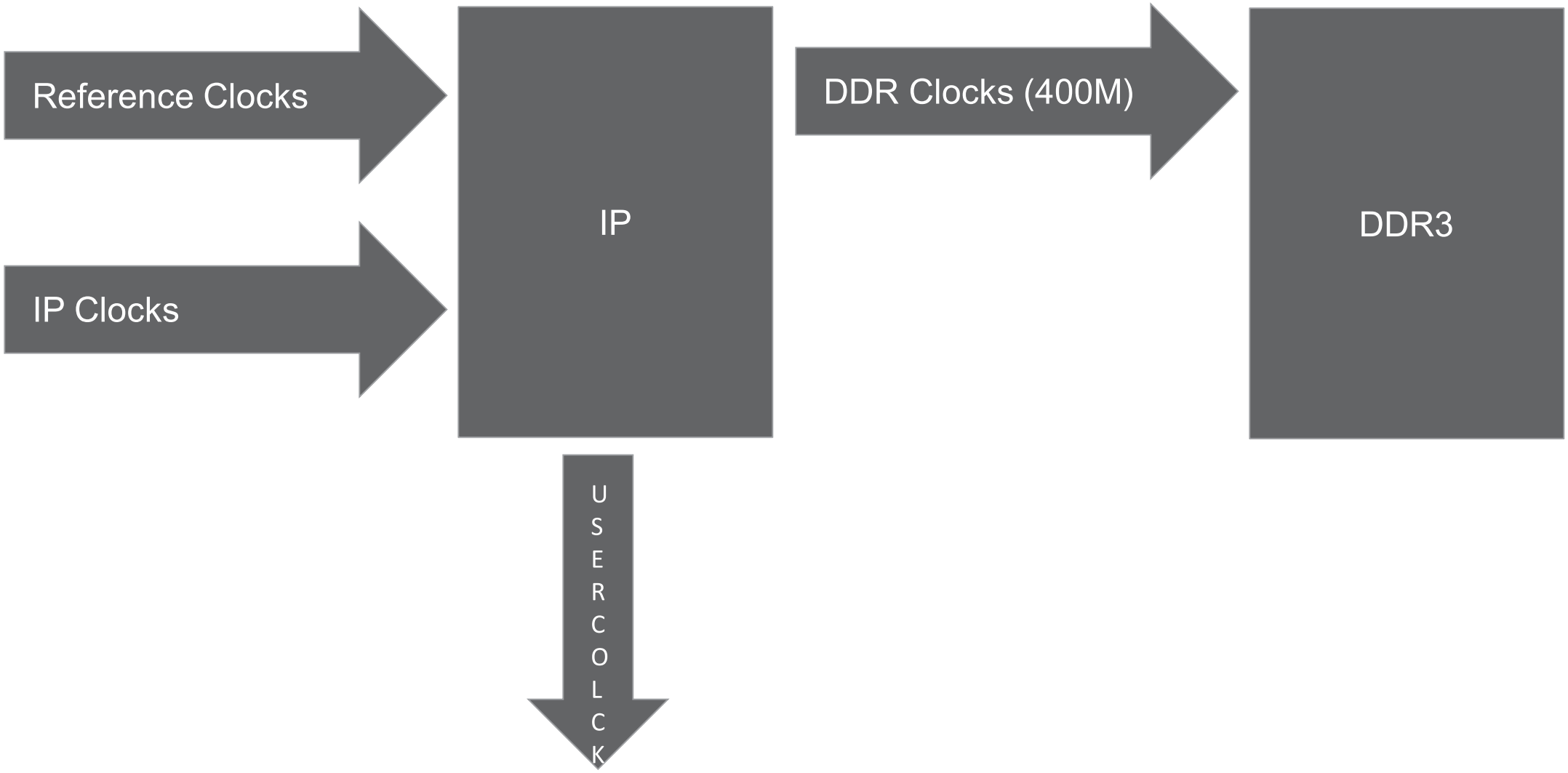
MIG 7 Series IP Overview

AMD
together we advance_

Agenda

- Customizing the IP
- Simulating the Example Design
- Simulating Read / Write with AXI VIP
- Connecting the MIG to a Custom Design
- Connecting the MIG to two AXI Master VIP using AXI Smart Connect



Customizing the IP Step 1

Project Summary

Overview | Dashboard

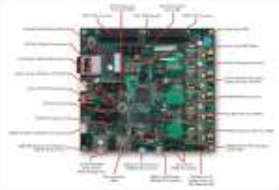
Settings Edit

Project name:	MIG_lab_example_design
Project location:	C:/Users/User/Desktop/vince/0.meself/0.Agenda_vince/3.Basic Environment Infrastructure/MIG_lab_example_design/MIG_lab_example_design
Product family:	Zynq-7000
Project part:	Zynq 7000 ZC702 Evaluation Board (xc7z020clg484-1)
Top module name:	Not defined
Target language:	Verilog
Simulator language:	Mixed

Board Part

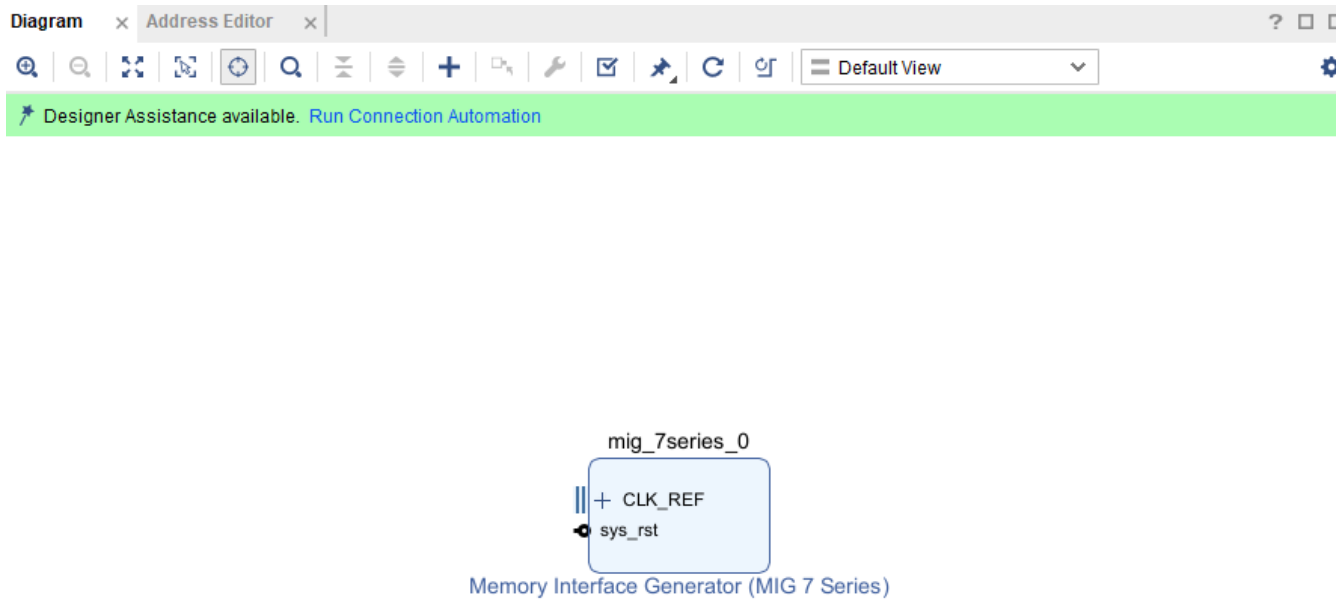
Display name:	Zynq 7000 ZC702 Evaluation Board
Board part name:	xilinx.com:zc702:part0:1.4
Board revision:	1.0
Connectors:	No connections
Repository path:	C:/Xilinx/Vivado/2022.2/data/xhub/boards
URL:	www.xilinx.com/zc702
Board overview:	Zynq 7000 ZC702 Evaluation Board

Changes

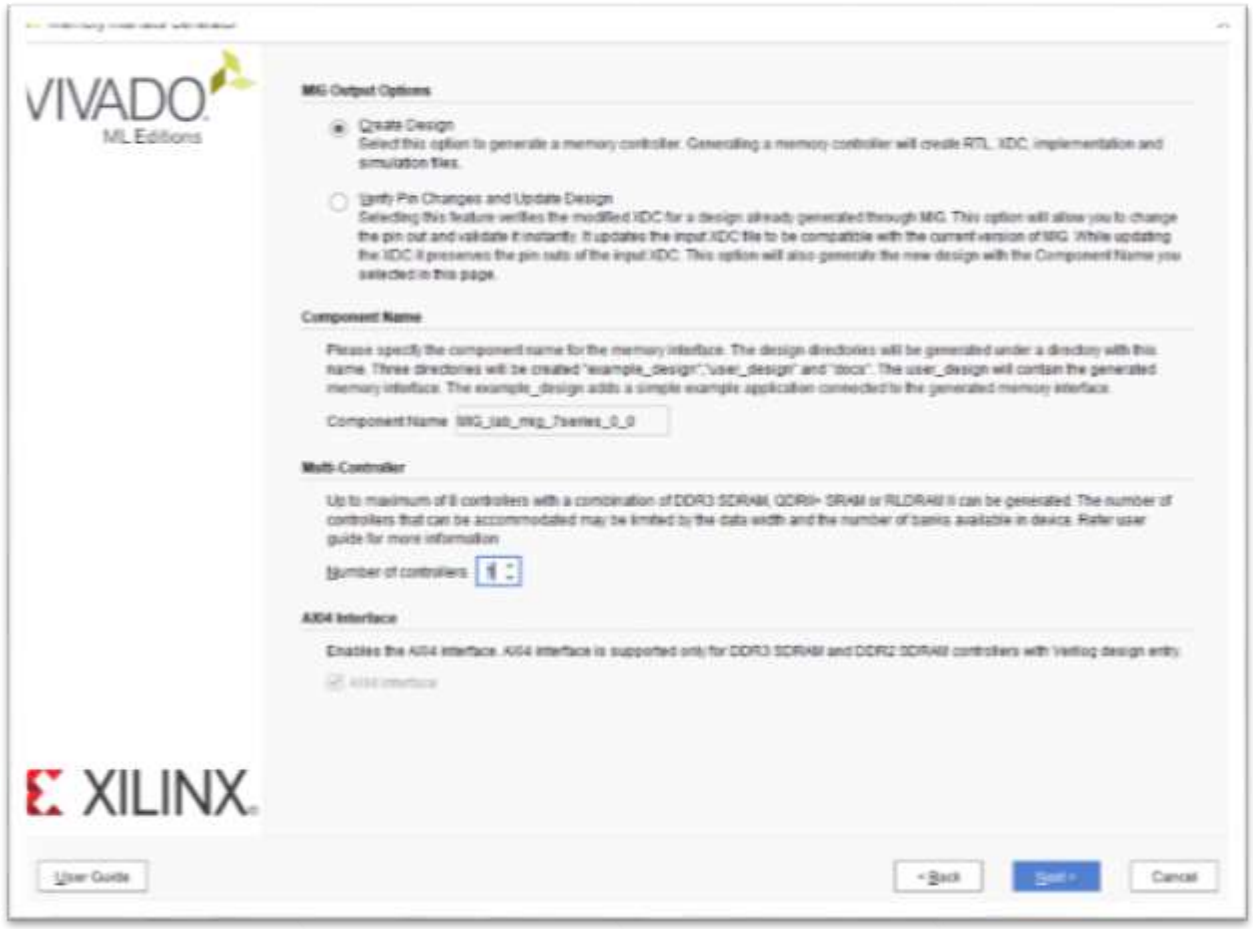


Customizing the IP

Create a new block diagram (BD) and use the IP catalog to add a new IP to the BD - in this case, the “Memory Interface Generator (MIG 7 Series)” core. If using a board, a prepackaged MIG may be available. We can customize it by double clicking it.



Memory Interface Generator



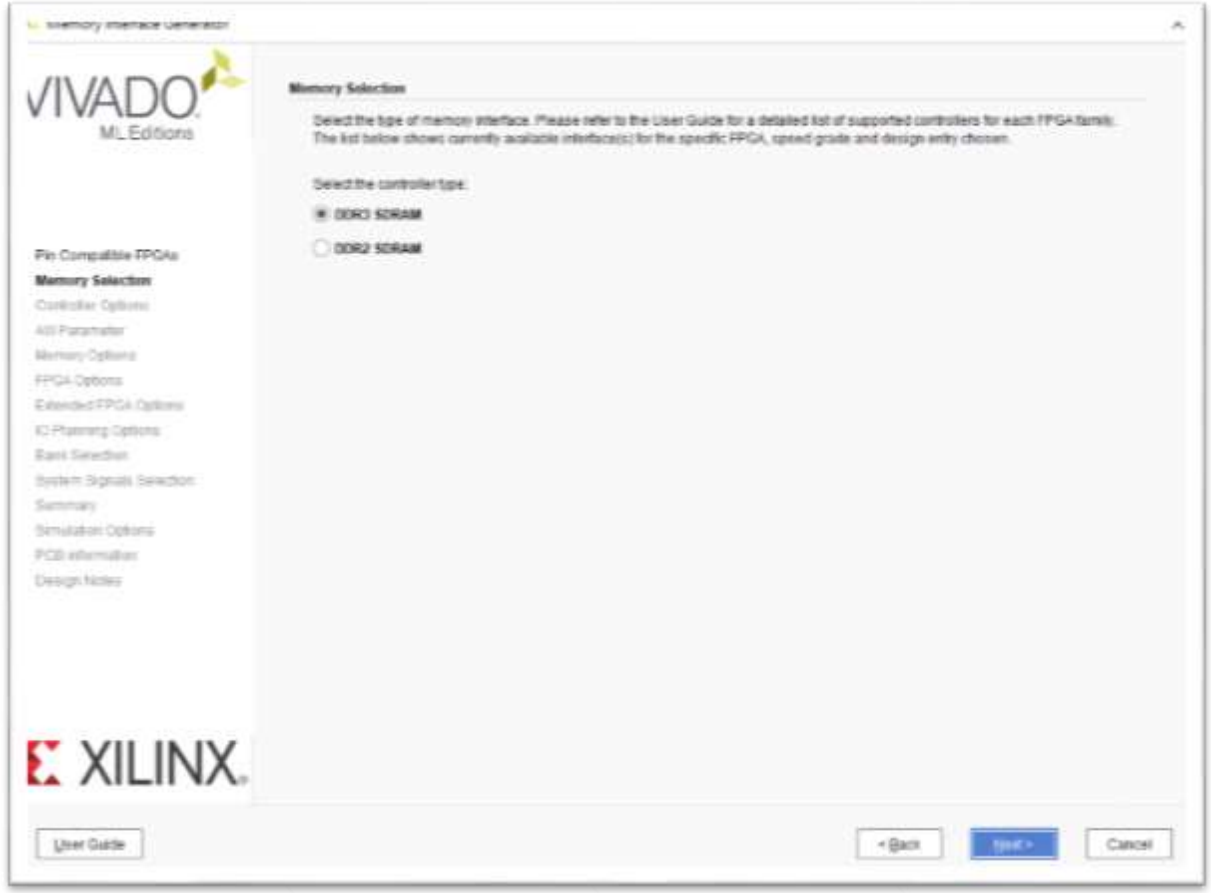
Select the “Create Design”
Click “Next”
(“Number of Controllers” mean is numbers of controller you need)
“AXI4 Interface” MIG using AXI4 Interface.

•Make sure the AXI4 interface is enabled and select DDR3 SDRAM.

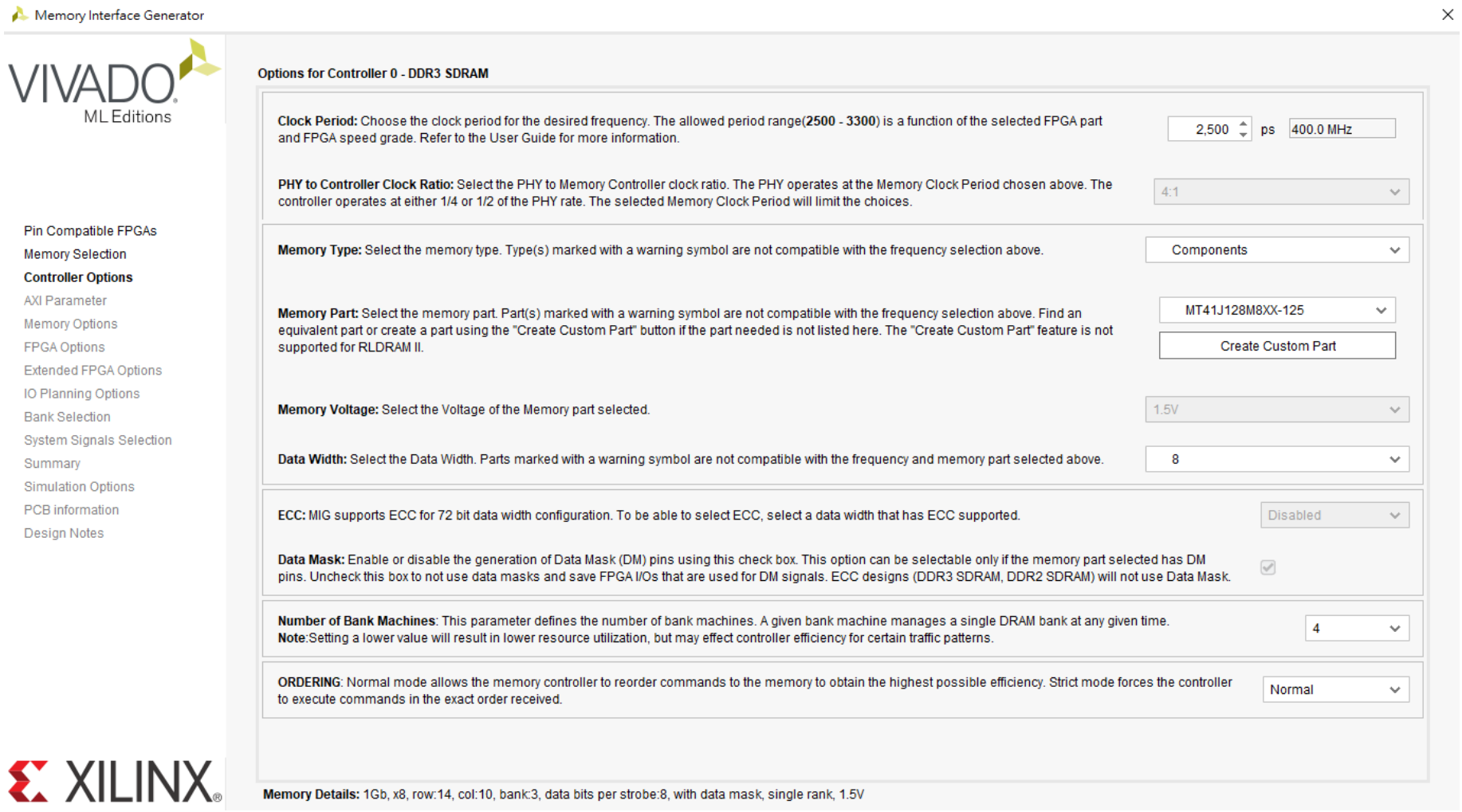
Memory Interface Generator

Pin Compatible FPGAs

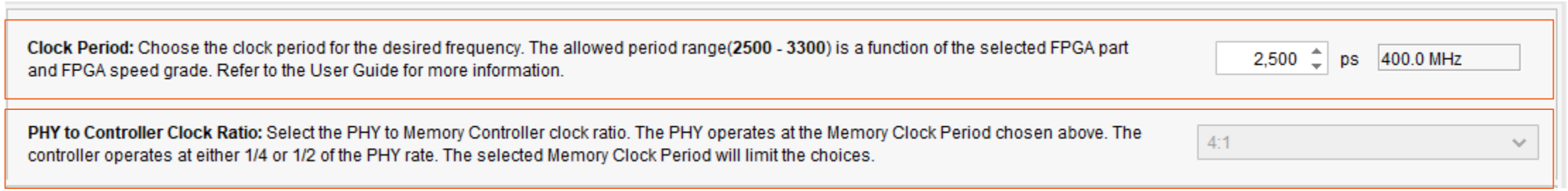
Memory selection



Memory Interface Generator - Option for Controller



Memory Interface Generator - Option for Controller



1

2

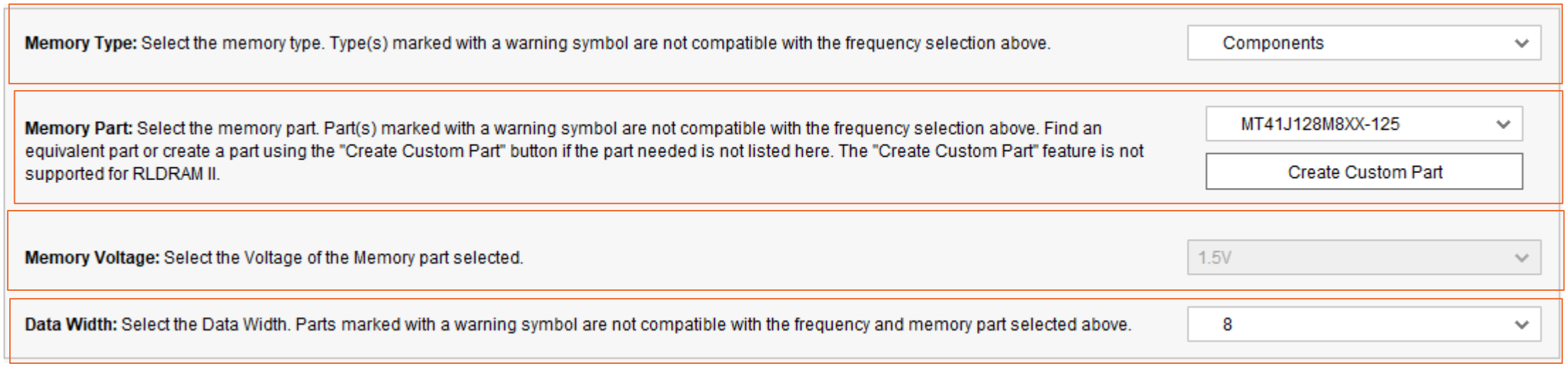
1.

The desired clock period must be between 2500 and 3300ps. For now, use 2500ps (400 MHz), as this is the speed of the actual physical DDR3 RAM transactions.

2.

Make sure the PHY to Controller Clock Ratio is **4:1 (ensuring that the physical DDR RAM will operate at 400 MHz, but the controller stays at 100 MHz i.e., ui_clk = 100 MHz).**
The ui_clk using by User want to reading the MIG controller address.

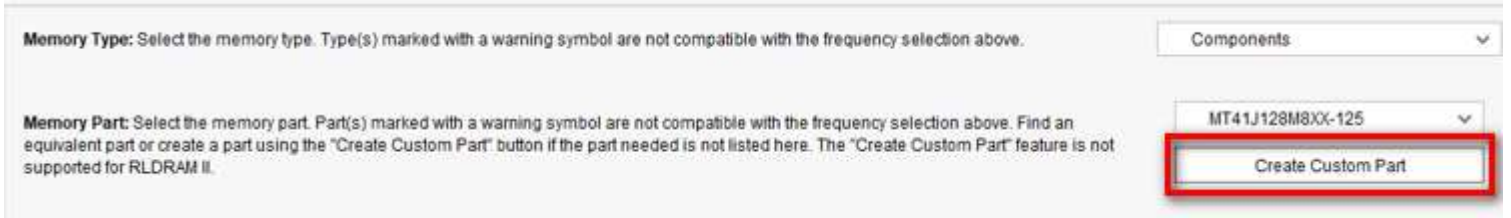
Memory Interface Generator - Option for Controller



- Memory Type – This feature selects the type of memory parts used in the design.
- Memory Part – This option selects a memory part for the design. Selections can be made from the list or a new part can be created. Note: For a complete list of memory parts available, see Answer Record: 54025.
- Data Width – The data width value can be selected here based on the memory type selected earlier. The list shows all supported data widths for the selected part. One of the data widths can be selected. These values are generally multiples of the individual device data widths. In some cases, the width might not be an exact multiple. For example, 16 bits is the default data width for x16 components, but eight bits is also a valid value.
- Data Mask – This option allocates data mask pins when selected. This should be deselected to deallocate data mask pins and increase pin efficiency. Also, this is disabled for memory parts that do not support data mask.

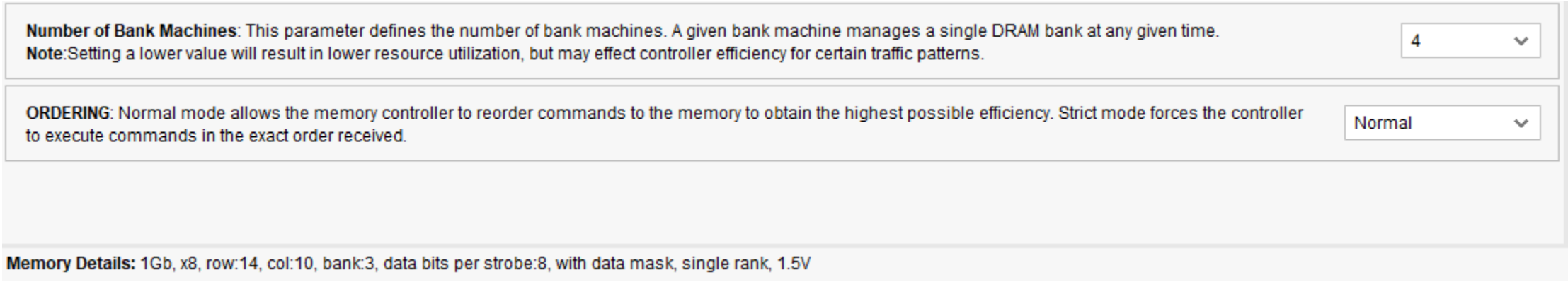
https://www.crucial.tw/support/articles-faq-memory/common-memory-specifications

Memory Interface Generator - Option for Controller Create Custom Part



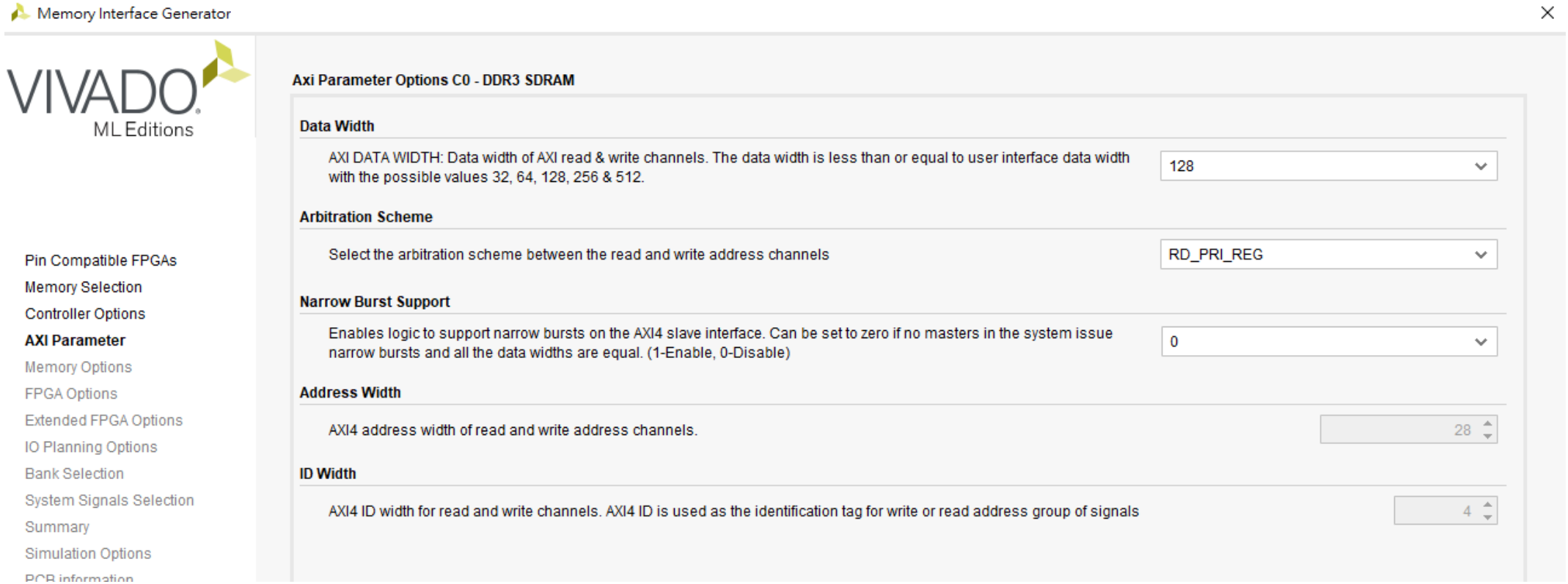
1. On the Controller Options page select the appropriate frequency. Either use the spin box or enter a valid value using the keyboard. Values entered are restricted based on the minimum and maximum frequencies supported.
2. Select the appropriate memory part from the list. If the required part or its equivalent is unavailable, a new memory part can be created. To create a custom part, click the Create Custom Part below the Memory Part pull-down menu. A new page appears, as shown in Figure.
3. Select the suitable base part from the Select Base Part list.
4. Enter the appropriate memory part name in the text box.
5. Edit the value column as needed.
6. Select the suitable values from the Row, Column, and Bank options as per the requirements.
7. After editing the required fields, click Save. The new part is saved with the selected name. This new part is added in the Memory Parts list on the Controller Options page. It is also saved into the database for reuse and to produce the design.
8. Click Next to display the Memory Options page (or the AXI Parameter Options page if AXI Enable is checked on the Memory Type selection page).

Memory Interface Generator - Option for Controller



- Number of Bank Machines – The list shows the number of bank machines that are supported for the selected design configuration.
- Ordering – This feature allows the Memory Controller to reorder commands to improve the memory bus efficiency.
- Memory Details – The bottom of the Controller Options Figure displays the details for the selected memory configuration Figure.

Memory Interface Generator – AXI Parameter



MIG Pin Definition

1,DDR3 level standard

DDR3 level standard is SSTL15, 1.5V+- 0.075V

2,DDR3 electrical interface parameters

Pin definition : inout [31:0] ddr3_dq;

Pin description : Data input and output, bidirectional data.

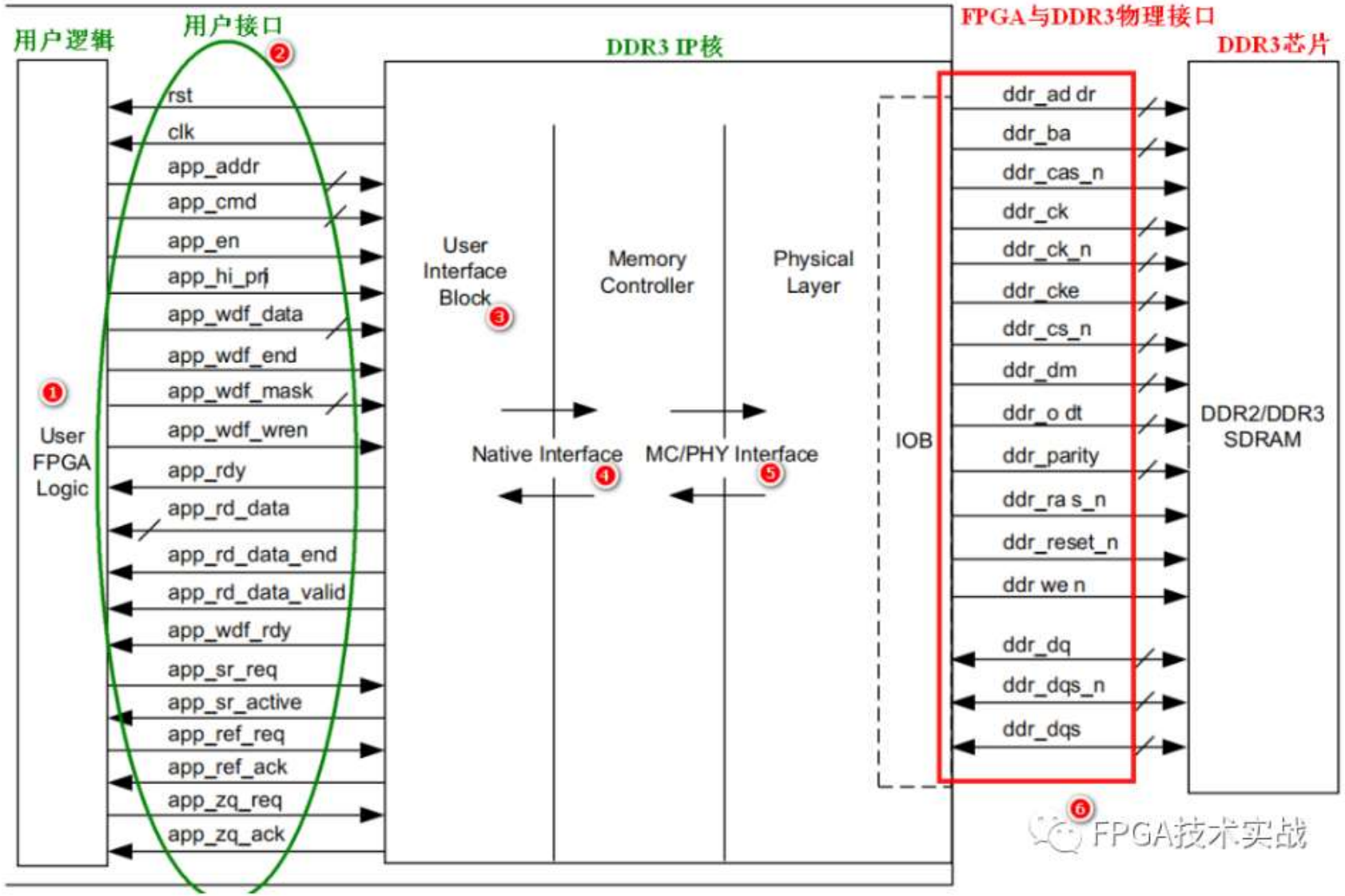
3, ddr3_dps_p & ddr3_dps_n

Pin definition : inout [3:0] ddr3_dps_n/p

Pin description : Data strobe when data is read, is the output for DDR3, and the edges are aligned with the data being read. When writing data, for DDR3 it is the input and the cener is aligned with the write data.

4,

Solution Design



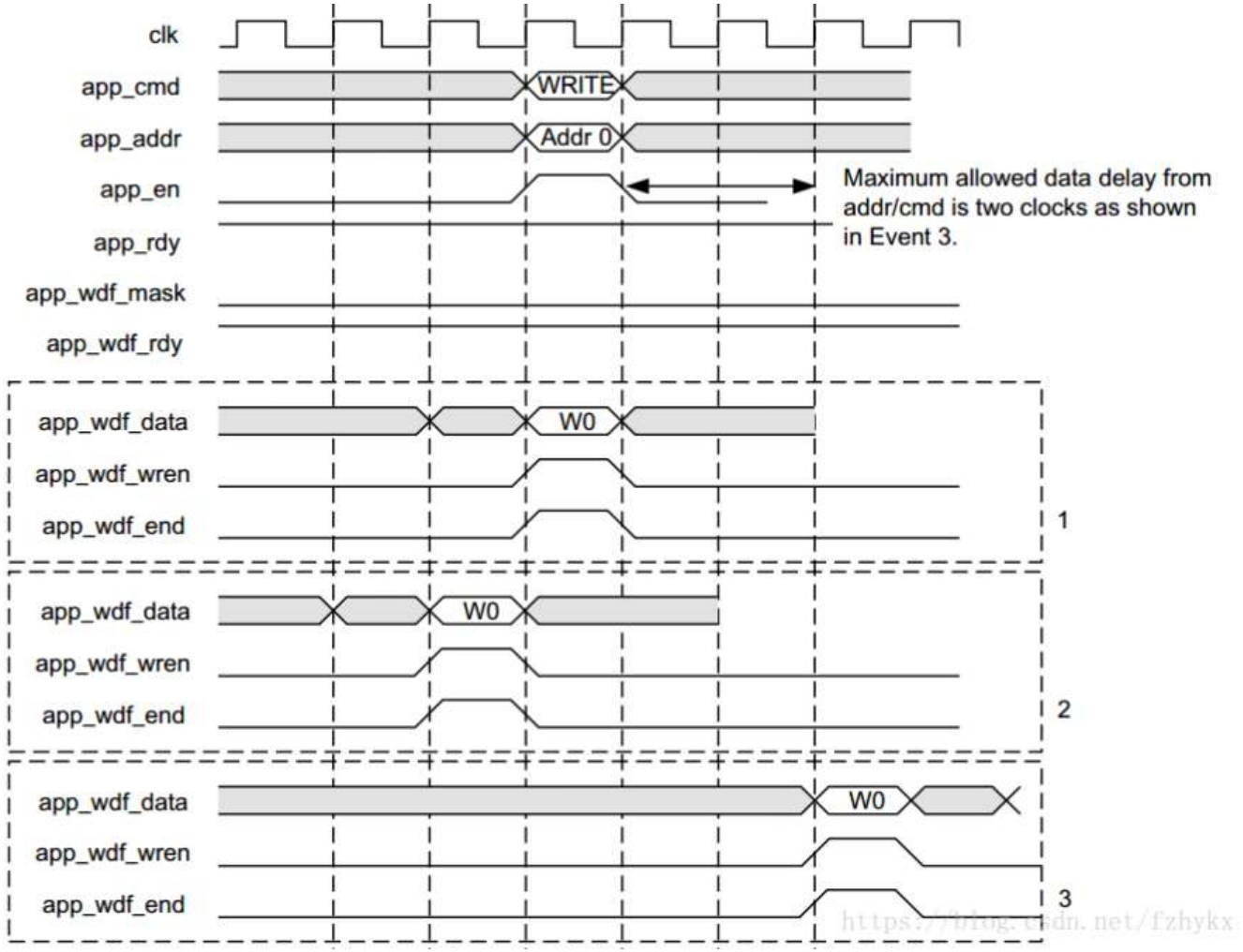
Pin	Direction	Note
ddr3_addr	Output	
ddr3_ba	Output	
ddr3_cas_n	Output	
ddr3_ck_n	Output	
ddr3_ck_p	Output	
ddr3_cke	Output	
ddr3_ras_n	Output	
ddr3_reset_n	Output	復位狀態信號
ddr3_we_n	Output	
ddr3_dq	inout	
ddr3_dps_n	inout	
ddr3_dps_p	inout	
Init_calib_complete	Output	初始化完成信號
ddr3_cs_n	Output	
ddr3_dm	Output	
ddr3_odt	Output	

sys_clk_i	input	The clk for board
clk_ref_i	input	The clk for board
sys_rst	Input	The clk for reset of board

Pin	Direction	Note
app_addr	input	要操作地址每次step 為8
app_cmd	input	寫000 讀001
app_en	input	Enable single
app_wdf_data	input	Write data [255:0]
app_wdf_end	input	Last data single
app_wdf_wren	input	Write enable single
app_rd_data	output	Read data [255:0]
app_rd_data_end	output	Last data single()
app_rd_data_valid	output	Read enable
app_rdy	output	
app_wdf_rdy	output	
app_sr_req	Input	Set 0
app_ref_req	Input	Set 0
app_zq_req	Input	Set 0
app_sr_active	Output	
app_ref_ack	Output	
app_zq_ack	Output	
ui_clk	Output	Users clk
ui_clk_sync_rst	Output	Users reset
app_wdf_mask	input	Keep single

Time analyze

- Write the timing (wr_en high) , we can perform operations when the **app_rdy** and **app_wdf_ready** all are **high**, The command pull **app_en high** and **write the data and address at the same time.**(DDR allows the WR_EN signal to lag within two CMD and two CLOCK, but it is still recommended to write within the same cycle.)



Read timing

- read the timing (rd_en high) , we can perform operations when the **app_en** and **app_addr** all are **high**,than waiting valid single (valid data), **How many commands were issued (CLK) and how many commands were responded to (CLK).**

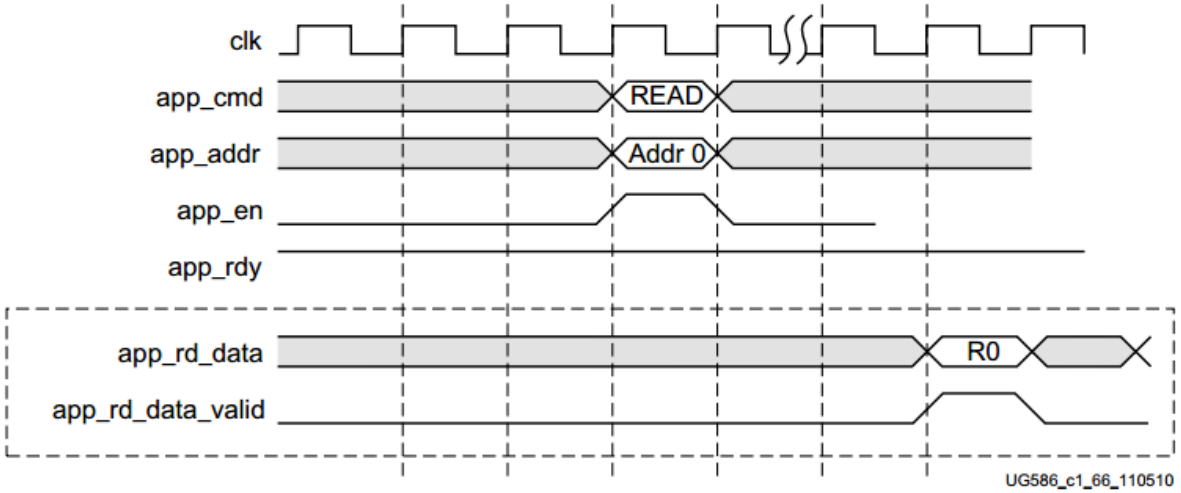
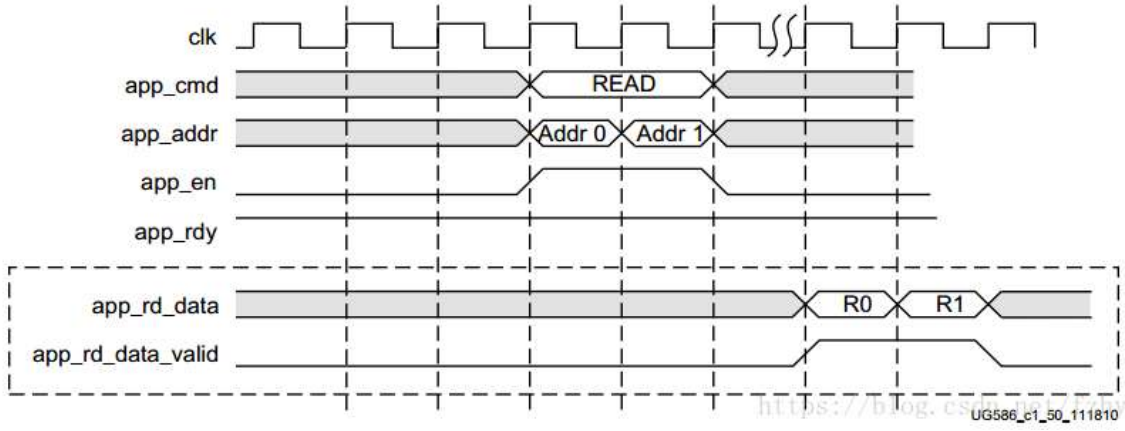


Figure 1-81: **4:1 Mode UI Interface Read Timing Diagram (Memory Burst Type = BL8)**




AMD