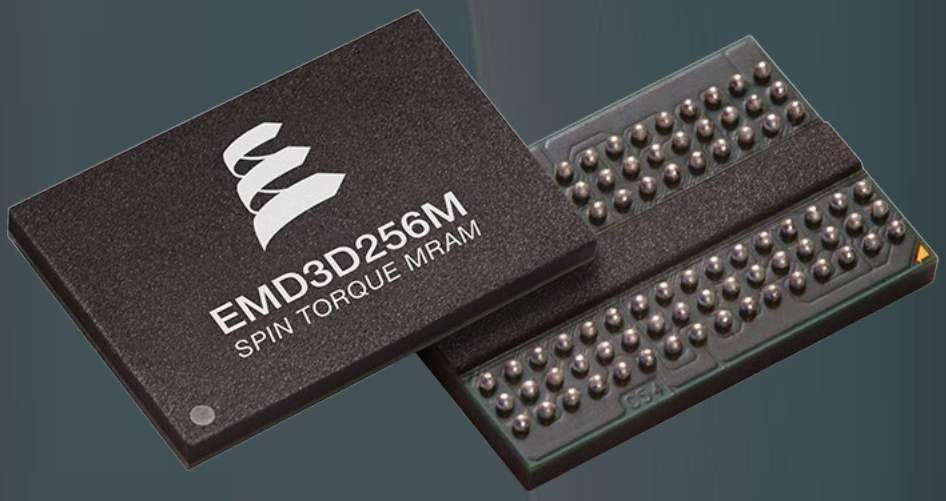
EVERSPIN® TECHNOLOGIES

App Note

ST-DDR3 Design Guide For Xilinx FPGA Controllers



Introduction

Spin-transfer Torque Magnetoresistive Random Access Memory (STT-MRAM) is a technology that delivers performance, persistence and durability as a DDR3-like memory called ST-DDR3.

With an interface that is designed around JEDEC standards, systems can utilize STT-MRAM in their designs with the described modifications to the memory controller to comprehend the persistence of STT-MRAM.

This document will help engineers understand how to enable a Xilinx FPGA memory controller to communicate with persistent ST-DDR3 memory.

Enabling ST-DDR3

ST-DDR3 is STT-MRAM with a DDR3 interface. This means that ST-DDR3 is persistent and the designer needs to comprehend what persistent memory means and how it differs from traditional, volatile DDR3 memory. The entire process starts with a known good 4Gb DDR3 SDRAM-1333 Memory Interface Generator (MIG) that is generated from the Xilinx Vivado development environment. The primary deviations from the 4Gb DDR3 SDRAM controller are:

1. Timing (increase row access timing, increase counter widths and reduce CAS page sizes)
2. Power-up (calibration – anti-scribble mode enabled during calibration)
3. Power-down (scramming or moving all relevant data into the persistent memory array)
4. Performance (increase pipeline depth and increase data transfer efficiency)

Note: Also required for a robust ST-DDR3 persistent memory design, but outside the scope of this document is implementation of a Double Bit Error Correction (DEC) scheme. Details to follow in a subsequent ECC specific Application Note.

DDR3 SDRAM-1333 Memory Interface

In the Xilinx design environment, the DDR3 interface logic will be generated based upon input parameters that represent the speed and timing characteristics of a 4Gb SDRAM DDR3-1333. Since the MIG cannot create interface logic using parameters outside of the current JEDEC standard, a JEDEC compatible DDR3 controller must be created as a preliminary first step. Since the Everspin 256Mb ST-DDR3 1333 device most closely resembles a 4Gb DDR3-1333 SDRAM device, use the timing values from the 4Gb DDR3 SDRAM 1333 spec SDRAM DDR3L-1333 (Table 8, Timing Parameters Used for IDD Measurements – Clock Units, please reference the -15 part only). If the -15 device is not available, choose the -15E device and ensure CL = 10. Once the DDR3 interface logic has been created, the timing, power-up, power-down and performance parameters can be modified to enable ST-DDR3 persistent memory.

It is highly recommended after creating a MIG, an example testbench be created in Vivado by right clicking on the .xci file, and selecting the menu item called "Open IP Example Design...". Creating an example design will create a new Vivado project with all the test files required to simulate your newly created MIG. See the Xilinx MIG creation tutorial Designing a Memory Interface and Controller with Vivado MIG for UltraScale and the Memory Interfaces Design Hub - UltraScale DDR3/DDR4 Memory.

Note: All MIG creation and changes were performed using Vivado 2017.2 and Vivado 2018.1.

Xilinx FPGA Controller Modifications

The changes required to the existing Xilinx MIG DDR3 controller to enable ST-DDR3 operation as mentioned earlier are categorized as Timing, Power-up, Power-down and Performance modifications.

Timing

Table 1 lists the key timing parameters for a 4Gb SDRAM-1333 device and its associated changes required to enable an ST-DDR3-1333 STT-MRAM device. Use Table 1 as a check to ensure all timing values listed in the ST-DDR3 column match all timing parameters in the modified ST-DDR3 MIG.

Table 1 - Key timing parameters for DDR3 and ST-DDR3

	Symbol	DDR3 DRAM		ST-DDR3 STT-MRAM	
		ns (min)	ck (min)	ns (min)	ck (min)
Clock Period	tCK	1.5		1.5	
Cas Latency	CL		10		10
Cas Write Latency	CWL		7		7
Column to Column command delay	tCCD		4		4
Internal READ to first data	tAA	15	10	14	10
ACTIVE to internal READ or WRITE delay time	tRCD	15	10	95	64
Precharge command period	tRP	15	10	66	44
ACTIVE to ACTIVE command period	tRC	51	34	170	114
ACTIVE to Precharge command period	tRAS	36	24	103	69
Write Recovery, WRITE to Precharge delay time	tWR	15	10	15	10
ACT to ACT Command Period, different banks	tRRD	6	4	30	20
Four ACTIVE Window	tFAW	30	20	120	80
REFRESH to ACT command delay (1Gb to 8Gb)	tRFC		74 – 234		Not Used

Column width and Counter differences

Table 1 above lists all key timing changes and Table 2 lists the corresponding column and counter width differences.

Table 2 - Column width and counter differences for ST-DDR3

Parameter (bits)	JEDEC DDR3	256Mb ST-DDR3
IO Width	x8	x8
Page size	8192	512
tRASf	3	6
TXN_FIFO_DEPTH	4	8
TXN_FIFO_PWIDTH	2	3
CAS_FIFO_DEPTH	4	8
CAS_FIFO_PWIDTH	2	3
trcd_cntr / trcd_cntr_nxt	4	6
trp_cntr	5	7
tras_cntr_rb / tras_cntr_rb_nxt	4	6
Column Address Width (bits)	$A_0 - A_9$ (10)	$A_0 - A_5$ (6)

Making the CAS page sizes smaller and increasing the counter widths has performance implications that will be explained briefly in the performance section below.

Power-up

The detailed Power-up sequence for DDR memories can be seen in the Technical Note from Micron titled TN-46-08: Initialization Sequence for DDR SDRAM . During Power Up this particular sequence of 20 steps needs to be processed in the documented order to ensure proper operation with the following exceptions:

1. The Refresh command is not used for ST-DDR3. Steps 15 & 17 are Auto Refresh commands the STT-MRAM devices will simply ignore.
2. During the calibration step, which is after the initialization steps, put the ST-DDR3 device into NOMEM mode via Mode Register 2 or MR2[8] = 1 prior to calibration and clear post calibration MR2[8] = 0 to prevent data corruption during write leveling. This is also referred to as Anti-scribbling. Since ST-DDR3 is persistent memory, writing to locations during calibration might over-write known good data, so this feature needs to be disabled during the calibration phase.

Mode Register Settings

Full Mode Register compatibility is supported and should be set as follows during power-up initialization or after a reset for proper ST-DDR3 1333 operation. Mode Register programming should be performed in the following sequence:

1. Mode Register 2 (MR2) – 0x0110 MR2[8] = 1
2. Mode Register 3 (MR3) – 0x0000
3. Mode Register 1 (MR1) – 0x0044
4. Mode Register 0 (MR0) – 0x0b60
5. After Calibration and before normal operation
6. Mode Register 2 (MR2) – 0x0010 MR2[8] = 0

Note: MR2[8] = 1 is NOMEM or anti-scribbling mode enabled to prevent over-writing known good data in the persistent memory array. MR2[8] = 0 disables NO-MEM mode prior to writing to the persistent memory array.

Power-down (Scram)

Scram is not a command, or an opcode. Scram is a power down procedure. It literally means it's time to leave quickly and in this case, power is going away and the controller needs to guarantee persistence of all open pages, buffers, DRAM contents and write them to the persistent array as quickly as possible. To guarantee persistence, executing a Precharge (PRE) or Precharge All (PREA) command must always be performed to move data into the persistent memory array.

In some designs that use both DRAM/MRAM and the DRAM capacity exceeds the size of the MRAM capacity, the controller needs to always guarantee (during normal operation) that important data that requires persistence and residing in DRAM never exceeds the size of the MRAM array. If the above condition is met, a normal power down sequence should not take any longer than 10 microseconds for most situations but is design dependent and should be calculated, simulated and measured to guarantee all important data is written to the persistent memory array by executing a

Precharge(PRE) or Precharge All (PREA) command. Below is the scramming procedure:

1. The controller has been told to shut down or detects that the input supply voltage (+12Vdc) is either slumping, over-voltage, over-current or an over temperature event has occurred. The DC-to-DC output bulk capacitance needs to be sufficient to keep the FPGA and MRAM and/or DRAM running long enough to finish the next 4 steps. Most designs will easily meet the uptime requirement by meeting the bulk capacitance transient current requirements but should not be assumed. See the Board Level Checklist section below.
2. Finish all pending MRAM accesses at MIG top level. The control logic finishes delivering all data and status information to the MRAM MIG interface.
3. Set **power_fail_has_scramed** MRAM MIG input signal, and hold asserted. This tells the MIG that the control logic has finished and that the MIG should also scram, pushing all writes to the MRAM persistent array. It should also disable periodic reads. The control logic should not do reads and writes to the MRAM MIG while **power_fail_has_scramed** is asserted.
4. **inflight_writes** is just a status output signal. After writes are done, the MIG executes a Precharge All (PREA) command to close all open pages and store the data in the persistent memory array.
5. Wait for **ddr3_cntr_power_fail_complete** output signal. This indicates that all pending writes the MIG had are in the MRAM persistent memory and all pages are closed. Continue to hold **power_fail_has_scramed** to the MRAM MIG input signal as it also disables periodic reads.
6. It is safe to power off MRAM without losing data.
7. If the control logic decides to start-up again (ex. after a power glitch) without an actual power-up reboot, the control logic can de-assert **power_fail_has_scramed** and the MIG will clear **ddr3_cntr_power_fail_complete**. Periodic reads will be re-enabled, and reads and writes can start again.

Performance

The following is a list of STT-MRAM optimizations to increase system performance:

1. Disable or greatly extend the refresh interval (tREFI) since ST-DDR3 does not require Refresh.
2. Set address ordering/mapping for highest application performance and lowest wear to ROW-BANK-COL.
3. Increase command / data queue depths to enable look ahead activate / Precharge to manage the longer row access latencies and smaller CAS page sizes of STT-MRAM.
4. Achieving high bus utilization by accessing data on every clock tick.

DDR3 MIG Changes

Table 3 below summarizes the MIG changes for each category and lists which modules within the DDR3 MIG controller for a XCKU060-2FFVA1156E that have been affected. Everspin has supplied a Linux diff output in the form of a patch file corresponding to each of the eleven affected modules. Making all changes to each module will enable all timing, power-up, power-down and performance features documented above.

Table 3 - Modified parameters and IP modules affected

Category	STT-MRAM Timing Parameters and Performance Changes	MRAM_ddr3_v2_0.sv	MRAM_ddr3_v2_0_ddr3.sv	MRAM_ddr3_v2_0_ddr3_mem_infc.sv	ddr3_v1_4_cal.sv	ddr3_v1_4_mc.sv	ddr3_v1_4_mc_arb_mux_p.sv	ddr3_v1_4_mc_group.sv	ddr3_v1_4_mc_ref.sv	ddr3_v1_4_ui.sv	ddr3_v1_4_ui_rd_data.sv	ddr3_v1_4_ui_wr_data.sv
	File Number	1	2	3	4	5	6	7	8	9	10	11
Timing	Timing settings and counter width changes		x			x	x	x				
Power-up	Anti-scribbling changes (NO-MEM mode)				x							
Power-down	SCRAM input signal to drain writes with CAS page closes	x	x	x		x			x			
Power-down	Created SCRAM output status signals	x	x	x		x		x		x		
Performance	Auto pre-charges on the 8th BL8 of a CAS page		x			x				x	x	x
Performance	FIFO-DEPTH doubled							x				
Performance	Changed to emit requests faster							x				

DDR3 MIG diff output files

There are eleven patch files that can be downloaded [from the Everspin website](). Each file listed is a Linux diff output file that was created by comparing an 4Gb DDR3 MIG against a 256Mb ST-DDR3 created MIG and saved with a .patch extension. The patch files can be applied as a manual recipe the engineer can follow to make appropriate modifications or use the Linux patch command to update the existing modules to create output files that creates a new ST-DDR3 compatible MIG controller module. A suggested design process would be:

1. Create a 4Gb DDR3 compatible MIG in Vivado.
2. Create a simulated test bench by right clicking on the .xci file and selecting the Vivado menu item called "Open IP Example Design...", Vivado creates a new project directorywith the "_0_ex" appended to the end of the project name. The new project directory will look something like the following:

   `/<home directory>/<user defined MIG name>_0_ex`

   `/<user defined MIG name>_0_ex.srcs/sources_1/ip/<user defined MIG name>/rtl`
3. Simulate using the Xilinx generated testbench to ensure the 4Gb DDR3 MIG is operational
4. Run the 11 patch files
5. Re-Synthesize and Re-simulate your environment to ensure the 256Mb STT-DDR3 MIG is operational

Under the rtl subdirectory please find 5 subdirectories titled:

`cal clocking controller ip_top ui`

Move patch files 1-3 into **ip_top**, move patch file 4 into **cal**, move patch files 5-8 into **controller** and patch files 9-11 into **ui**.

Run each patch against their DDR3 MIG equivalents. The newly created output file is a new ST-DDR3 compatible module.

Use the linux patch command in the following manner:

```
$patch [original file] -i [patch file] -o [output file]
```

The original file corresponds to the original DDR3 MIG module file. All patch files are listed below:

1-ddr3_0.sv.patch

2-ddr3_0_ddr3.sv.patch

3-ddr3_mem_intfc.sv.patch

4-ddr3_v1_4_cal.sv.patch

5-ddr3_v1_4_mc.sv.patch

6-ddr3_v1_4_mc_arb_mux_p.sv.patch

7-ddr3_v1_4_mc_group.sv.patch

8-ddr3_v1_4_mc_ref.sv.patch

9-ddr3_v1_4_ui.sv.patch

10-ddr3_v1_4_ui_rd_data.sv.patch

11-ddr3_v1_4_ui_wr_data.sv.patch

Each patch file corresponds to a module listed in Table 3 above. Once the patch command has been successfully completed, the output file needs to be the same name as the original corresponding DDR3 MIG file name. One suggested process to ensure each file is safely backed up before modifying any files could use the following process:

```
$cp filename.sv filename.sv.org

$patch filename.sv.org
      -i filename.sv.patch
      -o filename.sv.save


$cp filename.sv.save filename.sv
```

If the patch command does not successfully complete, making the edits manually may be required. Please note, this process is not guaranteed and requires proper simulation of all changes.

nvNITRO Schematic files

nvNITRO is an Everspin-developed reference platform that was used to create an STT-MRAM based PCIe Gen3 x8 PCIe plugin card. The Xilinx controller is a XCKU060-2FFVA1156E FPGA, connecting 36 STT-MRAM devices, using 4 ranks for a total of 1GB of persistent memory. The Orcad design files are available per user request from Everspin.

nvNITRO Schematics

The nvNITRO reference schematics in PDF format are available per user request from Everspin.

Collateral items

The following collateral items are available:

Accessible by download

- MIG Patch files

Accessible by request from Everspin:

- Orcad schematic files for nvNITRO
- nvNITRO schematics in PDF format
- Allegro viewable board files

Board Level Checklist

The following is a list of board level items the design engineer needs to consider to ensure a successful design.

1. Follow established high-speed routing and hardware design guidelines from the PCI-Sig, JEDEC, Xilinx and Micron for PCIe and DDR3 based designs.
2. The same high speed signaling layout guidelines and best practices that govern DDR3, also apply to ST-DDR3. Please see DDR3 SDRAM Unbuffered DIMM Design Specification (JEDEC login required). This same specification indicates the amount of bulk and decoupling capacitance required per device (See Table 12 of the DDR3 Unbuffered DIMM Design Specification).
3. Xilinx provides a PCB high speed design guideline that also includes the amount of decoupling and bulk capacitance required per device (see Table 1-12 in the UltraScale Architecture PCB User Guide). Xilinx also provides PCB high speed design guides for other families of devices too.
4. For STT-DDR3 power requirements (see Table 10 in the ST-DDR3 256Mb specification https://www.everspin.com/file/156503/download).
5. For Xilinx FPGA power requirements, see the power estimator Xilinx Power Estimator (XPE) to estimate worst case power usage.
6. Schematic capture and FPGA pin assignments using the Xilinx Vivado development tool should be done iteratively to guarantee proper FPGA functionality and external signal routability.
7. As mentioned in the power-down section at the beginning of this document, a proper scramming sequence should take no longer than 10µs. Meeting all bulk decoupling requirements for transient load changes will almost always be much more bulk capacitance needed to maintain a hold time beyond 10µs. In most cases this will be in the 10's of ms range. This should not be assumed and should not replace calculating, simulating and measuring the amount of hold time required for each design.
8. VDDQ, VDD, VrefDQ, VrefCA, and ZQ for ST-DDR3 are the same as the DDR3 1.5V specification (see Table 4 below).
9. Use the tables on page 2 and 3 as a check to ensure all timing values listed in the ST-DDR3 column match all key timing parameters in the modified ST-DDR3 MIG.

Table 4 - Input voltages for ST-DDR3

Parameter	JEDEC DDR3	256Mb ST-DDR3
VDDQ, VDD, Vrefdq, Vrefca, ZQ	1.5V, 1.5V, VDDQ/2, VDDQ/2, VSSQ	1.5V, 1.5V, VDDQ/2, VDDQ/2, VSSQ

Conclusion

Everspin's STT-MRAM provides a superior alternative to DRAM-based controller architectures to solve I/O determinism challenges, enable enterprise-class performance and reliability without the need for alternate energy sources like batteries/supercapacitors, and deliver byte addressable persistent memory. STT-MRAM enables designers to optimize footprint, performance, endurance, retention, and reliability while at the same time reducing complexity and enabling advanced functionality.

Contact Information:

Author:
James Singer
Director, Applications Engineering

How to Reach Us:
www.everspin.com

E-Mail:
support@everspin.com
orders@everspin.com
sales@everspin.com

USA/Canada/South and Central America
Everspin Technologies
5670 W. Chandler Road, Suite 100
Chandler, Arizona 85226
+1-877-347-MRAM (6726)
+1-480-347-1111

Europe, Middle East and Africa
support.europe@everspin.com

Japan
support.japan@everspin.com

Asia Pacific
support.asia@everspin.com

Everspin Technologies, Inc.