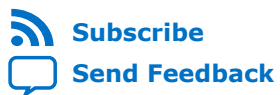




External Memory Interfaces Intel® Stratix® 10 FPGA IP Design Example User Guide

Updated for Intel® Quartus® Prime Design Suite: **21.1**

IP Version: **19.2.3**



[Subscribe](#)

[Send Feedback](#)

UG-20120 | 2021.03.29

Latest document on the web: [PDF](#) | [HTML](#)

Contents

| | |
|--|-----------|
| 1. Design Example Quick Start Guide for External Memory Interfaces Intel® Stratix® 10 FPGA IP | 3 |
| 1.1. Creating an EMIF Project..... | 4 |
| 1.2. Generating and Configuring the EMIF IP..... | 6 |
| 1.2.1. Intel Stratix 10 EMIF Parameter Editor Guidelines..... | 7 |
| 1.3. Generating the Synthesizable EMIF Design Example..... | 7 |
| 1.4. Generating the EMIF Design Example for Simulation..... | 10 |
| 1.5. Simulation Versus Hardware Implementation..... | 13 |
| 1.6. Simulating External Memory Interface IP With ModelSim..... | 14 |
| 1.7. Pin Placement for Intel Stratix 10 EMIF IP..... | 14 |
| 1.8. Compiling and Programming the Intel Stratix 10 EMIF Design Example..... | 17 |
| 1.9. Debugging the Intel Stratix 10 EMIF Design Example..... | 18 |
| 2. Design Example Description for External Memory Interfaces Intel Stratix 10 FPGA IP | 19 |
| 2.1. Synthesis Example Design..... | 19 |
| 2.2. Simulation Example Design..... | 21 |
| 2.3. Example Designs Interface Tab..... | 22 |
| 3. Document Revision History for External Memory Interfaces Intel Stratix 10 FPGA IP Design Example User Guide..... | 23 |

1. Design Example Quick Start Guide for External Memory Interfaces Intel® Stratix® 10 FPGA IP

A new interface and more automated design example flow is available for Intel® Stratix® 10 external memory interfaces.

The **Example Designs** tab in the parameter editor allows you to specify the creation of synthesis and simulation file sets which you can use to validate your EMIF IP.

You can generate an example design specifically for an Intel FPGA development kit, or for any EMIF IP that you generate.

Figure 1. General Design Example Workflows

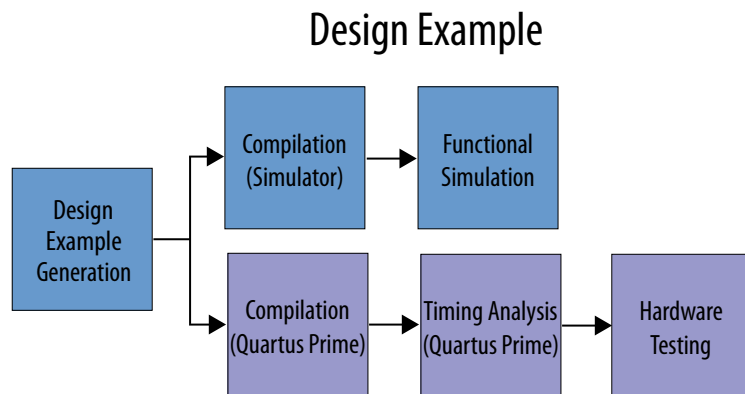
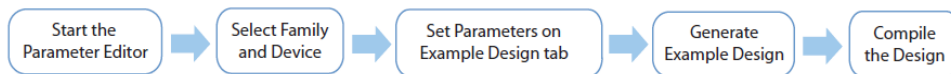


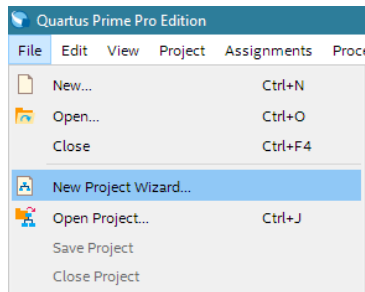
Figure 2. Generating an EMIF Example Design With an Intel Stratix 10 Development Kit



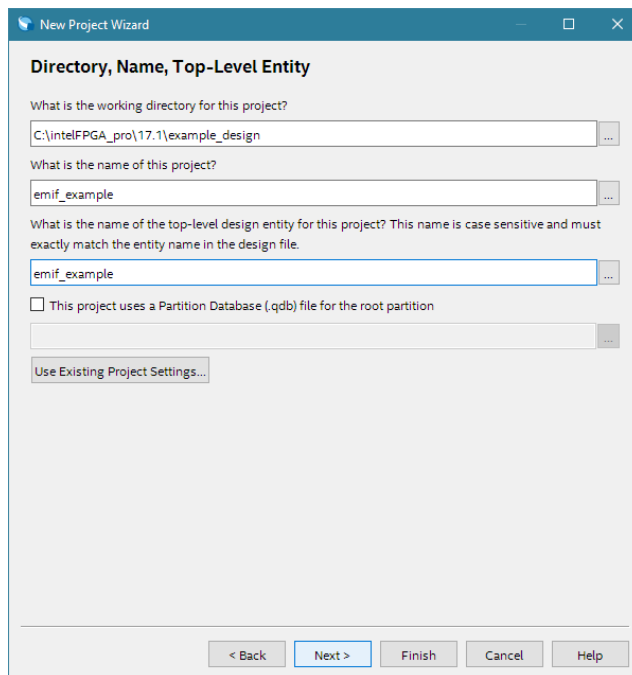
1.1. Creating an EMIF Project

For the Intel Quartus® Prime software version 17.1 and later, you must create an Intel Quartus Prime project before generating the EMIF IP and design example.

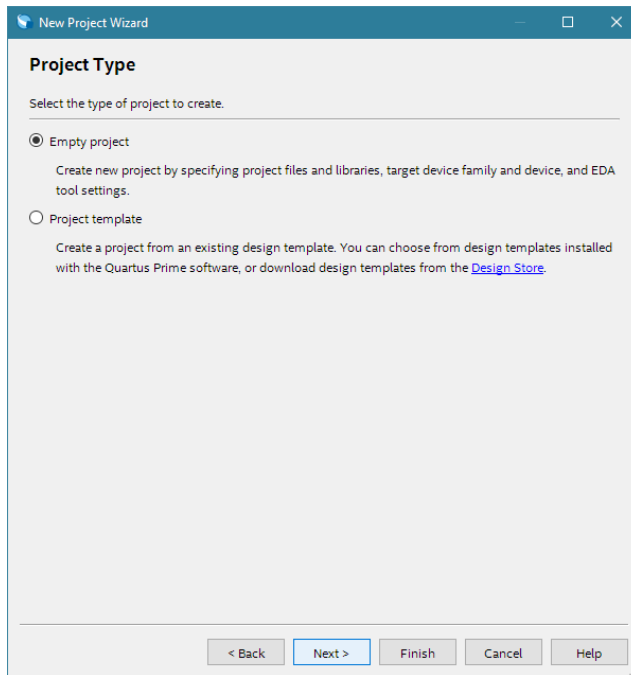
1. Launch the Intel Quartus Prime software and select **File ► New Project Wizard**. Click **Next**.



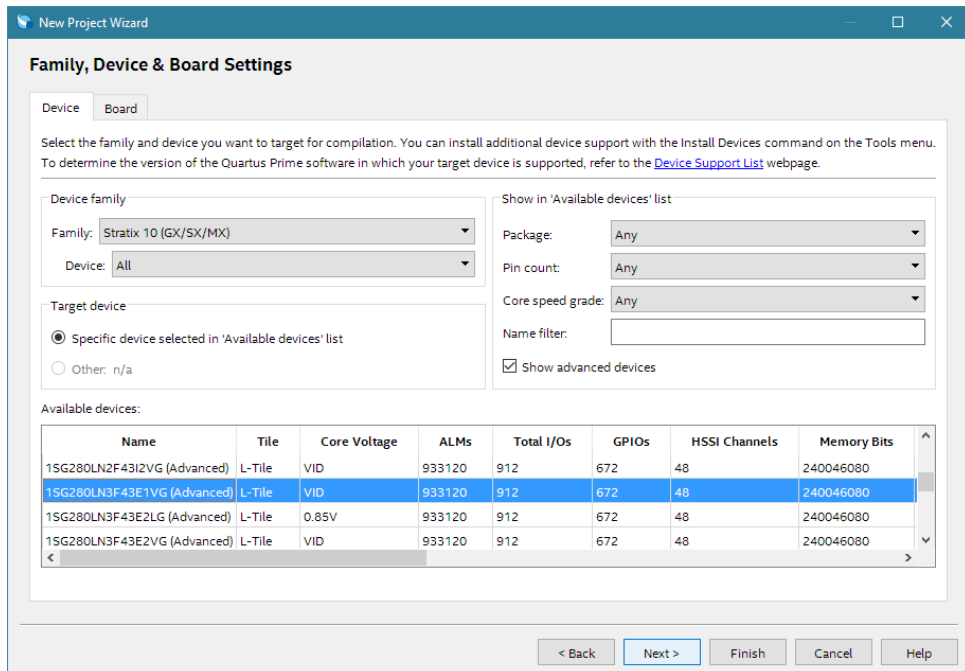
2. Specify a directory and name for the project that you want to create. Click **Next**.



3. Verify that **Empty Project** is selected. Click **Next** two times.



4. Under **Name filter**, type the device part number.
5. Under **Available devices**, select the appropriate device.

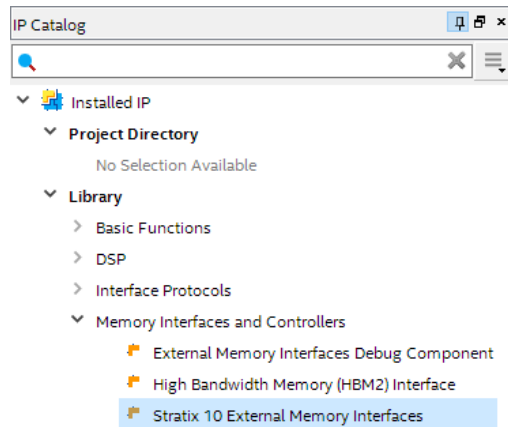


6. Click **Finish**.

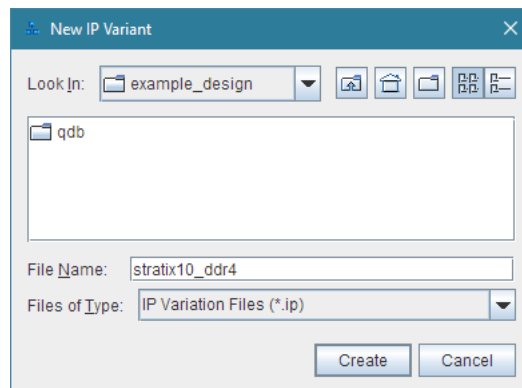
1.2. Generating and Configuring the EMIF IP

The following steps illustrate how to generate and configure the EMIF IP. This walkthrough creates a DDR4 interface, but the steps are similar for other protocols.

1. In the **IP Catalog window**, select **Intel Stratix 10 External Memory Interfaces**. (If the **IP Catalog window** is not visible, select **View > Utility Windows > IP Catalog**.)



2. In the **IP Parameter Editor**, provide an entity name for the EMIF IP (the name that you provide here becomes the file name for the IP) and specify a directory. Click **Create**.



3. The parameter editor has multiple tabs where you must configure parameters to reflect your EMIF implementation:

1.2.1. Intel Stratix 10 EMIF Parameter Editor Guidelines

Table 1. EMIF Parameter Editor Guidelines

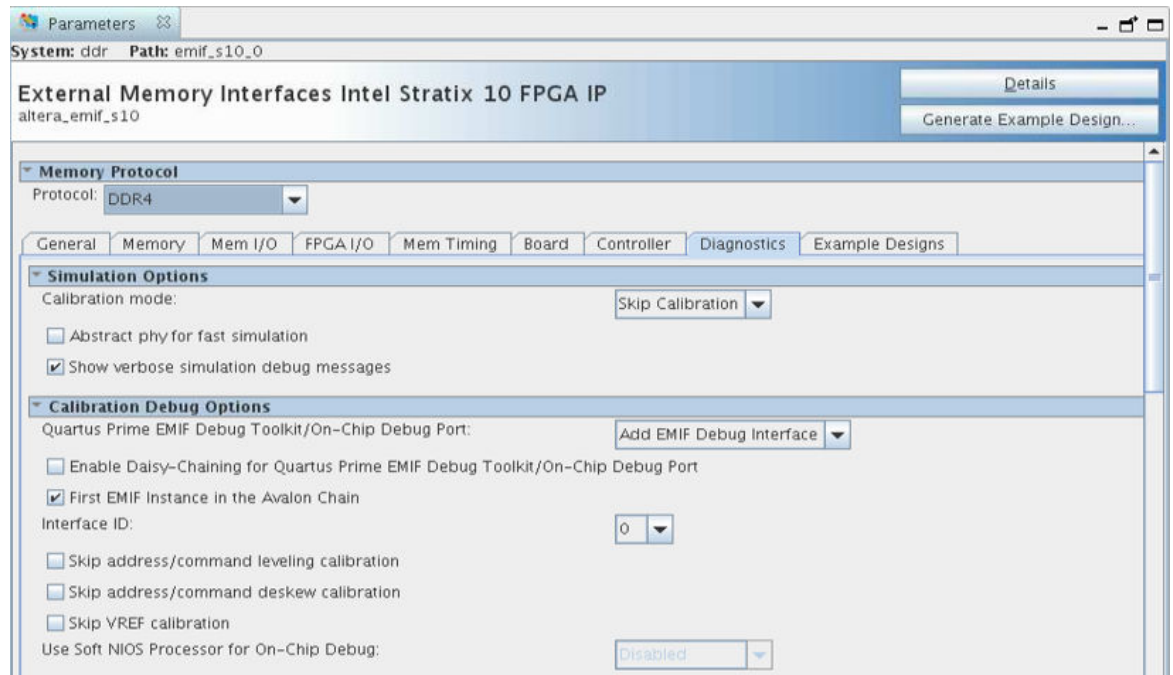
| Parameter Editor Tab | Guidelines |
|------------------------|---|
| General | Ensure that the following parameters are entered correctly: <ul style="list-style-type: none"> The speed grade for the device. The memory clock frequency. The PLL reference clock frequency. |
| Memory | <ul style="list-style-type: none"> Refer to the data sheet for your memory device to enter the parameters on the Memory tab. You should also enter a specific location for the ALERT# pin. (Applies to DDR4 memory protocol only.) |
| Mem I/O | <ul style="list-style-type: none"> For initial project investigations, you may use the default settings on the Mem I/O tab. For advanced design validation, you should perform board simulation to derive optimal termination settings. |
| FPGA I/O | <ul style="list-style-type: none"> For initial project investigations, you may use the default settings on the FPGA I/O tab. For advanced design validation, you should perform board simulation with associated IBIS models to select appropriate I/O standards. |
| Mem Timing | <ul style="list-style-type: none"> For initial project investigations, you may use the default settings on the Mem Timing tab. For advanced design validation, you should enter parameters according to your memory device's data sheet. |
| Board | <ul style="list-style-type: none"> For initial project investigations, you may use the default settings on the Board tab. For advanced design validation and accurate timing closure, you should perform board simulation to derive accurate intersymbol interference (ISI)/ crosstalk and board and package skew information, and enter it on the Board tab. |
| Controller | Set the controller parameters according to the desired configuration and behavior for your memory controller. |
| Diagnostics | You can use the parameters on the Diagnostics tab to assist in testing and debugging your memory interface. |
| Example Designs | The Example Designs tab lets you generate design examples for synthesis and for simulation. The generated design example is a complete EMIF system consisting of the EMIF IP and a driver that generates random traffic to validate the memory interface. |

For detailed information on individual parameters, refer to the appropriate chapter for your memory protocol in the *Intel Stratix 10 External Memory Interfaces IP User Guide*.

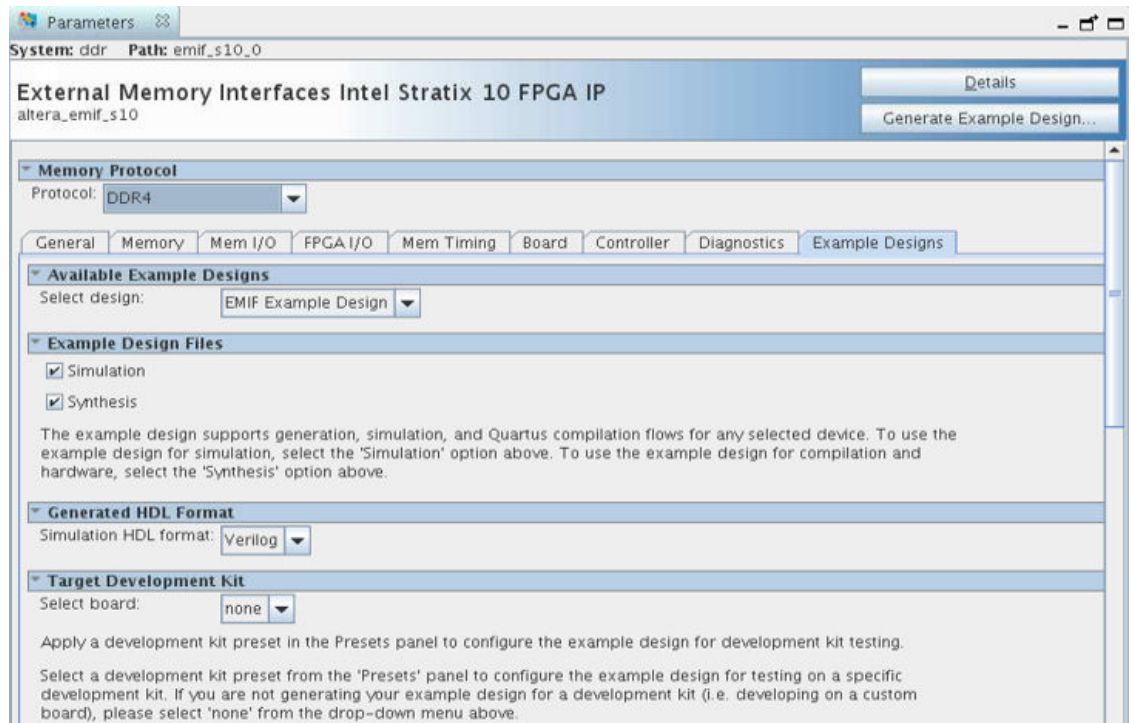
1.3. Generating the Synthesizable EMIF Design Example

For the Intel Stratix 10 development kit, it is sufficient to leave most of the Intel Stratix 10 EMIF IP settings at their default values. To generate the synthesizable design example, follow these steps:

1. On the **Diagnostics** tab, enable the **EMIF Debug Toolkit/On-Chip Debug Port** and **In-System-Sources-and-Probes** to provide access to the available debugging features.

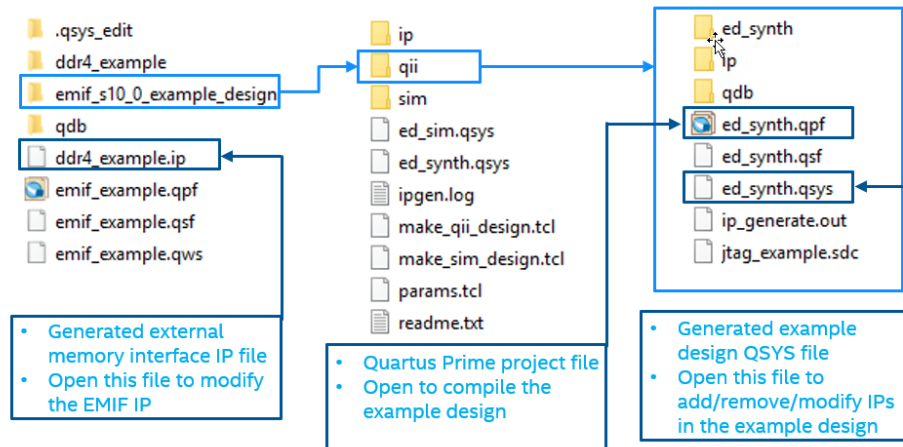


2. On the **Example Designs** tab, ensure that the **Synthesis** box is checked.
3. Configure the EMIF IP and click **Generate Example Design** in the upper-right corner of the window.



- Specify a directory for the EMIF design example and click **OK**. Successful generation of the EMIF design example creates the following fileset under a `qii` directory.

Figure 3. Generated Synthesizable Design Example File Structure



Note: If you don't select the **Simulation** or **Synthesis** checkbox, the destination directory will contain Platform Designer design files, which are not compilable by the Intel Quartus Prime software directly, but can be viewed or edited under the Platform Designer. In this situation you can run the following commands to generate synthesis and simulation file sets.

- To create a compilable project, you must run the `quartus_sh -t make_qii_design.tcl` script in the destination directory.
- To create a simulation project, you must run the `quartus_sh -t make_sim_design.tcl` script in the destination directory.

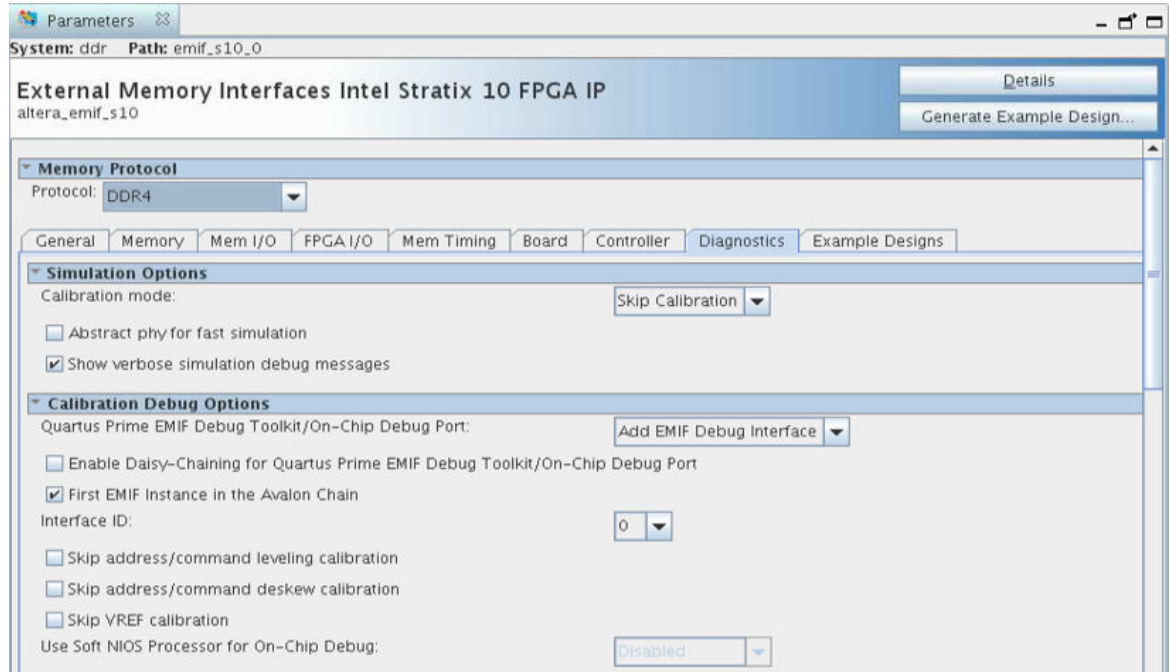
Related Information

- [Synthesis Example Design](#) on page 19
- [Intel Stratix 10 EMIF IP Parameter Descriptions for DDR3](#)
- [Intel Stratix 10 EMIF IP Parameter Descriptions for DDR4](#)
- [Intel Stratix 10 EMIF IP Parameter Descriptions for QDRII/II+/Xtreme](#)
- [Intel Stratix 10 EMIF IP Parameter Descriptions for QDR-IV](#)
- [Intel Stratix 10 EMIF IP Parameter Descriptions for RLDRAM 3](#)

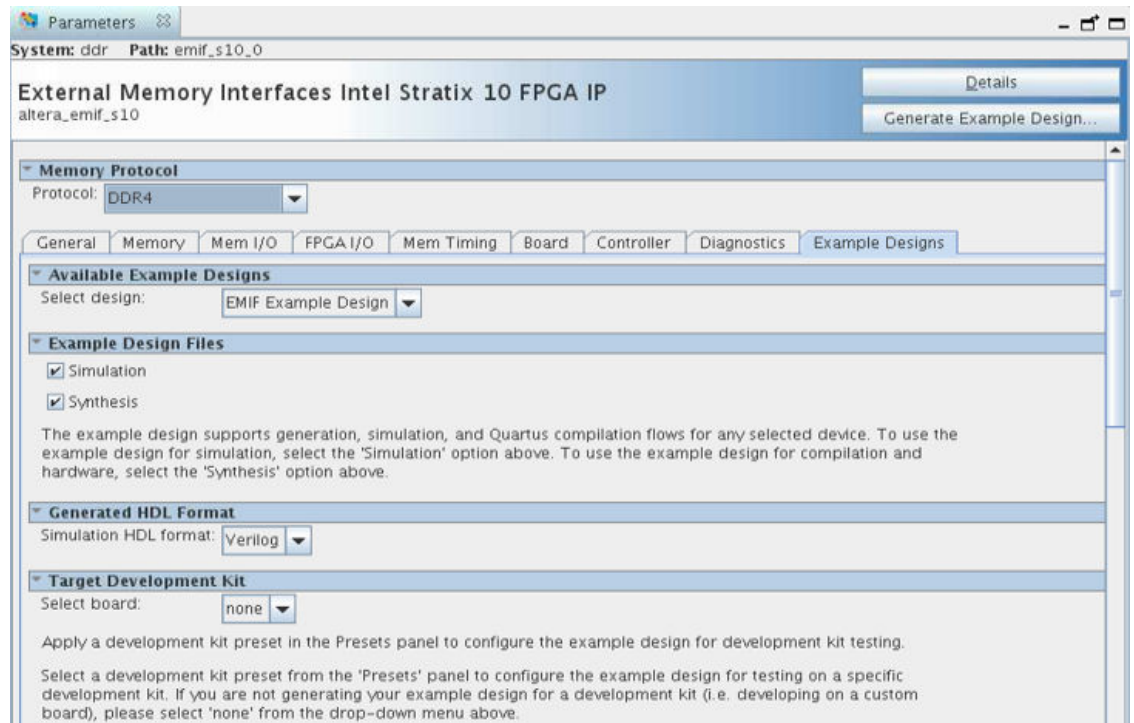
1.4. Generating the EMIF Design Example for Simulation

For the Intel Stratix 10 development kit, it is sufficient to leave most of the Intel Stratix 10 EMIF IP settings at their default values. To generate the design example for simulation, follow these steps:

1. On the **Diagnostics** tab, you can choose between two calibration modes: **Skip Calibration** and **Full Calibration**. (For details on these modes, refer to *Simulation Versus Hardware Implementation*, later in this chapter.) To reduce simulation time, select **Abstract PHY for fast simulation**.



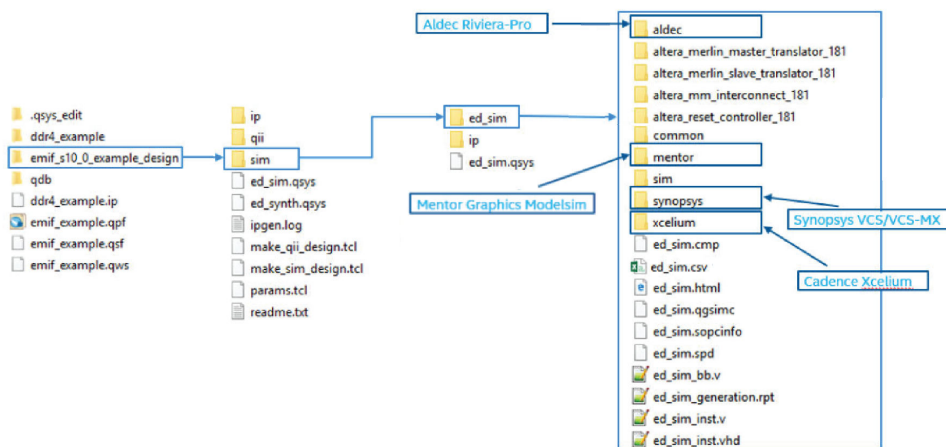
2. On the **Example Designs** tab, ensure that the **Simulation** box is checked. Also choose the required **Simulation HDL format**, either Verilog or VHDL.
3. Configure the EMIF IP and click **Generate Example Design** in the upper-right corner of the window.



4. Specify a directory for the EMIF design example and click **OK**.

Successful generation of the EMIF design example creates multiple file sets for various supported simulators, under a `sim/ed_sim` directory.

Figure 4. Generated Simulation Design Example File Structure



Note: If you don't select the **Simulation** or **Synthesis** checkbox, the destination directory will contain Platform Designer design files, which are not compilable by the Intel Quartus Prime software directly, but can be viewed or edited under the Platform Designer. In this situation you can run the following commands to generate synthesis and simulation file sets.

- To create a compilable project, you must run the `quartus_sh -t make_qii_design.tcl` script in the destination directory.
- To create a simulation project, you must run the `quartus_sh -t make_sim_design.tcl` script in the destination directory.

Related Information

- [Simulation Example Design](#) on page 21
- [Intel Stratix 10 EMIF IP - Simulating Memory IP](#)
- [Simulation Versus Hardware Implementation](#) on page 13

1.5. Simulation Versus Hardware Implementation

For external memory interface simulation, you can select either skip calibration or full calibration on the **Diagnostics** tab during IP generation.

EMIF Simulation Models

This table compares the characteristics of the skip calibration and full calibration models.

Table 2. EMIF Simulation Models: Skip Calibration versus Full Calibration

| Skip Calibration | Full Calibration |
|---|--|
| System-level simulation focusing on user logic. | Memory interface simulation focusing on calibration. |
| Details of calibration are not captured. | Captures all stages of calibration. |
| Has ability to store and retrieve data. | Includes leveling, per-bit deskew, etc. |
| Represents accurate efficiency. | |
| Does not consider board skew. | |

RTL Simulation Versus Hardware Implementation

This table highlights key differences between EMIF simulation and hardware implementation.

Table 3. EMIF RTL Simulation Versus Hardware Implementation

| RTL Simulation | Hardware Implementation |
|---|--|
| Nios® initialization and calibration code execute in parallel. | Nios initialization and calibration code execute sequentially. |
| Interfaces assert <code>cal_done</code> signal signal simultaneously in simulation. | Fitter operations determine the order of calibration, and interfaces do not assert <code>cal_done</code> simultaneously. |

You should run RTL simulations based on traffic patterns for your design's application. Note that RTL simulation does not model PCB trace delays which may cause a discrepancy in latency between RTL simulation and hardware implementation.

1.6. Simulating External Memory Interface IP With ModelSim

This procedure shows how to simulate the EMIF design example.

1. Launch the Mentor Graphics* ModelSim software and select **File > Change Directory**. Navigate to the `sim/ed_sim/mentor` directory within the generated design example folder.
2. Verify that the **Transcript** window is displayed at the bottom of the screen. If the **Transcript** window is not visible, display it by clicking **View > Transcript**.
3. In the **Transcript** window, run **source msim_setup.tcl**.
4. After **source msim_setup.tcl** finishes running, run **ld_debug** in the **Transcript** window.
5. After **ld_debug** finishes running, verify that the **Objects** window is displayed. If the **Objects** window is not visible, display it by clicking **View > Objects**.
6. In the **Objects** window, select the signals that you want to simulate by right-clicking and selecting **Add Wave**.
7. After you finish selecting the signals for simulation, execute **run -all** in the **Transcript** window. The simulation runs until it is completed.
8. If the simulation is not visible, click **View > Wave**.

Related Information

[Intel Stratix 10 EMIF IP - Simulating Memory IP](#)

1.7. Pin Placement for Intel Stratix 10 EMIF IP

This topic provides guidelines for pin placement.

Overview

Intel Stratix 10 FPGAs have the following structure:

- Each device contains between 2 and 3 I/O columns.
- Each I/O column contains up to 12 I/O banks.
- Each I/O bank contains 4 lanes.
- Each lane contains 12 general-purpose I/O (GPIO) pins.

General Pin Guidelines

The following points provide general pin guidelines:

- Ensure that the pins for a given external memory interface reside within a single I/O column.
- Interfaces that span multiple banks must meet the following requirements:
 - The banks must be adjacent to one another. For information on adjacent banks, refer to the *Intel Stratix 10 External Memory Interfaces IP User Guide*.
 - The address and command bank must reside in a center bank to minimize latency. If the memory interface uses an even number of banks, the address and command bank may reside in either of the two center banks.
- Unused pins can be used as general-purpose I/O pins.
- All address and command and associated pins must reside within a single bank.
- Address and command and data pins can share a bank under the following conditions:
 - Address and command and data pins cannot share an I/O lane.
 - Only an unused I/O lane in the address and command bank can be used for data pins.

Table 4. General Pin Constraints

| Signal Type | Constraint |
|---------------------|---|
| Data Strobe | All signals belonging to a DQ group must reside in the same I/O lane. |
| Data | Related DQ pins must reside in the same I/O lane. For protocols that do not support bidirectional data lines, read signals should be grouped separately from write signals. |
| Address and Command | Address and Command pins must reside in predefined locations within an I/O bank. |

Adjacent Banks

For banks to be considered adjacent, they must reside in the same I/O column. To determine if banks are adjacent, refer to the *Modular I/O banks Location and Pin Counts in Stratix 10 Devices* section located in the *Stratix 10 General Purpose I/O User Guide*.

When referring to tables in the *Stratix 10 General Purpose I/O User Guide*, it is safe to assume that all banks shown are adjacent, unless a ' - ' symbol is present; a ' - ' symbol indicates that the bank is not bonded out for the package.

Pin Assignments

To determine locations for all EMIF I/O pins you should reference the pin table for your device. When referring to the pin table, the bank numbers, I/O bank indices, and pin names are provided. You can find the pin indices for address and command pins in the *Stratix 10 Scheme Table* located on the Intel FPGA website.

You can perform pin assignments in a variety of ways. The recommended approach is to manually constrain some interface signals and let the Intel Quartus Prime Fitter handle the rest. This method consists of consulting the pin tables to find legal

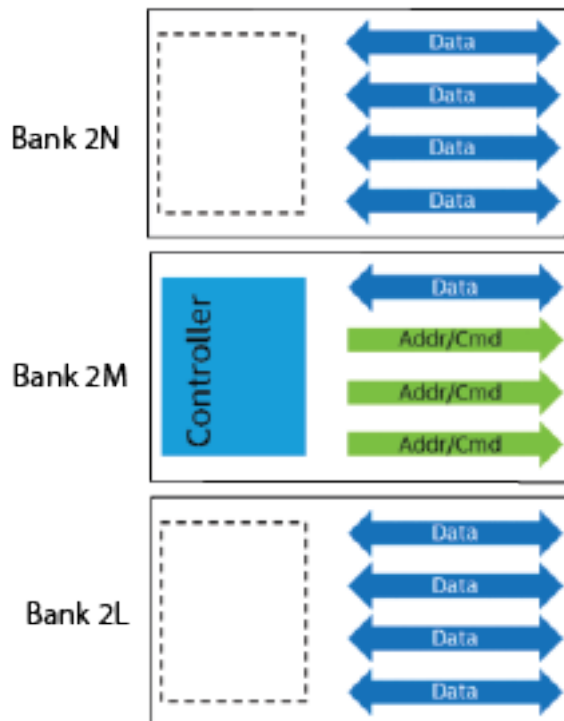
positions for some of the interface pins and assigning them through the .qsf file that is generated with the EMIF design example. For this method of I/O placement, you must constrain the following signals:

- CK0
- One DQS pin per group
- PLL reference clock
- RZQ

Based on the above constraints, the Intel Quartus Prime Fitter rotates pins within each lane as necessary. The following figure illustrates an example of pin assignments for a DDR3 x72 interface with the following selections:

- The address and command pin is placed in bank 2M and requires 3 lanes.
 - CK0 is constrained to pin 8 in bank 2M.
 - PLL reference clock pins are constrained to pins 24 and 25 in bank 2M.
 - RZQ is constrained to pin 26 in bank 2M.
- Data is placed in banks 2N, 2M, and 2L, and requires 9 lanes.
 - DQS groups 1-4 are placed in bank 2N.
 - DQS group 0 is placed in bank 2M.
 - DQS groups 5-8 are placed in bank 2L.

Figure 5. Pin Assignments Example: DDR3 x73 Interface



In this example, to constrain CK0 to pin 8 in bank 2M, you would add the following line to the .qsf file, based on the appropriate pin table:

```
set_location_assignment PIN_K31 -to CK0
```

The format of the above pin assignment can be applied to all pins:

```
set_location_assignment <pin name> -to <signal name>
```

Related Information

- [Modular I/O Banks in Intel Stratix 10 Devices](#)
- [Intel Stratix 10 EMIF IP DDR3](#)
- [Intel Stratix 10 EMIF IP for DDR4](#)
- [Intel Stratix 10 EMIF IP for QDRII/II+/Xtreme](#)
- [Intel Stratix 10 EMIF IP for QDR-IV](#)
- [Intel Stratix 10 EMIF IP for RLDRAM 3](#)

1.8. Compiling and Programming the Intel Stratix 10 EMIF Design Example

After you have made the necessary pin assignments in the .qsf file, you can compile the design example in the Intel Quartus Prime software.

1. Navigate to the Intel Quartus Prime folder containing the design example directory.
2. Open the Intel Quartus Prime project file, (.qpf).
3. To begin compilation, click **Processing > Start Compilation**. The successful completion of compilation generates an .sof file, which enables the design to run on hardware.
4. To program your device with the compiled design, open the programmer by clicking **Tools > Programmer**.
5. In the programmer, click **Auto Detect** to detect supported devices.
6. Select the Intel Stratix 10 device and then select **Change File**.
7. Navigate to the generated ed_synth.sof file and select **Open**.
8. Click **Start** to begin programming the Intel Stratix 10 device. When the device is successfully programmed, the progress bar at the top-right of the window should indicate **100% (Successful)**.

1.9. Debugging the Intel Stratix 10 EMIF Design Example

The EMIF Debug Toolkit is available to assist in debugging external memory interface designs. The toolkit allows you to display read and write margins and generate eye diagrams. After you have programmed the Intel Stratix 10 development kit, you can verify its operation using the EMIF Debug Toolkit.

1. To launch the EMIF Debug Toolkit, navigate to **Tools > System Debugging Tools > External Memory Interface Toolkit**.
2. Click **Initialize Connections**.
3. Click **Link Project to device**. A window appears; verify that the correct device is selected and that the correct `.sof` file is selected.
4. Click **Create Memory Interface Connection**. Accept the default settings by clicking **OK**.

The Intel Stratix 10 development kit is now set up to function with the EMIF Debug Toolkit, and you can generate any of the following reports by double-clicking on the corresponding option:

- **Rerun calibration**. Produces a calibration report summarizing the calibration status per DQ/DQS group along with the margins for each DQ/DQS pin.
- **Driver Margining**. Produces a report summarizing the read and write margins per I/O pin. This differs from calibration margining because driver margining is captured during user mode traffic rather than during calibration
- **Generate Eye Diagram**. Generates read and write eye diagrams for each DQ pin based on calibration data patterns.
- **Calibrate Termination**. Sweeps different termination values and reports the margins that each termination value provides. Use this feature to help select the optimal termination for the memory interface.

Related Information

[Intel Stratix 10 EMIF IP Debugging](#)

2. Design Example Description for External Memory Interfaces Intel Stratix 10 FPGA IP

When you parameterize and generate your EMIF IP, you can specify that the system create directories for simulation and synthesis file sets, and generate the file sets automatically.

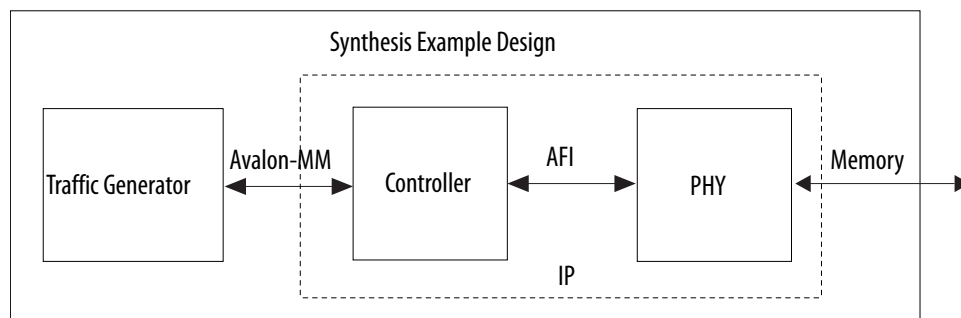
If you select **Simulation** or **Synthesis** under **Example Design Files** on the **Example Designs** tab, the system creates a complete simulation file set or a complete synthesis file set, in accordance with your selection.

2.1. Synthesis Example Design

The synthesis example design contains the major blocks shown in the figure below.

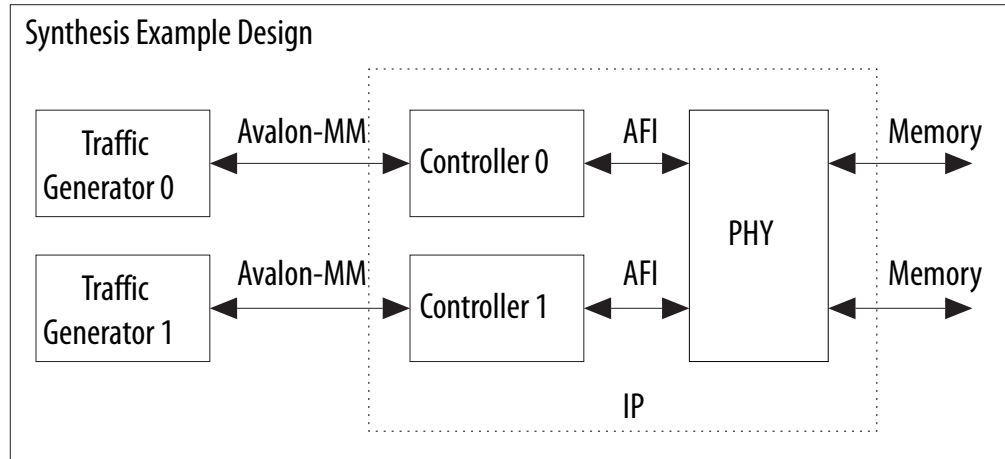
- A traffic generator, which is a synthesizable Avalon[®]-MM example driver that implements a pseudo-random pattern of reads and writes to a parameterized number of addresses. The traffic generator also monitors the data read from the memory to ensure it matches the written data and asserts a failure otherwise.
- An instance of the memory interface, which includes:
 - A memory controller that moderates between the Avalon-MM interface and the AFI interface.
 - The PHY, which serves as an interface between the memory controller and external memory devices to perform read and write operations.

Figure 6. Synthesis Example Design



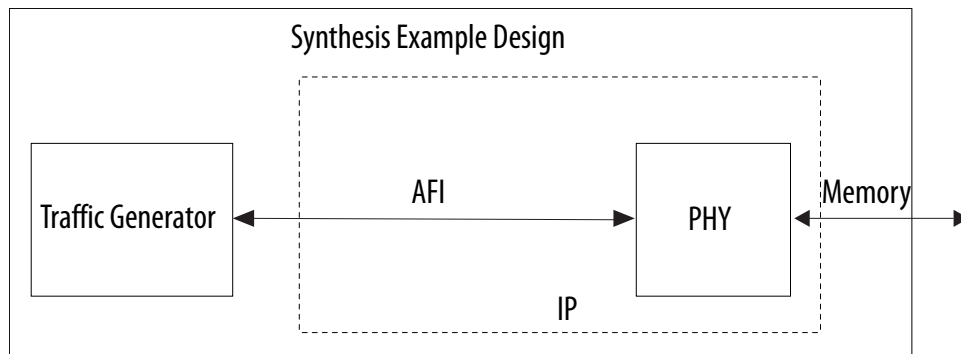
If you are using the Ping Pong PHY feature, the synthesis example design includes two traffic generators issuing commands to two independent memory devices through two independent controllers and a common PHY, as shown in the following figure.

Figure 7. Synthesis Example Design for Ping Pong PHY



If you are using RLDRAM 3, the traffic generator in the synthesis example design communicates directly with the PHY using AFI, as shown in the following figure.

Figure 8. Synthesis Example Design for RLDRAM 3 Interfaces



Note: If one or more of the **PLL Sharing Mode**, **DLL Sharing Mode**, or **OCT Sharing Mode** parameters are set to any value other than **No Sharing**, the synthesis example design will contain two traffic generator/memory interface instances. The two traffic generator/memory interface instances are related only by shared PLL/DLL/OCT connections as defined by the parameter settings. The traffic generator/memory interface instances demonstrate how you can make such connections in your own designs.

Note: Third-party synthesis flow as described in [Intel Quartus Prime Standard Edition User Guide: Third-party Synthesis](#) is not a supported flow for EMIF IP.

Related Information

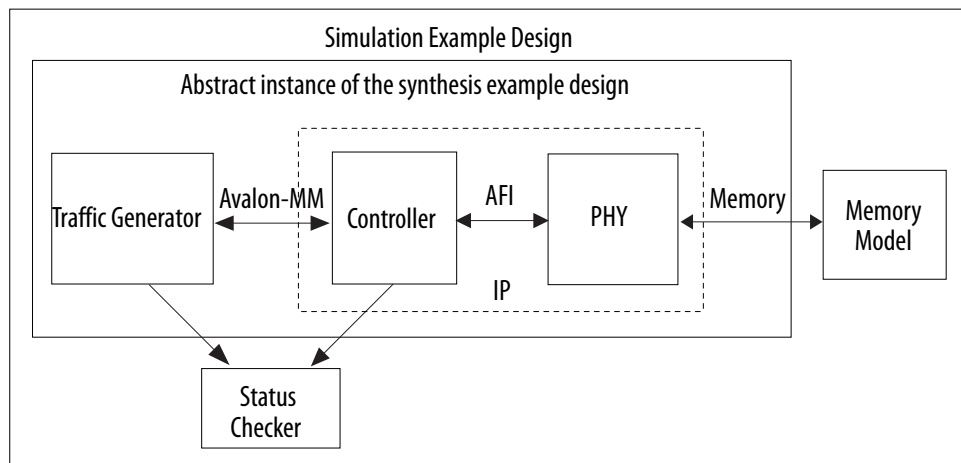
[Generating the Synthesizable EMIF Design Example](#) on page 7

2.2. Simulation Example Design

The simulation example design contains the major blocks shown in the following figure.

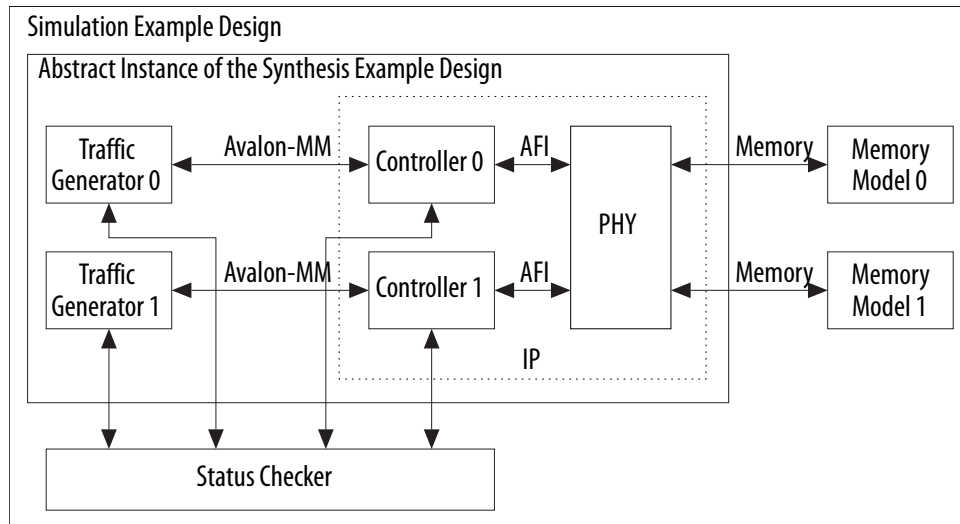
- An instance of the synthesis example design. As described in the previous section, the synthesis example design contains a traffic generator and an instance of the memory interface. These blocks default to abstract simulation models where appropriate for rapid simulation.
- A memory model, which acts as a generic model that adheres to the memory protocol specifications. Frequently, memory vendors provide simulation models for their specific memory components that you can download from their websites.
- A status checker, which monitors the status signals from the external memory interface IP and the traffic generator, to signal an overall pass or fail condition.

Figure 9. Simulation Example Design



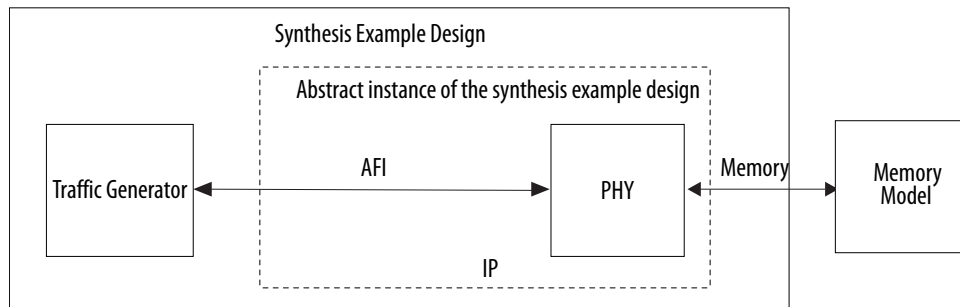
If you are using the Ping Pong PHY feature, the simulation example design includes two traffic generators issuing commands to two independent memory devices through two independent controllers and a common PHY, as shown in the following figure.

Figure 10. Simulation Example Design for Ping Pong PHY



If you are using RLDRAM 3, the traffic generator in the simulation example design communicates directly with the PHY using AFI, as shown in the following figure.

Figure 11. Simulation Example Design for RLDRAM 3 Interfaces



Related Information

[Generating the EMIF Design Example for Simulation](#) on page 10

2.3. Example Designs Interface Tab

The parameter editor includes an **Example Designs** tab which allows you to parameterize and generate your example designs.

Available Example Designs Section

The **Select design** pulldown allows you to select the desired example design. At present, **EMIF Example Design** is the only available choice, and is selected by default.

3. Document Revision History for External Memory Interfaces Intel Stratix 10 FPGA IP Design Example User Guide

| Document Version | Intel Quartus Prime Version | Changes |
|------------------|-----------------------------|--|
| 2021.03.29 | 21.1 | <ul style="list-style-type: none"> In the <i>Example Design Quick Start</i> chapter, removed references to the NCSim* simulator. |
| 2018.09.24 | 18.1 | <ul style="list-style-type: none"> Updated figures in the <i>Generating the Synthesizable EMIF Design Example</i> and <i>Generating the EMIF Design Example for Simulation</i> topics. |
| 2018.05.07 | 18.0 | <ul style="list-style-type: none"> Changed document title from <i>Intel Stratix 10 External Memory Interfaces IP Design Example User Guide</i> to <i>External Memory Interfaces Intel Stratix 10 FPGA IP Design Example User Guide</i>. Corrected bullet points in <i>Overview</i> section of the <i>Pin Placement for Intel Stratix 10 EMIF IP</i> topic. |

| Date | Version | Changes |
|---------------|------------|------------------|
| November 2017 | 2017.11.06 | Initial release. |

Intel Corporation. All rights reserved. Agilix, Altera, Arria, Cyclone, eASIC, Intel, the Intel logo, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

*Other names and brands may be claimed as the property of others.