



ALPHA DATA

**ADM-XRC-KU1
Standalone DDR4 Test
FPGA Design
Release: 1.0.0**

**Document Revision: 1.0
22 June 2016**

**© 2016 Copyright Alpha Data Parallel Systems Ltd.
All rights reserved.**

This publication is protected by Copyright Law, with all rights reserved. No part of this publication may be reproduced, in any shape or form, without prior written consent from Alpha Data Parallel Systems Ltd.

Head Office

Address: 4 West Silvermills Lane,
Edinburgh, EH3 5BD, UK
Telephone: +44 131 558 2600
Fax: +44 131 558 2700
email: sales@alpha-data.com
website: <http://www.alpha-data.com>

US Office

3507 Ringsby Court Suite 105,
Denver, CO 80216
(303) 954 8768
(866) 820 9956 toll free
sales@alpha-data.com
<http://www.alpha-data.com>

All trademarks are the property of their respective owners.

Table Of Contents

1	Introduction	1
1.1	Structure of this package	1
1.2	Known issues	2
2	Design description	3
2.1	Testbench	4
3	Building the Standalone DDR4 Test	6
3.1	Workaround for custom part implementation failure in Vivado	6
4	Using the Standalone DDR4 Test	8
4.1	Monitoring & controlling the Standalone DDR4 Test using VIO cores	8
5	Simulating the Standalone DDR4 Test	11
5.1	Workaround for custom part simulation error in Vivado	11
5.2	Vivado Simulator (XSIM) cannot be used to simulate this FPGA design	12

List of Tables

Table 1	Design Constraints	4
Table 2	Project creation scripts by configuration	6

List of Figures

Figure 1	Structure of this package	1
Figure 2	Block diagram of Standalone DDR4 Test FPGA Design	3
Figure 3	Block diagram of the testbench	5
Figure 4	Vivado Hardware Manager after debug Tcl script	9
Figure 5	Vivado Hardware Manager after configuring debug probes	10
Figure 6	Simulation settings	11

1 Introduction

Supported Vivado versions

This version of the **ADM-XRC-KU1 Standalone DDR4 Test FPGA Design** can be built with Vivado 2015.4 to 2016.2. Alpha Data cannot guarantee that this FPGA design will be fully compatible with releases of Vivado later than 2016.2.

Please review [Xilinx Quality Alert XCEN15040](#).

This document describes the **ADM-XRC-KU1 Standalone DDR4 Test FPGA Design**. For a detailed discussion of using Xilinx DDR4 SDRAM (MIG) IP with the ADM-XRC-KU1 reconfigurable computing card, refer to the document [Using Xilinx DDR4 SDRAM \(MIG\) IP with the ADM-XRC-KU1](#).

This FPGA design demonstrates the use of Xilinx DDR4 SDRAM (MIG) IP to interface to the DDR4 SDRAM fitted to an ADM-XRC-KU1 reconfigurable computing card.

It includes the following elements:

- Four DDR4 SDRAM controllers, one per bank, using **Xilinx DDR4 SDRAM (MIG) IP**.
- Four **memory exerciser** instances, one per DDR4 SDRAM controller.
- Four **Xilinx VIO cores**, one per **memory exerciser**, for controlling the memory test and issuing a reset to each bank.

The memory test can be initiated by manipulating the VIO cores using Vivado's Hardware Manager. Running the Standalone DDR4 Test in hardware is described in [Section 4](#).

A Tcl script for creating a Vivado project is provided. Refer to [Section 3](#) for a list of the available configurations and their respective Tcl scripts.

1.1 Structure of this package

The files and folders making up the Standalone DDR4 Test FPGA Design are organized as in [Figure 1](#) below:

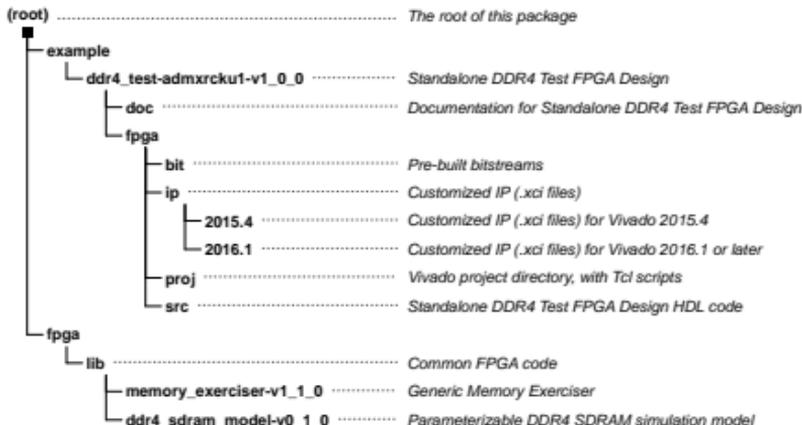


Figure 1 : Structure of this package

The root of this package, i.e. the directory which forms the root of the tree of directories and files making up this package, is referred to in the remainder of this document as **(root)**.

The base directory of the FPGA design, i.e. **(root)/example/ddr4_test-admxrcku1-v1_0_0** is referred to in the remainder of this document as **(design)**.

1.2 Known issues

The following issues are known to exist in this release of the Standalone DDR4 Test:

- 1) As of Vivado 2015.4 to 2016.2, the Vivado simulator (XSIM) cannot be used to simulate this FPGA design. This issue is discussed in [Section 5.2](#).
- 2) An issue in Vivado / DDR4 SDRAM (MIG) IP may result in simulation errors for DDR4 configurations using a custom part. This issue and the workaround are discussed in [Section 5.1](#).
- 3) An issue in Vivado / DDR4 SDRAM (MIG) IP may result in implementation failures for DDR4 configurations using a custom part. This issue and the workaround are discussed in [Section 3.1](#).

2 Design description

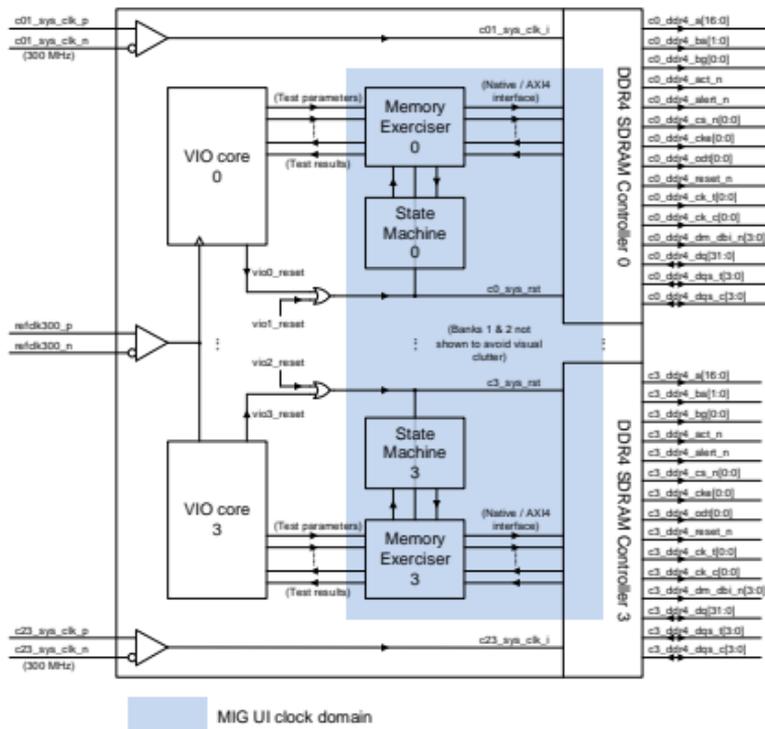


Figure 2 : Block diagram of Standalone DDR4 Test FPGA Design

The FPGA design depicted in Figure 2 has two main variants:

- Using the DDR4 SDRAM (MIG) IP with a 256-bit wide native interface. The top-level module is implemented by `ddr4_test_native.vhd`.
- Using the DDR4 SDRAM (MIG) IP with a 256-bit wide AXI4 interface. The top-level module is implemented by `ddr4_test_axi.vhd`.

In both variants, the structure of the FPGA design is the same; the difference is the protocol used by the DDR4 SDRAM controllers to communicate with the memory exercisers. In the native interface variant of the FPGA design, each memory exerciser drives the native interface of its associated DDR4 SDRAM controller. In the AXI4 interface variant of the FPGA design, the memory exerciser is an AXI4 master, driving the AXI4 slave interface of its associated DDR4 SDRAM controller.

The FPGA design consists of four similar sections, where each section contains a DDR4 SDRAM controller plus memory test logic and functions independently of the other sections.

Overall control of the system is accomplished via VIO cores 0 to 3. In order to perform a memory test, a VIO core pulses the **vioN_reset** signal, where N is 0 to 3. This asserts **cn_sys_rst**, which resets the corresponding memory controller and memory test logic. Once **cn_sys_rst** is deasserted, state machine N begins its sequence.

As well as the the **vioN_reset** signal, each VIO core outputs parameters for the memory test: the starting logical address (**ctln_offset**) number of logical words to test (**ctln_length + 1**). A logical word is defined to be a word of data as transferred on the DDR4 SDRAM (MIG) UI, whether native or AXI4. Although these signals are brought from the VIO clock domain into the DDR4 SDRAM MIG UI clock domain without any synchronization logic, they are effectively static during the memory test as long as the user does not change them after pulsing **vioN_reset**. The memory test parameters can be different for each bank of DDR4 SDRAM.

When **cn_sys_rst** is asserted and deasserted, the corresponding section of the system executes the following sequence independently of the other sections:

- 1 State machine N waits until the associated DDR4 SDRAM controller asserts **cn_init_calib_complete** and deasserts **cn_ui_clk_sync_rst**.
- 2 State machine N then pulses **ctln_go** for its associated memory exerciser, and transitions permanently to a "halt" state.
- 3 When memory exerciser N, which at this point is in its idle state, sees **ctln_go** asserted, it begins to execute a memory test. This consists of a number of phases of reads and writes of the corresponding bank of DDR3 SDRAM. Additionally, the memory exerciser deasserts **ctln_done** signal to indicate that a memory test is in progress. During the memory test, the memory exerciser checks data read back against expected data.
- 4 When memory exerciser N finishes the memory test, it asserts its **ctln_done** signal to indicate that a memory test is not in progress. The **ctln_done** signal, among others, can be monitored in Vivado Hardware Manager via VIO core N.

The constraints required by the FPGA design are located in **.xdc** files in the **src** folder. The constraints are split over several files according to function and are listed in [Table 1](#):

Constraint File in (design)/fpga/src/	Description
refclk300.xdc	300 MHz reference clock constraints
ddr4sdram.xdc	DDR4 SDRAM common constraints
ddr4sdram_locs_b0_x32.xdc	DDR4 SDRAM bank 0 constraints
ddr4sdram_locs_b1_x32.xdc	DDR4 SDRAM bank 1 constraints
ddr4sdram_locs_b2_x32.xdc	DDR4 SDRAM bank 2 constraints
ddr4sdram_locs_b3_x32.xdc	DDR4 SDRAM bank 3 constraints
ddr4_test_native.xdc	Native design variant constraints
ddr4_test_axi.xdc	AXI4 design variant constraints
bitstream.xdc	Bitstream generation constraints

Table 1 : Design Constraints

2.1 Testbench

Note

As of Vivado 2015.4 to 2016.2, the Vivado simulator (XSIM) cannot be used to simulate this FPGA design. This issue is discussed in [Section 5.2](#).

The testbench for simulating the design is **(design)/fpga/src/tb_ddr4_test.vhd**. When a simulation is

3 Building the Standalone DDR4 Test

Tcl scripts to create the Vivado projects for the various configurations of the FPGA design are found in the **(design)/fpga/proj** directory. These can be **sourced** within the Vivado GUI, or **sourced** by Vivado in batch mode. The available Tcl scripts are listed in [Table 2](#):

SDRAM density & speed	MIG options	Project creation script in (design)/fpga/proj
8 GiB 2400 MT/s	Native interface	mkxpr-8g_2400_x32_native-ku060_2e.tcl
		mkxpr-8g_2400_x32_native-ku115_2e.tcl
	AXI4 interface	mkxpr-8g_2400_x32_axi4-ku060_2e.tcl
		mkxpr-8g_2400_x32_axi4-ku115_2e.tcl

Table 2 : Project creation scripts by configuration

To generate a project, start a shell or command prompt, and issue a command of the following form:

```
cd /path/to/fpga/proj
vivado -mode batch -source <.tcl script>
```

(Windows users should use backslashes in the **cd** command, rather than forward slashes.)

For example, to generate the Vivado project for the configuration of DDR4 SDRAM (MIG) IP native interface and 8 GiB SDRAM @ 2400 MT/s (the 1st configuration in [Table 2](#) above), invoke Vivado as follows:

```
cd /path/to/fpga/proj
vivado -mode batch -source mkxpr-8g_2400_x32_native-ku060_2e.tcl
```

(Windows users should use backslashes in the **cd** command, rather than forward slashes.)

After one or more projects have been generated using the scripts listed in [Table 2](#), they can be opened in the Vivado GUI.

Scripts are also provided in the same directory to fully rebuild a Vivado project or all Vivado projects via the shell or command prompt. These are named similarly to the **mkxpr** scripts, except that the prefix is **rebuild**. For example, to rebuild the Vivado project for the configuration of DDR4 SDRAM (MIG) IP native interface and 8 GiB SDRAM @ 2400 MT/s, invoke Vivado as follows:

```
cd /path/to/fpga/proj
vivado -mode batch -source rebuild-8g_2400_x32_native-ku060_2e.tcl
```

(Windows users should use backslashes in the **cd** command, rather than forward slashes.)

Note

The pre-built bitstreams, under **(design)/fpga/bit/<configuration>**, are **not** overwritten when the FPGA design is built.

3.1 Workaround for custom part implementation failure in Vivado

After opening the Vivado project, when generating the output products for DDR4 SDRAM (MIG) IP that uses a custom part, the following error (among others) may occur during the Out-of-Context module synthesis run:

[Synth 8-524] part-select [151:144] out of range of prefix 'mcal_ADR' ...

This will prevent the successful implementation of the FPGA design. If this issue occurs, it can be worked around by performing the following steps:

- (1) Open the project in the Vivado GUI.
- (2) Select "Tools -> Run Tcl script..." from the main menu and select **(design)/fpga/proj/fix_custom_part.tcl**. This script unsets the properties on the DDR4 SDRAM (MIG) IP related to custom parts and then restores them. It then resets the IP's output products. This is sufficient to work around this issue.
- (3) **Without closing the project**, generate the output products of the DDR4 SDRAM (MIG) IP (or implement the design, which also generates the output products). This time, no critical warnings or errors should occur.

This workaround must be applied each time that (a) the project is closed and reopened **and** (b) the DDR4 SDRAM (MIG) IP's output products must again be generated.

4 Using the Standalone DDR4 Test

To use the Standalone DDR4 Test, first determine the configuration of interest; for example, **8g_2400_x32_native-ku060**. For convenience, **.bit** and **.Jtx** files are provided pre-built for each configuration, under the **(design)/fpga/bit/<configuration>/** directory. The available configurations are as per [Table 2](#).

Vivado Hardware Manager should then be used to configure the FPGA with the **.bit** file. The associated **.Jtx** file tells Vivado Hardware Manager about debug probes for VIO cores etc., and so must not be omitted when configuring the FPGA.

For convenience, a debug Tcl script is provided with each pre-built **.bit** file (in the same directory), for automating the process of opening Vivado Hardware Manager, connecting to the FPGA and downloading the **.bit** file. For a particular configuration, the path of the debug Tcl script is:

(design)/fpga/bit/<configuration>/debug-<configuration>.tcl

Each debug Tcl script is intended to be **sourced** within the Vivado GUI, and is designed to work correctly whether executed on the development machine with the corresponding Vivado project already open, or on a lab machine which does not have the corresponding Vivado project already open.

4.1 Monitoring & controlling the Standalone DDR4 Test using VIO cores

In Vivado Hardware Manager, the four VIO core instances are used to control and monitor the Standalone DDR4 Test.

To control the Standalone DDR4 Test, execute the debug Tcl script from the directory **(design)/fpga/bit/<configuration>/** for the configuration of interest. This will configure the FPGA with the appropriate **.bit** file and associate the ports of the VIO cores with recognizable signal names. At this point, Vivado Hardware Manager looks like this:

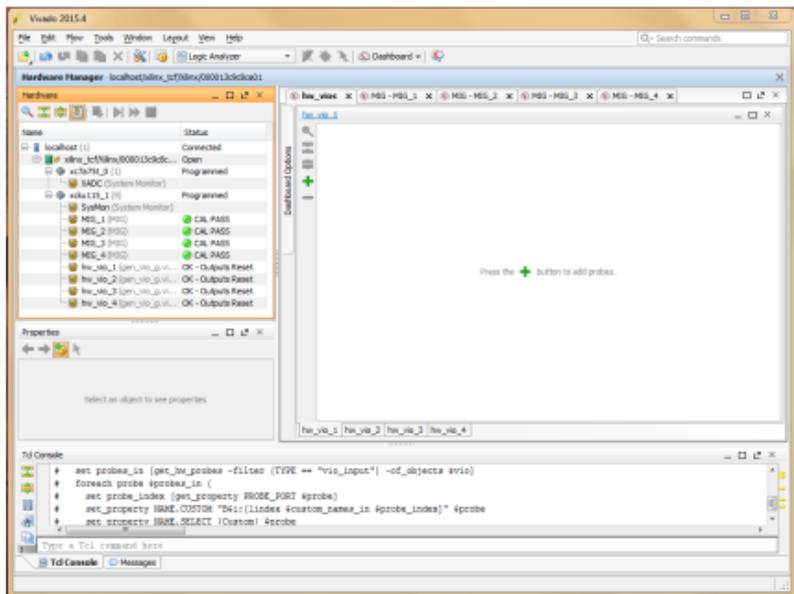


Figure 4 : Vivado Hardware Manager after debug Tcl script

Next, add the probes into the **hw_vios** tab. Note that the correspondence of **hw_vio_1**, **hw_vio_2**, **hw_vio_3** and **hw_vio_4** with the DDR4 SDRAMs may be reversed with respect to what might be expected. **hw_vio_1** may correspond to the fourth DDR3 SDRAM controller rather than the first, but it does not matter if this occurs.

Then, for each of **hw_vio_1**, **hw_vio_2**, **hw_vio_3** and **hw_vio_4**:

- Configure the **vio_reset** probe as **Active-High Button**. This can be clicked in order to reset the corresponding part of the FPGA design so that it performs calibration and memory test again.
- Configure the **init_calib_complete** and **ctl_done** probes as **LED** where **High Value Color** is **Green**.
- Configure the **ctl_error** probe as **LED** where **High Value Color** is **Red**.

At this point, Vivado Hardware Manager looks like this:

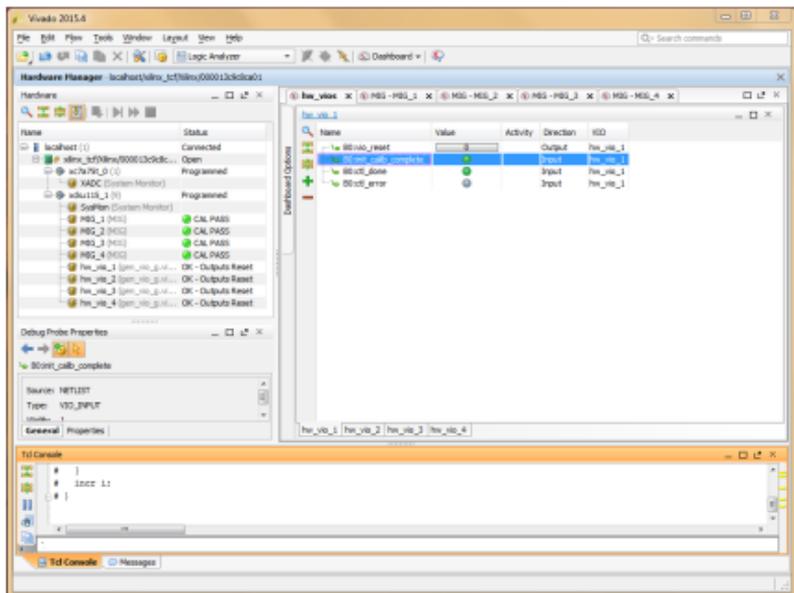


Figure 5 : Vivado Hardware Manager after configuring debug probes

It is likely that by the time the debug probes have been added to the **hw_vios** tab, the test that was initiated (automatically, when the FPGA was configured) has already completed. This situation is shown in Figure 5. To initiate another test, click the **B0:vio_reset**, **B1:vio_reset**, **B2:vio_reset** or **B3:vio_reset** virtual buttons.

Note

Alpha Data tests all ADM-XMC-KU1 reconfigurable computing cards before shipping to customers, so the **ctl_error** virtual LEDs should never be illuminated for any DDR4 SDRAM. The function of the debug probes and their usage in diagnosing a problem is currently outside the scope of this document, but may be added in a future version.

5 Simulating the Standalone DDR4 Test

Before simulating the Standalone DDR4 Test, the **Target simulator** and **Simulation set** must first be selected as desired, by selecting "Flow -> Simulation Settings" from the main menu:

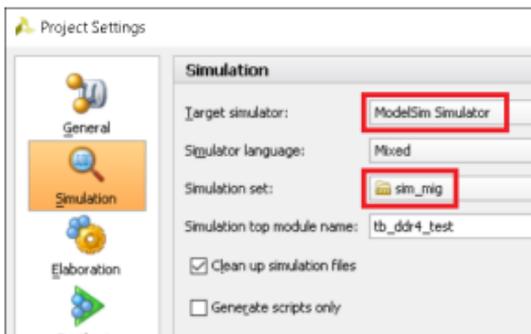


Figure 6 : Simulation settings

Target simulator may be any simulator supported by Vivado except for Vivado Simulator (XSIM). The reason for this is discussed in [Section 5.2](#) below.

To perform a simulation, use the **Run simulation** button in Vivado Flow Manager.

There are three possible ways for the simulation to terminate:

- If the memory test performed by the FPGA design finishes without data errors, the simulation outputs a message of the form **Simulation completed: PASSED** and terminates. This is the expected behaviour.
- If the memory test performed by the FPGA design finishes but detects data errors, the simulation outputs a message of the form **Simulation completed: FAILED** and terminates.
- If, for any reason, the memory test performed by the FPGA design does not finish after a fixed timeout period, the simulation outputs a message of the form **Simulation completed: TIMEOUT waiting for completion** and terminates.

5.1 Workaround for custom part simulation error in Vivado

When simulating the design and the **sim_mig** simset is active and the DDR4 SDRAM (MIG) IP uses a custom part definition, you may encounter an error during the compilation or elaboration phase relating to a signal **mcal_ADR**. This is caused by an issue in the DDR4 SDRAM (MIG) IP in Vivado 2015.4 to 2016.2 related to custom parts. To work around this issue, follow these steps:

- (1) Open the project in the Vivado GUI.
- (2) Select "Tools -> Run Tcl script..." from the main menu and select **(design)/fpga/proj/fix_custom_part.tcl**. This script unsets the properties on the DDR4 SDRAM (MIG) IP related to custom parts and then restores them. It then resets the IP's output products. This is sufficient to work around this issue.
- (3) **Without closing the project**, generate the output products of the DDR4 SDRAM (MIG) IP (or simulate the design, which also generates the output products). This time, no errors should occur.

This workaround must be applied each time that (a) the project is closed and reopened **and** (b) the DDR4 SDRAM (MIG) IP's output products must again be generated.

5.2 Vivado Simulator (XSIM) cannot be used to simulate this FPGA design

When attempting to simulate this FPGA design using the Vivado 2015.4 to 2016.2 Simulator (XSIM), an error of the form **ERROR: [VRFC 10-327] array size is larger than 2**31** is encountered.

This error relates to XSIM's handling of VHDL access data types. This issue may be resolved in a future version of Vivado.

Revision History

Date	Revision	Nature of change
22 June 2016	1.0	Initial version.