DQ (ALTDQ) and DQS (ALTDQS) Megafunctions

Introduction

The Quartus® II software provides parameterizable megafunctions ranging from simple arithmetic units, such as adders and counters, to advanced phase-locked loop (PLL) blocks, multipliers, and memory structures. These megafunctions are performance-optimized for Altera devices and therefore provide more efficient logic synthesis and device implementation, because they automate the coding process and save valuable design time. You should use these functions during design implementation so you can consistently meet your design goals.

General Description

This user guide discusses the following topics:

- General features of the ALTDQ and ALTDQS megafunctions
- Parameterization of the ALTDQ and ALTDQS megafunctions through the MegaWizard™ Plug-In Manager
- Port and parameter definitions of the ALTDQ and ALTDQS megafunctions

The ALTDQ and ALTDQS megafunctions allow you to control the functionality of the DDR I/O pins for each of the device families. Most of the features of the megafunction map directly into features of the I/O element (IOE) for each device family. For Cyclone® II devices that do not have DDR I/O registers in the IOE, the features are implemented in logic cells.

The ALTDQ and ALTDQS megafunctions are provided in the Quartus II software MegaWizard Plug-In Manager. You can configure the DQ and DQS pins as input, output, or bidirectional DDR pins on all the I/O banks of the device, depending on the specific custom external memory interface requirements. Both DQ and DQS are bidirectional (the same signals are used for both writes and reads). A group of DQ pins is associated with one DQS pin. Use the ALTDQ and ALTDQS megafunctions to configure the DQ and DQS paths, respectively.

Device Family Support

The ALTDQ and ALTDQ_DQS megafunctions support the following Altera® device families:

- Arria® GX
- Cyclone II
- Cyclone III
- Cyclone IV GX
- HardCopy® II

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

- Stratix®
- Stratix GX
- Stratix II
- Stratix II GX

ALTDQ Megafunction

The ALTDQ megafunction allows you to easily configure the DDR I/O elements in supported Altera devices for DQ data signal functionality. The ALTDQ megafunction is a variation of the ALTDDIO_BIDIR megafunction modified to be used with the ALTDQS megafunction.

The ALTDQ megafunction implements a DDR interface and offers many additional features, which include:

- Transmission and reception of data on both edges of the reference clock
- `ddioinclk` clock input for the negative-edge input register (available for Stratix II devices only)
- Active high asynchronous clear and clock-enable control inputs
- Registered or unregistered output-enable input

ALTDQS Megafunction

The ALTDQS megafunction allows you to easily configure the I/O elements of the data strobe (DQS) in supported Altera devices.

You typically use the ALTDQS megafunction used with the ALTDQ megafunction which provides the following features:

- A group of DQS pins used to strobe the read and write data in external DDR memory interfaces using a common DLL to phase shift the read strobe
- Implementing one DLL and a number of user-specified DQS pins (the maximum number of DQS supported by a DLL is also dependent on the device side)
- Clocks generated for the DQ negative-edge input registers from the DQSn pins that is the `dqddioinclk[]` signal (available for Stratix II devices only)
- Delay buffer setting output option
- Frequency settings of DQS inputs and system reference clock
- Active-high asynchronous clear and clock-enable control inputs
- Registered or unregistered output-enable input
- DQS outputs configurable as open drain mode
- Speed setting of the DQS and delay buffers as either low or high (available for Stratix II devices only)

Common Applications

The ALTDQ and ALTDQS megafunctions implement proprietary interfaces and variations of the external memories that require features not supported by the Altera SDRAM controller and external memory interfaces.

You should use the clear-text data path generated by the DDR SDRAM Controller to implement all DDR SDRAM, DDR2 SDRAM, RLDRAM II, and QDRII systems. This data path has been validated by Altera and you can generate this data path for many variations of these interfaces. To use the clear text data path, you need not purchase or instantiate the Altera DDR SDRAM controller IP.

For more information, refer to the *DDR & DDR2 SDRAM Controller Compiler User Guide*, *RLDRAM II Controller MegaCore Function User Guide*, and *QDRII SRAM Controller MegaCore Function User Guide*.

Resource Utilization and Performance

For details about the resource utilization and performance of ALTDQ and ALTDQS megafunctions, refer to the MegaWizard Plug-In Manager and the compilation reports for each device in the Quartus II software.

Getting Started

The instructions in this section require the Quartus II software version 9.1 or later. For operating system support information, refer to the Operation System Support page on the Altera website.

MegaWizard Plug-In Manager Page Option and Description for ALTDQS Megafunction

Table 1 defines the parameterization options that are available in the MegaWizard Plug-In Manager for the ALTDQS megafunction.

Table 1. MegaWizard Plug-In Manager Page Option and Description

Page	Option	Description
1	**Which action do you want to perform?**	You can select from the following options: **Create a new custom megafunction variation**, **Edit an existing custom megafunction variation**, or **Copy an existing custom megafunction variation**.
2a	**Select a megafunction from the list below**	Select **ALTDQS** from the I/**O** category.
	Which device family will you be using?	Specify the device family you want to use.
	Which type of output file do you want to create?	You can choose from **AHDL** (**.tdf**), **VHDL** (**.vhd**), or **Verilog HDL** (**.v**) as the output file type.
	What name do you want for the output file?	Specify the file name without the file extension.

Page 3 of the ALTDQS parameter editor is the **General** page. Table 2 on page 4 describes options available on page 3 of the ALTDQS megafunction.

Table 2. General Settings (Part 1 of 3)

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
Currently selected device family	Displays the currently selected device family.	Yes	Yes	Yes	Yes
What is the frequency of the DQS input(s) ?	This is the input DQS frequency. *(1)*	Yes	Yes	No	No
How many DQS pins would you like?	Specifies the number of DQS pins that are implemented or the number of DQS/DQSn pin pairs generated.The maximum number of pins possible depends on the chosen device and the **'dqsn_padio' port** option. *(2)*	Yes	Yes	Yes	Yes
Create an output enable for the DQS pins	This enables the DQS output path. Turn on this option to create an output enable for the DQS pins. If you select the **no output enable port is used** option, the `dqs_padio` signal drives out permanently.	Yes	Yes	Yes	Yes
Register the output enable	This enables the DQS OE path. Turn on this option to register the output enable port with the `outclk` signal.	Yes	Yes	Yes	Yes
What will control the DQS/nDQS delay chains?	■ **DLL feedback loop counter controls the DQS/nDQS delay chains** This is the default option. DLL inserts a delay equivalent to requested phase-shift at input clock frequency. Input clock frequency and phase-shift are set on page 4. *(3)* ■ **No DLL is used** No DLL and no delay is added between the DQS/nDQS and the `dqinclk` port.	Yes	Yes	Yes. Always uses DLL.	Yes. Have an option to choose either DLL or no DLL.

Table 2. General Settings (Part 2 of 3)

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
How should the delay chain be specified?	This can be either a setting in **ps** or a value from **0-63**. This is the user-requested delay on the clock delay control block. Delay is specified either by number of delay buffers used or desired time delay. Time delay is converted to number of buffers during compilation. For the actual buffer delay, refer to the respective device data sheet. These buffers have a fixed delay, which is not dependent on input clock frequency clock delay control circuit on each DQS pin allows a phase shift that center-aligns the incoming DQS signals within the data window of their corresponding DQ data signals. *(4)*	Yes	Yes *(5)*	No	No
Allow DQS to be disabled during read post-amble.	Inhibits the `ddioinclk` signal during read postamble (when the DQS transitions from 0 to Z). Stops the `ddioinclk` signal from creating false clocks as the DQS goes to tristate. If selected, adds an `enable_dqs` input port to stop the `ddioinclk` signal.	Yes	Yes *(6)*	No	No

Table 2. General Settings (Part 3 of 3)

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
Invert dqs_padio port (when driving output)	When you select this option, the `dqs_padio` port is inverted, if driven as an output.	Yes	Yes	No	No

Notes to Table 2:

(1) For supported DQS frequencies in these devices, refer to the "Cyclone II DDR Memory Support Overview" section of the *External Memory Interfaces in Cyclone II Devices* chapter in volume 1 of the *Cyclone II Device Handbook* or the "Introduction" section of the *External Memory Interfaces in Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*.

(2) For number of DQS/DQSn pair pins available in supported devices, refer to the *External Memory Interfaces in Arria GX Devices* chapter in volume 2 of the *Arria GX Device Handbook, External Memory Interfaces in Cyclone II Devices* chapter in volume 1 of the *Cyclone II Device Handbook, External Memory Interfaces in Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook, External Memory Interfaces in Stratix and Stratix GX Devices* chapter in volume 2 of the *Stratix Device Handbook, or the External Memory Interfaces in Stratix II and Stratix II GX Devices* chapter in volume 2 of the *Stratix Device Handbook*.

(3) The delay-locked loop (DLL) controls the delay chain settings to achieve a compensated delay for PVT. For example, you can use a DQS read strobe or clock that is edge-aligned to its associated read data to clock the data into I/O registers if the data is delayed before reaching the register. The DLL block computes the necessary delay settings by comparing the period of an input reference clock to the delay through an internal delay chain. For more information about DLL, refer to the "DQS Phase-Shift Circuitry" section of the *External Memory Interfaces in Stratix and Stratix GX Devices* chapter in volume 2 of the *Stratix Device Handbook*, *External Memory Interfaces in Stratix II and Stratix II GX Devices* chapter in volume 2 of the *Stratix Device Handbook*, and *External Memory Interfaces in Arria GX Devices* chapter in volume 2 of the *Arria GX Device Handbook*.

(4) For more information about the clock delay control block, refer to the "Clock Delay Control" section of the *External Memory Interfaces in Cyclone II Devices* chapter in volume 1 of the *Cyclone II Device Handbook*.

(5) For Cyclone III, Cyclone III GX, and Cyclone III LS devices, you must use the "Input Delay from Dual-Purpose Clock Pin" assignment in the Assignment Editor to set DQS clock delay.

(6) For more information about the DQS postamble circuitry, refer to the "DQS Postamble" section of the *External Memory Interfaces in Cyclone II Devices* chapter in volume 1 of the *Cyclone II Device Handbook*.

Page 4 of the ALTDQS parameter editor is the **General 2** page. Table 3 on page 7 describes options available on page 4 of the ALTDQS megafunction.

Table 3. General 2 Settings (Part 1 of 2)

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
What is the frequency of the DQS inputs(s)?	The input clock frequency for the `inclk` or `outclk` signals. For the supported frequencies, refer to the "External Memory" chapter in the respective device handbook.	No	No	Yes	Yes
What is the frequency mode?	Controls internal set up of delay chains. Available options depend upon the DQS frequency you entered. *(1)* For the respective modes, refer to the respective device datasheet.	No	No	Yes	Yes
What is the delay buffer mode?	Only available in custom frequency mode. Delay buffers can be set for **High** or **Low** delay modes.	No	No	Yes	Yes
What is the DLL delay chain length?	Option only available in custom frequency mode. A delay chain length of **10**, **12**, or **16** buffers may be implemented.	No	No	Yes	Yes
How much phase shifting would you like to use for the DQS clock?	Select phase-shift with pull-down options of **0**, **72**, or **90°**. The values calculated from previously specified delay buffer mode and DLL delay chain setting.	No	No	Yes	Yes
Allow DQS to be disabled during read post-amble	Inhibits the `ddioinclk` signal during read postamble (when DQS transitions from 0 to Z). This stops the `ddioinclk` signal from creating false clocks as the DQS goes into a tri-state. The device architecture cannot implement this option on the DQSn port. Therefore, if you select this option, the DQSn port may only be used as an output (or left used). The `ddioinclk` signal is inhibited by a register clocked by the DLL delayed DQS. The `dqs_areset` and `dqs_sreset` signals control this register. You must set the `dqs_sreset` signal to V_{CC} due to architectural constraints and control the `ddioinclk` signal using `dqs_areset` signal.	No	No	Yes	Yes

Table 3. General 2 Settings (Part 2 of 2)

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
How many valid half cycles of the inclk input should pass before the DLL simulates a lock?	Only affects simulation and has no affect on actual device operation. Use to reduce number of clock cycles for which a simulation must be run before the DLL locks. By setting this to **1**, the DLL immediately locks and simulation can begin transferring data.	No	No	Yes	Yes
How many invalid half clock cycles of the inclk input should pass before the DLL simulates a loss of lock? *(2)*	Only affects simulation and has no affect on actual device operation. Use to reduce number of clock cycles for which a simulation must be run before the DLL locks. By setting this to **1**, the DLL immediately locks and simulation can begin transferring data.	No	No	Yes	Yes

Notes to Table 3:

(1) Low/high refers to jitter mode. The DLL in Stratix II device DQS phase-shift circuitry can operate between 100 and 300 MHz in either fast lock mode or low jitter mode. Fast lock mode requires fewer clock cycles to calculate the input clock period, but the low jitter mode is more accurate. The DQS delay settings (the up/down counter output) are updated every eight clock cycles. If the low jitter mode is enabled, the phase comparator also issues a clock-enable signal to the up/down counter notifying the counter when to update the DQS settings. In low jitter mode, the enable signal is only active when the `upndn` signal is incremented or decremented by 4, otherwise the clock-enable is off and the DQS delay settings do not get updated. This enable signal is always active if the DLL is in fast lock mode.

(2) Stratix II devices do not support this feature, and the option is disabled in the MegaWizard Plug-In Manager for these devices.

Page 5 of the ALTDQS parameter editor is the **Output Registers** page. Table 4 on page 9 describes options available on page 5 of the ALTDQS megafunction.

Table 4. Output Register Settings

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
What effect should the 'dqs_areset' port have on output registers?	Use the `dqs_areset` port to asynchronously preset or clear output registers. If you select the **None** option, the signal is not instantiated and you can specify the power-up state of the output registers	Yes	Yes	Yes	Yes
What effect should the 'dqs_sreset' port have on output registers? *(1)*	Use the `dqs_sreset` port to synchronously preset or clear output registers. If you select the **None** option, the port is not instantiated. *(2)*	Yes	Yes	Yes	Yes
How should the output registers power-up? *(1)*	If you selected **None** for the **What effect should the 'dqs_areset' port have on output registers?** option, use this option to specify power-up condition of output registers.	Yes	Yes	Yes	Yes
Use clock enable for the output register *(3)*	Create the `outclkena` port (if not implemented for the output registers). Use as a clock enable for the output registers.	Yes	Yes	Yes	Yes

Notes to Table 4:

(1) Cyclone II devices do not support this feature. Option is disabled when Cyclone II device family is selected.

(2) This option is not available for Stratix II devices if the **Allow DQS to be disabled during read postamble** option has been selected on a previous page of the wizard (refer to Table 3 on page 7).

(3) This option is enabled only when **Register the output enable** option is turned on in page 3 of the MegaWizard Plug-In Manager.

Page 6 of the ALTDQS parameter editor is the **Output Enable Registers** page. Table 5 on page 10 describes options available on page 6 of the ALTDQS megafunction.

Table 5. Output Enable Registers Settings

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
What effect should the 'dqs_areset' port have on output enable registers?	Use the `dqs_areset` port to asynchronously **Preset** or **Clear** the output-enable registers. If set to **None**, the port is not instantiated and you have the option to specify the power-up state of output enable registers.	Yes	Yes	Yes	Yes
What effect should the 'dqs_sreset' port have on output enable registers? *(1)*	Use the `dqs_areset` port to synchronously **Preset** or **Clear** output enable registers. If set to **None**, the `dqs_areset` port is not instantiated. *(2)*	Yes	Yes	Yes	Yes
How should the output enable registers power-up? *(1)*	If **None** is selected for the **What effect should the 'dqs_areset' port have on output enable registers?**option, use this option to specify power-up condition of output-enable registers.	Yes	Yes	Yes	Yes
Hold output drive at high impedance for an extra half-clock cycle when output enable goes high	Delays DQS write mode by half a clock cycle. The DQS transitions from Z to 0, providing a cleaner start to sequence than a Z to 1 transition.	Yes	Yes	Yes	Yes
Use clock enable for the output enable register	Creates the `outclkena` port (if not implemented for output enable register). Use as a clock-enable for output enable register.	Yes	Yes	Yes	Yes

Notes to Table 5:

(1) Cyclone II devices do not support this feature. Option is disabled when Cyclone II device family is selected.

(2) This option is not available for Stratix II devices if the **Allow DQS to be disabled during read post-amble** option has been selected on a previous page of the MegaWizard Plug-In Manager (refer to Table 3 on page 7).

Page 7 of the ALTDQS parameter editor is the **Dsqn_padio Port** page. Table 6 on page 11 describes options available on page 7 of the ALTDQS megafunction.

Table 6. DQSn_padio Port Settings

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
How do you want to use the 'dqsn_padio' port?	Use negative DQS pin as an input during read cycles, and output during write cycles or a bidirectional signal for read and write. If the **Allow DQS to be disabled during read post amble** option has been selected on page 3, only **Not used** and **Output modes** options are available. If **Allow DQS to be disabled during read post-amble** option selected on page 3, all modes are available.	Yes	Yes	No	Yes
What is the phase shift when used in timing analysis?	This is the phase shift amount (**0°**, **30°**, **60°**, **90°**, **120°**) assumed during timing analysis. This is to compute the static delay during timing analysis.	Yes	Yes	No	Yes

Page 8 of the ALTDQS MegaWizard Plug-In Manager is the **DQS/nDQS Delay Chain** page. Table 7 on page 12 describes options available on page 8 of the ALTDQS megafunction.

Table 7. DQS/nDQS Delay Chain Port Settings

Option	Description	Device Supported			
		Cyclone II	Cyclone III, Cyclone III GX, Cyclone III LS	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
Offset options for DQS/nDQS delay chain	Allows tuning of the DQS delay chain. Depending on settings from wizard page 4, the offset applies either a coarse or fine delay. *(1)* When a static delay is added, the value of delay is equivalent to offset value multiplied by coarse or fine offset buffer delay. If a dynamic delay is selected, a `dll_offset` port is added to the megafunction. The unsigned integer values on `dll_offset` port may then be added, subtracted, or dynamically controlled with a `dll_addsub` port by selecting one of the options. For static offset, the value is added to the DLL feedback counter and output on the `dll_delayctrlout` output bus. The legal integer values are **–63** to **63**.	No	No	No	Yes
Enable the latches for the DQS delay chain setting *(2)*	These latches ensure that the offset value is not changed while the DQS transitions, and also determines whether the DQS delay buffer control signals are latched or not.	No	No	No	Yes
Create reset for the DLL	If enabled, a reset input is added to clear the DLL.	No	No	No	Yes

Note to Table 7:

(1) For more information, refer to Table 3 on page 7.

(2) For more information about utilizing the offset, refer to *AN550: Using the DLL Phase Offset Feature in Stratix II and HardCopy II Devices*.

Page 9 of the MegaWizard Plug-In Manager is the **EDA** page. This page lists the simulation model files needed to simulate the generated design files. On this page, you can enable the Quartus II software to generate a synthesis area and timing estimation netlist for this megafunction for use by third-party tools.

Page 10 of the MegaWizard Plug-In Manager is the **Summary** page. On this final page, the wizard displays a list of the types of files to be generated. The automatically generated Variation file contains wrapper code in the language you specified on page 2a. On this page, you can specify additional types of files to be generated. Choose from the AHDL Include file (<*function name*>**.inc**), VHDL component declaration file,

<*function name*>.**cmp**), Quartus II symbol file (<*function name*>**.bsf**), Instantiation template file (<*function name*>**.v**), and Verilog HDL black box file (<*function name*>**_bb.v**). If you select **Generate Netlist** on the **Simulation Model** page, the file for that netlist is also available. A gray checkmark indicates a file that is automatically generated, and a red checkmark indicates generation of an optional file.

MegaWizard Plug-In Manager Page Option and Description ALTDQ Megafunction

Table 8 defines the parameterization options that are available for the ALTDQ megafunction..

Table 8. MegaWizard Plug-In Manager Page Option and Description

Page	Options	Descriptions
1	**Which action do you want to perform?**	You can select from the following options: **Create a new custom megafunction variation**, **Edit an existing custom megafunction variation**, or **Copy an existing custom megafunction variation.**
2a	**Select a megafunction from the list below**	Select **ALTDQ** from the **I/O** category
	Which device family will you be using?	Specify the device family you want to use.
	Which type of output file do you want to create?	You can choose from **AHDL** (**.tdf**), **VHDL** (**.vhd**), or **Verilog HDL** (**.v**) as the output file type.
	What name do you want for the output file?	Specify the file name without the file extension.

Page 3 of the ALTDQ parameter editor is the **Parameter Settings** page. Table 9 on page 14 describes options available on page 3 of the ALTDQ megafunction.

Table 9. Parameter Settings

Option	Description	Supported Devices			
		Cyclone II	Cyclone III, Cyclone IV GX	Stratix, Stratix GX,	Stratix II, Stratix II GX, Arria GX, HardCopy II
Currently selected device family	Specifies the Altera device family you are using.	Yes	Yes	Yes	Yes
How many DQ pins would you like?	Specifies the width of the data buses. If you are using the Quartus II software version 6.0 or earlier, this megafunction displays output ports as `dataout_h[]` and `dataout_l[]`; the Quartus II software version 6.0 and later displays output ports as `dataout[]` and `dataout_ddio[]`.	Yes	Yes	Yes	Yes
Which asynchronous reset port would you like?	You can use the asynchronous clear (`aclr`) or the asynchronous preset (`aset`) as the asynchronous reset. If you do not use either clear option, you must specify whether the registers should power-up high or low.	Yes	Yes	Yes	Yes
Create a clock enable port for each clock port	Creates an input clock enable port (`inclken`) and an output clock.	Yes	Yes	Yes	Yes
Create an output enable port	Creates an output enable port (`oe`) for this instance of the ALTDQ instance.	Yes	Yes	Yes	Yes
Register output enable	Sets the `OE_REGISTER_MODE` parameter. When enabled, a register is placed in the OE path and the parameter is set to register. When disabled, parameter defaults to **NONE**.	Yes	Yes	Yes	Yes
Delay switch-on by a half clock cycle	Sets the `EXTEND_OE_DISABLE` parameter. When enabled, the pin does not drive out until the falling edge of the `outclk` signal. When enabled, the parameter is set to **TRUE**, otherwise it defaults to **FALSE**.	Yes	Yes	Yes	Yes
Invert Input Clock	If enabled, the first bit of data is captured on the rising edge of the input clock; if not enabled, it is captured on the falling edge of the input clock.	Yes	Yes	Yes	Yes
Use ddioinclk port (from DQSn bus)	Creates a `ddioinclk` port. This port clocks the negative edge triggered input/capture register of the ALTDQ instance.	Yes	Yes	Yes	Yes

Page 4 of the MegaWizard Plug-In Manager is the **EDA** page. This page lists the simulation model files needed to simulate the generated design files. On this page, you can enable the Quartus II software to generate a synthesis area and timing estimation netlist for this megafunction for use by third-party tools.

Page 5 of the MegaWizard Plug-In Manager is the **Summary** page. On this final page, the ALTDQ MegaWizard Plug-In Manager displays a list of the types of files to be generated. The automatically generated Variation file contains wrapper code in the language you specified on page 2a. On this page, you can specify additional types of files to be generated. Choose from the AHDL Include file (<*function name*>.**inc**), VHDL component declaration file, (<*function name*>**.cmp**), Quartus II symbol file (<*function name*>**.bsf**), Instantiation template file (<*function name*>**.v**), and Verilog HDL black box file (<*function name*>**_bb.v**). If you select **Generate Netlist** on the **Simulation Model** page, the file for that netlist is also available. A gray checkmark indicates a file that is automatically generated, and a red checkmark indicates generation of an optional file.

Design Example: Implement DDR I/O Interface

This design example uses the ALTDQ and ALTDQS megafunction to implement DDR I/O interface. This example uses the MegaWizard Plug-In Manager in the Quartus II software. As you go through the wizard, each page is described in detail.

Design Files

The design files are available in the Examples for ALTDQ and ALTDQS Megafunction User Guide page on the Altera website.

Example

In this example, you perform the following tasks:

- Create a DDR I/O interface using the ALTDQ and ALTDQS megafunctions and the MegaWizard Plug-in Manager
- Implement the design and assign the EP2C5T144C6 device to the project
- Compile and simulate the design

Create an ALTDQ Megafunction

Perform the following tasks to create an ALTDQS megafunction:

1. Unzip the **altdq_dqs_DesignExample.zip** to any working directory.
2. In the Quartus II software, open the **dq_dqs_ex.qar** project.
3. On the Tools menu, click **MegaWizard Plug-In Manager**.

Table 10 shows the MegaWizard Plug-In Manager page options and description you should select to create the example ALTDQS megafunction.

Table 10. Parameter Settings for ALTDQ Megafunction

Page	Option	Description
1	**Which action do you want to perform?**	Select **Create a new custom megafunction variation**
2a	**Select a megafunction from the list below**	Select **ALTDQ** from the **I/O** category
	Which device family will you be using?	Select **Cyclone II**
	Which type of output file do you want to create?	Select **Verilog HDL**
	What name do you want for the output file?	Browse to the folder **dq_dqs_ex_1.0_restored**. Name the file **dq**. (If asked if it is okay to overwrite an existing file, click **OK**)
3	**How many DQ pins would you like?**	Select **8**
	Which asynchronous reset port would you like?	**Select Asynchronous clear (aclr)**
	Create a clock enable port for each clock port	Turn off option
	Create an output enable port	Turn on option
	Register output enable	Turn off option
	Delay switch-on by a half clock cycle	Turn off option
	Invert input clock	Turn off option
	Use 'ddioinclk' port (from DQSn bus)	Turn off option

Page 4 of the MegaWizard Plug-In Manager allows you to specify options for stimulation and timing and resource estimation. This page normally lists the simulation libraries required for functional simulation by third-party tools. However, the ALTDQS megafunction does not have simulation model files, and cannot be simulated.

Page 5 of the MegaWizard Plug-In Manager allows you to specify the generated file types. The Variation file contains wrapper code in the HDL you specified on page 2a. You can optionally generate Pin Planner ports PPF file (**.ppf**), AHDL Include file (<*function name*>**.inc**), VHDL component declaration file (<*function name*>**.cmp**), Quartus II symbol file (<*function name*>**.bsf**), Instantiation template file (<*function name*>**.v**), and Verilog HDL black box file (<*function name*>**_syn.v**) is also available. A gray check marks indicate files that are always generated; the other files are optional and are generated only if selected (indicated by a red check mark). Turn on the boxes to select the files that you want the wizard to generate. Perform these steps to continue creating an ALTDQ megafunction:

1. Turn on the **Instantiation template file** and **Verilog 'Black Box' declaration file** options.
2. Turn off the **AHDL Include file**, **VHDL Component declaration file**, and **Quartus symbol file** options.
3. Click **Finish**.

The ALTDQ module is generated. The next section shows you how to create an ALTDQS megafunction.

Table 11 shows the MegaWizard Plug-In Manager page options and description you should select to create the example ALTDQS megafunction.

Table 11. Parameter Settings for ALTDQS Megafunction

Page	Option	Description
1	**Which action do you want to perform?**	Select **Create a new custom megafunction variation**
2a	**Select a megafunction from the list below**	Select **ALTDQS** from the **I/O** category
	Which device family will you be using?	Select **Cyclone II**
	Which type of output file do you want to create?	Select **Verilog HDL**
	What name do you want for the output file?	Browse to the folder **dq_dqs_ex_1.0_restored**. Name the file **dqs**. (If asked if it is okay to overwrite an existing file, click **OK**)
3	**How many DQS pins would you like?**	Select **1**
4	**What is the frequency of the DQS input(s)?**	Select **133.333 MHz**
	Create an output enable port	Turn on option
	Register the output enable	Turn off option
	How should the delay chain be specified?	Select **As delay chain setting** and select **50**
	Allow DQS to be disabled during read post-amble	Turn off option
	Invert 'dqs_padio' port (when driving output)	Turn off option
5	**What effect should the 'dqs_areset' port have on output registers?**	Select **Clear**
6	**How do you want to use the 'dqsn_padio' port?**	Select **Not Used**

Page 7 of the MegaWizard Plug-In Manager lists the simulation model file needed to properly simulate the generated design files. No further input is needed. Perform these steps to continue creating an ALTDQS megafunction:

1. Turn on the **Instantiation template file** and **Verilog 'Black Box' declaration file** options.
2. Turn off the **AHDL Include file**, **VHDL Component declaration file**, and **Quartus symbol file** options.
3. Click **Finish**.

The ALTDQS module is generated.

Combine ALTDQ and ALTDQS Modules to Create a DDR I/O Interface

This section describes how to create a new top-level Verilog HDL file that combines the ALTDQ and ALTDQS modules.

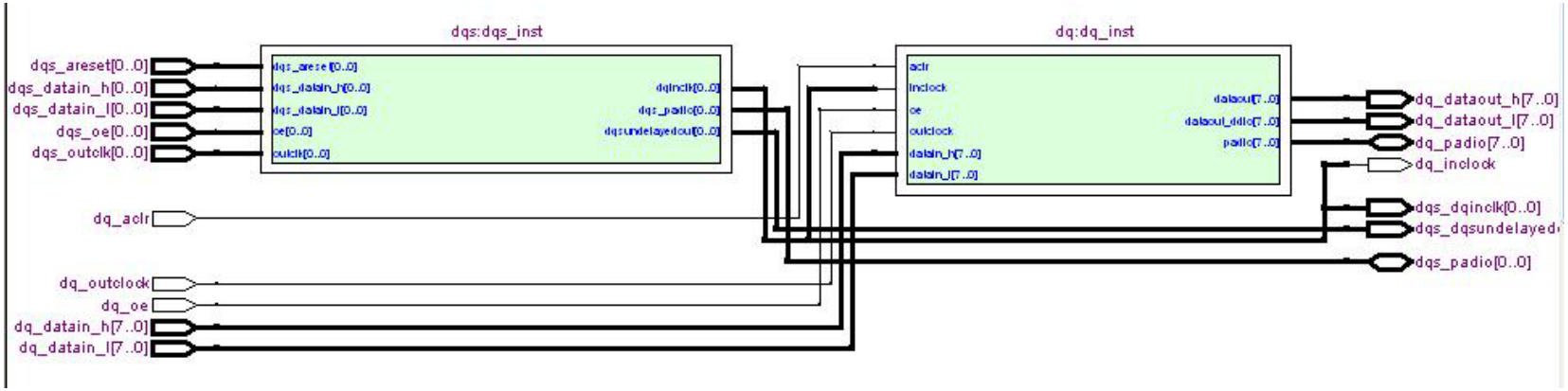
1. With the **dq_dqs_ex.qar** project open, open the **dq_dqs_ex.v** file.
2. Verify that the DQ and DQS functions are correctly connected in the top-level **dq_dqs_ex.v** file (refer to Figure 1 on page 18).
3. On the Project menu, click **Add/Remove Files in Project**. The **Settings** window appears.

4. In the **Settings** window, browse to the **dq_dqs_ex.v** file in the project folder. Click **Open**, then **Add** to add the file to the project.
5. Click **OK**.

The top-level file is now added to the project. You have now created the complete design file shown in Figure 1 on page 18. You can view the block diagram after compiling the project inthe RTL Viewer.

The schematic shown in Figure 1 on page 18 is not included in the project file.

Figure 1. DDR I/O Interface Using ALTDQ and ALTDQS Megafunctions



Implement the DDR I/O Interface Design

This section describes how to assign the EP2C5T144C6 device to the project and compile the project.

1. With the **dq_dqs_ex.qar** project open, on the Assignments menu, click **Device**. The device page of the **Settings** dialog box appears.
2. In the **Family** list, select **Cyclone II**.
3. In the **Target device** list, select **Specific device selected in 'Available devices' list**.
4. In the **Available devices** list, select **EP2C5T144C6**.
5. Leave the default settings for the other options on the **Settings** page and click **OK**.
6. On the Processing menu, click **Start Compilation**, compile the design.

7. When a message indicates that **Full compilation was successful**, click **OK**.

Simulate the DDR I/O Interface Design in ModelSim-Altera Tool

You can simulate the design in the ModelSim® tool to compare the results of both simulators.

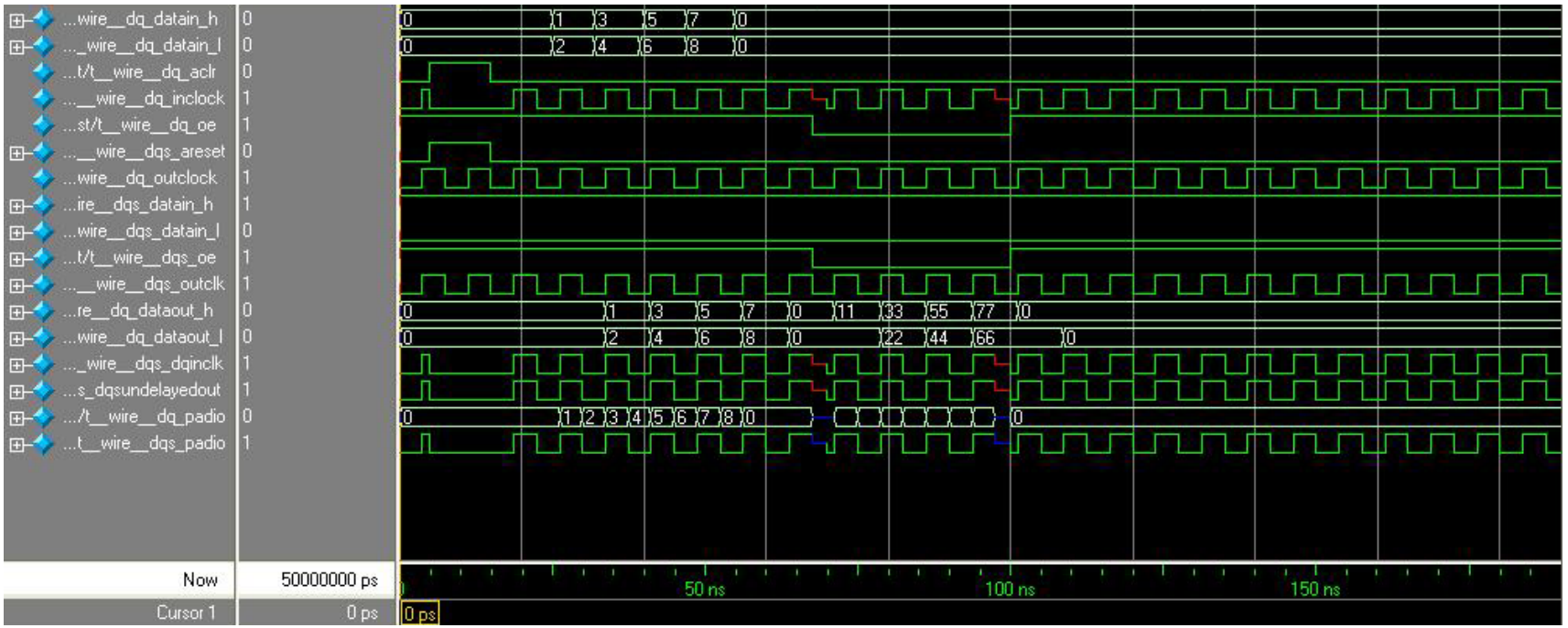
This user guide assumes that you are familiar with using the ModelSim-Altera tool before trying out the design example. If you are unfamiliar with this tool, refer to the ModelSim-Altera Software Support page on the Altera website. There are various links to topics such as installation, usage, and troubleshooting.

Set up the ModelSim-Altera simulator by performing the following steps.

1. Unzip the **altdq_dqs_msim.zip** file to any working directory on your PC.
2. Browse to the folder in which you unzipped the files and open the **altdqdqs.do** file in a text editor.
3. In line 1 of the **altdqdqs.do** file, replace <*insert_directory_path_here*> with the directory path of the appropriate library files. For example, `C:/Modeltech_ae/altera/verilog/cycloneii`
4. On the File menu, click **Save**.
5. Start **ModelSim-Altera**.
6. On the File menu, click **Change Directory**.
7. Select the folder in which you unzipped the files. Click **OK**.
8. On the Tools menu, click **Execute Macro**.
9. Select the **altdqdqs.do** file and click **Open**. This is a script file for ModelSim that automates all the necessary settings for the simulation.
10. Verify the results shown in the **Waveform Viewer** window.

You may need to rearrange signals, remove redundant signals, and change the radix to suit the results in the Quartus II Simulator. Figure 2 on page 20 shows the expected simulation results in ModelSim-Altera.

Figure 2. ModelSim Simulation Results



Specifications

This section describes the prototypes, component declarations, ports, and parameters of the ALTDQ and ALTDQS megafinctions. These ports and parameters are available to customize the ALTDQ and ALTDQS megafunctions according to your application.

Verilog HDL Prototype

You can locate the following Verilog HDL prototypes in the Verilog Design File (**.v**) **altera_mf.v** in the <*Quartus II installation directory*>**\eda\synthesis** directory.

ALTDQ

```
module altdq
#(     parameter     ddioinclk_input = "NEGATED_INCLK",
       parameter     intended_device_family = "unused",
       parameter     extend_oe_disable = "OFF",
       parameter     invert_input_clocks = "ON",
       parameter     number_of_dq = 1,
       parameter     oe_reg = "UNREGISTERED",
       parameter     power_up_high = "OFF",
       parameter     lpm_type = "altdq",
       parameter     lpm_hint = "unused")
(      input wire    aclr,
       input wire    aset,
       input wire    [number_of_dq-1:0]   datain_h,
       input wire    [number_of_dq-1:0]   datain_l,
       output wire   [number_of_dq-1:0]   dataout_h,
       output wire   [number_of_dq-1:0]   dataout_l,
       input wire    ddioinclk,
       input wire    inclock,
       input wire    inclocken,
       input wire    oe,
       input wire    outclock,
       input wire    outclocken,
       inout wire    [number_of_dq-1:0]   padio)/* synthesis syn_black_box=1 */;
endmodule //altdq
```

ALTDQS

```
module altdqs
#(      parameter     delay_buffer_mode = "low",
        parameter     delay_chain_mode = "static",
        parameter     intended_device_family = "unused",
        parameter     dll_delay_chain_length = 12,
        parameter     dll_delayctrl_mode = "normal",
        parameter     dll_jitter_reduction = "true",
        parameter     dll_offsetctrl_mode = "none",
        parameter     dll_phase_shift = "unused",
        parameter     dll_static_offset = "0",
        parameter     dll_use_reset = "false",
        parameter     dll_use_upndnin = "false",
        parameter     dll_use_upndninclkena = "false",
        parameter     dqs_ctrl_latches_enable = "true",
        parameter     dqs_delay_chain_length = 3,
        parameter     dqs_delay_chain_setting = "0",
        parameter     dqs_delay_requirement = "unused",
        parameter     dqs_edge_detect_enable = "false",
        parameter     dqs_oe_async_reset = "none",
        parameter     dqs_oe_power_up = "low",
        parameter     dqs_oe_register_mode = "register",
        parameter     dqs_oe_sync_reset = "none",
        parameter     dqs_open_drain_output = "false",
        parameter     dqs_output_async_reset = "none",
        parameter     dqs_output_power_up = "low",
        parameter     dqs_output_sync_reset = "none",
        parameter     dqs_use_dedicated_delayctrlin = "true",
        parameter     dqsn_mode = "none",
        parameter     extend_oe_disable = "true",
        parameter     gated_dqs = "false",
        parameter     has_dqs_delay_requirement = "true",
        parameter     input_frequency = "unused",
        parameter     invert_output = "false",
        parameter     number_of_dqs = 1,
        parameter     number_of_dqs_controls = 1,
        parameter     sim_invalid_lock = 100000,
        parameter     sim_valid_lock = 1,
        parameter     tie_off_dqs_oe_clock_enable = "false",
        parameter     tie_off_dqs_output_clock_enable = "false",
        parameter     lpm_type = "altdqs",
        parameter     lpm_hint = "unused")
(       input wire    dll_addnsub,
        output wire   [5:0]   dll_delayctrlout,
        input wire    [5:0]   dll_offset,
        input wire    dll_reset,
        input wire    dll_upndnin,
        input wire    dll_upndninclkena,
        output wire   dll_upndnout,
        output wire   [number_of_dqs-1:0]   dqddioinclk,
        output wire   [number_of_dqs-1:0]   dqinclk,
        input wire    [number_of_dqs_controls-1:0]   dqs_areset,
        input wire    [number_of_dqs-1:0]   dqs_datain_h,
        input wire    [number_of_dqs-1:0]   dqs_datain_l,
        input wire    [5:0]   dqs_delayctrlin,
        inout wire    [number_of_dqs-1:0]   dqs_padio,
        input wire    [number_of_dqs_controls-1:0]   dqs_sreset,
        inout wire    [number_of_dqs-1:0]   dqsn_padio,
        output wire   [number_of_dqs-1:0]   dqsundelayedout,
        input wire    [number_of_dqs-1:0]   enable_dqs,
        input wire    inclk,
        input wire    [number_of_dqs_controls-1:0]   oe,
        input wire    [number_of_dqs_controls-1:0]   outclk,
        input wire    [number_of_dqs_controls-1:0]   outclkena)/* synthesis syn_black_box=1 */;
endmodule //altdqs
```

VHDL Component Declaration

You can locate the following VHDL component declarations in the VHDL Design File (**.vhd**) **altera_mf.vhd** in the <*Quartus II installation directory*>**\libraries\vhdl\altera_mf** directory.

ALTDQ

```
component altdq
          generic (
                  ddioinclk_input :         string := "NEGATED_INCLK";
                  intended_device_family   :        string := "unused";
                  extend_oe_disable        :        string := "OFF";
                  invert_input_clocks      :        string := "ON";
                  number_of_dq    :         natural;
                  oe_reg  :       string := "UNREGISTERED";
                  power_up_high   :         string := "OFF";
                  lpm_hint        :         string := "UNUSED";
                  lpm_type        :         string := "altdq
          );
          port(
                  aclr    :       in std_logic := '0';
                  aset    :       in std_logic := '0';
                  datain_h        :        in std_logic_vector(number_of_dq-1 downto 0);
                  datain_l        :        in std_logic_vector(number_of_dq-1 downto 0);
                  dataout_h       :        out std_logic_vector(number_of_dq-1 downto 0);
                  dataout_l       :        out std_logic_vector(number_of_dq-1 downto 0);
                  ddioinclk       :        in std_logic := '0';
                  inclock :       in std_logic;
                  inclocken       :        in std_logic := '1';
                  oe      :       in std_logic := '1';
                  outclock        :         in std_logic;
                  outclocken      :         in std_logic := '1';
                  padio   :       inout std_logic_vector(number_of_dq-1 downto 0)
          );
end component;
```

ALTDQS

```
component altdqs
        generic (
                delay_buffer_mode        :        string := "low";
                delay_chain_mode         :        string := "static";
                intended_device_family   :        string := "unused";
                dll_delay_chain_length   :        natural := 12;
                dll_delayctrl_mode       :        string := "normal";
                dll_jitter_reduction     :        string := "true";
                dll_offsetctrl_mode      :        string := "none";
                dll_phase_shift  :       string := "unused";
                dll_static_offset        :        string := "0";
                dll_use_reset    :       string := "false";
                dll_use_upndnin  :       string := "false";
                dll_use_upndninclkena    :        string := "false";
                dqs_ctrl_latches_enable  :        string := "true";
                dqs_delay_chain_length   :        natural := 3;
                dqs_delay_chain_setting  :        string := "0";
                dqs_delay_requirement    :        string := "unused";
                dqs_edge_detect_enable   :        string := "false";
                dqs_oe_async_reset       :        string := "none";
                dqs_oe_power_up  :       string := "low";
                dqs_oe_register_mode     :        string := "register";
                dqs_oe_sync_reset        :        string := "none";
                dqs_open_drain_output    :        string := "false";
                dqs_output_async_reset   :        string := "none";
                dqs_output_power_up      :        string := "low";
                dqs_output_sync_reset    :        string := "none";
                dqs_use_dedicated_delayctrlin    :        string := "true";
                dqsn_mode        :       string := "none";
                extend_oe_disable        :        string := "true";
                gated_dqs        :       string := "false";
                has_dqs_delay_requirement        :        string := "true";
                input_frequency  :       string;
                invert_output    :       string := "false";
                number_of_dqs    :       natural;
                number_of_dqs_controls  :        natural := 1;
                sim_invalid_lock        :        natural := 100000;
                sim_valid_lock   :       natural := 1;
                tie_off_dqs_oe_clock_enable      :        string := "false";
                tie_off_dqs_output_clock_enable  :        string := "false";
                lpm_hint         :       string := "UNUSED";
                lpm_type         :       string := "altdqs"
        );
        port(
                dll_addnsub      :       in std_logic := '0';
                dll_delayctrlout         :        out std_logic_vector(5 downto 0);
                dll_offset       :       in std_logic_vector(5 downto 0) := (others => '0');
                dll_reset        :       in std_logic := '0';
                dll_upndnin      :       in std_logic := '0';
                dll_upndninclkena        :        in std_logic := '1';
                dll_upndnout     :       out std_logic;
                dqddioinclk      :       out std_logic_vector(number_of_dqs-1 downto 0);
                dqinclk  :       out std_logic_vector(number_of_dqs-1 downto 0);
                dqs_areset       :       in std_logic_vector(number_of_dqs_controls-1 downto 0) := (others => '0');
                dqs_datain_h     :       in std_logic_vector(number_of_dqs-1 downto 0);
                dqs_datain_l     :       in std_logic_vector(number_of_dqs-1 downto 0);
                dqs_delayctrlin  :       in std_logic_vector(5 downto 0) := (others => '0');
                dqs_padio        :       inout std_logic_vector(number_of_dqs-1 downto 0);
                dqs_sreset       :       in std_logic_vector(number_of_dqs_controls-1 downto 0) := (others => '0');
                dqsn_padio       :       inout std_logic_vector(number_of_dqs-1 downto 0);
                dqsundelayedout  :       out std_logic_vector(number_of_dqs-1 downto 0);
                enable_dqs       :       in std_logic_vector(number_of_dqs-1 downto 0) := (others => '1');
                inclk    :       in std_logic := '0';
                oe       :       in std_logic_vector(number_of_dqs_controls-1 downto 0) := (others => '1');
                outclk   :       in std_logic_vector(number_of_dqs_controls-1 downto 0);
                outclkena       :        in std_logic_vector(number_of_dqs_controls-1 downto 0) := (others => '1')
        );
end component;
```

VHDL LIBRARY-USE Declaration

The VHDL LIBRARY-USE declaration is not required if you use the VHDL component declaration.

```
LIBRARY altera_mf;
USE altera_mf.altera_mf_components.all;
```

Ports and Parameters

This section describes all of the ports and parameters that are available for the ALTDQ and ALTDQS megafunctions.

The parameter details are only relevant if you bypass the MegaWizard Plug-In Manager interface and use the megafunction as a directly parameterized instantiation in your design. The details of these parameters are hidden if you use the MegaWizard Plug-In Manager interface.

Table 12 shows the input ports for the ALTDQ megafunction.

Table 12. ALTDQ Megafunction Input Ports

Port Name	Required	Description
aclr	No	Asynchronous clear input. If the aclr port is connected, the aset port cannot be used.
aset	No	Asynchronous set input.If the aset port is connected, the aclr port cannot be used.
datain_h[]	Yes	Data to be output to the padio port at the rising edge of the outclk signal. The size of the port is dependent on the NUMBER_OF_DQ parameter.
datain_l[]	Yes	Data to be output to the padio port at the falling edge of the outclk signal. The size of the port is dependent on the NUMBER_OF_DQ parameter.
ddioinclk	No	Clock input for the negative-edge input register. If omitted, the default is GND. *(1)*
inclock	Yes	Clock input that drives the data strobe.
inclocken	No	Clock enable for the inclock port
oe	No	Output enable signal. The oe port defaults to V_{CC} when enabled.
outclock	Yes	Clock signal for the output and oe registers.
outclocken	No	Clock enable signal for each clock port.

Note to Table 12:

(1) Not available for Stratix and Stratix GX devices.

Table 13 shows the output ports for the ALTDQ megafunction.

Table 13. ALTDQ Megafunction Output Ports

Port Name	Required	Description
dataout_h[]	Yes	Data sampled from the padio port at the rising edge of the inclock signal. Output port [NUMBER_OF_DQ - 1..0] wide.
dataout_l[]	Yes	Data sampled from the padio port at the rising edge of the inclock signal. Output port [NUMBER_OF_DQ - 1..0] wide.

Table 14 shows the bidirectional ports for the ALTDQ megafunction.

Table 14. ALTDQ Megafunction Bidirectional Ports

Port Name	Required	Description
padio[]	Yes	Bidirectional double-data rate (DDR) port that should directly feed a bidirectional pin in the top-level design. The size of the bidirectional port is dependent on the NUMBER_OF_DQ parameter.

Table 15 shows the parameters for the ALTDQ megafunction.

Table 15. ALTDQ Megafunction Parameters

Parameter	Type	Required	Description
DDIOINCLK_INPUT	String	No	Specifies whether to feed the ddioinclk ports. The values are DQSB_BUS or NEGATED_INCLK. If omitted, the default value is NEGATED_INCLK. You can use the DQSB_BUS value with Stratix II devices only.
EXTEND_OE_DISABLE	String	No	Specifies whether to use the second OE register. The values are TRUE or FALSE. If omitted, the default value is FALSE.
INVERT_INPUT_CLOCKS	String	No	Specifies whether to invert the input clocks. The INVERT_INPUT_CLOCKS parameter should be used with DDR memory. When the input clock is inverted, the first bit of data is captured on a rising-edge clock; when the input clock in not inverted, the first bit of data is captured on a falling-edge clock.
NUMBER_OF_DQ	Integer	Yes	Specifies the number of DQ pins.
OE_REG	String	No	Specifies whether to register the oe port. The values are REGISTERED or UNREGISTERED.
POWER_UP_HIGH	String	No	Specifies the power-up condition of the I/O registers. The values are ON or OFF. If omitted, the default value is OFF.
LPM_HINT	String	No	Allows you to specify Altera-specific parameters in VHDL Design Files (**.vhd**). The default is UNUSED.
LPM_TYPE	String	No	Identifies the library of parameterized modules (LPM) entity name in VHDL Design Files.
INTENDED_DEVICE_FAMILY	String	No	This parameter is used for modeling and behavioral simulation purposes. Create the ALTDQ megafunction with the MegaWizard Plug-in Manager to calculate the value for this parameter.

Table 16 shows the input ports for the ALTDQS megafunction.

Table 16. ALTDQS Megafunction Input Ports

Port Name	Required	Description
dqs_areset	No	Asynchronous set or reset signal for DQS output and output enable registers.
dqs_datain_h[]	Yes	Data input port for DQS output register which outputs on rising edge of outclk signal.The size of the input port is dependent on the NUMBER_OF_DQS parameter.
dqs_datain_l[]	Yes	Data input port for DQS output register which outputs on falling edge of outclk signal. The size of the input port is dependent on the NUMBER_OF_DQS parameter.
dqs_sreset	No	Synchronous set or reset signal for DQS output and output-enable registers.
inclk	Yes	System reference clock port that drives DLL.
oe	No	Output enable for DQS output registers. When enabled, the oe port defaults to V_{CC}.
outclk	Yes	Clock to DQS output and output-enable registers.
outclkena	No	Clock enable port for DQS output and oe registers. The oe port defaults to V_{CC} when enabled.
dll_addnsub	No	Bus for DLL delay setting offset that adds or subtracts. If omitted, value is GND. *(1)*
dll_upndnin	No	Data input for DLL delay setting offset. If omitted, value is GND. *(1)*
dll_offset[]	No	Data input for DLL delay setting offset. The width of the input port is 6-bit wide. If omitted, value is GND. *(1)*
dqs_delayctrlin[]	No	Control input to DQS delay buffers. The width of the input port is 6-bit wide. If omitted, value is GND. *(1)*
dll_upndninclkena	No	Clock enable for DLL delay setting offset. If omitted, value is GND. *(1)*
enable_dqs	No	Specifies whether DQS is disabled during post-amble read. The enable_dqs port is only available if the GATED_DQS parameter is specified to TRUE. *(2)*

Notes to Table 16:

(1) Available for Stratix II devices only.

(2) Available for Cyclone II devices only.

Table 17 shows the output ports for the ALTDQS megafunction.

Table 17. ALTDQS Megafunction Output Ports (Part 1 of 2)

Port Name	Required	Description
dqinclk[]	Yes	Phase shifted DQS strobe generated for DQ input registers from the DQS input. Width of bus is equal to number of DQS pins. The size of the output port is dependent on the NUMBER_OF_DQS parameter.
dll_delayctrlout[]	No	Delay buffer setting output.If omitted, the default value is GND. The width of the output port is 6-bit wide. *(1)*
dll_upndnout	No	Output for DLL phase comparator. *(1)*
dqddioinclk[]	No	Clocks generated for DQ negative-edge input registers from DQSn pins. The width of the bus is equal to the number of DQS pins. The size of the output port is dependent on the NUMBER_OF_DQS parameter. *(1)*

Table 17. ALTDQS Megafunction Output Ports (Part 2 of 2)

Port Name	Required	Description
dqsundelayedout	No	Non-delayed outputs from the DQS pins. Width of bus is equal to number of DQS pins. The size of the output port is dependent on the NUMBER_OF_DQS parameter.

Note to Table 17:

(1) Available for Stratix II devices only.

Table 18 shows the bidirectional ports for the ALTDQS megafunction.

Table 18. ALTDQS Megafunction Bidirectional Ports

Port Name	Required	Description
dqs_padio[]	Yes	Bidirectional DQS pins. Width of bus is equal to number of DQS pins. The size of the bidirectional port is dependent on the NUMBER_OF_DQS parameter.
dqsn_padio[]	No	Bidirectional DQSn pins. Width of bus is equal to number of DQSn pins. The size of the bidirectional port is dependent on the NUMBER_OF_DQS parameter. *(1)*

Note to Table 18:

(1) Available for Stratix II devices only.

Table 19 shows the parameters for the ALTDQS megafunction.

Table 19. ALTDQS Megafunction Parameters (Part 1 of 4)

Parameter	Type	Required	Description
DLL_PHASE_SHIFT	String	Yes	Specifies DLL phase shift. The values are 0, 72, or 90.
DQS_OE_ASYNC_RESET	String	No	Specifies whether the dqs_areset port clears, presets, or has no effect on the oe register. The values are CLEAR, PRESET, or NONE. If DQS_OE_ASYNC_RESET parameter is specified to CLEAR or PRESET, dqs_areset port is required. If omitted, the default value is NONE.
DQS_OE_POWER_UP	String	No	Specifies power-up condition of the oe registers. The values are HIGH or LOW. If omitted, the default value is LOW.
DQS_OE_REGISTER_MODE	String	No	Specifies whether the oe port is registered. The values are REGISTER or NONE. If omitted, the default value is NONE.
DQS_OE_SYNC_RESET	String	No	Specifies whether the dqs_sreset port clears, presets, or has no effect on the oe register. The values are CLEAR, PRESET, or NONE. If DQS_OE_SYNC_RESET parameter is specified to CLEAR or PRESET, the dqs_sreset port is required. If omitted, the default value is NONE.
DQS_OPEN_DRAIN_OUTPUT	String	No	Specifies whether to use open drain mode. The values are TRUE or FALSE. If omitted, the default value is FALSE.
DQS_OUTPUT_ASYNC_RESET	String	No	Specifies whether the dqs_areset port clears, presets, or has no effect on the DQS output registers. The values are CLEAR, PRESET, or NONE. If the DQS_OUTPUT_ASYNC_RESET port is specified to CLEAR or PRESET, the dqs_areset is required. If omitted, the default value is NONE.
DQS_OUTPUT_POWER_UP	String	No	Specifies power-up condition of DQS output registers. The values are HIGH or LOW. If omitted, the default value is LOW.

Table 19. ALTDQS Megafunction Parameters (Part 2 of 4)

Parameter	Type	Required	Description
DQS_OUTPUT_SYNC_RESET	String	No	Specifies whether the dqs_sreset port clears, presets, or has no effect on DQS output registers. The values are CLEAR, PRESET, or NONE. If DQS_OUTPUT_SYNC_RESET port is specified to CLEAR or PRESET, dqs_sreset is required. If omitted, the default value is NONE.
EXTEND_OE_DISABLE	String	No	Specifies whether to use second oe register. The values are TRUE or FALSE. When the EXTEND_OE_DISABLE is set to TRUE, the output drive is held at high impedance for an extra half clock cycle when the oe port goes high.
INPUT_FREQUENCY	String	No	Specifies frequency of DQS inputs and system reference clock.
NUMBER_OF_DQS	Integer	Yes	Specifies number of DQS pins that are implemented.
SIM_INVALID_LOCK	Integer	No	Specifies number of half-cycles that DLL keeps signal locked after a bad clock is detected. The default is 32 half-cycles.
SIM_VALID_LOCK	Integer	No	Specifies number of half-cycles required before the DLL locks onto signal. The default value is 1.
TIE_OFF_DQS_OUTPUT_CLOCK_ENABLE	String	No	Specifies whether clock enable for output registers is tied-off (does not affect the output registers). The values are TRUE or FALSE. If omitted, the default value is FALSE.
TIE_OFF_DQS_OE_CLOCK_ENABLE	String	No	Specifies whether clock enable for oe registers that are controlled by the outclkena port should be tied off. The values are TRUE or FALSE. If omitted, the default value is FALSE.
DELAY_BUFFER_MODE	String	No	Specifies speed of DLL and DQS delay buffers. The values are LOW or HIGH. If omitted, the default value is LOW. *(1)*
DLL_DELAY_CHAIN_LENGTH	Integer	No	Specifies number of delay buffers used in DLL loop. The values are 0, 1, 2, 3, or 4. If omitted, the default value is 3. *(1)*
DLL_DELAYCTRL_MODE	String	No	Specifies delay control mode used to feed DQS and DQSn delay buffers. The values are NORMAL, NORMAL_OFFSET, OFFSET_ONLY, or NONE. If omitted, the default value is NORMAL. *(1)*
DLL_JITTER_REDUCTION	String	No	Enables or disables jitter reduction on the dll_delayctrlout output ports. The values are TRUE or FALSE. If omitted, the default value is TRUE. *(1)*
DLL_OFFSETCTRL_MODE	String	No	Specifies DLL phase offset mode used with DQS delay buffer control. The values are DYNAMIC_ADD, DYNAMIC_SUB, DYNAMIC_ADDNSUB, STATIC, or NONE. If omitted, the default value is NONE. *(1)*
DLL_STATIC_OFFSET	Integer	No	Adds a value to the DLL feedback counter and output on the dll_delayctrlout output bus. The legal integer values are –63 to 63. If omitted, default is 0. If the DLL_OFFSETCTRL_MODE parameter is set to a value other than STATIC, the DLL_STATIC_OFFSET parameter is ignored. *(1)*
DLL_USE_UPNDNIN	String	No	Specifies whether to use the dll_upndnin port to update the DLL counter. The values are TRUE or FALSE. If omitted, default value is FALSE. If the DLL_USE_UPNDNIN parameter is set to TRUE, DLL_JITTER_REDUCTION parameter must be set to FALSE. *(1)*

Table 19. ALTDQS Megafunction Parameters (Part 3 of 4)

Parameter	Type	Required	Description
DLL_USE_UPNDNINCLKENA	String	No	Specifies whether to use the dll_upndninclkena port as a clock enable for DLL counter. The values are TRUE or FALSE. If omitted, the default value is FALSE. If the DLL_USE_UPNDNINCLKENA parameter is set to TRUE, the DLL_USE_UPNDNINCLKENA parameter overrides DLL control of the clock enable for DLL counter. *(1)*
DQS_CTRL_LATCHES_ ENABLE	String	No	Enables or disables latches for DQS delay buffer control signals. The values are TRUE or FALSE. If omitted, default is TRUE. If the DLL_DELAYCTRL_MODE parameter is set to NONE, the DQS_CTRL_LATCHES_ENABLE parameter cannot be set to TRUE. *(1)*
DQS_DELAY_CHAIN_ LENGTH	Integer	No	Specifies number of delay buffers used in DQS delay chain. The values are 0, 1, 2, 3, or 4. If omitted, the default value is 3. *(1)*
DQS_EDGE_DETECT_ ENABLE	String	No	Specifies whether edge detection prevents updates to DQS delay buffer control latches during a DQS transition. The values are TRUE or FALSE. If omitted, default is FALSE. If DQS_CTRL_LATCHES_ENABLE parameter is set to FALSE, DQS_EDGE_DETECT_ENABLE parameter cannot be set to TRUE. *(1)*
DQS_USE_DEDICATED_ DELAYCTRLIN	String	No	Specifies whether the DLL directly feeds the DQS delay buffer control signals. The values are TRUE or FALSE. If omitted, default is FALSE. If the DLL_DELAYCTRL_MODE parameter is set to NONE, the DQS_USE_DEDICATED_DELAYCTRLIN parameter cannot be set to TRUE. If the DQS_CTRL_LATCHES_ENABLE parameter is set to TRUE, the DQS_USE_DEDICATED_DELAYCTRLIN parameter cannot be set to FALSE. *(1)*
DQSN_MODE	String	No	Specifies whether to use the dqsb_padio port. The values are NONE, INPUT, OUTPUT, or BIDIR. If omitted, default is NONE. If the DQS_CTRL_LATCHES_ENABLE parameter is set to TRUE, the DQS_USE_DEDICATED_DELAYCTRLIN parameter cannot be set to FALSE. *(1)*
GATED_DQS	String	No	Specifies whether to AND DQS output with a register clocked by delayed DQS signal. The values are TRUE or FALSE. If omitted, default is FALSE. The GATED_DQS parameter can only be used when the DQSN_MODE parameter is set to NONE or OUTPUT. *(2)*
DELAY_CHAIN_MODE	String	No	Specifies delay chain mode. *(1)* There are two modes: ■ DLL Feedback Loop Counter used to control the DQS/nDQS delay chains ■ No DLL used
DQS_DELAY_CHAIN_ SETTING	Integer	No	Specifies value of delay chain. The legal values range from 0 through 63. The DQS_DELAY_CHAIN_SETTING parameter is ignored if the HAS_DQS_DELAY_REQUIREMENT parameter is specified to TRUE. *(3)*
DQS_DELAY_REQUIREMENT	String	No	Specifies delay requirement value. This parameter is available only if the HAS_DQS_DELAY_REQUIREMENT parameter is specified to TRUE.
HAS_DQS_DELAY_ REQUIREMENT	String	No	Specifies whether to use a delay requirement. The values are TRUE or FALSE. If omitted, the default value is FALSE.

Table 19. ALTDQS Megafunction Parameters (Part 4 of 4)

Parameter	Type	Required	Description
LPM_HINT	String	No	Allows you to specify Altera-specific parameters in VHDL Design Files (**.vhd**). Default is UNUSED.
LPM_TYPE	String	No	Identifies library of parameterized modules (LPM) entity name in VHDL Design Files.
INTENDED_DEVICE_FAMILY	String	No	This parameter is used for modeling and behavioral simulation purposes. Create the ALTDQS megafunction with the MegaWizard Plug-in Manager to calculate the value for this parameter.

Notes to Table 19:

(1) Available for Stratix II devices only.

(2) Available for Stratix II and Cyclone II devices only.

(3) Available for Cyclone II devices only.

Document Revision History

Table 20 shows the revision history for this document.

Table 20. Document Revision History

Date	Version	Changes
November 2010	3.2	■ Added output ports for the ALTDQ megafunction ■ Added prototype and component declarations
November 2009	3.1	Updated for Quartus II v9.1: ■ Updated "Device Family Support" on page 1. ■ Removed "Resource Utilization and Performance" section. ■ Removed MegaWizard Plug-In Manager figures. ■ Added "MegaWizard Plug-In Manager Page Option and Description for ALTDQS Megafunction" on page 3, "MegaWizard Plug-In Manager Page Option and Description ALTDQ Megafunction" on page 13.
August 2009	3.0	Maintainence release.
July 2008	2.1	■ Updated template; no other changes
August 2006	2.0	■ Updated screen shots ■ Added ModelSim-Altera simulation procedure ■ Minor text edits for the Quartus® II software version 6.0 release
April 2005	1.1	Minor corrections to tables 2-2 and 2-8.
March 2005	1.0	Initial release.