof` file and configures the target FPGA over a JTAG interface.
	📁 software 　　📁 util 　　　　📁 program_over_pcie	Contains the example application that accepts `.rbf` files and reconfigures the subsequent personas in the PR region over PCIe interface.
	📁 software 　　📁 util 　　　　📁 common	Contains a library of functions that you use to read or write to the device over PCIe.
Driver files	📁 software 　　📁 driver	Contains the source code of the Linux driver for Arria 10 PR over PCIe reference design.
SignalTap® II File	📄 a10_pcie_devkit_cvp.stp	Contains the signals the reference design uses in the SignalTap II on-chip logic analyzer.

continued...

Type	File/Folder	Description
Synopsys Design Constraints Files	a10_pcie_devkit_cvp.sdc	Synthesis constraints for the design.
	auxiliary.sdc	Provides exceptions.
	jtag.sdc	Auto-generated constraints from `pcie_subsystem_alt_pr.ip` file.
Quartus Prime Project File	a10_pcie_devkit_cvp.qpf	Contains all the revisions.
Quartus Prime Settings Files	a10_pcie_devkit_cvp.qsf	Base revision settings file for DDR4 access persona.
	ddr4_access.qsf	Synthesis revision settings file for DDR4 access persona.
	register_file.qsf	Synthesis revision settings file for register file persona.
	basic_arithmetic.qsf	Synthesis revision settings file for basic arithmetic persona.
	pr_example_template_synth.qsf	Synthesis revision settings file for example persona.
	a10_pcie_devkit_cvp_ddr4_access.qsf	Implementation revision settings file for DDR4 access persona.
	a10_pcie_devkit_cvp_register_file.qsf	Implementation revision settings file for register file persona.
	a10_pcie_devkit_cvp_basic_arithmetic.qsf	Implementation revision settings file for basic arithmetic persona.
	pr_example_template_impl.qsf	Implementation revision settings file for example persona.
PR Setup Script	setup.tcl	Specifies the PR flow compilation.