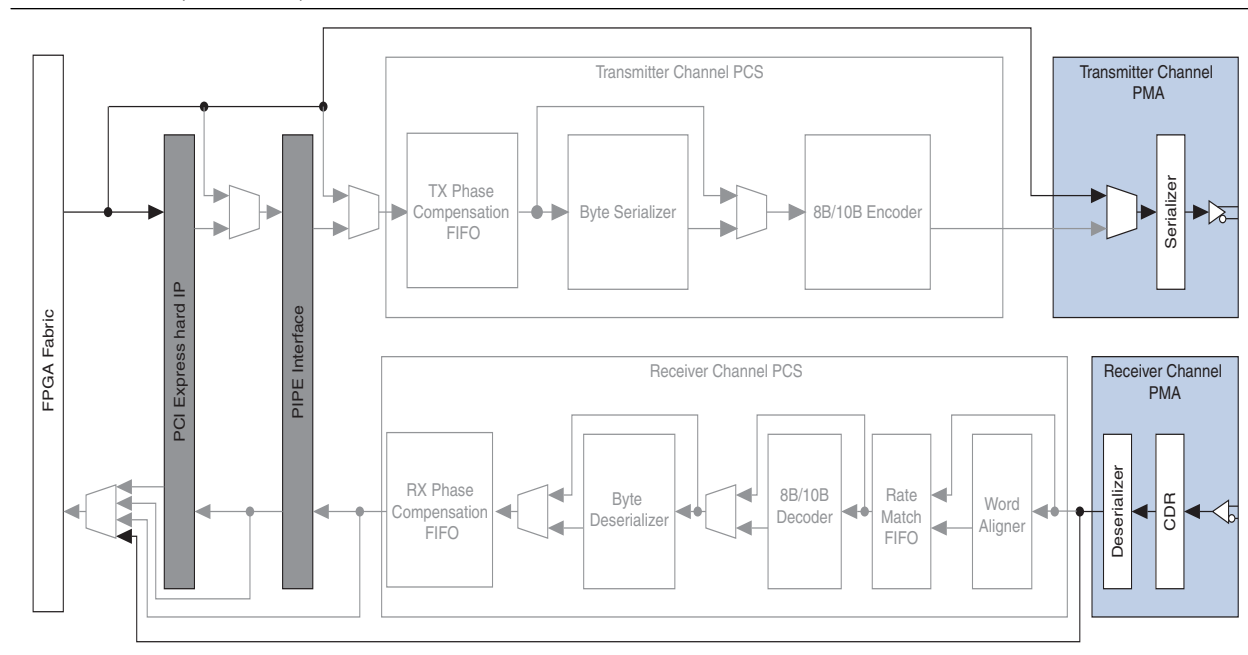


This application note describes the method to achieve timing closure for designs that use transceivers in Basic (PMA Direct) mode in Altera’s Stratix® IV GX or Stratix IV GT FPGAs.

Introduction

Transceiver channels configured in Basic (PMA Direct) functional mode only use the physical medium attachment (PMA) blocks of the transceiver channels. The physical coding sub-layer (PCS) blocks of all channels are bypassed, as shown in [Figure 1](#).

Figure 1. Basic (PMA Direct) Functional Mode



The interface between the transceivers and the FPGA core introduces significant delay on the clocks that are forwarded from the transceivers to the user logic in the FPGA core. A phase-compensation FIFO buffer in the transceiver PCS block compensates for the phase difference (due to delays) between the transceiver clocks used internally and the transceiver clocks routed to and from the FPGA core. When transceivers are used in Basic (PMA Direct) functional mode, the phase-compensation FIFO buffer is bypassed because the entire PCS block is bypassed. The result is the timing requirement is not easily met.



For more information about Basic (PMA Direct) functional mode, refer to the [Stratix IV Transceiver Architecture](#) chapter in volume 2 of the *Stratix IV Device Handbook*.

Achieving Timing Closure

The following factors determine if your design can achieve timing closure with the transceiver channels configured in Basic (PMA Direct) functional mode:

- Device density
- Speed grade of the device
- Channel width and data rate of the transceiver channel
- Design attributes such as resource utilization and placement

If any of these design factors change, the design must be re-verified for timing violations.

Check for Timing Violations

Use the TimeQuest Timing Analyzer to generate the timing reports, as shown in [Figure 2](#). **Report All Summaries** provides a high-level summary of all the timing violations and slack, while **Report Timing** provides the detailed timing information required to analyze the top failing paths.



This application note is aimed at resolving transceiver—FPGA core interface timing violations only. You must follow best practices to resolve any timing violations that exist in the FPGA core.

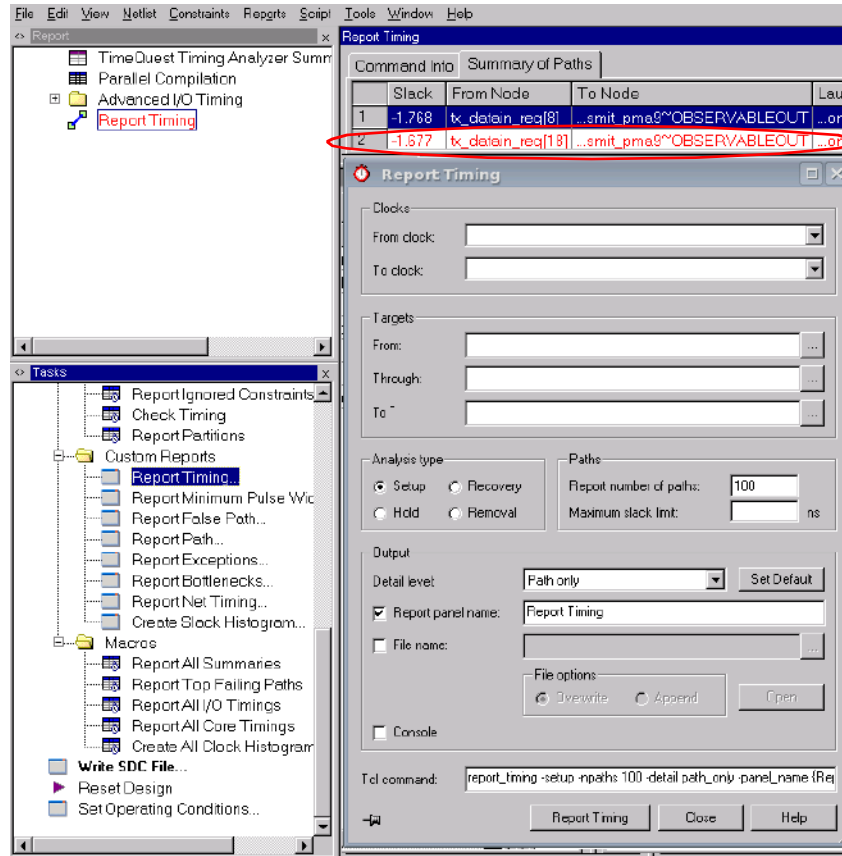
To identify whether the failing paths are associated with the transceiver—FPGA core interface, look for the following nodes in the timing report:

- For the transmit side, the destination node name contains the word "OBSERVABLEOUT". [Figure 2](#) shows the failing path from tx_datain_reg in the FPGA core to the transmit_pma_~OBSERVABLEOUT node in the transceiver
- For the receive side, the source node name contains the word "recoverdataout"



For more information about the Timing Report Generation, refer to the [Quartus II TimeQuest Timing Analyzer](#) chapter in volume 3 of the [Quartus II Handbook](#).

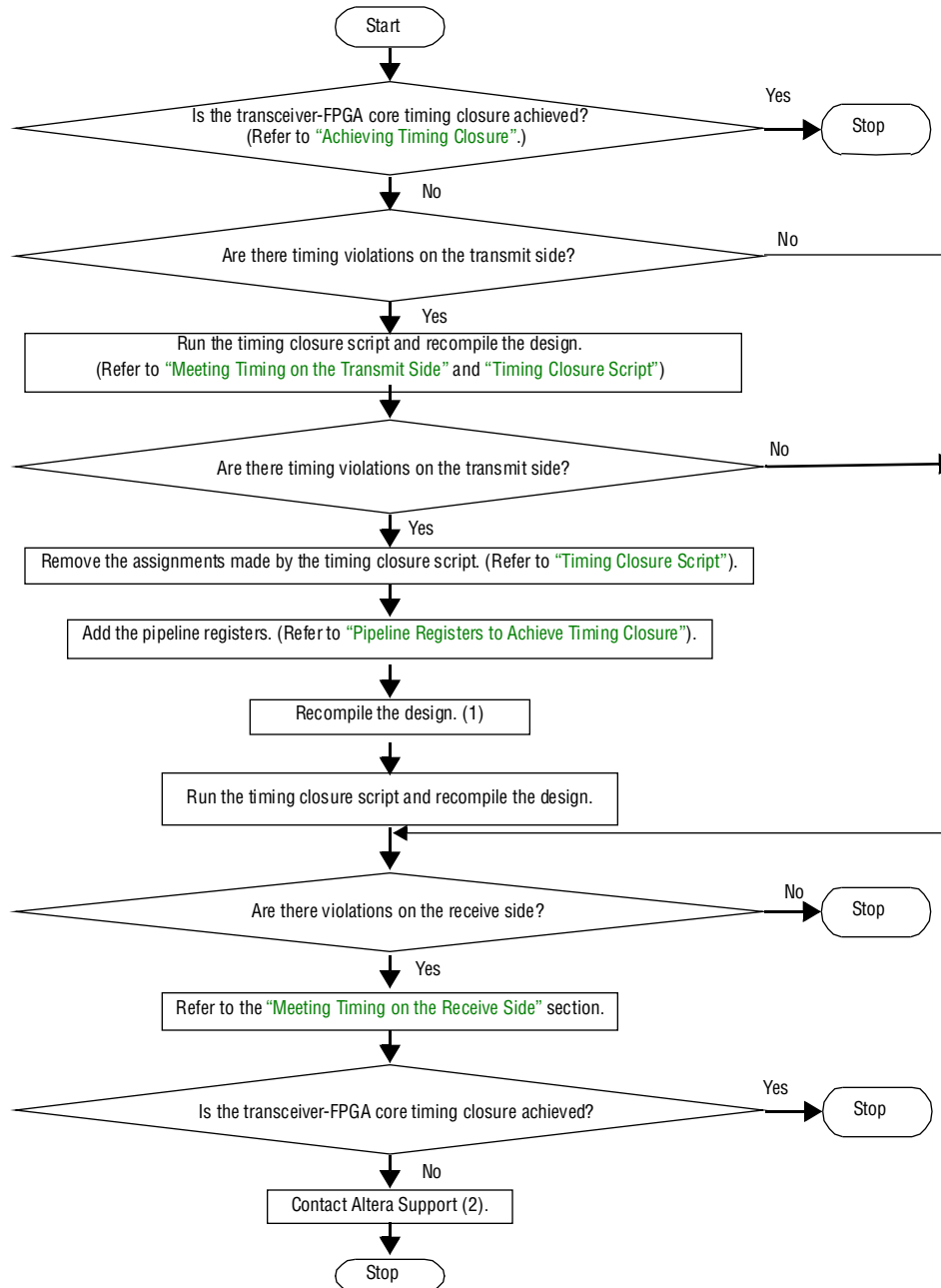
Figure 2. Generating a Timing Report Using the TimeQuest Timing Analyzer



Steps to Achieve Timing Closure

Figure 3 shows the high-level steps that you must take to achieve timing closure.

Figure 3. Steps to Achieve Timing Closure



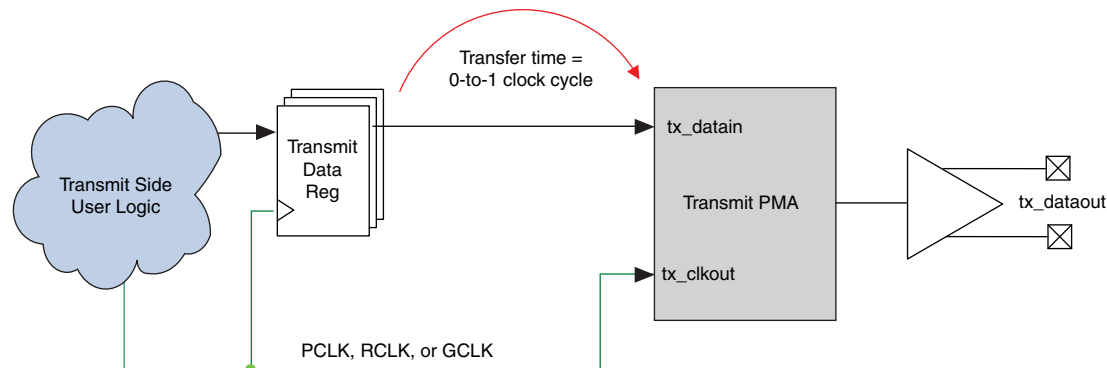
Notes to Figure 3:

- (1) A recompile is necessary before running the script to allow the script to add the appropriate placement and clock assignments.
- (2) For more information about Altera support, refer to [mySupport](#).

Meeting Timing on the Transmit Side

Figure 4 shows a sample design that has a single channel configured in Basic (PMA Direct) functional mode that cannot achieve timing closure. The data registers in Transmit Data Reg are the transmit-side user logic that drives the data input port of the transmit PMA. Placement of Transmit Data Reg is critical to achieve timing closure. The data registers driving a channel must be placed as close as possible to the tx_datain port of the transmit PMA.

Figure 4. Single Channel Configured in Basic (PMA Direct) Functional Mode—Timing Closure not Achieved



Routing a clock onto a global net incurs additional insertion delays that can reduce the timing margin available for a design. Clock networks that span a large area of the device, such as the global clock (GCLK) network or regional clock (RCLK) network, can have a larger clock insertion delay than a local clock network, such as the periphery clock (PCLK) network.

You can use the GCLK and RCLK only up to a certain data rate to clock the transmit-side user logic for transceivers configured in Basic (PMA Direct) functional mode. For higher data rates, you must use the PCLK to clock the transmit-side user logic. Using a local clock network improves the timing margin by reducing the by reducing the clock insertion delay. If it is not feasible to clock the entire transmit-side user logic using the PCLK, clock only Transmit Data Reg using the PCLK.

If you still cannot meet the timing requirement or there is not enough timing margin, add a stage of negative-edge-triggered registers in the FPGA core between the registers that feed the transmit data to the transceiver and the transceiver's data input port to meet the timing requirement. The negative-edge-triggered registers provide an additional setup timing margin equal to half the time period of the core clock at the expense of the hold time margin, thus meeting the setup time requirements of the transmit side of the transceiver at higher data rates.



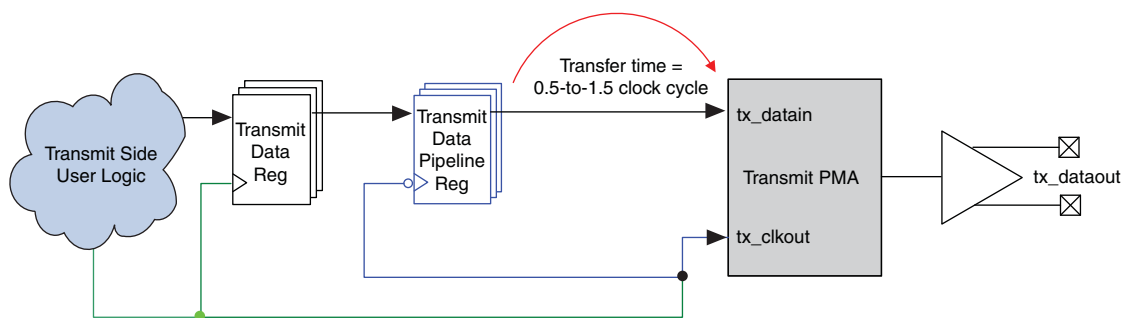
In this application note, the negative-edge-triggered registers used to meet the timing requirement are referred to as “pipeline registers.”

Pipeline Registers to Achieve Timing Closure

Figure 4 shows a sample design that uses transceivers in Basic (PMA Direct) functional mode. This sample does not use pipeline registers to meet the timing requirement.

Figure 5 shows using pipeline registers to meet the timing requirement. The pipeline registers (Transmit Data Pipeline Reg) are introduced between the transmit data registers of the transmit-side user logic in the core (Transmit Data Reg) and the transmit data input port of the transceiver (tx_datain). The result is an additional timing margin of half a clock period of the transmit clock forwarded to core (tx_clkout). With the introduction of the pipeline registers, the transfer time of the transmit data changes from a 0-to-1 clock cycle to a 0.5-to-1.5 clock cycle of tx_clkout.

Figure 5. Negative-Edge-Triggered Pipeline Registers to Meet the Timing Requirement



Pipeline registers add a latency of one clock cycle to the transmit-side data path.

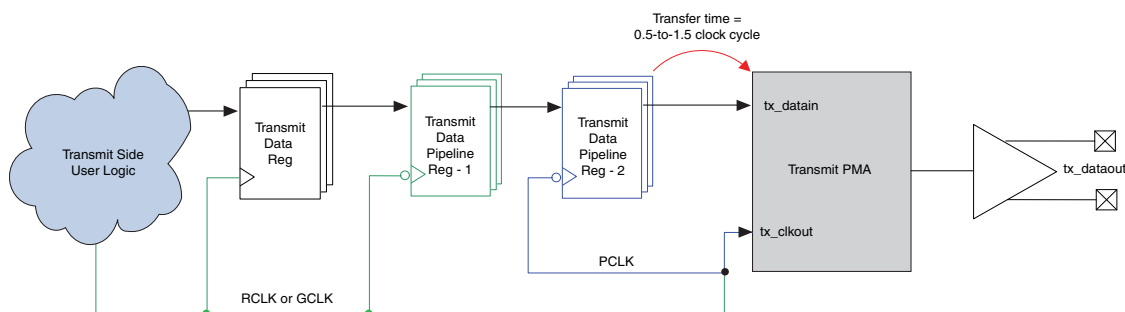
Use the scheme in Figure 5 only if one of the following two conditions applies:


- The PCLK does not clock the transmit-side user logic (including Transmit Data Reg) nor Transmit Data Pipeline Reg
- The PCLK clocks both transmit-side user logic (including Transmit Data Reg) and Transmit Data Pipeline Reg

You may have to use the PCLK at higher data rates to meet the timing requirement. The implementation changes if you only use the PCLK to clock the pipeline registers and not the entire transmit-side user logic (including Transmit Data Reg).


Figure 6 shows the modified implementation scheme.

Figure 6. Negative-Edge Triggered Pipeline Registers to Meet the Timing Requirement at Higher Data Rates



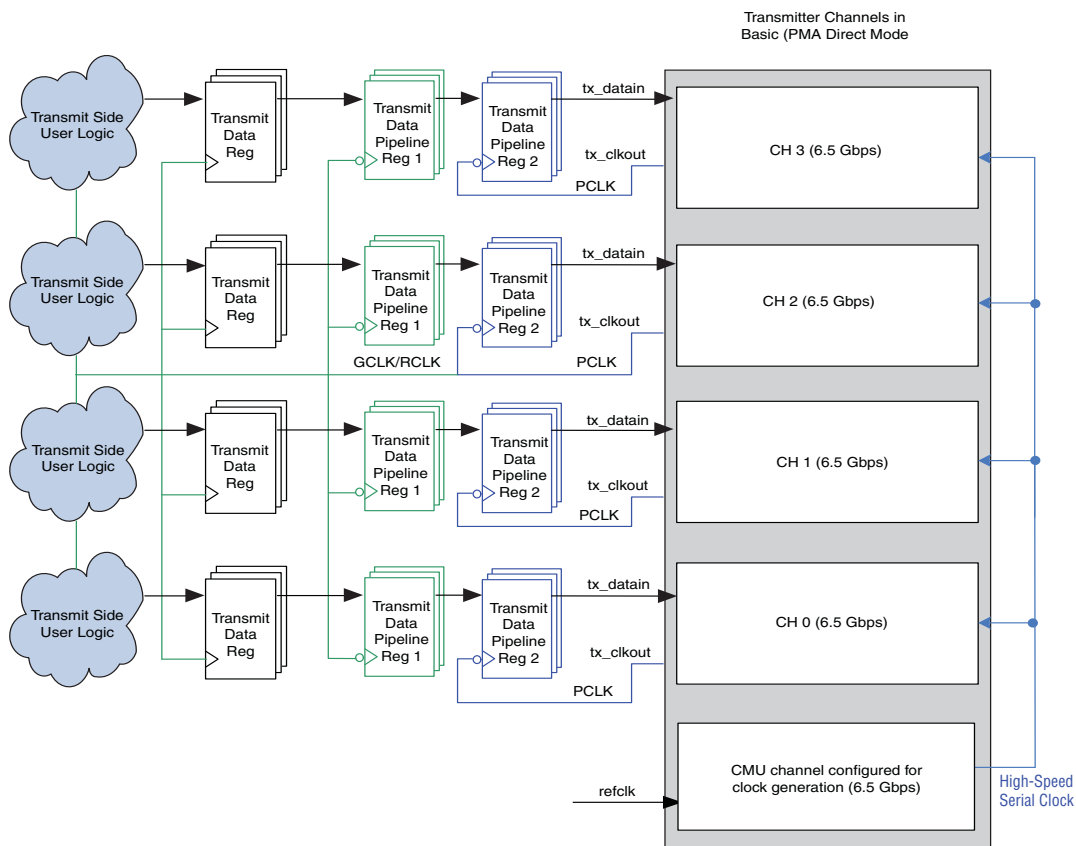
 Two stages of pipeline registers add a latency of two clock cycles to the transmit-side data path.

An additional stage of pipeline registers is added to meet the setup time between Transmit Data Pipeline Reg and Transmit Data Reg, as shown in [Figure 5 on page 6](#). The first stage of pipeline registers (Transmit Data Reg-1) is clocked by the same clock (RCLK or GCLK) that is being used to clock the transmit-side user logic (including Transmit Data Reg). The second stage of pipeline registers (Transmit Data Reg-2) is clocked by the PCLK. The additional stage of pipeline registers compensates for the clock skew between the faster PCLK and the slower RCLK or GCLK.

 Transmit Data Reg-1 is not required if the transmit-side user logic (including Transmit Data Reg) is negative-edge clocked.

Use pipeline registers to meet the timing requirement for multiple channels (bonded or non-bonded), as shown in [Figure 7](#). The sample design has four channels in a transceiver block configured in Basic (PMA Direct) xN mode at a data rate of 6.5 Gbps. The channel width for the interface is 20 bits wide. The transmit-side user logic is clocked with tx_clkout of channel 2. The pipeline registers are clocked using tx_clkout of individual channels with the PCLK network to meet the timing requirement at 6.5 Gbps.

Figure 7. Multiple Channels Using Pipeline Registers



You must modify the Synopsis Design Constraint (.sdc) file for the TimeQuest Timing Analyzer to account for the pipeline registers. As mentioned earlier, the transfer time for the transmit data changes from 0-to-1 clock cycles to 0.5-to-1.5 clock cycles with the addition of pipeline registers. To reflect this change, the setup time attributes for all pipeline registers must be modified by adding the following commands to the .sdc:

```
set_multicycle_path -setup -from [get_registers < pipeline registers >] 2
```

where, *<pipeline registers>* are the last stage of pipeline registers interfacing the transmit PMA (the Transmit Data Pipeline Reg 2 in Figure 7).

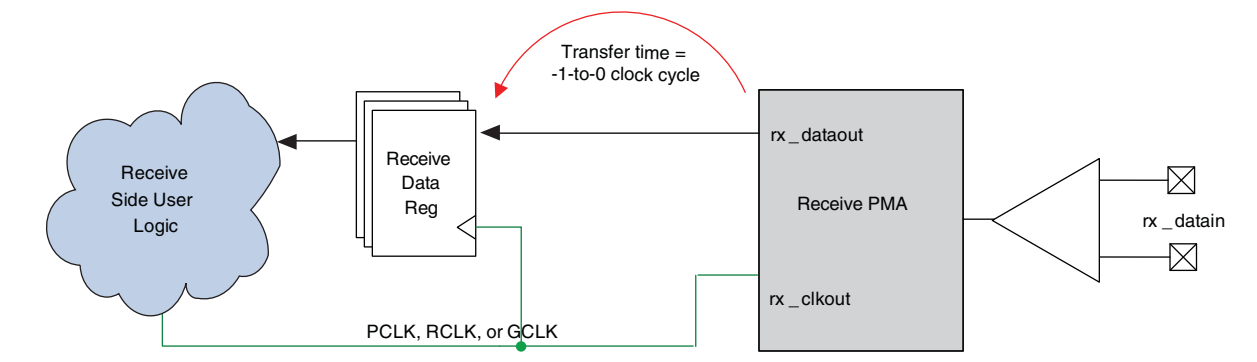
Meeting Timing on the Receive Side

Every channel on the receive side uses its own clock and the receive clock path is significantly slower than the receive data path. Because of this, you may observe hold time violations on the receive side of the FPGA fabric-Transceiver interface. To fix the hold time violations on the receive side, add the following constraints in the .sdc:

```
set_multicycle_path -setup -to [get_registers Receive Data Reg*] 0
```

where, Receive Data Reg are the registers used to capture the RX data from the rx_dataout port of the RX PMA in the FPGA core. Figure 8 shows how the timing constraint changes the transfer time from -1-to-0 clock cycles to account for the delay on the data path on the receive side.

Figure 8. Timing Constraint Changes due to Delay on the Data Path (Receive Side)



Timing Closure Script

The primary objective of the timing closure script is to add assignments to the design to meet timing. The script adds the following assignments to the project:

- Register placement—The script looks for the registers in a pre-compiled design that are used to interface with the transmit and receive PMA. Location assignments are added to these registers to optimally place these register to meet the timing requirement.
- Clock selection—The script also adds clock assignments to the registers used to interface to the transmit and receive PMA. Typically, you would use a clock with the least skew with respect to the data path to the meet timing requirement.

The script adds the assignments to the project's Quartus® II project settings file (.qsf). Every assignment added has a "hssi_place" tag so you can find them from the Quartus Assignment Editor, if necessary. To see the tags in the Assignment Editor, right-click on the column header and enable the Tag column.

 The timing closure script only runs on designs that have already achieved a fit. The script requires placement of all transceiver logic and all registers interfacing with the transceivers.

The script consists of two types of files. First is the master script file **pmadirect_ff_placer.tcl**; the second is a set of device-specific map files *<device name>_map.tcl*; for example, **EP4GX230_map.tcl**. To run the script you must have both the master script file and the map for the specific target device.

To run the script, copy the script files to the project directory and run the following command from within the project directory or from within the .tcl console in the Quartus II software:

```
quartus_cdb -t pmadirect_ff_placer.tcl <project> <switches>
```


where,

<project> = Project name (This is the name of the .qpf for the project. Do not include the .qpf extension.).

<switches> = Switches available.


-undo : = Removes all the assignments made previously added by the script.

-test : = shows the assignments the script adds without changing the .qsf.

 You may get errors running the script if you copy the text from this application note directly to the command prompt. To avoid these errors, type the command in full at the command prompt and execute it.


To remove the assignments added by the script run the following command:

```
quartus_cdb -t pmadirect_ff_placer.tcl <project> -undo
```

 The scripts, called "AN580_scripts.zip", are available for download from the [Literature: Application Notes](#) page on the Altera® website. Currently, the scripts for the EP4S360 and EP4S110 devices are not available. However, these scripts will be included in a future release.

Summary

Meeting timing for Basic (PMA Direct) mode has been simplified with the help of the timing closure script. The script eliminates the manual process of determining the best location and clock assignments. This also saves a considerable amount of compile time because there is no need to iterate fitting and timing analysis to determine the best assignments to achieve timing closure.

 If you are using the PLL method to achieve timing closure (described in the previous version of this application note) and you cannot meet timing, Altera recommends following the method described in this version of the application note to meet timing.

Document Revision History

Table 1 shows the revision history for this application note.

Table 1. Template Revision History

Date	Revision	Changes Made
February 2010	2.0	<ul style="list-style-type: none"> ■ Updated Page 1. ■ Updated the “Achieving Timing Closure”, “Steps to Achieve Timing Closure”, and “Summary” sections. ■ Updated Figure 1, Figure 4, Figure 5, Figure 6, Figure 7, and Figure 8. ■ Added link to the associated scripts. ■ Removed the “Design Considerations” and “PLL Method” sections. ■ Minor text edits.
June 2009	1.0	Initial release.



101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support
www.altera.com/support

Copyright © 2010 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 Corporation. 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.



I.S. EN ISO 9001