This chapter describes the Altera-recommended design flow for successfully implementing external memory interfaces in Altera devices. Altera recommends that you create an example top-level file with the desired pin outs and all interface IP instantiated, which enables the Quartus® II software to validate your design and resource allocation before PCB and schematic sign off. Use the “Design Checklist” on page 2–6, to verify whether you have performed all the recommended steps in creating a working and robust external memory interface.

Figure 2–1 shows the design flow to provide the fastest out-of-the-box experience with external memory interfaces in Altera devices. This topic directs you where to find information on how to perform each step of the recommended design flow. The flow assumes that you are using Altera IP to implement the external memory interface.

For design examples that follow the recommended design flow in this chapter, refer to Volume 6: Design Flow Tutorials of the External Memory Interface Handbook.
Figure 2–1. External Memory Interfaces Design Flowchart

1. Start Design
2. Select Device
3. Determine Board Layout
4. Perform Board Level Simulations
5. Adjust Termination and Drive Strength
6. Do Signals Meet Electrical Requirements?
   - Yes: Adjust Constraints and Compile Design
   - No: Perform Functional Simulation
7. Does Simulation Give Expected Results?
   - No: Debug Design
   - Yes: Verify Timing
8. Does the Design Have Positive Margin?
   - No: Adjust Constraints
   - Yes: Complete SOPC Builder System
9. Specify Parameters
10. Instantiate PHY and Controller
11. Optional: Perform Timing Simulation
12. Does Simulation Give Expected Results?
    - No: Debug Design
    - Yes: Verify Design Functionality on Board
13. Is Design Working?
    - No: Design Done
    - Yes: Design Done

UniPHY-Based Designs Only on Arria II GX and Stratix IV Devices
Select a Device

For more information on selecting a device, refer to the Device and Pin Planning section in volume 2 of the External Memory Interface Handbook.

Determine Board Layout

Altera recommends prelayout SI simulations (line simulations) should take place before board layout and that you use these parameters and rules during the initial design development cycle. Advanced I/O timing and board trace models now directly impact device timing closure.

In addition, the termination scheme that you use, the drive strength setting on the FPGA, and the loading seen by the driver can directly affect the signal integrity. You must understand the tradeoffs between the different types of termination schemes and the effects of output drive strengths and loading, to choose the best possible settings for your designs.

For more information, refer to the Board Layout Guidelines section in volume 2 of the External Memory Interface Handbook.

Perform Board-Level Simulations

To determine the correct board constraints, perform board-level simulations to see if the settings provide the optimal signal quality. With many variables that can affect the signal integrity of the memory interface, simulating the memory interface provides an initial indication of how well the memory interface performs. There are various EDA simulation tools available to perform board-level simulations. The simulations should be performed on the data, data strobe, control, command, and address signals. If the memory interface does not have good signal integrity, adjust the settings, such as drive strength setting, termination scheme or termination values to improve the signal integrity (realize that changing these settings affects the timing and it may be necessary to go back to the timing closure if these change).

For detailed information about understanding the different effects on signal integrity design, refer to the Board Layout Guidelines section in volume 2 of the External Memory Interface Handbook.

Enter topology information from your board-level simulations into the Quartus II board trace model information. The typical information required includes, but is not limited to, the following values:

- Near and far trace lengths
- Near and far trace distributed inductance
- Near and far trace distributed capacitance
- Near end deration capacitor values (if fitted)
- Far end capacitive (IC) load
- Far end termination values
Device-Side Termination

Many Altera devices support both series and parallel OCT resistors to improve signal integrity. OCT eliminates the need for external termination resistors on the FPGA side, which simplifies board design and reduces overall board cost. You can dynamically switch between the series and parallel OCT resistor depending on whether the FPGA devices are performing a write or a read operation. The OCT features offer user-mode calibration to compensate for any variation in VT during normal operation to ensure that the OCT values remain constant. The parallel and series OCT features are available in either 25 or 50 $\Omega$ settings.

Memory-Side Termination

The DDR2, DDR3 SDRAM, and QDR II SRAM have a dynamic parallel ODT feature that you can turn on when the FPGA is writing to the memory and turn off when the FPGA is reading from the memory. To further improve signal integrity, DDR2 SDRAM supports output drive strength control so that the driver can better match the transmission line. DDR3 SDRAM devices additionally support calibrated output impedances.

For more information on available settings of the ODT, the output drive strength features, and the timing requirements for driving the ODT pin, refer to your DDR2 or DDR3 SDRAM datasheet.

Adjust Termination and Drive Strength

Although the recommended terminations are based on the simulations and experimental results, you must perform simulations, either using I/O buffer information specification (IBIS) or HSPICE models, to determine the quality of signal integrity on your designs.

Any changes made to the board should also be made in the board trace model in the Quartus II software.

For information on Altera-recommended terminations for memory interfaces, refer to the Board Layout Guidelines section in volume 2 of the External Memory Interface Handbook.

Instantiate PHY and Controller

After selecting the appropriate device and memory type, create a project in the Quartus II software that targets the device and memory type.

When implementing external memory interfaces, Altera recommends that you use Altera memory interface IP, which includes a PHY that you can use with the Altera high-performance controller or with your own custom controller.

Instantiating the PHY and controller includes the following steps:

- Specify parameters
- Perform functional simulation
- Add constraints and compile design
Verify Timing

For more information about specifying parameters, adding constraints, and compiling, refer to the DDR and DDR2 SDRAM High-Performance Controller and ALTMEMPHY IP User Guide section and the DDR3 SDRAM High-Performance Controller and ALTMEMPHY IP User Guide section in volume 3 of the External Memory Interface Handbook.

For more information about simulation, refer to the Simulation section in volume 4 of the External Memory Interface Handbook.

Adjust Constraints

In the timing report of the design, you can see the worst case setup and hold margin for the different paths in the design. If the setup and hold margin are unbalanced, achieve a balanced setup and hold margin by adjusting the phase setting of the clocks associated with these paths.

For example, for the address and command margin, the address and command outputs are clocked by an address and command clock that can be different with respect to the system clock, which is -30°. The system clock controls the clock outputs going to the memory. If the report timing script indicates that using the default phase setting for the address and command clock results in more hold time than setup time, adjust the address and command clock to be less negative than the default phase setting with respect to the system clock, so that there is less hold margin. Similarly, adjust the address and command clock to be more negative than the default phase setting with respect to the system clock if there is more setup margin.

For more information on adjusting constraints, refer to the Timing Analysis section in volume 4 of the External Memory Interface Handbook.

Perform Timing Simulation

This step is optional, but recommended to ensure that the IP is working properly. This step only applies to UniPHY-based interfaces (on Arria® II GX and Stratix® IV devices only), as ALTMEMPHY-based interfaces do not support timing simulation.

For more information about simulating, refer to the Simulation section in volume 4 of the External Memory Interface Handbook.

Verify Design Functionality

Perform system level verification to correlate the system against your design targets, using the Altera SignalTap® II logic analyzer.

For more information about using the SignalTap II analyzer, refer to the Debugging section in volume 4 of the External Memory Interface Handbook.
This topic contains a design checklist that you can use when implementing external memory interfaces in Altera devices.

**Select Device**

1. **☐** Select the memory interface frequency of operation and bus width.
   
   For information about selecting memory, refer to the Memory Standard Overview section in volume 1 of the External Memory Interface Handbook.

2. **☐** Select the FPGA device density and package combination that you want to target.
   
   For information about selecting an Altera device, refer to the Device and Pin Planning section in volume 2 of the External Memory Interface Handbook.

3. **☐** Ensure that the target FPGA device supports the desired clock rate and memory bus width. Also the FPGA must have sufficient I/O pins for the DQ/DQS read and write groups.
   
   For detailed device resource information, refer to the relevant device handbook chapter on external memory interface support.
   
   For information about supported clock rates for external memory interfaces, refer to the External Memory Interface System Specifications section in volume 1 of the External Memory Interface Handbook.

**Determine Board Layout**

4. **☐** Select the termination scheme and drive strength settings for all the memory interface signals on the memory side and the FPGA side.

5. **☐** Ensure you apply appropriate termination and drive strength settings on all the memory interface signals, and verify using board level simulations.

6. **☐** Use board level simulations to pick the optimal setting for best signal integrity. On the memory side, Altera recommends the use of external parallel termination on input signals to the memory (write data, address, command, and clock signals).
   
   For information, refer to the Board Layout Guidelines section in volume 2 of the External Memory Interface Handbook.

**Perform Board Level Simulations**

7. **☐** Perform board level simulations, to ensure electrical and timing margins for your memory interface

8. **☐** Ensure you have a sufficient eye opening using simulations. Use the latest FPGA and memory IBIS models, board trace characteristics, drive strength, and termination settings in your simulation.
   
   Any timing uncertainties at the board level that you calculate using simulations must be used to adjust the input timing constraints to ensure the accuracy of Quartus II timing margin reports. For example crosstalk, ISI, and slew rate deration.
   
   For information, refer to the Board Layout Guidelines section in volume 2 of the External Memory Interface Handbook.

**Instantiate PHY and Controller**
Parameterize and instantiate the Altera external memory IP for your target memory interface.

You have the following three choices in implementing a memory interface in the Quartus II software:

- Using the Altera memory controller and PHY.
  - For information about how to implement a specific memory interface, refer to Volume 3: Implementing Altera Memory Interface IP of the External Memory Interface Handbook.
  - The Parameter Settings chapter of each section describes the IP supported features page by page.

- Using the Altera PHY with your own custom controller.
  - For information about how to implement a specific memory interface, refer to Volume 3: Implementing Altera Memory Interface IP of the External Memory Interface Handbook.
  - The Parameter Settings chapter of each section describes the IP supported features page by page. The Functional Description chapter describes the workings of the PHY and how you can connect a custom controller to the PHY.

- Implement a custom PHY and custom controller.
  For information about creating custom IP, refer to Volume 5: Implementing Custom Memory Interface PHY of the External Memory Interface Handbook.

Ensure that you perform the following actions:

- Pick the correct memory interface data rates, width, and configurations.
- For DDR, DDR2, and DDR3 SDRAM interfaces, ensure that you derate the tIS, tIH, tDS, and tDH parameters, as necessary.
- Include the board skew parameter for your board.

Connect the PHY’s local signals to your driver logic and the PHY’s memory interface signals to top-level pins.

Ensure that the local interface signals of the PHY are appropriately connected to your own logic. If the ALTMEMPHY megafunction is compiled without these local interface connections, you may encounter compilation problems, when the number of signals exceeds the pins available on your target device.

For more information about the example top-level file, refer to the Functional Description chapter for the relevant memory controller.

You may also use the example top-level file as an example on how to connect your own custom controller to the Altera memory PHY.

Perform Functional Simulation

Simulate your design using the RTL functional model.

Use the IP functional simulation model with your own driver logic, testbench, and a memory model, to ensure correct read and write transactions to the memory.

You may need to prepare the memory functional model by setting the speed grade and device bus mode.

For more information about simulation, refer to the Simulation section in volume 4 of the External Memory Interface Handbook.

Add Contraints
13. Add timing constraints. The wizard-generated .sdc file adds timing constraints to the interface. However, you may need to adjust these settings to best fit your memory interface configuration.

14. Add pin settings and DQ group assignments. The wizard-generated .tcl file includes I/O standard and pin loading constraints to your design.

15. Ensure that generic pin names used in the constraint scripts are modified to match your top-level pin names. The loading on memory interface pins is dependent on your board topology (memory components).

16. Add pin location assignments. However, you need to assign the pin location assignments manually using the Pin Planner.

17. Ensure that the example top-level file or your top-level logic is set as top-level entity.

18. Adjust optimization techniques, to ensure the remaining unconstrained paths are routed with the highest speed and efficiency:
   a. On the Assignments menu click Settings.
   b. Select Analysis & Synthesis Settings.
   c. Select Speed under Optimization Technique.
   d. Expand Fitter Settings.
   e. Turn on Optimize Hold Timing and select All Paths.
   f. Turn on Optimize Fast Corner Timing.
   g. Select Standard Fit under Fitter Effort.

19. Provide board trace delay model. For accurate I/O timing analysis, you specify the board trace and loading information in the Quartus II software. This information should be derived and refined during your board development process of prelayout (line) simulation and finally post-layout (board) simulation. Provide the board trace information for the output and bidirectional pins through the board trace model in the Quartus II software.

   For more information, refer to the Add Constraints chapter for the relevant memory standard in volume 3 of the External Memory Interface Handbook or refer to Volume 6: Design Flow Tutorials.

Compile Design and Verify Timing

20. Compile your design and verify timing closure using all available models.

21. Run the wizard-generated <variation_name>_report_timing.tcl file, to generate a custom timing report for each of your IP instances. Run this process across all device timing models (slow 0°C, slow 85°C, fast 0°C).

22. If there are timing violations, adjust your constraints to optimize timing

23. As required, adjust PLL clock phase shift settings or appropriate timing and location assignments margins for the various timing paths within the IP.

For information, refer to the Timing Analysis section in volume 4 of the External Memory Interface Handbook.

Perform Timing Simulation
24. ✗ Perform gate-level or timing simulation to ensure that all the memory transactions meet the timing specifications with the vendor’s memory model. Timing simulation is only supported with UniPHY-based memory interfaces.

For more information about simulation, refer to the Simulation section in volume 4 of the External Memory Interface Handbook.

**Verify Design Functionality**

25. ✗ Verify the functionality of your memory interface in the system

For more information, refer to Volume 6: Design Flow Tutorials. in the External Memory Interface Handbook.