

Using the DDR2 SDRAM on Intel's DE4 Board with Verilog Designs

For Quartus[®] Prime 18.1

1 Introduction

This tutorial explains how DDR2 memory modules connected to Intel's DE4 Development and Education board can be used with a Nios[®] II system implemented by using the Intel[®] Platform Designer tool. The discussion is based on the assumption that the reader has access to a DE4 board and is familiar with the material in the tutorial *Introduction to the Intel Platform Designer Tool* and the tutorial *Intel FPGA Monitor Program Tutorial for Nios II*.

The screen captures in the tutorial were obtained using the Quartus[®] Prime version 18.1; if other versions of the software are used, some of the images may be slightly different.

Contents:

- Example Nios II System
- The DDR2 SDRAM Interface
- Using the Platform Designer tool to Generate the Nios II System
- Integration of the Nios II System into the Quartus Prime Project
- Using the Clock Crossing Bridge IP Core

2 Background

The introductory tutorial *Introduction to the Intel Platform Designer Tool* explains how the memory in an FPGA chip can be used in the context of a simple Nios II system. For practical applications it is necessary to have a much larger memory. The Intel DE4 board contains two DDR2 SODIMM (Small outline dual inline memory modules) slots that can be used to expand the amount of memory available to the FPGA. To provide access to the DDR2 SODIMMs, the Platform Designer tool implements a *DDR2 SDRAM Controller with UniPHY* circuit that generates the signals needed to interface with DDR2 SODIMMs. The DDR2 standard requires careful timing between the memory modules and the system, so the *DDR2 SDRAM Controller with UniPHY* circuit uses a reference clock signal to produce two clock signals: one for the system and one for the memory module.

3 Example Nios[®] II System

As an illustrative example, we will add the DDR2 SDRAM to the Nios II system described in the *Introduction to the Intel Platform Designer Tool* tutorial. Figure 1 gives the block diagram of our example system.

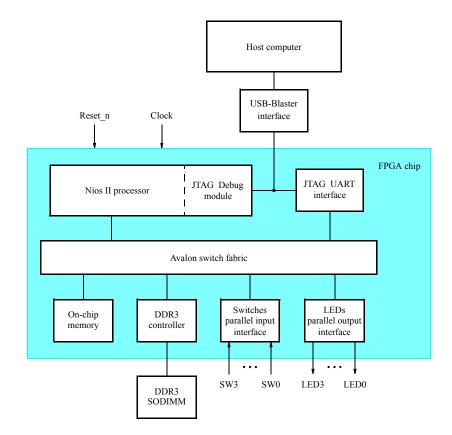


Figure 1. Example Nios II system implemented on the DE4 board.

The system realizes a trivial task. Four toggle switches on the DE4 board, SW3 - 0, are used to turn on or off the four green LEDs, LED3 - 0. The switches are connected to the Nios II system by means of a parallel I/O interface configured to act as an input port. The LEDs are driven by the signals from another parallel I/O interface configured to act as an output port. To achieve the desired operation, the four-bit pattern corresponding to the state of the switches has to be sent to the output port to activate the LEDs. This will be done by having the Nios II processor execute an application program. Continuous operation is required, such that as the switches are toggled the lights change accordingly.

The introductory tutorial showed how we can use the Platform Designer tool to design the hardware needed to implement this task, assuming that the application program which reads the state of the toggle switches and sets the LEDs accordingly is loaded into a memory block in the FPGA chip. In this tutorial, we will explain how DDR2 SODIMMs on the DE4 can be included in the system in Figure 1, so that our application program can be run from the DDR2 SDRAM rather than from the on-chip memory.

Doing this tutorial, the reader will learn about:

- Using the Platform Designer tool to include a DDR2 SDRAM Interface for a Nios II-based system
- Interfacing components clocked by different frequency signals on the DE4 board

4 The DDR2 SDRAM Interface

The signals needed to communicate with the DDR2 SODIMMs are shown in Figure 2. All of the signals can be provided by the DDR2 SDRAM Controller that can be generated by using the Platform Designer tool.

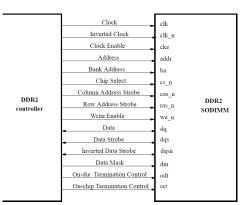


Figure 2. The DDR2 SDRAM signals.

5 Using the Platform Designer tool to Generate the Nios[®] II System

Our starting point will be the Nios II system discussed in the *Introduction to the Intel Platform Designer Tool* tutorial, which we implemented in a project called *lights*. We specified the system shown in Figure 3.

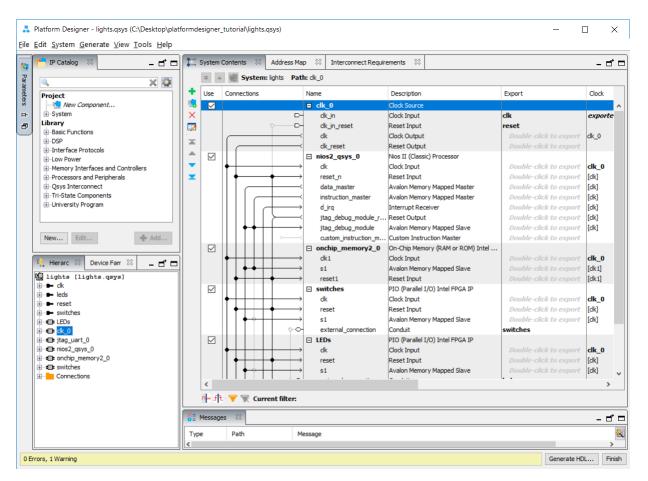


Figure 3. The Nios II system defined in the introductory tutorial.

If you saved the *lights* project, then open this project in the Quartus Prime software and then open the Platform Designer tool. Otherwise, you need to create and implement the project, as explained in the introductory tutorial, to obtain the system shown in the figure.

The DDR2 controller requires you to specify the parameters of your particular DDR2 SODIMM for it to function correctly. A list of necessary parameters are given in Table 1. The tutorial provides the parameters for the D2SH28081XH25AA SODIMM manufactured by DSL Memory Specialties and supplied with the DE4 at the time of writing. If you use a different memory module, you will have search your module's datasheet for the parameters listed in Table 1.

For Quartus[®] Prime 18.1

Memory Parameter	K4B2G0846C-HCK0 Timing Values
Memory device speed	400 MHz
CAS Latency	6 cycles
Row address width	14 bits
Column address width	10 bits
Bank address width	3 bits
Address and control setup to CK clock rise [tIS]	375 ps
Address and control hold after CK clock rise [tIH]	375 ps
Data setup to clock (DQS) rise [tDS]	250 ps
Data hold after clock (DQS) rise [tDH]	250 ps
DQS, DQS to DQ skew, per group, per access [tDQSQ]	200 ps
DQ output hold time from DQS, DQS [tQHS]	300 ps
DQS output access time from CK, CK [tDQSCK]	350 ps
First latching edge of DQS to associated clock edge [tDQSS]	0.25 cycles
DQS Differential High Pulse Width [tDQSH]	0.35 cycles
DQS falling edge hold time from CK [tDSH]	0.2 cycles
DQS falling edge to CK setup time [tDSS]	0.2 cycles
Memory initialization time at power-up [tINIT]	200 us
Load mode register command period [tMRD]	5 cycles
Active to precharge time [tRAS]	40.0 ns
Active to read or write time [tRCD]	15.0 ns
Precharge command period [tRP]	15.0 ns
Refresh command interval [tREFI]	7.8 us
Auto-refresh command interval [tRFC]	127.5 ns
Write recovery time [tWR]	15.0 ns
Write to read period [tWTR]	3 cycles
Four active window time [tFAW]	37.5 ns
RAS to RAS delay time [tRRD]	7.5 ns
Read to precharge time [tRTP]	7.5 ns

Table 1. Parameters for the D2SH28081XH25AA SODIMM supplied with the DE4

To add the DDR2 controller, in the window of Figure 3 select Memory Interfaces and Controllers > Memory Interfaces with UniPHY > DDR2 SDRAM Controller with UniPHY and click Add. A window depicted in Figure 4 appears.

		ntroller w	ith UniPH	IY Intel FPGA IF	,	-	umentatio	
oCoro' altera_mem	n_if_ddr2_emif					Generate I	Example [Design
					^	Presets		
Parameters								
onstraint scripts, an e				d PHY and Controller HDL,		Project	New to	create
Interface Type							EDEC DD	R 2-106
Enable Hard Exter	rnal Memory Interf	ace					EDEC DD	
PHY Settings Memo	vy Parameters M	emory Timina	Board Setting	s Controller Settings D	iagnostics		EDEC DD	R2-1066
		chory mining	oodra betung	s controller becangs D	lagi lostica		EDEC DD	
General Sett	ings						EDEC DD	
Speed Grade:		2	2 ~				EDEC DD	
Generate P	HY only						EDEC DD EDEC DD	
							EDEC DD	
Clocks							EDEC DD	
Memory clock fr	equency:	4	ю.0	MHz		+ 3	EDEC DD	R2-533
Achieved memo	ry clock frequency:	- 4	юо.о	MHz			EDEC DD	R2-533
PLL reference d	ock frequency:		0.0	MHz			EDEC DD	
Rate on Avalon-				198.12			EDEC DD EDEC DD	
			half \checkmark	-			EDEC DD	
Achieved local o	lock frequency:	2	0.00	MHz			EDEC DD	
Enable AFI	half rate clock						EDEC DD	
							EDEC DD	R2-800
Advanced PH	IY Settings						EDEC DD	
Advanced of	lock phase control						ICRON N	
Additional addre	ess and command d	lock phase: 0	.0	Degrees			ICRON N	
I/O standard:			STL-18 🗸				ICRON N	
PLL sharing mod	le:		lo sharing v			• N	ICRON N	MT 18HTF
DLL sharing mod	le:		lo sharing 🗸				ICRON N	
OCT sharing mo			lo sharing 🗸				ILCRON N	
Reconfigurable			-	1			TODONIA	
Recornigurable	FLE LOCAUON:	T	fop_Bottom ∨			<		
					×	Apply	l	Update.
					-			

Figure 4. Add the DDR2 Controller.

Set the Speed Grade to 2, the Memory clock frequency parameter to 400.0 MHz, the PLL reference clock frequency to 50.0 MHz, Rate on Avalon-MM interface to Half, and leave other settings in PHY Settings as default. Click Memory Parameters to show the window in Figure 5

Memory vendor: Hynix Memory format: Unbuffered DIMM Memory device speed grade: 400.0 Memory device speed grade: 400.0 Memory device speed grade: 400.0 Memory device speed grade: 64 DQ/DQS group size: 8 Number of DQS groups: 8 Number of slots: 1 Number of slots: 1 Number of slots: 2 Row address width: 10 Bank-address width: 10 Bank-address width: 3 DQS# Enable MicRoN MT18HT MicRoN MT18HT </th <th></th> <th></th> <th>Do</th> <th>cumentation</th>			Do	cumentation
Enable Hard External Memory Interface Project Apply memory parameters from the manufacturer data sheet Apply device presets from the preset list on the right. Memory vendor: Mynix Memory format: Unbuffered DIMM Memory device speed grade: 400.0 Memory device speed grade: 10 Q/Q/QS groups: 8 Number of clocks: 2 Row address width: 10 Memory Initialization Options MICRON MT18HT			Generate	Example Design.
Enable Hard External Memory Interface Project Apply memory parameters from the manufacturer data sheet Apply device presets from the preset list on the right. Memory vendor: Mynix Memory format: Unbuffered DIMM Memory device speed grade: 400.0 Memory device speed grade: 10 Q/Q/QS groups: 8 Number of clocks: 2 Row address width: 10 Memory Initialization Options MICRON MT18HT				
PHY Settings Memory Parameters Apply memory parameters from the manufacturer data sheet Apply device presets from the preset list on the right. Memory vandor: Hynix Memory format: Unbuffered DIMM Memory device speed grade: 400.0 Memory device speed grade: 1 QC/QCS groups ize: 8 Number of locks: 1 2 Row address width: 10 Bank-address width: 10 Bank-address width: 10 Memory Diss PDCS DCR2-533 Burst Length: 8 Micron MT18HT Mi			 Presets 	
Apply memory parameters from the manufacturer data sheet Apply device presets from the preset list on the right. Memory vendor: Hynix Memory format: Unbuffered DIMM v Memory device speed grade: 400.0 Memory device speed grade: 8 Number of DQS groups: 8 Number of folds: 2 Row address width: 14 Column address width: 10 Bable DM pins DQ DQS # Enable MicRon MT18HT Mic	Enable Hard External Memory Interface			
Apply device presets from the preset list on the right. Memory format: Ubbuffered DIMM ✓ Memory format: Ubbuffered DIMM ✓ Memory device speed grade: 400.0 ♥ MHz Total interface width: 64 DQ/DQS groups ize: 8 Number of DQS groups: 8 Number of oldsts: 1 ✓ Number of oldsts: 2 ✓ Row address width: 10 DQS = Enable Memory Initialization Options MICRON MT18HT Read Burst Type: Sequential ✓ MicRoN MT18HT	PHY Settings Memory Parameters Memory 1	Timing Board Settings Controller Settings Diagnostics		
Memory Vendol. Hymx v Memory format: Unbuffered DIMM v Memory device speed grade: 400.0 v Memory device speed grade: 400.0 v Memory device speed grade: 400.0 v Memory device speed grade: 8 v DQ/DQS group size: 8 v B v 1 v Number of DQS groups: 8 Number of slots: 1 v Number of clocks: 2 v Row address width: 14 10 3 Bank-address width: 3 0 DQS # Enable Enable DM pins v Hencory Initialization Options Address and command parity MocRon MT18HT MicRon MT1				JEDEC DDR2-106
Memory format: Unbuffered DIMM v Memory device speed grade: 400.0 v Memory device speed grade: 400.0 v Memory device speed grade: 400.0 v Memory device speed grade: 8 v Number of Jo25 groups ize: 8 v Number of Jo25 groups: 8 Number of slots: 1 v Number of slots: 1 v Number of slots: 2 v Row address width: 10 Bank-address width: 10 DQS# Enable JEDEC DDR2-533 V Memory Initialization Options JEDEC DDR2-533 V Memory CAS latency setting: 8 v MicRON MT18HT MiCRON MT18HT MicRON MT18HT	Memory vendor:	Hypix		JEDEC DDR2-106
Address width: 64 DQ/DQS group size: 8 ∨ B ∨ JEDEC D0R2-400 JEDEC DR2-533 JEDEC DR2-533 Number of slots: 1 ∨ Number of slots: 1 ∨ Number of clocks: 2 ∨ Row address width: 14 Column address width: 10 Bank-address width: 3 DQS # Enable JEDEC D0R2-400 MCRON MT18HT JEDEC D0R2-533 JEDEC D0R2-553 JEDEC D0R2-563 JEDEC D0R2-567 JEDEC D0R2-667 JEDEC D0R2-667 JEDEC D0R2-667 JEDEC D0R2-667 JEDEC D0R2-667 JEDEC D0R2-800 JEDEC D0R2-800 MICRON MT18HT MICRON MT18HT<	Memory format:		••••	JEDEC DDR2-106
Total interface width: 64 DQ/DQS group size: 8 ✓ Number of DQS groups: 8 ✓ Number of Jolds: 1 ✓ Number of slots: 1 ✓ Number of clocks: 2 ✓ Row address width: 14 Column address width: 10 Bank-address width: 3 JEDEC DDR2-400 JEDEC DDR2-533 Bank-address width: 3 JEDEC DDR2-667 JEDEC DDR2-687 MICRON MT18HT MICRON MT18HT MICRON MT18HT MICRON MT18HT MICRON MT36HT <			•	
Lotal interface Width: 64 DQ/DQS group size: 8 Number of DQS groups: 8 Number of DQS groups: 8 Number of fanks per slot: 1 Image: State s	· · · ·	400.0 V MHz		
DQ/DQS group size: 8 Number of DQS groups: 8 Number of lods: 1 1 - Number of slots: 1 Number of slots: 2 Number of clods: 2 Row address width: 14 Column address width: 10 Bank-address width: 10 JEDEC DDR2-667 JEDEC DDR2-687 JEDEC DDR2-800 JEDEC DDR2-800 JEDEC DDR2-800 JEDEC DDR2-800 JEDEC DDR2-800 JEDEC DDR2-81 MICRON MT 16HT MICRON MT 16HT MICRON MT 16HT MICRON MT 18HT MICRON M	Total interface width:	64		
Number of slots: 1 Number of slots: 1 Number of ranks per slot: 1 1 2 Row address width: 14 10 10 Bank-address width: 10 10 10 Bank-address width: 3 10 10 DQS# Enable 10 DQS# Enable 10 Image: Document of Docks: 10 DQS# Enable 10 Image: Dock address width: 3 10 10 DQS# Enable 10 Image: Dock address width: 3 10 10 DQS# Enable 10 Image: Dock address width: 3 10 10 DQS# Enable 10 Image: Dock address width: 3 10 10 DQS# Enable 10 Image: Dock address width: 3 11 10 DQS# Enable 10 Image: Dock address width: 3 11 10 DDS # Enable 10 Image: Dock address width: 10 Image: Dock address address width: 10 Image: Dock address width: 10 Image: Dock address address address width: 10 Image: Dock address address address width: <	DQ/DQS group size:	8 ~		JEDEC DDR2-400
Number of slots: 1 v Number of slots: 1 v Number of ranks per slot: 1 v 1 v 2 v Row address width: 14 10 10 Bank-address width: 10 10 3 Bank-address width: 3 10 10 DQS# Enable 10 MicRon MT16HT MicRon MT18HT M	Number of DOS groups:	9	•	JEDEC DDR2-533
Number of ranks per slot: 1 Number of clocks: 2 Row address width: 14 10 10 Bank-address width: 10 10 10 Bank-address width: 3 10 10 DQS# Enable 10 DQS# Enable 10 DQS# Enable 10 Memory Initialization Options 10 McRcon MT 16HT MiCRCON MT 16HT MiCRON MT 16HT MiCRON MT 16HT <td< td=""><td></td><td></td><td>• • •</td><td>JEDEC DDR2-533</td></td<>			• • •	JEDEC DDR2-533
Number of docks: 2 Row address width: 14 10 JEDEC DDR2-667 JEDEC DDR2-657 JEDEC DDR2-657 JEDEC DDR2-650 JEDEC DDR2-650			••••	JEDEC DDR2-533
Name of Bods. 2 Row address width: 14 10 10 11 10 11 10 11 10 11 10 11 10 12 10 13 10 14 10 15 10 16 10 17 10 18 10 19 10 10 10 10 10 11 10 11 10 12 10 13 10 14 10 15 10 16 10 17 10 18 10 18 10 19 10 10 10 10 10 11 10 11 10 11 10 12 10 13 10 14 10 14 10 15 10 16 10 17 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 10 18 1	Number of ranks per slot:	1 🗸	•	
Row address width: 14 Column address width: 10 Bank-address width: 3 Bank-address width: 3 Description JEDEC DDR2-800 JEDEC DDR2-800 MICRON MT16HT MICRON MT18HT MICRON MT18HT MICRON MT18HT MICRON MT18HT MICRON MT36HT <td>Number of clocks:</td> <td>2 🗸</td> <td>•</td> <td></td>	Number of clocks:	2 🗸	•	
Column address width: 10 Bank-address width: 3 Bank-address width: 3 Dasset and command parity Memory Initialization Options McRoon MT 16HT McRoon MT 16HT McRoon MT 16HT McRoon MT 16HT Microon MT 18HT Microon MT 18HT </td <td>Row address width:</td> <td>14</td> <td></td> <td></td>	Row address width:	14		
Bank-address width: 3 3 3 Enable DM pins > DQS# Enable > McRON MT1shT > Address and command parity > McRON MT1shT	Column address width:	10	IF	
✓ Enable DM pins ✓ DQS# Enable ✓ Memory Initialization Options ✓ Mack Register 0 Burst Length: 8 ∨ Read Burst Type: Sequential ∨ DLL precharge power down: Fast exit ∨ MicRON MT 18HT MicRON MT 18HT Output drive steringt: 6 ∨ MicRON MT 38HT MicRON MT 38HT MicRON MT 18HT MicRON MT 18HT				JEDEC DDR2-800
✓ DQ\$# Enable → JEDEC DDR2-800 ✓ Memory Initialization Options → MICRON MT 16HT ▲ Address and command parity → MICRON MT 16HT Mode Register 0 → MICRON MT 16HT Burst Length: 8 ∨ Read Burst Type: Sequential ∨ DLL precharge power down: Fast exit ∨ Mode Register 1 → MICRON MT 36HT Output drive strength setting: 0 ∨ Memory additive CAS latency setting: 0 ∨ Memory additive CAS latency setting: 0 ∨ Memory on-die termination (ODT) setting: 50 ∨ Mode Register 2 SRT Enable: 2x refresh rate ∨	Bank-address width:	3	•	JEDEC DDR2-800
✓ DQS# Enable ● MICRON MT 16HT ✓ Memory Initialization Options ● MICRON MT 16HT ● MICRON MT 16HT ● MICRON MT 18HT ● MICRON MT 18HT ● MICRON MT 18HT ● MICRON MT 18HT ● MICRON MT 18HT ● MICRON MT 18HT ● MICRON MT 36HT ● MICRON MT 47H1 ● MICRON MT 47H1 ● MICRON M	Enable DM pins		• • •	JEDEC DDR2-800
✓ Memory Initialization Options ● MICRON MT 16HT ● MICRON MT 16HT ● MICRON MT 18HT ● MICRON MT 18HT ● MICRON MT 36HT ● MICRON MT 36HT ● MICRON MT 36HT ● MICRO	DOS# Epable		••••	
• Memory Initialization Options • MICRON MT I6HT • Address and command parity • MICRON MT 16HT • Marce Register 0 • MICRON MT 18HT Burst Length: 8 ∨ Read Burst Type: Sequential ∨ DLL precharge power down: Fast exit ∨ Memory CAS latency setting: 6 ∨ Mode Register 1 Output drive strength setting: MucRON MT36HT MICRON MT36HT Memory additive CAS latency setting: 0 ∨ Memory additive CAS latency setting: 0 ∨ Memory additive CAS latency setting: 0 ∨ Menory additive CAS latency setting: 0 ∨ Memory additive CAS latency setting: 0 ∨ Maccon MT47H1 Miccon MT47H1 Miccon MT47H1 Miccon MT47H3 Miccon MT47H3 Miccon MT47H3			••••	
Address and command parity Hode Register 0 Burst Length: 8 ∨ Read Burst Type: Sequential ∨ DLL precharge power down: Fast exit ∨ MiCRON MT 18HT MICRON MT 18HT MICRON MT 18HT MICRON MT 38HT MICRON MT 47H1 MICRON MT 47H1 MICRON MT 4	* Memory Initialization Options		•	
Hode Register 0 MiCRON MT 18HT MiCRON MT 48HT MiCRON MT 48HT MiCRON MT 47H1 MiCRON M				
Mode Register 0 MicRon MT 18HT MicRon MT 38HT MicRon MT 47H1 MicRon M	Address and command party			
Read Burst Type: Sequential v DLL precharge power down: Fast exit v Memory CAS latency setting: 6 v Mode Register 1 MICRON MT 3HT Output drive strength setting: 1 v Memory additive CAS latency setting: 0 v Memory additive CAS latency setting: 0 v Memory on-die termination (ODT) setting: 50 v Mode Register 2 SRT Enable:	Mode Register 0			
Read Burst Type: Sequential \not Sequenti \not Sequential \not Sequential \not Sequential \not S	Burst Length:	8 ~	•	MICRON MT 18HT
DLL precharge power down: Fast exit v MicRON MT 18HT Memory CAS latency setting: 6 v Mode Register 1 MicRON MT 47H1 Output drive strength setting: 0 v Memory or die termination (ODT) setting: 50 v Mode Register 2 SRT Enable:	Read Burst Type:		• • •	MICRON MT 18HT
Memory CAS latency setting: 6 v Mode Register 1 0 utput drive strength setting: Output drive strength setting: 0 v Memory additive CAS latency setting: 0 v MicRON MT47H1 MicRON MT47H1 MicRON MT47H3 MicRON MT47H3 MicRON MT47H3 MicRON MT47H3 MicRON MT47H3 V			••••	
Mode Register 1 Image: Solution of the second provided and the second pr		Fast exit 🗸		
Mode Register 1 Output drive strength setting: Full Memory additive CAS latency setting: 0 Memory additive CAS latency setting: 0 Memory on-die termination (ODT) setting: 50 Mode Register 2 SRT Enable: 2x refresh rate v	Memory CAS latency setting:	6 🗸		
Output drive strength setting: Full MICRON MT 47h1 MICRO	Mode Register 1			
Memory additive CAS latency setting: 0 Memory additive CAS latency setting: 0 Memory on-die termination (ODT) setting: 0 Mode Register 2 SRT Enable:	-			
Memory on-die termination (ODT) setting: 50 V Mode Register 2 SRT Enable: 2x refresh rate V			· · · · •	
Mode Register 2 SRT Enable: 2x refresh rate v	Memory additive CAS latency setting:	0 ~	••	MICRON MT47H1
Mode Register 2 SRT Enable: 2x refresh rate v	Memory on-die termination (ODT) setting:	50 🗸		
2xTelleande V	Mode Register 2		<	MICRON MT47H3
2xTelleande V	SRT Enable:	2v refresh rate		
>			Appl	y Update.

Figure 5. DDR2 Controller Memory Parameters Window.

Set the Memory vendor to Hynix, the Memory format to Unbuffered DIMM, the Memory device speed grade to 400.0 MHz, the Total interface width to 64, the Number of Clocks to 2, the Row address width to 14, and the Column address width to 10. In the Memory Initialization Options, set the Memory CAS latency setting to 6, Memory on-die termination (ODT) setting to 50. The other settings should be left at their default values. Click Memory Timing to get to the window shown in Figure 6.

	R2 SDRAM C a_mem_if_ddr2_emif	Controller with UniPHY Intel FPGA IP	Documentation Generate Example Desi
Interface Ty	pe		^ Presets
Enable Hard	d External Memory In	nterface	
PHY Settings	Memory Parameters	Memory Timing Board Settings Controller Settings Diagnostics	Project
	e presets from the pr	e manufacturer data sheet reset list on the right. 19	Library JEDEC DDR2- JEDEC DDR2- JEDEC DDR2- JEDEC DDR2-
tIH (base):	375 p	95	JEDEC DDR2- JEDEC DDR2-
tDS (base):	250 p	05	JEDEC DDR2-
tDH (base):	250 p	05	JEDEC DDR2-
tDQSQ:		5	JEDEC DDR2-
tQHS:		95	JEDEC DDR2-
tDQSCK:		5	JEDEC DDR2-
tDQSS:		:vdes	JEDEC DDR2-
tDQSH:		ydes	JEDEC DDR2-
tDSH:			JEDEC DDR2-
tDSS:		ydes	JEDEC DDR2-
tINIT:		ydes	JEDEC DDR2-
		15	JEDEC DDR2-
tMRD:		ydes	JEDEC DDR2- JEDEC DDR2-
tRAS:		15	MICRON MT1
tRCD:	15.0 n	15	MICRON MT1
tRP:	15.0 n	15	MICRON MT1
tREFI:	7.8 u	IS	MICRON MT 1
tRFC:	127.5 n	ıs	MICRON MT1
tWR:	15.0 n	15	MICRON MT1
tWTR:	3 с	ydes	MICRON MT 1
tFAW:	37.5 n	15	MICRON MT1
tRRD:	7.5 n	15	MICRON MT3
tRTP:	7.5 n	IS	MICRON MT3
			MICRON MT4
			MICRON MT4
			MICRON MT4
			 MICRON MT4
			Apply Upd

Figure 6. DDR2 Controller Memory Timing Window.

Set the timing parameters to the values shown in Table 1 then click on Finish.

If you wish to save the settings of this controller to save time when making another system, press New in the lowerright of the window shown in Figure 7. This will open up a dialog that allows you to give your preset a name and then save it. Now in Figure 7 press Finish to add the component to Platform Designer. Right-click on the component and rename it to *DDR2_Controller*. You should now have the system shown in Figure 7.

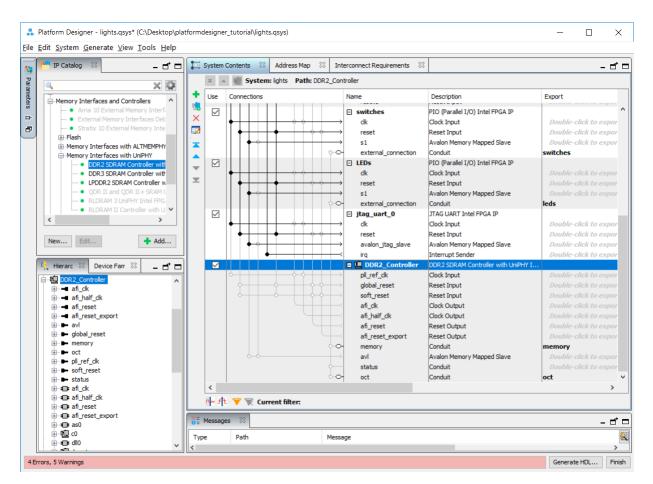


Figure 7. Platform Designer system with the new DDR2 Controller.

Make the following connections:

- Connect the *pll_ref_clk* port of *DDR2_Controller* to the *clk* port of *clk_0*.
- Connect global_reset and soft_reset ports of DDR2_Controller to clk_reset port of clk_0.
- Connect the *jtag_debug_module_reset* port of the Nios II processor to the *soft_reset* port of *DDR2_Controller*.
- Connect the clock input of the *nios2_processor*, *onchip_memory*, *switches*, *LEDs*, and *jtag_uart* to the clock output, *afi_clk* of *DDR2_Controller*.
- Connect the data_master and instruction_master ports of the NIOS II processor to the avl port of DDR2_Controller.

Double click on the *avl* base address of the *DDR2_Controller* and set it to **0x4000_0000**. Your system should now look similar to the one in Figure 8. Right-click on the Nios II processor component to get to the window in Figure 9. Set the Reset Vector memory and Exception Vector memory to DDR2_Controller.avl and press Finish to return

For Quartus[®] Prime 18.1

to the window in Figure 8. Click on Generate HDL... > Generate to generate your system and then close Platform Designer.

Use	Connections	Name	Description	Export	Clock	Base	End	IRQ	Tags
\checkmark		⊟ clk_0	Clock Source						
		clk_in	Clock Input	clk	exported				
		clk_in_reset	Reset Input	reset					
	<	clk	Clock Output	Double-clic	clk_0				
		clk_reset	Reset Output	Double-clic	1				
\checkmark		🗆 nios2_qsys	Nios II (Classic) Processor						
	↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓	clk	Clock Input	Double-clic	DDR2_Co				
	$ \bullet \bullet \bullet \to \bullet$	reset_n	Reset Input	Double-clic					
		data_master	Avalon Memory Mapped Master	Double-clic					
		instruction_master	Avalon Memory Mapped Master	Double-clic					
	$ \downarrow $	d_irq	Interrupt Receiver	Double-clic		IRQ 0	IRQ 31	ı←∖	
		jtag_debug_modul	Reset Output	Double-clic	[clk]				
	$ \bullet \bullet \longrightarrow$	jtag_debug_module	Avalon Memory Mapped Slave	Double-clic	[clk]	© 0x0000_1800	0x0000_1fff		
	× · · · · · · · · · · · · · · · · · · ·	custom_instructio	Custom Instruction Master	Double-clic	1				
\checkmark		onchip_memory2	On-Chip Memory (RAM or ROM)						
	$ \diamond \phi \leftrightarrow \rightarrow$	clk1	Clock Input	Double-clic	DDR2_Co				
	$ \bullet \bullet \longrightarrow$	s1	Avalon Memory Mapped Slave	Double-clic		● 0x0000_0000	0x0000_0fff		
	$ \bullet + \bullet \bullet \rightarrow$	reset1	Reset Input	Double-clic	[clk1]		_		
\checkmark		switches	PIO (Parallel I/O)						
	$ \diamond \phi \phi \rightarrow$	clk	Clock Input	Double-clic	DDR2_Co				
	$ \bullet + \bullet \bullet \rightarrow$	reset	Reset Input	Double-clic	[clk]				
	$ \bullet \bullet \longrightarrow$	s1	Avalon Memory Mapped Slave	Double-clic	[clk]	© 0x0000_2010	0x0000_201f		
	~~	external_connection	Conduit	switches					
\checkmark		🗆 LEDs	PIO (Parallel I/O)						
	$ \diamond \diamond \rightarrow$	clk	Clock Input	Double-clic	DDR2_Co				
	$ \bullet + \bullet \bullet \rightarrow$	reset	Reset Input	Double-clic					
	$ \bullet \bullet \longrightarrow$	s1	Avalon Memory Mapped Slave	Double-clic	[clk]	© 0x0000_2000	0x0000_200f		
	~~	external_connection		leds					
\checkmark		🗆 jtag_uart	JTAG UART						
	$ \diamond \diamond \rightarrow$	clk	Clock Input		DDR2_Co				
		reset	Reset Input	Double-clic					
	♦ ♦ →	avalon_jtag_slave	Avalon Memory Mapped Slave	Double-clic	[clk]		0x0000_2027		
		irq	Interrupt Sender	Double-clic	[clk]			⊨	
\checkmark		DDR2_Controller	DDR2 SDRAM Controller with Un						
	Image: Image	pll_ref_clk	Clock Input	Double-clic	clk_0				
	\bullet	global_reset	Reset Input	Double-clic	1				
	$ \bullet \bullet \circ \circ \rightarrow$	soft_reset	Reset Input	Double-clic	1				
		afi_clk	Clock Output		DDR2_Con				
		afi_half_clk	Clock Output		DDR2_Con				
		afi_reset	Reset Output	Double-clic	1				
		afi_reset_export	Reset Output	Double-clic	1				
	~~	memory	Conduit	memory					
	$\bullet \bullet \longrightarrow$	avl	Avalon Memory Mapped Slave			<pre></pre>	0x7fff_ffff		
		status	Conduit	Double-clic	1				
	-0	oct	Conduit	oct					

Figure 8. Final Platform Designer system with DDR2 Controller.

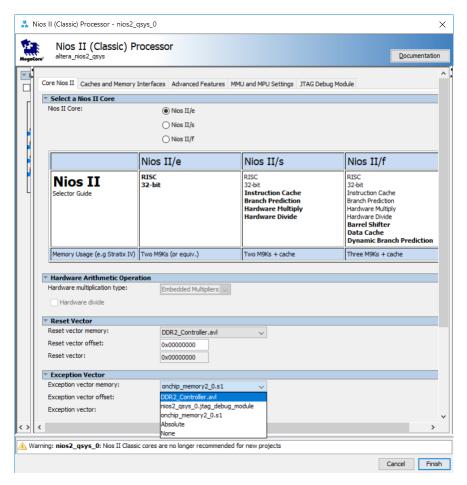


Figure 9. Changing the reset and exception vectors of the NIOS II processor.

6 Integration of the Nios[®] II System into the Quartus[®] Prime Project

Now, we have to instantiate the expanded Nios II system in the top-level Verilog module, as we have done in the tutorial *Introduction to the Intel Platform Designer Tool*. The module is named *lights*, because this is the name of the top-level design entity in our Quartus Prime project.

The new top-level module is presented in Figure 10. The input and output ports of the module use the pin names for the 50-MHz clock, *OSC_50_BANK3*, pushbutton switches, *BUTTON*, toggle switches, *SLIDE_SW*, and LEDs, *LED*, as used in our original design. They also use the pin names *M1_DDR2_addr*, *M1_DDR2_ba*, *M1_DDR2_cas_n*, *M1_DDR2_cke*, *M1_DDR2_clk*, *M1_DDR2_clk_n*, *M1_DDR2_cs_n*, *M1_DDR2_dm*, *M1_DDR2_dq*, *M1_DDR2_dqs*, *M1_DDR2_dqs*, *M1_DDR2_odt*, *M1_DDR2_odt*, *M1_DDR2_ras_n*, *M1_DDR2_we_n*, *M1_DDR2_oct_rdn*, *M1_DDR2_oct_rup* which correspond to the DDR2 SDRAM signals indicated in Figure 2. The pin assignments are included in the file *DE4.qsf* that is included with the design files for this tutorial.

```
// Implements the augmented Nios II system on the DE4 board.
// Inputs: SW3-0 are parallel port inputs to the Nios II system.
             BUTTONO is the active-low system reset.
// Outputs: LEDR3-0 are parallel port outputs from the Nios II system.
             DDR2 ports correspond to the signals in Figure 2
module lights (OSC_50_BANK3, BUTTON, LED, SLIDE_SW, M1_DDR2_addr, M1_DDR2_ba,
    M1_DDR2_cas_n, M1_DDR2_cke, M1_DDR2_clk, M1_DDR2_clk_n, M1_DDR2_cs_n,
    M1_DDR2_dm, M1_DDR2_dq, M1_DDR2_dqs, M1_DDR2_dqsn, M1_DDR2_odt,
    M1_DDR2_ras_n, M1_DDR2_we_n, M1_DDR2_oct_rdn, M1_DDR2_oct_rup);
                    OSC 50 BANK3;
    input
    input
                   BUTTON;
            [3:0]
    input
            [3:0]
                   SLIDE_SW;
    output [7:0]
                   LED;
    output
           [13:0] M1 DDR2 addr;
    output
           [2:0]
                   M1_DDR2_ba;
    output
                   M1_DDR2_cas_n;
    output [1:0]
                   M1_DDR2_cke;
    inout
            [1:0]
                   M1_DDR2_clk;
    inout
           [1:0]
                   M1_DDR2_clk_n;
    output [1:0]
                   M1_DDR2_cs_n;
    output [7:0]
                   M1_DDR2_dm;
           [63:0] M1_DDR2_dq;
    inout
                   M1_DDR2_dqs;
    inout
           [7:0]
    inout
                   M1 DDR2 dqsn;
           [7:0]
    output [0:0]
                   M1_DDR2_odt;
    output
                   M1_DDR2_ras_n;
    output
                   M1_DDR2_we_n;
    input
                   M1_DDR2_oct_rdn;
    input
                    M1 DDR2 oct rup;
```

Figure 10. The top-level module that instantiates the expanded Nios II system. (Part a)

// Instantiate the Nios II system module generated by the Platform Designer tool
 nios_system system (

```
(OSC_50_BANK3),
         .clk_clk
         .reset_reset_n
                             (BUTTON[0]),
         .switches_export (SLIDE_SW),
        .leds_export (LED),
.memory_mem_a (M1_DDR2_addr),
.memory_mem_ck (M1_DDR2_clk),
         .memory_mem_ck_n (M1_DDR2_clk_n),
         .memory_mem_cke
                             (M1_DDR2_cke),
         .memory_mem_cs_n (M1_DDR2_cs_n),
         .memory_mem_dm (M1_DDR2_dm),
         .memory_mem_ras_n (M1_DDR2_ras_n),
         .memory_mem_cas_n (M1_DDR2_cas_n),
         .memory_mem_we_n (M1_DDR2_we_n),
         .memory_mem_dq (M1_DDR2_dq),
         .memory_mem_dqs (M1_DDR2_dqs),
         .memory_mem_dqs_n (M1_DDR2_dqsn),
         .memory_mem_odt (M1_DDR2_odt),
         .oct_rdn (M1_DDR2_oct_rdn),
.oct_rup (M1_DDR2_oct_rup)
    );
endmodule
```

```
Figure 10. The top-level module that instantiates the expanded Nios II system. (Part b).
```

Perform the following:

- Enter the code in Figure 10 into a file called *lights.v*. Add this file and the *nios_system.qip* file produced by the Platform Designer tool to your Quartus Prime project.
- Import the pin assignments from the QSF file included in the design files for this tutorial.
- Perform analysis and synthesis of the design by clicking Processing > Start > Analysis and Synthesis
- Click Tools > Tcl Scripts... to open the window in Figure 11. Select and run the script Project > nios_system > synthesis > submodules > nios_system_DDR2_Controller_p0_pin_assignments.tcl; the script is required to correctly set the differential pins needed by the DDR2 SDRAM Interface.

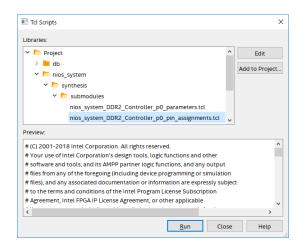


Figure 11. TCL Scripts window in Quartus Prime.

- Compile the project.
- Use the Intel FPGA Monitor Program, which is described in the tutorial *Intel FPGA Monitor Program Tutorial for Nios II* to download and test the system on the DE4 board. Use the assembly program from the tutorial *Intel FPGA Monitor Program Tutorial for Nios II* to test the system; it has been reproduced for you in Figure 12.

If successful, the lights on the DE4 board will respond to the operation of the toggle switches.

.equ .equ .global	Switches LEDs, 0x start		
start:			
_start.			
	movia	r2,	Switches
	movia	r3,	LEDs
loop:	ldbio	r4,	0(r2)
	stbio	r4,	0(r3)
	br	100	p

Figure 12. Assembly language code to control the lights.

7 Using the Clock Crossing Bridge IP Core

A clock crossing bridge allows components clocked by different frequency clock signals to interface and work with each other. This allows you to have low-speed and high-speed components in the same system without compromising the performance of your high-speed components. We will now modify our Nios system so that the human interface components (LEDs and switches) are run by a 50 MHz clock and the other components are run by the clock generated by the DDR2 SDRAM controller component.

Add the clock crossing bridge component to your Platform Designer system by selection Basic Functions > Bridges and Adaptors > Memory Mapped > Avalon-MM Clock Crossing Bridge. The window in Figure 13 will appear. Accept the default settings and press Finish. Right-click on the component and rename it $Clock_Crossing_Bridge$.

👗 Avalon-MM Clock Crossing Bridge - mm_clo	ck_crossing_bridge_0	×
Avalon-MM Clock Crossin altera_avalon_mm_dock_crossing_bridge		Documentation
🕆 Block Diagram		
Show signals	Data Data width:	32
	Symbol width:	
mm_clock_crossing_bridge_0	Symbol widen.	8
	▼ Address	
m0_clk clock avalon m0	Address width:	10
m0_reset reset	Use automatically-determined addres	s width
s0_clk clock	Automatically-determined address width:	10
s0_reset	Address units:	SYMBOLS V
s0 avaion		
	* Burst	
altera_avalon_mm_clock_crossing_bridge	Maximum burst size (words):	1 🗸
	* FIFOs	
	Command FIFO depth:	4 ~
	Response FIFO depth:	4 🗸
	Master dock domain synchronizer depth:	2
	Slave clock domain synchronizer depth:	2
		Cancel Finish

Figure 13. The Avalon-MM Clock Crossing Bridge

Make the following changes to your Nios system:

- Remove the connections from the *data_master* port of *nios2_processor* and the *s1* ports of *switches* and *LEDs*.
- Connect the *m0* port of *Clock_Crossing_Bridge* to the *s1* ports of *switches* and *LEDs*.

- Connect the *data_master* port of *nios2_processor* to the *s0* port of *Clock_Crossing_Bridge*.
- Connect the *clk* port of *clk_0* to the *s0_clk* port of *Clock_Crossing_Bridge* and the *clk* ports of *switches* and *LEDs*.
- Connect the *afi_clk* port of *DDR2_Controller* to the *m0_clk* port of *Clock_Crossing_Bridge*.
- Connect the *jtag_debug_module_reset* of *nios2_processor* to the *m0_reset* and *s0_reset* ports of *Clock_Crossing_Bridge*.
- Connect the *clk_reset* port of *clk_0* to the *m0_reset* port of *Clock_Crossing_Bridge*.

Double-click on the base address of *Clock_Crossing_Bridge* and set it to 0x1400. Similarly, set the base address of *switches* to 0x0000 and *LEDs* to 0x0010. The Nios processor will access these components at the address: bridge base address + component base address. For *switches* this will be address 0x1400 and for *LEDs* this will be 0x1410. The complete system is shown in Figure 14. Regenerate the system then recompile the top-level Verilog module. Finally use the updated assembly program given in Figure 15 to test your system with the Intel FPGA Monitor Program.

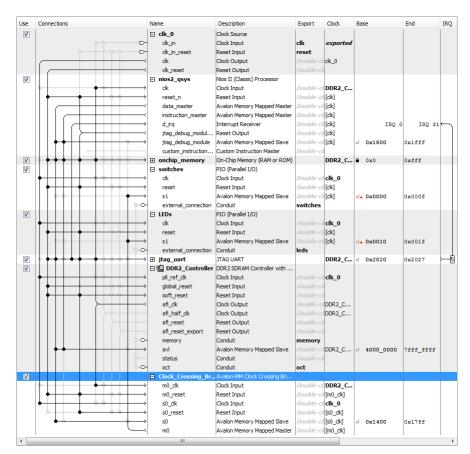


Figure 14. The final Nios II system.

.equ .equ .global	Switches LEDs, 0x _start		
_start:			
	movia	r2,	Switches
	movia	r3,	LEDs
loop:	ldbio	r4,	0(r2)
	stbio	r4,	0(r3)
	br	100	p

Figure 15. New Assembly language code to control the lights.

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

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