Defense contractor uses ScanWorks PFx tool to solve tough DDR testing problem for Zynq-7000 based design Case Study: In the Defense industry, product quality, reliability and longevity are key. Systems availability can have life and death impact every day in the theater. Particularly important is the stability and performance of memory devices on electronics systems. This Case Study involves hardware engineer at a defense contractor doing a new design for the U.S. Military. Due to the nature of military systems development, the branch or contractor cannot be mentioned, but the application and solution can be helpful to anyone doing a Zynq-7000 based design.

The problem the designer had was testing DDR memory on his board, as the memory was not operational; hence, the platform would not boot. His first step toward a solution was to pull out the 800+ page Zynq-7000 SoC Technical Reference Manual (TRM) and find the section on memory controller initialization. Upon reviewing the documentation, it told him to use Auto Calibration first; and if that did not work, then try Semi-Auto Calibration; and if that did not work, then the last option was Manual Calibration. After reading all this, the thought came to him that "Someone MUST have written software to do this!"; otherwise this was going to take a lot of time to implement the software for his own custom solution. His job was to get the design validated, not spend valuable time developing some proprietary tool to test the design.

After a Google search he came to the ASSET website and reviewed the ScanWorks product video on DDR Tuning and Test for Zynq-7000. After watching the video, and reviewing the website information, he convinced himself that "Someone HAS written software to do this!". He used the information from the web and his own credibility within his organization to convince management to make the tool investment. The good news was that he had found a commercially available supported product, which could root-cause his issue.

The essence of his problem was that the firmware could not read or write to the DDR memory on his board, and hence he could not run any tests. After acquiring the ScanWorks tool, and with about an hour of help from an ASSET application engineer, he was up and running with the tool. Through the use of the ScanWorks DDR Tuning and Test tool he was able to find that bits on the board were swapped and after further investigation a layout wiring error was the root cause. Fixing the layout problem ultimately solved his memory testing issue. The result was in less than a day the design problem was identified, and the layout change to fix the signal routing scheduled.

He saved valuable time and the tool paid for itself by getting the project back on schedule. Thus, he stayed focused on the value he adds – developing the design functionality.



Ramping Up on the New Tool Quickly

Figure 1 RIC-1000 and Zedboard

Here is the sequence of events that the engineer and ASSET went through to identify root cause and resolve the issue.

To establish a working baseline, an ASSET application engineer (AE), had the customer connect to a known working target (Zedboard) shown in Figure 1 and load the built-in example project provided within the tool structure. Next the AE walked the customer through the board setup and specific configuration settings to enable the Tuning and Testing of the Zedboard. These included setting the DDR Configuration to Automatic, and the parameters for the DDR settings (DDR Frequency, DDR clock period, Number of Chip Selects, etc.) to their specific values that are provided and shown in Figure 2.



Figure 2 DDR board settings



Figure 3 DDR Memory model

The learning continued as they examined the DDR3 Memory part parameters that needed to be configured (Figure 3). The data is provided from the memory device specification or from a library within the tool. For the Zedboard it was a simple matter of pointing to the Micron device within the library. If the model is not in the library, the AE (or customer) can configure one. Generally, this is accomplished by starting with a similar device, and making only the necessary changes (as few as possible to save time). With that information, it was now possible to configure the DDR controller. The DDR PHY initialization was accomplished via a simple pull-down menu, which loaded the proper registers within the DDR Memory controller as shown in Figure 4 and the action result in shown in Figure 5



Figure 4 Configure DDR Controller and device initialization



Figure 5 DDR PHY Configuration Complete

The next steps involved running the interactive tests and discussing the differences between the structural tests and the more sophisticated types of tests provided. These steps are necessary to ensure that the DDR PHY was correctly configured as shown in Figure 6.



Figure 6 DDR PHY and memory test

Pix Developer (1.2.0.0) (DDR3 Test and Tune)		
le Tools View		Hel
Add Disconnect Run		
*		CONNECT DDR TFTP
et	Cuttore Controle	
Recipite 0000 Test and Test	Sinh make	
Piget burs fet are fulle		
- D Fature: [1]	Write DDS/Write leveling Measurement results:	
ddl Reture	Byte 0: center 146, window (99-193) 94/256ths clock	
🗄 🔓 Taska	Byte 2: center 140, window (30-100) 100/256ths clock	
Task: PFx DOR	Byte 3: center 145, window (92-198) 106/255ths clock	
-@ cldf Task>	Marile Date New Merson and Area late	
	Brte 0: center 133. window (145-241) 56/256ths clock	
	Byte 1: center 191, window (144-238) 94/256ths clock	
	Byte 2: center 191, window (142-241) 99/256ths clock	
	afte 3: Genter 193, Window (140-24)) LU//Zacine clock	
	Read DQS Gate Delay Measurement results:	
	Byte 0: center 309, window (136-482) 346/256ths clock	
	Byte 1: center 311, window (134-488) 354/256ths clock Byte 2: center 283, window (109-458) 349/256ths clock	
Gonfiguration (PFx DDR)	 Byte 3: center 209, window (116-463) 347/256ths clock 	
E-Im Board		
Board Initialization	Rete O: center 70, window (20-121) 101/256ths clock	
H Copts	Byte 1: center 69, window (13-128) 112/286ths clock	
H-122 DDR Setings	Byte 2: center 68, window (15-122) 107/256ths clock	
- III Board Specific Zing7000	Byte 3: center 71, window (20-122) 101/156ths clock	
DDR3 Memory Part:micron_ddr3J_MT41K128M16-125.ksc	Updated Board Specific Symq7000 Settings	
0-0 DOR Ture	0 -> \$ddr.memotrl.auto_training	
- B DOR PHY Training	0x00000046 -> Sphy_rd dogs_cfg0 // VHY Need Data Eye: Byte0 0x00000045 -> Solw wd dogs_ofot // DHY Read Data Eye: Byte1	
	0x00000044 -> \$phy rd dgs ofg2 // PHY Read Data Eye: Byte2	
- S DDR Waking 0 Test	Ox00000047 -> \$phy_rd_dqm_ofg3 // PHY Read Data Eye: Byte3	
DOR Address Test	 Ox0000002 -> Sphy_wr_dgs_cfg0 // PHY Write DIS delay: Byte0 Ox00000000 -> Sobs.wr.dow.ofs1 // DHY Mrite DDS delay: Bote1 	
B DDR Smultaneously Switching Output Test	0x0000008C -> \$phy wr dga ofg2 // VEY Write DQS delay: Byte2	
- 🚱 DDR Noise Bunt Test	Ox00000091 -> \$phy_wr_dqm_ofg3 // PHY Write DQS delay: Byte3	
- B DOR Noise Test	DECODDIDIS -> Sphy_we_cfg0 // PWT Read DDS Gate delay: Byte0 DeCODDIDIST -> Subr we_cfg1 // DWT Read DDS Gate delay: Byte1	
- 159 DUH Hai Bant Test	0x0000011B -> \$phy_we_cfq2 // VHY Read DQS Gate delay: Byte2	
- 19 DOR Full Word Test	Ox00000121 -> \$phy_we_ofg3 // PEY Read DQS Gate delay: Byte3	
- 🕲 DOR Snutural Test	0x000000Cl -> Swr_data_siv0 // PBT Write Data Eye: Byte0 0x00000000F -> Swr_data_siv1 // PBT Write Data Eye: Byte1	
- 🔞 DDR Noize Group Test	OxCOCOCODEF -> Sur_data_siv2 // FEY Write Data Eye: Byte2	
- 🚱 DOR Comprehensive Test	On000000C1 -> Swr data alv3 // VEY Write Data Eye: Byte3	
- Group DDR Constrained Random Test	0x00045412 -> Solvy Luit ratio(// PHT_INIT_RATIO) 0x00045400 -> Solvy init ratio(// PHT_INIT_RATIO)	
Gy DUR Performance Test - Access Time	OnOODINCIC -> \$phy_init_ratio2 // PHY_INIT_RAVID2	
- DDR Complexity Test	On00040411 -> \$phy_init_ratio3 // VHY_INIT_RATIO3	
- 19 DDR Performance Test - Access Time	· Bight-clock on the 'NDE Sume' under and calent "Debits DDE Settions from Suming Results" to sume these charges	to more project
	[PASSID]	
py Delete Up Down		
ct CiScanWorks/Projects/PFx_Zedboard_Example_Project/Zedboard/DDR3 Test and	Tune/DDR3 Test and Tune/DDR3 Test and Tune project (Zyrrq-7010 Z-7020) (PFx DDR)	Elapsed: 000:14:17:

The final steps involved running the tuning to get the correct values needed and then copying the tuning results to the project and retesting. Throughout this process, as the turning results were being displayed in the Console view, a discussion ensued about the data provided by the target agent running in the On-Chip Memory (OCM). All of this was accomplished to establish a baseline of understanding between the customer and the applications engineer, and to exchange target and tool knowledge.

Solving the Problem

With the baseline of knowledge established, they switched over to the customer's board and loaded the project. The customer and applications engineer examined the project parameter settings and were able to verify all the necessary setup parameters were correct. The next step was to connect to the customer target, thereby downloading the DDR Tune and Test firmware into the OCM. Note that this is done completely out-of-band via JTAG, eliminating the need for booting the board entirely. Next began the DDR PHY initialization. After successful completion, structural tests (Walking 0s, Walking 1s, and Address test) were executed. These tests failed. With the failure data provided by the tool the team was able to examine the schematics and pinpoint the root cause of the problem, which as previously mentioned was a layout issue.

Summary

This case study demonstrates how the ScanWorks DDR Tuning and Test tool can help validate designs using Xilinx Zynq-7000 devices. The tool allows the designer to focus on creating and implementing a great design without spending time writing custom software to test the design.