

# How-To Configure and Implement Altera\_PLL Cascading Feature

**Introduction:** This document describes the PLL cascading feature support using the Altera\_PLL megafunction and implementation. Compared to the conventional PLL Cascading; Altera\_PLL Cascading feature uses a dedicated cascading clock path between a pair of fPLLs to achieve better jitter performance and save global clock resources.

## 1. Set Up

The Quartus® II software version 12.1 and the Altera\_PLL megafunction version 12.1 are used throughout this How-To document to describe the required connections and usage for the cascading feature.

## 2. Upstream and Downstream PLL configurations

The Upstream PLL configuration uses the following ports:

- refclk input
  - clock output(s)
  - cascade out bus (bundles all the output clocks)
- Below figure 2-1 shows the typical upstream configuration.

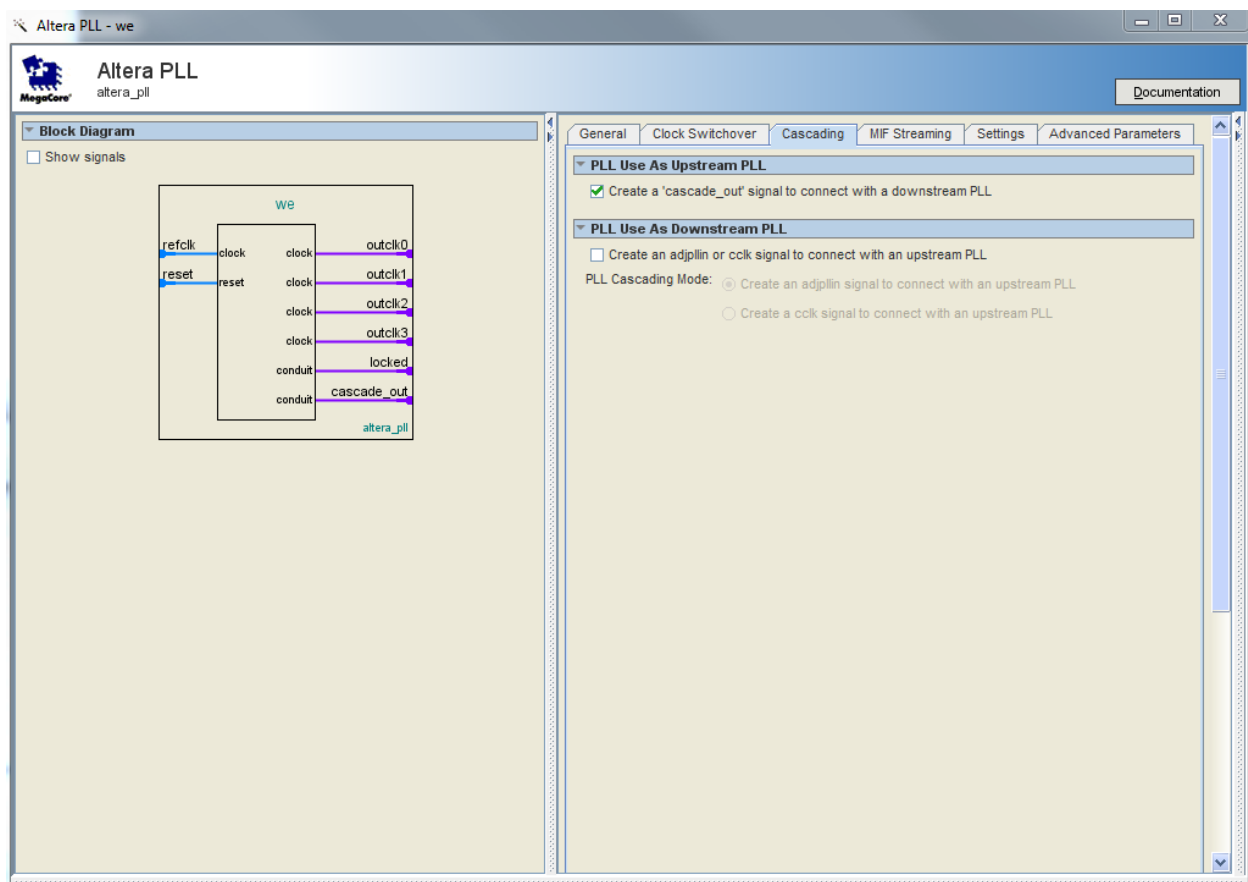
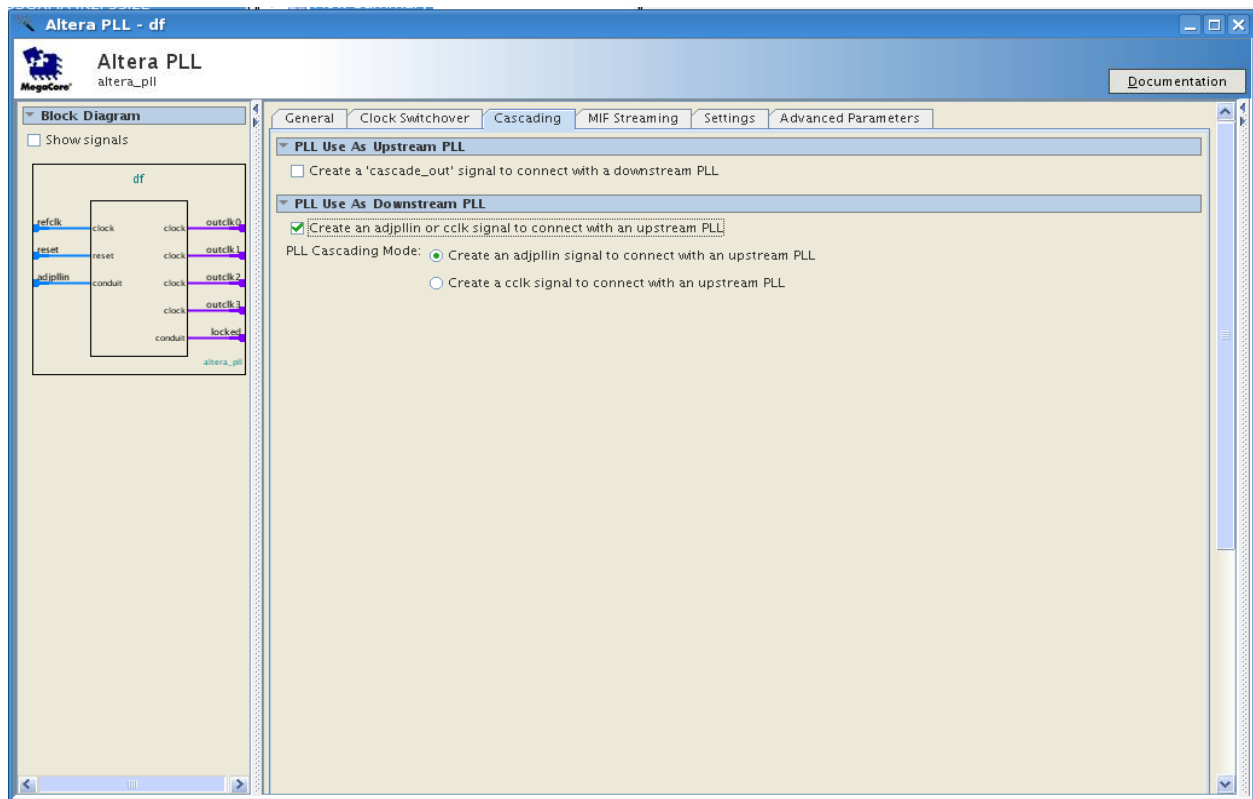


Figure 2-1: Altera\_PLL megafunction Upstream Configuration

The Downstream PLL configuration uses the following ports:

- refclk
- output clocks
- adjpllcn or cclk (to be used as a cascade input reference clock)

Below figure 2-2 shows the typical downstream configuration.



**Figure 2-2:** Altera\_PLL megafunction Downstream Configuration

### 3. Altera Cascade PLL Implementation:

- The Downstream refclk input should remain unconnected and the internal mux will select the adjpll or cclk as the cascade input clock depending on the selection you make in the megafunction.
- To ensure the Downstream fPLL is reset and synchronized properly; Altera recommends using the Upstream “locked” output port to connect to the downstream fPLL reset port as shown in figure 3-1.
- The Upstream cascade\_out is a bus that bundles all of the output clocks. The Downstream PLL needs to be connected to one of these output clocks at the adjpll or cclk input port. Below figure 3-1 shows the Cascade fPLL design implementation.

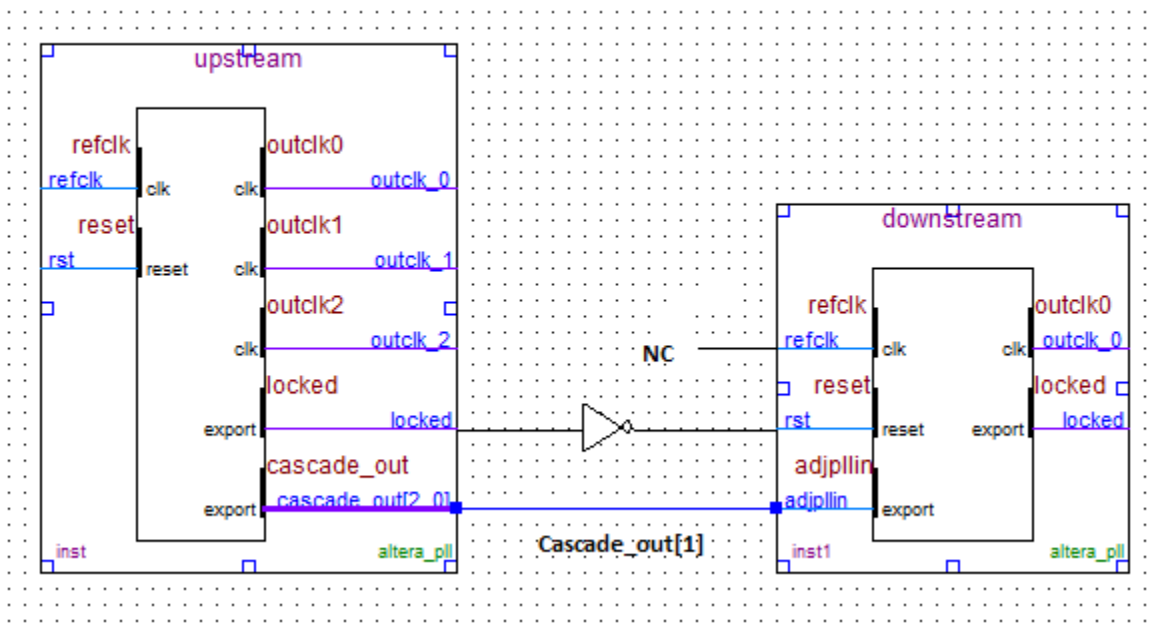


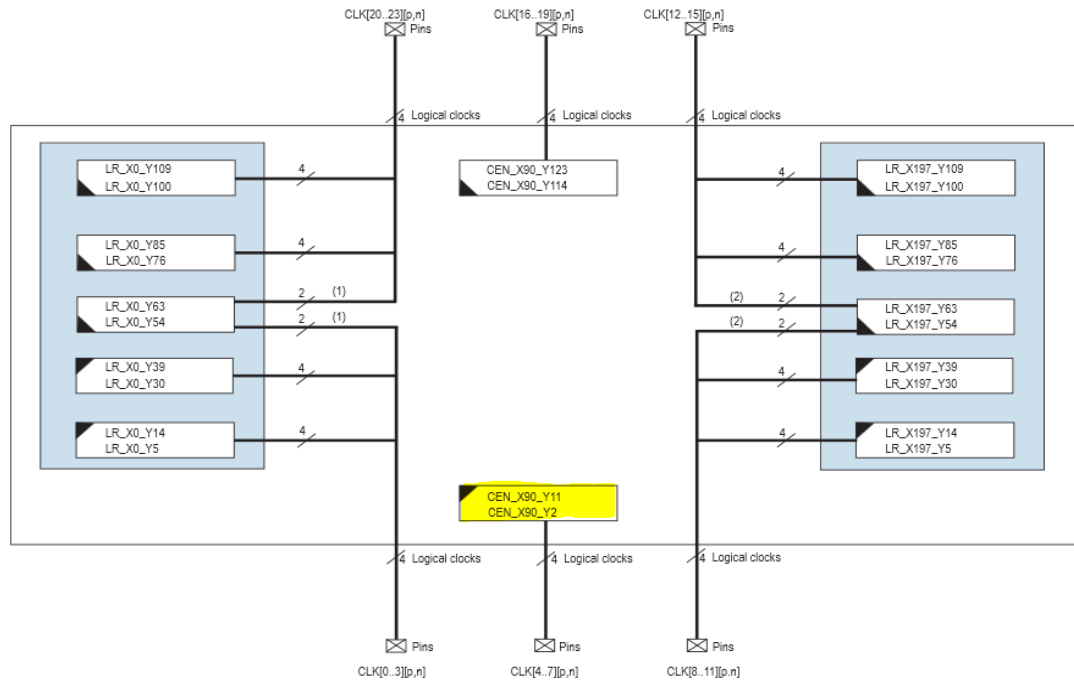
Figure 3-1: Cascade fPLL Implementation

- PLL location assignments are required in the Assignment Editor when using the Cascade PLL feature in Quartus II software version 12.1. Figure 3-2 shows a possible f PLL pair for Cascade fPLL implementation (highlighted in yellow) for Stratix® V GX B5 and B6 devices. Below are fPLL location assignment examples:

```

- set_location assignment FRACTIONALPLL_X90_Y2_N0 -to
"pll_u1|altera_pll:altera_pll_i|altera_stratixv_pll:stratixv_base:fp
ll_0|fp11"
- set_location assignment FRACTIONALPLL_X90_Y11_N0 -to
"pll_u2|altera_pll:altera_pll_i|altera_stratixv_pll:stratixv_base:fp
ll_0|fp11"

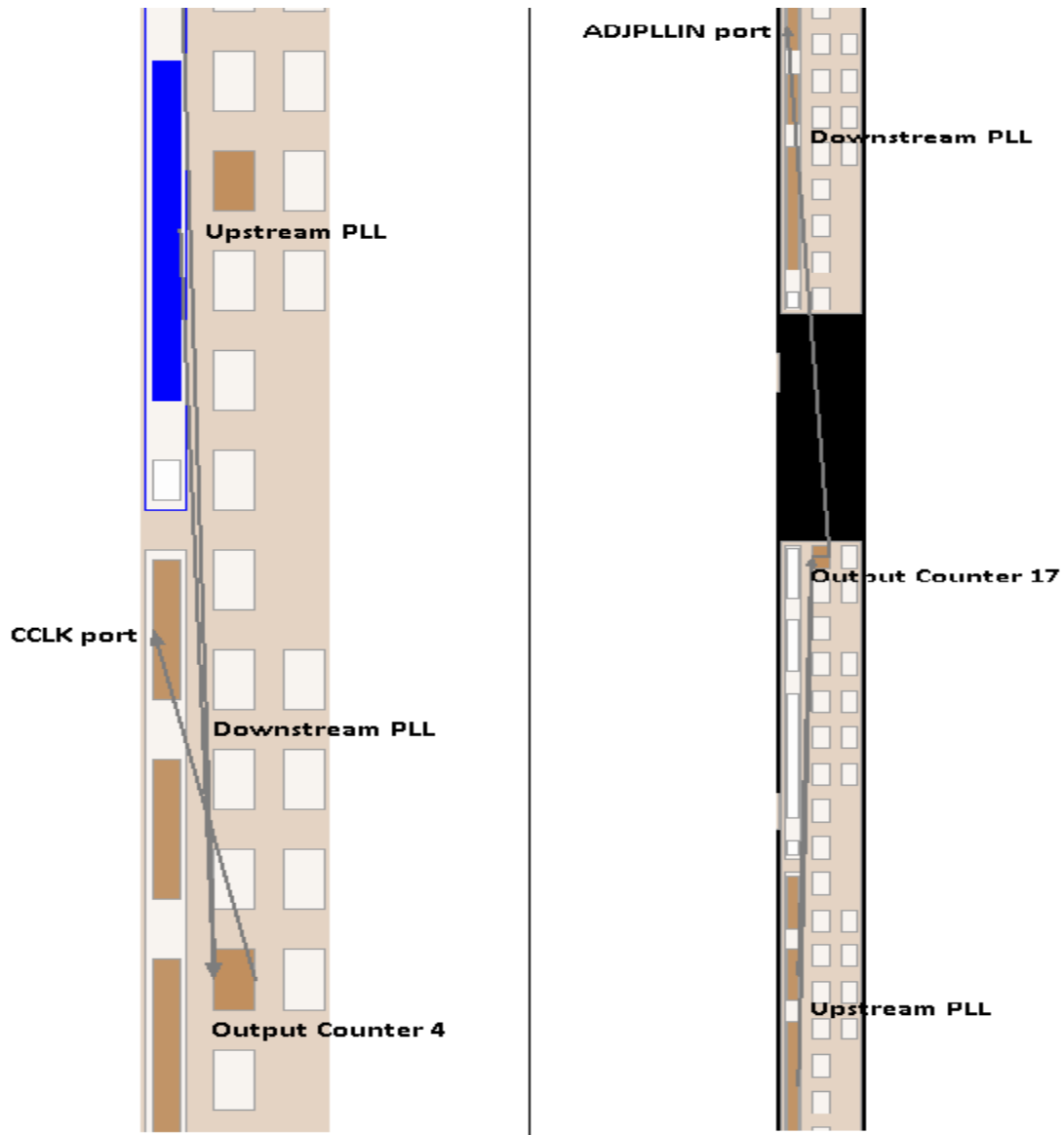
```



**Figure 3- 2** Stratix V GX B5 & B6 fPLL location

- TimeQuest Requirements :** You will encounter a warning message in Quartus II software version 12.1 when cascading fPLLs during the design compilation when using the global clock constraints command in the project.sdc file shows below:
  - Global Clock Constraints Command:
    - Derive\_pll\_clocks - create\_base\_clocks
  - Warning Message:
    - The master clock for this clock assignment could not be derived. Clock:u\_clks\_resets|u\_pll\_c|ddf2\_pll\_c\_inst|altera\_pll\_i|stratixv\_pll|fp11\_0|fp11|vcoph[1] was not created"
  - Work around:
    - You must create constraints for all the fPLL output clocks and below is one of the fPLL clock output constraint example:
      - create\_clock -name clock\_a -period 4.082 [get\_pins {pll\_a|altera\_pll\_i|stratixv\_pll|counter[0].output\_counter|divclk}]

- Adjpll原因 vs cclk cascading implementation:** Adjpll原因 is used for inter-cascading between fPLLs that are not in the same fPLL pair and cclk is used for intra-cascading within the same pair of fPLLs. Routing will be different as adjpll原因 crosses fPLL boundaries. Figure 3 -3 shows the physical adjpll原因 and cclk cascading implementation from the Chip Planner. The C-Counter Clock source from output counter 4 or output counter 13 is used for cclk connection. When using the adjpll原因 input, it will be connected to the nearest output counter from the adjacent fPLL.



**Figure 3- 3** adjpll原因 vs cclk cascading implementation

#### **4. Conclusion**

This document demonstrated how to use the Altera\_PLL megafunction to configure the PLL as an upstream or downstream PLL using a dedicated cascading PLL clock path, as well as highlighted the assignments and .sdc requirements when using this feature.

## 2. Revision History

<b>Revision</b>	<b>Changes Made</b>	<b>Date</b>
V1.0	Initial release	March 2013

© 2013 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.