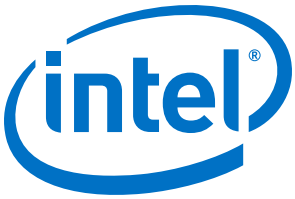


External Memory Interfaces Intel® Arria® 10 FPGA IP Design Example User Guide

Updated for Intel® Quartus® Prime Design Suite: **18.1**

UG-20118 | 2018.09.24
Latest document on the web: **PDF** | **HTML**

Subscribe
Send Feedback

Contents

1. Design Example Quick Start Guide for External Memory Interfaces Intel® Arria® 10 FPGA IP 3
1.1. Creating an EMIF Project..... 4
1.2. Generating and Configuring the EMIF IP..... 6
1.2.1. Intel Arria 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..... 12
1.6. Simulating External Memory Interface IP With ModelSim..... 13
1.7. Pin Placement for Intel Arria 10 EMIF IP..... 14
1.8. Compiling and Programming the Intel Arria 10 EMIF Design Example..... 15
1.9. Debugging the Intel Arria 10 EMIF Design Example..... 15

2. Design Example Description for External Memory Interfaces Intel Arria 10 FPGA IP ... 17
2.1. Synthesis Example Design..... 17
2.2. Simulation Example Design..... 18
2.3. Example Designs Interface Tab..... 20

3. Document Revision History for External Memory Interfaces Intel Arria 10 FPGA IP Design Example User Guide..... 21

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

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

The **Example Designs** tab in the parameter editor allows you to specify the creation of synthesis and/or 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

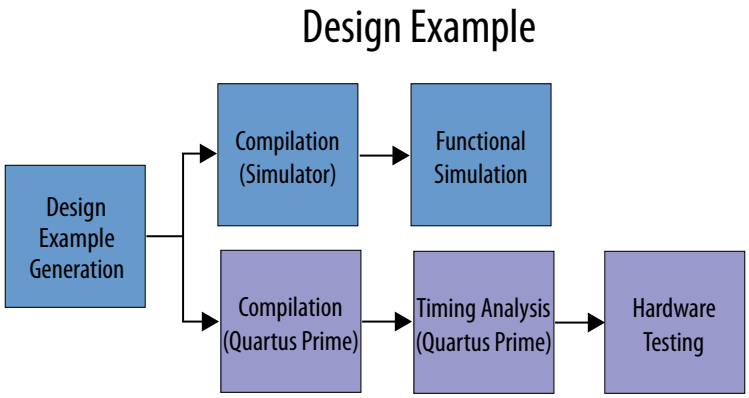
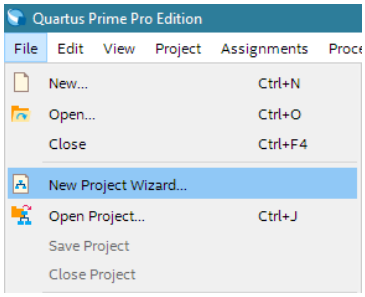


Figure 2. Generating an EMIF Example Design With an Intel Arria 10 Development Kit

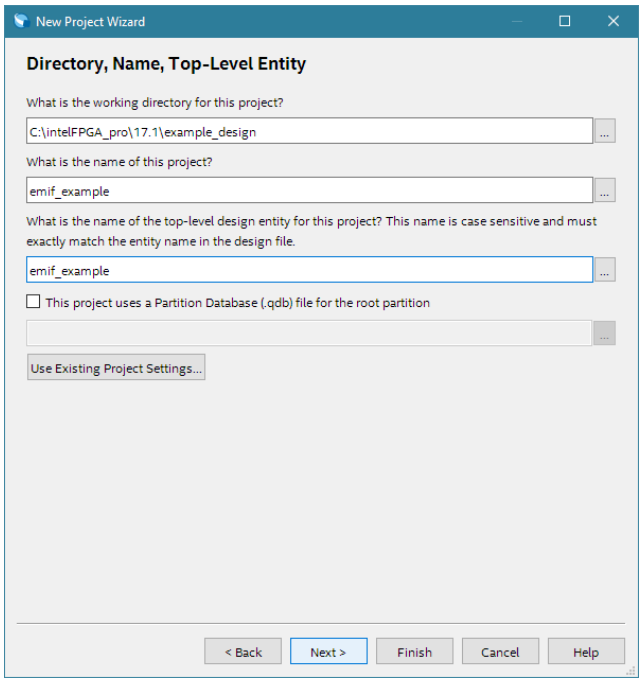
1.1. Creating an EMIF Project

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.

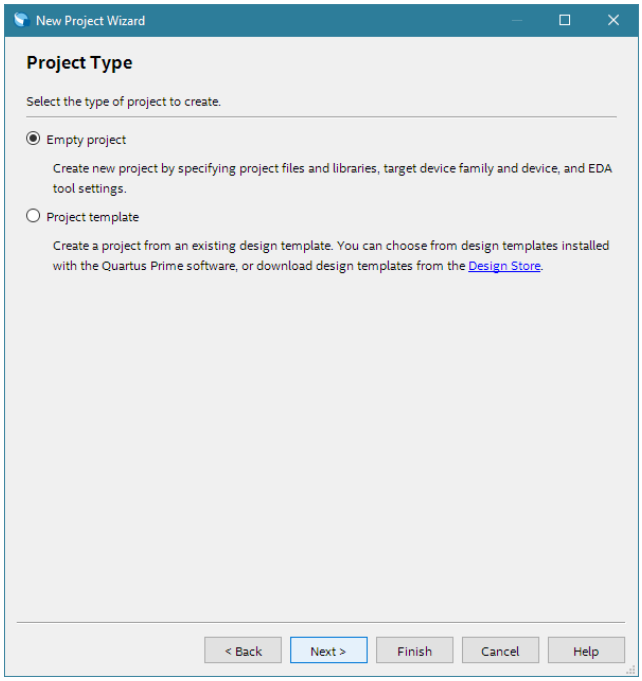
1. Launch the Intel Quartus Prime software and select **File ➤ New Project Wizard**. Click **Next**.



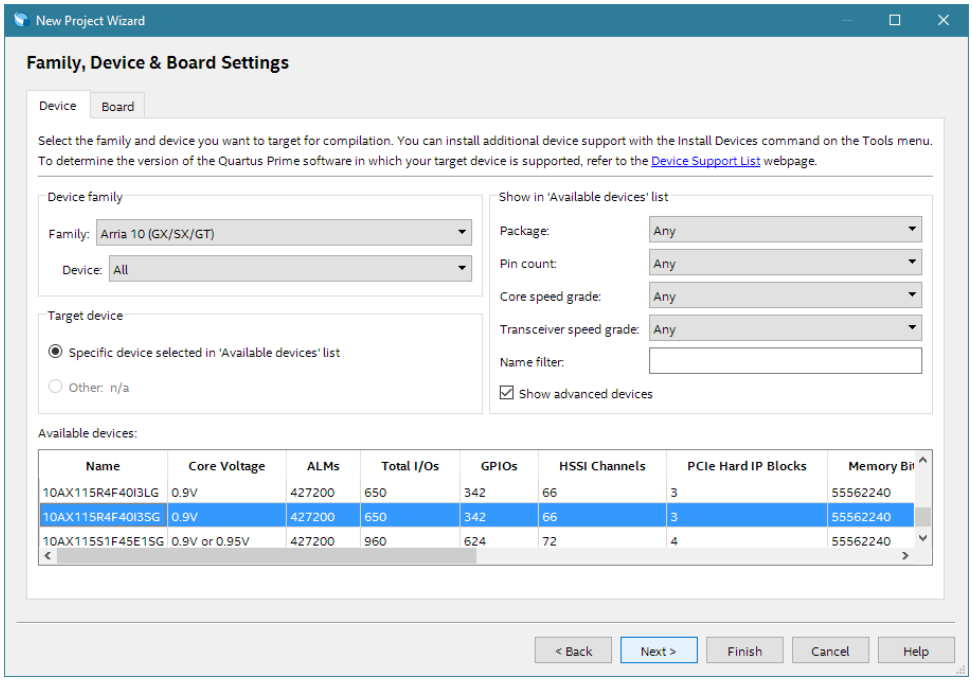
2. Specify a directory and name for the project that you want to create. Click **Next**.



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.

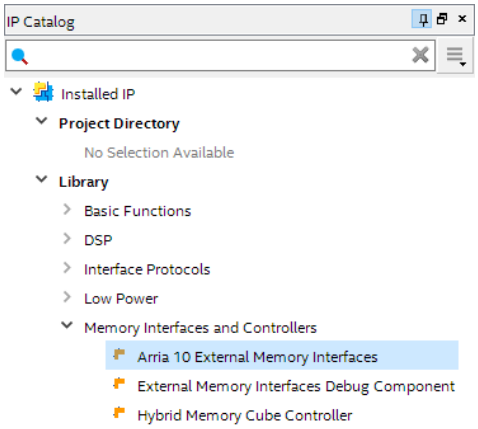


6. Click **Finish**.

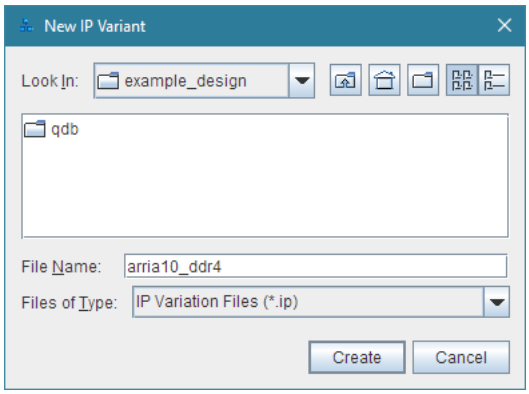
1.2. Generating and Configuring the EMIF IP

The following steps illustrate how to generate and configure the EMIF IP. The steps are similar regardless of the memory protocol that you are targeting.

1. In the **IP Catalog window**, select **Intel Arria 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 Arria 10 EMIF Parameter Editor Guidelines

Table 1. EMIF Parameter Editor Guidelines

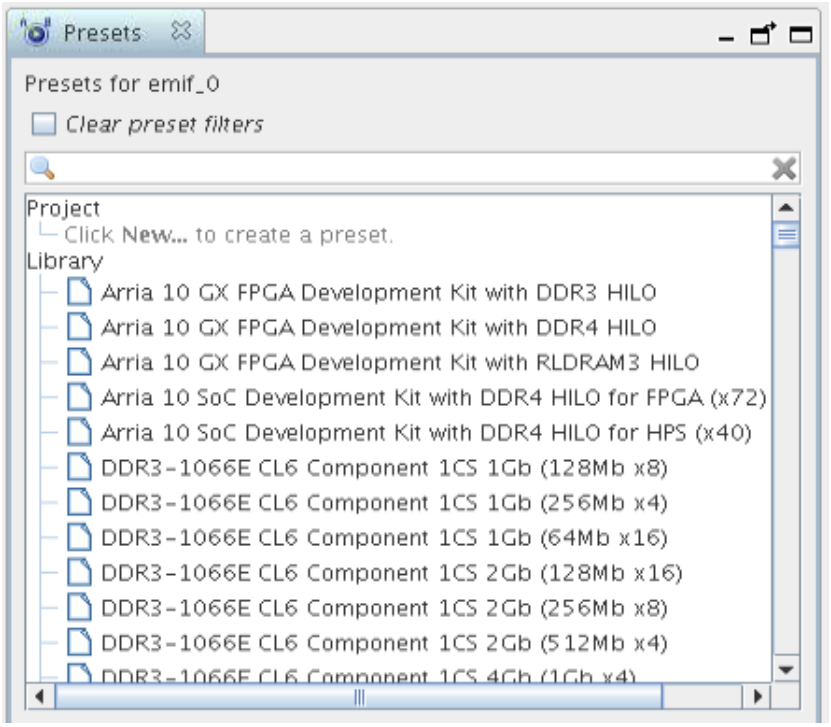
Parameter Editor Tab	Guidelines
General	Ensure that the following parameters are entered correctly: • The speed grade for the device. • The memory clock frequency. • The PLL reference clock frequency.
Memory	• Refer to the data sheet for your memory device to enter the parameters on the **Memory** tab. • You should also enter a specific location for the ALERT# pin. (Applies to DDR4 memory protocol only.)
Mem I/O	• 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	• 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	• 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	• 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 Arria 10 External Memory Interfaces IP User Guide*.

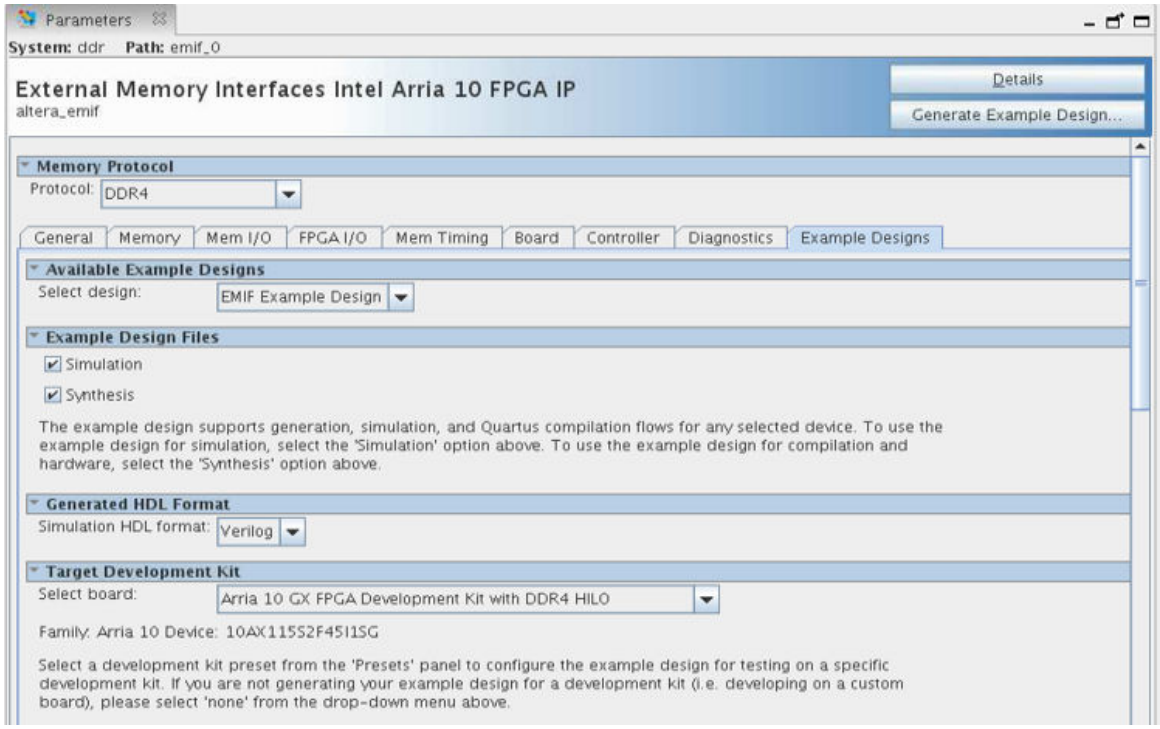
1.3. Generating the Synthesizable EMIF Design Example

For the Intel Arria 10 development kits, there are presets that automatically parameterize the EMIF IP and generate pinouts for the specific board.

1. Verify that the **Presets** window is visible. If the **Presets** window is not visible, display it by selecting **View ➤ Presets**.
2. In the **Presets** window, select the appropriate development kit preset and click **Apply**.

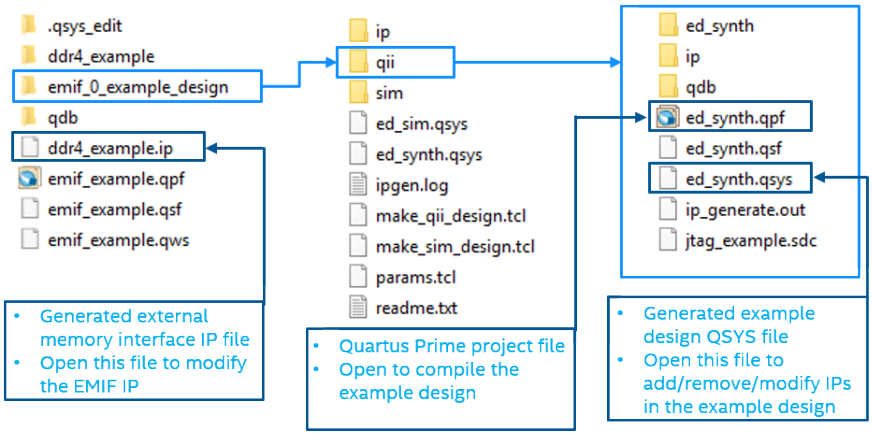


3. Configure the EMIF IP and click **Generate Example Design** in the upper-right corner of the window.



4. Specify a directory for the EMIF design example and click **OK**. Successful generation of the EMIF design example creates 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.

The **Select board** pulldown in this section applies the appropriate development kit pin assignments to the example design.

- This setting is available only when you turn on the **Synthesis** checkbox in the **Example Design Files** section.
- This setting must match the applied development kit present, or else an error message appears.

If the value *None* appears in the **Select board** pulldown, it indicates that the current parameter selections do not match any development kit configurations. You may apply a development kit-specific IP and related parameter settings by selecting one of the presets from the preset library. When you apply a preset, the current IP and other parameter settings are set to match the selected preset. If you want to save your current settings, you should do so before you select a preset. If you do select a preset without saving your prior settings, you can always save the new preset settings under a different name

If you want to generate the example design for use on your own board, set **Select board** to *None*, generate the example design, and then add pin location constraints.

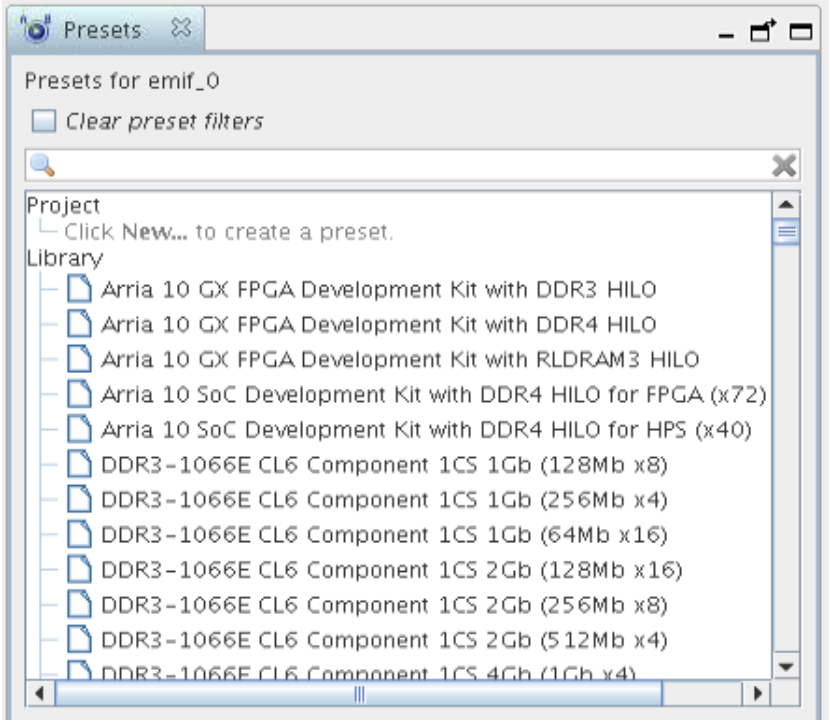
Related Information

- Synthesis Example Design on page 17
- Intel Arria 10 EMIF IP Parameter Descriptions for DDR3
- Intel Arria 10 EMIF IP Parameter Descriptions for DDR4
- Intel Arria 10 EMIF IP Parameter Descriptions for QDRII/II+/Xtreme
- Intel Arria 10 EMIF IP Parameter Descriptions for QDR-IV
- Intel Arria 10 EMIF IP Parameter Descriptions for RLDRAM 3
- Intel Arria 10 EMIF IP Parameter Descriptions for LPDDR3

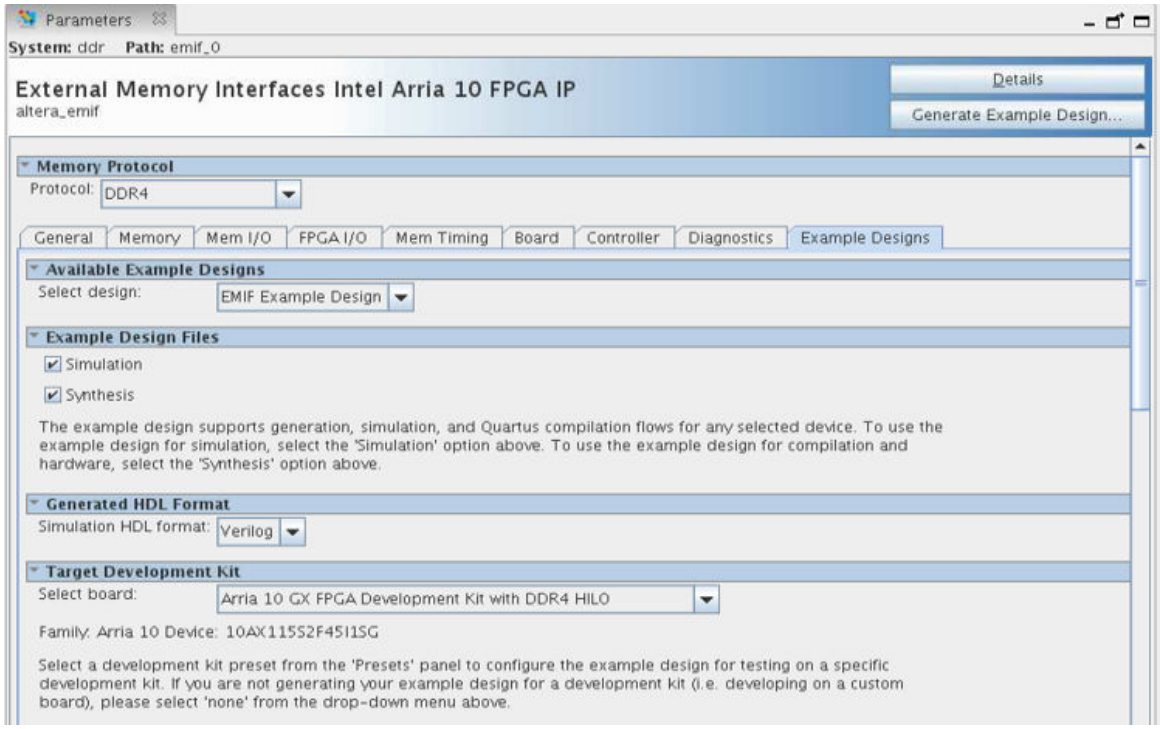
1.4. Generating the EMIF Design Example for Simulation

For the Intel Arria 10 development kits, there are presets that automatically parameterize the EMIF IP and generate pinouts for the specific board.

1. Verify that the **Presets** window is visible. If the **Presets** window is not visible, display it by selecting **View ➤ Presets**.
2. In the **Presets** window, select the appropriate development kit preset and click **Apply**.



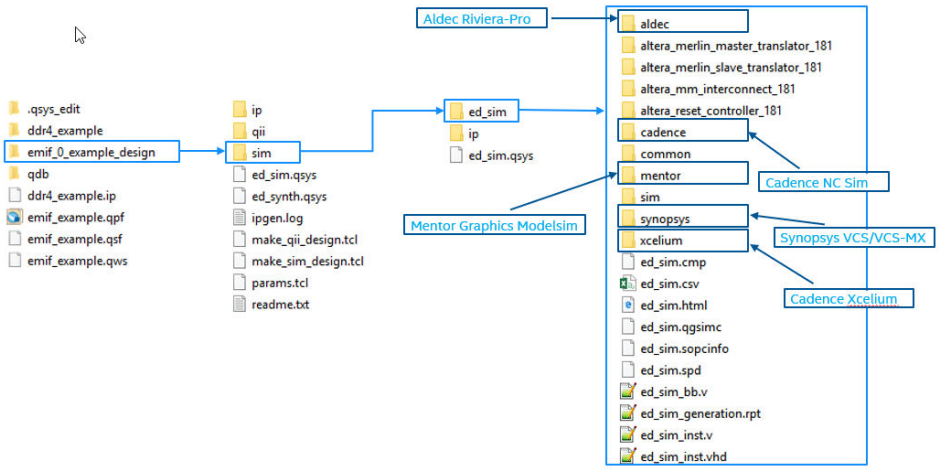
3. Configure the EMIF IP and click **Generate Example Design** in the upper-right corner of the window.



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 18
- Intel Arria 10 EMIF IP - Simulating Memory IP

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 `cal_done` signal signal simultaneously in simulation.	Fitter operations determine the order of calibration, and interfaces do not assert `cal_done` 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 Arria 10 EMIF IP - Simulating Memory IP

1.7. Pin Placement for Intel Arria 10 EMIF IP

This topic provides guidelines for pin placement.

Overview

Intel Arria 10 FPGAs have the following structure:

- Each device contains 2 I/O columns.
- Each I/O column contains up to 8 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 Arria 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. DM/DBI pins must be paired off with a DQ pin for proper operation. 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.

Pin Assignments

If you applied a development kit preset during IP generation, all pin assignments for the development kit are automatically generated and can be verified in the `.qsf` file that is generated with the design example.

Related Information

- Intel Arria 10 EMIF IP DDR3
- Intel Arria 10 EMIF IP for DDR4
- Intel Arria 10 EMIF IP for QDRII/II+/Xtreme
- Intel Arria 10 EMIF IP for QDR-IV
- Intel Arria 10 EMIF IP for RLDRAM 3
- Intel Arria 10 EMIF IP for LPDDR3

1.8. Compiling and Programming the Intel Arria 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 `.qsf` 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 Arria 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 Arria 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 Arria 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 Arria 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 Arria 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 Arria 10 EMIF IP Debugging

2. Design Example Description for External Memory Interfaces Intel Arria 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.

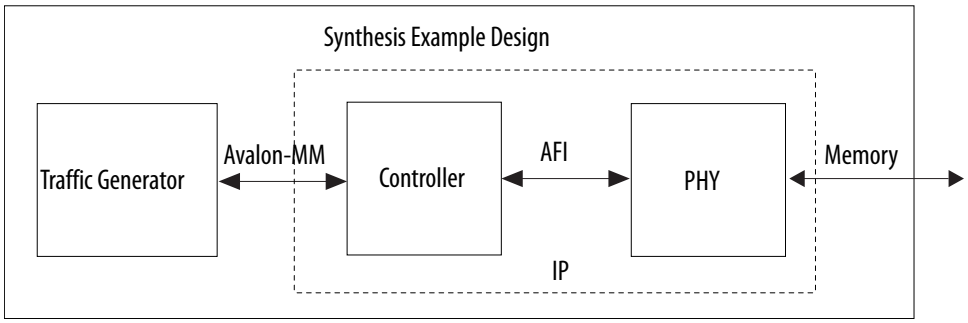
If you select **Simulation** or **Synthesis** under **Example Design Files** on the **Example Designs** tab, the system creates a complete simulation file set or a complete synthesis file set, in accordance with your selection.

2.1. Synthesis Example Design

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

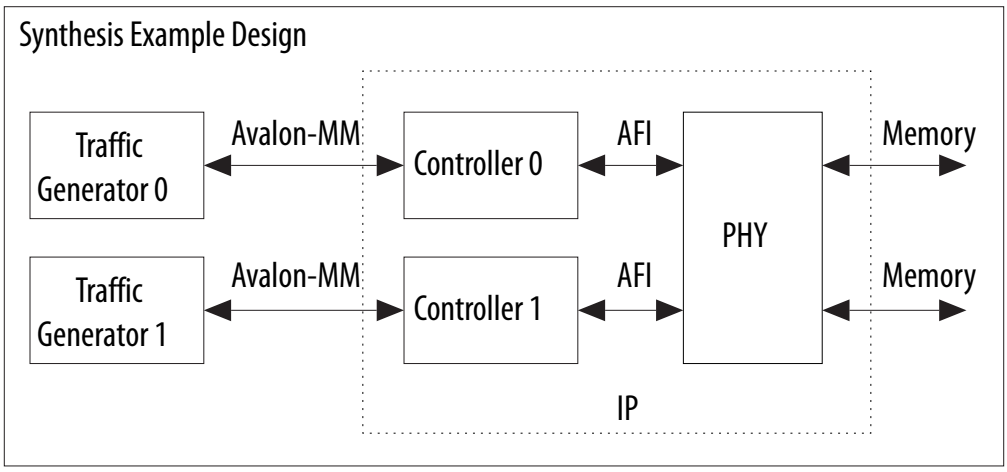
- A traffic generator, which is a synthesizable Avalon®-MM example driver that implements a pseudo-random pattern of reads and writes to a parameterized number of addresses. The traffic generator also monitors the data read from the memory to ensure it matches the written data and asserts a failure otherwise.
- 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 5. Synthesis Example Design



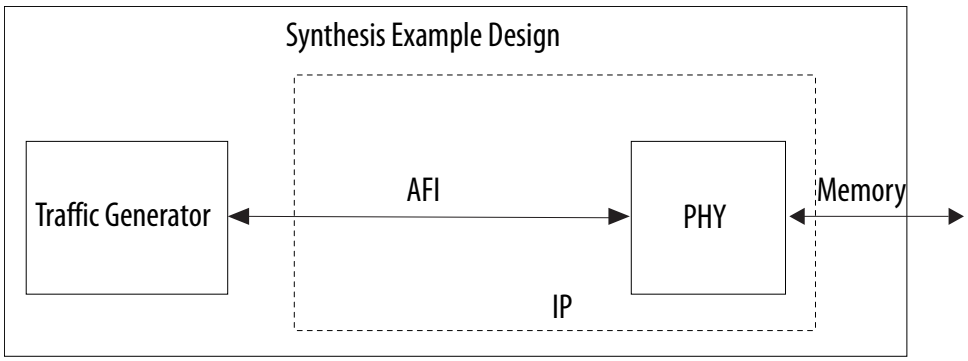
If you are using the Ping Pong PHY feature, the synthesis example design includes two traffic generators issuing commands to two independent memory devices through two independent controllers and a common PHY, as shown in the following figure.

Figure 6. 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 7. 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.

Related Information

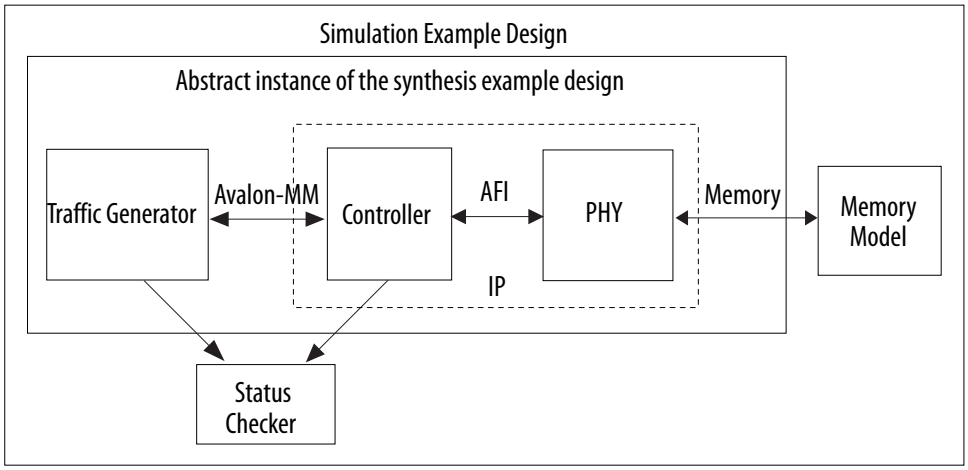
Generating the Synthesizable EMIF Design Example on page 7

2.2. Simulation Example Design

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

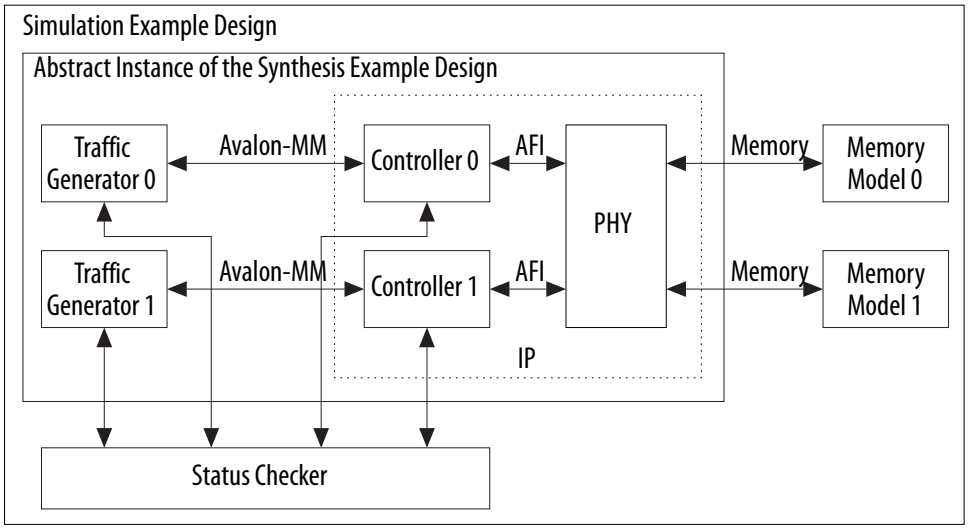
- An instance of the synthesis example design. As described in the previous section, the synthesis example design contains a traffic generator and an instance of the memory interface. These blocks default to abstract simulation models where appropriate for rapid simulation.
- A memory model, which acts 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 8. Simulation Example Design



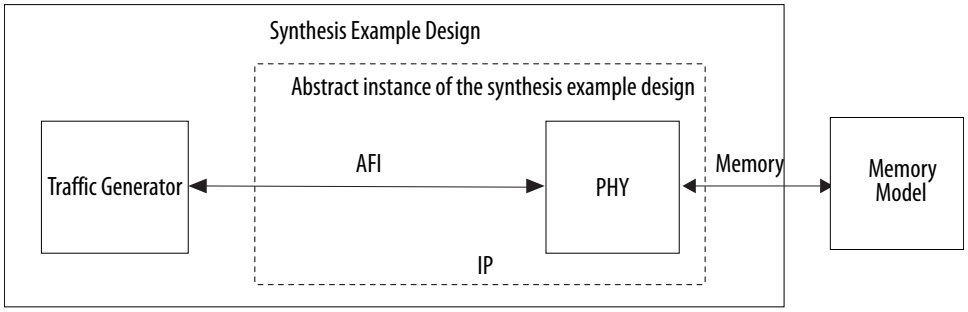
If you are using the Ping Pong PHY feature, the simulation example design includes two traffic generators issuing commands to two independent memory devices through two independent controllers and a common PHY, as shown in the following figure.

Figure 9. 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 10. Simulation Example Design for RLDRAM 3 Interfaces



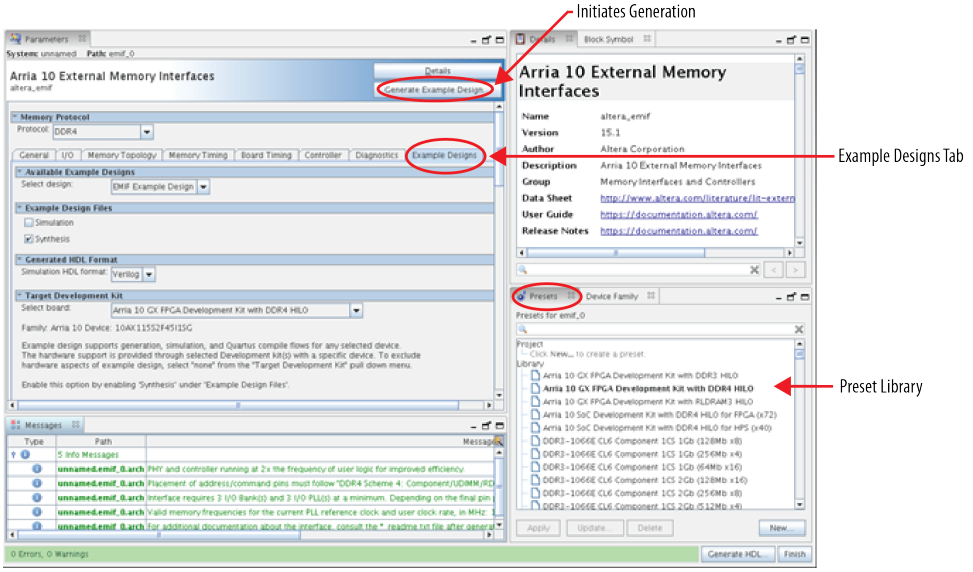
Related Information

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.l

Figure 11. Example Designs Tab in the External Memory Interfaces Parameter Editor



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 Arria 10 FPGA IP Design Example User Guide

Document Version	Intel Quartus Prime Version	Changes
2018.09.24	18.1	• Updated figures in the *Generating the Synthesizable EMIF Design Example* and *Generating the EMIF Design Example for Simulation* topics.
2018.05.07	18.0	• Changed document title from *Intel Arria 10 External Memory Interfaces IP Design Example User Guide* to *External Memory Interfaces Intel Arria 10 FPGA IP Design Example User Guide*. • Corrected bullet points in *Overview* section of the *Pin Placement for Intel Arria 10 EMIF IP* topic.

Date	Version	Changes
November 2017	2017.11.06	Initial release.