

Test DDR or DDR2 SDRAM Interfaces on Hardware Using the Example Driver

June 2006 ver 1.2

Application Note 380

Introduction

This application note describes how to test DDR or DDR2 SDRAM interfaces on Altera® development boards using the Altera DDR or DDR2 SDRAM Controller MegaCore® function-generated example driver. The example driver—a stand-alone synthesizable circuit—demonstrates the DDR or DDR2 SDRAM interface. You can use these instructions to quickly build a DDR or DDR2 SDRAM interface on one of the Altera boards and see it working; or use the same principles to establish whether the DDR or DDR2 SDRAM interface on your board is working as expected, independently of any other circuit.

This application note describes a DDR2 SDRAM Controller example driver, but is applicable to the Altera DDR SDRAM Controller.

Figure 1 shows the example system block diagram.



Figure 1. Example System Block Diagram

This application note details the following topics that help you build a stand-alone synthesizable circuit that demonstrates the DDR2 SDRAM interface:

- "Overview" on page 2
- "Set Up the Quartus II Project" on page 3
- "Generate a DDR2 SDRAM Controller MegaCore Function" on page 5
- "Edit the PLL" on page 14
- "Compile the Design" on page 17
- "Select the Board Pin Outs" on page 16
- "Set Up the SignalTap II Logic Analyzer" on page 18
- "Program the Device" on page 23

Overview

A PC running the Quartus[®] II software downloads the device programming file and monitors the activity on the DDR2 SDRAM Controller local interface. The Quartus II SignalTap[®] II utility captures the activity on the DDR2 SDRAM Controller local interface via the JTAG connector.

The driver is a self-checking test generator for the DDR2 SDRAM controller. The driver uses a state machine to write data patterns to a range of column addresses, within a range of row addresses in all memory banks. The driver then reads back the data from the same locations, and checks that the data matches. The pnf (pass not fail) output transitions low if any read data fails the comparison. There is also a pnf_per_byte output, which shows the comparison on a per byte basis. The test_complete output transitions high for a clock cycle at the end of the write then read sequence. After this transition the test restarts from the beginning and repeats indefinitely.



For more information on pnf_per_byte, refer to "Appendix A: Interpret the pnf_per_byte Output" on page 24.

The data patterns are generated with an 8-bit linear feedback shift register (LFSR) per byte—each LFSR has a different initialization seed.

The application note requires the following hardware and software:

- CycloneTM II PCI Development Board, available in the PCI High-Speed Development Kit, Cyclone II Edition
- DDR2 SDRAM Controller MegaCore function
- Quartus II software

The principles in this application note are the same for any Altera development board.

Set Up the Quartus II Project

To set up the Quartus II project, follow these steps:

- 1. Follow the instructions in the *PCI High-Speed Development Kit*, *Cyclone II Edition, Getting Started User Guide* to correctly install your Cyclone II PCI Development Board.
- 2. Start the Quartus II software and create a new project by choosing **New Project Wizard** (File menu).
- On page 3 of 5 of the New Project Wizard in the Family drop-down box choose Cyclone II. In the Available Devices list choose EP2C35F672C6.
- 4. Click Finish.

••••

For more information, see the DDR & DDR2 SDRAM Controller Compiler User Guide.

Fitter Effort

You must ensure the Quartus II Fitter Effort is set to standard, for the best timing placements—timing placements are essential for a DDR2 SDRAM interface.

To set the fitter effort, choose Settings > Fitter Settings > Fitter effort = Standard Fit (highest effort) (Assignments menu), see Figure 2.

Figure 2. Fitter Settings

Category:	
Category: General - Files User Libraries (Current Project) - Device Timing Analysis Settings - Design Entry/Synthesis - Simulation - Timing Analysis - Formal Verification - Physical Synthesis - Board-Level - Compilation Process Settings	Fitter Settings Specify options for fitting. Timing-driven compilation Image: Optimize hold timing: Image: Optimize hold timage: Optimage: Optimize hold timing:
Early I iming Estimate Incremental Compilation Timing Analysis Processing Analysis & Synthesis Settings VHDL Input Verilog HDL Input Default Parameters Synthesis Netlist Optimizations Fitter Settings Physical Synthesis Optimizations Assembler Design Assistant Sinnal Tan III onic Analyzer	Fast Fit (up to 50% faster compilation / may reduce fmax) Auto Fit (reduce Fitter effort after meeting timing requirements) Desired worst case slack (margin): Imit to one fitting attempt Seed: More Settings
Signal ap it Boge Analyzer Logic Analyzer Interface Signal Probe Settings Simulator Settings Simulation Power PowerPlay Power Analyzer Settings	Description: Controls the fitter's trade-off between performance and compilation speed. Auto Fit adjusts the fitter optimization effort to minimize compilation time, while still achieving the design timing requirements. The FITTER_AUTO_EFFORT_DESIRED_SLACK_MARGIN option can be used to request that Auto Fit apply sufficient optimization effort to achieve additional timing margin.

Unused Pins

You must ensure other unused pins on the device are tri-stated inputs (because the unused pins still attach to various devices on the development board), by following these steps: Choose Assignments > Settings > Device > Device & Pin Options > Unused Pins, and for Reserve all unused pins select As inputs, tri-stated (see Figure 3). Click OK, and click OK.

Figure 3. Device & Pin Options



Generate a DDR2 SDRAM Controller MegaCore Function

To generate a DDR2 SDRAM Controller MegaCore Function, follow these steps:

- 1. Choose **MegaWizard Plug-in Manager** (Tools menu), select **Create a new custom megafunction variation** and click **Next**.
- 2. In the Device drop-down box choose **Cyclone II**. For the output file type select **VHDL** or **Verilog HDL**, and enter a name, for example, **test**.

The *<variation name>* must be a different name from the project name and the top-level design entity name.

3. Choose DDR2 SDRAM Controller *<version>* in the Interfaces *>* Memory Controllers directory.

4. Click **Next** (see Figure 4).

Figure 4. Select the Megafunction



Parameterize the DDR2 SDRAM Controller

To parameterize the DDR2 SDRAM Controller, follow these steps:

1. Click Step 1: Parameterize.

2. In the **Presets** list, choose **Micron MT47H16M16BG-5E** (see Figure 5), which selects the correct settings on each tab for this device.

Figure 5. Choose Memory Device

😵 Para	meterize - I	DDR2 SDRAM (Controlle	90							
		Presets	Micron	MT47H1	I6M16BG	-5E		Clock Speed:		MHz (6000 ps)	
			M	cron 256	iMbit ×16	DDR2-40) chip	Device:	EP2C35 F672 C6		
Memory	Controller	Controller Timings	Memory	Timings	Board T	imings Pr	roject Settings	Manual Timings			
Memor	ry Interface —										
Data	bus width:			16	~	Local widt	h = 32				
Numl	ber of chip sele	ects:		1	*						
Numl	ber of chip sele	ects per DIMM:		1	*						
Numl	ber of clock pa	irs from FPGA to r	nemory:	1	*						
Memor	ry Properties -										
Mem	ory size:	32	MB								
Row	address bits:	13	*		Register	red DIMM					
Colu	mn address bit	:s: 9	~	۲	Unbuffe	red memo	ry				
Bank	address bits:	2	*								
Prec	harge address	bit: 10	*								
DQE	oits per DQS pir	n: o	s that inclu	ide DM r	nins						
			o criac inter	300 DHTp	/// 1.2						

- 3. Click the **Controller Settings** tab.
- 4. Turn on **Insert extra pipeline registers on address and command outputs** (see Figure 6), which inserts an extra pipeline stage between the DDR2 SDRAM Controller and the input-output element (IOE) register to improve f_{MAX}. Do not change any other settings.

Figure 6. Controller Settings

Parameterize - DDR2 SDRAM Controller	
Presets: Micron MT47H16M168G-5E Micron 256Mbit x16 DDR2	Clock Speed: MHz (6000 ps) 2-400 chip Device: EP2C35 F672 C6
Memory Controller Controller Timings Memory Timings Board Timings	Project Settings Manual Timings
Cocal Interface	Clocking Options Use DQS for read capture Use non-migratable DQ, DQS, and DM pins
Memory Initialization Options	Use fed-back clock
ODT Setting:DisabledOhmCAS latency:3.0Burst length:4Burst type:• SequentialInterleavedDrive strength:• NormalReducedImage: Memory device DLL enable	Memory Controller Options Insert pipeline registers on address and command outputs Insert extra pipeline registers in the datapath Clock address/command output registers on the negative edge User controlled refresh
■ DLL Reference Clock Options ■ Insert logic to allow the DLL to update only during the memory refr	esh period

- 5. Click the **Board Timings** tab.
- 6. For the **FPGA Clock output to memory chip clock input, nominal delay**, enter **700ps**; for the **Memory DQ/DQS outputs to FPGA inputs, nominal delay**, enter **700ps** (see Figure 7 on page 9).
- These settings are for the Cyclone II PCI Development Board. For other Altera board settings, see "Appendix B. Useful Development Board Information" on page 26.

Figure 7. Board Timings

@ P	ara	meterize	- D	DR2	SDR	AM C	Cont	trolle	r												X
					Pr	esets	:	Micron Mic	MT47H tron 25	16M160 6Mbit ×	3G-5E 16 DDR2	-400 chip		C	lock Speed: Device:	EP2	C35 F6	72 C6	MHz (6000	ps)	
Mem	iory	Controller	Co	ntrol	ler Tin	nings	Mei	mory ⁻	Fimings	Board	l Timings	Project	Settings	Manua	al Timings						
Pi	n Lo Dir Pir Pir	ading Manual pin k n loading on n loading on	FPG	contr A DQ	ol)/DQ5 dress/	pins: comm	hand	pins:	4		pF pF										
	pard	Trace Delay	ys	A CIO	ck pin	s:					_ p-										
	FPGA	A clock outp	ut to	mer	nory d	hip clo	ock ir	nput, r	nominal	delay:	700	1	ps								
	Mem	ory DQ/DQS	5 out	puts:	to FP	GA inp	puts,	, nomir	nal dela	y:	700	1	ps								
1	Fed-	back clock ti	race	, nom	ninal d	elay:					200	0	ps								
	Toler	ance of nor	minal	boar	rd dela	ays +,	-				5		%								
	Wors	st trace skev	w be	twee	n DQj	DQS/	DM ir	n any	one dat	a grou	p: 20		ps								

7. Click **Finish** on the Parameterize - DDR SDRAM Controller window.

Choose DQS Group Placement for the DDR2 SDRAM Controller

To choose DQS group placement for the DDR2 SDRAM Controller, follow these steps:

- 1. Click Step 2: Constraints.
- 2. Set 2T = 0 and 4T = 1 (settings for the Cyclone II PCI Development board). See Figure 8.

Figure 8. Constraints

Constraints - DDR2 SDRAM Controller	
Cselected Device	
Family: Cyclone II	
Device: EP2C35	
Package: F672	
Speed Grade: C6	
1T 🔜 🗸 3T 💙 5T 💙 4T 1 💌 2T 0 💙 OT 🛚	•
2L Y Top	2R 💌
0L Y	0R 💌
11 💌	1R 💌
3L V Bottom	3R 💌
18 💙 38 💙 58 💙 48 💙 28 💙 08 🗙	·
	ОК

- Assignments made here must match your board layout—which is design dependant. The IP Toolbench-generated constraints set up the pin assignments, LogicLockTM regions, IO standards, and other constraints. Therefore, these groupings must match the pin out on the board.
- For the settings for other Altera boards, run the appropriate reference board's constraints Tcl file, which is in the \lib directory of the MegaCore function. See "Select the Board Pin Outs" on page 16.
- 3. Click **OK** on the Constraints DDR2 SDRAM Controller window.

Set Up Simulation

To set up simulation, follow these steps:

This application note explains the steps to simulate a Verilog HDL design. Follow the same steps to simulate a VHDL design.

To generate an IP functional simulation model for your MegaCore function, follow these steps:

1. Click **Step 3: Set Up Simulation** in IP Toolbench (see Figure 9).



Figure 9. IP Toolbench—Set Up Simulation

2. Turn on Generate Simulation Model (see Figure 10).

Figure 10. Generate Simulation Model



- 3. Choose the language in the Language list.
 - To use the IP Toolbench-generated testbench, choose the same language that you chose for your variation.
- 4. Click OK.

For instructions and tips on functional simulation, see "Appendix C. Perform Functional Simulation" on page 27.

Generate the DDR2 SDRAM Controller

To generate the DDR2 SDRAM Controller, follow these steps:

- 1. Click Step 4: Generate on the DDR2 SDRAM Controller window.
- 2. When the **MegaCore Function Generation Successful** message appears, click **Exit** on the Generation DDR2 SDRAM Controller window.

Increase the Example Driver Address Range

Figure 11 shows a system-level diagram including the example instance that the DDR2 SDRAM Controller MegaCore function creates for you.





To test more of the memory, you can increase the example driver address range, if not go to "Edit the PLL" on page 14. To increase the address range of the example driver, follow these steps:

- In the Quartus II software, choose Open (File menu) and choose <variation name>_driver.vhd or .v, in this example choose test_example_driver.vhd or .v.
- 2. Search for the following line in VHDL:

```
MAX_ROW <= std_logic_vector'("000000000011");</pre>
```

For Verilog HDL, search for the following line:

Assign $MAX_ROW = 3;$

3. Change the line to the following VHDL code:

```
MAX_ROW <= to_stdlogicvector("000000000001" sll
(memory row bits -1));
```

For Verilog HDL, change the line to the following code:

Assign MAX_ROW = 1<<(memory row bits - 1);

Replace *memory row bits* with your value for the memory row bits on the Memory Settings tab.

4. Search for the following line in VHDL:

MAX_COL <= std_logic_vector'("0000010000");</pre>

For Verilog HDL, search for the following line:

Assign MAX_COL = 16;

5. Change the line to the following VHDL code:

MAX_COL <= to_stdlogicvector("0000000001" sll (memory column bits -1));

For Verilog HDL, change the line to the following code:

Assign MAX_COL = 1<<(memory column bits - 1);

Replace *memory column bits* with your value for the memory column bits on the Memory Settings tab.

Edit the PLL

The IP Toolbench-generated PLL has an input to output clock ratio of 1:1 and a clock frequency that you entered in IP Toolbench. However, the Cyclone II PCI Development Board uses a 100-MHz input clock. To update the PLL for the design, follow these steps:

- 1. Choose **MegaWizard Plug-in Manager** (Tools menu), select **Edit an existing custom megafunction variation** and click **Next**.
- 2. Choose ddr_pll_cycloneii.vhd and click Next (see Figure 12).

	m megaruncu	on variation file (do you wish to e	edit?	
Look in:	🔄 DDR_Bo	ard_test	•	<u></u>	
🔲 db	1.v				
ddr_pl_	ycloneii.v				
testbend	h				
	and a set of				
File name:	aar_pii_cy	ycionelii. V		_	
Files of type:	All megafu	unction files (*.td	f, *.vhd, *.v) 💌		
	🔽 Show	only wizard-gene	erated files		
Return to	this page for	another edit ope	ration		
The current r	negafunction	variation is base	d on the megafi	unction showr	below. If
you want to u	ise a different	megafunction a	s the basis for y	our changes t	o this
Manafan K			in the second se		
Megalunctio	n name:	ALTPLE			

Figure 12. Choose ddr_pll_cycloneii

3. On ALTPLL [page 3 of 10], in What is the frequency of the inclock0 input? enter 100 MHz (see Figure 13). Click Next four times.

This setting is design dependant, and may be different in your design.

Figure 13. Set the PLL Frequency

MegaWizard Plug-In Manager [page 3 of 10]	
ALTPLL Version 6.0	About Documentation
Parameter Image Output Image Image Page Settings Clocks Library Page Page<	2004 _ Economicador
<u>General/Modes</u>] > Scan/Lock > Clock switchover > ddr_pll_cycloneii	Able to implement the requested PLL
inclk0 frequency: 100.000 MHz c0.	General Which device family will you be using?
Clk Ratio Ph (dg) DC (%) CZ, c0 1666607/1000000 0.00 50.00 c1 1666607/1000000 60.00 c2 166667/1000000 60.00 c2 166667/1000000 60.00 c2 166667/1000000 60.00 c2 166667/1000000 60.00 c3 c4	Which device speed grade will you be using? What is the frequency of the inclock0 input? 100.00 MHz
Cyclone II	Set up PLL in LVDS mode Deta rate: Not Available Mbps
	PLL type Which PLL type will you be using? O Fast PLL O Enthanced PLL O Enthanced PLL Select the PLL type automatically.
	Operation mode How will the PLL outputs be generated? Use the feedback path inside the PLL
	In Normal Mode In Source-Syndronous Compensation Mode In Zero Delay Buffer Mode With no compensation
	Create an 'fbm' input for an external feedback (External Feedback Mode) Which output clock will be compensated for?
	Cancel < Back Bext > Einish

 On ALTPLL [page 7 of 10], for C0 select Enter output clock Frequency and in Requested Settings enter 166.7MHz (see Figure 14). This setting should match your memory clock speed. Ignore the Cannot Implement the Requested PLL error message and click Next.

Figure 14. Set Clock Speeds for Each PLL Output

MegaWizard Plug-In Manager [page 6 of 10]		X
ALTPLL Version 6.0	_	About Documentation
Parameter ZOutput Settings Clocks Summary Library Page CLK0 CLK1 CLK2		
ddr_pll_cycloneii	c0 - Core/External Output Clock Able to implement the requested PLL	
Incili0 Incili0 (neguency: 100.000 MHz 00.000 MHz 00.011 00	Use this dock Clock Tap Settings Enter output clock frequency: Enter output clock parameters: Clock multiplication factor Clock division factor Clock duty cycle (%) More Details >>	Requested settings Actual settings 166.6667000 MHz 166.66667 1 1 0.00 deg 0.00 50.00 50.00
		Per Clock Feasibility Indicators C0 C1 C2

- 5. Repeat for C1, C2. For C1 clock, set the **Clock phase shift** to –90 degrees; for the C2 clock, –135 degrees.
- 6. Click Finish.
- In IP Toolbench, if you want to regenerate your design, turn off **Automatically generate the PLL**, so IP Toolbench does not overwrite the changes that you made to the PLL.

Select the Board Pin Outs

To select the appropriate pin out for the Cyclone II PCI Development Board, follow these steps:

1. Choose **Tcl Scripts** (Tools menu).

 Choose cycloneii_pci_rev_a_pins in the c:/MegaCore/ddr_ddr2_sdram-<version>/lib/ directory and click Run (see Figure 15).

Figure 15. Pin Out Tcl Script



There is one file for each supported Altera memory development board. For your own board design, manually create one of these files using on of the files as a guide or use the Quartus II Assignment Editor to assign your pins.

When the script is complete, the following message displays:

```
Info: Successfully loaded and ran Tcl Script File
"C:\MegaCore\ddr_ddr2_sdram-
<version>\lib\cycloneii_pci_rev_a_pins.tcl"
```

Compile the Design

Before the Quartus II software compiles the design, it runs the IP Toolbench-generated Tcl constraints script, **auto_add_constraints.tcl**.

 Choose Start Compilation (Processing menu), which runs the add constraints scripts, compiles the design, and performs timing analysis.



For more information on the constraints script and timing analysis, see the DDR & DDR2 SDRAM Controller Compiler User Guide.

When the compilation is complete, the Quartus II processing messages tab displays the post-compilation timing analysis results. The results are also written to the *<variation name>_post_summary.txt* file in your project directory.

The results show how much slack you have for each of the various timing requirements—negative slack means that you are not meeting timing.

If the verify timing script reports that your design meets timing, you have successfully generated and implemented your DDR SDRAM Controller.

Set Up the SignalTap II Logic Analyzer

To set up your SignalTap II settings to observe your design working on your board, follow these steps:

1. Choose SignalTap II Logic Analyzer (Tools menu).

2. In the Signal Configuration window, click the ... button (see Figure 16).

Figure 16. Signal Configuration Window

gnal Configuration:	
Clock:	
Data:	
Sample depth: Nodes allocated:	
128 💌 💿 Auto 🔿 Manual: 🛛 🚍	
RAM type:	
М4К	
Buffer acquisition mode:	
Circular: Fre trigger position	
O Segmented: 128 1 bit segments	
Trigger	
Trigger levels: Nodes allocated:	
1 • Auto C Manual:	
Trigger in:	
Source:	
Pattern: Don't Care	
Trigger out:	
Target:	
Lough Active High	
Latency delay:	

3. In the Named box enter *clk* and click List (see Figure 17).

amed: *clk*	Filter:	ignalTa	ap II: pre-sy	nthesis 💌	Customize	List	0K.
ook in: Iddr				•	Include subentities	Stop 🔨	Cancel
odes Found:				Selected Nodes	:		
Name	Assignments			Name		Assignments T	
<pre>Description ====================================</pre>	Unassigned						
stratix_dqs_ref_clk	Unassigned						
test_auk_ddr_dll:dll clk	Unassigned	(
altddio_out:dqs_ref_clk_out_plach	Unassigned	(
■ altddio_out:dqs_ref_clk_out_plaset	Unassigned	(
altddio_out:dqs_ref_clk_out_pld	Unassigned	(
altddio_out:dqs_ref_clk_out_pld	Unassigned	(
Paltddio_out:dqs_ref_clk_out_pld	Unassigned	(
▶ altddio_out:dqs_ref_clk_out_p d	Unassigned	(>>				
altddio_out:dqs_ref_clk_out_pld	Unassigned	(
altddio_out:dqs_ref_clk_out_pld	Unassigned	(<				
altddio_out:dqs_ref_clk_out_plmux	Unassigned	(
altddio_out:dqs_ref_clk_out_ploe	Unassigned	(
altddio_out:dqs_ref_clk_out_plo	Unassigned	(
	Unassigned	(
▶ altddio_out:dqs_ref_clk_out_p o	Unassigned	(
test_example_driver:driver <u> clk</u>	Unassigned						
ddr_pll_stratix:g_stratixpll_ddr_pll	Unassigned	(
🖻 test test ddr. sdramicik	Unassigned	- (-					

- Choose test_example_driver:driver|clk in the Nodes Found list and click > to add to the Selected Nodes list.
- 5. Click OK.
- 6. In the Signal Configuration window, choose the following settings:
 - In the **Sample depth** box choose **512**
 - In the **RAM type** box choose **M-RAM**
 - In Buffer acquisition mode select Circular: Center trigger position
- 7. Choose Add nodes (Edit menu).
- Do not add any DDR SDRAM interface signals (DQ or DQS), because the additional logic, which the SignalTap II logic analyzer adds, adversely affects your timings.

8. In the Named box enter *local* and click List (see Figure 18).

Figure 18. Add SignalTap II Nodes

de Finder						
amed: *local*	Filter: SignalTa	ap II: pre-sy	nthesis 💌	Customize	List	0K.
ook in: IS_PCI_PRO_DDR_Build			▼	Include subentities	Stop Start nod	e search incel
odes Found:			Selected Nodes:			
Name	Assignments 🔺		Name		Assignments T	
example_driver:driverllocal_bank_a	Unassigned					
example_driver:driverllocal_bank_a	Unassigned					
example_driver:driverllocal_bank_a	Unassigned					
<pre>example_driver:driverllocal_be</pre>	Unassigned					
example_driver:driverllocal_be[0]	Unassigned					
example_driver:driver local_be[1]	Unassigned					
example_driver:driver local_be[2]	Unassigned					
example_driver:driver[local_be[3]	Unassigned					
example_driver:driver local_be[4]	Unassigned	>>				
example_driver:driverllocal_be[5]	Unassigned					
example_driver:driver local_be[6]	Unassigned	<				
example_driver:driver local_be[7]	Unassigned	11				
example_driver:driver local_be[8]	Unassigned					
example_driver:driver local_be[9]	Unassigned					
example_driver:driver local_be[10]	Unassigned					
example_driver:driver local_be[11]	Unassigned					
example_driver:driver local_be[12]	Unassigned					
example_driver:driver local_be[13]	Unassigned					
example_driver:driverllocal_be[14]	Unassigned 🗾					
(► E		•		Þ	

- 9. Choose the following signals in the **Nodes Found** list and click > to add to the **Selected Nodes** list:
 - example_driver:driver|local_rdata
 - example_driver:driver|local_rdata_valid
 - example_driver:driver|local_read_req
 - example_driver:driver|local_wdata
 - example_driver:driver|local_wdata_req
 - example_driver:driver|local_write_req
- 10. In the Named box enter *pnf * and click List.
- Choose the following signals in the Nodes Found list and click > to add to the Selected Nodes list:
 - pnf
 - pnf_per_byte
- 12. In the Named box enter *test_complete* and click List.

- Choose the test_complete signal in the Nodes Found list and click > to add to the Selected Nodes list.
- 14. Click **OK** in the Node Finder window.
- 15. To reduce the SignalTap logic size, turn off **Trigger Enable** on the following signals (see Figure 19):
 - example_driver:driver|local_rdata
 - example_driver:driver|local_wdata
 - pnf_per_byte

Figure 19. Trigger Enable

auto_signattap_0			Lock mode:	.ock mode: 📄 Allow all changes 💽					
		Node	Incremental	Debug Port	Data Enable	Trigger Enable	Trigger Levels		
Туре	Alias	Name	Route	Out	278/Auto	6/Auto	1 🔽 Basic 💽		
		iver:driver local_rdata	Γ						
\odot		er:driver local_rdata_valid	Γ	-83	V	J.			
\odot		iver:driver local_read_req		-83	V	N			
٥		er:driver local_wdata			N				
۲		iver:driver local_write_req		-453	V	N.			
\odot		dr_sdram local_write_req		-83	N	N			
•		pnf		-83	V	ম			
\odot					v				
\odot		test_complete		-83	V	N			
🔊 Da	ata 🔀	Setup							

16. Right click on the test_complete Trigger Levels cell and set to trigger on a **Rising Edge** (see Figure 20).

Type Alias Name Route Out 278/Auto 6/Auto 1/r Basic Image: Solution of the state of the st	278/Auto 6/Auto 1 IV Basic I IV IV IV IV IV IV IV IV IV IV IV			Node	Incremental	Debug Port	Data Enable	Trigger Enable	Trigger Levels	
Image: System Image: S	マロ マロ マロ 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 <	Туре	Alias	Name	Route	Out	278/Auto	6/Auto	1 🔽 Basic 🖃	Ī
Image: construction of the state o	전 지 지 200 지 지 지	٥		iver:driver local_rdata		-67	ম			1
Image: system Image: s	지 지 지 지 지 지 지 표 지 지 지 표 지 지 지 표 지 지 지 표 지 지 지 표 지 지 지 표 지 지 지 표 지 지 지 표 지 지 지	•		er:driver local_rdata_valid		-858	N	N		1
Image:		\odot		iver:driver local_read_req		-858	N	N		1
with the second seco	照 지 지 照 지 지 지 照 지 지 지 지 지 지 지 지 지 지 지 지 지 지 지	\odot		er:driver local_wdata	Γ		N			1
🕡dr_sdram local_write_req 🔲 🚳 🔽 🔽	照 지 지 照 지 지 지 지 지 지 지 Month Care South Care South Care South Care	۲		iver:driver local_write_req	Γ		v	<u> </u>		1
				dr_sdram local_write_req			N	N N		1
💿 ph 🗌 🖓 🖓 🕅	Image: Constraint of the second se	\odot		pnf			T	N		1
Image: Book of the second	Don't Care	\odot					v			
💿 🚺 test_complete	💹 Don't Care			test_complete		-458			ک	
1										Rising Edge
\sim	C Pailing Edg									High Either Edge
	✓ Failing Edge ✓ Rising Edge 1 High X Either Edge	N D.	da 🗖							

Figure 20. Rising Edge

- 17. Choose **Save** (File menu), and choose **Yes** to the prompt **Do you** want to enable SignalTap II File stp1.stp for the current project?
- 18. Re-compile the design to add the SignalTap II probes, by choosing **Start Compilation** (Processing menu).
- When compilation is complete, connect your download cable (for example, ByteBlaster[™] II download cable) to the JTAG port on the development board.
- 20. In the SignalTap II logic analyzer in the JTAG Chain Configuration window:
 - In the Hardware list, choose ByteBlasterII [LPT1]
 - In the Device list, choose **EP2C35**
 - In the SOF Manager list, choose <project name>.sof

Program the Device

To program the device, follow these steps:

1. Click the **Program Device** icon that is next to SOF Manager (see Figure 21).

Figure 21. Program Device

Hardware:	ByteBlasterMV [LPT1]	Setup
Device:	@2: EP1S25/_HARDCOPY_FPGA_PROTO	Scan Chain
>> SOF	Manager: 📥 🛛 test_platform.sof	

2. Click **Run Analysis** to run once; click **Autorun Analysis** to run continuously. See Figure 22.

Figure 22. Analysis

France Allow	News	64	40	20	16	0	10	27	40	E4	00
Type Allas	Name	-04	-40		-10	Y			40	04	
<ri></ri>	sdram local_read_req										
•	river local_rdata_valid										
Image: A start and a start	⊞sdram local_rdata	Daaaaaa				00000000		02308	i654h		
6	dram local_wdata		20000000	20000000	20000000		0000000	CC72AD13h		20000000	
٠	dram local_write_req										
•	river local_wdata_req					-					
•	pnf										
6	⊞ pnf per bγte						Fh				
•	test_complete					<u> </u>					
<											>
N Data	8 Setun	_									_
	- oomp										
				\sim							

Appendix A: Interpret the pnf_per_byte Output

Figure 23 shows an example of how to interpret the pnf_per_byte output. This example uses a 24-bit wide data bus—three DQS pins and six pnf_per_byte signals. The numbers on the rising and falling edges of the DQS signals represent the pnf_per_byte[5:0] bus. For example, if pnf_per_byte[3] is zero and all other pnf_per_byte outputs are high, there is an error on the data clocked by the DQS[0] falling edge.

Figure 23. Interpret the pnf_per_byte Output



Figure 24 shows example local_rdata, local_wdata, and pnf_per_byte signals for this example.

The pnf_per_byte output is three cycles after local_rdata.

Figure 24. Example Signals

local_rdata_valid												
local_rdata	Error	В	С	D	E	F	G		Н		('	J
local_wdata	A	В	С	D	E	F	G	н		Т		(J
pnf_per_byte				ЗВ	ЗF	ЗF	ЗF	ЗF	ЗF	ЗF	3F	

Table 1 shows generally how to interpret any errors on the pnf_per_byte signal.

Table 1. Interpret pnf_per_byte Errors						
Error in Position	Interpretation					
A	If the postamble logic is too late, you can miss capturing A, by not enabling on time.					
Н	If the postamble logic is too early, you can miss capturing F; by disabling too soon, you see E twice.					

Appendix B. Useful Development Board Information

Table 2 shows a summary of the board level design information for various Altera IP evaluation boards.

Table 2. Altera Evaluation Boards Note (1)											
Board	SDRAM	f _{MAX} (MHz)	FPGA-to-Memory Delay (ps)	Memory-to-FPGA Delay (ps)							
Stratix Memory Demonstration Board 1	DDR	200	1,400	1,400							
Stratix PCI Development Board	DDR	200	1,200	1,200							
Stratix PCI High-Speed Development Board	DDR	167	1,200	1,200							
Stratix GX Video Development Board	DDR	_	1,000	1,000							
Stratix II Memory Demonstration Board 1	DDR	200	1,400	1,400							
Stratix II Memory Demonstration Board 2	DDR2	267	1,400	1,400							
Nios II Development Board, Cyclone II Edition	DDR	167	550	550							
Cyclone™ DDR Memory Board	DDR	133	500	500							
Twister DDR-SDRAM Evaluation Kit	DDR	133	500	500							
Cyclone II DSP Development Board	DDR2	167	1,400	1,400							
Cyclone II PCI Development Board	DDR2	167	700	700							

Note to Table 2:

 The figures are for operation at room temperature and have not been verified over the full process, voltage, temperature (PVT) range.

Appendix C. Perform Functional Simulation

The DDR2 SDRAM Controller generates all the controller files with the testbench. The testbench files are located in the *<project name*>\testbench directory.

The Verilog HDL testbench (*<variation name>_sim_tb.v*) requires the modifications to match to your particular memory model. To simulate the design, follow these steps:

- Download the simulation model of the memory type that you selected in the DDR2 SDRAM Controller - Parametrize window into the *<project name>*\testbench directory.
- 2. Copy any parameter file (if separate from the model) in to the *<project name>*\testbench\modelsim directory.
- 3. Open <variation name>_sim_tb.v in a text editor.
- 4. Locate the line generic_ddr_sdram_rtl memory_0_0.
- 5. Replace generic_ddr_sdram_rtl with the <modelname>.
- 6. Ensure all signal names match those used in your model.
- 7. Repeat for all memory instances in the testbench.
 - The automatic testbench generation assumes each memory model is a ×8 device—has one DQS per DQ group and each chip is a single ×8 device.

If you have a ×16 device, follow these steps to ensure the testbench DQ, DQS, and DM signals match the model, otherwise go to "Increase the Example Driver Address Range" on page 12.

- 1. Check the if def statements within the parameters file and add the appropriate define statements to the top of the memory model. For example, for Micron models set the speed grade and number of DQ to DQS pins.
- 2. Start the ModelSim simulator and change directory to the **\testbench\modelsim**.
- 3. Type the following commands

set memory_model <model name>
source <variation_name>_ddr_sdram_vsim.tcl

For example to use two Micron MT47H16M16BG -5E ×16 DDR devices to make a 32-bit DDR SDRAM interface, follow these steps:

- 1. In the DDR SDRAM IP Toolbench, select **Micron MT47H16M16BG -5E** in the **Presets:** list.
 - If the memory model is not available, add the model to the **memory_types.dat** file in the *<DDR installation directory>***ddr_ddr2_sdram-v#****constraints** directory.
- 2. Click the Memory tab and select 32 for the Data bus width.
- 3. Click Finish.
- 4. In IP Toolbench, click **Set Up Simulation** and turn on generation for a Verilog HDL simulation model.
- 5. Click Finish.
- 6. In IP Toolbench, click **Generate** to generate the custom MegaCore variation.
- 7. Download the **MT47H16M16BG -5E** Verilog HDL model from the Micron website.
- 8. Extract the **ddr.v** model to the *<project name >*\testbench directory.
- 9. Rename the **ddr_parameters.vh** file to **ddr_parameters.v** and move it to the *<project name>*\testbench\modelsim directory.
- 10. Open *<variation name>_sim_tb.v* in a text editor.
- The IP Toolbench-generated testbench creates 4 × 8 devices. However, the DDR memory model is a ×16, so you require a 2 × 16 memory model instantiation. Locate the following line:

generic_ddr_sdram_rtl memory_0_0 (

12. Replace generic_ddr_sdram_rtl with the model name, for example ddr eg.

ddr memory_0_0 (

- 13. Repeat for memory_0_2.
- 14. Delete the instance memory_0_1 and memory_0_3 as these are not required.

15. Change the width of DQ, DQS, and DM to correctly match the model. For example change the following code:

```
//generic ddr sdram rtl memory 0 0(
                  (mem_dq[8* (0+1) - 1 : 8 * 0]),
11
          .Dq
                  (mem_dqs[0]),
11
          .Dqs
11
          .Addr
                  (a_delayed[11: 0]),
11
          .Ba
                  (ba_delayed),
11
          .Clk
                  (clk_to_ram),
11
          .Clk_n (clk_to_ram_n),
          .Cke
                  (cke delayed[0]),
11
11
                  (cs_n_delayed[0]),
          .Cs n
11
          .Ras_n (ras_n_delayed),
11
          .Cas_n (cas_n_delayed),
11
          .We_n
                  (we_n_delayed),
11
          .Dm
                  (dm_delayed[0])
11
      );
```

To the following code:

```
ddr2 memory_0_0 (
            (mem_dq[15:0]),//Updated
     .Dq
     .Dqs
            (mem_dqs[1:0]), //Updated
     .Addr
            (a_delayed[11: 0]),
    .Ba
            (ba_delayed),
    .Clk
            (clk_to_ram),
     .Clk_n (clk_to_ram_n),
            (cke delayed[0]),
    .Cke
    .Cs_n (cs_n_delayed[0]),
    .Ras_n (ras_n_delayed),
    .Cas n (cas n delayed),
     .We_n (we_n_delayed),
     .Dm
            (dm_delayed[1:0])
                                //Updated
);
```

and:

```
ddr2 memory_0_2 (
   .Dq
          (mem_dq[31:16]),//Updated
     .Dqs
            (mem_dqs[3:2]),//Updated
     .Addr
            (a_delayed[11: 0]),
     .Ba
            (ba_delayed),
     .Clk
            (clk to ram),
     .Clk_n (clk_to_ram_n),
     .Cke
            (cke_delayed[0]),
     .Cs_n (cs_n_delayed[0]),
     .Ras_n (ras_n_delayed),
     .Cas_n (cas_n_delayed),
     .We_n (we_n_delayed),
     .Dm
            (dm_delayed[3:2])//Updated
);
```

16. Save the testbench.

17. Open the DDR model and set the following define statements:

`define sg5E `define x16

These statements ensure the model is behaving as a $\times 16$ device with the correct speed grade.

- Start the ModelSim simulator and change directory to the <project>\testbench\modelsim directory.
- 19. At the command prompt type the following command:
- set memory_model ddr
- On the Tools menu, click Execute Macro and select *<variation* name>_ddr_sdram_vsim.tcl.



101 Innovation Drive San Jose, CA 95134 (408) 544-7000 www.altera.com Applications Hotline: (800) 800-EPLD Literature Services: literature@altera.com Copyright © 2006 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make change es to any products and services at any time without notice. Altera assumes no responsibility or liability

arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

