AN 558: Implementing Dynamic Reconfiguration in Arria II Devices

AN-558-3.8 Application Note

This application note describes how to use the dynamic reconfiguration feature and why you may want use this feature to reconfigure your Arria® II transceivers. It describes the four dynamic reconfiguration modes—offset cancellation for receiver channels, analog control, data rate division in transmitter (TX), and channel and TX PLL select/reconfig. The application note also describes the dynamic reconfiguration ports available in the ALTGX_RECONFIG instance and the FPGA fabric-transceiver channel interface signals.

☞ Unless otherwise stated, the examples used in this application note are for Arria II GX devices. If you are using Arria II GZ devices, the design implementation is similar to Arria II GX devices except for the size of certain port widths.

Arria II GX and GZ transceivers allow you to dynamically reconfigure various channel and clock multiplier unit (CMU) settings without powering down the device.

Table 1 lists the reasons you may want to reconfigure the transceivers and the reconfiguration modes for various reconfiguration requirements.

Table 1. Reconfiguration Modes for Various Reconfiguration Requirements

Reason for Reconfiguration	Reconfiguration Mode to Use	Stages (1)
Counter offset variations due to process, voltage, and temperature for analog circuits This feature is mandatory if you use receivers.	Offset Cancellation	I-IV
Fine-tune signal integrity by adjusting the transmitter and receiver buffer settings while bringing up a link	Analog controls	I-IV
Increase or decrease the data rate (/1, /2, /4) for auto negotiation	Data rate division in TX or Channel and TX PLL select/reconfig (2)	I-IV
Support multiple protocols with the same transceivers to add design flexibility	Analog controls (3) Channel and TX PLL select/reconfig	I-VI
Reset the CMU PLLs through Channel and TX PLL select/reconfig (4)	Channel and TX PLL select/reconfig	I-VI

Notes to Table 1:

(1) These are the setup stages required for specific reconfiguration modes. For more information, refer to "Setup Guide" on page 3.

(2) Channel and TX PLL select/reconfig is required if the negotiated data rate is not a multiple of 2 or 4. In that case, you need Stages I-VI.

(3) Analog controls may have to be adjusted for certain protocols.

(4) The reset ports of the CMU phase-locked loops (PLLs) are not provided in an ALTGX instance. You can work around this limitation by reconfiguring the CMU PLLs with the same settings. The reconfiguration process resets the selected CMU PLLs.

101 Innovation Drive
San Jose, CA 95134
www.altera.com

Reconfiguration Modes

Arria II GX and GZ dynamic reconfiguration modes include the following:

- **Offset Cancellation for Receiver Channels—**Always enabled for dynamic reconfiguration to counter the offset variations due to process, voltage, and temperature (PVT). The receiver buffer and receiver clock data recovery (CDR) require offset cancellation. Offset cancellation is automatically executed after each time the device is powered on. The control logic for offset cancellation is integrated into the dynamic reconfiguration controller. You must connect the ALTGX_RECONFIG instance to the ALTGX instances with the receiver channels in your design. For more information, refer to "Offset Cancellation Control for the Receiver Channels" on page 17.

  ☞ This feature is mandatory if you use receivers.

- **Analog control—**Use this mode to the modify pre-emphasis, equalization, DC gain, and voltage output differential (V_{OD}) settings. For more information, refer to "Analog Control" on page 18.
- **Data rate division in TX—**Use this mode to modify the data rate of the transmitter channel in multiples of 1, 2, and 4 by modifying the TX local divider block. This mode is available only for the transmit side with a non-bonded configuration. For more information, refer to "Data Rate Division in TX Operation" on page 28.

  For more information about data rate division in TX mode, refer to the *Transceiver Clocking for Arria II Devices* chapter in volume 2 of the *Arria II Device Handbook*.

- **Channel and TX PLL select/reconfig**—This mode is divided into the following four reconfiguration modes (for more information, refer to "Channel and TX PLL select/reconfig" on page 30):
  - **CMU PLL reconfiguration—**Use this mode to reconfigure the CMU PLL without affecting the remaining blocks of the transceiver channel. When you reconfigure the CMU PLL of a transceiver block to run at a different data rate, all the transceiver channels listening to this CMU PLL also are reconfigured to the new data rate. Channel settings are not affected.
  - **Channel and CMU PLL reconfiguration—**Use this mode to reconfigure a transceiver channel to a different functional mode and data rate by reconfiguring the channels and the CMU PLL.
  - **Channel reconfiguration with TX PLL select—**Use this mode to reconfigure the protocol of a transceiver channel by switching between the two TX CMU PLLs. The memory initialization file **(.mif)** contains the protocol setting and the TX PLL multiplex selection. Both CMU PLLs in a transceiver block are programmed through device programming and are not reconfigured in this mode.
  - **Central control unit reconfiguration—**Use this mode to reconfigure the central control unit (CCU) of the transceiver to switch between bonded physical coding sublayer (PCS) configurations with the same data width, such as Basic x4 and x8, XAUI, or PCI Express® (PCIe®) x4 and x8.

The following Arria II modes are not supported:

- Enable and disable pseudo-random binary sequence (PRBS) or built-in self test (BIST)
- Switching between a **Receiver-only** channel and a **Transmitter-only** channel
- Switching between a bonded x4 mode and a bonded x8 mode
- Switching between configurations that have rate matching enabled and configurations that do not have rate matching enabled

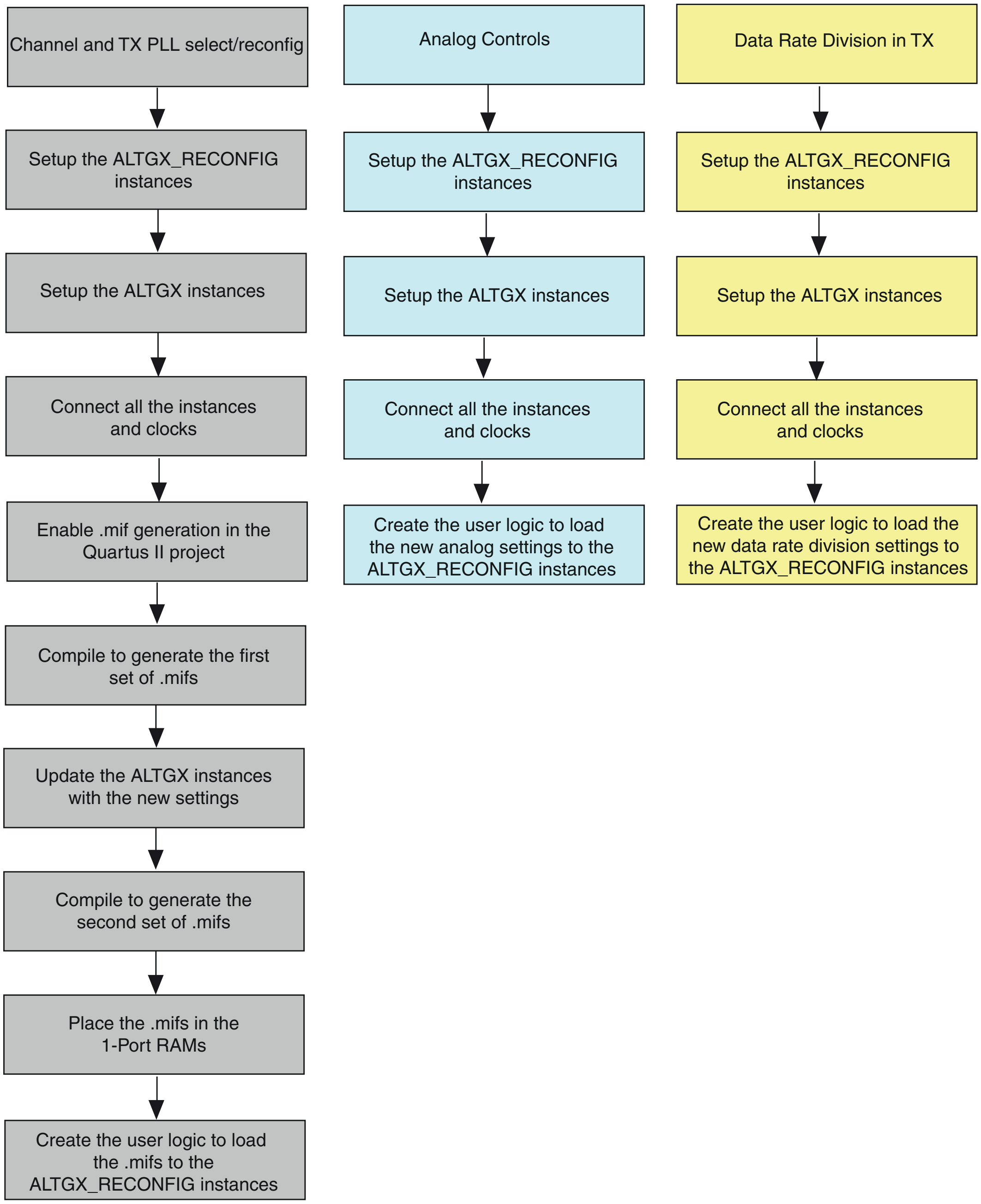
Setup Guide

Configuration mode implementation setup consists of the following stages:

- Stage I: Create the ALTGX_RECONFIG Instance
- Stage II: Enable Dynamic Reconfiguration in the ALTGXB Megafunction
- Stage III: Connect the ALTGX_RECONFIG with the ALTGX Instances
- Stage IV: Connect the Clock Ports of the ALTGX and ALTGX_RECONFIG Instances
- Stage V: Generate a .mif for Channel and TX PLL select/reconfig
- Stage VI: Create a 1-Port RAM for Channel and TX PLL select/reconfig

Figure 1 shows the design flow for each reconfiguration mode.

Figure 1. Design Flows


Channel and TX PLL select/reconfig
Setup the ALTGX_RECONFIG instances
Setup the ALTGX instances
Connect all the instances and clocks
Enable .mif generation in the Quartus II project
Compile to generate the first set of .mifs
Update the ALTGX instances with the new settings
Compile to generate the second set of .mifs
Place the .mifs in the 1-Port RAMs
Create the user logic to load the .mifs to the ALTGX_RECONFIG instances
Analog Controls
Setup the ALTGX_RECONFIG instances
Setup the ALTGX instances
Connect all the instances and clocks
Create the user logic to load the new analog settings to the ALTGX_RECONFIG instances
Data Rate Division in TX
Setup the ALTGX_RECONFIG instances
Setup the ALTGX instances
Connect all the instances and clocks
Create the user logic to load the new data rate division settings to the ALTGX_RECONFIG instances

The ALTGX instance represents the transceiver instance generated by the ALTGX MegaWizard™ Plug-In Manager. This term is used when the various inputs, outputs, and connections to the transceiver channels are described.

The ALTGX_RECONFIG instance represents the dynamic reconfiguration controller instance generated by the ALTGX_RECONFIG MegaWizard Plug-In Manager. The instance is created with FPGA resources.

The **.mif** stores the settings of each ALTGX instance and is read by user logic to the ALTGX_RECONFIG instance to reconfigure the ALTGX instance.

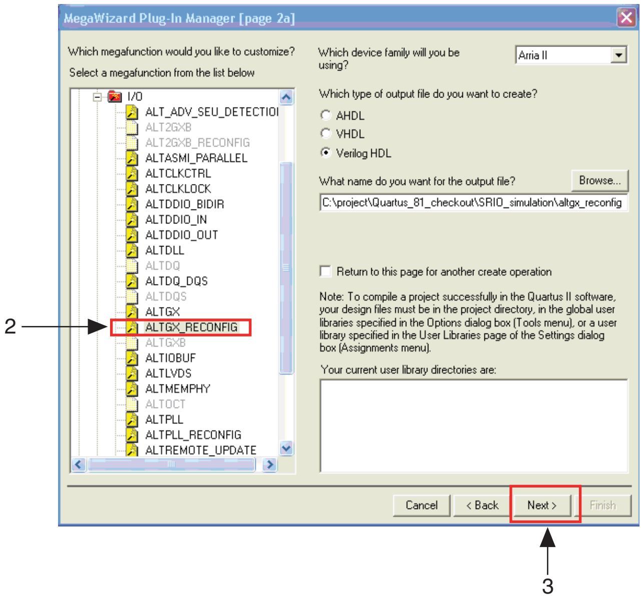
Stage I: Create the ALTGX_RECONFIG Instance

To create the ALTGX_RECONFIG instance, perform the following steps:

1. Open the MegaWizard Plug-In Manager.
2. Select **ALTGX_RECONFIG**.
3. Click **Next**.

   Figure 2 shows page 2a of the MegaWizard Plug-In Manager with these steps highlighted.

Figure 2. MegaWizard Plug-In Manager [page 2a]



MegaWizard Plug-In Manager [page 2a]

Which megafunction would you like to customize?

Select a megafunction from the list below

Which device family will you be using? Arria II

Which type of output file do you want to create?

○ AHDL
○ VHDL
● Verilog HDL

What name do you want for the output file?

C:\project\Quartus_81_checkout\SRIO_simulation\altgx_reconfig

□ Return to this page for another create operation

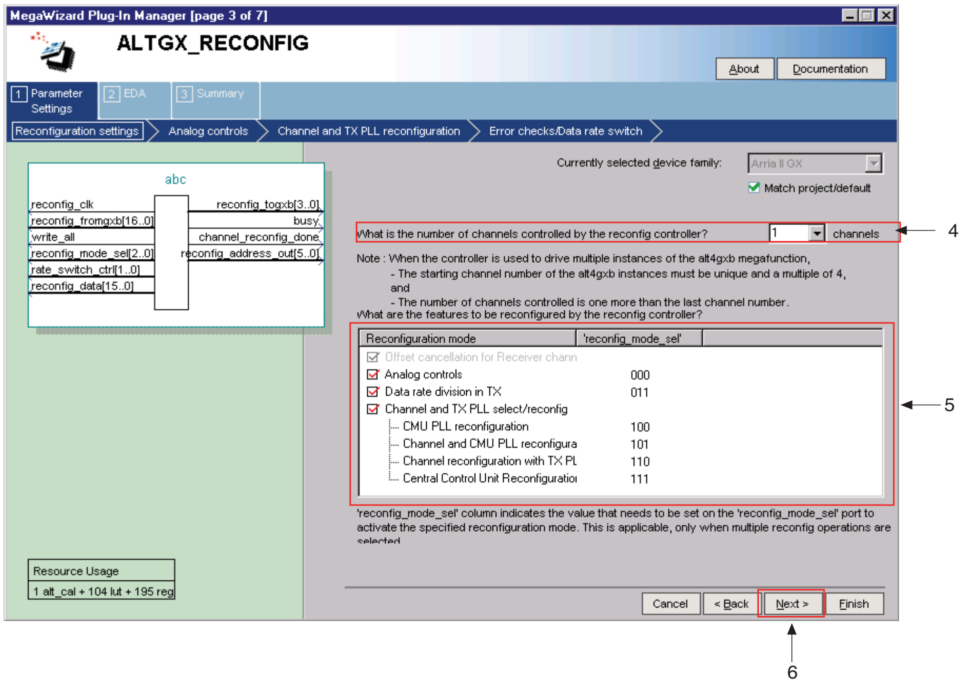
Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu).

Your current user library directories are:

4. Choose the number of channels to be controlled by the reconfiguration controller.
5. Select the reconfiguration modes for your design. In this example, **Analog controls**, **Data rate division in TX**, and **Channel and TX PLL select/reconfig** are selected.
6. Click **Next**.

Figure 3 shows page 3 of the MegaWizard Plug-In Manager with these steps highlighted.

Figure 3. MegaWizard Plug-In Manager [page 3 of 7]—Reconfiguration Settings



Currently selected device family: Arria II GX

☑ Match project/default

What is the number of channels controlled by the reconfig controller? 1 channels

Note : When the controller is used to drive multiple instances of the alt4gxb megafunction,
- The starting channel number of the alt4gxb instances must be unique and a multiple of 4,
and
- The number of channels controlled is one more than the last channel number.

What are the features to be reconfigured by the reconfig controller?

Reconfiguration mode	'reconfig_mode_sel'
☑ Offset cancellation for Receiver chann	
☑ Analog controls	000
☑ Data rate division in TX	011
☑ Channel and TX PLL select/reconfig	
CMU PLL reconfiguration	100
Channel and CMU PLL reconfigura	101
Channel reconfiguration with TX PL	110
Central Control Unit Reconfiguratio	111

'reconfig_mode_sel' column indicates the value that needs to be set on the 'reconfig_mode_sel' port to activate the specified reconfiguration mode. This is applicable, only when multiple reconfig operations are selected

Resource Usage

1 alt_cal + 104 lut + 195 reg

The MegaWizard Plug-In Manager guides you through the entire setup process. You can also refer to the corresponding sections to set up each mode.

Stage II: Enable Dynamic Reconfiguration in the ALTGXB Megafunction

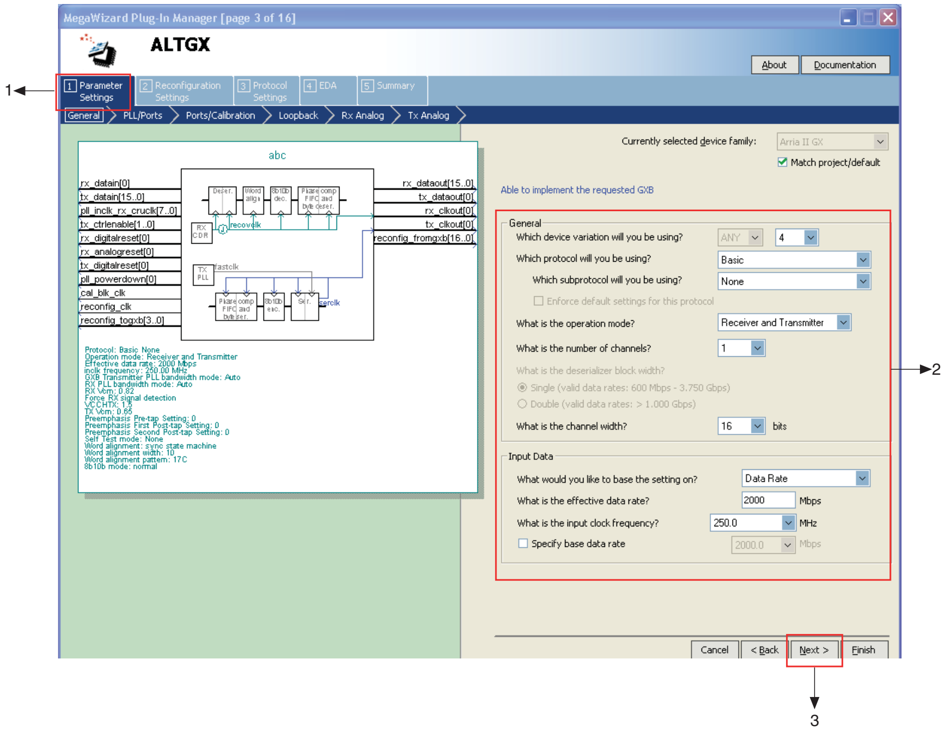
To enable dynamic reconfiguration in the ALTGXB megafunction, perform the following steps:

1. Click the **Parameter Settings** tab.
2. Complete the settings for the design implementation. If you use both CMU PLLs in an ALTGX instance, the **Parameter Settings** tab is the **Main PLL** setting. For channel and TX PLL select/reconfig, the **.mif** generated after compilation stores the settings selected on this tab.

3. Click **Next** to reconfigure the remaining settings on the **Parameter Settings** tab.

   Figure 4 shows page 3 of the MegaWizard Plug-In Manager with these steps highlighted.

Figure 4. MegaWizard Plug-In Manager [page 3 of 16]—Parameter Settings



```
Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: Auto
RX PLL bandwidth mode: Auto
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.5
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: None
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
```

Currently selected device family: Arria II GX

☑ Match project/default

Able to implement the requested GXB

General

Which device variation will you be using? ANY 4

Which protocol will you be using? Basic

Which subprotocol will you be using? None

☐ Enforce default settings for this protocol

What is the operation mode? Receiver and Transmitter

What is the number of channels? 1

What is the deserializer block width?

● Single (valid data rates: 600 Mbps - 3.750 Gbps)

○ Double (valid data rates: > 1.000 Gbps)

What is the channel width? 16 bits

Input Data

What would you like to base the setting on? Data Rate

What is the effective data rate? 2000 Mbps

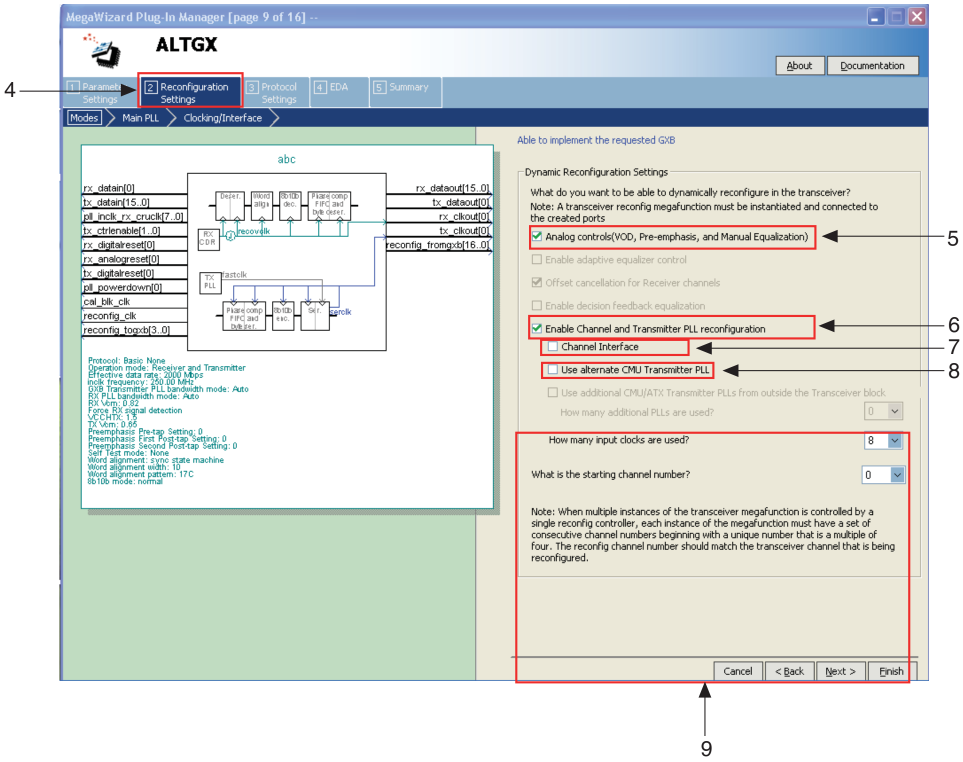
What is the input clock frequency? 250.0 MHz

☐ Specify base data rate 2000.0 Mbps

4. Click the **Reconfiguration Settings** tab after you complete all the settings on the **Parameter Settings** tab.
5. Check **Analog controls** if you plan to reconfigure the PMA analog settings.
6. Check **Enable Channel and Transmitter PLL reconfiguration** if you plan to use data rate division in TX and channel and TX PLL select/reconfig.
7. Check **Channel Interface** if you plan to reconfigure the FPGA fabric-transceiver interface signals.
8. Check **Use alternate CMU Transmitter PLL** if you plan to reconfigure the data rate/protocol of the transceiver channels by switching to another CMU PLL within the same transceiver block. Enabling this setting creates the **Alt PLL** tab for you to complete the settings for this CMU PLL and the new protocol. For more information, refer to "Main PLL and Alternate PLL" on page 53.
9. Complete the remaining settings and click **Next**.

Figure 5 shows page 9 of the MegaWizard Plug-In Manager with these steps highlighted.

Figure 5. MegaWizard Plug-In Manager [page 9 of 16]—Reconfiguration Settings, Modes



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: Auto
RX PLL bandwidth mode: Auto
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.5
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: None
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Dynamic Reconfiguration Settings

What do you want to be able to dynamically reconfigure in the transceiver?

Note: A transceiver reconfig megafunction must be instantiated and connected to the created ports

☑ Analog controls(VOD, Pre-emphasis, and Manual Equalization)

□ Enable adaptive equalizer control

☑ Offset cancellation for Receiver channels

□ Enable decision feedback equalization

☑ Enable Channel and Transmitter PLL reconfiguration

□ Channel Interface

□ Use alternate CMU Transmitter PLL

□ Use additional CMU/ATX Transmitter PLLs from outside the Transceiver block

How many additional PLLs are used? 0

How many input clocks are used? 8

What is the starting channel number? 0

Note: When multiple instances of the transceiver megafunction is controlled by a single reconfig controller, each instance of the megafunction must have a set of consecutive channel numbers beginning with a unique number that is a multiple of four. The reconfig channel number should match the transceiver channel that is being reconfigured.

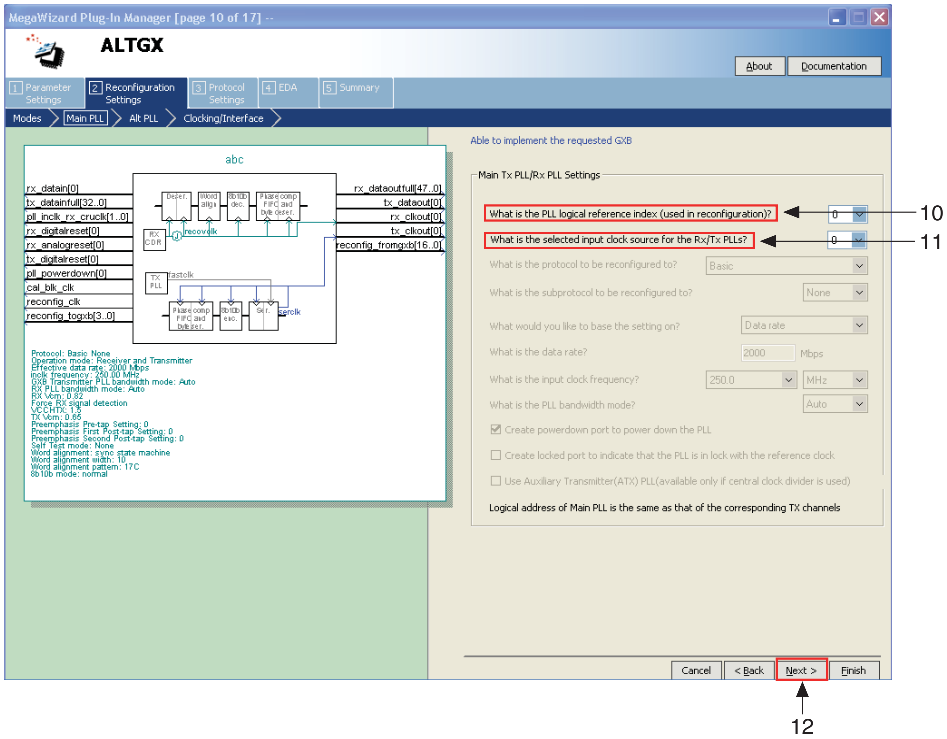
10. Assign the main PLL used for the protocol specified on the **Parameter Settings** tab, with **0** or **1**. This reference index links to the protocol. The PLL logical reference index is available only when you enable **Use alternate CMU Transmitter PLL** (refer to step 8).

11. Specify the input clock source from all the available clock sources specified in step 9.

12. Click **Next**.

Figure 6 shows page 10 of the MegaWizard Plug-In Manager with these steps highlighted. Note that the Atl PLL was created in Step 8.

Figure 6. MegaWizard Plug-In Manager [page 10 of 17]—Reconfiguration Settings, Main PLL



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: Auto
RX PLL bandwidth mode: Auto
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.5
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: None
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Main Tx PLL/Rx PLL Settings

What is the PLL logical reference index (used in reconfiguration)? 0

What is the selected input clock source for the Rx/Tx PLLs? 0

What is the protocol to be reconfigured to? Basic

What is the subprotocol to be reconfigured to? None

What would you like to base the setting on? Data rate

What is the data rate? 2000 Mbps

What is the input clock frequency? 250.0 MHz

What is the PLL bandwidth mode? Auto

☑ Create powerdown port to power down the PLL

☐ Create locked port to indicate that the PLL is in lock with the reference clock

☐ Use Auxiliary Transmitter(ATX) PLL(available only if central clock divider is used)

Logical address of Main PLL is the same as that of the corresponding TX channels

For more information, refer to "Guidelines for the logical_tx_pll_sel and logical_tx_pll_sel_en Ports" on page 72 and "Guidelines for Specifying the Input Reference Clocks" on page 76.

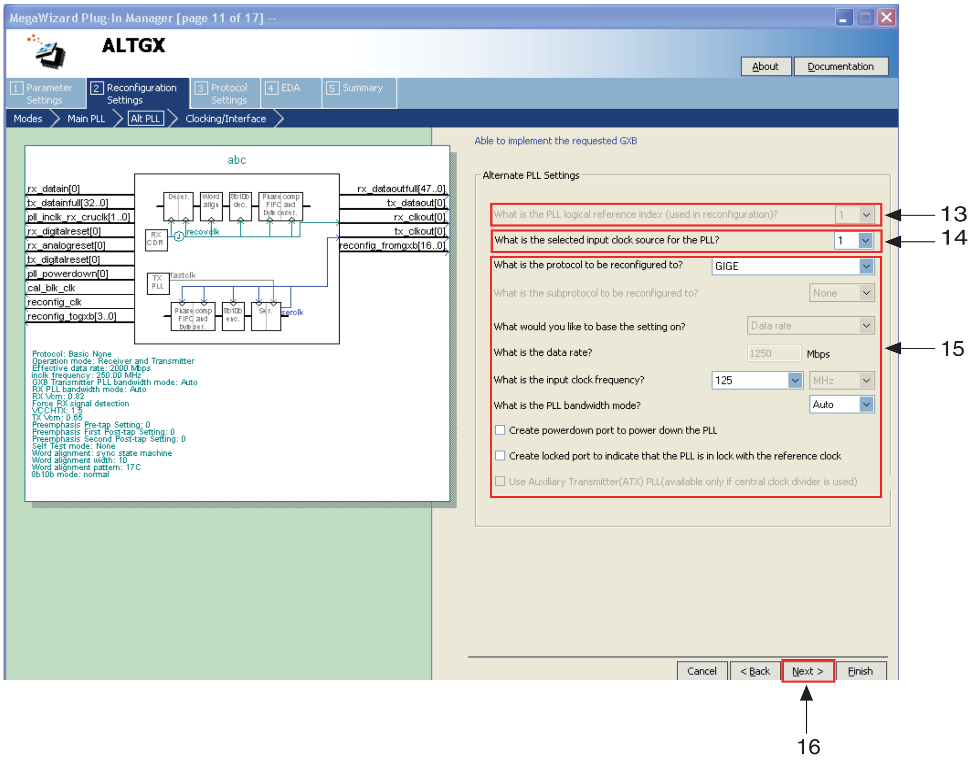
☞ The **Alt PLL** tab is only available when you enable **Use alternate CMU Transmitter PLL** (refer to step 8).

> ☞ The Alternate PLL is automatically set to the complement value of the main PLL.

13. Specify the input clock source for the Alternate PLL.
14. Complete the protocol settings for the Alternate PLL.
15. Click **Next**.

Figure 7 shows page 11 of the MegaWizard Plug-In Manager with these steps highlighted.

Figure 7. MegaWizard Plug-In Manager [page 11 of 17]—Reconfiguration Settings, Alt PLL



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: Auto
RX PLL bandwidth mode: Auto
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.5
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: None
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Alternate PLL Settings

What is the PLL logical reference index (used in reconfiguration)? 1

What is the selected input clock source for the PLL? 1

What is the protocol to be reconfigured to? GIGE

What is the subprotocol to be reconfigured to? None

What would you like to base the setting on? Data rate

What is the data rate? 1250 Mbps

What is the input clock frequency? 125 MHz

What is the PLL bandwidth mode? Auto

□ Create powerdown port to power down the PLL

□ Create locked port to indicate that the PLL is in lock with the reference clock

□ Use Auxiliary Transmitter(ATX) PLL(available only if central clock divider is used)

As the design flow indicates, you may also have to generate a **.mif** for backward reconfiguration. To do this, you must swap between the settings on the **Parameter Settings** and **Alt PLL** tabs to generate another **.mif**; for example, from GIGE to Basic. The main PLL is then set to **GIGE** and the Alt PLL is set to **BASIC**. However, the logical PLL reference index must be consistent with the protocol, so the PLL logical reference index 0 remains for BASIC protocol and the PLL logical reference index 1 remains for GIGE protocol.

For **.mif** generation, refer to "Stage V: Generate a .mif for Channel and TX PLL select/reconfig" on page 14 and "Stage VI: Create a 1-Port RAM for Channel and TX PLL select/reconfig" on page 16.

16. Select the clocking schemes for the transmitters and receivers.

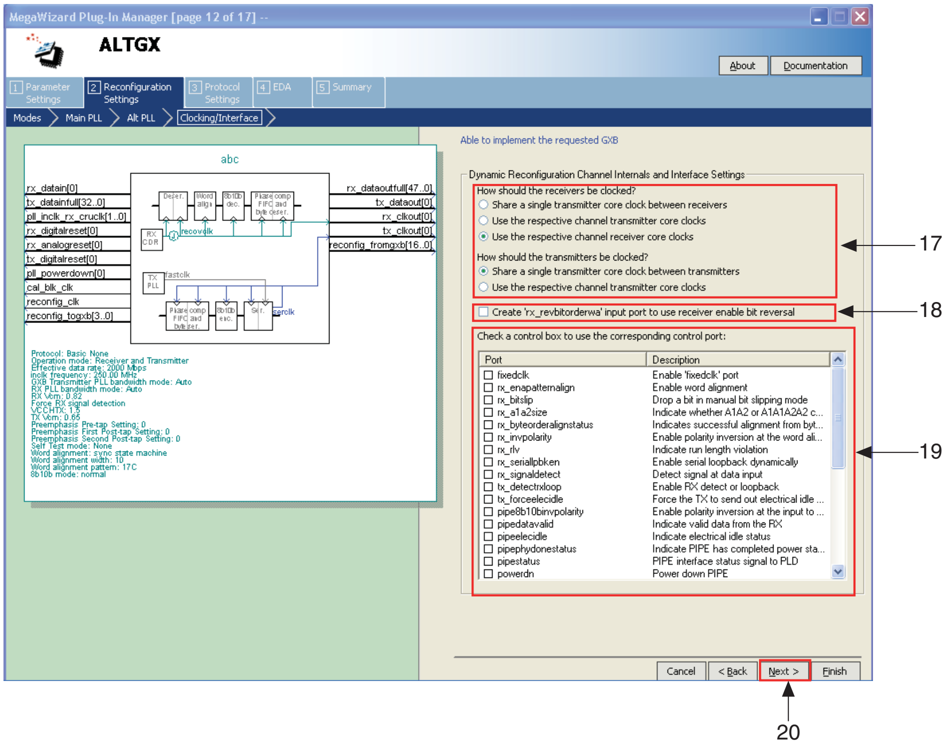
17. Enable receiver bit reversal.

18. Check the interface signals that are required for this protocol. This option is only available when you enable the **Channel Interface** option to allow additional interface signals for the new protocol (refer to step 7).

19. Click **Next** to exit **Reconfiguration Settings**.

Figure 8 shows page 12 of the MegaWizard Plug-In Manager with these steps highlighted.

Figure 8. MegaWizard Plug-In Manager [page 11 of 17]—Reconfiguration Settings, Clocking/Interface



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: Auto
RX PLL bandwidth mode: Auto
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.5
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: None
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Dynamic Reconfiguration Channel Internals and Interface Settings

How should the receivers be clocked?

○ Share a single transmitter core clock between receivers
○ Use the respective channel transmitter core clocks
● Use the respective channel receiver core clocks

How should the transmitters be clocked?

● Share a single transmitter core clock between transmitters
○ Use the respective channel transmitter core clocks

□ Create 'rx_revbitorderwa' input port to use receiver enable bit reversal

Check a control box to use the corresponding control port:

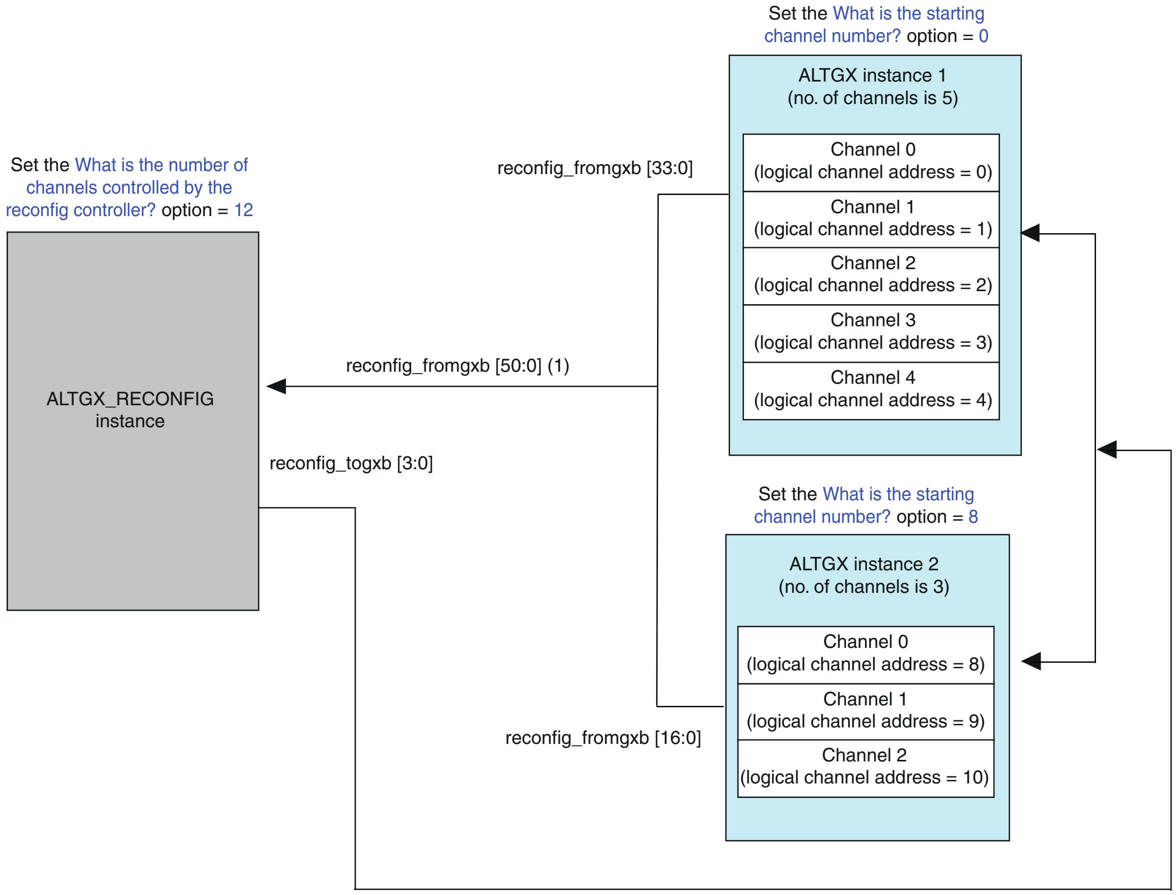
Port	Description
□ fixedclk	Enable 'fixedclk' port
□ rx_enapatternalign	Enable word alignment
□ rx_bitslip	Drop a bit in manual bit slipping mode
□ rx_a1a2size	Indicate whether A1A2 or A1A1A2A2 c...
□ rx_byteorderalignstatus	Indicates successful alignment from byt...
□ rx_invpolarity	Enable polarity inversion at the word ali...
□ rx_rlv	Indicate run length violation
□ rx_seriallpbken	Enable serial loopback dynamically
□ rx_signaldetect	Detect signal at data input
□ tx_detectrxloop	Enable RX detect or loopback
□ tx_forceelecidle	Force the TX to send out electrical idle ...
□ pipe8b10binvpolarity	Enable polarity inversion at the input to ...
□ pipedatavalid	Indicate valid data from the RX
□ pipeelecidle	Indicate electrical idle status
□ pipephydonestatus	Indicate PIPE has completed power sta...
□ pipestatus	PIPE interface status signal to PLD
□ powerdn	Power down PIPE

For more information about the **Reconfiguration Settings** tab, refer to "Clocking/Interface Options" on page 54.

Stage III: Connect the ALTGX_RECONFIG with the ALTGX Instances

Connect the ALTGX_RECONFIG instance with the ALTGX instance, as shown in Figure 9.

Figure 9. Connect the ALTGX_RECONFIG Instance with the ALTGX Instance


Set the What is the starting channel number? option = 0
ALTGX instance 1
(no. of channels is 5)
Channel 0
(logical channel address = 0)
Channel 1
(logical channel address = 1)
Channel 2
(logical channel address = 2)
Channel 3
(logical channel address = 3)
Channel 4
(logical channel address = 4)
reconfig_fromgxb [33:0]
Set the What is the number of channels controlled by the reconfig controller? option = 12
ALTGX_RECONFIG instance
reconfig_fromgxb [50:0] (1)
reconfig_togxb [3:0]
Set the What is the starting channel number? option = 8
ALTGX instance 2
(no. of channels is 3)
Channel 0
(logical channel address = 8)
Channel 1
(logical channel address = 9)
Channel 2
(logical channel address = 10)
reconfig_fromgxb [16:0]

Note to Table 12:

(1) The starting channel number must be a multiple of four.

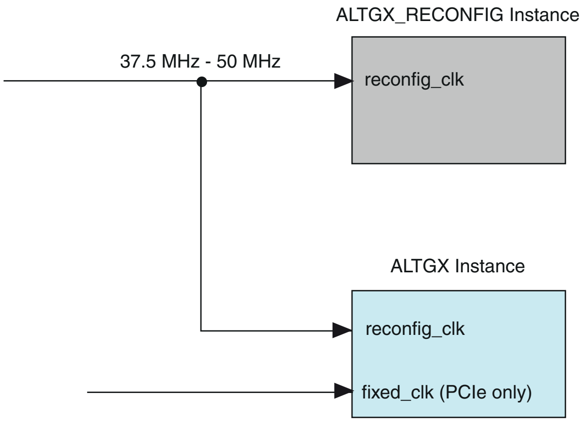
To connect the ALTGX_RECONFIG instance with the ALTGX instance, follow these steps:

1. Connect the `reconfig_togxb` ports of both the ALTGX and ALTGX_RECONFIG instances.
2. Connect the `reconfig_fromgxb` port of the ALTGX instance that has the highest **What is the starting channel number?** option to the MSB of the `reconfig_fromgxb` port of the ALTGX_RECONFIG instance (ALTGX instance 2 in Figure 9).

Stage IV: Connect the Clock Ports of the ALTGX and ALTGX_RECONFIG Instances

Connect the reconfig_clk and fixed_clk (for PCIe only) of the ALTGX instance and the ALTGX_RECONFIG instance, as shown in Figure 10.

Figure 10. Clock Ports Connection for ALTGX and ALTGX_RECONFIG Instances *(1)*, *(2)*



Notes to Figure 10:

(1) For **Transmitter only** and PCIe, the frequency range for ALTGX and ALTGX_RECONFIG is 2.5 MHz to 50 MHz. Offset cancellation is not required for transmitters and is accomplished using a fixed clock in PCIe mode.

(2) The reconfig_clk signal must be a free-running clock sourced from an I/O clock pin. Also, Altera recommends driving the reconfig_clk signal on a global clock resource.

Using a GPLL to Drive the reconfig_clk Port

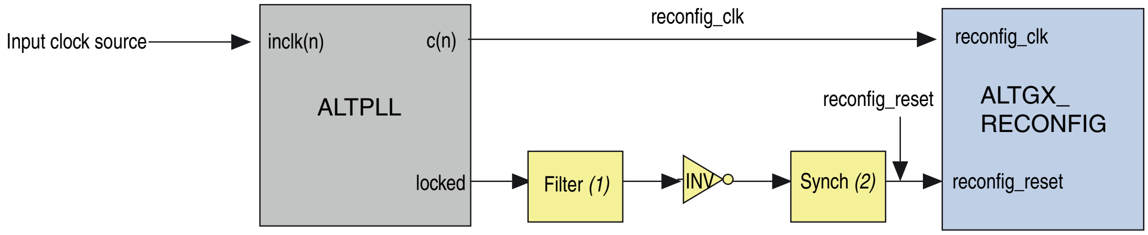
If you are not using a free running clock from a non-transceiver I/O clock pin to provide clock to the reconfig_clk port, use a general-purpose PLL (GPLL) to generate reconfig_clk sourced from an I/O clock pin. However, if you are using a GPLL to drive the reconfig_clk port, you must be aware of the following requirements (refer to Figure 11):

- You must add a milliseconds filter with the locked stability counter to filter any glitch of the ALTPLL lock status signal.

  ☞ The filter circuit is optional if you have verified that the input clock source to the ALTPLL inclk is not jittery. If the input clock source is not jittery, ALTPLL is able to lock without a glitch at the beginning of the PLL lock time.

- You must assert reconfig_reset until the GPLL clock output is stable. Insert an inverter between the GPLL locked output and reconfig_input to assert reconfig_reset.
- The reconfig_reset input is a synchronous reset. To synchronize the reconfig_reset input to the reconfig_clk domain, add a synchronizer between the inverter and the reconfig_reset port of the ALTGX_RECONFIG Controller.

Figure 11. Dynamic Reconfiguration reconfig_clk Requirements Sourced from a GPLL



Notes to Figure 11:

(1) This is a milliseconds filter with a locked stability counter.

(2) The output of the D flipflop is synchronized to the `reconfig_clk` domain.

After the `busy` signal deasserts after initialization, reasserting `reconfig_reset` does not restart the offset cancellation process.

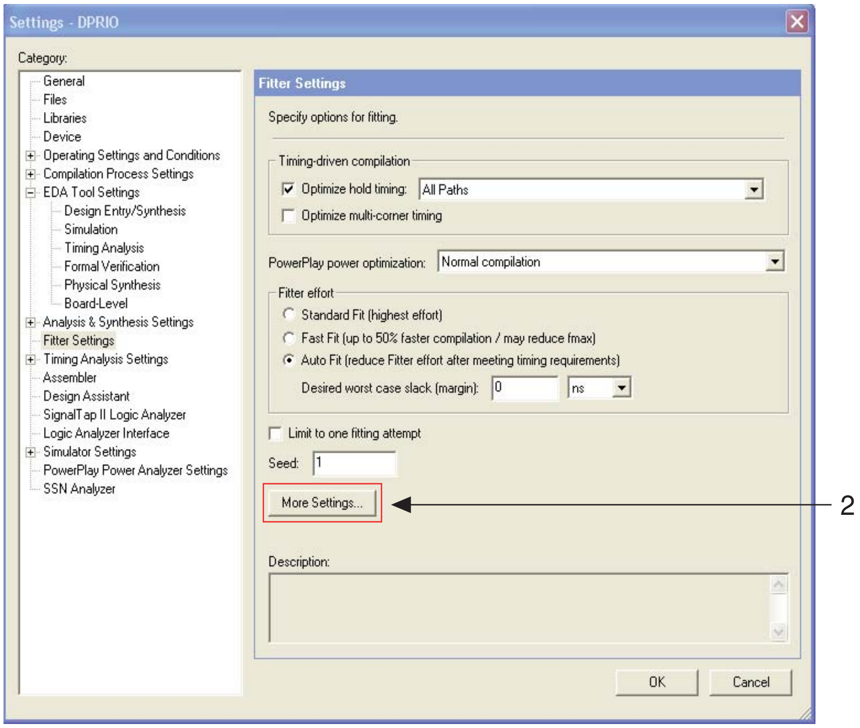
To enable the `reconfig_reset` port to the `altgx_reconfig` block, you must enable the **Channel and TX PLL select/reconfig** option.

Stage V: Generate a .mif for Channel and TX PLL select/reconfig

To generate a **.mif** for channel and TX PLL select/reconfig, follow these steps:

1. On the Assignments menu of the Quartus II software, select **Settings**.
2. In the **Settings** dialog box, select **Fitter Settings**, then click **More Settings** (Figure 12).

Figure 12. More Settings Tab



Specify options for fitting.

Timing-driven compilation

☑ Optimize hold timing: All Paths

□ Optimize multi-corner timing

PowerPlay power optimization: Normal compilation

Fitter effort

○ Standard Fit (highest effort)

○ Fast Fit (up to 50% faster compilation / may reduce fmax)

● Auto Fit (reduce Fitter effort after meeting timing requirements)

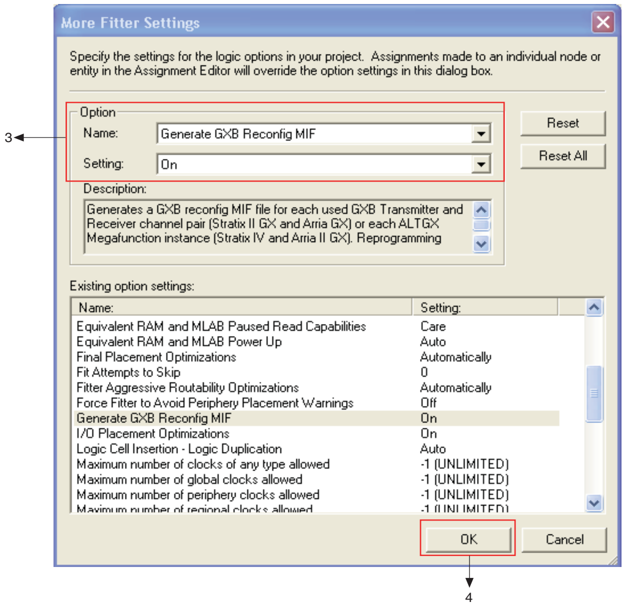
Desired worst case slack (margin): 0 ns

□ Limit to one fitting attempt

Seed: 1

3. In the **Option** box of the **More Fitter Settings** page, set the **Generate GXB Reconfig MIF** option to **On** based on the dynamic reconfiguration mode enabled (Figure 13).
4. Click **OK**.

Figure 13. More Fitter Settings Page



Specify the settings for the logic options in your project. Assignments made to an individual node or entity in the Assignment Editor will override the option settings in this dialog box.

Option

Name: Generate GXB Reconfig MIF

Setting: On

Description:

Generates a GXB reconfig MIF file for each used GXB Transmitter and Receiver channel pair (Stratix II GX and Arria GX) or each ALTGX Megafunction instance (Stratix IV and Arria II GX). Reprogramming

Existing option settings:

Name:	Setting:
Equivalent RAM and MLAB Paused Read Capabilities	Care
Equivalent RAM and MLAB Power Up	Auto
Final Placement Optimizations	Automatically
Fit Attempts to Skip	0
Fitter Aggressive Routability Optimizations	Automatically
Force Fitter to Avoid Periphery Placement Warnings	Off
Generate GXB Reconfig MIF	On
I/O Placement Optimizations	On
Logic Cell Insertion - Logic Duplication	Auto
Maximum number of clocks of any type allowed	-1 (UNLIMITED)
Maximum number of global clocks allowed	-1 (UNLIMITED)
Maximum number of periphery clocks allowed	-1 (UNLIMITED)
Maximum number of regional clocks allowed	-1 (UNLIMITED)

The file name is based on the ALTGX instance name (<***instance name***>**.mif**). One design can have multiple **.mif** files and you can use one **.mif** to reconfigure multiple channels.

You can generate multiple **.mif** files efficiently with either of the following two methods without pin location assignments.

Method 1:

1. Configure the ALTGX instance with one protocol setting using the MegaWizard Plug-In Manager.
2. Under **Tcl Console**, type the following:

   ```
   exec altgx_mifgen [ALTGX.v file] [output MIF file]
   ```

   The file is created in the project directory.
3. Repeat steps 1 and 2 for alternate configurations.

Method 2:

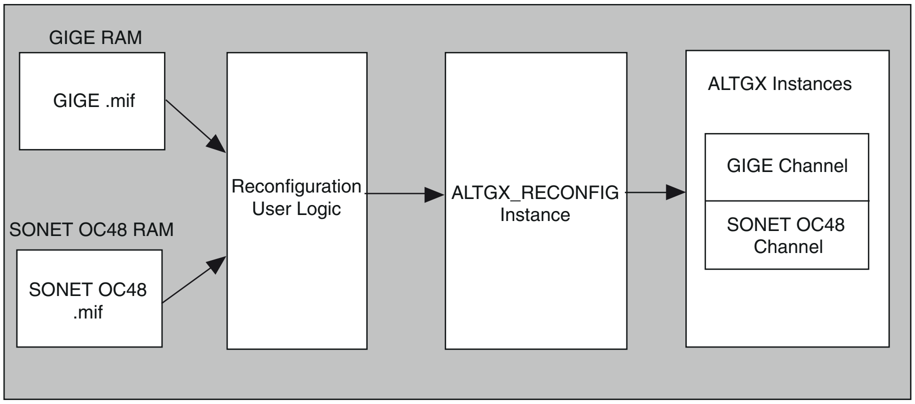
1. In the top-level design, instantiate all configurations of the ALTGX instantiation for which the **.mif** is required.
2. Connect the appropriate clock inputs of all the ALTGX instantiations.
3. Compile the design. The **.mif** files are generated for all ALTGX configurations.
4. The **.mif** is located in the *<Project_DIR>/reconfig_mif* folder.

Stage VI: Create a 1-Port RAM for Channel and TX PLL select/reconfig

This section describes how to create a 1-Port RAM for channel and TX PLL select/reconfig mode.

Figure 14 shows the implementation of TX PLL select/reconfig mode.

Figure 14. Implementation of TX PLL select/reconfig Mode



Store the **.mif** in on-chip or off-chip memory and connect it to the dynamic reconfiguration controller, as shown in Figure 14.

When applying a **.mif** in the user design, be sure to do the following:

- Use the RAM: 1-PORT megafunction to instantiate a memory block with the sizes listed in Table 2.

Table 2. ALTGX Configuration Memory Sizes

ALTGX Configuration	Memory Size (Bit)
Duplex + Central control unit	60 x 16
Duplex	55 x 16
Receiver only	38 x 16
Transmitter only	19 x 16

- Choose the size of the memory block based on the size of the **.mif** generated.
- Instantiate the **.mif** in the memory block.
- Create user logic to read the individual **.mif** files from the RAM. For more information, refer to "Channel and TX PLL select/reconfig" on page 30.

Whenever a **.mif** is applied to a channel, the PMA analog controls for that channel are set to the default settings in the ALTGX instance.

Offset Cancellation Control for the Receiver Channels

The Arria II device provides an offset cancellation circuit per receiver channel to counter the offset variations due to PVT. The offset cancellation logic corrects these offsets. The receiver buffer and CDR require offset cancellation.

After the device is powered on each time, offset cancellation is automatically executed. The control logic for offset cancellation is integrated into the dynamic reconfiguration controller. You must connect the ALTGX_RECONFIG instance to the ALTGX instances with the receiver channels in your design. Also, you must connect `reconfig_fromgxb`, `reconfig_togxb`, and the necessary clock signals to both the ALTGX_RECONFIG and ALTGX (with receiver channels) instances.

Offset Cancellation for Receiver Channels Operation

Every ALTGX instance for **Receiver and Transmitter** or **Receiver-only** configurations requires that offset cancellation for the **receiver channels** option is enabled on the **Reconfig** tab of the ALTGX MegaWizard Plug-In Manager. This option is enabled by default for the two configurations. This option is disabled for the **Transmitter-only** configuration.

☞ The `gxb_powerdown` signal must not be asserted during the offset cancellation sequence.

For example, assume you have a scenario where the design has ALTGX instances with channels of both **Transmitter-only** and **Receiver-only** configurations. You must also include the **Transmitter-only** channels while setting the **What is the starting channel number?** option in the ALTGX instance and while setting the **What is the number of channels controlled by the reconfig controller?** option in the ALTGX_RECONFIG instance for receiver offset cancellation.

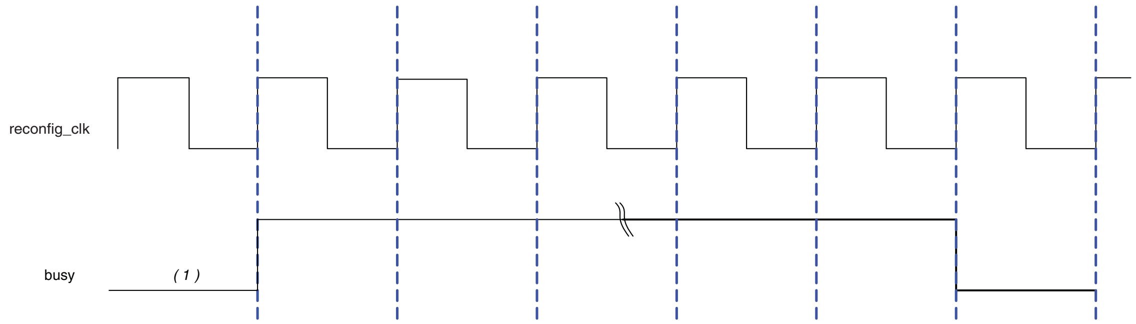
In this scenario, the `busy` signal behavior is as follows (refer to Figure 15):

- After the device powers up, the `busy` signal remains low for the first `reconfig_clk` clock cycle.
- The `busy` signal then is asserted for the second `reconfig_clk` clock cycle, when the dynamic reconfiguration controller initiates the offset cancellation process and the receiver path disconnects from the I/O.
- De-assertion of the `busy` signal indicates successful completion of the offset cancellation process.

CAUTION When the `busy` signal is high, do not assert the `write_all` signal or `read` signal to indicate either a write or read operation. Ensure the `busy` signal is low before asserting the `write_all` signal or `read` signal high (for a write or read instruction).

Figure 15 shows the dynamic reconfiguration signals transitioning during offset cancellation on the receiver channels.

Figure 15. Dynamic Reconfiguration Signals Transition During Offset Cancellation on Receiver Channels



Note to Figure 15:

(1) After the device powers up, the busy signal remains low for the first reconfig_clk cycle.

Due to the offset cancellation process, the transceiver reset sequence has changed. For more information, refer to the *Reset Control and Power Down in Arria II Devices* chapter in volume 2 of the *Arria II Device Handbook*.

Functional Simulation of the Offset Cancellation Process

You must connect the ALTGX_RECONFIG instances to the ALTGX instances in your design for functional simulation. Functional simulation uses a reduced timing model of the dynamic reconfiguration controller. Therefore, the duration of the dynamic reconfiguration process is 16 reconfig_clk clock cycles for functional simulation only. The gxb_powerdown signal must not be asserted during the offset cancellation sequence (for functional simulation and silicon).

Analog Control

You can dynamically reconfigure the PMA controls of a transceiver channel with the following methods:

- Using logical_channel_address to reconfigure specific transceiver channels (refer to "Method 1: Using logical_channel_address to Reconfigure Specific Transceiver Channels" on page 19)
- Writing the same control signals for all transceiver channels (refer to "Method 2: Writing the Same Control Signals to Control All the Transceiver Channels" on page 21)
- Writing different control signals for all transceiver channels at the same time (refer to "Method 3: Writing Different Control Signals for all the Transceiver Channels at the Same Time" on page 24)

For each method, you can additionally use the rx_tx_duplex_sel port.

Enable the `rx_tx_duplex_sel` port by selecting the **Use 'rx_tx_duplex_sel' port to enable RX only, TX only or duplex reconfiguration** option on the **Error checks/Data rate switch** tab of the ALTGX_RECONFIG MegaWizard Plug-In Manager. This option is available only when you select the **Analog controls** option on the **Reconfiguration settings** tab of the ALTGX_RECONFIG MegaWizard Plug-In Manager. For more information, refer to "Port Definition" on page 37.

Method 1: Using logical_channel_address to Reconfigure Specific Transceiver Channels

Enable the `logical_channel_address` port by selecting the **Use 'logical_channel_address' port** option on the **Analog controls** tab. Method 1 is applicable only for a design where the dynamic reconfiguration controller controls more than one channel.

You can additionally reconfigure either the receiver portion, transmitter portion, or both the receiver and transmitter portions of the transceiver channel by setting the corresponding value on the `rx_tx_duplex_sel` input port. For more information, refer to "Port Definition" on page 37.

Connecting the PMA Control Ports

The selected PMA control ports remain fixed in width, regardless of the number of channels controlled by the ALTGX_RECONFIG instance:

- `tx_vodctrl` and `tx_vodctrl_out` are fixed to 3 bits
- `tx_preemp` and `tx_preemp_out` are fixed to 5 bits
- `rx_eqdcgain` and `rx_eqdcgain_out` are fixed to 3 bits
- `rx_eqctrl` and `rx_eqctrl_out` are fixed to 4 bits

Write Transaction

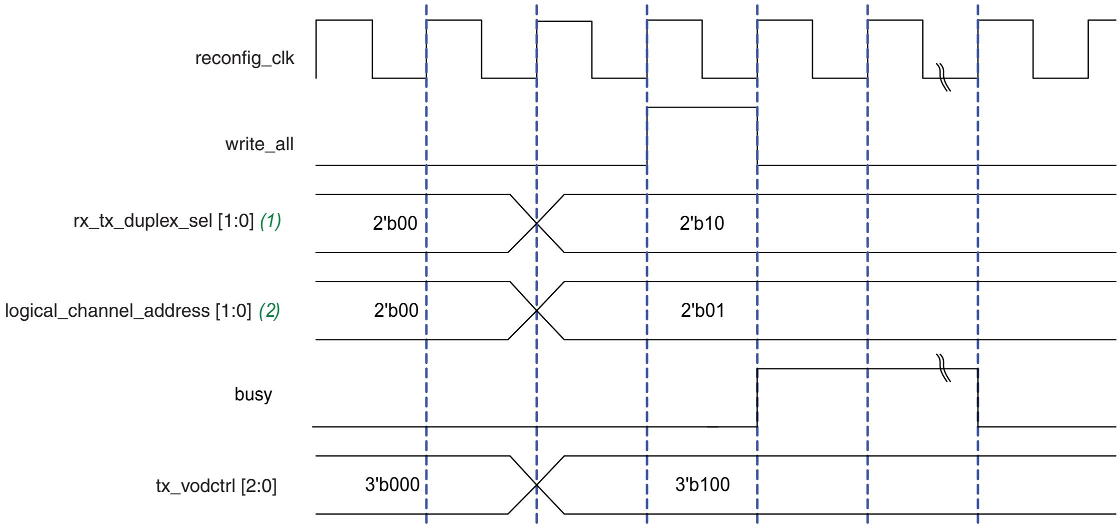
To complete a write transaction, perform the following steps:

1. Set the selected PMA control ports to the desired settings (for example, `tx_vodctrl` = **3'b100**).
2. Set the `logical_channel_address` input port to the logical channel address of the transceiver channel whose PMA controls you want to reconfigure.
3. Set the `rx_tx_duplex_sel` port to **2'b10** so that only the transmit PMA controls are written to the transceiver channel.
4. Ensure that the `busy` signal is low before you start a write transaction.
5. Assert the `write_all` signal for one `reconfig_clk` clock cycle.

The `busy` output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy writing the PMA control values. When the write transaction has completed, the `busy` signal goes low.

Figure 16 shows the write transaction waveform for Method 1.

Figure 16. Write Transaction Waveform—Use logical_channel_address Port Option



Notes to Figure 16:

(1) In this waveform example, you are writing to the transmitter portion of the channel only.

(2) In this waveform example, the number of channels connected to the dynamic reconfiguration controller is four. Therefore, the `logical_channel_address` port is 2 bits wide.

Read Transaction

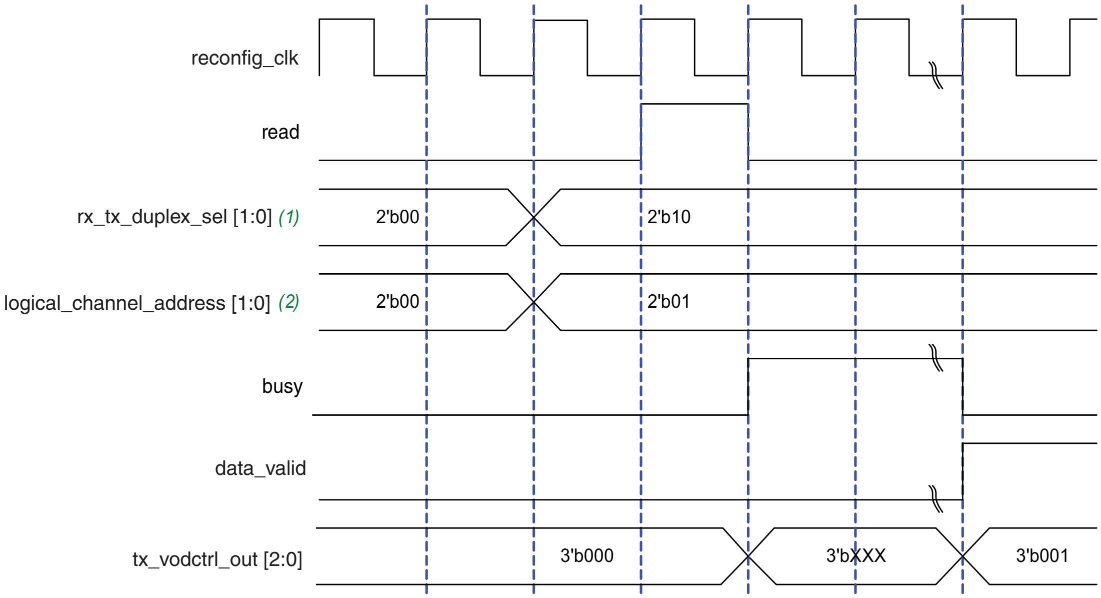
For example, to read the existing V_{OD} values from the transmit V_{OD} control registers of the transmitter portion of a specific channel controlled by the ALTGX_RECONFIG instance, perform the following steps:

1. Set the `logical_channel_address` input port to the logical channel address of the transceiver channel whose PMA controls you want to read (for example, `tx_vodctrl_out`).
2. Set the `rx_tx_duplex_sel` port to **2'b10** so that only the transmit PMA controls are read from the transceiver channel.
3. Ensure that the `busy` signal is low before you start a read transaction.
4. Assert the read signal for one `reconfig_clk` clock cycle. This initiates the read transaction.

The `busy` output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy reading the PMA control values. When the read transaction has completed, the `busy` signal goes low. The `data_valid` signal is asserted to indicate that the data available at the read control signal is valid.

Figure 17 shows the read transaction waveform for Method 1.

Figure 17. Read Transaction Waveform—Use 'logical_channel_address port' Option



Notes to Figure 17:

(1) In this waveform example, you want to read from the transmitter portion of the channel only.

(2) In this waveform example, the number of channels connected to the dynamic reconfiguration controller is four. Therefore, the `logical_channel_address` port is 2 bits wide.

Simultaneous write and read transactions are not allowed.

Method 2: Writing the Same Control Signals to Control All the Transceiver Channels

Method 2 does not require the `logical_channel_address` port. The PMA controls of all the transceiver channels connected to the ALTGX_RECONFIG instance are reconfigured.

The **Use the same control signal for all the channels** option is available on the **Analog controls** tab of the ALTGX_RECONFIG MegaWizard Plug-In Manager. If you enable this option, the width of the PMA control ports are fixed as follows:

PMA Control Ports Used in a Write Transaction

- `tx_vodctrl` is fixed to 3 bits
- `tx_preemp` is fixed to 5 bits
- `rx_eqdcgain` is fixed to 3 bits
- `rx_eqctrl` is fixed to 4 bits

PMA Control Ports Used in a Read Transaction

- tx_vodctrl_out is 3 bits per channel
- tx_preemp_out is 5 bits per channel
- rx_eqdcgain_out is 3 bits per channel
- rx_eqctrl_out is 4 bits per channel

For example, assume the number of channels controlled by the dynamic reconfiguration controller is two, tx_vodctrl_out is 6 bits wide.

Write Transaction

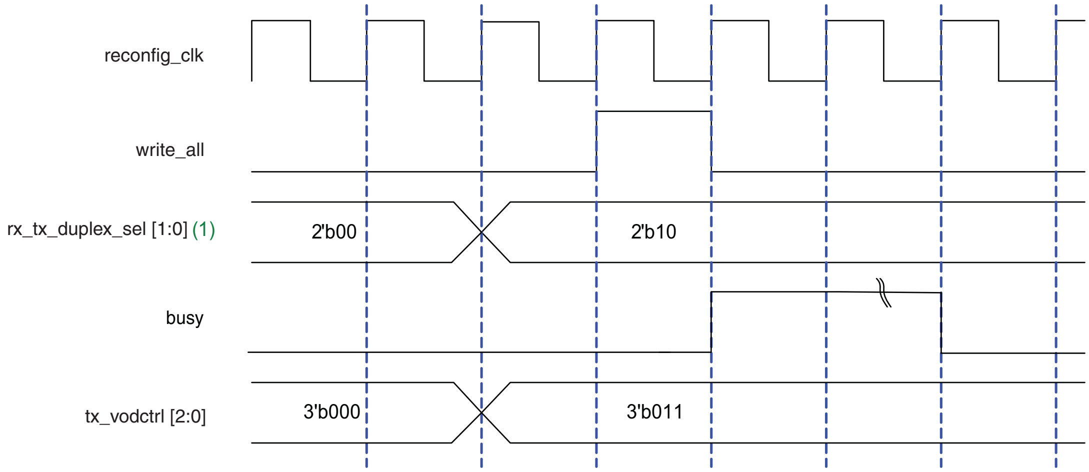
The value you set at the selected PMA control ports is written to all the transceiver channels connected to the ALTGX_RECONFIG instance.

For example, assume you have enabled tx_vodctrl in the ALTGX_RECONFIG MegaWizard Plug-In Manager to reconfigure the V_{OD} of the transceiver channels. To complete a write transaction to reconfigure the V_{OD}, perform the following steps:

1. Before you initiate a write transaction, set the selected PMA control ports to the desired settings (for example, tx_vodctrl = **3'b000**).
2. Set the rx_tx_duplex_sel port to **2'b10** so that only the transmit PMA controls are written to the transceiver channel.
3. Ensure that the busy signal is low before you start a write transaction.
4. Assert the write_all signal for one reconfig_clk clock cycle. This initiates the write transaction.
5. The busy output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy writing the PMA control values. When the write transaction has completed, the busy signal goes low.

Figure 18 shows the write transaction for Method 2.

Figure 18. Write Transaction Waveform—Use the same control signal for all the channels Option



Note to Figure 18:

(1) In this waveform example, you want to write to the transmitter portion of the channel only.

Read Transaction

If you want to read the existing values from a specific channel connected to the ALTGX_RECONFIG instance, observe the corresponding byte positions of the PMA control output port after the read transaction is completed.

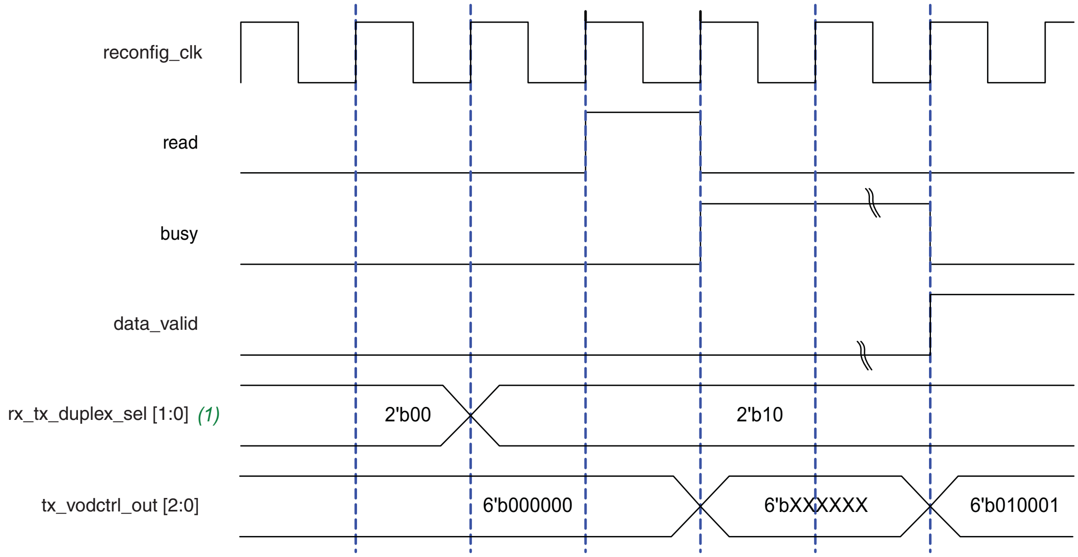
For example, if the number of channels controlled by the ALTGX_RECONFIG is two, the tx_vodctrl_out is 6 bits wide. The tx_vodctrl_out[2:0] signal corresponds to channel 1 and the tx_vodctrl_out[5:3] signal corresponds to channel 2.

To complete a read transaction to the V_{OD} values of the second channel, perform the following steps:

1. Before you initiate a read transaction, set the rx_tx_duplex_sel port to **2'b10** so that only the transmit PMA controls are read from the transceiver channel.
2. Ensure that the busy signal is low before you start a read transaction.
3. Assert the read signal for one reconfig_clk clock cycle. This initiates the read transaction.
4. The busy output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy reading the PMA control settings.
5. When the read transaction has completed, the busy signal goes low. The data_valid signal is asserted, indicating that the data available at the read control signal is valid.
6. To read the current V_{OD} values in channel 2, observe the values in tx_vodctrl_out[5:3].

In the waveform example shown in Figure 19, the transmit V_{OD} settings written in channels 1 and 2 prior to the read transaction are 3'b001 and 3'b010, respectively.

Figure 19. Read Transaction Waveform—Use the same control signal for all the channels Option Enabled



Note to Figure 19:

(1) In this waveform example, you want to read from the transmitter portion of all the channels only.

☞ Simultaneous write and read transactions are not allowed.

Method 3: Writing Different Control Signals for all the Transceiver Channels at the Same Time

If you disable the **Use the same control signal for all the channels** option, the PMA control ports for a write transaction are separate for each channel. If you disable this option, the width of the PMA control ports are fixed as follows:

PMA Control Ports Used in a Write Transaction

- `tx_vodctrl` is 3 bits per channel
- `tx_preemp` are 5 bits per channel
- `rx_eqdcgain` is 3 bits per channel
- `rx_eqctrl` is 4 bits per channel

For example, if you have two channels, the `tx_vodctrl` is 6 bits wide (`tx_vodctrl [2:0]` corresponds to channel 1 and `tx_vodctrl [5:3]` corresponds to channel 2).

PMA Control Ports Used in a Read Transaction

The width of the PMA control ports for a read transaction are always separate for each channel (refer to "Connecting the PMA Control Ports" on page 19).

Write Transaction

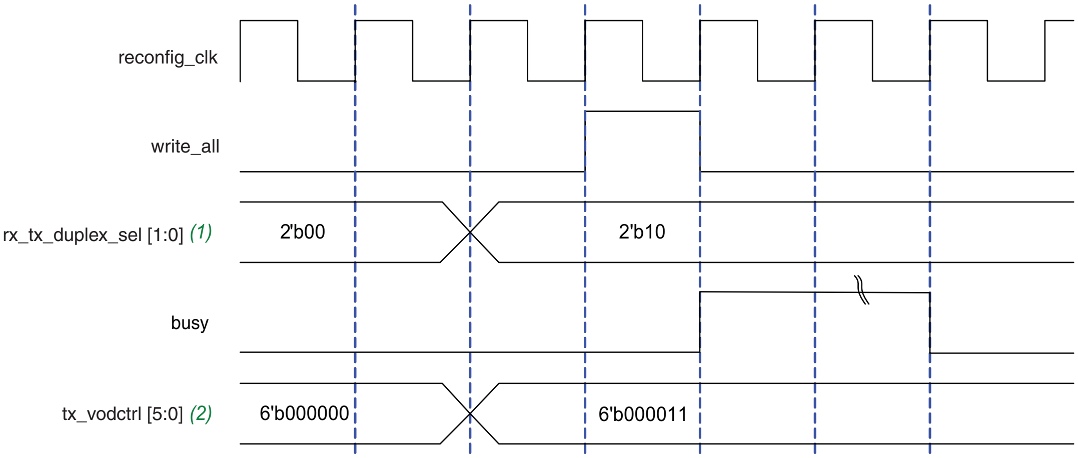
Because the PMA controls of all the channels are written, if you want to reconfigure a specific channel connected to the ALTGX_RECONFIG instance, set the new value at the corresponding PMA control port of the channel under consideration and retain the previously stored values in the other active channels with a read transaction prior to this write transaction.

For example, if the number of channels controlled by the ALTGX_RECONFIG instance is two, the `tx_vodctrl` signal in this case would be 6 bits wide. The `tx_vodctrl[2:0]` signal corresponds to channel 1 and the tx`_vodctrl[5:3]` signal corresponds to channel 2.

- To dynamically reconfigure the PMA controls of only channel 2 with a new value, first perform a read transaction to retrieve the existing PMA control values from `tx_vodctrl_out[5:0]`. Use the `tx_vodctrl_out[2:0]` value for `tx_vodctrl[2:0]` to write in channel 1. By doing so, channel 1 is overwritten with the same value.
- Perform a write transaction. This ensures that the new values are written only to channel 2 while channel 1 remains unchanged.

Figure 20 shows a write transaction waveform with the **Use the same control signal for all the channels** option disabled.

Figure 20. Write Transaction Waveform—Use the same control signal for all the channels Option Disabled



Notes to Figure 20:

(1) In this waveform example, you want to write to the transmitter portion of the channel only.

(2) In this waveform example, the number of channels controlled by the dynamic reconfiguration controller (the ALTGX_RECONFIG instance) is two and the `tx_vodctrl` control port is enabled.

☞ Simultaneous write and read transactions are not allowed.

Read Transaction

The read transaction is described in "Read Transaction" on page 20.

Design Example with Analog Control

The following example describes one possible topology with one ALTGX_RECONFIG instance to control multiple ALTGX instances using method 1 (refer to "Method 1: Using logical_channel_address to Reconfigure Specific Transceiver Channels" on page 19).

One Reconfiguration Controller Connected to Multiple ALTGX Instances

For example, assume you have the following scenario:

- ALTGX_RECONFIG instance
- ALTGX instance 1 with five channels
- ALTGX instance 2 with three channels

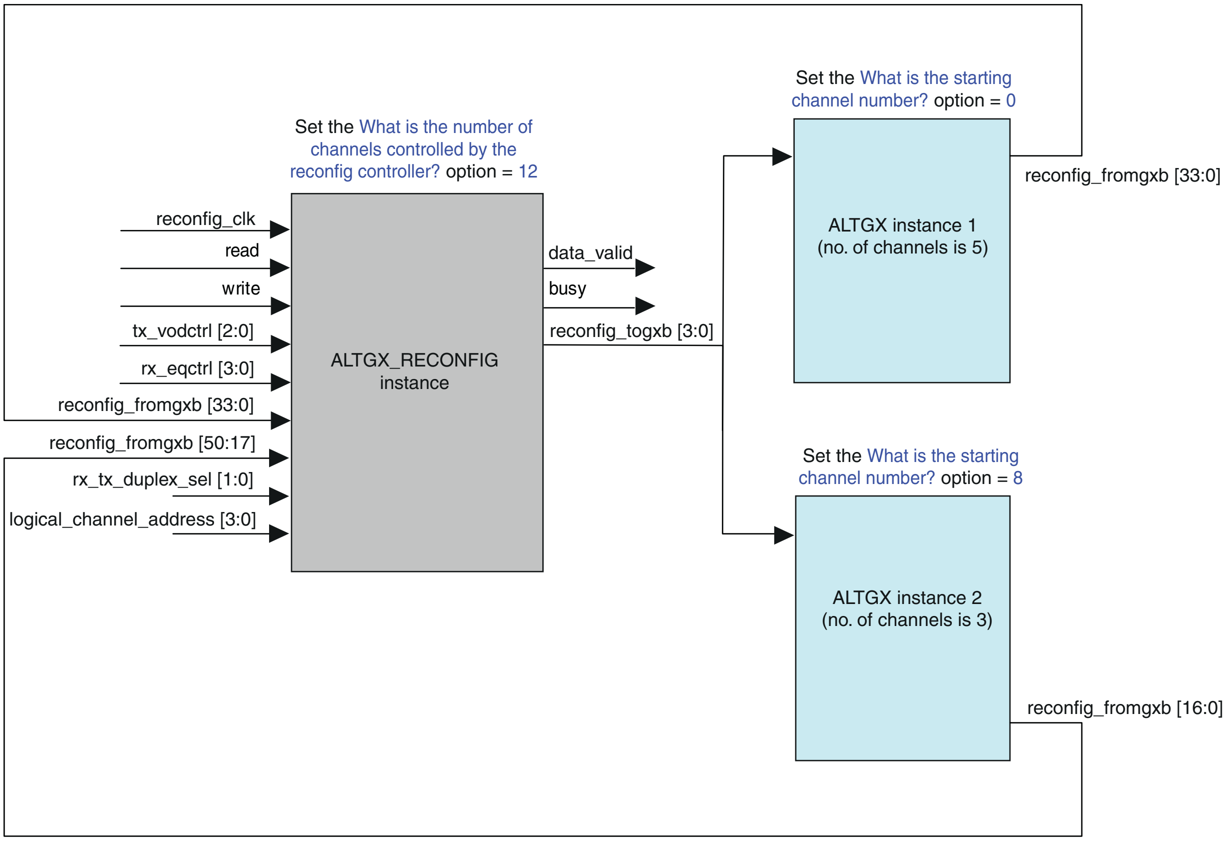
For this example, assume the following:

- ALTGX instance 1 and ALTGX instance 2 cannot be physically packed into the same transceiver block.
- One dynamic reconfiguration controller controls both the ALTGX instances.

- You want to dynamically reconfigure the transmit V_{OD} PMA control (`tx_vodctrl`) of the first channel of ALTGX instance 1 and receiver equalization PMA control (`rx_eqctrl`) of the second channel of ALTGX instance 2.
- You are using the logical channel addressing feature in the ALTGX megafunction.

Figure 21 shows the ALTGX_RECONFIG instance connected to both ALTGX instance 1 and ALTGX instance 2.

Figure 21. Example 1 for PMA Controls Reconfiguration


Set the What is the number of channels controlled by the reconfig controller? option = 12
reconfig_clk
read
write
tx_vodctrl [2:0]
rx_eqctrl [3:0]
reconfig_fromgxb [33:0]
reconfig_fromgxb [50:17]
rx_tx_duplex_sel [1:0]
logical_channel_address [3:0]
ALTGX_RECONFIG instance
data_valid
busy
reconfig_togxb [3:0]
Set the What is the starting channel number? option = 0
ALTGX instance 1 (no. of channels is 5)
reconfig_fromgxb [33:0]
Set the What is the starting channel number? option = 8
ALTGX instance 2 (no. of channels is 3)
reconfig_fromgxb [16:0]

The following are the typical steps that help set up the dynamic reconfiguration process.

Setting the ALTGX Instances

1. Set up the logical channel addressing for both ALTGX instances.
2. Enable the **Analog controls** (**VOD**, **pre-emphasis**, and **manual equalization**) option on the **Reconfig** tab of the ALTGX MegaWizard Plug-In Manager.

Setting the ALTGX_RECONFIG Instance

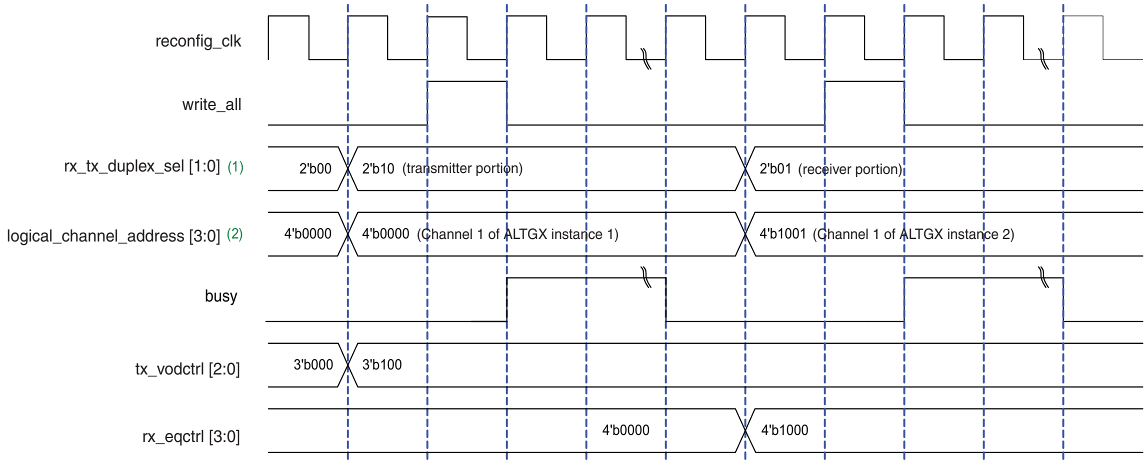
3. Set up the logical channel control for the ALTGX_RECONFIG instance.
4. Select the `rx_tx_duplex_sel [1:0]` port on the **Error checks/Data rate switch** tab of the ALTGX_RECONFIG MegaWizard Plug-In Manager.
5. Select the `tx_vodctrl` and `rx_eqctrl` controls. `tx_vodctrl` is 3 bits wide and `rx_eqctrl` is 4 bits wide.

ALTGX Instances and ALTGX_RECONFIG Instance Connections

To connect the ALTGX instances to the ALTGX_RECONFIG instance, refer to the steps described in "Stage III: Connect the ALTGX_RECONFIG with the ALTGX Instances" on page 12.

Figure 22 shows the write transaction waveform for this example.

Figure 22. Write Transaction Waveform


reconfig_clk
write_all
rx_tx_duplex_sel [1:0] (1)
2'b00
2'b10 (transmitter portion)
2'b01 (receiver portion)
logical_channel_address [3:0] (2)
4'b0000
4'b0000 (Channel 1 of ALTGX instance 1)
4'b1001 (Channel 1 of ALTGX instance 2)
busy
tx_vodctrl [2:0]
3'b000
3'b100
rx_eqctrl [3:0]
4'b0000
4'b1000

Error Indication of Analog Control Reconfiguration

The ALTGX_RECONFIG MegaWizard Plug-In Manager provides an error status signal when you select the **Enable illegal mode checking** option or the **Enable self recovery** option on the **Error checks/data rate switch** tab. The conditions under which the error signal is asserted are:

- **Enable illegal mode checking** option—When you select this option, the dynamic reconfiguration controller checks whether an attempted operation falls under one of the following conditions. The dynamic reconfiguration controller detects these conditions within two `reconfig_clk` cycles, deasserts the `busy` signal, and asserts the `error` signal for two `reconfig_clk` cycles.
  - PMA controls, read operation—None of the output ports (`rx_eqctrl_out`, `rx_eqdcgain_out`, `tx_vodctrl_out`, and `tx_preemp_out`) are selected in the ALTGX_RECONFIG instance.

    The `read` signal is asserted.
  - PMA controls, write operation—None of the input ports (`rx_eqctrl`, `rx_eqdcgain`, `tx_vodctrl`, and `tx_preemp`) are selected in the ALTGX_RECONFIG instance.

    The `write_all` signal is asserted.
- **Enable self recovery** option—When you select this option, the ALTGX_RECONFIG MegaWizard Plug-In Manager provides the error output port. The dynamic reconfiguration controller quits an operation if it did not complete within the expected number of clock cycles. After recovering from the illegal operation, the dynamic reconfiguration controller deasserts the `busy` signal and asserts the `error output` port for two `reconfig_clk` cycles.

Data Rate Division in TX Operation

The following sections describe the steps involved in a write and read transaction for data rate division in TX mode.

Data Rate Division in TX—Write Transaction

To complete a write transaction in data rate division in TX mode, perform the following steps:

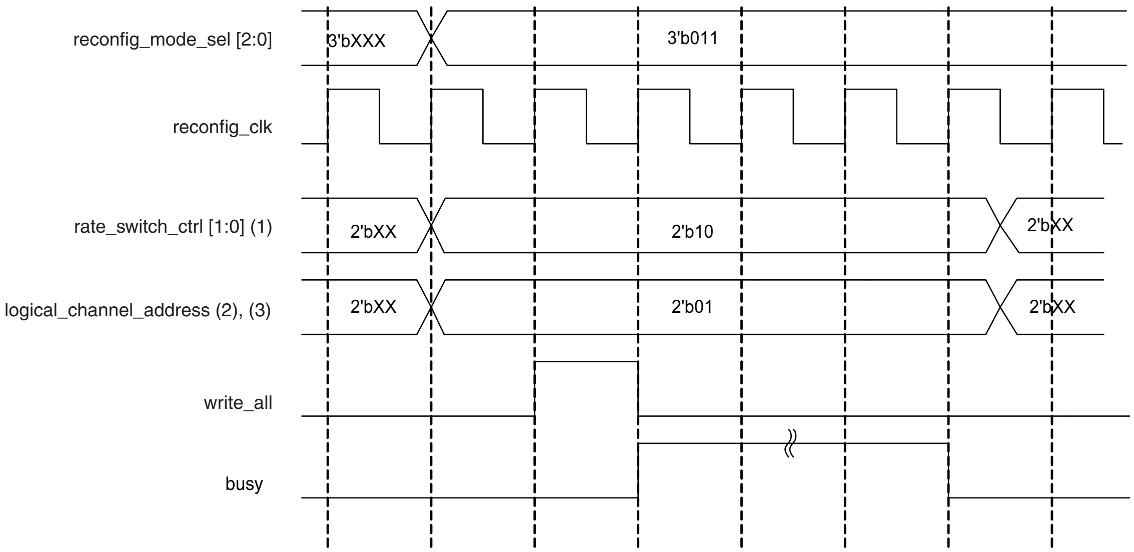
1. Set the `reconfig_mode_sel [2:0]` signal to **011** to activate this mode.
2. Set the `rate_switch_ctrl [1:0]` signal to the corresponding TX local divider setting.
3. Set the `logical_channel_address` port to the logical channel address of the transmitter channel whose local divider settings you want to reconfigure.
4. Ensure that the `busy` signal is low.
5. Initiate a write transaction by asserting the `write_all` signal for one `reconfig_clk` cycle.

After performing the `rate_switch_ctrl` reconfiguration, always perform the subsequent reconfiguration twice.

You must be aware of both the device operating range and the compatible data rate of a protocol before you enable this feature. There are no legal checks that are imposed by the Quartus II software because data rate division in TX mode is an on-the-fly control feature.

Figure 23 shows a write transaction waveform in this mode.

Figure 23. Data Rate Division in TX—Write Transaction



Notes to Figure 23:

(1) In this waveform example, you want to reconfigure the local divider settings of the transmitter channel to **Divide by 4**. Therefore, the value set at `rate_switch_ctrl [1:0]` is **2'b10**.

(2) In this waveform example, the value set in the **What is the number of channels controlled by the reconfig controller?** option of the ALTGX_RECONFIG MegaWizard Plug-In Manager is **4**. Therefore, the `logical_channel_address` input is 2 bits wide.

(3) In this waveform example, you want to reconfigure the local divider settings of the transmitter channel whose logical channel address is **2'b01**.

Data Rate Division in TX—Read Transaction

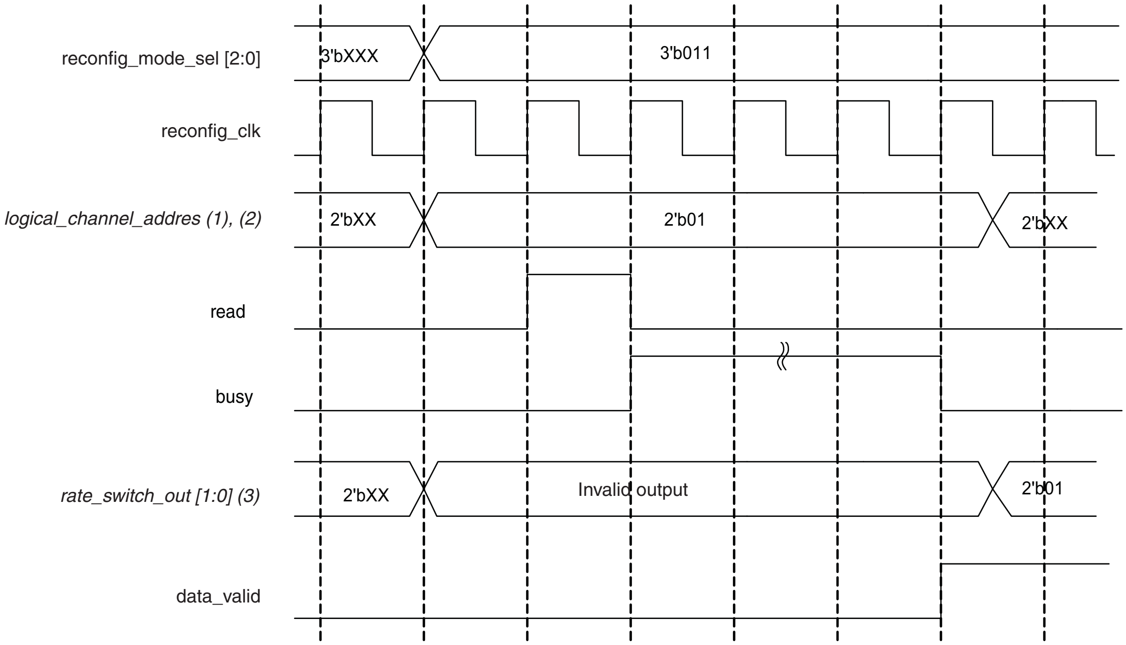
To complete a read transaction in data rate division in TX mode, perform the following steps:

1. Set the `reconfig_mode_sel [2:0]` signal to **011** to activate this mode.
2. Select the `rate_switch_out [1:0]` to read out the existing TX local divider settings.
3. Set the `logical_channel_address` port to the logical channel address of the transmitter channel whose local divider settings you want to read.
4. Ensure that the `busy` signal is low.
5. Assert the `read` signal for one `reconfig_clk` cycle.

If you want to read the existing local divider settings of the transmitter channel, select the **Use 'rate_switch_out' port** option to read out the current data rate division option on the **Error checks/Data rate switch** tab in the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Figure 24 shows a read transaction waveform in this mode.

Figure 24. Data Rate Division in TX—Read Transaction



Notes to Figure 24:

(1) In this waveform example, the existing local divider settings of the transmitter channel are **Divide by 2**. Therefore, the value read out at `rate_switch_out [1:0]` is **2'b01**.

(2) In this waveform example, the value set in the **What is the number of channels controlled by the reconfig controller?** option of the ALTGX_RECONFIG MegaWizard Plug-In Manager is **4**. Therefore, the `logical_channel_address` input is 2 bits wide.

(3) In this waveform example, you want to read the existing local divider settings of the transmitter channel whose logical channel address is **2'b01**.

☞ Do not perform a read transaction in date rate division in TX mode if you did not select `rate_switch_out [1:0]` in the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Channel and TX PLL select/reconfig

All sub-modes under channel and TX PLL select/reconfig mode share the same write transactions. The only differences are the reconfiguration mode you specify in `reconfig_mode_sel` and the reconfiguration duration. For more information, refer to "Dynamic Reconfiguration Duration" on page 78.

Central control unit reconfiguration requires a two-step process. For more information, refer to "Central Control Unit Reconfiguration Mode" on page 69.

In channel and TX PLL select/reconfig mode, only a write transaction can occur—no read transactions are allowed.

To complete a write transaction, perform the following steps:

1. Set the reconfig_mode_sel[2:0] control signal to use the channel reconfiguration feature.
2. Set the rx_tx_duplex_sel port to enable the transmitter or receiver or the receiver and transmitter portion for reconfiguration.
3. Set the logical_channel_address port to specify the logical channel address of the transceiver channel.
4. Ensure the busy signal is low and assert the write_all signal for one reconfig_clk clock cycle.
5. Wait for a
   - low on the busy signal indicates the transaction is complete.
   - high on the reconfig_address_en signal indicates the system is ready to receive the next **.mif** word.
6. Provide the next **.mif** word indicated by the reconfig_address_out port.
7. Repeat steps 4–6 until the channel_reconfig_done signal goes high.

You can optionally choose to trigger write_all once by selecting the continuous write operation in the ALTGX_RECONFIG MegaWizard Plug-In Manager. The Quartus II software then continuously writes all words required for reconfiguration.

The following section provides an example of a CMU PLL reconfiguration.

CMU PLL Reconfiguration Example

To complete a CMU PLL reconfiguration, perform the following steps:

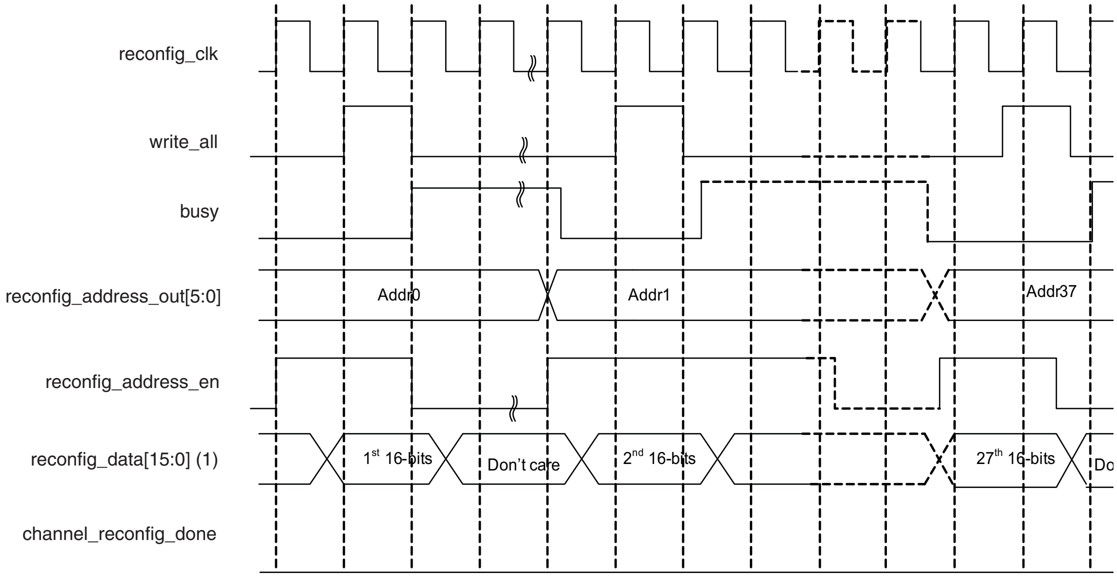
1. Set the reconfig_mode_sel [2:0] signal to **100** to activate this mode.
2. Ensure that the busy signal is low.
3. Initiate a write transaction by asserting the write_all signal for one reconfig_clk cycle to write the first 16-bit word of the **.mif**. Similarly, initiate a write transaction to write all the words of the **.mif**. Use the reconfig_address_out_en port to determine when to initiate the next write transaction.

The dynamic reconfiguration controller asserts the busy signal for every write transaction you initiate. The busy signal remains asserted until the complete 16-bit word has been written.

The dynamic reconfiguration controller automatically increments the values on the reconfig_address_out port. During reconfiguration, the dynamic reconfiguration controller powers down the CMU PLL until new values are written.

Figure 25 shows the timing waveform of the CMU PLL reconfiguration operation.

Figure 25. CMU PLL Reconfiguration .mif Write Transaction with a Duplex Channel



The dynamic reconfiguration controller asserts the `channel_reconfig_done` signal to indicate that the CMU PLL reconfiguration is complete. The `logical_channel_address` port is not applicable in CMU PLL reconfiguration mode, even though it is available as an input.

During CMU PLL reconfiguration, the ALTGX_RECONFIG instance automatically powers down only the selected CMU PLL, not the other CMU PLL, until it completes reconfiguring the selected CMU PLL. The ALTGX_RECONFIG instance does not provide external ports to control the CMU PLL power down. When you reconfigure the CMU PLL, the `pll_locked` signal goes low. Therefore, after reconfiguring the transceiver, wait for the `pll_locked` signal from the ALTGX instance before continuing normal operation.

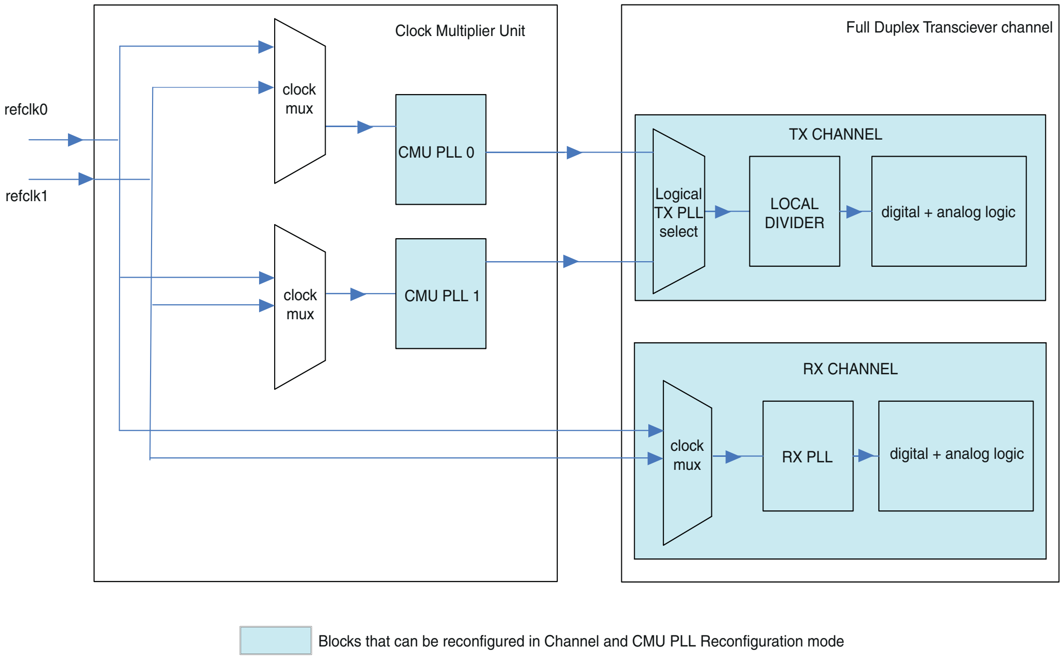
Design Example with CMU PLL Reconfiguration

For example, consider the following scenario:

- The design has four **Transmitter-only** ALTGX instances in the same transceiver block.
- All the four instances are configured in Basic functional mode and have a 2.5 gigabits per second (Gbps) data rate.
- All four channels can listen to `CMU PLL 0`.
- The input reference clock used by `CMU PLL 0` is 100 MHz.
- You want to reconfigure all the four channels identically to 2 Gbps together.

Figure 26 shows this example scenario before and after dynamic reconfiguration.

Figure 26. CMU PLL Reconfiguration Example



You can achieve the CMU PLL reconfiguration by reconfiguring the `CMU PLL 0` once to run for 2 Gbps. This, in turn, changes the transmit data rate of the four channels listening to this `CMU PLL 0`.

Table 3 lists CMU PLL reconfiguration examples for the ALTGX instances and an ALTGX_RECONFIG instance.

Table 3. CMU PLL Reconfiguration Example (Part 1 of 2)

ALTGX Instances		ALTGX_RECONFIG Instance	
ALTGX Setting	**Four TX Only Instances**	**ALTGX_RECONFIG Setting**	**ALTGX_RECONFIG Instance**
What is the effective data rate? option	2 Gbps	**Channel and TX PLL select/reconfig** option	Enabled
Enable Channel and Transmitter PLL Reconfiguration option	Enabled	Use the **reconfig_address_out** option	Enabled

Table 3. CMU PLL Reconfiguration Example (Part 2 of 2)

ALTGX Instances		ALTGX_RECONFIG Instance	
ALTGX Setting	**Four TX Only Instances**	**ALTGX_RECONFIG Setting**	**ALTGX_RECONFIG Instance**
What is the main transmitter PLL reference index? option	1	Use the **reconfig_address_out** option	Enabled
How many input clocks? option	2		
What is the selected input clock source for the Transmitter PLL and Receiver PLL? option	1		
What is clock 0 in-put frequency? option	125 MHz		
What is clock 1 in-put frequency? option	100 MHz		

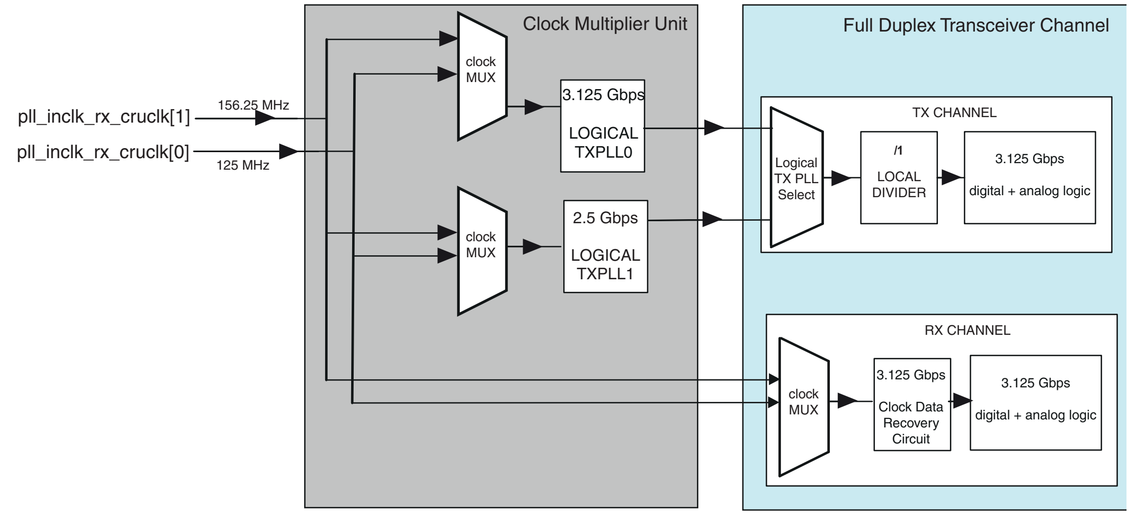
After generating the **.mif**, follow the steps listed in **"CMU PLL Reconfiguration Example" on page 31**.

Channel and CMU PLL Reconfiguration Example

Figure 27 has the following configuration:

- The full-duplex channel with the main CMU PLL configured to 3.125 Gbps with a 156.25 MHz reference clock.
- The alternate CMU PLL is configured to 2.5 Gbps with a 125 MHz reference clock.
- Assume that the logical `tx pll` value is set to **0** for the main CMU PLL.

Figure 27. Channel and CMU PLL Reconfiguration in a Transceiver Block



For example, assume you want to switch to the following two modes:

Mode1:

- Full-duplex channel with the main CMU PLL configured to 3.125 Gbps data with a 156.25 MHz reference clock.
- Assume that the logical `tx pll` is set to **0** for the main CMU PLL and rate matcher is not enabled in the ALTGX MegaWizard Plug-In Manager.
- The alternate CMU PLL is configured to 2.5 Gbps with a 125 MHz reference clock.

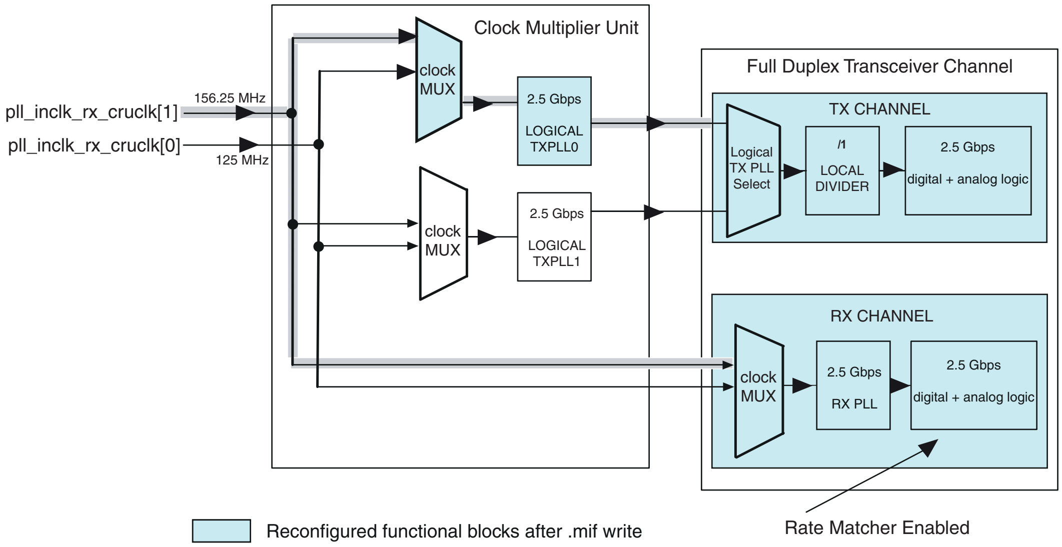
Mode2:

- Full-duplex channel with the main CMU PLL configured to 2.5 Gbps data with a 156.25 MHz reference clock.
- Assume that the logical `tx pll` is set to **0** for the main CMU PLL.
- Rate matcher is enabled in the ALTGXB MegaWizard Plug-In Manager.
- The alternate CMU PLL is configured to 2.5 Gbps using a 125 MHz reference clock.

The **.mif** is generated for Mode1 and Mode2.

Figure 28 shows the functional blocks that are reconfigured after the dynamic reconfiguration controller writes the mode2 **.mif**.

Figure 28. Reconfigured Functional Blocks after the Channel and TX PLL Reconfiguration



On the receive side, the rate matcher is enabled after reconfiguration because the Mode2 **.mif** contains settings to enable the rate matcher block.

Dynamic Reconfiguration Design Considerations

This section describes important design considerations.

General

Be aware of the following general considerations:

- Only one ALTGX_RECONFIG instance can connect to one transceiver block.
- Unsupported features:
  - Enable and disable PRBS or BIST.
  - Switching from a **Receiver-only** channel to a **Transmitter-only** channel.
  - Switching between bonded x4 modes and bonded x8 modes.
- Do not run or reconfigure the ALTGX instance when the `busy` signal is high.
- You must assert the `write_all` and `read` signals for only one cycle in all reconfiguration modes.
- After reconfiguration, the new channel settings will remain even after `gxb_powerdown` or `pll_powerdown` is asserted.

Offset Cancellation for Receiver Channels

Be aware of the following considerations for offset cancellation for the receiver channels:

- Offset cancellation is required and enabled by default whenever a design consists of receiver channels.
- The offset cancellation process changes the transceiver reset sequence.
- Offset cancellation is done automatically regardless of user inputs to the ALTGX_RECONFIG instances.
- During offset cancellation, you must not assert the `gxb_powerdown` signal.
- You must instantiate and connect all the ports of ALTGX_RECONFIG instances to the ALTGX instances.

PMA Analog Control

PMA analog control has the following limitation:

- Simultaneous write and read transactions are not allowed.

Data Rate Division in TX

Be aware of the following considerations for data rate division in TX mode:

- Only modify the TX data rate, not the RX data rate.
- Ensure that the new TX data rate is within device and protocol specifications.
- Not applicable for bonded channels that do not use TX local dividers.

- A read transaction is available only if you select `rate_switch_out [1:0]` in the ALTGX_RECONFIG MegaWizard Plug-In Manager.
- There are no legal checks by the Quartus II software to verify the device operating range and the compatible data rate of a protocol before you enable this feature.

Channel and TX PLL Select/Reconfig

Be aware of the following considerations for channel and TX PLL select/reconfig mode:

- The **.mif** generated after compilation contains protocol settings on the **Parameter Settings** tab.
- Channel and TX PLL select/reconfig mode does not allow a read operation to read the settings of an ALTGX instance.
- Each **.mif** contains only one setting of an ALTGX instance, which may contain multiple channels.
- Main and Alt PLL logical reference indexes must remain consistent in all **.mif** files.
- After reconfiguring the transceiver, wait for the `pll_locked` signal from the ALTGX instance before continuing normal operation.

CMU Reconfiguration

Be aware of the following considerations for CMU reconfiguration:

- Reconfiguring the CMU PLL affects all the channels connected to it.
- CMU reconfiguration mode powers down only the selected CMU PLL.
- The `logical_channel_address` port is not applicable in CMU PLL reconfiguration mode, even though the `logical_channel_address` port is available as an input.
- The values at `logical_tx_pll_sel` and `logical_tx_pll_sel_en` must be held at a constant logic level until reconfiguration is completed.

Channel Reconfiguration with TX PLL Select Mode

Be aware of the following consideration for channel reconfiguration with TX PLL select mode:

- The channel reconfiguration with TX PLL select mode reconfigures only the channel setting and the TX PLL multiplexer without changing TX PLL parameter settings.

Port Definition

Table 4 lists the input control ports and output status ports of the dynamic reconfiguration controller for Arria II GX devices.

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 1 of 8)

Port Name	Input/ Output	Description
Clock Inputs to ALTGX_RECONFIG Instance		
`reconfig_clk`	Input	The frequency range of the `reconfig_clk` clock depends on the following transceiver channel configuration modes: ■ **Receiver only** (37.5 MHz to 50 MHz) ■ **Receiver and Transmitter** (37.5 MHz to 50 MHz) ■ **Transmitter only** (2.5 MHz to 50 MHz) By default, the Quartus II software assigns a global clock resource to this port. The `reconfig_clk` clock must be a free-running clock sourced from an I/O clock pin. Do not use dedicated transceiver REFCLK pins or any clocks generated by transceivers. NOTE: If you want to source the `reconfig_clk` port from a non-free running clock source, there are additional requirements. For more information, refer to "Using a GPLL to Drive the reconfig_clk Port" on page 13.
ALTGX and ALTGX_RECONFIG Interface Signals		
`reconfig_fromgxb`	Input	An output port in the ALTGX instance and an input port in the ALTGX_RECONFIG instance. The `reconfig_fromgxb` signal is transceiver-block based. Therefore, the width of this signal increases in steps of 17 bits per transceiver block. In the ALTGX MegaWizard Plug-In Manager, the width of the `reconfig_fromgxb` signal depends on the following: ■ Whether the channels configured in the ALTGX instance are regular transceiver channels or PMA-only channels. ■ The number of channels you select in the **What is the number of channels?** option in the **General** screen. For example, if the channels in the ALTGX instance are regular transceiver channels and if you select the number of channels as follows: 1 ≤ Channels ≤ 4, then the output port `reconfig_fromgxb` = 17 bits 5 ≤ Channels ≤ 8, then the output port `reconfig_fromgxb` = 34 bits 9 ≤ Channels ≤ 12, then the output port `reconfig_fromgxb` = 51 bits However, if the channels in the ALTGX instance are PMA-only channels and if you select the number of channels as follows: Number of PMA-only channels = n, then the output port `reconfig_fromgxb` = n*17 bits For example, `reconfig_fromgxb` = 6 * 17 bits for 6 PMA-only channels. In the ALTGX_RECONFIG MegaWizard Plug-In Manager, the width of the `reconfig_fromgxb` signal depends on the value you select in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen. For example, if you select the total number of channels controlled by ALTGX_RECONFIG instance as follows: 1 ≤ Channels ≤ 4, then the input port `reconfig_fromgxb` = 17 bits 5 ≤ Channels ≤ 8, then the input port `reconfig_fromgxb` = 34 bits 9 ≤ Channels ≤ 12, then the input port `reconfig_fromgxb` = 51 bits

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 2 of 8)

Port Name	Input/ Output	Description
`reconfig_fromgxb` (continued)	Input	To connect the `reconfig_fromgxb` port between the ALTGX_RECONFIG instance and multiple ALTGX instances, follow these rules: ■ Connect `reconfig_fromgxb[16:0]` of ALTGX Instance 1 to `reconfig_fromgxb[16:0]` of the ALTGX_RECONFIG instance. Connect the `reconfig_fromgxb[]` port of the next ALTGX instance to the next available bits of the ALTGX_RECONFIG instance and continue until complete. ■ Connect the `reconfig_fromgxb` port of the ALTGX instance, which has the highest **What is the starting channel number?** option, to the MSB of the `reconfig_fromgxb` port of the ALTGX_RECONFIG instance. The Quartus II Fitter produces an error if the dynamic reconfiguration option is enabled in the ALTGX instance but the `reconfig_fromgxb` and `reconfig_togxb` ports are not connected to the ALTGX_RECONFIG instance.
`reconfig_togxb[3:0]`	Output	An input port of the ALTGX instance and an output port of the ALTGX_RECONFIG instance. You must connect the `reconfig_togxb[3:0]` input port of every ALTGX instance controlled by the dynamic reconfiguration controller to the `reconfig_togxb[3:0]` output port of the ALTGX_RECONFIG instance. The width of the `reconfig_togxb[3:0]` port is always fixed to 4 bits.
`reconfig_reset`	Input	An optional signal that you can use to reset the ALTGX_RECONFIG instance. The `reconfig_reset` signal must be held high for at least one clock cycle to take effect. When feeding into the `reconfig_reset` port, the reset signal must be synchronized to the `reconfig_clk` domain. NOTE: If you want to source the `reconfig_clk` port from a non-free running clock source, there are additional requirements. For more information, refer to "Using a GPLL to Drive the reconfig_clk Port" on page 13.
FPGA Fabric and ALTGX_RECONFIG Interface Signals		
`write_all`	Input	Assert this signal for one `reconfig_clk` clock cycle to initiate a write transaction from the ALTGX_RECONFIG instance to the ALTGX instance. You can use the `write_all` signal in two ways for **.mif**-based modes: ■ Continuous write operation—Select the **Enable continuous write of all the words needed for reconfiguration** option to pulse the `write_all` signal only once for writing a whole **.mif**. The **What is the read latency of the MIF contents** option is available for selection in this case only. Enter the desired latency in terms of the `reconfig_clk` cycles. ■ Regular write operation—When you disable the **Enable continuous write of all the words needed for reconfiguration** option, every word of the **.mif** requires its own write cycle. ⚠ CAUTION When the `busy` signal is high, do not assert the `write_all` signal or `read` signal to indicate either a write or read operation. Ensure the `busy` signal is low before asserting the `write_all` signal or `read` signal high (for a write or read instruction).

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 3 of 8)

Port Name	Input/ Output	Description
`busy`	Output	Use this signal to indicate the busy status of the dynamic reconfiguration controller during offset cancellation. After the device powers up, the `busy` signal remains low for the first `reconfig_clk` clock cycle. The `busy` signal then is asserted and remains high when the dynamic reconfiguration controller performs offset cancellation on all the receiver channels connected to the ALTGX_RECONFIG instance. Deassertion of the `busy` signal indicates the successful completion of the offset cancellation process. ■ PMA controls reconfiguration mode—The `busy` signal is high when the dynamic reconfiguration controller performs a read or write transaction. ■ All other dynamic reconfiguration modes—The `busy` signal is high when the dynamic reconfiguration controller writes the **.mif** into the transceiver channel. CAUTION When the `busy` signal is high, do not assert the `write_all` signal or `read` signal to indicate either a write or read operation. Ensure the `busy` signal is low before asserting the `write_all` signal or `read` signal high (for a write or read instruction).
`read`	Input	Assert this signal for one `reconfig_clk` clock cycle to initiate a read transaction. The `read` port is applicable only to the PMA controls reconfiguration mode. The `read` port is available when you select **Analog controls** in the **Reconfiguration settings** screen and select at least one of the PMA control ports in the **Analog controls** screen. CAUTION When the `busy` signal is high, do not assert the `write_all` signal or `read` signal to indicate either a write or read operation. Ensure the `busy` signal is low before asserting the `write_all` signal or `read` signal high (for a write or read instruction).
`data_valid`	Output	Applicable only to PMA controls reconfiguration mode. The `data_valid` port indicates the validity of the data read from the transceiver by the dynamic reconfiguration controller. The current data on the output read ports is the valid data ONLY if `data_valid` is high. The `data_valid` signal is enabled when you enable at least one PMA control port used in read transactions; for example, `tx_vodctrl_out`.
`error`	Output	Indicates that an unsupported operation is attempted. You can select the `error` port in the **Error checks/Data rate switch** screen. The dynamic reconfiguration controller deasserts the `busy` signal and asserts the `error` signal for two `reconfig_clk` cycles when you attempt an unsupported operation.
`logical_channel_address [8:0]`	Input	Enabled by the ALTGX_RECONFIG MegaWizard Plug-In Manager when you enable the **Use 'logical_channel_address' port for Analog controls reconfiguration** option in the **Analog controls** screen. The width of the `logical_channel_address` port depends on the value you set in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen. You can enable the `logical_channel_address` port only when the number of channels controlled by the dynamic reconfiguration controller is more than one.

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 4 of 8)

Port Name	Input/ Output	Description
`rx_tx_duplex_sel[1:0]`	Input	A 2-bit wide signal. You can select the `rx_tx_duplex_sel[1:0]` port in the **Error checks/Data rate switch** screen. The advantage of using the optional `rx_tx_duplex_sel[1:0]` port is that it allows you to reconfigure only the transmitter portion of a channel, even if the channel configuration is duplex. For a setting of: ■ `rx_tx_duplex_sel[1:0]` = 2'b00—the transmitter and receiver portion of the channel is reconfigured. ■ `rx_tx_duplex_sel[1:0]` = 2'b01—the receiver portion of the channel is reconfigured. ■ `rx_tx_duplex_sel[1:0]` = 2'b10—the transmitter portion of the channel is reconfigured.
Analog Settings Control/Status Signals		
`tx_vodctrl[2:0]` [1]	Input	An optional transmit buffer V_{OD} control signal. `tx_vodctrl[2:0]` is 3 bits per transmitter channel. The number of settings varies based on the transmit buffer supply setting and the termination resistor setting on the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager. The width of the `tx_vodctrl[2:0]` signal is fixed to 3 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. The following shows the V_{OD} values corresponding to the `tx_vodctrl` settings for 100-Ω termination. For more information, refer to the “Programmable Output Differential Voltage” section of the *Device Datasheet for Arria II Devices* chapter in volume 3 of the *Arria II Device Handbook*. `tx_vodctrl[2:0]` — Corresponding ALTGX instance settings 3’b000 — N/A 3’b001 — 1 3’b010 — 2 3’b011 — N/A 3’b100 — 4 3’b101 — 5 3’b110 — 6 3’b111 — 7

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 5 of 8)

Port Name	Input/ Output	Description
`tx_preemp[4:0]` (1)	Input	An optional pre-emphasis write control for the first post-tap for the transmit buffer. Depending on what value you set at the `tx_preemp[4:0]` input, the controller dynamically writes the value to the first post-tap control register of the transmit buffer. The width of the `tx_preemp[4:0]` signal is fixed to 5 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. `tx_preemp[4:0]` Corresponding ALTGX instance settings 00000 0 00001 1 00010 2 00011 3 00100 4 00101 5 00110 6 All other values N/A
`rx_eqctrl[3:0]` (1)	Input	An optional write control to write an equalization control value for the receive side of the PMA. The width of the `rx_eqctrl[3:0]` signal is fixed to 4 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. `rx_eqctrl[1:0]` Corresponding ALTGX instance settings 0 0 (Low) 1 1 2 2 3 3 (High) NOTE: The upper 2-bits are unused or reserved.
`rx_eqdcgain[1:0]` (1), (2)	Input	An optional equalizer DC gain write control. The width of the `rx_eqdcgain[1:0]` signal is fixed to 2 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. The following values are the legal settings allowed for this signal: 2'b00 => 0 dB 2'b01 => 3 dB 2'b10 => 6 dB All other values => N/A For more information, refer to the "Programmable Equalization and DC Gain" section of the *Transceiver Architecture in Arria II Devices* chapter in volume 2 of the *Arria II Handbook*.

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 6 of 8)

Port Name	Input/ Output	Description
`tx_vodctrl_out[2:0]`	Output	An optional transmit V_{OD} read control signal. The `tx_vodctrl_out[2:0]` signal reads out the value written into the V_{OD} control register. The width of the `tx_vodctrl_out[2:0]` output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`tx_preemp_out[4:0]`	Output	An optional first post-tap, pre-emphasis read control signal. The `tx_preemp_out[4:0]` signal reads out the value written by its input control signal. The width of the `tx_preemp_out[4:0]` output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`rx_eqctrl_out[3:0]`	Output	An optional read control signal to read the setting of equalization setting of the ALTGX instance. The width of the `rx_eqctrl_out[3:0]` output signal depends on the number of channels controlled by the dynamic reconfiguration controller. NOTE: The upper 2-bits are unused or reserved.
`rx_eqdcgain_out[1:0]`	Output	An optional equalizer DC gain read control signal. The `rx_eqdcgain_out[1:0]` signal reads out the settings of the ALTGX instance DC gain. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
Transceiver Channel Reconfiguration Control/Status Signals		
`reconfig_mode_sel[2:0]`	Input	Set the following values at this signal to activate the appropriate dynamic reconfiguration mode: 3’b000 = PMA controls reconfiguration mode. This is the default value. 3’b011 = data rate division in transmitter mode 3’b100 = CMU PLL reconfiguration mode 3’b101 = channel and CMU PLL reconfiguration mode 3’b110 = channel reconfiguration with transmitter PLL select mode 3’b111 = central control unit reconfiguration mode `reconfig_mode_sel[]` is available as an input only when you enable more than one dynamic reconfiguration mode.
`reconfig_address_out [5:0]`	Output	The `reconfig_address_out[5:0]` signal is always available for you to select in the **Channel and TX PLL reconfiguration** screen. The `reconfig_address_out[5:0]`signal is applicable only in the dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. The `reconfig_address_out[5:0]`signal represents the current address used by the ALTGX_RECONFIG instance when writing the **.mif** into the transceiver channel. The `reconfig_address_out[5:0]` signal increments by 1, from 0 to the last address, then starts at 0 again. You can use this signal to indicate the end of all the **.mif** write transactions (`reconfig_address_out[5:0]` changes from the last address to 0 at the end of all the **.mif** write transactions).

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 7 of 8)

Port Name	Input/ Output	Description
reconfig_address_en	Output	An optional signal you can select in the **Channel and TX PLL reconfiguration** screen. The reconfig_address_en signal is applicable only in dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. The dynamic reconfiguration controller asserts reconfig_address_en to indicate that reconfig_address_out[5:0] has changed. The reconfig_address_en signal is asserted only after the dynamic reconfiguration controller completes writing one 16-bit word of the **.mif**.
reset_reconfig_address	Input	An optional signal you can select in the **Channel and TX PLL reconfiguration** screen. The reset_reconfig_address signal is applicable only in dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. Enable the reset_reconfig_address signal and assert the signal for one reconfig_clk clock cycle if you want to reset the reconfiguration address used by the ALTGX_RECONFIG instance during reconfiguration.
reconfig_data[15:0]	Input	The reconfig_data[15:0] signal is applicable only in the dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. The reconfig_data[15:0] signal is a 16-bit word carrying the reconfiguration information. The signal is stored in a **.mif** that you must generate. The ALTGX_RECONFIG instance requires that you provide reconfig_data [15:0]on every **.mif** write transaction using the write_all signal.
reconfig_address[5:0]	Input	Available for selection only in the **.mif**-based transceiver channel reconfiguration modes.
rate_switch_ctrl[1:0]	Input	Available when you select **data rate division in transmitter** mode. Based on the value you set here, the divide-by setting of the local divider in the transmitter channel is modified. The legal values for this port are: 2'b00 = Divide by 1 2'b01 = Divide by 2 2'b10 = Divide by 4 2'b11 = Not supported
rate_switch_out[1:0]	Input	Available when you select **data rate division in transmitter** mode. You can read the existing local divider settings of a transmitter channel at this port. The decoding for this signal is listed below: 2'b00 = Division of 1 2'b01 = Division of 2 2'b10 = Division of 4 2'b11= Not supported
logical_tx_pll_sel	Input	At this port you specify the identity of the transmitter PLL you want to reconfigure. You can also specify the identity of the transmitter PLL that you want the transceiver channel to listen to. When you enable the logical_tx_pll_sel signal, the value set at this signal overwrites the logical_tx_pll value contained in the **.mif**. The value at the logical_tx_pll_sel port must be held at a constant logic level until reconfiguration is done.

Table 4. Arria II GX Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 8 of 8)

Port Name	Input/ Output	Description
logical_tx_pll_sel_en	Input	If you want to use the logical_tx_pll_sel port only under some conditions and use the logical_tx_pll value contained in the **.mif** otherwise, enable this optional logical_tx_pll_sel_en port. Only when logical_tx_pll_sel_en is enabled and set to **1** does the dynamic reconfiguration controller use logical_tx_pll_sel to identify the transmitter PLL. The value at the logical_tx_pll_sel_en port must be held at a constant logic level until reconfiguration is done.
channel_reconfig_done	Output	This channel_reconfig_done signal goes high to indicate that the dynamic reconfiguration controller has finished writing all the words of the **.mif**. The channel_reconfig_done signal is automatically deasserted at the start of a new dynamic reconfiguration write sequence. The channel_reconfig_done signal is applicable only in channel and CMU PLL reconfiguration and channel reconfiguration with transmitter PLL select modes.

Notes to Table 4:

(1) Not all combinations of input bits are legal values.

(2) In PCIe mode, this input must be tied to 01 to be PCI E-compliant.

Table 5 lists the input control ports and output status ports of the dynamic reconfiguration controller for Arria II GZ devices.

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 1 of 8)

Port Name	Input/ Output	Description
Clock Inputs to ALTGX_RECONFIG Instance		
reconfig_clk	Input	The frequency range of this clock depends on the following transceiver channel configuration modes: ■ **Receiver only** (37.5 MHz to 50 MHz) ■ **Receiver and Transmitter** (37.5 MHz to 50 MHz) ■ **Transmitter only** (2.5 MHz to 50 MHz) By default, the Quartus II software assigns a global clock resource to the reconfig_clk port. The reconfig_clk clock must be a free-running clock sourced from an I/O clock pin. Do not use dedicated transceiver REFCLK pins or any clocks generated by transceivers. NOTE: If you want to source the reconfig_clk port from a non-free running clock source, there are additional requirements. For more information, refer to "Using a GPLL to Drive the reconfig_clk Port" on page 13.

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 2 of 8)

Port Name	Input/ Output	Description
ALTGX and ALTGX_RECONFIG Interface Signals		
reconfig_fromgxb	Input	An output port in the ALTGX instance and an input port in the ALTGX_RECONFIG instance. The reconfig_fromgxb signal is transceiver-block based. Therefore, the width of the reconfig_fromgxb signal increases in steps of 17 bits per transceiver block. In the ALTGX MegaWizard Plug-In Manager, the width of the reconfig_fromgxb signal depends on the following: ■ Whether the channels configured in the ALTGX instance are regular transceiver channels or PMA-only channels. ■ The number of channels you select in the **What is the number of channels?** option in the **General** screen. For example, if the channels in the ALTGX instance are regular transceiver channels and if you select the number of channels as follows: 1 ≤ Channels ≤ 4, then the output port reconfig_fromgxb = 17 bits 5 ≤ Channels ≤ 8, then the output port reconfig_fromgxb = 34 bits 9 ≤ Channels ≤ 12, then the output port reconfig_fromgxb = 51 bits However, if the channels in the ALTGX instance are PMA-only channels and if you select the number of channels as follows: Number of PMA-only channels = n, then the output port reconfig_fromgxb = n*17 bits For example, reconfig_fromgxb = 6 * 17 bits for 6 PMA-only channels. In the ALTGX_RECONFIG MegaWizard Plug-In Manager, the width of this signal depends on the value you select in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen. For example, if you select the total number of channels controlled by ALTGX_RECONFIG instance as follows: 1 ≤ Channels ≤ 4, then the input port reconfig_fromgxb = 17 bits 5 ≤ Channels ≤ 8, then the input port reconfig_fromgxb = 34 bits 9 ≤ Channels ≤ 12, then the input port reconfig_fromgxb = 51 bits
	Input	To connect the reconfig_fromgxb port between the ALTGX_RECONFIG instance and multiple ALTGX instances, follow these rules: ■ Connect reconfig_fromgxb[16:0] of ALTGX Instance 1 to reconfig_fromgxb[16:0] of the ALTGX_RECONFIG instance. Connect the reconfig_fromgxb[] port of the next ALTGX instance to the next available bits of the ALTGX_RECONFIG instance and continue until complete. ■ Connect the reconfig_fromgxb port of the ALTGX instance, which has the highest **What is the starting channel number?** option, to the MSB of the reconfig_fromgxb port of the ALTGX_RECONFIG instance. The Quartus II Fitter produces an error if the dynamic reconfiguration option is enabled in the ALTGX instance but the reconfig_fromgxb and reconfig_togxb ports are not connected to the ALTGX_RECONFIG instance.

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 3 of 8)

Port Name	Input/ Output	Description
reconfig_togxb[3:0]	Output	An input port of the ALTGX instance and an output port of the ALTGX_RECONFIG instance. You must connect the reconfig_togxb[3:0] input port of every ALTGX instance controlled by the dynamic reconfiguration controller to the reconfig_togxb[3:0] output port of the ALTGX_RECONFIG instance. The width of the reconfig_togxb[3:0] port is always fixed to 3 bits.
reconfig_reset	Input	Optional signal that you can use to reset the ALTGX_RECONFIG instance. The reconfig_reset signal must be held high for at least one clock cycle to take effect. When feeding into the reconfig_reset port, the reset signal must be synchronized to the reconfig_clk domain. NOTE: If you want to source the reconfig_clk port from a non-free running clock source, there are additional requirements. For more information, refer to "Using a GPLL to Drive the reconfig_clk Port" on page 13.
FPGA Fabric and ALTGX_RECONFIG Interface Signals		
write_all	Input	Assert this signal for one reconfig_clk clock cycle to initiate a write transaction from the ALTGX_RECONFIG instance to the ALTGX instance. You can use the write_all signal in two ways for **.mif**-based modes: ■ Continuous write operation—Select the **Enable continuous write of all the words needed for reconfiguration** option to pulse the write_all signal only once for writing a whole **.mif**. The **What is the read latency of the MIF contents** option is available for selection in this case only. Enter the desired latency in terms of the reconfig_clk cycles. ■ Regular write operation—When you disable the **Enable continuous write of all the words needed for reconfiguration** option, every word of the **.mif** requires its own write cycle. CAUTION When the busy signal is high, do not assert the write_all signal or read signal to indicate either a write or read operation. Ensure the busy signal is low before asserting the write_all signal or read signal high (for a write or read instruction).
busy	Output	Use this signal to indicate the busy status of the dynamic reconfiguration controller during offset cancellation. After the device powers up, the busy signal remains low for the first reconfig_clk clock cycle. The signal then is asserted and remains high when the dynamic reconfiguration controller performs offset cancellation on all the receiver channels connected to the ALTGX_RECONFIG instance. Deassertion of the busy signal indicates the successful completion of the offset cancellation process. ■ PMA controls reconfiguration mode—The busy signal is high when the dynamic reconfiguration controller performs a read or write transaction. ■ All other dynamic reconfiguration modes—The busy signal is high when the dynamic reconfiguration controller writes the **.mif** into the transceiver channel. CAUTION When the busy signal is high, do not assert the write_all signal or read signal to indicate either a write or read operation. Ensure the busy signal is low before asserting the write_all signal or read signal high (for a write or read instruction).

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 4 of 8)

Port Name	Input/ Output	Description
read	Input	Assert this signal for one reconfig_clk clock cycle to initiate a read transaction. The read port is applicable only to the PMA controls reconfiguration mode. The read port is available when you select **Analog controls** in the **Reconfiguration settings** screen and select at least one of the PMA control ports in the **Analog controls** screen. CAUTION When the busy signal is high, do not assert the write_all signal or read signal to indicate either a write or read operation. Ensure the busy signal is low before asserting the write_all signal or read signal high (for a write or read instruction).
data_valid	Output	Applicable only to PMA controls reconfiguration mode. The data_valid port indicates the validity of the data read from the transceiver by the dynamic reconfiguration controller. The current data on the output read ports is the valid data ONLY if data_valid is high. The data_valid signal is enabled when you enable at least one PMA control port used in read transactions; for example, tx_vodctrl_out.
error	Output	Indicates that an unsupported operation is attempted. You can select the error port in the **Error checks/Data rate switch** screen. The dynamic reconfiguration controller deasserts the busy signal and asserts the error signal for two reconfig_clk cycles when you attempt an unsupported operation.
logical_channel_address [8:0]	Input	Enabled by the ALTGX_RECONFIG MegaWizard Plug-In Manager when you enable the **Use 'logical_channel_address' port for Analog controls reconfiguration** option in the **Analog controls** screen. The width of the logical_channel_address port depends on the value you set in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen. You can enable the logical_channel_address [8:0] port only when the number of channels controlled by the dynamic reconfiguration controller is more than one.
rx_tx_duplex_sel[1:0]	Input	A 2-bit wide signal. You can select the rx_tx_duplex_sel[1:0] signal in the **Error checks/Data rate switch** screen. The advantage of using the optional rx_tx_duplex_sel[1:0] port is that it allows you to reconfigure only the transmitter portion of a channel, even if the channel configuration is duplex. For a setting of: ■ rx_tx_duplex_sel[1:0] = 2'b00—the transmitter and receiver portion of the channel is reconfigured. ■ rx_tx_duplex_sel[1:0] = 2'b01—the receiver portion of the channel is reconfigured. ■ rx_tx_duplex_sel[1:0] = 2'b10—the transmitter portion of the channel is reconfigured.

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 5 of 8)

Port Name	Input/ Output	Description
Analog Settings Control/Status Signals		
`tx_vodctrl[2:0]` (1)	Input	An optional transmit buffer V_{OD} control signal. The `tx_vodctrl[2:0]` signal is 3 bits per transmitter channel. The number of settings varies based on the transmit buffer supply setting and the termination resistor setting on the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager. The width of the `tx_vodctrl[2:0]` signal is fixed to 3 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. The following shows the V_{OD} values corresponding to the `tx_vodctrl` settings for 100-Ω termination. For more information, refer to the "Programmable Output Differential Voltage" section of the *Device Datasheet for Arria II Devices* chapter in volume 3 of the *Arria II Handbook*. `tx_vodctrl[2:0]` Corresponding ALTGX instance settings 3'b000 N/A 3'b001 1 3'b010 2 3'b011 3 3'b100 4 3'b101 5 3'b110 6 3'b111 7
`tx_preemp[4:0]` (1)	Input	An optional pre-emphasis write control for the first post-tap for the transmit buffer. Depending on what value you set at this `tx_preemp[4:0]` input, the controller dynamically writes the value to the first post-tap control register of the transmit buffer. The width of the `tx_preemp[4:0]` signal is fixed to 5 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen.
`rx_eqctrl[3:0]` (1)	Input	An optional write control to write an equalization control value for the receive side of the PMA. The width of the `rx_eqctrl[3:0]` signal is fixed to 4 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. NOTE: The upper 2-bits are unused or reserved.

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 6 of 8)

Port Name	Input/ Output	Description
`rx_eqdcgain[2:0]` (1), (2)	Input	An optional equalizer DC gain write control. The width of the `rx_eqdcgain[2:0]` signal is fixed to 2 bits if you enable either the **Use 'logical_channel_address' port for Analog controls reconfiguration** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. The following values are the legal settings allowed for this signal: 3'b00 => 0 dB 3'b01 => 3 dB 3'b10 => 6 dB All other values => N/A For more information, refer to the "Programmable Equalization and DC Gain" section of the *Transceiver Architecture in Arria II Devices* chapter in volume 2 of the *Arria II Device Handbook*.
`tx_vodctrl_out[2:0]`	Output	An optional transmit V_{OD} read control signal. The `tx_vodctrl_out[2:0]` signal reads out the value written into the V_{OD} control register. The width of the `tx_vodctrl_out[2:0]` output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`tx_preemp_out[4:0]`	Output	An optional first post-tap, pre-emphasis read control signal. The `tx_preemp_out[4:0]` signal reads out the value written by its input control signal. The width of the `tx_preemp_out[4:0]` output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`rx_eqctrl_out[3:0]`	Output	An optional read control signal to read the setting of equalization setting of the ALTGX instance. The width of the `rx_eqctrl_out[3:0]` output signal depends on the number of channels controlled by the dynamic reconfiguration controller. NOTE: The upper 2-bits are unused or reserved.
`rx_eqdcgain_out[2:0]`	Output	An optional equalizer DC gain read control signal. The `rx_eqdcgain_out[2:0]` signal reads out the settings of the ALTGX instance DC gain. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
Transceiver Channel Reconfiguration Control/Status Signals		
`reconfig_mode_sel[2:0]`	Input	Set the following values at this signal to activate the appropriate dynamic reconfiguration mode: 3'b000 = PMA controls reconfiguration mode. This is the default value. 3'b011 = data rate division in transmitter mode 3'b100 = CMU PLL reconfiguration mode 3'b101 = channel and CMU PLL reconfiguration mode 3'b110 = channel reconfiguration with transmitter PLL select mode 3'b111 = central control unit reconfiguration mode `reconfig_mode_sel[]` is available as an input only when you enable more than one dynamic reconfiguration mode.

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 7 of 8)

Port Name	Input/ Output	Description
reconfig_address_out [5:0]	Output	Always available for you to select in the **Channel and TX PLL reconfiguration** screen. The reconfig_address_out[5:0] signal is applicable only in the dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. The reconfig_address_out[5:0] signal represents the current address used by the ALTGX_RECONFIG instance when writing the **.mif** into the transceiver channel. The reconfig_address_out[5:0] signal increments by 1, from 0 to the last address, then starts at 0 again. You can use the reconfig_address_out[5:0] signal to indicate the end of all the **.mif** write transactions (reconfig_address_out[5:0] changes from the last address to 0 at the end of all the **.mif** write transactions).
reconfig_address_en	Output	An optional signal you can select in the **Channel and TX PLL reconfiguration** screen. The reconfig_address_en signal is applicable only in dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. The dynamic reconfiguration controller asserts reconfig_address_en to indicate that reconfig_address_out[5:0] has changed. The reconfig_address_en signal is asserted only after the dynamic reconfiguration controller completes writing one 16-bit word of the **.mif**.
reset_reconfig_address	Input	An optional signal you can select in the **Channel and TX PLL reconfiguration** screen. The reset_reconfig_address signal is applicable only in dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. Enable the reset_reconfig_address signal and assert it for one reconfig_clk clock cycle if you want to reset the reconfiguration address used by the ALTGX_RECONFIG instance during reconfiguration.
reconfig_data[15:0]	Input	Applicable only in the dynamic reconfiguration modes grouped under the **Channel and TX PLL select/reconfig** option. The reconfig_data[15:0] is a 16-bit word carrying the reconfiguration information. The signal is stored in a **.mif** that you must generate. The ALTGX_RECONFIG instance requires that you provide reconfig_data [15:0] on every **.mif** write transaction using the write_all signal.
reconfig_address[5:0]	Input	Available for selection only in the **.mif**-based transceiver channel reconfiguration modes.
rate_switch_ctrl[1:0]	Input	Available when you select **data rate division in transmitter** mode. Based on the value you set here, the divide-by setting of the local divider in the transmitter channel is modified. The legal values for this port are: 2'b00 = Divide by 1 2'b01 = Divide by 2 2'b10 = Divide by 4 2'b11 = Not supported

Table 5. Arria II GZ Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 8 of 8)

Port Name	Input/ Output	Description
rate_switch_out[1:0]	Input	Available when you select **data rate division in transmitter** mode. You can read the existing local divider settings of a transmitter channel at this port. The decoding for this signal is listed below: 2'b00 = Division of 1 2'b01 = Division of 2 2'b10 = Division of 4 2'b11= Not supported
logical_tx_pll_sel	Input	At this port you specify the identity of the transmitter PLL you want to reconfigure. You can also specify the identity of the transmitter PLL that you want the transceiver channel to listen to. When you enable the logical_tx_pll_sel signal, the value set at this signal overwrites the logical_tx_pll value contained in the **.mif**. The value at the logical_tx_pll_sel port must be held at a constant logic level until reconfiguration is done.
logical_tx_pll_sel_en	Input	If you want to use the logical_tx_pll_sel port only under some conditions and use the logical_tx_pll value contained in the **.mif** otherwise, enable this optional logical_tx_pll_sel_en port. Only when logical_tx_pll_sel_en is enabled and set to **1** does the dynamic reconfiguration controller use logical_tx_pll_sel to identify the transmitter PLL. The value at this port must be held at a constant logic level until reconfiguration is done.
channel_reconfig_done	Output	Goes high to indicate that the dynamic reconfiguration controller has finished writing all the words of the **.mif**. The channel_reconfig_done signal is automatically deasserted at the start of a new dynamic reconfiguration write sequence. The channel_reconfig_done signal is applicable only in channel and CMU PLL reconfiguration and channel reconfiguration with transmitter PLL select modes.

Notes to Table 5:

(1) Not all combinations of input bits are legal values.

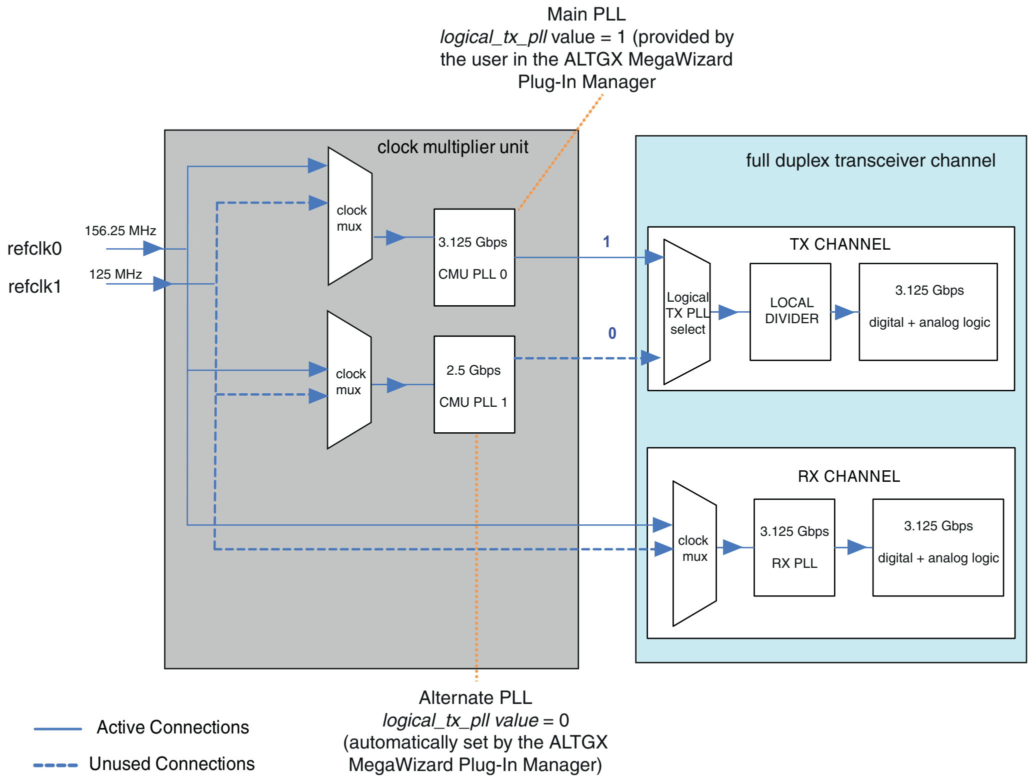
(2) In PCIe mode, this input must be tied to 001 to be PCI E-compliant.

Main PLL and Alternate PLL

To reconfigure the CMU PLL during run time, you need the flexibility to select one of the two CMU PLLs (Figure 29).

For example, assume you want the transceiver channel to listen to `CMU PLL 0` and you want to reconfigure `CMU PLL 0`.

Figure 29. Main PLL and Alternate PLL


Main PLL
logical_tx_pll value = 1 (provided by the user in the ALTGX MegaWizard Plug-In Manager

Alternate PLL
logical_tx_pll value = 0 (automatically set by the ALTGX MegaWizard Plug-In Manager)

Select `CMU PLL 0` by specifying its identity in the ALTGX MegaWizard Plug-In Manager. This identification is referred to as the logical `tx pll` value. This value provides a logical identification to `CMU PLL 0` and associates it with a transceiver channel, without requiring knowledge of its physical location.

In the ALTGX MegaWizard Plug-In Manager, the transmitter PLL configuration set in the **General** tab is called the main PLL. When you provide the main PLL with a logical `tx pll` value, for example **1**, the alternate PLL automatically takes the complement value 0. The logical `tx pll` value for the main PLL is stored along with the other transceiver channel information in the generated **.mif**.

You can reuse the **.mif** generated for one CMU PLL to reconfigure the other CMU PLL in the same or other transceiver blocks. For more information, refer to "Guidelines for Reusing .mif Files" on page 70.

Provide the logical `tx pll` value for the main PLL in the **What is the main PLL logical reference index?** option on the **Main PLL** tab.

CMU PLL reconfiguration mode is also useful when used in combination with dynamic reconfiguration mode and channel reconfiguration with CMU PLL select.

Consider that you have one transceiver channel listening to one CMU PLL and you want to reconfigure the transceiver channel to a different data rate. You can first use CMU PLL reconfiguration mode (set the `reconfig_mode_sel` to **100**) and reconfigure the second unused CMU PLL of the transceiver block to the desired data rate. For more information, refer to "CMU PLL Reconfiguration Example" on page 31.

You can then use channel reconfiguration with CMU PLL select mode (set the `reconfig_mode_sel` to **110**) and reconfigure the transceiver channel to listen to the second reconfigured CMU PLL.

The main PLL corresponds to the CMU PLL configuration set on the **General** tab of the ALTGX MegaWizard Plug-In Manager and the alternate PLL corresponds to the CMU PLL configuration set on the **Reconfig Alt PLL tab.**

Clocking/Interface Options

This section describes the **Clocking/Interface** options.

The core clocking setup describes the transceiver core clocks that are the write and read clocks of the TX Phase Compensation FIFO and the RX Phase Compensation FIFO, respectively. Core clocking is classified as:

- "Transmitter Core Clocking"
- "Receiver Core Clocking" on page 56

Transmitter Core Clocking

Transmitter core clocking refers to the clock that is used to write the parallel data from the FPGA fabric into the TX Phase Compensation FIFO. You can use one of the following clocks to write into the TX Phase Compensation FIFO:

- `tx_coreclk`—Use a clock of the same frequency as `tx_clkout` from the FPGA fabric to provide the write clock to the TX Phase Compensation FIFO. If you use `tx_coreclk`, it overrides the `tx_clkout` options in the ALTGX MegaWizard Plug-In Manager.
- `tx_clkout`—The Quartus II software automatically routes `tx_clkout` to the FPGA fabric and back into the TX Phase Compensation FIFO.

Option 1: Share a Single Transmitter Core Clock Between Transmitters

- Enable this option if you want `tx_clkout` of the first channel (channel 0) of the transceiver block to provide the write clock to the TX Phase Compensation FIFOs of the remaining channels in the transceiver block.
- This option is typically enabled when all channels of a transceiver block are of the same functional mode and data rate and are reconfigured to the identical functional mode and data rate.

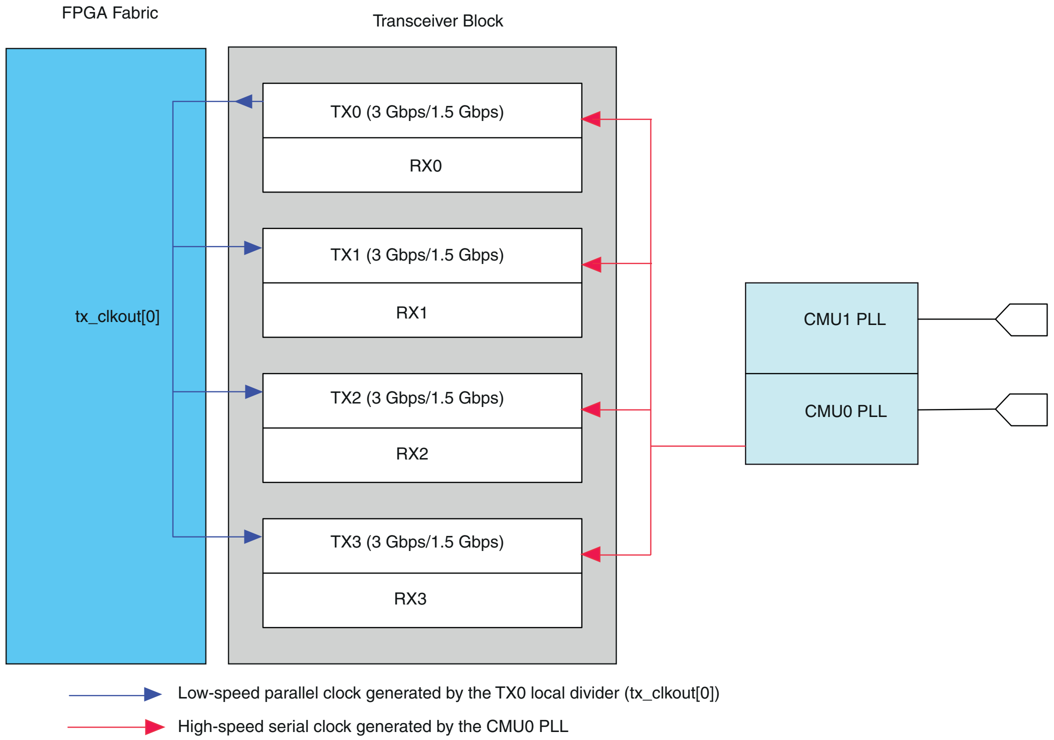
For example, assume the following scenario:

- Four regular transceiver channels configured at 3 Gbps and in the same functional mode.
- Channel and CMU PLL reconfiguration mode is enabled in the ALTGX_RECONFIG MegaWizard Plug-In Manager.
- You want to reconfigure all four regular transceiver channels to 1.5 Gbps and vice versa.

Option 1 is applicable in this scenario because it saves clock resources.

Figure 30 shows the sharing of channel 0's `tx_clkout` between all four regular channels of a transceiver block.

Figure 30. Option 1 for Transmitter Core Clocking (Channel and CMU PLL Reconfiguration Mode)



Option 2: Use the Respective Channel Transmitter Core Clocks

- Enable Option 2 if you want the individual transmitter channel `tx_clkout` signals to provide the write clock to their respective TX Phase Compensation FIFOs.
- Option 2 is typically enabled when each transceiver channel is reconfigured to a different functional mode using channel reconfiguration.

For example, assume the following scenario:

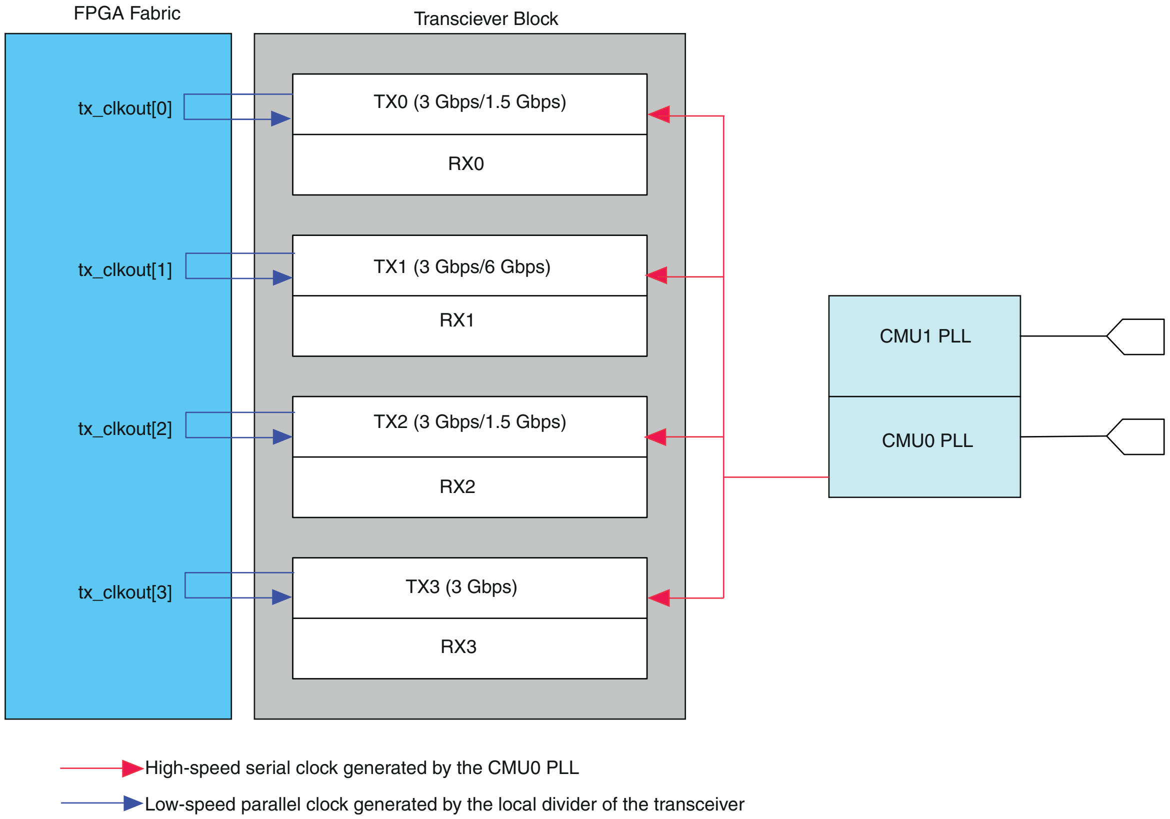
- Four regular transceiver channels configured at 3 Gbps in the same functional modes.

- Channel and CMU PLL reconfiguration mode is enabled in the ALTGX_RECONFIG MegaWizard Plug-In Manager.
- You want to reconfigure each of the four regular transceiver channels to different data rates and different functional modes.

Option 2 is applicable in this scenario because the design requires all four regular transceiver channels to be reconfigured to different data rates and functional modes. You can reconfigure each channel to a different functional mode using the channel and CMU PLL reconfiguration mode.

Figure 31 shows how each transmitter channel's `tx_clkout` signal provides a clock to the TX Phase Compensation FIFOs of the respective transceiver channels.

Figure 31. Option 2 for Transmitter Core Clocking (Channel and CMU PLL Reconfiguration Mode)



Receiver Core Clocking

Receiver core clocking refers to the clock that is used to read the parallel data from the RX Phase Compensation FIFO into the FPGA fabric. You can use one of the following clocks to read from the RX Phase Compensation FIFO:

- `rx_coreclk`—Use a clock of the same frequency as `rx_clkout` from the FPGA fabric to provide the read clock to the RX Phase Compensation FIFO. If you use `rx_coreclk`, it overrides the `rx_clkout` options in the ALTGX MegaWizard Plug-In Manager.
- `rx_clkout`—The Quartus II software automatically routes `rx_clkout` to the FPGA fabric and back into the RX Phase Compensation FIFO.

☞ The **Clocking/Interface** screen is not available for PMA-only channels.

Option 1: Share a Single Transmitter Core Clock Between Receivers

- Enable Option 1 if you want `tx_clkout` of the first channel (channel 0) of the transceiver block to provide the read clock to the RX Phase Compensation FIFOs of the remaining receiver channels in the transceiver block.
- Option 1 is typically enabled when all channels of a transceiver block are in a Basic or a protocol configuration with rate matching enabled and are reconfigured to another Basic or a protocol configuration with rate matching enabled.

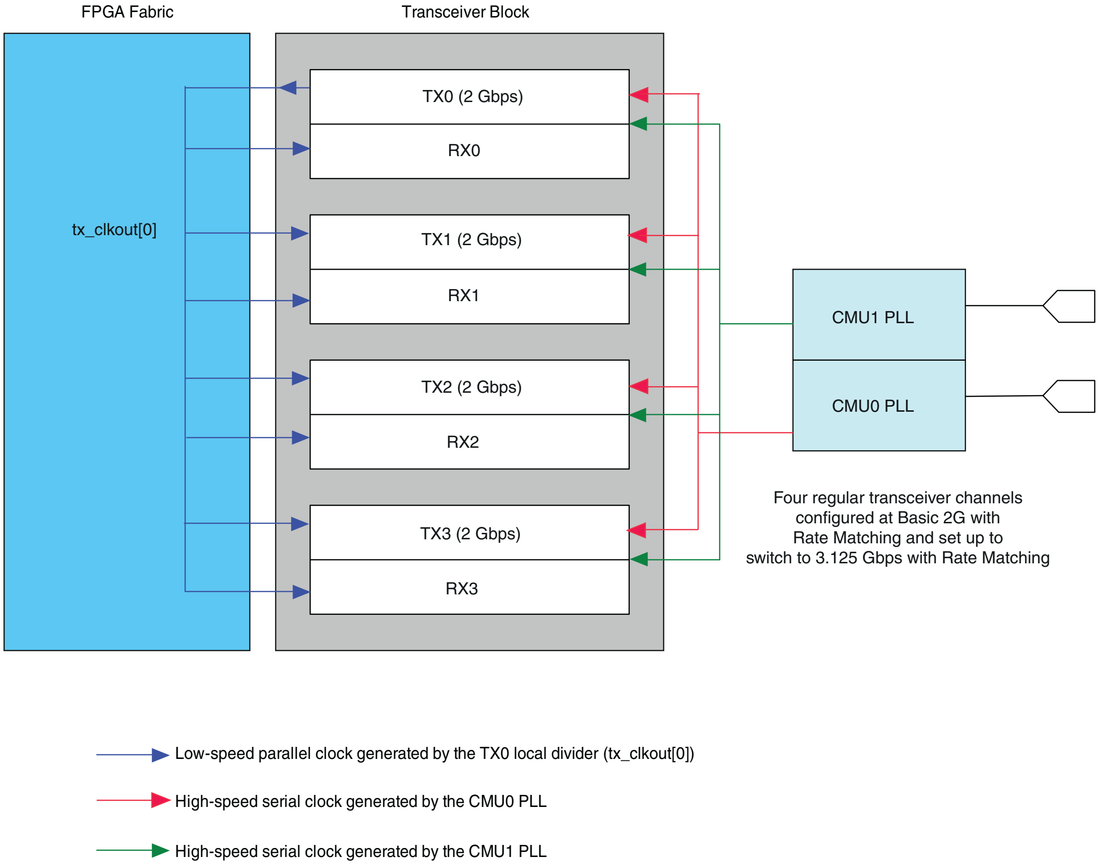
For example, assume the following scenario:

- Four regular transceiver channels configured to Basic 2 Gbps functional mode with rate matching enabled.
- Channel and CMU PLL reconfiguration mode is enabled in the ALTGX_RECONFIG MegaWizard Plug-In Manager.
- You want to reconfigure all four regular transceiver channels to a 3.125 Gbps configuration with rate matching enabled.

Option 1 is applicable in this scenario.

Figure 32 shows the sharing of channel 0's `tx_clkout` between all four channels of a transceiver block.

Figure 32. Option 1 for Receiver Core Clocking (Channel and CMU PLL Reconfiguration Mode)


FPGA Fabric
Transceiver Block
tx_clkout[0]
TX0 (2 Gbps)
RX0
TX1 (2 Gbps)
RX1
TX2 (2 Gbps)
RX2
TX3 (2 Gbps)
RX3
CMU1 PLL
CMU0 PLL
Four regular transceiver channels configured at Basic 2G with Rate Matching and set up to switch to 3.125 Gbps with Rate Matching
Low-speed parallel clock generated by the TX0 local divider (tx_clkout[0])
High-speed serial clock generated by the CMU0 PLL
High-speed serial clock generated by the CMU1 PLL

Option 2: Use the Respective Channel Transmitter Core Clocks

- Enable Option 2 if you want the individual transmitter channel's `tx_clkout` signal to provide the read clock to its respective RX Phase Compensation FIFO.
- Option 2 is typically enabled when all transceiver channels have rate matching enabled with different data rates and are reconfigured to another Basic or a protocol functional mode with rate matching enabled.

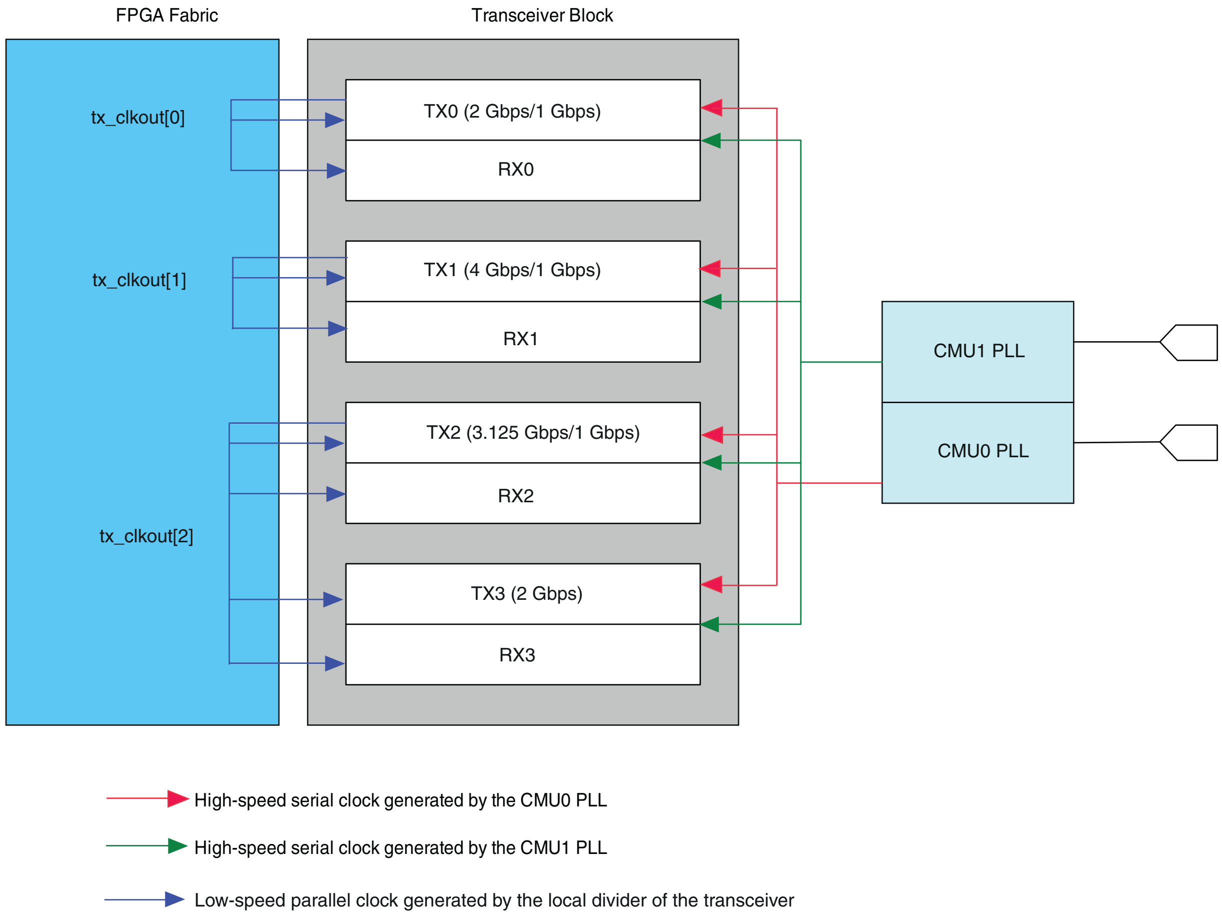
For example, assume the following scenario:

- TX0/RX0: You want to dynamically reconfigure a Basic 1 Gbps configuration with rate matching enabled to a Basic 2 Gbps configuration with rate matching enabled.
- TX1/RX1: You want to dynamically reconfigure a Basic 4 Gbps configuration with rate matching enabled to a Basic 1 Gbps configuration with rate matching enabled.
- TX2/RX2 and TX3/RX3: You want to dynamically reconfigure a Basic 3.125 Gbps configuration with rate matching enabled to a 1 Gbps configuration with rate matching and vice versa.
- Channel and CMU PLL reconfiguration mode is enabled in the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Option 2 is applicable because the design requires the individual transceiver channels to be reconfigured with different data rates to another Basic or a protocol functional mode with rate matching. Therefore, you can reconfigure each channel to another Basic or a protocol functional mode with rate matching enabled and a different data rate.

Figure 33 shows the respective `tx_clkout` of each channel clocking the respective channels of a transceiver block.

Figure 33. Option 2 for Receiver Core Clocking (Channel and CMU PLL Reconfiguration Mode)


FPGA Fabric
Transceiver Block
tx_clkout[0]
tx_clkout[1]
tx_clkout[2]
TX0 (2 Gbps/1 Gbps)
RX0
TX1 (4 Gbps/1 Gbps)
RX1
TX2 (3.125 Gbps/1 Gbps)
RX2
TX3 (2 Gbps)
RX3
CMU1 PLL
CMU0 PLL
High-speed serial clock generated by the CMU0 PLL
High-speed serial clock generated by the CMU1 PLL
Low-speed parallel clock generated by the local divider of the transceiver

Option 3: Use the Respective Channel Receiver Core Clocks

- Enable Option 3 if you want the individual channel's `rx_clkout` signal to provide the read clock to its respective RX Phase Compensation FIFO.
- Option 3 is typically enabled when the channel is reconfigured from Basic or a protocol configuration with or without rate matching to another Basic or a protocol configuration with or without rate matching. However, both Basic protocol configurations must match—either both configurations have rate matching enabled or both configurations have rate matching disabled.

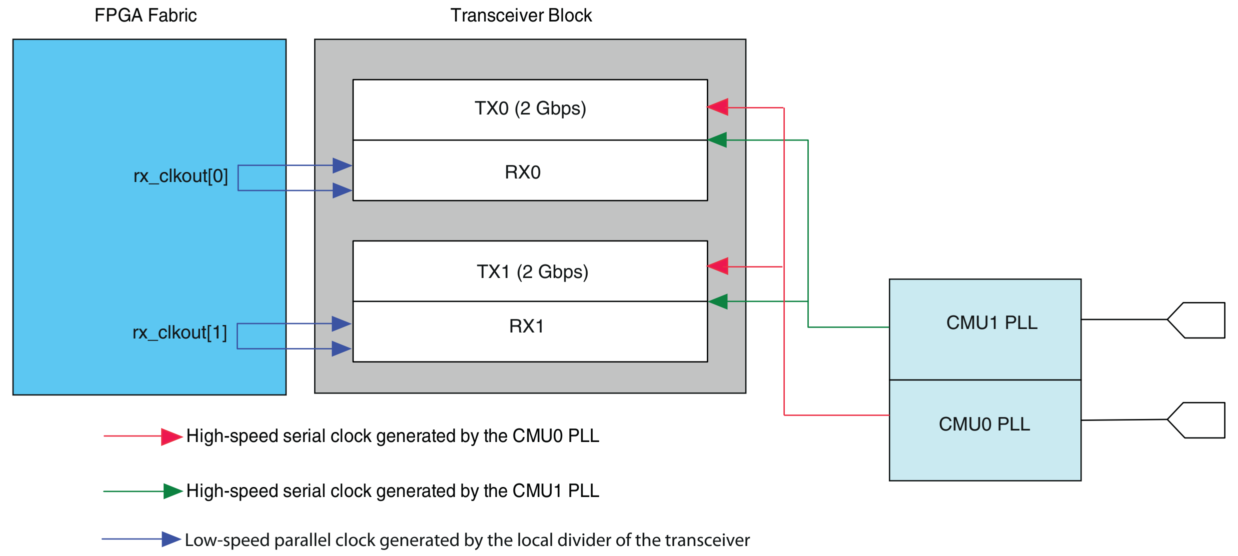
For example, assume the following scenario:

- TX1/RX1: GIGE configuration to a SONET/SDH OC48 configuration.
- TX2/RX2: Basic 2.5 Gbps configuration with rate matching disabled to a Basic 1.244 Gbps configuration with rate matching disabled.
- Channel and CMU PLL reconfiguration mode is enabled in the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Option 3 is applicable in this scenario.

Figure 34 shows the respective `rx_clkout` of each channel clocking the respective receiver channels of a transceiver block.

Figure 34. Option 3 for Receiver Core Clocking (Channel and CMU PLL Reconfiguration Mode)


FPGA Fabric
Transceiver Block
TX0 (2 Gbps)
RX0
TX1 (2 Gbps)
RX1
rx_clkout[0]
rx_clkout[1]
CMU1 PLL
CMU0 PLL
High-speed serial clock generated by the CMU0 PLL
High-speed serial clock generated by the CMU1 PLL
Low-speed parallel clock generated by the local divider of the transceiver

Channel Interface

Enable the channel interface option if the reconfiguration of the transceiver channel involves the following changes:

- The reconfigured channel has a changed FPGA fabric-transceiver channel interface data width
- The reconfigured channel has changed input control signals and output status signals
- The reconfigured channel has enabled and disabled the static PCS blocks of the transceiver channel

These input signals are available when you enable the channel interface option:

- `tx_datainfull`—The width of this input signal depends on the number of channels you set up in the ALTGX MegaWizard Plug-In Manager. `tx_datainfull` is 33 bits wide per channel for GX devices and 44 bits wide per channel for GZ devices. This signal is available for **Transmitter-only** and **Receiver and Transmitter** configurations only. This port replaces the existing `tx_datain` port.
- `rx_dataoutfull`—The width of this output signal depends on the number of channels you set up in the ALTGX MegaWizard Plug-In Manager. `rx_dataoutfull` is 47 bits wide per channel for GX devices and 64 bits wide per channel for GZ devices. This signal is available for **Receiver-only** and **Receiver and Transmitter** configurations only. This port replaces the existing `rx_dataout` port.

The Quartus II software completes legality checks to validate the connectivity of `tx_datainfull` and `rx_dataoutfull` and the various control and status signals you enable in the **Clocking/Interface** screen. For example, the Quartus II software allows you to select and connect the `pipestatus` and `powerdn` signals. The Quartus II software assumes that you are planning to switch to and from PCIe functional mode.

Table 6 lists the tx_datainfull[43:0] FPGA fabric-transceiver channel interface signals.

Table 6. tx_datainfull[43:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 1 of 3) (1)

FPGA Fabric-Transceiver Channel Interface Description	Transmit Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)
8-bit FPGA fabric-Transceiver Channel Interface	tx_datainfull[7:0]: 8-bit data (tx_datain)
	The following signals are used only in 8B/10B modes:
	tx_datainfull[8]: Control bit (tx_ctrlenable)
	tx_datainfull[9] Transmitter force disparity Compliance (PCIe) (tx_forcedisp) in all modes except PCIe. For PCIe mode, (tx_forcedispcompliance) is used
	For non-PIPE: tx_datainfull[10]: Forced disparity value (tx_dispval) For PCIe: tx_datainfull[10]: Forced electrical idle (tx_forceelecidle)
10-bit FPGA fabric-Transceiver Channel Interface	tx_datainfull[9:0]: 10-bit data (tx_datain)
16-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 16/20 bits	Two 8-bit Data (tx_datain) tx_datainfull[7:0] - tx_datain (LSByte) and tx_datainfull[18:11] - tx_datain (MSByte)
	The following signals are used only in 8B/10B modes:
	tx_datainfull[8] - tx_ctrlenable (LSB) and tx_datainfull[19] - tx_ctrlenable (MSB)
	Force Disparity Enable tx_datainfull[9] - tx_forcedisp (LSB) and tx_datainfull[20] - tx_forcedisp (MSB)
	Force Disparity Value tx_datainfull[10]: tx_dispval (LSB) and tx_datainfull[21] - tx_dispval (MSB)

Table 6. tx_datainfull[43:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 2 of 3) (1)

FPGA Fabric-Transceiver Channel Interface Description	Transmit Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)
16-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 8/10 bits	Two 8-bit Data (`tx_datain`) `tx_datainfull[7:0]` - `tx_datain` (LSByte) and `tx_datainfull[29:22]` - `tx_datain` (MSByte)
	The following signals are used only in 8B/10B modes:
	Two Control Bits (`tx_ctrlenable`) `tx_datainfull`[8] - `tx_ctrlenable` (LSB) and `tx_datainfull`[30] - `tx_ctrlenable` (MSB)
	Force Disparity Enable For non-PIPE: `tx_datainfull[9]` - `tx_forcedisp` (LSB) and `tx_datainfull[31]` - `tx_forcedisp` (MSB) For PCIe: `tx_datainfull[9]` - `tx_forcedispcompliance` (LSB) and `tx_datainfull[31]` - Tie to 0
	Force Disparity Value For non-PIPE: `tx_datainfull[10]`: `tx_dispval` (LSB) and `tx_datainfull[32]` - `tx_dispval` (MSB) For PCIe: `tx_datainfull[10]` - `tx_forceelecidle` and `tx_datainfull[32]` - `tx_forceelecidle`
20-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 20 bits	Two 10-bit Data (`tx_datain`) `tx_datainfull[9:0]` - `tx_datain` (LSByte) and `tx_datainfull[20:11]` - `tx_datain` (MSByte)
20-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 10 bits	Two 10-bit Data (`tx_datain`) `tx_datainfull[9:0]` - `tx_datain` (LSByte) and `tx_datainfull[31:22]` - `tx_datain` (MSByte)

Table 6. tx_datainfull[43:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 3 of 3) [(1)]

FPGA Fabric-Transceiver Channel Interface Description	Transmit Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)

32-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 16/20 bits	Four 8-bit Data (`tx_datain`)
	`tx_datainfull[7:0]` - `tx_datain` (LSByte) and
	`tx_datainfull[18:11]`
	`tx_datainfull [29:22]`
	`tx_datainfull[40:33]` - `tx_datain` (MSByte)

	The following signals are used only in 8B/10B modes:

	Four Control Data (`tx_ctrlenable`)
	`tx_datainfull[8]` - `tx_ctrlenable` (LSByte) and
	`tx_datainfull[19]`
	`tx_datainfull [30]`
	`tx_datainfull[41]` - `tx_ctrlenable` (MSByte)

	Force Disparity Enable (`tx_forcedisp`)
	`tx_datainfull[9]` - `tx_forcedisp` (LSB) and
	`tx_datainfull[20]`
	`tx_datainfull [31]`
	`tx_datainfull[42]` - `tx_forcedisp` (MSB)

	Force Disparity value (`tx_dispval`)
	`tx_datainfull[10]` - `tx_dispval` (LSB) and
	`tx_datainfull[21]`
	`tx_datainfull [32]`
	`tx_datainfull[43]` - `tx_dispval` (MSB)

40-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 20 bits	Four 10-bit Data (`tx_datain`)
	`tx_datainfull[9:0]` - `tx_datain` (LSByte) and
	`tx_datainfull[20:11]`
	`tx_datainfull [31:22]`
	`tx_datainfull[42:33]` - `tx_datain` (MSByte)

Note to Table 6:

(1) For all transceiver-related ports, refer to the “Transceiver Port Lists” section in the *Transceiver Architecture in Arria II Devices* chapter in volume 2 of the *Arria II Device Handbook*.

Table 7 lists the rx_dataoutfull[63:0] FPGA fabric-transceiver channel interface signals.

Table 7. rx_dataoutfull[63:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 1 of 5)

FPGA Fabric-Transceiver Channel Interface Description	Receive Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)
8-bit FPGA fabric-Transceiver Channel Interface	**The following signals are used in 8-bit 8B/10B modes:**
	rx_dataoutfull[7:0]: 8-bit decoded data (rx_dataout)
	rx_dataoutfull[8]: Control bit (rx_ctrldetect)
	rx_dataoutfull[9]: Code violation status signal (rx_errdetect)
	rx_dataoutfull[10]: rx_syncstatus
	rx_dataoutfull[11]: Disparity error status signal (rx_disperr)
	rx_dataoutfull[12]: Pattern detect status signal (rx_patterndetect)
	rx_dataoutfull[13]: Rate Match FIFO deletion status indicator (rx_rmfifodatadeleted) in non-PCIe/PCIe modes.
	rx_dataoutfull[14]: Rate Match FIFO insertion status indicator (rx_rmfifodatainserted) in non-PCIe/PCIe modes.
	rx_dataoutfull[14:13]: PCIe mode (rx_pipestatus)
	rx_dataoutfull[15]: 8B/10B running disparity indicator (rx_runningdisp)
	The following signals are used in 8-bit SONET/SDH mode:
	rx_dataoutfull[7:0]: 8-bit un-encoded data (rx_dataout)
	rx_dataoutfull[8]: rx_a1a2sizeout
	rx_dataoutfull[10]: rx_syncstatus
	rx_dataoutfull[11]: Reserved
	rx_dataoutfull[12]: rx_patterndetect
10-bit FPGA fabric-Transceiver Channel Interface	rx_dataoutfull[9:0]: 10-bit un-encoded data (rx_dataout)
	rx_dataoutfull[10]: rx_syncstatus
	rx_dataoutfull[11]: 8B/10B disparity error indicator (rx_disperr)
	rx_dataoutfull[12]: rx_patterndetect
	rx_dataoutfull[13]: Rate Match FIFO deletion status indicator (rx_rmfifodatadeleted) in non-PCIe/PCIe modes
	rx_dataoutfull[14]: Rate Match FIFO insertion status indicator (rx_rmfifodatainserted) in non-PCIe/PCIe modes
	rx_dataoutfull[15]: 8B/10B running disparity indicator (rx_runningdisp)

Table 7. rx_dataoutfull[63:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 2 of 5)

FPGA Fabric-Transceiver Channel Interface Description	Receive Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)
16-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 16/20 bits	Two 8-bit unencoded Data (`rx_dataout`) `rx_dataoutfull[7:0]` - `rx_dataout` (LSByte) and `rx_dataoutfull[23:16]` - `rx_dataout` (MSByte)
	The following signals are used in 16-bit 8B/10B mode:
	Two Control Bits `rx_dataoutfull[8]` - `rx_ctrldetect` (LSB) and `rx_dataoutfull[24]` - `rx_ctrldetect` (MSB)
	Two Receiver Error Detect Bits `rx_dataoutfull[9]` - `rx_errdetect` (LSB) and `rx_dataoutfull[25]`- `rx_errdetect` (MSB)
	Two Receiver Sync Status Bits `rx_dataoutfull[10]` - `rx_syncstatus` (LSB) and `rx_dataoutfull[26]`- `rx_syncstatus` (MSB)
	Two Receiver Disparity Error Bits `rx_dataoutfull[11]` - `rx_disperr` (LSB) and `rx_dataoutfull[27]` - `rx_disperr` (MSB)
	Two Receiver Pattern Detect Bits `rx_dataoutfull[12]` - `rx_patterndetect` (LSB) and `rx_dataoutfull[28]` - `rx_patterndetect` (MSB)
	`rx_dataoutfull[13]` and `rx_dataoutfull[45]`: Rate Match FIFO deletion status indicator (`rx_rmfifodatadeleted`) in non-PCIe/PCIe modes
	`rx_dataoutfull[14]` and `rx_dataoutfull[46]`: Rate Match FIFO insertion status indicator (`rx_rmfifodatainserted`) in non-PCIe/PCIe modes
	Two 2-bit PCIe Status Bits `rx_dataoutfull[14:13]` - `rx_pipestatus` (LSB) and `rx_dataoutfull[30:29]`- `rx_pipestatus` (MSB)

Table 7. rx_dataoutfull[63:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 3 of 5)

FPGA Fabric-Transceiver Channel Interface Description	Receive Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)
16-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 8/10 bits	Two 8-bit Data rx_dataoutfull[7:0] - rx_dataout (LSByte) and rx_dataoutfull[39:32] - rx_dataout (MSByte)
	The following signals are used in 16-bit 8B/10B mode:
	Two Control Bits rx_dataoutfull[8] - rx_ctrldetect (LSB) and rx_dataoutfull[40] - rx_ctrldetect (MSB)
	Two Receiver Error Detect Bits rx_dataoutfull[9] - rx_errdetect (LSB) and rx_dataoutfull[41]- rx_errdetect (MSB)
	Two Receiver Sync Status Bits rx_dataoutfull[10] - rx_syncstatus (LSB) and rx_dataoutfull[42]- rx_syncstatus (MSB)
	Two Receiver Disparity Error Bits rx_dataoutfull[11] - rx_disperr (LSB) and rx_dataoutfull[43] - rx_disperr (MSB)
	Two Receiver Pattern Detect Bits rx_dataoutfull[12] - rx_patterndetect (LSB) and rx_dataoutfull[44] - rx_patterndetect (MSB)
	rx_dataoutfull[13] and rx_dataoutfull[45]: Rate Match FIFO deletion status indicator (rx_rmfifodatadeleted) in non-PCIe/PCIe modes
	rx_dataoutfull[14] and rx_dataoutfull[46]: Rate Match FIFO insertion status indicator (rx_rmfifodatainserted) in non-PCIe/PCIe modes
	Two 2-bit PCIe Status Bits rx_dataoutfull[14:13] - rx_pipestatus (LSB) and rx_dataoutfull[46:45]- rx_pipestatus (MSB)
	rx_dataoutfull[15] and rx_dataoutfull[47]: 8B/10B running disparity indicator (rx_runningdisp)

Table 7. rx_dataoutfull[63:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 4 of 5)

FPGA Fabric-Transceiver Channel Interface Description	Receive Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)
16-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 8/10 bits (continued)	**The following signals are used in 16-bit SONET/SDH mode:**
	Two 8-bit Data `rx_dataoutfull[7:0]` - `rx_dataout` (LSByte) and `rx_dataoutfull[39:32]` - `rx_dataout` (MSByte)
	Two Receiver Alignment Pattern Length Bits `rx_dataoutfull[8]` - `rx_a1a2sizeout` (LSB) and `rx_dataoutfull[40]`- `rx_a1a2sizeout` (MSB)
	Two Receiver Sync Status Bits `rx_dataoutfull[10]` - `rx_syncstatus` (LSB) and `rx_dataoutfull[42]` - `rx_syncstatus` (MSB)
	Two Receiver Pattern Detect Bits `rx_dataoutfull[12]` - `rx_patterndetect` (LSB) and `rx_dataoutfull[44]` - `rx_patterndetect` (MSB)
20-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 20 bits	Two 10-bit Data (`rx_dataout`) `rx_dataoutfull[9:0]` - `rx_dataout` (LSByte) and `rx_dataoutfull[25:16]` - `rx_dataout` (MSByte)
	Two Receiver Sync Status Bits `rx_dataoutfull[10]` - `rx_syncstatus` (LSB) and `rx_dataoutfull[26]` - `rx_syncstatus` (MSB)
	`rx_dataoutfull[11]` and `rx_dataoutfull[27]`: 8B/10B disparity error indicator (`rx_disperr`)
	Two Receiver Pattern Detect Bits `rx_dataoutfull[12]` - `rx_patterndetect` (LSB) and `rx_dataoutfull[28]` - `rx_patterndetect` (MSB)
	`rx_dataoutfull[13]` and `rx_dataoutfull[29]`: Rate Match FIFO deletion status indicator (`rx_rmfifodatadeleted`) in non-PCIe/PCIe modes
	`rx_dataoutfull[14]` and `rx_dataoutfull[30]`: Rate Match FIFO insertion status indicator (`rx_rmfifodatainserted`) in non-PCIe/PCIe modes
	`rx_dataoutfull[15]` and `rx_dataoutfull[31]`: 8B/10B running disparity indicator (`rx_runningdisp`)

Table 7. rx_dataoutfull[63:0] FPGA Fabric-Transceiver Channel Interface Signal Descriptions (Part 5 of 5)

FPGA Fabric-Transceiver Channel Interface Description	Receive Signal Description (Based on Arria II Supported FPGA Fabric-Transceiver Channel Interface Widths)
20-bit FPGA fabric-Transceiver Channel Interface with PCS-PMA set to 10 bits	Two 10-bit Data `rx_dataoutfull[9:0]` - `rx_dataout` (LSByte) and `rx_dataoutfull[41:32]` - `rx_dataout` (MSByte)
	Two Receiver Sync Status Bits `rx_dataoutfull[10]` - `rx_syncstatus` (LSB) and `rx_dataoutfull[42]` - `rx_syncstatus` (MSB)
	`rx_dataoutfull[11]` and `rx_dataoutfull[43]`: 8B/10B disparity error indicator (`rx_disperr`)
	Two Receiver Pattern Detect Bits `rx_dataoutfull[12]` - `rx_patterndetect` (LSB) and `rx_dataoutfull[44]` - `rx_patterndetect` (MSB)
	`rx_dataoutfull[13]` and `rx_dataoutfull[45]`: Rate Match FIFO deletion status indicator (`rx_rmfifodatadeleted`) in non-PCIe/PCIe modes
	`rx_dataoutfull[14]` and `rx_dataoutfull[46]`: Rate Match FIFO insertion status indicator (`rx_rmfifodatainserted`) in non-PCIe/PCIe modes
	`rx_dataoutfull[15]` and `rx_dataoutfull[47]`: 8B/10B running disparity indicator (`rx_runningdisp`)

Control and Status Signals for Channel Reconfiguration

Table 8 lists the various control input signals involved in channel reconfiguration mode.

Table 8. Control and Status Signals for Channel Reconfiguration Mode

Signal	Description
`logical_channel_address [8:0]`	Use this control signal to select the specific channel you want to dynamically reconfigure using this mode. Based on the value you set at the `logical_channel_address [8:0]` port, the dynamic reconfiguration controller writes the **.mif** contents to the transceiver channel you specify. The `logical_channel_address [8:0]` signal is enabled when the number of channels controlled by the dynamic reconfiguration controller is more than one. Because channel reconfiguration is done on a per-channel basis, you must use the `logical_channel_address [8:0]` signal and provide the necessary logical channel address to write the **.mif** words so that a successful channel reconfiguration is achieved for that channel.
`reset_reconfig_address`	Use this optional control signal to reset the `reconfig_address_out` value to **0**. This reset control signal is only applicable in channel reconfiguration.
`reconfig_mode_sel[2:0]`	Refer to Table 4 on page 38.
`write_all`	Refer to Table 4 on page 38.

Table 8. Control and Status Signals for Channel Reconfiguration Mode

Signal	Description
Status Signals	
`reconfig_address_en`	This is an optional output signal. The ALTGX_RECONFIG instance asserts the `reconfig_address_en` signal to indicate the change in value on the `reconfig_address_out` port. The `reconfig_address_en` signal only is asserted after the dynamic reconfiguration controller completes writing the 16-bit data.
`reconfig_address_out [4:0]`	This is an optional output signal. The `reconfig_address_out [4:0]` signal provides the address value that you can use to read the appropriate word from the **.mif**. Use the value in the `reconfig_address_out [4:0]` port in combination with the `reconfig_address_en` signal to decide when to initiate a new write transaction.
`channel_reconfig_done`	This signal is available when you select the **Channel Reconfiguration** option in the dynamic reconfiguration controller. The `channel_reconfig_done` port indicates that the ALTGX_RECONFIG instance has finished writing all the words of a **.mif** in a sequence. The `channel_reconfig_done` signal is useful for user logic to implement reset recommendations during and after dynamic reconfiguration. For information about transceiver resets, refer to the “Reset Recommendations” section of the *Reset Control and Power Down in Arria II Devices* chapter in volume 2 of the *Arria II Device Handbook*.
`error`	Refer to “Error Indication of Analog Control Reconfiguration” on page 27.
`busy`	Refer to Table 4 on page 38.

Central Control Unit Reconfiguration Mode

You can use central control unit reconfiguration mode and one of the **Channel and TX PLL select/reconfig** options to reconfigure between bonded modes with the same data width, such as from Basic x4 to XAUI, or from PIPE x8 to Basic x8. There are two steps. First, you must reconfigure the CMU PLLs and each channel individually, then you must reconfigure the CMU clock divider. These two steps use the same **.mif** generated in “Stage V: Generate a .mif for Channel and TX PLL select/reconfig” on page 14.

For more information, refer to the *Transceiver Clocking in Arria II Devices* chapter in volume 2 of the *Arria II Device Handbook*.

The **.mif** contains all the information and settings of an ALTGX instance.

For example, to dynamically reconfigure an ALTGX instance in a Basic x4 configuration to a XAUI configuration, you must first:

1. Use channel and CMU reconfiguration to reconfigure each channel to XAUI functional mode. The ALTGX_RECONFIG instance then reads the corresponding **.mif** addresses and writes to the ALTGX instance. With four channels in this example, repeat this procedure four times.

2. After `channel_reconfig_done` is asserted, use central control unit reconfiguration mode to reconfigure the CCU portion of the transceiver from Basic to XAUI functional mode. The ALTGX_RECONFIG instance then continues to read the corresponding addresses from the same **.mif** and writes to the ALTGX instance. You can set `logical_channel_address` to any one of the addresses within the ALTGX instance.

   Use the same **.mif** for both of the these steps. In step 1, a partial **.mif** is written and in step 2, the remaining contents of the **.mif** is written. In step 1, reconfigure all the channels one-by-one. In step-2, reconfiguration of the central control unit is ALTGX-instance based.

☞ Dynamic reconfiguration is not available if you use the PCIe hard IP block.

CCU Reconfiguration Exception

Dynamic reconfiguration is not supported between configurations that have rate matching enabled and configurations that do not have rate matching enabled. Table 9 lists the configurations that do not support CCU reconfiguration.

Table 9. Reconfigurations that Do Not Support CCU Reconfiguration

Configuration From	Configuration To	CCU Reconfiguration
PCIe x4	Basic x4 without Rate Matching	Not supported
XAUI	Basic x4 without Rate Matching	Not Supported
Basic x4 with Rate Matching	Basic x4 without Rate Matching	Not Supported

☞ CCU reconfiguration is supported in all configurations that both have rate matching enabled or in all configurations that both do not have rate matching enabled.

Guidelines for Reusing .mif Files

To configure the transceiver PLLs and receiver CDRs for multiple data rates, it is important to understand the input reference clock requirements. This information helps you to efficiently create the clocking scheme for reconfiguration and to reuse the **.mif** files across all channels in the device. This section describes the clocking enhancements and the implications of using input clocks from various clock sources.

The available clock inputs appear as a `pll_inclk_rx_cruclk[]` port and can be provided from the inter-transceiver block lines (also known as ITB lines), from the global clock networks that are driven by an input pin or by a PLL cascade clock.

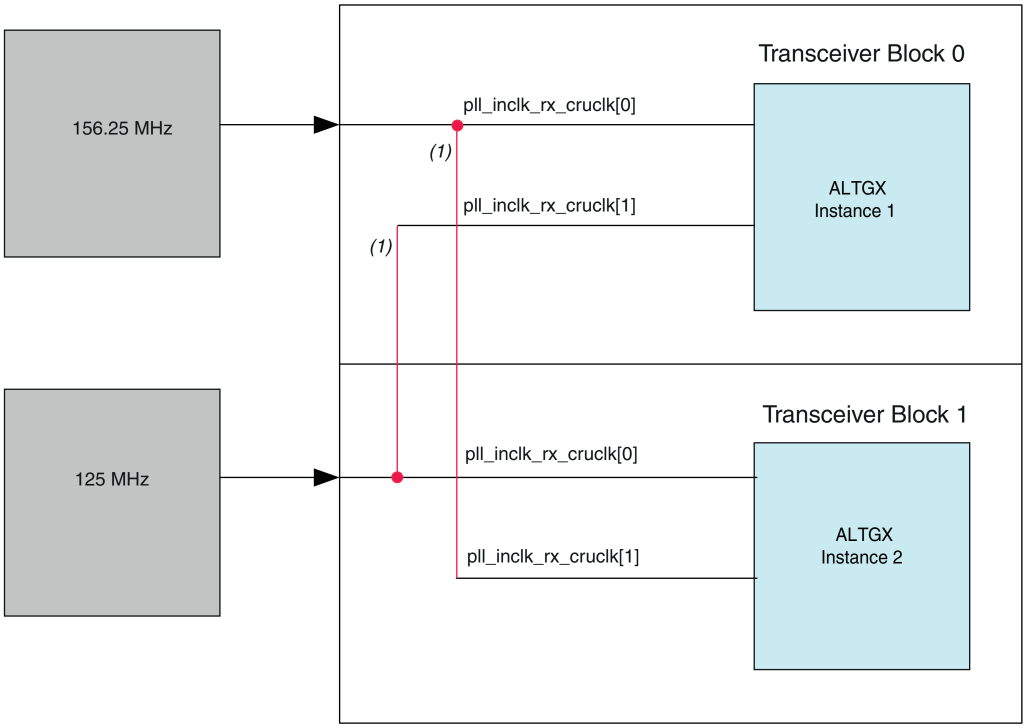
For more information about input reference clocking, refer to the "Input Reference Clocking" section of the *Transceiver Clocking in Arria II Devices* chapter in volume 2 of the *Arria II Device Handbook*.

The following section describes the clocking requirements to reuse **.mif** files.

The **.mif** contains information about the input clock multiplexer settings and the functional blocks that you selected during the ALTGX MegaWizard Plug-In Manager instantiation. You can use a **.mif** to dynamically reconfigure any of the other transceiver channels in the device if the order of the clock inputs is consistent. For example, assume that a **.mif** is generated for a transceiver channel in transceiver block 0 and the input clock source is connected to the `pll_inclk_rx_cruclk[0]` port. When you use the generated **.mif** for a channel in other transceiver blocks (for example, transceiver block 1), the you must connect the same clock source to the `pll_inclk_rx_cruclk[0]` port. Figure 35 and Figure 36 show the incorrect and correct order of input reference clocks, respectively.

In Figure 35, the clocking is incorrect when reusing the **.mif** because the input reference clock is not connected to the corresponding `pll_inclk_rx_cruclk[]` ports in the two instances.

Figure 35. Incorrect Input Reference Clock Connections When Reusing a .mif

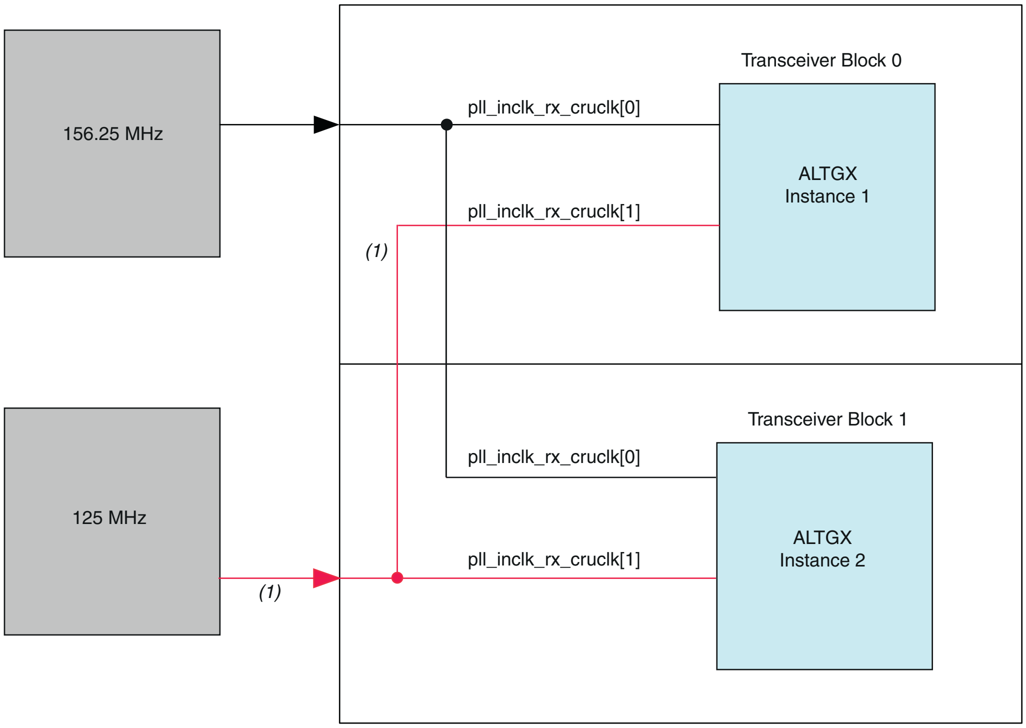


Note to Figure 35:

(1) The red lines represent the alternate source of REFCLK.

Figure 36 shows the correct input reference clock connections when reusing a **.mif**.

Figure 36. Correct Input Reference Clock Connections When Reusing a .mif



Note to Figure 36:

(1) The red lines represent the alternate source of REFCLK.

You can only reuse the **.mif** generated for a transceiver channel on one side of the device for a transceiver channel on the other side of device if the input reference clock frequencies and order of the `pll_inclk_rx_cruclk[]` ports in the ALTGX instances on both sides are identical.

In addition to the input reference clock requirements when reusing a **.mif**, refer to "Guidelines for the logical_tx_pll_sel and logical_tx_pll_sel_en Ports" on page 72 for additional ways to reuse a **.mif**.

Guidelines for the logical_tx_pll_sel and logical_tx_pll_sel_en Ports

This section describes when to enable the `logical_tx_pll_sel` and `logical_tx_pll_sel_en` ports and how to use them in the following dynamic reconfiguration modes:

- Channel and CMU PLL reconfiguration mode
- Channel reconfiguration with transmitter PLL select mode
- CMU PLL reconfiguration mode

These are optional input ports to the ALTGX_RECONFIG instance.

Table 10 on page 74 lists the conditions under which the dynamic reconfiguration controller uses either the `logical_tx_pll_sel` port value or the logical reference index value stored in the **.mif**.

Figure 37 shows the `logical_tx_pll_sel` and `logical_tx_pll_sel_en` ports.

Figure 37. Using logical_tx_pll_sel and logical_tx_pll_sel_en Ports

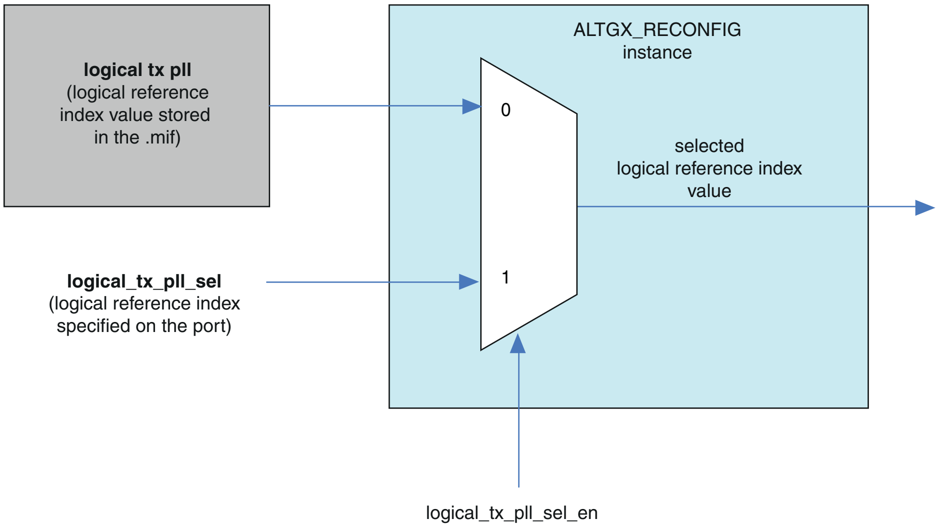

logical tx pll
(logical reference index value stored in the .mif)
ALTGX_RECONFIG instance
0
1
selected logical reference index value
logical_tx_pll_sel
(logical reference index specified on the port)
logical_tx_pll_sel_en

Figure 38 shows the required signal transitions to reconfigure the CMU PLL with a `logical_tx_pll` value of **1**. Keep the `logical_tx_pll_sel` and `logical_tx_pll_sel_en` signals at a constant logic level until the dynamic reconfiguration controller asserts the `channel_reconfig_done` signal.

Figure 38. Signal Transitions to Reconfigure a Transmitter PLL with a Reference Index Value of 1

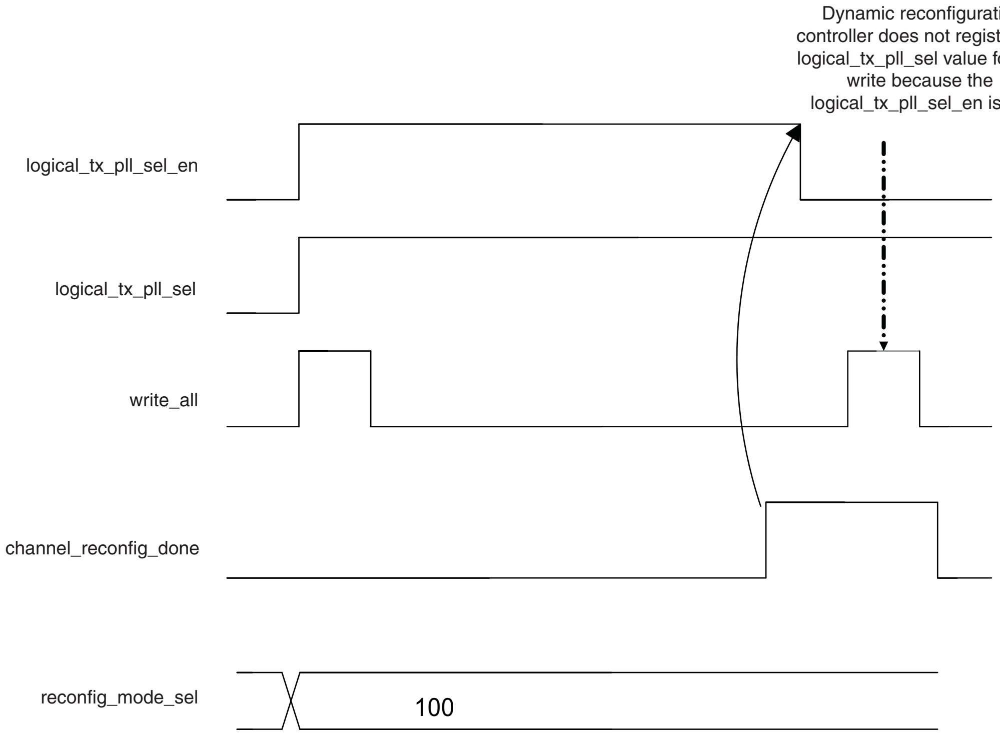

Dynamic reconfiguration controller does not register the logical_tx_pll_sel value for this write because the logical_tx_pll_sel_en is low
logical_tx_pll_sel_en
logical_tx_pll_sel
write_all
channel_reconfig_done
reconfig_mode_sel
100

Table 10 lists how the dynamic reconfiguration controller selects between the logical reference index stored in the **.mif** (logical `tx pll`) and the logical reference index specified at the `logical_tx_pll_sel` port.

Table 10. Various Combinations of the logical_tx_pll_sel and logical_tx_pll_sel_en Ports

logical_tx_pll_sel	logical_tx_pll_sel_en	Logical Reference Index Value Selected by the ALTGX_RECONFIG Instance
Enabled	Enabled and value is 1	Value on the `logical_tx_pll_sel` port
Enabled	Enabled and value is 0	Logical reference index value stored in the **.mif** (logical `tx pll`)
Enabled	Disabled	Value on the `logical_tx_pll_sel` port
Enabled	Disabled	Logical reference index value stored in the **.mif** (logical `tx pll`)

When you configure a transceiver channel in the ALTGX MegaWizard Plug-In Manager, Altera recommends keeping track of the transmitter PLL that drives the channel.

☞ The `logical_tx_pll_sel` port does not modify the transceiver settings on the receiver side.

If you enable both the `logical_tx_pll_sel` and `logical_tx_pll_sel_en` ports, reconfigure the transmitter PLL. Keep the `logical_tx_pll_sel` and `logical_tx_pll_sel_en` signals at a constant logic level until the dynamic reconfiguration controller asserts the `channel_reconfig_done` signal.

Table 11 lists the two conditions under which you can reuse the **.mif** files when using the `logical_tx_pll_sel` and `logical_tx_pll_sel_en` ports.

Table 11. Two Conditions Under Which You can Reuse .mif Files (logical_tx_pll_sel and logical_tx_pll_sel_en)

Condition 1: Reuse the **.mif** created for one CMU PLL on the other CMU PLL of the same transceiver block.		**Condition 2:** Reuse the **.mif** created for one transmitter PLL on the transmitter PLL of another transceiver block.	
Channel and CMU PLL Reconfiguration and CMU PLL Reconfiguration	**Channel Reconfiguration with Transmitter PLL Select**	**Channel and CMU PLL Reconfiguration and CMU PLL Reconfiguration**	**Channel Reconfiguration with Transmitter PLL Select**
Assume that you create a **.mif** containing the desired ALTGX settings to reconfigure the CMU0 PLL. Assume that the logical reference index you assigned to CMU0 PLL is **0**. ■ You can reuse this **.mif** created for CMU0 PLL on CMU1 PLL of the same transceiver block if you want to reconfigure CMU1 PLL to the new data rate information stored in the **.mif**. ■ You must set `logical_tx_pll_sel` to the logical reference index of CMU1 PLL (**1'b1**) and `logical_tx_pll_sel_en` to **1'b1** and then write this **.mif** into the transceiver channel. By doing so, the dynamic reconfiguration controller overwrites the logical `tx pll` value stored in the **.mif** with the logical reference index of CMU1 PLL.	Assume that the transceiver channel listens to CMU1 PLL and the logical reference index assigned to it is **0**. ■ Generate a **.mif** for these settings. ■ When you use channel reconfiguration with transmitter PLL select mode and reconfigure the transceiver channel with this **.mif**, the transceiver channel is reconfigured to listen to CMU1 PLL. ■ If you want to reconfigure the transceiver channel to listen to CMU0 PLL instead, you can reuse this **.mif**. ■ You must set `logical_tx_pll_sel` to the logical reference index of CMU0 PLL (**1'b1**) and `logical_tx_pll_sel_en` to **1'b1** and then write this **.mif** into the transceiver channel.	Assume that you create a **.mif** containing the desired ALTGX settings to reconfigure the transmitter PLL of a transceiver block. Assume that the logical reference of the transmitter PLL is **1**. ■ You can reuse this **.mif** created to reconfigure the transmitter PLL of another transceiver block under the following condition: ■ You want to reconfigure the transmitter PLL of the other transceiver block to exactly the same data rate information stored in the **.mif**. ■ You must set `logical_channel_address` to the logical channel address of the transmitter PLL you intend to reconfigure.	Assume that you create a **.mif** containing the logical reference index of the transmitter PLL that the reconfigured transceiver channel needs to listen to. ■ Assume that the transmitter PLL used is CMU0 PLL and the logical reference index assigned is **0**. ■ When you use channel reconfiguration with transmitter PLL select mode and reconfigure the transceiver channel with this **.mif**, the transceiver channel is reconfigured to listen to CMU0 PLL. ■ If you want to reconfigure this transceiver channel to listen to another transmitter PLL outside the transceiver block, you can reuse this **.mif**, provided the intended data rate is the same.

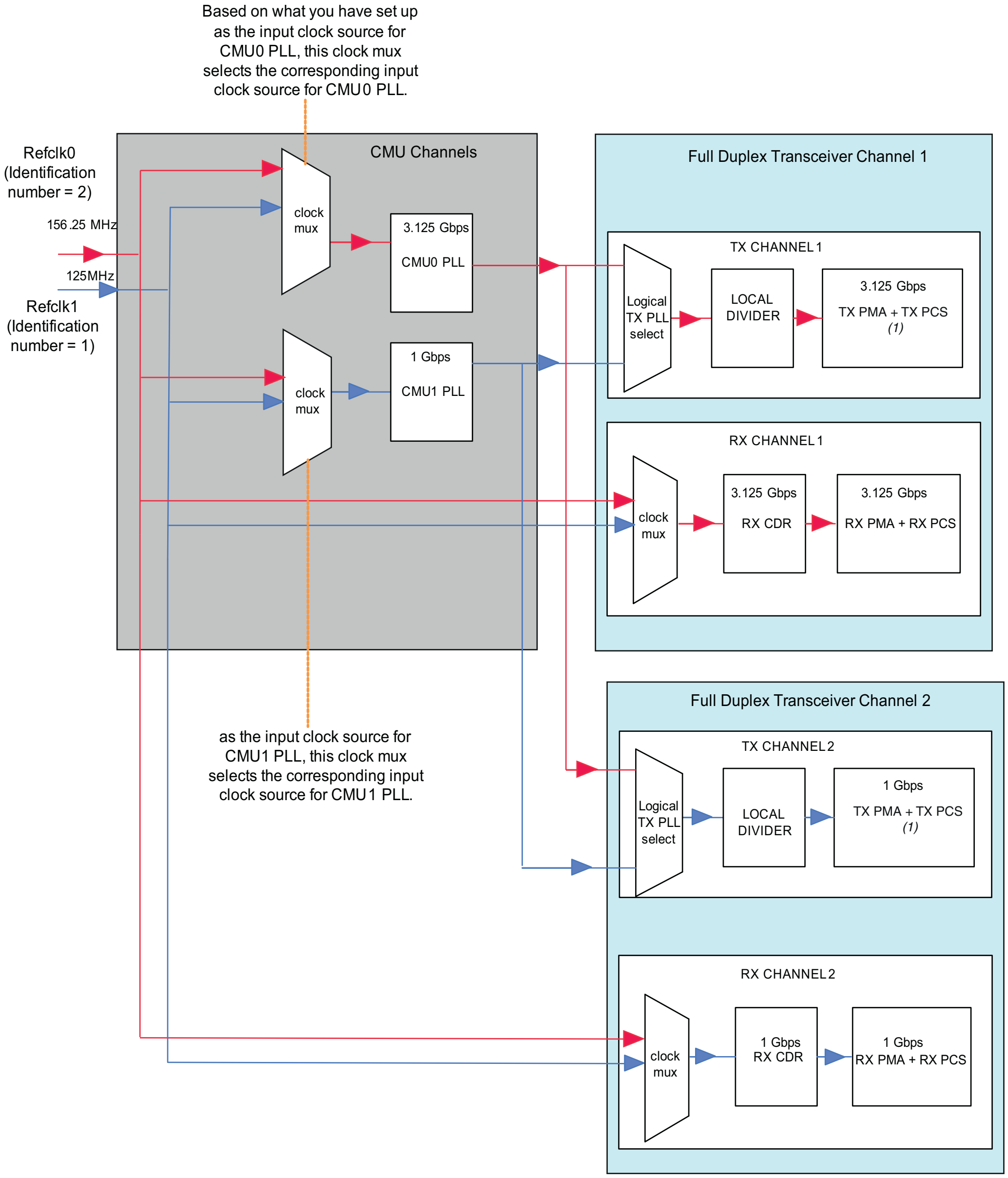
Guidelines for Specifying the Input Reference Clocks

The following are guidelines for setting up the input reference clocks in the **Reconfiguration Settings** screen of the ALTGX MegaWizard Plug-In Manager.

- Assign the identification numbers to all input reference clocks that are used by the transmitter PLLs in their corresponding PLL screens. You can set up a maximum of 10 input reference clocks and assign identification numbers from 1 to 10.
- Keep the identification numbers consistent for all the **.mif** files generated in the design.
- Maintain the input reference clock frequencies settings for all the **.mif** files.

Figure 39 shows an example scenario where the input reference clocks connections to the transceiver channels are based on what you set as the input clock source for each of the CMU transmitter PLLs within a transceiver block.

Figure 39. Input Reference Clocks Connections to the Transceiver Channels


Based on what you have set up as the input clock source for CMU0 PLL, this clock mux selects the corresponding input clock source for CMU0 PLL.
Refclk0 (Identification number = 2)
156.25 MHz
125MHz
Refclk1 (Identification number = 1)
CMU Channels
clock mux
3.125 Gbps
CMU0 PLL
clock mux
1 Gbps
CMU1 PLL
as the input clock source for CMU1 PLL, this clock mux selects the corresponding input clock source for CMU1 PLL.
Full Duplex Transceiver Channel 1
TX CHANNEL 1
Logical TX PLL select
LOCAL DIVIDER
3.125 Gbps
TX PMA + TX PCS (1)
RX CHANNEL 1
clock mux
3.125 Gbps
RX CDR
3.125 Gbps
RX PMA + RX PCS
Full Duplex Transceiver Channel 2
TX CHANNEL 2
Logical TX PLL select
LOCAL DIVIDER
1 Gbps
TX PMA + TX PCS (1)
RX CHANNEL 2
clock mux
1 Gbps RX CDR
1 Gbps RX PMA + RX PCS

Note to Figure 39:

(1) Depending on the mode you select, the PCS unit may or may not be present.

Dynamic Reconfiguration Duration

Dynamic reconfiguration duration is the number of cycles the busy signal is asserted when the dynamic reconfiguration controller performs write transactions, read transactions, or offset cancellation of the receiver channels.

PMA Controls Reconfiguration Duration

The following section shows an example estimate of the number of reconfig_clk clock cycles the busy signal is asserted during PMA controls reconfiguration using Method 1, Method 2, or Method 3.

PMA Controls Reconfiguration Duration When Using Method 1

The logical_channel_address port is used in "Method 1: Using logical_channel_address to Reconfigure Specific Transceiver Channels" on page 19. The write and read transaction durations are described in the following sections.

Write Transaction Duration

When writing values to the following PMA controls, the busy signal is asserted for 260 reconfig_clk clock cycles for each of these controls:

- tx_preemp (pre-emphasis control first post-tap)
- tx_vodctrl (voltage output differential)
- rx_eqctrl (equalizer control)
- rx_eqdcgain (equalizer DC gain)

Read Transaction Duration

When reading the existing values of the following PMA controls, the busy signal is asserted for 130 reconfig_clk clock cycles for each of these controls. The data_valid signal is then asserted after the busy signal goes low.

- tx_preemp_out (pre-emphasis control first post-tap)
- tx_vodctrl_out (voltage output differential)
- rx_eqctrl_out (equalizer control)
- rx_eqdcgain_out (equalizer DC gain)

PMA Controls Reconfiguration Duration When Using Method 2 or Method 3

The logical_channel_address port is not used in "Method 2: Writing the Same Control Signals to Control All the Transceiver Channels" on page 21 or "Method 3: Writing Different Control Signals for all the Transceiver Channels at the Same Time" on page 24. The write and read transaction durations are described in the following sections.

Write Transaction Duration

When writing values to the following PMA controls, the busy signal is asserted for 260 reconfig_clk clock cycles per channel for each of these controls:

- tx_preemp (pre-emphasis control first post-tap)
- tx_vodctrl (voltage output differential)
- rx_eqctrl (equalizer control)
- rx_eqdcgain (equalizer DC gain)

Read Transaction Duration

When reading the existing values of the following PMA controls, the busy signal is asserted for 130 reconfig_clk clock cycles per channel for each of these controls. The data_valid signal is then asserted after the busy signal goes low.

- tx_preemp_out (pre-emphasis control first post-tap)
- tx_vodctrl_out (voltage output differential)
- rx_eqctrl_out (equalizer control)
- rx_eqdcgain_out (equalizer DC gain)

Offset Cancellation Duration

When the device powers up, the busy signal remains low for the first reconfig_clk clock cycle. Offset cancellation control is only for the receiver channels. The ALTGX_RECONFIG instance takes approximately 18,307 reconfig_clk clock cycles per channel for **Receiver-only** and **Receiver and Transmitter** channels. It takes approximately 877 reconfig_clk clock cycles per channel for **Transmitter-only** channels to determine if the channel under reconfiguration is a receiver channel or not. The ATLGX_RECONFIG instance requires an additional 130,000 clock cycles for these values to take effect. The ALTGX_RECONFIG instance takes approximately two reconfig_clk clock cycles per channel for the unused logical channels.

For example, to demonstrate offset cancellation duration, assume the following:

- One ALTGX_RECONFIG instance is connected to two ALTGX instances.
- ALTGX Instance 1 has one **Transmitter-only** channel (logical_channel_address = 0).
- ALTGX Instance 2 has one **Receiver-only** channel (logical_channel_address = 4).

For this example, the ALTGX_RECONFIG instance consumes the following number of reconfig_clk clock cycles for offset cancellation:

- 877 cycles for the **Transmitter-only** channel
- 18,307 cycles for the **Receiver-only** channel
- Two cycles each for non-existent channels with logical_channel_addresses = 1, 2, and 3. 130,000 cycles as a baseline for the values to take affect

The offset cancellation duration for the ALTGX_RECONFIG instance to reconfigure the **Transmitter-only** channel, **Receiver-only** channel, non-existent logical channels 1, 2, and 3 = 149,190 cycles (877 +18,307 +6 + 130,000).

Dynamic Reconfiguration Duration for Channel and Transmitter PLL Select/Reconfig Modes

Table 12 lists the number of `reconfig_clk` clock cycles it takes for the dynamic reconfiguration controller to reconfigure various parts of the transceiver channel and CMU PLL.

Table 12. Dynamic Reconfiguration Duration for Transceiver Channel and CMU PLL Reconfiguration

Transceiver Portion Under Reconfiguration	Number of reconfig_clk Clock Cycles
Transmitter channel reconfiguration	1,518
Receiver channel reconfiguration	5,255
Transmitter and receiver channel reconfiguration	6,762
CMU PLL only reconfiguration	863
Transmitter channel and CMU PLL reconfiguration	2,370
Transceiver channel and CMU PLL reconfiguration	7,614
Central control unit reconfiguration	927

Reduced .mif Reconfiguration

This mode is available only for **.mif**-based transceiver channel reconfiguration modes.

This is an optional feature that allows faster reconfiguration and faster simulation time. For example, if you intend to make minor changes to the transceiver channel, this might involve a change of only a few words in the **.mif**.

The following example changes only the termination setting:

- Assume that the only word difference is word address 32.
- Instead of loading the entire **.mif**, you can use **altgx_diffmifgen.exe** to generate a new **.mif**. This new **.mif** only contains the modified words.
- The new **.mif** is 22 bits wide, compared with 16 bits wide in the regular **.mif**.
- There are 6 bits of address in addition to 16 bits of data:

  <addr 6 bits> <data 16 bits>
- Enable the **Use 'reconfig_address' to input address from the .mif in reduced .mif reconfiguration** option in the **Channel and TX PLL Reconfiguration** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager.
- Use the `reconfig_data[15:0]` port to connect the 16 bits of data from the new **.mif**.
- Use the `reconfig_address[5:0]` port to connect the 6 bits of address from the new **.mif**.

Using altgx_diffmifgen.exe

Browse to the project directory in which the Quartus II software is installed. For example, **altgx_diffmifgen.exe** is available in the following directory:

C:\altera\10.1\quartus\bin

The syntax for using this **.exe** is as follows:

altgx_diffmifgen.exe <*a.mif*> <*b.mif*>

The **altgx_diffmifgen.exe** requires two or more ALTGX **.mif** files.

Combining Transceiver Channels with Dynamic Reconfiguration Enabled

You can combine the transceiver channels in a design into the same physical transceiver block by assigning the `tx_dataout` and `rx_datain` pins of the channels to the same transceiver block.

Requirements

When you enable dynamic reconfiguration, the Quartus II software has certain requirements for combining multiple transceiver channels in the same physical transceiver block:

- All channels that you want to combine in the same transceiver block must have the same options enabled on the **Reconfiguration Settings** tab of the ALTGX MegaWizard Plug-In Manager. For example, when you enable the **Analog controls (VOD, pre-emphasis, and manual equalization)** option, you must enable the same option for all the other channels to be combined.
- Assign transceiver pins to the same group using the GXB TX PLL Reconfiguration Group assignment.
- When combining a **Transmitter-only** channel with a **Receiver-only** channel, both must go through a reset sequence, even if the transmitter or receiver is reconfigured.
- Combining channels does not affect the logical channel address of the combined channel. For example, before combining, `logical_channel_address 0` is a **Transmitter-only** channel and `logical_channel_address 4` is an **Receiver-only** channel. After combining, the addresses remain unchanged, although they are physically the same channel.
- All channels must be controlled by the same ALTGX_RECONFIG (dynamic reconfiguration controller) instance. The transceiver channels connected to multiple ALTGX_RECONFIG instances cannot be combined into the same physical transceiver block, even if they are configured to the same functional mode and data rate.

Document Revision History

Table 13 lists the revision history for this application note.

Table 13. Document Revision History (Part 1 of 2)

Date	Version	Changes
August 2016	3.8	■ Added an instruction for what to do following a `rate_switch_ctrl` reconfiguration to the “Data Rate Division in TX—Write Transaction” section.
February 2015	3.7	■ Updated the values for `tx_preemp` in Table 4.
October 2013	3.6	■ Updated the “Stage V: Generate a .mif for Channel and TX PLL select/reconfig” section.
June 2013	3.5	■ Updated Table 4, Table 5. ■ Updated the “Using a GPLL to Drive the reconfig_clk Port” section.
July 2012	3.4	■ Added the Feedback icon. ■ Corrected “tx_datainfull[32:0]” to “tx_datainfull[43:0]”, and “rx_dataoutfull[47:0]” to “rx_dataoutfull[63:0]”, to include the 44-bits and 64-bits channel width in Arria II GZ.
November 2011	3.3	■ Added the “Using a GPLL to Drive the reconfig_clk Port” section. ■ Updated the “Write Transaction” section of “Method 1: Using logical_channel_address to Reconfigure Specific Transceiver Channels”. ■ Updated the “Offset Cancellation for Receiver Channels Operation” and “Channel Interface” sections. ■ Updated Table 4, Table 5, Table 6, and Table 7. ■ Converted document to the new template. ■ Minor text edits.
July 2011	3.2	■ Updated the “Reconfiguration Modes”, “Option 3: Use the Respective Channel Receiver Core Clocks” and “Central Control Unit Reconfiguration Mode” sections. ■ Minor text edits.
December 2010	3.1	■ Updated to include Arria II GZ information. ■ Added Table 5. ■ Updated the “PMA Controls Reconfiguration Duration When Using Method 1”, “PMA Controls Reconfiguration Duration When Using Method 2 or Method 3”, and “Using altgx_diffmifgen.exe” sections. ■ Converted to the new template. ■ Minor text edits.
July 2010	3.0	■ Updated Table 4 (the tx_vodctrl[2:0] section). ■ Updated the “General”, “Central Control Unit Reconfiguration Mode”, and “Stage V: Generate a .mif for Channel and TX PLL select/reconfig” sections. ■ Removed Figure 11. ■ Minor text edits.
January 2010	2.1	■ Added Central Control Unit reconfiguration clock cycles to Table 9.

Table 13. Document Revision History (Part 2 of 2)

Date	Version	Changes
January 2010	2.0	■ Added new sections Setup Guide, Important Considerations, Central Control Unit Reconfiguration Mode, and Reduced .mif Reconfiguration ■ Reorganized ■ Text edits throughout
March 2009	1.0	Initial release.