DFI

DDR PHY Interface

DFI 4.0 Specification

APRIL 27, 2018

Release Information

Rev #	Date	Change
1.0	30 Jan 2007	Initial Release
2.0	17 Jul 2007	Modifications/Additions for DDR3 Support
2.0	21 Nov 2007	Additional modifications/additions for DDR3 support. Added read and write leveling. Changes approved by the Technical Committee for DDR3 support.
2.0	21 Dec 2007	Removed references to data eye training for PHY Evaluation mode, added a gate training-specific mode signal, corrected references and clarified read training.
2.0	11 Jan 2008	Modified wording; standardized notations in figures, clarified terminology for read and write leveling.
2.0	26 Mar 2008	Added timing parameter t_{rdlvl_en} and t_{wrlvl_en}, signal dfi_rdlvl_edge.
2.1	2 Oct 2008	Added initial LPDDR2 support and corrected minor errors from 2.0 release.
2.1	24 Nov 2008	Added frequency change protocol, signal timing definitions, t_{rdlvl_load} and t_{wrlvl_load} timing parameters and adjusted diagrams accordingly.
2.1	30 Jan 2009	Added DFI logo.
2.1	31 Mar 2009	Updated width of dfi_rdlvl_edge, corrected erroneous figures, updated t_{rdlvl_en} and t_{wrlvl_en} definitions.
2.1	20 May 2009	Added low power control interface, modified leveling request signal description to include frequency change, added dfi_data_byte_disable signal and t_{phy_wrdata} timing parameters. Added DIMM support to the status interface and updated frequency ratios from an example to a defined method. Updated frequency ratios information for new proposals, modified default values and requirements for some training interface signals, incorporated LPDDR2 training operations changes
2.1	22 Jun 2009	Expanded frequency ratio information to include vectored read data, expanded use of dfi_init_start, added timing diagrams for 1:4 frequency ratio systems
2.1.1	23 Mar 2010	Added reference to the parity interface to the Overview. Changed dfi_parity_in signal to have a phase index. Modified description of dfi_freq_ratio signal to make it optional except for MCs/PHYs that support multiple frequency ratios. Expanded figure 32 into two figures to represent odd and even timing parameters.
2.1.1	01 Apr 2010	Changed minimum value for t_{lp_wakeup}.
2.1.1	20 Apr 2010	Corrected figure 3 timing violation. Corrected erroneous sentence for 2T timing. Corrected figure 35 t_{phy_wrlat} timing. Correct incorrect references to t_{phy_wrlat} in frequency ratio read examples.
2.1.1	27 May 2010	Added Figure 4 and text to explain differences between Figure 3 and 4.
2.1.1	09 Jun 2010	Modified text in dfi_init_start and surrounding figures 3 and 4 for more clarity.
3.0	21 May 2012	Added DDR4 DRAM support for: CRC, CA parity timing, CRC and CA parity errors, DBI, leveling support, and CA modifications. Added DFI read data rotation clarification, read data pointer resynchronization, independent timing of DFI read data valid per data slice, data path chip select, error interface, and programmable parameters. Renamed PHY evaluation mode. Removed MC evaluation mode and $\mathbf{t_{phy\_wrdelay}}$ timing parameter. Added support for refresh during training, multiple CS training, enhancements to the update interface and the idle bus definition
3.1	19 May 2012	Added support for LPDDR3. Enhanced the Low Power Control Interface to have separate control and data requests. Added the PHY-Requested Training Interface to enable PHY-independent training in non-DFI training mode.
--	14 Nov 2013	Synchronized book files to 3.1 in advance of upcoming changes from JM.

--	21 Nov 2013	Incorporated review corrections.
--	21 Mar 2014	Incorporated committee comments, corrected erroneous cross references, fine-tuned formatting, fine-tuned typographical items.
4.0	04 Aug 2017	Merged DFI 4.0 Spec Addendum to DFI 3.1. Added support for LPDDR4, DB training, per-slice read leveling, DFI read/write chip select, write DQ training, PHY master interface, frequency indicator, DFI disconnect protocol, DFI data bit disabling, slice parameter, geardown mode, DFI feature and matrix topology matrix, 3D stack support and inactive CS support. Also modified CA training, write leveling strobe and changed the DFI training to be optional. Enhanced DFI read data eye training sequence, update interface for self-refresh exit
	20 Jul 2017	Incorporated review changes from 4.0 Addendum merge.

Proprietary Notice

No part of this document may be copied or reproduced in any form or by any means without prior written consent of Cadence.

Cadence makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability or fitness for a particular purpose. Information in this document is subject to change without notice. Cadence assumes no responsibility for any errors that may appear in this document.

Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy, or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraphs (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Destination Control Statement

All technical data contained in this product is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Trademarks

Cadence and the Cadence logo are registered trademarks of Cadence Design Systems, Inc.

All other products or brand names mentioned are trademarks or registered trademarks of their respective holders.

End User License Agreement2

1.Subject to the provisions of Clauses 2, 3, 4, 5 and 6, Cadence hereby grants to licensee ("Licensee") a perpetual, nonexclusive, nontransferable, royalty free, worldwide copyright license to use and copy the DFI (DDR PHY Interface) specification (the "DFI Specification") for the purpose of developing, having developed, manufacturing, having manufactured, offering to sell, selling, supplying or otherwise distributing products which comply with the DFI Specification.

2.THE DFI SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES EXPRESS, IMPLIED OR STATUTORY, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF SATISFACTORY QUALITY, MERCHANTABILITY, NONINFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE.

3.No license, express, implied or otherwise, is granted to Licensee, under the provisions of Clause 1, to use Cadence's or any other person or entity participating in the development of the DFI Specification listed herein (individually

"Participant," collectively "Participants") trade name, or trademarks in connection with the DFI Specification or any products based thereon. Nothing in Clause 1 shall be construed as authority for Licensee to make any representations on behalf of Cadence or the other Participants in respect of the DFI Specification.

4.NOTWITHSTANDING ANYTHING ELSE WILL CADENCE'S TOTAL AGGREGATE LIABILITY FOR ANY CLAIM, SUIT, PROCEEDING OR OTHERWISE, RELATING IN ANYWAY TO THE DFI SPECIFICATION EXCEED $1.00USD.

5.NOTWITHSTANDING ANYTHING ELSE WILL ANY PARTICIPANT'S TOTAL AGGREGATE LIABILITY FOR ANY CLAIM, SUIT, PROCEEDING OR OTHERWISE, RELATING IN ANYWAY TO THE DFI SPECIFICATION EXCEED $1.00USD.

6.Licensee agrees that Cadence and the Participants may use, copy, modify, reproduce and distribute any written comments or suggestions ("Communications") provided regarding the DFI Specification by Licensee and that Licensee will not claim any proprietary rights in the DFI Specification, or implementations thereof by any Participant or third party, as a result of the use of the Communications in developing or changing the DFI Specification. Cadence and the participants will have no confidentiality obligations with respect to the Communications and Licensee will not include any confidential information of Licensee or any third party in any Communications.

Participants

ARM Cadence Intel LSI Samsung ST Synopsys

Contents

List of Tables

List of Figures

1.0 Overview

The DDR PHY Interface (DFI) is an interface protocol that defines the signals, timing parameters and programmable parameters required to transfer control information and data across the DFI and between the MC and the PHY. The programmable parameters are options defined by the MC, PHY, or system and programmed into the MC and/or the PHY. DFI applies to: DDR1, DDR2, DDR3, DDR4, DDR4 RDIMM, DDR4 LRDIMM, LPDDR1, LPDDR2, LPDDR3, LPDDR4 DRAMs.

The DFI protocol does not encompass all of the features of the MC or the PHY, nor does the protocol put any restrictions on how the MC or the PHY interface to other aspects of the system.

This specification is organized by the interface groups listed in Table 1, “Interface Groups”.

TABLE 1. *Interface Groups*

Interface Group	Description
Control	Required to drive the address, command and control signals to the DRAM devices.
Write Data	Used to send write and receive valid read data across the DFI.
Read Data	
Update	Provides an ability for the MC or the PHY to initiate idling the DFI bus.
Status	Used for system initialization, feature support and to control the presence of valid clocks to the DRAM interface.
Training	Used to execute gate training, read data eye training, write leveling and CA training operations.
Low Power Control	Allows the PHY to enter power-saving modes.
Error	Used to communicate error information from the PHY to the MC.
Data Buffer Training	Used to execute DB-to-DRAM training.
PHY Master	Used to allow the PHY to control the DFI bus.
Disconnect Protocol	Allows an ongoing handshake to be broken.
Geardown	Uses the geardown mode function of the DRAM

Within each interface group are signals and parameters. Some signals are applicable only to certain DRAM types. All of the DFI signals must use the corresponding parameters.

Changes in the DFI protocol between versions may result in incompatibilities between MCs and PHYs which were designed to adhere to different versions of the DFI specification. If using devices designed for different versions of the DFI specification, review the changes associated with the corresponding versions and ensure changes will not interfere with interoperability in a specific configuration.

All figures are provided for illustrative purposes only. Timing diagrams may illustrate condensed or otherwise unrealistic signal timing.

A glossary of terminology used in this specification can be found in Table 48, “Glossary of Terms”.

2.0 Architecture

The DFI specification requires a DFI clock and DFI-defined signals that must be driven directly by registers referenced to a rising edge of the DFI clock. There are no rules dictating the source of the DFI clock, nor are there restrictions on how the DFI-defined signals are received. For DFI interoperability between the MC and the PHY, ensure compatibility in:

- Signal widths
- Interconnect timing
- Timing parameters
- Frequency ratio
- Function

Interconnect timing compatibility between the MC and the PHY at target frequencies is determined by the specification of the output timing for signals driven and the setup and hold requirements to receive these signals on the DFI per device, as defined by the device.

The DFI specification does not dictate absolute latencies or a fixed range of values that must be supported by each device. Certain DFI timing parameters can be specified as fixed values, maximum values, or as constants based on other values in the system.

DFI timing parameters must be held constant while commands are being executed on the DFI bus; however, if necessary, DFI timing parameters may be changed during a frequency change or while the bus is in the idle state. For more information on timing, refer to Section 5.0, “Signal Timing”.

The DFI specification identifies the DFI signals relevant to a specific implementation based upon support for specific DRAM device(s), optional features and frequency ratio. For more information on which signals are relevant to a specific implementation, refer to Table 3, “DFI Signal Requirements”.

The MC and the PHY must operate at a common frequency ratio. For matched frequency systems, the DFI write data bus width is generally twice the width of the DRAM data bus. For frequency ratio systems, the DFI write data bus width will be multiplied proportional to the frequency ratio to allow the MC to send all of the DRAM-required write data to the PHY in a single DFI clock cycle. The write data must be delivered with the DFI data words aligned in ascending order.

- In a matched frequency system, the MC and the PHY operate with a 1:1 ratio.
- In a frequency ratio system, the MC and the PHY operate with a common frequency ratio of 1:2 or 1:4; the PHY must be able to accept a command on any and all phases. The frequency ratio depends on the relationship of the reference clocks for the MC and the PHY.
- Phase-specific signals with a suffix of “_pN”, with the phase number N (e.g., **dfi_wrdata_pN**), replace the matched frequency signals for the control, write data, read data and status interface signals. Phase-specific signals allow the MC to drive multiple commands in a single clock cycle.
- Data word-specific signals with a suffix of “_wN”, with the DFI data word number N (e.g., **dfi_rddata_wN**), replace the matched frequency signals for the read interface to distinguish how DRAM words are transferred across the DFI bus.
- Variable pulse width-specific signals with a suffix of “aN”, with the PHY clock cycle N (e.g., **dfi_alert_n_aN**), replace the matched frequency signals for the status interface to maintain the pulse width during transmission of error signals from the memory system to the PHY.

For all signal types, the suffix for phase 0/data word 0/clock cycle 0 is optional.

For more information on frequency ratios, refer to Section 4.9, “Frequency Ratios Across the DFI”.

2.1 Optional Protocols

Optional protocols handle data bus inversion (DBI), cyclic redundancy check (CRC), system frequency change, command/address (CA) parity, low power and the error interface. For more information on optional protocols, refer to Section 4.4, “Data Bus Inversion”, Section 4.5.3, “Cyclic Redundancy Check”, Section 4.10, “Frequency Change”, Section 4.11, “CA Parity Signaling and CA Parity, CRC Errors”, Section 4.12.3, “Initiating a Training Operation” and Section 4.14, “Error Signaling”.

2.2 DFI Feature Requirements

The DFI specification defines the MC-PHY interface for numerous memory protocols and topologies. Not all DFI features are applicable or required for any particular memory sub-system. Features are divided into global and memory-specific features to aid interoperability and optimal system design.

2.2.1 Global Features

Global features apply to all memory topologies. While these features are valid across all memory sub-systems, they are not always required. For example, any system can utilize “low power control” to reduce system power. However, the requirement of “low power control” is based on system trade-offs and constraints that are outside the scope of DFI. The list of global features are:

- Frequency ratios across DFI
- Frequency change
- Low power control
- Error signaling
- Update interface
- PHY master interface
- Clock disabling
- Data bit enable
- DFI disconnect
- Independent channel support

For specific signal requirements, refer to Table 3, “DFI Signal Requirements”.

2.2.2 Memory Topology-Specific Features

A subset of DFI features is not globally applicable to all memory sub-systems. A matrix of memory topology and features is defined in Table 2, “Features by Memory Topology”. While the feature matrix defines when a feature applies to a specific memory topology, each feature is not necessarily required. Specific feature requirements are based on system trade-offs and constraints that are outside the scope of DFI. For specific signal requirements per feature, refer to Table 2, “Features by Memory Topology”.

TABLE 2. *Features by Memory Topology*

Memory Class	Topologies	Applicable Features [a,b]
DDR1	Discrete, DIMM	No additional features beyond global features
DDR2	Discrete, Unbuffered DIMM, Registered DIMM	No additional features beyond global features
DDR3	Discrete, Unbuffered DIMM	• Read data-eye training • Read gate training • Write leveling • Write data-eye training [c] • DFI disconnect during training
	Registered DIMM	• All DDR3 discrete, unbuffered DIMM features • CA parity • CA parity errors
DDR4	Discrete, Unbuffered DIMM	• Read data-eye training • Read gate training • Write leveling • Write date-eye training [c] • DFI disconnect during training • Read data bus inversion • Write data bus inversion • Write data CRC • Write data CRC errors • CA parity • CA parity errors • Geardown mode
	Registered DIMM	All DDR4 discrete and unbuffered DIMM features
	Load Reduced DIMM	• All DDR4 discrete and unbuffered DIMM features • Data buffer training
LPDDR1	Any	No additional features beyond global features
LPDDR2	S2, S4	• Read data-eye training • Read gate training • Write data-eye training [c] • DFI disconnect during training
	NVM	• All LPDDR2 S2, S4 features • Data not valid
LPDDR3	Any	• Read data-eye training • Read gate training • Write date-eye training [c] • CA training • DFI disconnect during training

TABLE 2. *Features by Memory Topology*

Memory Class	Topologies	Applicable Features [a,b]
LPDDR4	Any	• Read data-eye training • Read gate training • Write leveling • Write date-eye training [c] • CA training • Read data bus inversion • Write data bus inversion • Combined and multi-configuration channel support • DFI disconnect during training

[a] DFI defines features for specific memory class and topology. However, features can be extended beyond memory classes that are explicitly supported in DFI. This topic is outside the scope of DFI.

[b] Feature support is limited to features that are included in a specific DFI version. For example, DFI support of DFI disconnect is not applicable to DFI 3.1 or previous MC and PHY components

[c] JEDEC has defined explicit support of write data training for LPDDR4. DFI has extended this to other DRAM classes by using functional write and read commands, starting with DFI 4.0. For DFI 3.1 and earlier, write data eye training is not supported on DFI.

Items as follows describe some limits of the feature topology matrix and serve to simplify the matrix and DFI.

- Package types (PoP, MCP, DDP, QDP, etc.) are not defined in DFI and therefore are not part of the feature topology matrix.
- Features that are defined completely with a signal, with no other DFI ramifications, are not included in the feature topology matrix. This list includes the following items:
  - Termination: defined by **dfi_odt**
  - Periodic training: defined by **dfi_lvl_periodic**
  - Training patterns: defined through **dfi_lvl_pattern**
  - Per rank delay line support: defined by **dfi_rddata_cs** / **dfi_wrdata_cs**

  For these cases, support is defined in the DFI Signal Requirements section.

DRAM features supported but not explicitly covered by DFI are not included in the matrix. These include:

- DQ VREF training
- Per DRAM addressability

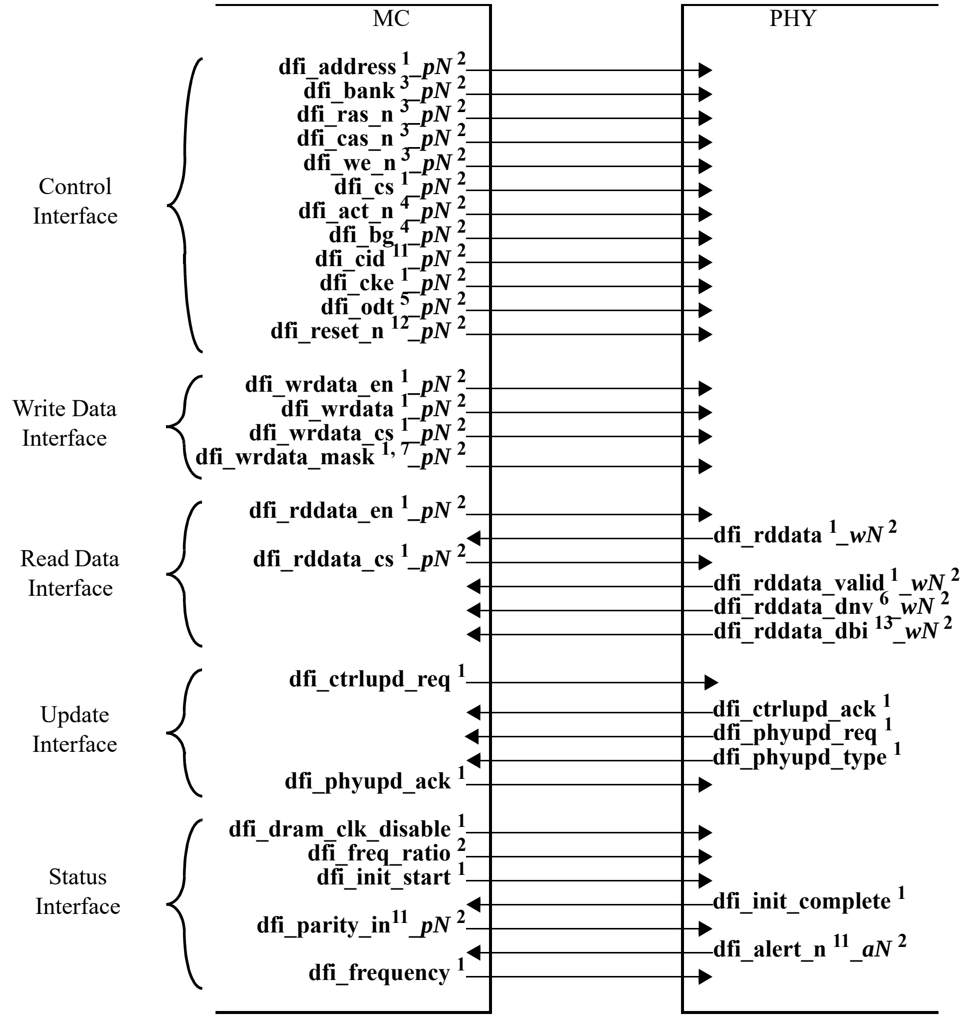
The DRAM type and system configuration determine the types of training available to a system; a system may or may not utilize each type of training. If training is supported, the system may utilize DFI training or support a different training method.

2.2.3 DFI Signals

The DFI signals associated with each interface group, and the device originating the signal, are shown in Figure 1, “Block Diagram of Interface Signals: Control, Write, Read, Update and Status”, Figure 2, “Block Diagram of Interface Signals: Training” and Figure 3, “Block Diagram of Interface Signals: Low Power Control, Error, DB Training, PHY Master and

Geardown”. The signal requirements and parameters associated with each signal are listed in Table 3, “DFI Signal Requirements”. Other signals may exist between the MC and the PHY for a particular implementation. The signals are listed functionally within each interface group.

FIGURE 1. *Block Diagram of Interface Signals: Control, Write, Read, Update and Status*

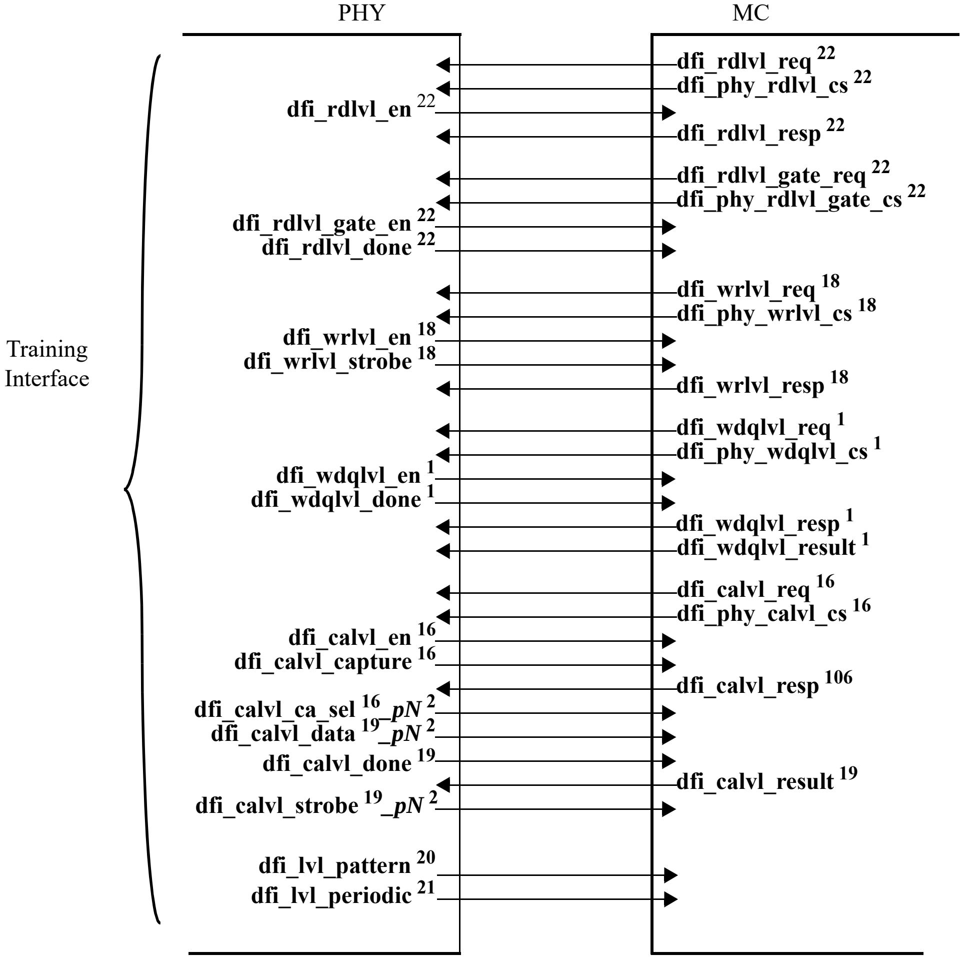


1. Used by all DRAMs.
2. Optional suffix for frequency ratio systems.
3. Used with DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs.
4. Used with DDR4 DRAM only.
5. Used with DDR2, DDR3, DDR4 and LPDDR3 DRAMs.
6. Used with LPDDR2 DRAM only.
7. Dual-function signal. In DDR4/LPDDR4 systems with write DBI enabled, the signal transforms from a mask to a write DBI signal.
8. Used with DDR2, DDR3, LPDDR2 and LPDDR3 DRAMs.
9. Used with DDR3, DDR4 and LPDDR3 DRAMs.
10. Used with LPDDR3 DRAMs only.
11. Used with DDR3 and DDR4 DRAMs.
12. Used with DDR3, DDR4 and LPDDR4 DRAMs.
13. Used with DDR4 and LPDDR4 DRAMs.

Italicized text indicates that the phase/word/cycle is optional.

14. Used with DDR4, LPDDR3 and LPDDR4 DRAMs.
15. Used with DDR3, DDR4, LPDDR2 and LPDDR3 DRAMs.
16. Used with LPDDR3 and LPDDR4 DRAMs.
17. Used with DDR4 LRDIMMs only.
18. Used with DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs.
19. Used with LPDDR4 DRAM only.
20. Used with DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.
21. Used with DDR4, LPDDR2 and LPDDR3 DRAMs.
22. Used with DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.

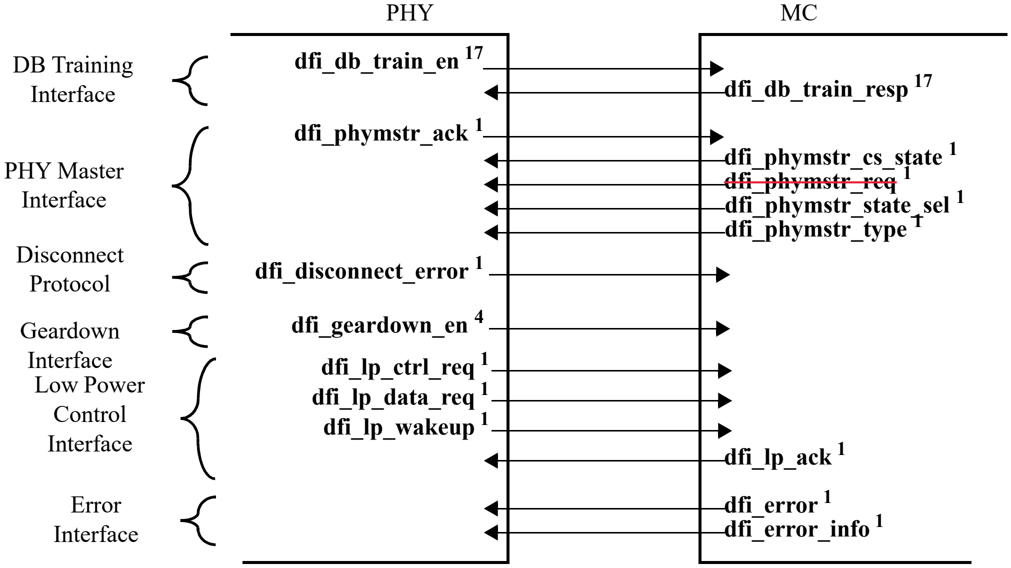
FIGURE 2. *Block Diagram of Interface Signals: Training*



1. Used by all DRAMs.
2. Optional suffix for frequency ratio systems.
3. Used with DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs.
4. Used with DDR4 DRAM only.
5. Used with DDR2, DDR3, DDR4 and LPDDR3 DRAMs.
6. Used with LPDDR2 DRAM only.
7. Dual-function signal. In DDR4/LPDDR4 systems with write DBI enabled, the signal transforms from a mask to a write DBI signal.
8. Used with DDR2, DDR3, LPDDR2 and LPDDR3 DRAMs.
9. Used with DDR3, DDR4 and LPDDR3 DRAMs.
10. Used with LPDDR3 DRAMs only.
11. Used with DDR3 and DDR4 DRAMs.
12. Used with DDR3, DDR4 and LPDDR4 DRAMs.
13. Used with DDR4 and LPDDR4 DRAMs.
14. Used with DDR4, LPDDR3 and LPDDR4 DRAMs.
15. Used with DDR3, DDR4, LPDDR2 and LPDDR3 DRAMs.
16. Used with LPDDR3 and LPDDR4 DRAMs.
17. Used with DDR4 LRDIMMs only.
18. Used with DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs.
19. Used with LPDDR4 DRAM only.
20. Used with DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.
21. Used with DDR4, LPDDR2 and LPDDR3 DRAMs.
22. Used with DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.

Italicized text indicates that the phase/word/cycle is optional.

FIGURE 3. *Block Diagram of Interface Signals: Low Power Control, Error, DB Training, PHY Master and Geardown*



1. Used by all DRAMs.
2. Optional suffix for frequency ratio systems.
3. Used with DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs.
4. Used with DDR4 DRAM only.
5. Used with DDR2, DDR3, DDR4 and LPDDR3 DRAMs.
6. Used with LPDDR2 DRAM only.
7. Dual-function signal. In DDR4/LPDDR4 systems with write DBI enabled, the signal transforms from a mask to a write DBI signal.
8. Used with DDR2, DDR3, LPDDR2 and LPDDR3 DRAMs.
9. Used with DDR3, DDR4 and LPDDR3 DRAMs.
10. Used with LPDDR3 DRAMs only.
11. Used with DDR3 and DDR4 DRAMs.
12. Used with DDR3, DDR4 and LPDDR4 DRAMs.
13. Used with DDR4 and LPDDR4 DRAMs.

Italicized text indicates that the phase/word/cycle is optional.

14. Used with DDR4, LPDDR3 and LPDDR4 DRAMs.
15. Used with DDR3, DDR4, LPDDR2 and LPDDR3 DRAMs.
16. Used with LPDDR3 and LPDDR4 DRAMs.
17. Used with DDR4 LRDIMMs only.
18. Used with DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs.
19. Used with LPDDR4 DRAM only.
20. Used with DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.
21. Used with DDR4, LPDDR2 and LPDDR3 DRAMs.
22. Used with DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.

To determine which signals are required for a specific configuration, review Table 3, “DFI Signal Requirements”. This table identifies the signals associated with each interface group, the parameters associated with each signal, and whether the signal is applicable, required, or optional for each device.

Each signal is device-specific and has corresponding parameters which must be used. Multiple parameter types may apply to a signal. Timing parameters are indicated with the prefix **t** (e.g., $\mathbf{t}_{\mathbf{xxxxx\_xxxxx}}$). Programmable parameters are indicated with a prefix to indicate the defining device and a suffix (e.g., $\mathbf{phy}_{xxxx\_en}$). The signals are listed functionally within each interface group.

TABLE 3. *DFI Signal Requirements*

Control Interface Group			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_act_n_*pN*	t_{ctrl_delay}	Required for DDR4. [b]	Required for DDR4. [b]
dfi_address_*pN*	t_{ctrl_delay}	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_bank_*pN*	t_{ctrl_delay}	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_bg_*pN*	t_{ctrl_delay}	Required for DDR4. [b]	Required for DDR4. [b]
dfi_cas_n_*pN*	t_{ctrl_delay}	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_cid_*pN*	t_{ctrl_delay}	Required for DDR3 and DDR4 3D stack devices. [b]	Required for DDR3 and DDR4 3D stack devices. [b]
dfi_cke_*pN*	t_{ctrl_delay}	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_cs_*pN*	t_{cmd_lat} t_{ctrl_delay}	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_odt_*pN*	$\mathbf{phy}_{crc\_mode}$ t_{ctrl_delay}	Required for DDR2, DDR3, DDR4 and LPDDR3 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR2, DDR3, DDR4 and LPDDR3 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]

TABLE 3. *DFI Signal Requirements*

dfi_ras_n_*pN*	t_{ctrl_delay}	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_reset_n_*pN*	t_{ctrl_delay}	Required for DDR3, DDR4 and LPDDR4 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR3, DDR4 and LPDDR4 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_we_n_*pN*	t_{ctrl_delay}	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR1, DDR2, DDR3, DDR4 and LPDDR1 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
Write Interface Group			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_wrdata_*pN*	phy_{crc_mode} t_{phy_wrdata} t_{phy_wrlat}	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_wrdata_cs_*pN*	$t_{phy\_wrcsgap}$ $t_{phy\_wrcslat}$	Required for all DRAMs if any of the training features are supported, otherwise, optional. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Optional. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_wrdata_en_*pN*	dfi_{rw_length} phy_{crc_mode} t_{cmd_lat} $t_{phy\_crcmax\_lat}$ $t_{phy\_crcmin\_lat}$ t_{phy_wrdata} t_{phy_wrlat} t_{wrdata_delay}	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]

TABLE 3. *DFI Signal Requirements*

dfi_wrdata_mask_*pN*	$\mathbf{phy}_{\mathbf{crc\_mode}}$ $t_{\mathbf{phy\_wrdata}}$ $t_{\mathbf{phy\_wrlat}}$	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
Read Interface Group			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_rddata_*wN*	$t_{\mathbf{phy\_rdlat}}$ $t_{\mathbf{rddata\_en}}$	Required for all DRAMs. Suffix (_wN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]
dfi_rddata_cs_*pN*	$t_{\mathbf{phy\_rdcsgap}}$ $t_{\mathbf{phy\_rdcslat}}$	Required for all DRAMs if read training is supported, otherwise, optional. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Optional. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_rddata_dbi_*wN*	$\mathbf{phy}_{\mathbf{dbi\_mode}}$ $t_{\mathbf{phy\_rdlat}}$ $t_{\mathbf{rddata\_en}}$	Applicable for DDR4 and LPDDR4 only. Required when MC DBI support is enabled and $\mathbf{phy}_{\mathbf{dbi\_mode}} = \mathbf{0}$. Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]	Applicable for DDR4 and LPDDR4 only. Required when MC DBI support is enabled and $\mathbf{phy}_{\mathbf{dbi\_mode}} = \mathbf{0}$. Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]
dfi_rddata_dnv_*wN*	$t_{\mathbf{phy\_rdlat}}$ $t_{\mathbf{rddata\_en}}$	Required for LPDDR2 DRAM. [b] Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]	Required for LPDDR2 DRAM. [b] Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]
dfi_rddata_en_*pN*	$t_{\mathbf{phy\_rdlat}}$ $t_{\mathbf{rddata\_en}}$ $\mathbf{dfi}_{\mathbf{rw\_length}}$	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for all DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
dfi_rddata_valid_*wN*	$t_{\mathbf{phy\_rdlat}}$ $t_{\mathbf{rddata\_en}}$	Required for all DRAMs. Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]	Required for all DRAMs. Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]

TABLE 3. *DFI Signal Requirements*

Update Interface Group			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_ctrlupd_ack	$t_{ctrlupd\_max}$ $t_{ctrlupd\_min}$	Required for all DRAMs.	Optional.
dfi_ctrlupd_req	$t_{ctrlupd\_interval}$ $t_{ctrlupd\_max}$ $t_{ctrlupd\_min}$	Required for all DRAMs.	Optional.
dfi_phyupd_ack	t_{phyupd_typeX}	Required for all DRAMs.	Optional.
dfi_phyupd_req	t_{phyupd_resp} t_{phyupd_typeX}	Required for all DRAMs.	Optional.
dfi_phyupd_type	t_{phyupd_typeX}	Required for all DRAMs.	Optional.
Status Interface Group			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_alert_n_*aN*	$\mathbf{phy}_{crc\_mode}$ t_{parin_lat} $t_{phy\_crcmax\_lat}$ $t_{phy\_crcmin\_lat}$	Required for the following systems: • DDR3 RDIMM systems • DDR4 systems that support CRC, CA parity, or both. In all other cases, this signal is not required, but can optionally be included. Requirement because of CRC is unrelated to $\mathbf{phy}_{crc\_mode}$ value. Requirement because of CA parity is unrelated to location (MC, PHY) that the parity is generated. Suffix (_aN) required for frequency ratio systems to replicate information across the word. [a]	Required for the following systems: • DDR3 RDIMM systems • DDR4 systems that support CRC, CA parity, or both. In all other cases, this signal is not required, but can optionally be included. Requirement because of CRC is unrelated to $\mathbf{phy}_{crc\_mode}$ value. Requirement because of CA parity is unrelated to location (MC, PHY) that the parity is generated. Suffix (_aN) required for frequency ratio systems to replicate information across the word. [a]
dfi_dram_clk_disable	$t_{dram\_clk\_disable}$ $t_{dram\_clk\_enable}$	Required for all DRAMs.	Required for all DRAMs.
dfi_freq_ratio		Required if the system supports multiple frequency ratios. [b] Not applicable if the system supports a single ratio.	Required if the system supports multiple frequency ratios. Not applicable if the system supports a single ratio.
dfi_frequency		Required for all DRAMs.	Optional.

TABLE 3. *DFI Signal Requirements*

dfi_init_complete	$t_{init\_complete}$ $t_{init\_complete\_min}$ t_{init_start}	Required for all DRAMs.	Required for all DRAMs.
dfi_init_start	$t_{init\_complete}$ t_{init_start} $t_{init\_start\_min}$	equired for systems supporting frequency change.	Required for systems supporting frequency change.
dfi_parity_in_*pN*	t_{parin_lat}	Required for the following systems: • DDR3 RDIMM systems • DDR4 systems that support CA parity In all other cases, this signal is not required, but can optionally be included. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Optional. Only relevant for the following systems when the PHY requires the MC to generate the parity information: • DDR3 RDIMM systems • DDR4 systems that support CA parity. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
Training Interface Group - Read Training			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_phy_rdlvl_cs	t_{rdlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Only applicable when read training is supported.
dfi_phy_rdlvl_gate_cs	t_{rdlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Only applicable when read training is supported.
dfi_rdlvl_done		Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs when read data eye training is supported.[b]
dfi_rdlvl_en	$mc_{rdlvl\_slice\_group[n]}$ $phy_{rdlvl\_slice\_group[n]}$ t_{rdlvl_en} t_{rdlvl_max} t_{rdlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs when read training is supported.[b]
dfi_rdlvl_gate_en	t_{rdlvl_en} t_{rdlvl_max} t_{rdlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs when read training is supported. [b]

TABLE 3. *DFI Signal Requirements*

dfi_rdlvl_gate_req	t_{rdlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Only applicable when read training is supported.
dfi_rdlvl_req	$mc_{rdlvl\_slice\_group[n]}$ $phy_{rdlvl\_slice\_group[n]}$ t_{rdlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Only applicable when read training is supported.
dfi_rdlvl_resp	t_{rdlvl_max}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs when read training is supported.[b]
(not associated with a signal)	phy_{rdlvl_en}	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs when read training is supported.[b]
(not associated with a signal)	$phy_{rdlvl\_gate\_en}$	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable to DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs when read training is supported. [b]
Training Interface Group - Write Leveling			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_phy_wrlvl_cs	t_{wrlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs that support write leveling in PHY evaluation mode. [b]	Optional when write leveling in PHY evaluation mode is supported.
dfi_wrlvl_en	t_{wrlvl_en} t_{wrlvl_max} t_{wrlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs that support write leveling in PHY evaluation mode. [b]	Optional. Applicable for DDR3, DDR4,LPDDR3 and LPDDR4 DRAMs when write leveling in PHY evaluation mode is supported. [b]
dfi_wrlvl_req	t_{wrlvl_resp}	Optional. Applicable to DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs that support write leveling in PHY evaluation mode. [b]	Optional when write leveling in PHY evaluation mode is supported.
dfi_wrlvl_resp	t_{wrlvl_max}	Optional. Applicable to DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs that support write leveling in PHY evaluation mode. [b]	Optional. Applicable for DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs when write leveling in PHY evaluation mode is supported. [b]
dfi_wrlvl_strobe_*pN*	t_{wrlvl_en} t_{wrlvl_ww}	Optional. Applicable to DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs that support write leveling in PHY evaluation mode. [b]	Optional. Applicable for DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs when write leveling in PHY evaluation mode is supported. [b]

TABLE 3. *DFI Signal Requirements*

(not associated with a signal)	$\mathbf{phy}_{\mathbf{wrlvl\_en}}$	Optional. Applicable to DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs that support write leveling in PHY evaluation mode. [b]	Optional. Applicable for DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs when write leveling in PHY evaluation mode is supported. [b]
Training Interface Group - Write DQ Training			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_phy_wdqlvl_cs		Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]
dfi_wdqlvl_done		Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]
dfi_wdqlvl_en	$\mathbf{t}_{\mathbf{wdqlvl\_en}}$ $\mathbf{t}_{\mathbf{wdqlvl\_max}}$ $\mathbf{t}_{\mathbf{wdqlvl\_resp}}$	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]
dfi_wdqlvl_req	$\mathbf{t}_{\mathbf{wdqlvl\_resp}}$	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]
dfi_wdqlvl_resp	$\mathbf{t}_{\mathbf{wdqlvl\_max}}$	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]
dfi_wdqlvl_result		Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]
(not associated with a signal)	$\mathbf{t}_{\mathbf{wdqlvl\_rw}}$ $\mathbf{t}_{\mathbf{wdqlvl\_ww}}$ $\mathbf{phy}_{\mathbf{wdqlvl\_bst}}$	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]	Optional. Applicable to any DRAMs that support write DQ training in PHY evaluation mode. [b]
Training Interface Group - CA Training			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_calvl_ca_sel_*pN*	$\mathbf{t}_{\mathbf{calvl\_cs\_ca}}$ $\mathbf{t}_{\mathbf{calvl\_ca\_sel}}$	Optional. Applicable to LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR3 and LPDDR4 DRAMs when CA training is supported. [b]
dfi_calvl_capture	$\mathbf{t}_{\mathbf{calvl\_capture}}$ $\mathbf{t}_{\mathbf{calvl\_cc}}$ $\mathbf{t}_{\mathbf{calvl\_en}}$	Optional. Applicable to LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR3 and LPDDR4 DRAMs when CA training is supported. [b]
dfi_calvl_data_*pN*	$\mathbf{t}_{\mathbf{calvl\_data}}$ $\mathbf{t}_{\mathbf{calvl\_strobe}}$	Optional. Applicable to LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR4 DRAMs when CA training is supported. [b]

TABLE 3. *DFI Signal Requirements*

dfi_calvl_done		Optional. Applicable to LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR4 DRAMs when CA training is supported. [b]
dfi_calvl_en	t_{calvl_en} t_{calvl_max} t_{calvl_resp}	Optional. Applicable to LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR3 and LPDDR4 DRAMs when CA training is supported. [b]
dfi_calvl_req	t_{calvl_resp}	Optional. Applicable to LPDDR3 and LPDDR4 DRAMs. [b]	Optional when CA training is supported.
dfi_calvl_resp	t_{calvl_max}	Optional. Applicable to LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR3 and LPDDR4 DRAMs when CA training is supported. [b]
dfi_calvl_result		Optional. Applicable to LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR4 DRAMs when CA training is supported. [b]
dfi_calvl_strobe_*pN*	t_{calvl_strobe}	Optional. Applicable to LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR4 DRAMs when CA training is supported. [b]
dfi_phy_calvl_cs	t_{calvl_resp}	Optional. Applicable to LPDDR3 and LPDDR4 DRAMs. [b]	Optional when CA training is supported.
(not associated with a signal)	phy_{calvl_en}	Optional. Applicable to LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable for LPDDR3 and LPDDR4 DRAMs when CA training is supported. [b]
Training Interface Group - Leveling			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_lvl_pattern	t_{rdlvl_en} t_{rdlvl_max} t_{rdlvl_resp}	Optional. Applicable to DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]	Optional. Applicable for DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs. [b]
dfi_lvl_periodic	t_{rdlvl_en} t_{rdlvl_max} t_{rdlvl_resp} t_{wrlvl_en} t_{wrlvl_max} t_{wrlvl_resp}	Optional. Applicable to DDR4, LPDDR2 and LPDDR3 DRAMs. [b]	Optional.
Low Power Control Interface Group (optional)			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_lp_ack	t_{lp_resp} t_{lp_wakeup}	Supported for all DRAM types. Required when low power is supported.	Supported for all DRAM types. Required when low power is supported.

TABLE 3. *DFI Signal Requirements*

dfi_lp_ctrl_req	t_{lp_resp}	Supported for all DRAM types. Required when low power is supported.	Supported for all DRAM types. Required when low power is supported.
dfi_lp_data_req	t_{lp_resp}	Supported for all DRAM types. Required when low power is supported.	Supported for all DRAM types. Required when low power is supported.
dfi_lp_wakeup	t_{lp_wakeup}	Supported for all DRAM types. Required when low power is supported.	Supported for all DRAM types. Required when low power is supported.
Error Interface Group (optional)			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_error	t_{error_resp}	Supported for all DRAM types. Required when error interface is supported.	Supported for all DRAM types. Required when error interface is supported.
dfi_error_info	t_{error_resp}	Optional.	Optional.
DB Training Group (optional)			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_db_train_en	$t_{phy\_db\_train\_en}$	Supported for DDR4 LRDIMMs and PHY evaluation mode training only.	Supported for DDR4 LRDIMMs and PHY evaluation mode training only.
dfi_db_train_resp_*wN*	$t_{phy\_db\_train\_resp}$	Supported for DDR4 LRDIMMs and PHY evaluation mode training only.	Supported for DDR4 LRDIMMs and PHY evaluation mode training only.
PHY Master Interface Group			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_phymstr_ack	$t_{phymstr\_resp}$	Required for all DRAMs.	Optional.
dfi_phymstr_cs_state		Required for all DRAMs.	Optional.
dfi_phymstr_req	$t_{phymstr\_resp}$ $t_{phymstr\_typeX}$	Required for all DRAMs.	Optional.
dfi_phymstr_state_sel		Required for all DRAMs.	Optional.
dfi_phymstr_type		Required for all DRAMs.	Optional.
(not associated with a signal)	$t_{phymstr\_rfsh}$	Required for all DRAMs.	Optional.

TABLE 3. *DFI Signal Requirements*

Disconnect Protocol Group			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_disconnect_error	$t_{\text{calvl\_disconnect}}$ $t_{\text{calvl\_disconnect\_error}}$ $t_{\text{ctrlupd\_disconnect}}$ $t_{\text{ctrlupd\_disconnect\_error}}$ $t_{\text{phymstr\_disconnect}}$ $t_{\text{phymstr\_disconnect\_error}}$ $t_{\text{phyupd\_disconnect}}$ $t_{\text{phyupd\_disconnect\_error}}$ $t_{\text{rdlvl\_disconnect}}$ $t_{\text{rdlvl\_disconnect\_error}}$ $t_{\text{rdlvl\_gate\_disconnect}}$ $t_{\text{rdlvl\_gate\_disconnect\_error}}$ $t_{\text{wdqlvl\_disconnect}}$ $t_{\text{wdqlvl\_disconnect\_error}}$ $t_{\text{wrlvl\_disconnect}}$ $t_{\text{wrlvl\_disconnect\_error}}$	Optional for all DRAMs.	Optional.
Geardown Interface			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_geardown_en	$t_{\text{geardown\_delay}}$	Required for DDR4 if geardown support is desired.	Optional.

a. For frequency ratio systems, replicates signals into phase/data word/clock cycle-specific buses that define the validity of the data for each phase N *(pN)*/data word N *(wN)*/clock cycle N *(aN)*, as applicable. The phase 0 suffixes are not required.

b. Other DRAMs must hold this signal in the idle state.

3.0 Interface Signal Groups

3.1 Control Interface

The control interface handles the transmission of signals required to drive the address, command and control signals to the DRAM devices; the interface includes signals and timing parameters. The signals are intended to be passed to the DRAM devices in a manner that maintains the timing relationship among the signals on the DFI; the $\mathbf{t_{ctrl\_delay}}$ timing parameter defines the delay introduced between the DFI interface and the DRAM interface.

Some of the control interface signals are DRAM technology-specific and are only required if the associated technology is used. Examples of DRAM technology-specific control interface signals are:

- **dfi_reset_n** is specific to DDR3, DDR4 and LPDDR4 DRAMs
- **dfi_odt** is specific to DDR2, DDR3, DDR4 and LPDDR3 DRAMs

The following signals must be held at constant values when present in an LPDDR2, LPDDR3 or LPDDR4 implementation: **dfi_cid**, **dfi_bank**, **dfi_bg**, **dfi_act_n**, **dfi_ras_n**, **dfi_cas_n** and **dfi_we_n**.

For LPDDR2, LPDDR3 and LPDDR4 DRAMs, the CA bus is mapped onto to the **dfi_address** bus. The following signals must be held at constant values when present in an LPDDR2,LPDDR3 or LPDDR4 implementation: **dfi_cid**, **dfi_bank**, **dfi_bg**, **dfi_act_n**, **dfi_ras_n**, **dfi_cas_n**, and **dfi_we_n**.

The **dfi_address** bus must have a minimum of 20 bits to hold the LPDDR2 or LPDDR3 rising and falling DDR CA bus for the entire clock period. The PHY is responsible for transmitting the 20-bit **dfi_address** bus as a double data rate 10-bit output, transmitting to the LPDDR2 or LPDDR3 DRAM on the rising and falling CA phases. The LPDDR2/LPDDR3 interface mapping is detailed in Table 4, “Bit Definitions of the dfi_address bus for LPDDR2 and LPDDR3”.

TABLE 4. *Bit Definitions of the **dfi_address** bus for LPDDR2 and LPDDR3*

dfi_address	⎍	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CA Bus	↑											LPDDR2/LPDDR3 1									
												9	8	7	6	5	4	3	2	1	0
	↓	LPDDR2/LPDDR3 2																			
		9	8	7	6	5	4	3	2	1	0										

For LPDDR4, the CA bus is a SDR (single data rate) CA bus and is only 6 bits wide.

During CA training, the **dfi_cke** and **dfi_cs** signals have additional functionality. For details on the signal functionality, refer to Table 5, “Control Signals”.

For frequency ratio systems, the buses/signals of the control interface are replicated into phase-specific signals with a suffix of “_pN” that defines the signal value for each phase N of the DFI PHY clock. Phase 0 may exclude the suffix if desired. The MC may issue commands on any phase to communicate with the PHY. For example, the MC may issue commands only on a single phase, such as phase 0, or may issue commands on any combination of phases; the PHY must be able to accept a command on any and all phases.

The signals associated with the control interface are listed in Table 5, “Control Signals”. For more information on the control interface, refer to Section 4.2, “Control Signals”. For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 5. *Control Signals*

Signal	From	Width	Default	Description
dfi_act_n or **dfi_act_n_pN** [a]	MC	1 bit	- [b]	DFI activate signal. This signal is used for encoding DRAM commands. The following signals define all or a subset of the command encoding: **dfi_act_n**, **dfi_cas_n**, **dfi_ras_n**, **dfi_we_n**.
dfi_address or **dfi_address_pN** [a]	MC	DFI Address Width	- [b, c]	DFI address bus. These signals define the address information. The PHY must preserve the bit ordering of the **dfi_address** signals when it sends this data to the DRAM devices. For DDR4 DRAMs, the **dfi_address** bus defines the column address and a portion of the row address. DDR4 devices do not use the **dfi_address** bits [16:14] since DDR4 devices transmit the row address bits [16:14] on **dfi_ras_n**, **dfi_cas_n** and **dfi_we_n**. For larger density devices, **dfi_address** bits A17 and above are utilized. Consequently, when a larger density device interfaces with a DDR4 system, there can be gaps in the address bus. For systems that support multiple DRAM classes, all or a subset of the **dfi_address** bits [16:14] can be used to address a non-DDR4 DRAM. For LPDDR2 and LPDDR3 DRAMs, the **dfi_address** bus maps to the CA bus as described in Section 3.1, “Control Interface”.
dfi_bank or **dfi_bank_pN** [a]	MC	DFI Bank Width	- [b]	DFI bank bus. These signals define the bank information. The PHY must preserve the bit ordering of the **dfi_bank** signals when it sends the DFI bank data to the DRAM devices.
dfi_bg or **dfi_bg_pN** [a]	MC	DFI Bank Group Width	- [b]	DFI bank group. This signal defines the bank group of a command. The PHY must preserve the bit ordering of the **dfi_bg** signals when it sends the DFI bank group data to the DRAM devices.
dfi_cas_n or **dfi_cas_n_pN** [a]	MC	DFI Control Width	0x1 [d]	DFI column address strobe. This signal is used for encoding DRAM commands. The following signals define all or a subset of the command encoding: **dfi_act_n**, **dfi_cas_n**, **dfi_ras_n**, **dfi_we_n**.
dfi_cid or **dfi_cid_pN** [a]	MC	DFI Chip ID Width	- [e]	DFI chip ID. This signal defines the chip ID. This signal is required for 3D stacked solutions.

TABLE 5. *Control Signals*

Signal	From	Width	Default	Description
dfi_cke or **dfi_cke_pN** [a]	MC	DFI CKE Width	0x0 [f] 0x1 [f]	DFI clock enable. This signal defines the clock enable. The MC must drive CKE signals in all phases. The PHY must be able to accept a command on any and all phases for DFI frequency ratio compliance. For LPDDR3 and LPDDR4 memories, during CA training, the **dfi_cke** signal is used in training sequence to enable the output drivers on the DRAM.
dfi_cs or **dfi_cs_pN** [a]	MC	DFI Chip Select Width	0x1	DFI chip select. This signal defines the chip select. For LPDDR3 and LPDDR4 memories, during CA training, the **dfi_cs** signal is used as the calibration command which is transmitted on the bit corresponding to the chip select currently being trained. For frequency ratio systems, the calibration command must be asserted on a single phase to create a single cycle DRAM pulse. The polarity of this signal is defined by the polarity of the corresponding memory signal. For 3DS operation, also refer to the **dfi_cid** signal.
dfi_odt or **dfi_odt_pN** [a]	MC	DFI ODT Width	0x0	DFI on-die termination control bus. These signals define the ODT. The MC must drive ODT signals in all phases. The PHY must be able to accept a command on any and all phases for DFI frequency ratio compliance.
dfi_ras_n or **dfi_ras_n_pN** [a]	MC	DFI Control Width	0x1 [d]	DFI row address strobe. This signal is used for encoding DRAM commands. The following signals define all or a subset of the command encoding: **dfi_act_n**, **dfi_cas_n**, **dfi_ras_n**, **dfi_we_n**.
dfi_reset_n or **dfi_reset_n_pN** [a]	MC	DFI Reset Width	0x0 0x1 [g]	DFI reset bus. These signals define the RESET. The PHY must preserve the bit ordering of the **dfi_reset_n** signals when it sends the DFI chip ID data to the DRAM devices.
dfi_we_n or **dfi_we_n_pN** [a]	MC	DFI Control Width	0x1 [d]	DFI write enable signal. This signal is used for encoding DRAM commands. The following signals define all or a subset of the command encoding: **dfi_act_n**, **dfi_cas_n**, **dfi_ras_n**, **dfi_we_n**.

[a] For frequency ratio systems, replicates signals into phase/data word/clock cycle-specific buses that define the validity of the data for each phase N (pN)/data word N (wN)/clock cycle N (aN), as applicable. The phase 0 suffixes are not required.

[b] This signal is not meaningful during initialization; no default value is required.

[c] For LPDDR2 and LPDDR3 memory systems, the **dfi_address** bus must be driven with an NOP until **dfi_init_complete** is asserted.

[d] This signal has multiple purposes with DDR4 devices. For all commands that have **dfi_act_n** de-asserted, this signal communicates command encoding similar to the functionality defined for other DRAM devices. When **dfi_act_n** is asserted, the signal transmits upper row address bits with the following address mapping: **dfi_cas_n** → A15, **dfi_ras_n** → A16, **dfi_we_n** → A14.

[e] This signal initializes to 0 during initialization; a default value of 0 is required.

[f] Most DRAMs define CKE as low at reset; some devices, such as LPDDR1, LPDDR2 and LPDDR3, define CKE as high at reset. The default value should adhere to the DRAM definition.

g. In general, the **dfi_reset_n** signal is defined as low at reset; however, in some cases it may be necessary to hold **dfi_reset_n** high during initialization.

The timing parameters associated with the control interface are listed in Table 6, “Control Timing Parameters”.

TABLE 6. *Control Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{cmd\_lat}}$	MC	0x0	- [a]	DFI PHY clock cycles [b]	Specifies the number of DFI clocks after the **dfi_cs** signal is asserted until the associated CA signals are driven.
$\mathbf{t_{ctrl\_delay}}$	PHY	0x0	- [a]	DFI clock cycles	Specifies the number of DFI clock cycles from the time that any control signal changes and when the change reaches the DRAM interface. If the DFI clock and the DRAM clock are not phase-aligned, this timing parameter should be rounded up to the next integer value.

a. The minimum supportable value is 0; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

b. For matched frequency systems, a DFI PHY clock is identical to the DFI clock. For frequency ratio systems, this timing parameter is defined in terms of DFI PHY clock cycles.

3.2 Write Data Interface

The write data interface handles the transmission of write data across the DFI from the MC to the PHY; this interface includes signals, timing parameters and programmable parameters related to write data transfers.

Table 7, “Write Data Signals” describes the signals **dfi_wrdata** (write data bus), **dfi_wrdata_cs** (write data chip select), **dfi_wrdata_en** (write data and data mask enable) and **dfi_wrdata_mask** (write data byte mask).

The **dfi_wrdata** bus transfers write data from the MC to the PHY. The **dfi_wrdata_en** signal indicates to the PHY that valid **dfi_wrdata** and **dfi_wrdata_mask** will be transmitted in $\mathbf{t_{phy\_wrdata}}$ cycles.

3.2.1 Write Data Mask/Write DBI

The **dfi_wrdata_mask** signal has two functions. If the DBI feature (described in Section 4.4, “Data Bus Inversion”) is not enabled, **dfi_wrdata_mask** defines the bytes within the **dfi_wrdata** signals that will be written to DRAM. Alternately, if the DBI feature is enabled and $\mathbf{phy_{dbi\_mode}}$ = 0, the **dfi_wrdata_mask** signal becomes a write DBI signal and indicates whether the write data is inverted. When both functions are enabled, refer to the JEDEC specification for the use of the **dfi_wrdata_mask** signal.

3.2.2 Write Data Chip Select

If data chip select is enabled, the **dfi_wrdata_cs** signal indicates the corresponding chip select which is accessed for the associated write data to independently compensate for timing differences on the data interface accessing different chip selects. In a 3DS solution, control, MRW and training activities are to be limited to physical ranks.

3.2.3 Write Data CRC

If the MC generates the CRC, the MC sends the appropriate CRC data word across the DFI bus using the existing **dfi_wrdata** signals and adjusts control signal timing to handle the additional data word.

Table 8, “Write Data Timing Parameters” describes the write timing parameters $\mathbf{t_{phy\_wrcsgap}}$, $\mathbf{t_{phy\_wrcslat}}$, $\mathbf{t_{phy\_wrdata}}$, $\mathbf{t_{phy\_wrlat}}$, $\mathbf{t_{wrdata\_delay}}$, $\mathbf{t_{phy\_crcmax\_lat}}$ and $\mathbf{t_{phy\_crcmin\_lat}}$.

The $\mathbf{t_{phy\_wrcsgap}}$ timing parameter specifies the minimum number of additional DFI PHY clocks required between commands when changing the target chip select driven on the **dfi_wrdata_cs** signal and defines a minimum additional delay between commands when changing the target chip select as required by the PHY. The $\mathbf{t_{phy\_wrcslat}}$ parameter specifies the number of DFI PHY clocks between when a write command is sent on the DFI control interface and when the associated **dfi_wrdata_cs** signal is asserted and has a delay defined relative to the command to maximize timing flexibility.

The $\mathbf{t_{phy\_wrdata}}$ parameter specifies the number of DFI PHY clock cycles between when the **dfi_wrdata_en** signal is asserted to when the associated write data is driven on the **dfi_wrdata** bus. The $\mathbf{t_{phy\_wrlat}}$ parameter specifies the number of DFI PHY clock cycles between when a write command is sent on the DFI control interface and when the **dfi_wrdata_en** signal is asserted. The $\mathbf{t_{wrdata\_delay}}$ parameter specifies the number of DFI clocks from the time that the **dfi_wrdata_en** signal is asserted and when the corresponding write data transfer completes on the DRAM bus. The PHY-defined $\mathbf{t_{phy\_crcmax\_lat}}$ and $\mathbf{t_{phy\_crcmin\_lat}}$ timing parameters create a time window around an error occurrence on the DFI bus. The sequence of events occurs as follows:

1. The PHY sends a write command, followed by associated data and CRC values to the DRAM. A problem could arise during transmission of the values.
2. When the DRAM logic compares the data and CRC values to each other, if it detects mismatch(es), it asserts the ALERT_N signal on the memory bus.
3. The PHY propagates the ALERT_N value to the MC through the **dfi_alert_n** signal.

The MC can use these timing parameters to pinpoint the command or set of commands associated with the error condition and reissue commands after the CRC error is addressed. The CRC error timing parameters define the relationship between **dfi_wrdata_en** and **dfi_alert_n** signals.

Table 9, “Write Data Programmable Parameters” describes the programmable parameters applicable when CRC and DBI features are enabled. When the optional CRC feature is enabled in DFI, the parameter determines whether the MC or the PHY performs CRC generation and validation. When the optional DBI feature is enabled in DFI, the PHY-defined $\mathbf{phy_{dbi\_mode}}$ parameter determines whether DBI generation and data inversion is performed by the MC or the PHY.

3.2.4 Frequency Ratio

For frequency ratio systems, the signals are replicated into phase-specific signals with a suffix of “_pN” that defines the signal value for each phase N of the PHY clock. Phase 0 may exclude the suffix if desired.

3.2.5 Write Data Signals and Parameters

The signals and parameters in the write data interface are listed in Table 7, “Write Data Signals”, Table 8, “Write Data Timing Parameters”, and Table 9, “Write Data Programmable Parameters”.

For more information on the write data interface, refer to Section 4.5, “Write Transactions”. For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 7. *Write Data Signals*

Signal	From	Width	Default	Description
dfi_wrdata or **dfi_wrdata_pN** [a]	MC	DFI Data Width	- [b]	Write data. These signals transfer write data from the MC to the PHY $\mathbf{t}_{\mathbf{phy\_wrdata}}$ cycles after the **dfi_wrdata_en** signal is asserted and continues transferring data for the number of cycles that the **dfi_wrdata_en** signal is asserted.
dfi_wrdata_cs or **dfi_wrdata_cs_pN** [a]	MC	DFI Physical Rank Width [c]	- [b]	DFI write data chip select. The polarity of this signal is the same as the polarity of the **dfi_cs** signal. This signal serves two functions as follows: • During write leveling, **dfi_wrdata_cs** indicates the chip select that is currently active. • During non-leveling operation, **dfi_wrdata_cs** indicates the chip select that is accessed or targeted for associated write data.
dfi_wrdata_en or **dfi_wrdata_en_pN** [a]	MC	DFI Data Enable Width [d]	0x0	Write data and data mask enable. This signal indicates to the PHY that valid **dfi_wrdata** will be transmitted in $\mathbf{t}_{\mathbf{phy\_wrdata}}$ cycles. Both $\mathbf{t}_{\mathbf{phy\_wrlat}}$ and $\mathbf{t}_{\mathbf{phy\_wrdata}}$ may be defined as zero. Ideally, there is a one-to-one correspondence between **dfi_wrdata_en** bits and PHY data slices. The **dfi_wrdata_en** [0] signal corresponds to the lowest segment of **dfi_wrdata** signals.
dfi_wrdata_mask or **dfi_wrdata_mask_pN** [a]	MC	DFI Data Width / 8	- [b]	Write data byte mask. This bus is used for transferring either the write data mask or the write DBI information, depending on system/DRAM settings. It uses the same timing as the **dfi_wrdata** signal. The polarity of this signal is defined by the polarity of the corresponding memory signal. • **dfi_wrdata_mask** [0] = Masking or DBI for the **dfi_wrdata** [7:0] signals • **dfi_wrdata_mask** [1] = Masking or DBI for the **dfi_wrdata** [15:8] signals, etc. If the **dfi_wrdata** bus is not a multiple of 8, the uppermost bit of the **dfi_wrdata_mask** signal corresponds to the most significant partial byte of data.

[a] For frequency ratio systems, replicates signals into phase/data word/clock cycle-specific buses that define the validity of the data for each phase N (pN)/data word N (wN)/clock cycle N (aN), as applicable. The phase 0 suffixes are not required.

[b] This signal is not meaningful during initialization; no default value is required.

[c] In a 3DS solution control, MRW and training activity are limited to Physical Ranks.

[d] Since all bits of the **dfi_wrdata_en** signal are identical, the width of the signal on the MC side and the PHY side may be different; the PHY is not required to use all of the bits.

The timing parameters that are associated with the write data interface are listed in Table 8, “Write Data Timing Parameters”.

TABLE 8. *Write Data Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{phy\_crcmax\_lat}$	System	0x1	- [a]	DFI PHY clock cycles [b]	This parameter specifies the maximum number of DFI PHY cycle clocks between **dfi_wrdata_en** (the DFI cycle that is associated with CRC code being transmitted) and the associated CRC error that is transmitted on **dfi_alert_n**. The PHY samples the CRC code on the DFI interface. The MC samples the associated CRC error on **dfi_alert_n**. Use this parameter with $t_{phy\_crcmin\_lat}$ to determine a window of time that the erroneous data transmits across the DFI bus.
$t_{phy\_crcmin\_lat}$	System	0x0	- [a]	DFI PHY clock cycles [b]	This parameter specifies the minimum number of DFI PHY cycle clocks between **dfi_wrdata_en** (the DFI cycle that is associated with CRC code being transmitted) and the associated CRC error that is transmitted on **dfi_alert_n**. The PHY samples the CRC code on the DFI interface. The MC samples the associated CRC error on **dfi_alert_n**. Use this parameter with $t_{phy\_crcmax\_lat}$ to determine a window of time that the erroneous data transmits across the DFI bus.
$t_{phy\_wrcsgap}$	PHY	0x0	- a	DFI PHY clock cycles [b]	This parameter specifies the minimum number of additional DFI PHY clocks (or DFI PHY clock) cycles that are required between commands when changing the target physical rank that is driven on the **dfi_wrdata_cs** signal. This parameter must be supported in the MC transaction-to-transaction timing. The minimum assertion duration of **dfi_wrdata_cs** is determined by $t_{phy\_wrcsgap}$ + dfi_{rw_length} [c].
$t_{phy\_wrcslat}$	PHY	0x0	- [a]	DFI PHY clock cycles [b]	This parameter specifies the number of DFI PHY clock cycles from the time that a write command is sent on the DFI control interface and when the associated **dfi_wrdata_cs** signal is asserted.
t_{phy_wrdata}	PHY	0x0 [a]	- [a]	DFI PHY clock cycles [b]	This parameter specifies the number of DFI PHY clock cycles from the time that the **dfi_wrdata_en** signal is asserted and when the associated write data is driven on the **dfi_wrdata** signal. The parameter adjusts the relative time between enable and data transfer with no effect on performance. DFI 1.0 and DFI 2.0 MCs support a t_{phy_wrdata} value of only 1. The MC should support a range of t_{phy_wrdata} values. A PHY is designed to operate at a single t_{phy_wrdata} value.

TABLE 8. *Write Data Timing Parameters (Continued)*

Parameter	Defined By	Min	Max	Unit	Description
t_{phy_wrlat}	PHY	0x0 [a]	- [a]	DFI PHY clock cycles [b]	This parameter specifies the number of DFI PHY clock cycles from the time that a write command is sent on the DFI control interface and when the **dfi_wrdata_en** signal is asserted. NOTE: This parameter may be specified as a fixed value, or as a constant that is based on other fixed values in the system.
t_{wrdata_delay}	System	0x0	-	DFI clock cycles	This parameter specifies the number of DFI clocks from the time that the **dfi_wrdata_en** signal is asserted and when the corresponding write data transfer completes on the DRAM bus.

a. The minimum supportable value is 0; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

b. This timing parameter is defined in terms of DFI PHY clock cycles for frequency ratio systems. For matched frequency systems, a DFI PHY clock is identical to the DFI clock.

c. The $\mathbf{dfi_{rw\_length}}$ value is the total number of DFI clocks required to transfer one DFI read or write command worth of data. For a matched frequency system, $\mathbf{dfi_{rw\_length}}$ would typically equal (burst length/2). For a frequency ratio system, $\mathbf{dfi_{rw\_length}}$ is defined in terms of DFI PHY clocks and would typically equal (burst length/2). Additional DFI clock (or DFI PHY clock) cycles must be added for the CRC data transfer.

The programmable parameters associated with the write data interface are listed in Table 9, “Write Data Programmable Parameters”.

TABLE 9. *Write Data Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{phy_{crc\_mode}}$	PHY	Sends CRC data as part of the data burst. • ’b0 = CRC code generation and validation performed in the MC. • ’b1 = CRC code generation and validation performed in the PHY.
$\mathbf{phy_{dbi\_mode}}$	PHY	Determines which device generates DBI and inverts the data. • ’b0 = DBI generation and data inversion performed in the MC. • ’b1 = DBI generation and data inversion performed in the PHY.

3.3 Read Data Interface

The read data transaction handles the capture and return of data across the DFI; the interface includes signals, timing parameters and a programmable parameter.

The width is defined to be replicated and multiply driven to each of the PHY data slices for interconnect simplicity.

Table 10, “Read Data Signals” describes the signals **dfi_rddata** (read data bus), **dfi_rddata_cs** (data path chip select), **dfi_rddata_en** (read data enable), **dfi_rddata_valid** (read data valid indicator), the DRAM-specific LPDDR2 signal **dfi_rddata_dnv** (DFI data not valid) and the DDR4/LPDDR4 signal **dfi_rddata_dbi**. When the **dfi_rddata_dbi** signal is used, it is sent with the **dfi_rddata** signal.

3.3.1 Read DBI

If the DBI feature is enabled and $\mathbf{phy_{dbi\_mode}}$ = 0, the MC captures read DBI data transmitted from the PHY and selectively inverts the read data based on DBI as required.

3.3.2 Read Data Chip Select

When accessing different chip-selects, it may be desirable to independently compensate for timing differences on the data interface. In this case, the PHY may require knowledge of the target chip select for each read transaction. The **dfi_rddata_cs** signal provides the target data path chip select value to each of the PHY data slices. In a 3DS solution control, MRW and training activities are to be limited to physical ranks.

The data path chip select signal **dfi_rddata_cs** is defined similar to the **dfi_cs** signal and has a separate timing parameter, $\mathbf{t_{phy\_rdcslat}}$. The delay is defined relative to the command to maximize timing flexibility. Additionally, the $\mathbf{t_{phy\_rdcsgap}}$, timing parameter defines a minimum additional delay between commands when changing the target chip select as required by the PHY.

3.3.3 Read Data Valid

The **dfi_rddata_valid** signal allows each PHY data slice to return **dfi_rddata** independently. The **dfi_rddata_valid** signal width is equivalent to the number of PHY data slices.

When valid data is being transferred, the **dfi_rddata_valid** signal must be asserted. This signal is a response from the PHY to **dfi_rddata_en** assertion by the MC. Additionally, there is a one-to-one correspondence between **dfi_rddata_en** assertion clocks and **dfi_rddata_valid** assertion clocks.

DFI dictates a timing relationship from **dfi_rddata_en** to **dfi_rddata_valid**, specified by $\mathbf{t_{phy\_rdlat}}$; DFI does not dictate an exact number of cycles. The **dfi_rddata_valid** signal can assert earlier than the maximum delay, and does not need to be held for consecutive cycles if the $\mathbf{t_{phy\_rdlat}}$ value is met for every transfer.

Table 11, “Read Data Timing Parameters” describes the timing parameters $\mathbf{t_{phy\_rdcsgap}}$, $\mathbf{t_{phy\_rdcslat}}$, $\mathbf{t_{rddata\_en}}$ and $\mathbf{t_{phy\_rdlat}}$.

The $\mathbf{t_{phy\_rdlat}}$ parameter defines the maximum number of cycles allowed from the assertion of the **dfi_rddata_en** signal to the assertion of the **dfi_rddata_valid** signal for all data slices. This parameter is specified by the system, but the exact value of this parameter is not determined by the DFI specification.

The $\mathbf{t_{rddata\_en}}$ and $\mathbf{t_{phy\_rdlat}}$ timing parameters must be held constant while commands are being executed on the DFI bus; however, if necessary, the timing parameters may be changed when the bus is in the idle state. These parameters work together to define a maximum number of cycles from the assertion of a read command on the DFI control interface to the assertion of the **dfi_rddata_valid** signal, indicating the first cycle of the read data. Read data may be returned earlier by asserting the **dfi_rddata_valid** signal before $\mathbf{t_{phy\_rdlat}}$ cycles have expired. When the signal **dfi_rddata_valid** is asserted, the entire DFI read data word from the associated data slice must be valid. For the LPDDR2 DFI, the signal **dfi_rddata_dnv** must also be sent with the read data signal **dfi_rddata** when the **dfi_rddata_valid** signal is asserted.

3.3.4 Frequency Ratio

For frequency ratio systems, the read data enable signal is replicated into phase-specific signals with a suffix of “_pN” that defines the signal value for each phase N of the DFI PHY clock relative to the DFI clock. The read data, read data valid and read data not valid signals are replaced with DFI data word-specific signals with a suffix of "_wN" to specify the DFI data word N. For all signal types, the suffix for phase 0/data word 0/clock cycle 0 is optional.

3.3.5 Read Data Signals and Parameters

The signals and parameters in the read data interface are listed in Table 10, "Read Data Signals", Table 11, "Read Data Timing Parameters", and Table 12, "Read Data Programmable Parameter". Table 12, "Read Data Programmable Parameter" describes the programmable parameter $\mathbf{phy}_{\mathbf{dbi\_mode}}$ which applies to both write and read signals.

For more information on the read data interface, refer to Section 4.6, "Read Transactions". For more information on which signals are required and which signals are optional, refer to Table 3, "DFI Signal Requirements".

TABLE 10. *Read Data Signals*

Signal	From	Width	Default	Description
dfi_rddata or **dfi_rddata_wN** [a]	PHY	DFI Data Width	- [b]	Read data bus. This bus transfers read data from the PHY to the MC. Read data is expected to be received at the MC within $\mathbf{t}_{\mathbf{phy\_rdlat}}$ cycles after the **dfi_rddata_en** signal is asserted.
dfi_rddata_cs or **dfi_rddata_cs_pN** [a]	MC	DFI Physical Rank Width [c]	- [d]	DFI read data chip select. The polarity of this signal is the same as the polarity of the **dfi_cs** signal. This signal has two functions. During read training, **dfi_rddata_cs** indicates the chip select that is currently active. During non-leveling operation, **dfi_rddata_cs** indicates which chip select is accessed or targeted for associated read data.
dfi_rddata_dbi or **dfi_rddata_dbi_wN** [a]	PHY	DFI DBI Width	- [b]	Read data DBI. This signal is sent with **dfi_rddata** bus indicating DBI functionality. This signal is used only when $\mathbf{phy}_{\mathbf{dbi\_mode}}$ = 0. The polarity of this signal is defined by the polarity of the corresponding memory signal.
dfi_rddata_dnv or **dfi_rddata_dnv_wN** [a]	PHY	DFI Data Width / 8	0x0	DFI data not valid. The timing is the same as for the **dfi_rddata_valid** signal. The **dfi_rddata_dnv** [0] signal correlates to the **dfi_rddata** [7:0] signals, the **dfi_rddata_dnv** [1] signal correlates to the **dfi_rddata** [15:8] signals, etc. If the **dfi_rddata** bus is not a multiple of 8, the uppermost bit of the **dfi_rddata_dnv** signal corresponds to the most significant partial byte of data.
dfi_rddata_en or **dfi_rddata_en_pN** [a]	MC	DFI Data Enable Width [e]	0x0	Read data enable. This signal indicates to the PHY that a read operation to memory is underway and identifies the number of data words to be read. The **dfi_rddata_en** signal must be asserted $\mathbf{t}_{\mathbf{rddata\_en}}$ cycles after the assertion of a read command on the DFI control interface and remains valid for the duration of contiguous read data expected on the **dfi_rddata** bus. Ideally, there is a single **dfi_rddata_en** bit for each PHY data slice. The **dfi_rddata_en** [0] signal corresponds to the lowest segment of **dfi_rddata** signals.

TABLE 10. *Read Data Signals*

Signal	From	Width	Default	Description
dfi_rddata_valid or **dfi_rddata_valid_wN** [a]	PHY	DFI Read Data Valid Width	0x0	Read data valid indicator. Each bit of the **dfi_rddata_valid** signal is asserted with the corresponding **dfi_rddata** for the number of cycles that data is being sent. The timing is the same as for the **dfi_rddata** bus. The width of the **dfi_rddata_valid** signal is equivalent to the number of PHY data slices. Ideally, there is a one-to-one correspondence between a **dfi_rddata_valid** signal bit and each PHY data slice. The **dfi_rddata_valid**[0] signal corresponds to the lowest segment of the **dfi_rddata** signals.

[a] For frequency ratio systems, replicates signals into phase/data word/clock cycle-specific buses that define the validity of the data for each phase N (pN)/data word N (wN)/clock cycle N (aN), as applicable. The phase 0 suffixes are not required.

[b] This signal is not meaningful during initialization; no default value is required.

[c] In a 3DS solution control, MRW and training activity are limited to Physical Ranks.

[d] The default state for this signal is the inactive value. The polarity of this signal is dependent on the memory.

[e] Since all bits of the **dfi_rddata_en** signal are identical, the width of the signal on the MC side and the PHY side may be different; the PHY is not required to use all of the bits.

The timing parameters associated with the read data interface are listed in Table 11, “Read Data Timing Parameters”.

TABLE 11. *Read Data Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{phy\_rdcsgap}}$	PHY	0x0	- [a]	DFI PHY clock cycles [b]	Specifies the minimum number of additional DFI PHY clocks required between commands when changing the target physical rank driven on the **dfi_rddata_cs** signal. This parameter needs to be supported in the MC transaction-to-transaction timing. The minimum assertion duration of **dfi_rddata_cs** is determined by $\mathbf{t_{phy\_rdcsgap}}$ + $\mathbf{dfi_{rw\_length}}$ [c].
$\mathbf{t_{phy\_rdcslat}}$	PHY	0x0	- [a]	DFI PHY clock cycles [b]	Specifies the number of DFI PHY clocks between when a read command is sent on the DFI control interface and when the associated **dfi_rddata_cs** signal is asserted.
$\mathbf{t_{phy\_rdlat}}$	PHY	0x0	- [a]	DFI PHY clock cycles [b]	Specifies the maximum number of DFI PHY clock cycles allowed from the assertion of the **dfi_rddata_en** signal to the assertion of each of the corresponding bits of the **dfi_rddata_valid** signal.
$\mathbf{t_{rddata\_en}}$	System	0x0	- [a]	DFI PHY clock cycles [b]	Specifies the number of DFI PHY clock cycles from the assertion of a read command on the DFI to the assertion of the **dfi_rddata_en** signal. NOTE: This parameter may be specified as a fixed value, or as a constant based on other fixed values in the system.

[a] The minimum supportable value is 0; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

b. For matched frequency systems, a DFI PHY clock is identical to the DFI clock. For frequency ratio systems, this timing parameter is defined in terms of DFI PHY clock cycles.

c. The $\mathbf{dfi}_{\mathbf{rw\_length}}$ value is the total number of DFI clocks required to transfer one DFI read or write command worth of data. For a matched frequency system, $\mathbf{dfi}_{\mathbf{rw\_length}}$ would typically equal (burst length/2). For a frequency ratio system, $\mathbf{dfi}_{\mathbf{rw\_length}}$ is defined in terms of DFI PHY clocks and would typically equal (burst length/2). Additional DFI clock (or DFI PHY clock) cycles must be added for the CRC data transfer.

The programmable parameter associated with the read data interface is listed in Table 12, "Read Data Programmable Parameter".

TABLE 12. *Read Data Programmable Parameter*

Parameter	Defined By	Description
$\mathbf{phy}_{\mathbf{dbi\_mode}}$	PHY	Determines which device generates DBI and inverts the data. • 'b0 = DBI generation and data inversion performed in the MC. • 'b1 = DBI generation and data inversion performed in the PHY.

3.4 Update Interface

The update interface facilitates various commands that might require interruption of DRAM signal transmission on the DFI. These commands include timing adjustments, calibration, training, etc. This interface defines signals and timing parameters. To ensure that updates do not interfere with signals on the DRAM interface, the DFI supports update modes when the DFI bus is placed in an idle state.

When the DFI bus is in an idle state, the control interface is not sending any commands and all read and write data has transferred on the DFI bus. The data has reached its destination (DRAM or MC) and the write data transfer has completed on the DRAM bus; The state of the DRAM bus is unchanged. The DFI specification supports both MC-initiated and PHY-initiated updates. For more information on the update interface, refer to *Section 4.7, "Update"*.

The MC initiates an update request by asserting the **dfi_ctrlupd_req** signal, following initialization and training (if required). The PHY can acknowledge or ignore the request. If the PHY acknowledges the request, by asserting the **dfi_ctrlupd_ack** signal, the protocol described in Section 4.7.1, "MC-Initiated Update" must be followed.

The $\mathbf{t}_{\mathbf{ctrlupd\_interval}}$ parameter defines the maximum interval at which the MC can assert **dfi_ctrlupd_req** signals. A control update request and acknowledge via **dfi_ctrlupd_req** / **dfi_ctrlupd_ack** is required immediately before a self-refresh exit command runs.

If a PHY initiates an update request by asserting the **dfi_phyupd_req** signal, the MC must acknowledge the request by asserting the **dfi_phyupd_ack** signal. The DFI specifies up to 4 different update PHY-initiated request modes. Each mode differs only in the number of cycles that the DFI interface must be suspended while the update occurs. During this time, the MC is responsible for placing the system in a state where the DFI bus is suspended from all activity other than activity specifically related to the update process being executed. For more details, refer to Section 4.7.2, "PHY-Initiated Update".

The DFI specification does not require the PHY to issue update requests nor does the specification specify an interval in which requests must be offered. If the PHY offers update requests, it must follow the specified protocol.

It is possible that both update request signals (**dfi_ctrlupd_req** and **dfi_phyupd_req**) could be asserted at the same time. When both request signals are driven, the MC and the PHY could violate the protocol by simultaneously acknowledging the other's request. To prevent this situation, the MC is not permitted to assert both **dfi_ctrlupd_req** and

dfi_phyupd_ack at the same time. If **dfi_ctrlupd_req** is asserted at the same time as **dfi_phyupd_req**, the PHY is permitted to de-assert **dfi_phyupd_req**, though it is not required to be de-asserted. This is the only situation in which the PHY is permitted to de-assert the **dfi_phyupd_req** signal without an acknowledge from the MC. Since it is the PHY (not the MC) that uses the Update interface to request the DFI bus to be IDLE, the PHY should not de-assert **dfi_phyupd_req** unless the signal is no longer required due to the assertion of **dfi_ctrlupd_req**. The acknowledged request must follow the appropriate protocol.

The signals associated with the update interface are listed in Table 13, “Update Interface Signals”. For more information on the update interface, refer to *Section 4.7, “Update”*. For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 13. *Update Interface Signals*

Signal	From	Width	Default	Description
dfi_ctrlupd_ack	PHY	1 bit	0x0	MC-initiated update acknowledge. The **dfi_ctrlupd_ack** signal is asserted to acknowledge an MC-initiated update request. The PHY is not required to acknowledge this request. While this signal is asserted, the DFI bus must remain in the idle state except for transactions specifically associated with the update process. If the PHY acknowledges the request, the **dfi_ctrlupd_ack** signal must be asserted before $\mathbf{t_{ctrlupd\_min}}$ occurs and the **dfi_ctrlupd_req** signal de-asserts. If the PHY ignores the request, the **dfi_ctrlupd_ack** signal must remain de-asserted until the **dfi_ctrlupd_req** signal is de-asserted. The **dfi_ctrlupd_req** signal is guaranteed to be asserted for at least $\mathbf{t_{ctrlupd\_min}}$ cycles. The **dfi_ctrlupd_ack** signal cannot be asserted after $\mathbf{t_{ctrlupd\_min}}$, occurs, even if **dfi_ctrlupd_req** is still asserted.

TABLE 13. *Update Interface Signals*

Signal	From	Width	Default	Description
dfi_ctrlupd_req	MC	1 bit	0x0	MC-initiated update request. The **dfi_ctrlupd_req** signal is used with an MC-initiated update to indicate that the DFI will be in the idle state for some time, in which case the PHY may perform an update. The **dfi_ctrlupd_req** signal must be asserted for a minimum of $\mathbf{t_{ctrlupd\_min}}$ cycles and a maximum of $\mathbf{t_{ctrlupd\_max}}$ cycles. A **dfi_ctrlupd_req** signal assertion is an invitation for the PHY to update and does not require a response. The behavior of the **dfi_ctrlupd_req** signal is dependent on the **dfi_ctrlupd_ack** signal: • If the update is acknowledged by the PHY, the **dfi_ctrlupd_req** signal remains asserted as long as the **dfi_ctrlupd_ack** signal is asserted, but **dfi_ctrlupd_ack** must de-assert before $\mathbf{t_{ctrlupd\_max}}$ expires. While **dfi_ctrlupd_req** is asserted, the DFI bus remains in the idle state except for transactions specifically associated with the update process. • If the update is not acknowledged, the **dfi_ctrlupd_req** signal may de-assert at any time after $\mathbf{t_{ctrlupd\_min}}$ occurs and before $\mathbf{t_{ctrlupd\_max}}$ expires. • The MC may de-assert the **dfi_ctrlupd_req** signal to disconnect the handshake through the disconnect protocol.
dfi_phyupd_ack	MC	1 bit	0x0	PHY-initiated update acknowledge. The **dfi_phyupd_ack** signal is used for a PHY-initiated update to indicate that the DFI is idle and remains in the idle state until the **dfi_phyupd_req** signal de-asserts. The MC must assert the **dfi_phyupd_ack** signal within $\mathbf{t_{phyupd\_resp}}$ cycles of the **dfi_phyupd_req** signal, and should keep this signal asserted as long as the **dfi_phyupd_req** signal remains asserted. The **dfi_phyupd_ack** signal must de-assert upon the detection of **dfi_phyupd_req** signal de-assertion. The **dfi_phyupd_req** cannot be re-asserted prior to the de-assertion of **dfi_phyupd_ack** for the previous transaction. The MC may de-assert the **dfi_phyupd_ack** signal to disconnect the handshake through the disconnect protocol. NOTE: If a **dfi_ctrlupd_req** occurs at the same time as a **dfi_phyupd_req**, the $\mathbf{t_{phyupd\_resp}}$ requirement is allowed to not be met since **dfi_phyupd_ack** and **dfi_ctrlupd_req** cannot be asserted simultaneously. While **dfi_phyupd_ack** is asserted, the DFI bus must remain in the idle state except for transactions specifically associated with the update process. The time period from when the **dfi_phyupd_ack** signal is asserted to when the **dfi_phyupd_req** signal is de-asserted is a maximum of $\mathbf{t_{phyupd\_typeX}}$ cycles, based on the **dfi_phyupd_type** signal.

TABLE 13. *Update Interface Signals*

Signal	From	Width	Default	Description
dfi_phyupd_req	PHY	1 bit	0x0	PHY-initiated update request. The **dfi_phyupd_req** signal is used for a PHY-initiated update to indicate that the PHY requires the DFI bus to be placed in an idle state and not send control, read or write commands or data for a specified period of time. The maximum time required is specified by the t_{phyupd_typeX} parameter associated with the **dfi_phyupd_type** signal. Once asserted, the **dfi_phyupd_req** signal must remain asserted until the request is acknowledged by the assertion of the **dfi_phyupd_ack** signal and the update has been completed. The MC must acknowledge this request. While this signal is asserted, the DFI bus must remain in the idle state other than any transactions specifically associated with the update process. The de-assertion of the **dfi_phyupd_req** signal triggers the de-assertion of the **dfi_phyupd_ack** signal.
dfi_phyupd_type	PHY	2 bits	- [a]	PHY-initiated update select. The **dfi_phyupd_type** signal indicates which one of the 4 types of PHY update times is being requested by the **dfi_phyupd_req** signal. The value of the **dfi_phyupd_type** signal determines which of the timing parameters (t_{phyupd_type0}, t_{phyupd_type1}, t_{phyupd_type2}, t_{phyupd_type3}) is relevant. The **dfi_phyupd_type** signal must remain constant during the entire time the **dfi_phyupd_req** signal is asserted.

[a] This signal is not meaningful during initialization; no default value is required.

The timing parameters associated with the update interface are listed in Table 14, “Update Timing Parameters”.

TABLE 14. *Update Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{ctrlupd\_interval}$	MC	- [a]	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the MC may wait between assertions of the **dfi_ctrlupd_req** signal.
$t_{ctrlupd\_max}$	MC	- [a]	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_ctrlupd_req** signal can assert.
$t_{ctrlupd\_min}$	MC	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles that the **dfi_ctrlupd_req** signal must be asserted.
t_{phyupd_resp}	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles after the assertion of the **dfi_phyupd_req** signal to the assertion of the **dfi_phyupd_ack** signal.

TABLE 14. *Update Timing Parameters (Continued)*

Parameter	Defined By	Min	Max	Unit	Description
t_{phyupd_type0}	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phyupd_req** signal may remain asserted after the assertion of the **dfi_phyupd_ack** signal for **dfi_phyupd_type** = 0x0. The **dfi_phyupd_req** signal may de-assert at any cycle after the assertion of the **dfi_phyupd_ack** signal.
t_{phyupd_type1}	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phyupd_req** signal may remain asserted after the assertion of the **dfi_phyupd_ack** signal for **dfi_phyupd_type** = 0x1. The **dfi_phyupd_req** signal may de-assert at any cycle after the assertion of the **dfi_phyupd_ack** signal.
t_{phyupd_type2}	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phyupd_req** signal may remain asserted after the assertion of the **dfi_phyupd_ack** signal for **dfi_phyupd_type** = 0x2. The **dfi_phyupd_req** signal may de-assert at any cycle after the assertion of the **dfi_phyupd_ack** signal.
t_{phyupd_type3}	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phyupd_req** signal may remain asserted after the assertion of the **dfi_phyupd_ack** signal for **dfi_phyupd_type** = 0x3. The **dfi_phyupd_req** signal may de-assert at any cycle after the assertion of the **dfi_phyupd_ack** signal.

a. The minimum supportable value is 1; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

3.5 Status Interface

The status interface signal conveys status information between the MC and the PHY for initialization and clock control to the DRAM devices.

3.5.1 *Initialization*

Initialization uses the **dfi_init_start** and **dfi_init_complete** signals. These signals are also used during a frequency change request, which Section 3.5.4, “Frequency Ratio” describes in detail. At initialization, the **dfi_init_start** signal indicates to the PHY that the optional **dfi_freq ratio** signal from the MC is defined and valid=. and the **dfi_init_complete** signal indicates that the PHY is ready to accept DFI transactions.

For more information on initialization, refer to Section 4.1, “Initialization”.

3.5.2 *Reduced Data Buses*

Not all bits of the DFI data buses (**dfi_wrdata** and **dfi_rddata**) are required to be used for data transfers. However, the MC and the PHY must both support the same partial use of data signals to remain compatible. For example, if the MC disables the upper bits (most significant bits) of the data bus, the PHY must be able to operate accurately with the remaining byte lanes. The DFI specification does not define supported active/inactive patterns, and therefore care must be taken to insure the interoperability of the MC and the PHY.

3.5.3 Clock Disabling

The **dfi_dram_clk_disable** signal is used by the MC to inform the PHY when to enable/disable the clock to the DRAMs. The timing parameters $\mathbf{t_{dram\_clk\_disable}}$ and $\mathbf{t_{dram\_clk\_enable}}$ define the timing of the DRAM clock enable/disable relative to the **dfi_dram_clk_disable** signal.

For more information on the DFI clock interface, refer to Section 4.8, "DFI Clock Disabling".

3.5.4 Frequency Ratio

The optional **dfi_freq_ratio** signal is used for conveying frequency ratio information to the PHY. This static signal indicates the ratio expected by the MC and dictates how control, read and write information is passed across the DFI.

For more information on the frequency ratio protocol, refer to Section 4.9, "Frequency Ratios Across the DFI".

3.5.5 Frequency Change

The DFI specification defines a frequency change protocol between the MC and the PHY to allow the devices to change the clock frequency of the MC and the PHY without completely resetting the system.

The signals used in the frequency change protocol are **dfi_init_start** and **dfi_init_complete** during normal operation. The behavior of the **dfi_init_start** signal depends on the **dfi_init_complete** signal. A frequency change request is triggered when the MC asserts **dfi_init_start**. The PHY indicates the acceptance of the frequency change by asserting **dfi_init_complete**. The associated timing parameters are $\mathbf{t_{init\_start}}$ and $\mathbf{t_{init\_complete}}$.

For more information on the frequency change protocol, refer to Section 4.10, "Frequency Change".

3.5.6 Frequency Indicator

The MC controls the DFI during a frequency change operation and reports the new frequency to the PHY through the encoding of the **dfi_frequency** signal. There is no handshaking for this operation. The encoding of the **dfi_frequency** signal is defined by the PHY, the system, or both; the DFI specification does not impose any fixed frequency mapping. However, it is advisable that frequencies are mapped so that frequencies increase with increasing values of the **dfi_frequency** signal which will allow for setting up thresholds. 32 unique frequencies can be defined through this signal.

There is no initial value defined for the **dfi_frequency** signal; the system defines the initial frequency and must configure the MC to drive the corresponding value when it asserts the **dfi_init_start** signal during initialization. The PHY might wait for the **dfi_init_start** signal to assert. Or if the PHY has no dependencies on initial values of this signal, it might initialize without a **dfi_init_start** signal, including assertion of a **dfi_init_complete** signal. The MC must adhere to the defined functionality of **dfi_init_start**.

While the MC can support up to 32 unique values of the **dfi_frequency** signal, the $\mathbf{phy_{freq\_range}}$ parameter defines the number of frequencies that the PHY supports. The frequency range must be a number between 1 and 32, inclusive, and must start from the value of ZERO to the defined range. The PHY may only support a subset of these frequencies; if so, the PHY should clearly define supported encodings. The MC must be able to support the full range of values as defined by DFI and must never drive a value outside the selected range nor any value within the range that is defined as unsupported by the PHY.

The **dfi_frequency** signal can change any time when **dfi_init_start** is low and should be ignored at this time. Once the **dfi_init_start** signal is asserted, the **dfi_frequency** signal must be set to a legal value and remain unchanged.

3.5.7 CRC and CA Parity

The CRC and CA parity support signals are provided specifically for DDR3 and DDR4 DRAMs. The DDR3 and DDR4 systems include additional IO pins to receive command (CA) parity, communicate command parity and write CRC error status. In systems that require command parity support, the MC communicates its command parity setting on the **dfi_parity_in** signal. In systems that require either CRC or command parity support, the PHY reports CRC or command parity errors on the **dfi_alert_n** signal. For more information on CRC and CA parity support signals, refer to Section 3.6, “DFI Training Interface” and Section 3.2.3, “Write Data CRC”.

3.5.8 Status Interface Signals and Parameters

The signals and parameters in the status interface are listed in Table 15, “Status Interface Signals”, Table 16, “Status Timing Parameters”, and Table 17, “Status Interface Programmable Parameter”.

For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 15. *Status Interface Signals*

Signal	From	Width	Default	Description
dfi_alert_n or **dfi_alert_n_aN** [a]	PHY	DFI Alert Width	0x1	CRC or parity error indicator. This signal is driven when a CRC or command parity error is detected in the memory system. The PHY is not required to distinguish between a CRC and command parity error. The PHY holds the current state until the PHY error input transitions to a new value; the pulse width of the **dfi_alert_n** signal matches the pulse width of the DRAM subsystem error signal, plus or minus synchronization cycles.
dfi_dram_clk_disable	MC	DFI DRAM Clk Disable Width	0x0 [b]	DRAM clock disable. When active, this indicates to the PHY that the clocks to the DRAM devices must be disabled such that the clock signals hold a constant value. When the **dfi_dram_clk_disable** signal is inactive, the DRAMs should be clocked normally.
dfi_freq_ratio	MC	2 bits	- [b]	DFI frequency ratio indicator. This signal defines the frequency ratio for this system. This signal is required for MCs and PHYs that support multiple frequency ratios and the DFI frequency ratio protocol. This signal is optional for MCs and PHYs that support only a single frequency ratio or do not support the DFI frequency ratio protocol. This signal may only be defined during initialization and is expected to remain constant. Once defined, the MC drives the **dfi_init_start** signal to the PHY. • ’b00 = 1:1 MC:PHY frequency ratio (matched frequency) • ’b01 = 1:2 MC:PHY frequency ratio • ’b10 = 1:4 MC:PHY frequency ratio • ’b11 = Reserved

TABLE 15. *Status Interface Signals*

Signal	From	Width	Default	Description
dfi_frequency	MC	5 bits	-[a]	DFI frequency. This signal indicates the operating frequency for the system. This signal should change only at initialization, during a DFI frequency change operation, or other times that the system defines. The number of supported frequencies and the mapping of signal values to clock frequencies are defined by the PHY, system, or both. This signal should be constant during normal operation.
dfi_init_complete	PHY	1 bit	0x0	PHY initialization complete. The **dfi_init_complete** signal indicates that the PHY is able to respond to any proper stimulus on the DFI. All DFI signals that communicate commands or status must be held at their default values until the **dfi_init_complete** signal asserts. During a PHY re-initialization request (such as a frequency change), this signal is de-asserted. For a frequency change request, the de-assertion of the **dfi_init_complete** signal acknowledges the frequency change protocol. Once de-asserted, the signal should only be re-asserted within $\mathbf{t}_{\mathbf{init\_complete}}$ cycles after the **dfi_init_start** signal has de-asserted, and once the PHY has completed re-initialization.
dfi_init_start	MC	1 bit	0x0[c] 0x1	DFI setup stabilization or frequency change initiation. This optional signal can perform two functions. When **dfi_init_start** is asserted during initialization, the MC is indicating that the **dfi_freq_ratio** signal (if present) has been defined. When **dfi_init_start** is asserted during normal operation, the MC is requesting a frequency change. During initialization, when both **dfi_init_start** and **dfi_init_complete** are asserted for at least one DFI clock cycle, the MC can either hold or de-assert the **dfi_init_start** signal. For frequency change, the **dfi_init_start** signal must assert to trigger the event. After the rising edge of **dfi_init_start** occurs, the behavior of the **dfi_init_start** signal depends on the **dfi_init_complete** signal as follows: • If the PHY accepts the frequency change request, it must de-assert the **dfi_init_complete** signal within $\mathbf{t}_{\mathbf{init\_start}}$ cycles of the **dfi_init_start** assertion. The MC continues to hold the **dfi_init_start** signal asserted until the clock frequency change has been completed. The de-assertion should be used by the PHY to re-initialize on the new clock frequency. • If the frequency change is not acknowledged (the **dfi_init_complete** signal remains asserted), the **dfi_init_start** signal must de-assert after $\mathbf{t}_{\mathbf{init\_start}}$ cycles. • The frequency must assert **dfi_init_start** until **dfi_init_complete** is asserted at least one DFI clock. Initialization is complete when both the **dfi_init_start** and the **dfi_init_complete** signals are asserted simultaneously for at least one DFI clock.

TABLE 15. *Status Interface Signals*

Signal	From	Width	Default	Description
dfi_parity_in or **dfi_parity_in_pN** [a]	MC	1 bit	0x0 [d] 0x1[e]	Parity value. This signal has a one-to-one correspondence with each DFI command and is valid for 1 cycle. This value applies to the **dfi_address**, **dfi_bank**, **dfi_bg, dfi_act_n, dfi_cas_n**, **dfi_cid**, **dfi_ras_n** and **dfi_we_n** signals. This signal is relevant for only systems that support command parity. • ’b0 = An even number of the parity value signals are electrically high. • ’b1 = An odd number of the parity value signals are electrically high.

a. For frequency ratio systems, replicates signals into phase/data word/clock cycle-specific buses that define the validity of the data for each phase N (pN)/data word N (wN)/clock cycle N (aN), as applicable. The phase 0 suffixes are not required.

b. When **dfi_init_start** is asserted during initialization, **dfi_dram_clk_disable** must indicate the clocks that are being used. In normal operation, this signal can dynamically change.

c. The PHY may optionally wait for the **dfi_init_start** signal assertion before asserting the **dfi_init_complete** signal.

d. The value of this signal must be driven with the correct parity for the selected control interface signals.

The timing parameters associated with the status interface are listed in Table 16, “Status Timing Parameters”.

TABLE 16. *Status Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{dram\_clk\_disable}$ [a]	PHY	0x0	- [b]	DFI clock cycles [c]	Specifies the number of DFI clock cycles from the assertion of the **dfi_dram_clk_disable** signal on the DFI until the clock to the DRAMs at the PHY-DRAM boundary maintains a low value. NOTE: This parameter may be specified as a fixed value, or as a constant based on other fixed values in the system.
$t_{dram\_clk\_enable}$ [a]	PHY	0x0	- [b]	DFI clock cycles [c]	Specifies the number of DFI clock cycles from the de-assertion of the **dfi_dram_clk_disable** signal on the DFI until the first valid rising edge of the clock to the DRAMs at the PHY-DRAM boundary. NOTE: This parameter may be specified as a fixed value, or as a constant based on other fixed values in the system.
$t_{init\_complete}$	PHY	0x0	- [b]	DFI clock cycles [c]	Specifies the maximum number of DFI clock cycles after the de-assertion of the **dfi_init_start** signal to the re-assertion of the **dfi_init_complete** signal during a frequency change operation.
$t_{init\_complete\_min}$	PHY	0x1	- [a]	DFI clock cycles	Minimum number of DFI clocks before **dfi_init_complete** can be driven after a previous command, training event, or both. With CA training, such events are as follows: • Assertion of **dfi_cke** • Assertion of **dfi_cs** • Assertion of **dfi_calvl_strobe**

TABLE 16. *Status Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
t_{init_start}	MC	0x0	- [b]	DFI clock cycles [c]	During a frequency change operation, this parameter specifies the number of DFI clock cycles from the assertion of the **dfi_init_start** signal on the DFI until the PHY must respond by de-asserting the **dfi_init_complete** signal. If the **dfi_init_complete** signal is not de-asserted within this time period, the PHY indicates that it can not support the frequency change at this time. In this case, the MC must abort the request and release the **dfi_init_start** signal. When t_{init_start} expires, the PHY must not de-assert the **dfi_init_complete** signal. The MC can re-assert **dfi_init_start** at a later point.
$t_{init\_start\_min}$	PHY	0x1	- [a]	DFI clock cycles	Minimum number of DFI clocks before **dfi_init_start** can be driven after a previous command, training event, or both. With CA training, such events are as follows: • De-assertion of **dfi_cke** • Assertion of **dfi_calvl_capture** • Assertion of **dfi_calvl_strobe**
t_{parin_lat}	MC	0x0	- [b]	DFI PHY clock cycles [c]	Specifies the number of DFI PHY clocks between when the DFI command is asserted and when the associated **dfi_parity_in** signal is driven.
$t_{phy\_paritylat}$	PHY	0x4	- [b]	DFI clock cycles [c]	Specifies the maximum number of DFI clock cycles between when the **dfi_parity_in** signal is driven and when the associated **dfi_alert_n** signal is returned.

[a] If the DFI clock and the DRAM clock are not phase-aligned, this timing parameter should be rounded up to the next integer value.

[b] The minimum supportable value is 0; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

[c] For matched frequency systems, a DFI PHY clock is identical to the DFI clock. For frequency ratio systems, this timing parameter is defined in terms of DFI PHY clock cycles.

Previous DFI specifications included a **dfi_data_byte_disable** signal. That signal has been removed for lack of usefulness and been replaced by a programmable parameter. In a system where the MC supports an earlier version of the DFI specification and the PHY is a DFI 4.0 PHY, the MC’s **dfi_data_byte_disable** signal would not be connected and the PHY programmable parameter would be programmed to match the definition of that signal. In a system where a DFI 4.0 MC connects to an older DFI version PHY will require a programmable register that emulates the behavior of the **dfi_data_byte_disable** signal, and the **dfi_data_byte_disable** signal input on the PHY would be driven by external logic or be hardwired.

The $\mathbf{dfi}_{\mathbf{data\_bit\_enable}}$ programmable parameter defines the valid data bits for both the MC and PHY on both the **dfi_wrdata** and **dfi_rddata** interfaces. The parameter must be the same for both read and write data.

Memory devices can have multiple operating modes where the value of this parameter might be different in different modes. In this case, the device must define the parameter for each operating mode. For example, a device might operate with ECC on and ECC off. If so, the value of the parameter differs for both modes. If the parameter is defined the same for both the MC and PHY, then they will be compatible.

However, in some cases, the devices may define the parameter differently such that the devices can still operate together but the differences would be resolved at the bus interconnection of the devices. For example, in an ECC system, the two devices might place the ECC data in different locations within the data bus, which can be resolved by interconnecting the devices to the bus for aligning these signals as necessary.

The programmable parameters associated with the status interface are listed in Table 17, “Status Interface Programmable Parameter”.

TABLE 17. *Status Interface Programmable Parameter*

Parameter	Defined By	Description
dfi$_{\textbf{data\_bit\_enable}}$	MC/PHY	Defines the bits that are used for transferring valid data on both the **dfi_wrdata** and **dfi_rddata** buses. This parameter is defined by both the MC and the PHY. The parameter can have different values for different operating modes for each device. The parameter width is defined as the DFI data width.
phy$_{\textbf{freq\_range}}$	PHY	Defines the range of frequency values supported by the PHY. The frequency range must be a number between 1 and 32 inclusive and must start from the value of ZERO to the defined range. The PHY may only support a subset of these frequencies and the PHY must clearly define supported encodings.

3.6 DFI Training Interface

The DFI training interface enables increased accuracy at higher speeds in the alignment of critical timing signals on DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs; the interface includes signals, timing parameters and programmable parameters.

The DRAM type and system configuration determine the types of training available to a system; a system may or may not utilize each type of training. If training is supported, the system may utilize DFI training or support a different training method. There are four training operations; the first two operations (gate training and read data eye training) are collectively referred to as “read training”.

1. Gate training, used by DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.
2. Read data eye training, used by DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 DRAMs.
3. Write leveling, used by DDR3, DDR4, LPDDR3 and LPDDR4 DRAMs.
4. CA training, only applicable to the MCs and PHYs that support LPDDR3 and LPDDR4 DRAMs.

Support for each training operation is enabled or disabled through the corresponding programmable parameter and the enable may be implemented as programmable registers within the device.

The read training, write leveling and CA training signals that communicate from the MC to the PHY are multiply driven inside the MC to allow a direct connection from the MC to each PHY data slice and the signals must be driven with the same value.

The read training, write leveling and CA training signals that communicate from the PHY to the MC may be individually driven by each PHY data slice or collectively driven as a single signal.

More information on the training interface is provided in Section 4.12.1, “Training Operations in DFI Training Mode”. The signals, timing parameters and programmable parameters for the training interface are listed in Table 18, “Training Interface Signals”. The timing parameters associated with the training interface are listed in Table 19, “Training Interface

Timing Parameters”. The programmable parameters associated with the training interface are listed in Table 20, “Training Interface Programmable Parameters”.

The MC and the PHY can be synchronized with the defined training sequences, and the sequence required for each training operation, through the encoding of the **dfi_lvl_pattern** signal. For information on **dfi_lvl_pattern** encoding, refer to Section 4.12.4.3, “dfi_lvl_pattern Encoding”. For information on multiple training patterns used with LPDDR2, LPDDR3 and LPDDR4, refer to Table 44, “Data Calibration Pattern”.

Long or short training sequences can be defined with the **dfi_lvl_periodic** signal. For information on the **dfi_lvl_periodic** signal, refer to Section 3.6.7, “Periodic Training Flag”.

All sequences described in this document refer to DFI Training Modes. Training sequences can be run outside of these in “ PHY Independent mode” where they operate independently of the MC after initialization has completed. PHY Independent mode will not be detailed in this specification. For DFI compliance, the MC and the PHY may support either the =“PHY-Independent Mode” or “DFI Training Mode” for the applicable training operations where the mode is set per operation.

3.6.1 Read Training Operation

For read training in DFI training mode, the MC sets up the DRAM for gate training or read data eye training and periodically issues read commands. The PHY evaluates the data returned from the commands and adjusts the gate training and read data eye training capture timing accordingly.

Multiple training sequences can be defined, with either the MC or the PHY supplying the sequence information. In DFI training mode, the PHY is responsible for adjusting the delays and evaluating the responses from the DRAM. Therefore, it is the PHY, not the MC, which determines the sequences necessary to accomplish training. The MC must provide flexibility to run sequences required by the PHY and should not dictate a set of sequences.

A read training sequence can be initiated from the MC or the PHY. In both cases, the MC asserts a bit (or bits depending on the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$ and/or $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$ parameter) of the **dfi_rdlvl_en** signal to initiate or accept the training sequence, and the MC generally holds the enable signal asserted until the current training operation completes. During read training, **dfi_rddata_cs** indicates the chip select that is currently being trained. During read training, the **dfi_rddata_valid** and **dfi_rddata** signals are ignored. The MC may de-assert the set bit (or bits) of the **dfi_rdlvl_en signal** to disconnect the handshake through the disconnect protocol.

For more information on **dfi_rddata_cs**, refer to Section 3.3, “Read Data Interface” and Table 10, “Read Data Signals”.

In DDR4, LPDDR2 and LPDDR3 systems, the MC drives the **dfi_lvl_pattern** signal to the required training sequence. For LPDDR4 systems, the **dfi_lvl_pattern** signal provides an index to the required training sequence. The actual pattern that is associated with the index must be programmed or otherwise defined in both the MC and the PHY. New patterns are applicable only to read data eye training, not to gate training.

The **dfi_lvl_pattern** signal must be valid when any bit (or bits) of the **dfi_rdlvl_gate_en** (for read gate training) or **dfi_rdlvl_en** (for read data eye training) signal is asserted. The **dfi_lvl_pattern** signal may transition between training sequences when the **dfi_rdlvl_gate_en** or **dfi_rdlvl_en** signals are de-asserted. Due to a difference in the training patterns, the **dfi_lvl_pattern** signal definition is different for the different memory types that support this function.

All evaluations and delay changes are handled within the PHY. When the PHY finds the necessary edges and completes the read training, the PHY should drive the corresponding **dfi_rdlvl_resp** signal to ’b01, which informs the MC that the procedure is complete for that slice. The MC should then de-assert the set bit (or bits) of the **dfi_rdlvl_en** signal. When the **dfi_rdlvl_en** signal bit (or bits) de-asserts, the PHY should stop driving the **dfi_rdlvl_resp** signal.

For more information on read training, refer to Section 4.12, “DFI Training Operations”, Table 42, “DDR4 Encoding of dfi_lvl_pattern” and Table 44, “Data Calibration Pattern”.

3.6.2 Write Leveling Operation

The goal of write leveling is to locate the delay at which the write DQS rising edge aligns with the rising edge of the DRAM clock. By identifying this delay, the system can accurately align the write DQS within the DRAM clock.

A write leveling sequence can be initiated from the MC or the PHY. In both cases, the MC asserts a bit of the **dfi_wrlvl_en** signal to initiate or accept the leveling sequence, and the MC holds the corresponding enable signal bit asserted until the current training operation completes. During write leveling, **dfi_wrdata_cs** indicates the chip select that is currently being trained.

The signals used in write leveling are: **dfi_wrlvl_en**, **dfi_wrlvl_req** and **dfi_wrlvl_strobe**. The programmable parameters are $\mathbf{phy_{wrlvl\_en}}$ and $\mathbf{sys_{wrlvl\_strobe\_num}}$. The write leveling chip select signal is **dfi_phy_wrlvl_cs**.

For more information on write leveling signals, refer to Section 4.12.5, “Write Leveling”.

3.6.3 Write DQ Training Operation

The goal of write DQ training is to optimally place the clock in the center of the write DQ. A write leveling sequence can be initiated from the MC or the PHY. In both cases, the MC asserts a bit on the **dfi_wdqlvl_en** signal to initiate or accept the leveling sequence, and the holds the enable signal bit asserted until the current training operation completes. During write DQ training, **dfi_wrdata_cs** indicates the chip select that is currently being trained.

The signals used in write DQ training are: **dfi_wdqlvl_done**, **dfi_wdqlvl_en**, **dfi_wdqlvl_req**, **dfi_wdqlvl_resp** and **dfi_wdqlvl_result**. The programmable parameter is $\mathbf{phy_{wdqlvl\_bst}}$. The write leveling chip select signal is **dfi_phy_wdqlvl_cs**.

Write DQ training is new for DFI 4.0 and will be unused unless connecting a DFI 4.0 MC with a DFI 4.0 PHY.

For more information on write DQ training signals, refer to Section 4.12.8, “Write DQ Training”.

3.6.4 CA Training Operation

CA training supports increased frequency and the double-data rate address and command interface of LPDDR3 and the single data rate address and command interface of LPDDR4.

During CA training, the PHY may selectively not utilize any of the MC signals defined in the protocol, however the PHY must drive all signals to the MC, as defined. Some of the CA training signals from the MC to the PHY are multiply driven by the MC to allow a direct connection from the MC to the PHY memory command logic and/or each of the PHY memory data slices. All multiply driven signals originating from the MC to the PHY are driven with identical values.

The signals specific to CA training are: **dfi_calvl_capture** (CA training capture), **dfi_phy_calvl_cs** (CA training chip select), **dfi_calvl_en** (PHY CA training logic enable), **dfi_calvl_req** (PHY-initiated CA training request), **dfi_calvl_resp** (CA training response) and **dfi_calvl_ca_sel** (CA foreground pattern strobe) described in Table 18, “Training Interface Signals”. For LPDDR4 memories, there are 4 additional signals for CA VREF training: **dfi_calvl_strobe** (CA DQS strobe), **dfi_calvl_data** (DRAM VREF training value), **dfi_calvl_done** (VREF testing complete) and **dfi_calvl_result** (VREF testing result).

Signals modified to support CA training are: **dfi_cke** (DFI clock enable) and **dfi_cs** (DFI chip select). Both signals are described in Table 5, “Control Signals”.

CA training parameters $\mathbf{phy}_{\mathbf{calvl\_en}}$ (PHY CA training mode), $\mathbf{t}_{\mathbf{calvl\_capture}}$ (CA training capture delay from command), $\mathbf{t}_{\mathbf{calvl\_en}}$ (CA training enable time), $\mathbf{t}_{\mathbf{calvl\_max}}$ (CA training maximum time), $\mathbf{t}_{\mathbf{calvl\_resp}}$ (CA training response), $\mathbf{t}_{\mathbf{calvl\_cc}}$ (CA training chip select to chip select delay), $\mathbf{t}_{\mathbf{calvl\_cs\_ca}}$ (CA to CS delay), $\mathbf{t}_{\mathbf{calvl\_ca\_sel}}$ (CA_SEL Width), $\mathbf{t}_{\mathbf{calvl\_strobe}}$ (CA training setup time delay) and $\mathbf{t}_{\mathbf{calvl\_data}}$ (Data to data delay) are described in Table 20, “Training Interface Programmable Parameters”.

The $\mathbf{phy}_{\mathbf{calvl\_en}}$ programmable parameter specifies the PHY operating mode. When the parameter is asserted, DFI CA training is enabled in the PHY. When the parameter is de-asserted, DFI CA training is disabled in the PHY. The MC must support the $\mathbf{phy}_{\mathbf{calvl\_en}}$ parameter in both asserted and de-asserted modes; the PHY may support the $\mathbf{phy}_{\mathbf{calvl\_en}}$ parameter in one or both modes.

3.6.5 Training Interface Signals and Parameters

The signals and parameters in the training interface are listed in Table 18, “Training Interface Signals”, Table 19, “Training Interface Timing Parameters”, and Table 20, “Training Interface Programmable Parameters”.

For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_calvl_ca_sel or **dfi_calvl_ca_sel_pN**	MC	Defined by the $\mathbf{t}_{\mathbf{calvl\_ca\_sel}}$ timing parameter	0x0	CA foreground pattern strobe. This strobe indicates when the foreground pattern should be transmitted from the PHY to the DRAM. The timing from **dfi_calvl_ca_sel** to **dfi_cs** and the width of **dfi_calvl_ca_sel** is defined by DFI timing parameters.
dfi_calvl_capture	MC	DFI CA Training MC I/F Width	0x0	CA training capture. This signal is asserted for one DFI clock pulse; initiates the capture of the CA bus value from the DQ bus within the PHY. The capture pulse is asserted $\mathbf{t}_{\mathbf{calvl\_capture}}$ cycles after the calibration command is driven on **dfi_cs**.
dfi_calvl_data or **dfi_calvl_data_pN**	MC	7 bits per slice	- [a]	CA VREF training data. Defined per slice. Each slice receives a 7-bit DFI bus. The signal sets the value to transfer to DRAM for VREF training. The bit order and slice location of this signal is determined by the PHY. The PHY may choose to ignore this value altogether.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_calvl_done	MC	1 bit	0x0	CA VREF training complete. Completion of the MC iterating across different VREF training values. When the MC asserts this signal, the PHY should respond with a **dfi_calvl_resp** value of ’b01 or ’b11. The ’b01 value will indicate that the PHY requires more iterations. In this case, the MC should loop back to the first data value and iterate through the values again. If the PHY responds by driving the **dfi_calvl_resp** value of ’b11, training has completed. The **dfi_calvl_done** signal is valid when any bit in the **dfi_calvl_en** signal is asserted and should be unchanged during the time that the bit in the **dfi_calvl_en** signal is asserted. The signal is invalid when the corresponding bit of the **dfi_calvl_en** = ’b0. If the MC does not have anything to iterate on, **dfi_calvl_done** can be driven to a “1” all the time.
dfi_calvl_en	MC	DFI CA Training MC I/F Width	0x0	PHY CA training logic enable. This signal is asserted during CA training and can be used by the PHY to enable the associated logic to execute training. If the PHY initiated the CA training request (**dfi_calvl_req**), this serves as acknowledgment of the request. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice. • ’b1 = CA training enabled for this slice. The assertion of this signal immediately initiates the CA training process.
dfi_calvl_req	PHY	DFI CA Training PHY I/F Width	0x0	PHY-initiated CA training request. This is an optional signal for the PHY; other sources may be used to initiate CA training or the MC may initiate CA training independently. If the PHY asserts any bit of the **dfi_calvl_req** signal, the MC must acknowledge this request by asserting the corresponding bit of the **dfi_calvl_en** signal within $\mathbf{t_{calvl\_resp}}$ cycles, after which the PHY should de-assert the **dfi_calvl_req** signal bit. The PHY is not required to assert this signal during initialization or a frequency change operation. However, if the PHY does assert the request during a frequency change, the **dfi_calvl_req** signal must be asserted before or coincident with the assertion of the **dfi_init_complete** signal. There is one bit per slice.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_calvl_resp	PHY	2 bits	0x0	CA training response. This signal indicates that the PHY has completed the current CA training routine. The training sequence may consist of one or more training routines, and in the case of LPDDR3, to each of the two CA segments. The response is defined as follows: • ’b00 = Training in process or no action • ’b01 = The meaning of this setting is dependent on the memory type. • LPDDR3: Reserved • LPDDR4: Indicates that the current VREF sequence has completed. If the **dfi_calvl_done** signal is asserted when this value is driven on the **dfi_calvl_resp** signal, the MC should start from the initial VREF value. • ’b10 = The meaning of this setting is dependent on the memory type. • LPDDR3: Done with current training. Change the segment. • LPDDR4: Reserved • ’b11 = Training Complete For LPDDR3 memories, the **dfi_calvl_resp** signal is asserted during both phase 1 CA training (enabled with MR41) and phase 2 CA training (enabled with MR48). Removed since this signal is always 2 bits.
dfi_calvl_result	PHY	1 bit	0x0	VREF training result from PHY. Indicates how the current VREF value being driven by the MC compares with all previous values. • ’b0 = Current value is worse than or equal to the previous best value. • ’b1 = Current value is better than the previous best value. The **dfi_calvl_result** signal is valid when the **dfi_calvl_resp** signal is asserted (non-ZERO) and should be unchanged when the **dfi_calvl_resp** signal is asserted. This signal is not valid when the **dfi_calvl_resp** = ’b00.
dfi_calvl_strobe or **dfi_calvl_strobe_pN**	MC	DFI Data Slice Count	0x0	CA VREF training strobe. Defined per slice. Each assertion of 1 DFI clock generates a pulse of DQS to the DRAM. Asserting for multiple clocks generate the same number of consecutive DQS pulses. This signal interfaces to the data slices.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_lvl_pattern	MC	4 bits x DFI Read Training PHY I/F Width	0x0	Training pattern. Used to determine which training pattern should be run with the current read training operation. This is used with both gate training and read data eye training. For DDR4, LPDDR2 and LPDDR3 DRAMs, **dfi_lvl_pattern** also defines the pattern used for read training. • ’b0 = Issue 1 MRR command to MR32. • ’b1 = Issue 1 MRR command to MR40. For LPDDR4 DRAMs, **dfi_lvl_pattern** provides an index to the pattern used for read training. The detailed use of this signal is defined in Section 4.12.4.3, “dfi_lvl_pattern Encoding”. LPDDR4 also supports read training using a scratch pad FIFO within the DRAM. For more information on this functionality, refer to Section 4.12.8, “Write DQ Training”.
dfi_lvl_periodic	MC	DFI Leveling PHY I/F Width	0x0	Training length indicator (full or periodic). Global signal used for all training interfaces, which dictates the type of training required. • ’b0 = Full training required; long sequence used during initialization and for larger variations. • ’b1 = Periodic training required; shorter sequence used for small variations.
dfi_phy_calvl_cs	PHY	DFI Physical Rank Width[b] (verify this footnote will become b)	-[c] (adeela: verify this footnote later)[c]	CA training chip select. Indicates the target chip select associated with the current set bit of the **dfi_calvl_req** signal. This signal is only valid when the corresponding **dfi_calvl_req** bit is asserted. Only a single bit can be asserted during the request. If no bits are asserted, the MC determines which chip select(s) to target for the training request. The polarity of this signal is the same as the polarity of the **dfi_cs** signal. The PHY must not assert bits with $\mathbf{sys}_{\mathbf{cs\_state}}$ set to 'b0 (inactive chip selects).
dfi_phy_rdlvl_cs	PHY	DFI Physical Rank Width x DFI Read Training PHY I/F Width	-[c]	Read training chip select for read data eye training. Indicates the target chip select associated with the current **dfi_rdlvl_req** signal bit assertion. This signal is only valid when any bit of the **dfi_rdlvl_req** is asserted. Only a single bit can be asserted during the request. If no bits are asserted, the MC determines which chip select(s) to target for the training request. The polarity of this signal is the same as the polarity of the **dfi_cs** signal. The PHY must not assert bits with $\mathbf{sys}_{\mathbf{cs\_state}}$ set to 'b0 (inactive chip selects).

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_phy_rdlvl_gate_cs	PHY	DFI Physical Rank Width x DFI Read Training PHY I/F Width	-[c]	Read training chip select for gate training. Indicates the target chip select associated with the current **dfi_rdlvl_gate_req**. This signal is only valid when **dfi_rdlvl_gate_req** is asserted. Only a single bit can be asserted during the request. If no bits are asserted, the MC determines which chip select(s) to target for the training request. The polarity of this signal is the same as the polarity of the **dfi_cs** signal. The PHY must not assert bits with $\mathbf{sys}_{\mathbf{cs\_state}}$ set to 'b0 (inactive chip selects).
dfi_phy_wdqlvl_cs	PHY	DFI Physical Rank Width x DFI Write Leveling PHY I/F Width	-[c]	Write DQ training chip select. Indicates the target chip select associated with the current **dfi_wdqlvl_req**. This signal is only valid when a bit of the **dfi_wdqlvl_req** is asserted. Only a single bit can be asserted during the request. If no bits are asserted, the MC determines which chip select(s) to target for the training request. The polarity of this signal is the same as the polarity of the **dfi_cs** signal. The PHY must not assert bits with $\mathbf{sys}_{\mathbf{cs\_state}}$ set to 'b0 (inactive chip selects).
dfi_phy_wrlvl_cs	PHY	DFI Physical Rank Width x DFI Write Leveling PHY I/F Width	-[c]	Write leveling chip select. Indicates the target chip select associated with the current **dfi_wrlvl_req**. This signal is only valid when **dfi_wrlvl_req** is asserted. Only a single bit can be asserted during the request. If no bits are asserted, the MC determines which chip select(s) to target for the training request. The polarity of this signal is the same as the polarity of the **dfi_cs** signal. The PHY must not assert bits with $\mathbf{sys}_{\mathbf{cs\_state}}$ set to 'b0 (inactive chip selects).
dfi_rdlvl_done	MC	DFI Data Slice Count	0x0	Read training complete. The MC completes iterating across different training values (final training pattern, etc.). If the PHY completes the training sequence with **dfi_rdlvl_resp** = ’b11, training is complete. If the PHY completes the training sequence with **dfi_rdlvl_resp** = ’b01, training continues and the MC should restart with the beginning value and drive **dfi_rdlvl_done** accordingly. The **dfi_rdlvl_done** signal is valid when a bit (or multiple bits based on the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$ and/or $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$ parameter) of the **dfi_rdlvl_en** is asserted and should not be changed while that bit’s **dfi_rdlvl_en** = ’b01. The signal is invalid when **dfi_rdlvl_en** = ’b0. This signal is only used for read data eye training, not gate training.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_rdlvl_en	MC	DFI Read Leveling MC I/F Width	0x0	PHY read data eye training logic enable. This signal is asserted during read data eye training. If the PHY initiated the read data eye training request for a slice (**dfi_rdlvl_req**), this serves as an acknowledge of that request. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice • ’b1 = Read Data eye training logic enabled for this slice. The assertion of this signal immediately initiates read data eye training. If multiple slices must be enabled together, the user can specify the grouping in the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$ and/or $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$ parameter. This signal may enable specific training logic within the PHY. If any bit (or bits) of the **dfi_rdlvl_en** signal is (are) asserted, the MC and the PHY must reset their DFI read data word pointers for that slice to 0 on de-assertion of this signal. For more information, refer to Section 4.9.4, “Read Data Interface in Frequency Ratio Systems”.
dfi_rdlvl_gate_en	MC	DFI Read Leveling MC I/F Width	0x0	PHY gate training logic enable. This signal is asserted during gate training. If the PHY initiated the gate training request (**dfi_rdlvl_gate_req**), this serves as an acknowledge of that request. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice • ’b1 = Gate training logic enabled for this slice. The assertion of this signal immediately triggers gate training for this slice. If the **dfi_rdlvl_gate_en** signal is asserted for a slice, the MC and the PHY must reset their DFI read data word pointers to 0 on de-assertion of this signal. For more information, refer to Section 4.9.4, “Read Data Interface in Frequency Ratio Systems”.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_rdlvl_gate_req	PHY	DFI Read Leveling PHY I/F Width	0x0	PHY-initiated gate training request. The PHY initiates gate training, other sources may be used to initiate gate training, or the MC may initiate gate training independently. If the PHY asserts any bits of the **dfi_rdlvl_gate_req** signal, the MC must acknowledge the request by asserting the corresponding **dfi_rdlvl_gate_en** signal bit within $\mathbf{t_{rdlvl\_resp}}$ cycles, after which the PHY should de-assert the set bit of the **dfi_rdlvl_gate_req** signal. The PHY should not assert any bit of the **dfi_rdlvl_gate_req** during initialization and frequency change operations because the MC is responsible for gate training during these operations. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice • ’b1 = Gate training logic enabled for this slice. The assertion of this signal immediately triggers gate training for this slice.
dfi_rdlvl_req [a]	PHY	DFI Read Leveling PHY I/F Width	0x0	PHY-initiated read data eye training request. The PHY initiates read data eye training, other sources may be used to initiate read data eye training, or the MC may initiate read data eye training independently. If the PHY asserts any bit of the **dfi_rdlvl_req** signal for any slices that are in the $\mathbf{mc_{rdlvl\_slice\_group[n]}}$ and/or $\mathbf{phy_{rdlvl\_slice\_group[n]}}$, the MC must acknowledge the request by asserting all the corresponding bits of the **dfi_rdlvl_en** signal within $\mathbf{t_{rdlvl\_resp}}$ cycles, after which the PHY should de-assert the bit of the **dfi_rdlvl_req** signal. Note that the MC is not required to respond to each individual per-slice **dfi_rdlvl_req** signal concurrently. In addition, the PHY must initiate training early enough for the following situations: • For allowing for larger read training intervals in LRDIMM/ RDIMM systems • In cases where the MC does not respond to each individual per-slice **dfi_rdlvl_req** signal concurrently in LRDIMM/ RDIMM systems • In cases where the MC does not respond to each individual per-slice **dfi_rdlvl_req** signal concurrently NOTE: The MC must not assert a **dfi_rddata_en** signal for slices that do not assert **dfi_rdlvl_en** signals. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice • ’b1 = Read data eye training logic enabled for this slice. The assertion of this signal immediately triggers gate training for this slice.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_rdlvl_resp	PHY	DFI Read Training Response Width	0x0	Read training response. Indicates that status of the PHY training operation. • ’b00 = Training in process or no action • ’b01 = The meaning of this setting is dependent on the read training operation being performed. • Read gate training: Reserved • Read data eye training: Indicates that the current sequence has completed. If the **dfi_rdlvl_done** signal is asserted when this value is driven on the **dfi_rdlvl_resp** signal, the MC should start from the initial value. • ’b10 = Reserved • ’b11 = Training Complete When training is incomplete and the PHY responds with ’b01, the MC performs the following actions: • De-asserts the bit (or bits) of the **dfi_rdlvl_en** signal. • Sets up the next sequence. • Re-asserts the bit (or bits) of the **dfi_rdlvl_en** signal.
dfi_wdqlvl_done	MC	DFI Data Slice Count	0x0	Write DQ training complete. The MC completes iterating across different VREF training values. If the PHY completes with **dfi_wdqlvl_resp** = ’b11, training is complete. If the PHY completes the training sequence with **dfi_wdqlvl_resp** = ’b01, training continues and the MC should restart with the beginning VREF value and drive **dfi_wdqlvl_done** accordingly. The **dfi_wdqlvl_done** signal is valid when any bits of signal **dfi_wdqlvl_en** is asserted and should not be changed while the **dfi_wdqlvl_done** signal reports a value of [0]. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice. • ’b1 = CA training enabled for this slice. The assertion of this signal immediately initiates the CA training process.The signal is invalid when all bits of the **dfi_wdqlvl_en** = ’b0.
dfi_wdqlvl_en	MC	DFI Write DQ training MC I/F Width	0x0	PHY write DQ training logic enable. This signal is asserted during write DQ training. If the PHY initiated the write DQ training request (**dfi_wdqlvl_req**), this serves as an acknowledge of that request. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice. • ’b1 = Write DQ training enabled. The assertion of this signal initiates write DQ training.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_wdqlvl_req [a]	PHY	DFI Write DQ training PHY I/F Width	0x0	PHY write DQ leveling request. This is an optional signal for the PHY; other sources may be used to initiate write DQ training, or the MC may initiate write DQ training independently. The PHY may drive independent write DQ training requests from each data slice; however the MC must write train all data slices based on a single assertion of a single bit of the **dfi_wdqlvl_req** signal. If the PHY asserts a bit out of the **dfi_wdqlvl_req** signal, the MC must acknowledge the request by asserting the corresponding bit in the **dfi_wdqlvl_en** signal within $\mathbf{t_{wdqlvl\_resp}}$ cycles, after which the PHY should de-assert the **dfi_wdqlvl_req** signal bit.
dfi_wdqlvl_resp	PHY	DFI Write DQ Training Response Width	0x0	Write DQ training response. Indicates that the PHY has completed write DQ training. • ’b00 = Training in process or no action • ’b01 = Done with current VREF sequence. MC should update VREF value if applicable. If the **dfi_wdqlvl_done** signal is also asserted, MC should restart from initial value. • ’b10 = Reserved • ’b11 = Training complete • When training is incomplete and the PHY responds with ’b01, the MC performs the following actions: • De-asserts a bit of the **dfi_wdqlvl_en**. • Sets up the next VREF value. • Re-asserts **dfi_wdqlvl_en** signal bit.
dfi_wdqlvl_result	PHY	1 bit	0x0	Write DQ training result. Indicates how the current VREF value being driven by the MC compares with all previous values. • ’b0 = Current value is worse than or equal to the previous best value. • ’b1 = Current value is better than the previous best value. The **dfi_wdqlvl_result** signal is valid when **dfi_wdqlvl_resp** is asserted (non-ZERO) and should be unchanged when **dfi_wdqlvl_resp** is asserted. This signal is not valid when **dfi_wdqlvl_resp** = ’b00.
dfi_wrlvl_en	MC	DFI Write Leveling MC I/F Width	0x0	PHY write leveling logic enable. This signal is asserted during write leveling. If the PHY initiated the write leveling request (**dfi_wrlvl_req**), this serves as an acknowledge of that request. There is one bit per slice where, for each bit: • ’b0 = Normal operation for this slice • ’b1 = Write leveling enabled for this slice. The assertion of this signal initiates write leveling.

TABLE 18. *Training Interface Signals*

Signal	From	Width	Default	Description
dfi_wrlvl_req [a]	PHY	DFI Write Leveling PHY I/F Width	0x0	PHY write leveling request. This is an optional signal for the PHY; other sources may be used to initiate write leveling, or the MC may initiate write leveling independently. The PHY may drive independent write leveling requests from each data slice. If the PHY asserts a bit of the **dfi_wrlvl_req** signal, the MC must acknowledge the request by asserting the corresponding bit of the **dfi_wrlvl_en** signal within $\mathbf{t_{wrlvl\_resp}}$ cycles, after which the PHY should de-assert the **dfi_wrlvl_req** signal bit.
dfi_wrlvl_resp	PHY	DFI Write Leveling Response Width	0x0	Write leveling response. Indicates that the PHY has completed write leveling. The width is generally defined as a bit-per-PHY data slice with this encoding: • ’b0 = Writ e leveling in process or not active • ’b1 = Write leveling complete
dfi_wrlvl_strobe or **dfi_wrlvl_strobe_pN**	MC	DFI Data Slice Count	0x0	Write leveling strobe. Defined per slice. Asserts for one or more DFI clocks; initiates the capture of the write level response from the DQ bus within the PHY. The number of DFI clocks that the write leveling strobe is asserted for on each write leveling request a is specified by the $\mathbf{sys_{wrlvl\_strobe\_num}}$ parameter.

a. This signal is not meaningful during initialization; no default value is required.

b. In a 3DS solution control, MRW and training activity are limited to Physical Ranks.

c. Default value is the inactive state of this signal as defined by the memory type.

The timing parameters that are associated with the training interface are listed in Table 19, “Training Interface Timing Parameters”. All timing parameters are defined only once for the interface and must apply to all PHY data slices.

TABLE 19. *Training Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{calvl\_ca\_sel}}$	PHY	0x1	- [a]	PHY DFI clock cycles	In a system that uses frequency ratio, this timing parameter defines the width that is generated by combining the per-phase PHY inputs.
$\mathbf{t_{calvl\_capture}}$	PHY	0x1	- [a]	DFI clock cycles	Specifies the number of DFI clock cycles after the controller transmits the calibration command on **dfi_cs** before the **dfi_calvl_capture** pulse is driven.
$\mathbf{t_{calvl\_cc}}$	PHY	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles from one calibration command to the next calibration command.
$\mathbf{t_{calvl\_cs\_ca}}$	PHY	0x0	- [b]	PHY DFI clock cycles	Number of PHY DFI clocks from **dfi_calvl_ca_sel** assertion to **dfi_cs**.

TABLE 19. *Training Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{\text{calvl\_data}}$	System	0x1	-[a]	PHY	Minimum number of DFI PHY clocks from **dfi_calvl_data** assertion to **dfi_calvl_data** de-assertion that are required for meeting hold time at the DRAM.
$t_{\text{calvl\_en}}$	PHY	0x1	-[a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles from the assertion of a bit in the **dfi_calvl_en** signal to the de-assertion of the **dfi_cke** signal.
$t_{\text{calvl\_max}}$	MC	0x1	-[a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the MC will wait for a response (**dfi_calvl_resp**) to a bit of the CA training enable signal (**dfi_calvl_en)** being asserted.
$t_{\text{calvl\_resp}}$	MC	0x1	-[a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles after a CA training request is asserted for any slice (**dfi_calvl_req**) to when the MC will respond with the corresponding bit of the CA training enable signal (**dfi_calvl_en**).
$t_{\text{calvl\_strobe}}$	System	0x0	-[b]	PHY	Minimum number of DFI PHY clocks from **dfi_calvl_data** to **dfi_calvl_strobe** that are required for meeting setup time at the DRAM.
$t_{\text{rdlvl\_en}}$	PHY	0x1	-[a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles from the assertion of a bit (or bits) of the **dfi_rdlvl_en** or **dfi_rdlvl_gate_en** signal to the first read (DDR3 or DDR4) or mode register read (LPDDR2 or LPDDR3) command.
$t_{\text{rdlvl\_max}}$	MC	0x1	-[a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the MC waits for a response (**dfi_rdlvl_resp**) to a bit (or bits) of either of the read training enable signals (**dfi_rdlvl_en** or **dfi_rdlvl_gate_en**) being set.
$t_{\text{rdlvl\_resp}}$	MC	0x1	-[a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles after a read training request is asserted for any bit (**dfi_rdlvl_req** or **dfi_rdlvl_gate_req**) to when the MC responds with the corresponding bit (or bits) of the read training enable signal (**dfi_rdlvl_en** or **dfi_rdlvl_gate_en**) asserted. If there are no read or write transactions occurring on the interface, exceeding the response time is not an error condition. However, the MC must execute the requested training operation prior to restarting traffic.
$t_{\text{rdlvl\_rr}}$	PHY	0x1	-[a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles after the assertion of a read command to the next read command. • For DDR3, references a read data command. • For DDR4, references a MRS read command. • For LPDDR2 and LPDDR3 DRAMs, references a mode register read command. • For LPDDR4, references a MPC command.

TABLE 19. *Training Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
t_{wdqlvl_en}	PHY	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles from the assertion of any bit of the **dfi_wdqlvl_en** signal to the first write command (with **dfi_cs** asserted).
t_{wdqlvl_max}	MC	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the MC waits for a response (**dfi_wdqlvl_resp**) to a write DQ training enable signal (**dfi_wdqlvl_en**).
t_{wdqlvl_resp}	MC	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles after a write DQ training request is asserted for a single slice (**dfi_wdqlvl_req**) to when the MC responds with a write DQ training enable signal (**dfi_wdqlvl_en**). If there are no read or write transactions occurring on the interface, exceeding the response time is not an error condition. However, the MC must execute the requested training operation prior to restarting traffic.
t_{wdqlvl_rw}	PHY	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clocks from the last read in a calibration sequence to the first write in the next set of calibration commands. NOTE: The controller will issue read commands that correspond to the write commands for the PHY to check the success of the writes. The timing between the write and read commands are to be based on DRAM requirements. No DFI parameters are defined.
t_{wdqlvl_ww}	PHY	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clocks to be inserted between write commands during write DQ training. This timing parameter is applicable only when $phy_{wdqlvl\_bst} > 1$.
t_{wrlvl_en}	PHY	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles from the assertion of a bit in the **dfi_wrlvl_en** signal to the first assertion of the **dfi_wrlvl_strobe** signal.
t_{wrlvl_max}	MC	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the MC waits for a response (**dfi_wrlvl_resp**) to a bit set in the write leveling enable signal (**dfi_wrlvl_en**).
t_{wrlvl_resp}	MC	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles after a bit is set in the write leveling request signal (**dfi_wrlvl_req**) to when the MC responds with the corresponding write leveling enable signal (**dfi_wrlvl_en**) bit. If there are no read or write transactions occurring on the interface, exceeding the response time is not an error condition. However, the MC must execute the requested training operation prior to restarting traffic.
t_{wrlvl_ww}	PHY	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles after the assertion of the **dfi_wrlvl_strobe** signal to the next assertion of the **dfi_wrlvl_strobe** signal. The delay begins at the last rising clock edge for which the **dfi_wrlvl_strobe** signal is asserted.

[a] The minimum supportable value is 1; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

b. The minimum supportable value is 0; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

The programmable parameters associated with the training interface are listed in Table 20, "Training Interface Programmable Parameters".

TABLE 20. *Training Interface Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{mc}_{\mathbf{rdlvl\_slice\_group}}$	MC	Specifies a set of **dfi_rdlvl_en** signal bits that need to assert as a group. This parameter has single bit per slice. Groups defined by this parameter are mutually exclusive, and can be used to accomplish multiple pass training for RDIMM and LRDIMM. If the PHY asserts the **dfi_rdlvl_req** or **dfi_rdlvl_gate_req** with bit [n] asserted, the MC will respond by asserting the **dfi_rdlvl_en** signal for each slice as defined by the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$. As a result, completely independent leveling per slice is enabled by clearing each bit of $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$ to 'b0, and completely disabling leveling per slice (DFI 3.1 behavior) can be enabled by setting each bit of $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$ to 'b1. Slices can be grouped in the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group}}$ and/or $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group}}$ parameter. The $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group}}$ would specify pre-defined sets of **dfi_rdlvl_en** or **dfi_rdlvl_gate_en** signals that must be asserted as a group by the MC when the PHY asserts any of the corresponding bits of the **dfi_rdlvl_req** or **dfi_rdlvl_gate_req** signal. The $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group}}$ timing parameter indicates which slices are actually tied together for read training, while the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group}}$ indicates desired Controller behavior. Ideally, these two timing parameters should match for compatibility.
$\mathbf{phy}_{\mathbf{rdlvl\_slice\_group}}$	PHY	Specifies a set of **dfi_rdlvl_en** signal bits that need to assert as a group. This parameter has single bit per slice. Groups defined by this parameter are mutually exclusive, and can be used to accomplish multiple pass training for RDIMM and LRDIMM. If the PHY asserts the **dfi_rdlvl_req** or **dfi_rdlvl_gate_req** with bit [n] asserted, the MC will respond by asserting the **dfi_rdlvl_en** signal for each slice that is set in $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$. As a result, completely independent leveling per slice is enabled by clearing each bit of $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$ to 'b0, and completely disabling leveling per slice (DFI 3.1 behavior) can be enabled by setting each bit of $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$ to 'b1. Slices can be grouped in the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group}}$ and/or $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group}}$ parameter. The $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group}}$ would specify pre-defined sets of **dfi_rdlvl_en** or **dfi_rdlvl_gate_en** signals that must be asserted as a group by the MC when the PHY asserts any of the corresponding bits of the **dfi_rdlvl_req** or **dfi_rdlvl_gate_req** signal. The $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group}}$ timing parameter indicates which slices are actually tied together for read training, while the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group}}$ indicates desired Controller behavior. Ideally, these two timing parameters should match for compatibility.
$\mathbf{phy}_{\mathbf{calvl\_en}}$	PHY	Defines the CA training operating mode of the PHY. • 'b0 = DFI CA Training disabled • 'b1 = DFI CA Training enabled
$\mathbf{phy}_{\mathbf{rdlvl\_en}}$	PHY	Specifies whether the PHY is supporting read data eye training on the DFI bus. • 'b0 = Disabled • 'b1 = Enabled

TABLE 20. *Training Interface Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{phy_{rdlvl\_gate\_en}}$	PHY	Specifies whether the PHY is supporting gate training on the DFI bus. • ’b0 = Disabled • ’b1 = Enabled
$\mathbf{phy_{wdqlvl\_bst}}$	PHY	Specifies the number of consecutive write bursts that are issued during a training sequence. These are to be followed by the same number of consecutive read bursts for completing the training sequence. NOTE: The timing of the read and write bursts within a sequence defaults to back-to-back commands. No DFI parameter is defined.
$\mathbf{phy_{wrlvl\_en}}$	PHY	Specifies whether the PHY is supporting write leveling on the DFI bus. • ’b0 = Disabled • ’b1 = Enabled
$\mathbf{sys_{wrlvl\_strobe\_num}}$	MC	The minimum number clocks that the **dfi_wrlvl_strobe** signal is asserted during a write leveling training request. [a]

[a] The minimum supportable value is 1; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

3.6.6 dfi_lvl_pattern Encoding

To synchronize the MC and the PHY with the defined training sequences and the specific sequence required for each training operation, the MC transmits the **dfi_lvl_pattern** signal across the DFI bus to the PHY.

The **dfi_lvl_pattern** signal must be valid when the enable signal is asserted for gate training or read data eye training. Similar to the enable signal, the **dfi_lvl_pattern** signal may transition when training completes. For more information on **dfi_lvl_pattern** signal encoding, refer to Table 42, “DDR4 Encoding of dfi_lvl_pattern”.

3.6.7 Periodic Training Flag

The **dfi_lvl_periodic** control signal is a general purpose training signal that is relevant when the PHY can execute a long or short training sequence as needed.

The **dfi_lvl_periodic** signal defines the type of training required and determines if a full training sequence is required. The MC sets a single bit on the **dfi_lvl_periodic** signal to indicate to the PHY whether the system may require a longer training sequence (when initializing or re-starting from the self-refresh state), or whether the system only needs a short training (when in an operating state and simply tuning the delays).

The **dfi_lvl_periodic** signal is utilized with all training interfaces and all DRAM types that utilize training. The encoding of the **dfi_lvl_periodic** signal is defined in Table 18, “Training Interface Signals”.

3.6.8 PHY-Initiated Training in DFI Training Mode

The DFI Specification mandates that the MC respond to a PHY training request signal assertion within the period of time defined by $\mathbf{t_{calvl\_resp}}$, $\mathbf{t_{rdlvl\_resp}}$, $\mathbf{t_{wdqlvl\_resp}}$ or $\mathbf{t_{wrlvl\_resp}}$. However, this requirement may be ignored if there are no read or write transactions occurring on the interface as long as the MC does execute the requested training operation prior to restarting traffic. This allows for the condition in which training might be undesirable, such as when DRAM is in a self-refresh state.

In such a scenario, the PHY should hold the training request asserted until the MC response time has been reached. Once the timer has timed out, the PHY has the option to de-assert the per-slice request enable (**dfi_calvl_req**, **dfi_rdlvl_gate_req**, **dfi_rdlvl_req**, **dfi_wdqlvl_req** or **dfi_wrlvl_req**) or leave the request asserted; the MC must initiate the requested training prior to restarting data traffic.

3.7 Low Power Control Interface

The Low Power Control interface handles transmission of signals to enter and exit an idle state; the interface includes signals and timing parameters. Low Power Control is an optional feature for both the MC and the PHY unless the system requires a low power interface. It may be advantageous to place the PHY in a low power state when the MC has knowledge that the memory subsystem will remain in the idle state for a period of time. Depending on the state of the system, the MC communicates state information to the PHY allowing the PHY to enter the appropriate power saving state.

The Low Power Control interface consists of timing parameters and signals that inform the PHY of a low power mode opportunity, as well as how quickly the MC will require the PHY to resume normal operation. Table 21, “Low Power Control Interface Signals” describes the signals used in the low power control interface: **dfi_lp_ctrl_req** (low power opportunity control request), **dfi_lp_data_req** (low power opportunity data request), **dfi_lp_ack** (low power acknowledge) and **dfi_lp_wakeup** (low power wakeup time).

Once a low power request is asserted, the other low power request cannot be asserted until either the **dfi_lp_ack** signal is asserted or the low power request is aborted and a new low power request is generated. Once asserted, a low power request cannot be de-asserted unless the **dfi_lp_ack** is not asserted within $\mathbf{t_{lp\_resp}}$ or when exiting the low power handshake; when exiting, if both low power requests are asserted, both requests must be de-asserted simultaneously. Table 22, “Low Power Control Timing Parameters” describes the timing parameters: $\mathbf{t_{lp\_resp}}$ and $\mathbf{t_{lp\_wakeup}}$.

The signals associated with the low power interface are listed in Table 21, “Low Power Control Interface Signals”. More information on the low power control interface is provided in Section 4.13, “Low Power Control Handshaking”. For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 21. *Low Power Control Interface Signals*

Signal	From	Width	Default	Description
dfi_lp_ack	PHY	1 bit	0x0	Low power acknowledge. The **dfi_lp_ack** signal is asserted to acknowledge the MC low power opportunity request. The PHY is not required to acknowledge this request. If the PHY acknowledges the request, the **dfi_lp_ack** signal must be asserted within $\mathbf{t_{lp\_resp}}$ cycles after the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal assertion. Once asserted, this signal remains asserted until the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal de-asserts. The signal de-asserts within $\mathbf{t_{lp\_wakeup}}$ cycles after the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal de-asserts, indicating that the PHY is able to resume normal operation. If the PHY ignores the request, the **dfi_lp_ack** signal must remain de-asserted for the remainder of the low power mode opportunity. The **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is asserted for at least $\mathbf{t_{lp\_resp}}$ cycles.

TABLE 21. *Low Power Control Interface Signals*

Signal	From	Width	Default	Description
dfi_lp_ctrl_req	MC	1 bit	0x0	Low power opportunity control request. The **dfi_lp_ctrl_req** signal is used by the MC to inform the PHY of an opportunity to switch to a low power mode. When asserted, the MC indicates that no more commands will be sent on the Control Interface. The MC must assert a constant value on the **dfi_lp_wakeup** signal while this signal is asserted before the request is acknowledged by the PHY through the assertion of the **dfi_lp_ack** signal or until $\mathbf{t_{lp\_resp}}$ cycles have elapsed. The MC may increase the value of the **dfi_lp_wakeup** signal if both the **dfi_lp_ctrl_req** and **dfi_lp_ack** signals are asserted. Following the de-assertion of the **dfi_lp_ctrl_req** signal, the PHY has $\mathbf{t_{lp\_wakeup}}$ cycles to resume normal operation and de-assert the **dfi_lp_ack** signal.
dfi_lp_data_req	MC	1 bit	0x0	Low power opportunity data request. The **dfi_lp_data_req** signal is used by the MC to inform the PHY of an opportunity to switch to a low power mode. When asserted, the MC indicates that no more commands will be sent on the Data Interface. The MC must assert a constant value on the **dfi_lp_wakeup** signal while this signal is asserted before the request is acknowledged by the PHY through the assertion of the **dfi_lp_ack** signal or until $\mathbf{t_{lp\_resp}}$ cycles have elapsed. The MC may increase the value of the **dfi_lp_wakeup** signal if both the **dfi_lp_data_req** and **dfi_lp_ack** signals are asserted. Following the de-assertion of the **dfi_lp_data_req** signal, the PHY has $\mathbf{t_{lp\_wakeup}}$ cycles to resume normal operation and de-assert the **dfi_lp_ack** signal.

TABLE 21. *Low Power Control Interface Signals*

Signal	From	Width	Default	Description
dfi_lp_wakeup	MC	4 bits	- [a]	Low power wakeup time. The **dfi_lp_wakeup** signal indicates which one of the 16 wakeup times the MC is requesting for the PHY. The signal is only valid when the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is asserted. The **dfi_lp_wakeup** signal must remain constant until the **dfi_lp_ack** signal is asserted. Once the request has been acknowledged, the MC may increase the **dfi_lp_wakeup** signal, permitting the PHY to enter a lower power state. The PHY is not required to change power states in response to the wakeup time change. The MC may not decrease this value once the request has been acknowledged. The value of the **dfi_lp_wakeup** signal at the time that the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is de-asserted sets the $\mathbf{t_{lp\_wakeup}}$ time. • ’b0000 = $\mathbf{t_{lp\_wakeup}}$ is 16 cycles • ’b0001 = $\mathbf{t_{lp\_wakeup}}$ is 32 cycles • ’b0010 = $\mathbf{t_{lp\_wakeup}}$ is 64 cycles • ’b0011 = $\mathbf{t_{lp\_wakeup}}$ is 128 cycles • ’b0100 = $\mathbf{t_{lp\_wakeup}}$ is 256 cycles • ’b0101 = $\mathbf{t_{lp\_wakeup}}$ is 512 cycles • ’b0110 = $\mathbf{t_{lp\_wakeup}}$ is 1024 cycles • ’b0111 = $\mathbf{t_{lp\_wakeup}}$ is 2048 cycles • ’b1000 = $\mathbf{t_{lp\_wakeup}}$ is 4096 cycles • ’b1001 = $\mathbf{t_{lp\_wakeup}}$ is 8192 cycles • ’b1010 = $\mathbf{t_{lp\_wakeup}}$ is 16384 cycles • ’b1011 = $\mathbf{t_{lp\_wakeup}}$ is 32768 cycles • ’b1100 = $\mathbf{t_{lp\_wakeup}}$ is 65536 cycles • ’b1101 = $\mathbf{t_{lp\_wakeup}}$ is 131072 cycles • ’b1110 = $\mathbf{t_{lp\_wakeup}}$ is 262144 cycles • ’b1111 = $\mathbf{t_{lp\_wakeup}}$ is unlimited

[a] This signal is not meaningful during initialization; no default value is required.

The timing parameters associated with the low power interface are listed in Table 22, “Low Power Control Timing Parameters”.

TABLE 22. *Low Power Control Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{lp\_resp}}$	MC	0x1 [a]	0x7	DFI clock cycles	Specifies the maximum number of DFI clock cycles after the assertion of the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal to the assertion of the **dfi_lp_ack** signal.

TABLE 22. *Low Power Control Timing Parameters (Continued)*

Parameter	Defined By	Min	Max	Unit	Description
t_{lp_wakeup}	MC	16 (0x10)	-[b]	DFI clock cycles	Specifies the target maximum number of DFI clock cycles that the **dfi_lp_ack** signal may remain asserted after the de-assertion of the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal. The **dfi_lp_ack** signal may de-assert at any cycle after the de-assertion of the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal. Exceeding the maximum is not considered an error condition.

[a] It is recommended to fix this timing parameter at 7 cycles.

[b] The minimum supportable value is 0; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

3.8 Error Interface

The error interface handles the transmission of error information. The interface includes signals and timing parameters. The error interface is an optional feature for both the MC and the PHY.

In a DDR memory sub-system, the PHY may detect various error conditions including DRAM errors (e.g., ECC errors) or PHY-specific errors (e.g., loss of DLL lock or a read DQS error). In error-condition scenarios, it may be desirable to communicate the error information from the PHY to the MC for error reporting and other possible error responses. The MC is not required to take any action other than reporting errors.

The error interface defines two signals and a timing parameter. The PHY may support the **dfi_error** signal with or without **dfi_error_info**. The PHY may use a subset of the **dfi_error_info** signal. Signals that are not being driven must be tied LOW at the MC.

The signals associated with the error interface are listed in Table 23, “Error Signals”. The error signals are not phased for Frequency Ratio. For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 23. *Error Signals*

Signal	From	Width	Default	Description
dfi_error	PHY	DFI Error Width	0x0	DFI error. Indicates that the PHY has detected an error condition.
dfi_error_info	PHY	DFI Error Width x 4	0x0	DFI error source. Provides additional information about the source of the error detected. Only considered valid when **dfi_error** is asserted.

Typically, the width of the **dfi_error** signal would be equal to 1 bit per data slice plus 1 bit for control. The PHY may implement **dfi_error** as a single bit or any other width not exceeding the sum of the data slices + 1 bit for control. The MC should accept 1 bit per instance as defined by the sum of data slices plus 1 bit for control.

The **dfi_error_info** signal is 4 bits per instance. The number of instances should be the same for the **dfi_error** and **dfi_error_info** signals. The **dfi_error_info** signal is defined for some error types in this specification and may be further defined as design-specific errors by the PHY.

The error interface defines a maximum timing parameter, $\mathbf{t_{error\_resp}}$. The timing parameter defines the maximum delay between receiving a command / data and detection of an error associated with that command / data.

The timing parameter associated with the error interface is listed in Table 24, "Error Timing Parameter".

TABLE 24. *Error Timing Parameter*

Parameter	Defined By	Min	Max	Unit	Description
t_{error_resp}	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that may occur from the DFI bus transaction(s) which are known to be affected by the error condition and the assertion of the **dfi_error** signal.

a. The minimum supportable value is 1; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

3.9 DB Training Interface

The DB-to-DRAM training interface handles DB training in PHY evaluation mode; the interface includes signals and timing parameters. The DB interface is an optional feature for both the MC and the PHY and is only relevant when PHY evaluation mode is being used.

DDR4 LRDIMMs include a data buffer (DB) between the PHY and DRAM components. The DB will capture and re-drive the DQ/DQS bus to and from the DRAM and PHY. The DB-to-DRAM interface must be trained to accurately capture read data and drive write data and strobes. In PHY independent mode, this training is to occur without MC involvement. In PHY evaluation mode, the MC is to set up the training in the DB and DRAM and receive the training response.

The signals associated with the DB training interface are listed in Table 25, "DB Training Signals". For more information on which signals are required and which signals are optional, refer to Table 3, "DFI Signal Requirements".

TABLE 25. *DB Training Signals*

Signal	From	Width	Default	Description
dfi_db_train_en	MC	DFI Data Slice Count	0x0	DFI DB training enable. Enable training mode in the PHY. In this mode, the PHY performs the following tasks: • Transfers commands from the MC (commands will not use DQ/DQS bus) • Captures DQ inputs and transfer data to MC on response bus
dfi_db_train_resp or **dfi_db_train_resp_wN** [a]	PHY	DFI Data Slice Count x DQ per Slice	0x0	DFI DB training response. DQ data that is returned to MC for evaluation. All DQ bits are returned. DQ data is to be defined per-phase in frequency ratio systems.

All DB-to-DRAM training, which includes up to four training modes, uses these signals.

The DB training interface defines two timing parameters, $\mathbf{t_{phy\_db\_train\_en}}$ and $\mathbf{t_{phy\_db\_train\_resp}}$. The $\mathbf{t_{phy\_db\_train\_en}}$ timing parameter defines the minimum delay after the enable has been asserted that the MC can issue a training command, and the $\mathbf{t_{phy\_db\_train\_resp}}$ timing parameter defines the delay through the PHY.

The timing parameters associated with the DB training interface are listed in Table 26, “DB Training Timing Parameters”.

TABLE 26. *DB Training Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{phy\_db\_train\_en}}$	PHY	0x1	- [a]	PHY DFI clock cycles	Number of PHY DFI clocks after **dfi_db_train_en** asserts until the 1st training command can be issued.
$\mathbf{t_{phy\_db\_train\_resp}}$	PHY	0x1	- [a]	DFI clock cycles	Maximum number of DFI clocks required for transferring DQ data through the PHY. Specifically, this is the maximum number of DFI clocks from PHY DQ input to the PHY **dfi_db_train_resp** signal output.

a. The minimum supportable value is 1; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

3.10 PHY Master Interface

The PHY master interface provides a means for the PHY to take control of the DFI and DRAM buses, with the DRAM in a defined state. When the PHY has control of the bus, it performs all operations independent of the MC with or without accessing the DRAM. The PHY will request control and provide information on the desired state of the DRAMs when control is handed over. The MC will perform the requested action and then hand over control.

The PHY may request that the memories be placed in a specific state (such as idle or self-refresh) prior to being given control of the buses, or the PHY can indicate that the MC may transfer control in any of the supported states. The DFI bus must remain idle while the PHY is in control except that the MC may send memory refreshes if required as the MC will maintain the memory refresh timing for the system. The PHY will be responsible for forwarding all refresh information to the DRAMs before releasing the DFI bus back to the MC.

3.10.1 Inactive Chip Selects

When requesting control of the DFI/DRAM buses, the PHY will request that the memories be placed in a specific state. However, this only applies to active chip selects. Inactive chip selects are reflected by the associated $\mathbf{sys_{cs\_state}}$ bit being cleared to ’b0. When the PHY requests control of the bus, the MC should follow these guidelines for inactive chip selects:

- Maximum power saving mode (MPSM) or deep power-down (DPD) mode

  The MC should not unnecessarily bring the memory out of the low power mode to an IDLE or SREF state. In MPSM/DPD modes, the memory contents might not be maintained by the DRAM. The MC should retain the memory in its current state.
- Uninitialized memory

  The memory should be left uninitialized until the system determines otherwise.

- Chip select that is unpopulated or powered off

  Chip select should remain off-line until the system determines otherwise.

Since the **dfi_phymstr_req** / **dfi_phymstr_ack** signals are only a single-bit, and the PHY requires that the MC will not assert the acknowledge until all chip selects are in the requested state, it is possible that a long time may pass before the acknowledge can occur. This is particularly true for uninitialized chip selects or a chip select in MPSM or DPD, which may require a software intervention with a full initialization routine. This could result in a violation of the $\mathbf{t_{phymstr\_resp}}$ parameter. The behavior of the system on a tphymstr_resp violation is system-dependent and out of the scope of this document.

3.10.2 PHY Master Interface Signals and Parameters

The signals and parameters in the PHY master interface are listed in Table 27, “PHY Master Interface Signals”, Table 28, “PHY Master Interface Timing Parameters” and Table 29, “PHY Master Interface Programmable Parameter”.

For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

TABLE 27. *PHY Master Interface Signals*

Signal	From	Width	Default	Description
dfi_phymstr_ack	MC	1 bit	0x0	DFI PHY master acknowledge. When asserted, the MC places the DRAM in a known state (IDLE, self-refresh, or self-refresh power-down). When the **dfi_phymstr_ack** signal is asserted, the PHY is the master of DRAM bus. If required by the DRAM, the controller continues sending refresh commands on the DFI bus.

TABLE 27. *PHY Master Interface Signals*

Signal	From	Width	Default	Description
dfi_phymstr_cs_state	PHY	DFI Chip Select Width	0x0	DFI PHY master CS state. This signal indicates the state of the DRAM when the PHY becomes the master. Each memory rank uses one bit. • 'b0 = IDLE or self-refresh. The PHY specifies the required state, using the **dfi_phymstr_state_sel** signal. For LPDDR4, the self-refresh state is without power-down. • 'b1 = IDLE or self-refresh or self-refresh with power-down. The PHY does not specify the state; the MC can optionally choose any supported state. The MC closes all the pages prior to acknowledging the request from the PHY. This signal is valid only when the **dfi_phymstr_req** signal is asserted by the PHY and should remain constant while the **dfi_phymstr_req** signal is asserted. The self-refresh with power-down state is specific to LPDDR4. The **dfi_phymstr_cs_state** bit values are not relevant for chip selects with $\mathbf{sys}_{cs\_state}$ set to 'b0 (inactive chip selects). The MC can leave the chip selects with $\mathbf{sys}_{cs\_state}$ set to 'b0 in their current, inactive state, regardless of the corresponding **dfi_phymstr_cs_state** bit value. The PHY must not require these chip selects to be in IDLE or self-refresh states. The system must maintain a consistent, stable view of $\mathbf{sys}_{cs\_state}$ after **dfi_phymstr_req** is asserted to ensure synchronization between the MC and PHY.
dfi_phymstr_req	PHY	1 bit	0x0	DFI PHY master request. When asserted, the PHY requests control of the DFI bus. The systems must maintain a consistent, stable view of $\mathbf{sys}_{cs\_state}$ after **dfi_phymstr_req** is asserted for ensuring synchronization between the MC and PHY.
dfi_phymstr_state_sel	PHY	1 bit	0x0	DFI PHY master state select. Indication from the PHY to the MC whether the requested memory state is IDLE or self-refresh. If the per-CS **dfi_phymstr_cs_state** = 1, this signal does not apply for that chip select. The PHY does not place any requirement on the low power state of the memory, the state may be IDLE, self-refresh, or self-refresh with power-down. • If the per-CS **dfi_phymstr_cs_state** = 0, for that chip select: • ’b0 = The MC must place the memory on the associated CS in the IDLE state. • ’b1 = The MC must place the memory on the associated CS in the self-refresh state. If using LPDDR4 devices, the self-refresh state is without power-down. This signal is valid only when the **dfi_phymstr_req** signal is asserted by the PHY and should remain constant while that signal is asserted.

TABLE 27. *PHY Master Interface Signals*

Signal	From	Width	Default	Description
dfi_phymstr_type	PHY	2 bits	0x0	DFI PHY master request type. Indicates which one of the 4 types of PHY master interface times the **dfi_phymstr_req** signal is requesting. The value of the **dfi_phymstr_type** signal determines which one of the timing parameters ($\mathbf{t_{phymstr\_type0}}$, $\mathbf{t_{phymstr\_type1}}$, $\mathbf{t_{phymstr\_type2}}$, $\mathbf{t_{phymstr\_type3}}$) is relevant. The **dfi_phymstr_type** signal must remain constant during the entire time that the **dfi_phymstr_req** signal is asserted. The **dfi_phymstr_type** bit values are not relevant for chip selects with $\mathbf{sys_{cs\_state}}$ set to ’b0 (inactive chip selects). The MC can continue keeping the chip selects with $\mathbf{sys_{cs\_state}}$ set to ’b0 in their current, inactive state, regardless of the corresponding **dfi_phymstr_type** bit value. The PHY must not require these chip selects to be in IDLE or SREF states.

The timing parameters associated with the PHY master interface are listed in Table 28, “PHY Master Interface Timing Parameters”.

TABLE 28. *PHY Master Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{phymstr\_resp}}$	MC	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles after the **dfi_phymstr_req** signal asserts to the assertion of the **dfi_phymstr_ack** signal.
$\mathbf{t_{phymstr\_rfsh}}$	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the PHY requires for generating a refresh command to the DRAM after the PHY receives the refresh command from the MC.
$\mathbf{t_{phymstr\_type0}}$	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phymstr_req** signal may remain asserted after the assertion of the **dfi_phymstr_ack** signal for **dfi_phymstr_type** = 0x0. The **dfi_phymstr_req** signal may de-assert at any cycle after the assertion of the **dfi_phymstr_ack** signal.
$\mathbf{t_{phymstr\_type1}}$	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phymstr_req** signal may remain asserted after the assertion of the **dfi_phymstr_ack** signal for **dfi_phymstr_type** = 0x1. The **dfi_phymstr_req** signal may de-assert at any cycle after the assertion of the **dfi_phymstr_ack** signal.
$\mathbf{t_{phymstr\_type2}}$	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phymstr_req** signal may remain asserted after the assertion of the **dfi_phymstr_ack** signal for **dfi_phymstr_type** = 0x2. The **dfi_phymstr_req** signal may de-assert at any cycle after the assertion of the **dfi_phymstr_ack** signal.

TABLE 28. *PHY Master Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{phymstr\_type3}$	PHY	0x1	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_phymstr_req** signal may remain asserted after the assertion of the **dfi_phymstr_ack** signal for **dfi_phymstr_type** = 0x3. The **dfi_phymstr_req** signal may de-assert at any cycle after the assertion of the **dfi_phymstr_ack** signal.

[a] The actual value depends on the system.

The programmable parameter associated with the PHY master interface is listed in Table 29, “PHY Master Interface Programmable Parameter”.

TABLE 29. *PHY Master Interface Programmable Parameter*

Parameter	Defined By	Description
sys_{cs_state}	MC/PHY	Global parameter that defines the state of each chip select in the system. The width of sys_{cs_state} is DFI Chip Select Width, with each bit corresponding to a chip select in ascending order. • sys_{cs_state}[0]: State of chip select 0 • sys_{cs_state}[1]: State of chip select 1, etc. The value of each bit defines the current state of the corresponding chip select with the following encoding: • 'b0 = Inactive chip select that is in one of the following states: • Unpopulated, uninitialized, or powered off • Low power state where contents are not guaranteed for the DRAM, such as deep power-down mode or maximum power savings mode • 'b1 = Active chip select: An active chip select can be in an active, IDLE, or low power state for which the memory contents are maintained by the DRAM, MC, or both. The system must maintain a consistent, stable view of sys_{cs_state} between the PHY and MC for ensuring synchronization between the MC and PHY.

3.11 Definition of a Slice

Earlier versions of the DFI specification used the term “slice” to refer to a sub-set of the data interface, but the term was never defined explicitly. In MC’s, the term slice is generally used to define signal widths to fan out a signal from the MC to the PHY, or to define the number of signals that the PHY sends to the MC. However, in the PHY, data slices are generally individual components of the PHY data logic. These components operate independently; the DFI is defined to minimize slice to slice communication. Furthermore, the interface replicates data control signaling to allow point-to-point connectivity between the MC and PHY.

From a DFI perspective, all slices are expected to be identical but since the alignment of data to the interface is expected to be defined by the MC and the PHY, there is flexibility in the connection. Some systems do not have a data width that aligns to a slice boundary. But, since not all data bits are required to be connected, this is allowable. The

phy$_{\textbf{data\_slice\_width}}$ programmable parameter will resolve this issue and define how the MC and PHY place data so that any alignment issues can be resolved.

This parameter is defined by the PHY and specifies the width of a common slice component. It is acceptable for the PHY to have slices that are less than the defined slice width. In this scenario, the DFI data bit disables would be utilized to define which bits are not available.

The programmable parameter associated with the data slice definition is listed in Table 30, “DFI Data Slice Definition Programmable Parameters”.

TABLE 30. *DFI Data Slice Definition Programmable Parameters*

Parameter	Defined By	Description
phy$_{\textbf{data\_slice\_width}}$	PHY	PHY Data Slice Width. Defines the width of a common PHY data slice component. All slices are defined having the same width. Unused bits can be disabled by using the **dfi**$_{\textbf{data\_bit\_enable}}$ parameter.

The number of PHY slices is determined by dividing the DFI Data Width by the **phy**$_{\textbf{data\_slice\_width}}$ parameter. The MC and PHY must define the DFI Data Width of the device in terms of slices even if the device does not implement all of the signals. The unused signals should be clearly defined in terms of bit location by using the **dfi**$_{\textbf{data\_bit\_enable}}$ parameter. Otherwise, the number of slices might not be deterministic.

All DFI 4.0-compliant PHYs will need to define the DFI Slice Width, while all DFI 4.0-compliant MCs should define slice widths that are supported and adhere to the data width requirements for all DFI signals as defined by the DFI Slice Width. It is acceptable for the MC to drive more copies of fanned out signals than the PHY requires. These signals would be expected to be “no connects”.

The MC might have more input signals than defined for PHY-driven signals on the interface. The MC must be able to operate with these signals tied inactive at the MC. The DFI Data Width for the MC and PHY can be less than the width defined for the interface as previously discussed. In this case, the MC and/or PHY must clearly define how the data signals are to be connected to the slice-based interface. Any “no connects” to the interface should be clearly defined.

3.12 DFI Disconnect Protocol

The DFI specification defines several interface handshakes between the MC and the PHY. In some cases, the interface handshake can be terminated by only one of the devices, and the disconnect timing might be long or indeterminate. In some circumstances, it may be desirable for a device to disconnect the handshake. Some circumstances which might warrant a disconnect are:

- A high-priority operation that the system must continue to be fully operational
- To respond to an error condition even if the system stops being fully operational
- A timing or other such violation that would occur if the system did not disconnect in a reasonable time

When such a situation occurs, the disconnect should be relatively short, with a predictable disconnect time, leaving the system in a known state. The disconnect time is defined through multiple timing parameters, and the state is defined through the **dfi_disconnect_error** signal.

A MC that uses the disconnect protocol with an earlier DFI version PHY violates the earlier protocol and produces unpredictable results. For a DFI 4.0 MC to work with an earlier version of the PHY, the MC should implement a method for disabling the disconnect protocol. A PHY that works with a MC that is earlier than DFI 4.0 does not detect a disconnect and therefore will have no issues.

DFI 4.0 PHYs must be able to tolerate a disconnect requests on all interfaces. However, the PHY is not required to disconnect and can define the disconnect timing accordingly. The PHY can support either the error or QOS disconnect on any of the interfaces. However, if the PHY goes into an error state, it should not disconnect if **dfi_disconnect_error** = ’b0. The PHY should clearly define support for disconnect on all interfaces.

Presently, there are multiple interface handshakes in the DFI 4.0 specification as shown in Table 31, “MC / PHY Handshaking Interfaces and Signals”.

TABLE 31. *MC / PHY Handshaking Interfaces and Signals*

Interface	Handshaking Description	Signals	Supports the Disconnect Protocol
Update	Controller Update	**dfi_ctrlupd_req** / **dfi_ctrlupd_ack**	Yes
	PHY Update	**dfi_phyupd_req** / **dfi_phyupd_ack**	Yes
Training	Read Data Eye Training	**dfi_rdlvl_en / dfi_rdlvl_resp**	Yes
	Read Gate Training	**dfi_rdlvl_gate_en / dfi_rdlvl_resp**	Yes
	Write DQ Training	**dfi_wdqlvl_en / dfi_wdqlvl_resp**	Yes
	Write Leveling	**dfi_wrlvl_en / dfi_wrlvl_resp**	Yes
	CA Training	**dfi_calvl_en / dfi_calvl_resp**	Yes
Low Power Control	Low Power Control	**dfi_lp_ctrl_req** / **dfi_lp_ack** **dfi_lp_data_req** / **dfi_lp_ack**	No
PHY Master	PHY Master	**dfi_phymstr_req** / **dfi_phymstr_ack**	Yes
Status Interface	Frequency Change	**dfi_init_start** / **dfi_init_complete**	No

The signal associated with the disconnect protocol is listed in Table 32, “Disconnect Protocol Signal”.

TABLE 32. *Disconnect Protocol Signal*

Signal	From	Width	Default	Description
dfi_disconnect_error	MC	1 bit	0x0	DFI disconnect error. Indicates if the current disconnect is an error or a QOS (fully operational) request. If de-asserted, the disconnect request requires that the PHY remain fully operational after the disconnect. If asserted, the PHY might not be fully operational after the disconnect.

The disconnect protocol defines eight pairs of timing parameters; each affected interface has a $\mathbf{t}_{\mathbf{*\_disconnect}}$ and $\mathbf{t}_{\mathbf{*\_disconnect\_error}}$ timing parameter. The $\mathbf{t}_{\mathbf{*\_disconnect}}$ timing parameter is associated with **dfi_disconnect_error** = ’b0 and the $\mathbf{t}_{\mathbf{*\_disconnect\_error}}$ timing parameter is associated with **dfi_disconnect_error** = ’b1. Both parameters define the maximum number of clocks to disconnect.

The timing parameters associated with the disconnect protocol are listed in Table 33, “Disconnect Protocol Timing Parameters”.

TABLE 33. *Disconnect Protocol Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{calvl\_disconnect}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during CA training, from the de-assertion of any bit of the **dfi_calvl_en** signal to the assertion of **dfi_calvl_resp** when **dfi_disconnect_error** = ’b0.
$t_{calvl\_disconnect\_error}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during CA training, from the de-assertion of any bit of the **dfi_calvl_en** signal to the assertion of **dfi_calvl_resp** when **dfi_disconnect_error** = ’b1.
$t_{ctrlupd\_disconnect}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a control update sequence, from the de-assertion of **dfi_ctrlupd_req** to the de-assertion of **dfi_ctrlupd_ack** when **dfi_disconnect_error** = ’b0.
$t_{ctrlupd\_disconnect\_error}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a control update sequence, from the de-assertion of **dfi_ctrlupd_req** to the de-assertion of **dfi_ctrlupd_ack** when **dfi_disconnect_error** = ’b1.
$t_{phyupd\_disconnect}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a PHY update sequence, from the de-assertion of **dfi_phyupd_ack** to the de-assertion of **dfi_phyupd_req** when **dfi_disconnect_error** = ’b0.
$t_{phyupd\_disconnect\_error}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a PHY update sequence, from the de-assertion of **dfi_phyupd_ack** to the de-assertion of **dfi_phyupd_req** when **dfi_disconnect_error** = ’b1.
$t_{phymstr\_disconnect}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a PHY master request, from the de-assertion of **dfi_phymstr_req** to the de-assertion of **dfi_phymstr_ack** when **dfi_disconnect_error** = ’b0.
$t_{phymstr\_disconnect\_error}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a PHY master request, from the de-assertion of **dfi_phymstr_req** to the de-assertion of **dfi_phymstr_ack** when **dfi_disconnect_error** = ’b1.
$t_{rdlvl\_disconnect}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during read data eye training, from the de-assertion of any bit (or bits) of the **dfi_rdlvl_en** signal to the assertion of **dfi_rdlvl_resp** when **dfi_disconnect_error** = ’b0.
$t_{rdlvl\_disconnect\_error}$	PHY	0x0	-[a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during read data eye training, from the de-assertion of any bit (or bits) of the **dfi_rdlvl_en** signal to the assertion of **dfi_rdlvl_resp** when **dfi_disconnect_error** = ’b1.

TABLE 33. *Disconnect Protocol Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{rdlvl\_gate\_disconnect}$	PHY	0x0	- [a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during read gate training, from the de-assertion of any bit of the **dfi_rdlvl_gate_en** signal to the assertion of **dfi_rdlvl_resp** when **dfi_disconnect_error** = ’b0.
$t_{rdlvl\_gate\_disconnect\_error}$	PHY	0x0	- [a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during read gate training, from the de-assertion of any bit of the **dfi_rdlvl_gate_en** signal to the assertion of **dfi_rdlvl_resp** when **dfi_disconnect_error** = ’b1.
$t_{wdqlvl\_disconnect}$	PHY	0x0	- [a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during write DQ training, from the de-assertion of any bit of the **dfi_wdqlvl_en** signal to the assertion of **dfi_wdqlvl_resp** when **dfi_disconnect_error** = ’b0.
$t_{wdqlvl\_disconnect\_error}$	PHY	0x0	- [a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during write DQ training, from the de-assertion of any bit of the **dfi_wdqlvl_en** signal to the assertion of **dfi_wdqlvl_resp** when **dfi_disconnect_error** = ’b1.
$t_{wrlvl\_disconnect}$	PHY	0x0	- [a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during write training, from the de-assertion of any bit of the **dfi_wrlvl_en** signal to the assertion of **dfi_wrlvl_resp** when **dfi_disconnect_error** = ’b0.
$t_{wrlvl\_disconnect\_error}$	PHY	0x0	- [a]	PHY DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during write training, from the de-assertion of any bit of the **dfi_wrlvl_en** signal to the assertion of **dfi_wrlvl_resp** when **dfi_disconnect_error** = ’b1.

[a] The minimum supportable value is 1; the DFI does not specify a maximum value. The range of values supported is implementation-specific.

3.13 Geardown Mode

DDR4 DRAMs include a geardown mode for enabling command bus operation at high frequencies. Previously, systems would use 2T operation for increasing setup and hold times on the command bus, but 2T does not work for the CS signal. With geardown mode, all command/address (CA) bus signals can benefit from larger setup and hold times.

Larger setup and hold times are accomplished by operating the CA bus at ½ the frequency of memory clock. The DRAM internally samples the bus every other clock. To establish the sampling clock, synchronization between MC and DRAM is established during the geardown mode entrance procedure via a sync pulse. When this relationship is established, the MC can change the CA bus signals only on the boundaries that the sync pulse established.

In geardown mode, the PHY might need to change the alignment of the memory command relative to the memory clock. In normal operation, the memory command is generally aligned to the falling edge of the memory clock to center on the following rising edge. In geardown mode, the command should be aligned to the command's first clock rising edge to center on the command's second clock rising edge. To accomplish this, the PHY receives notice when the memory changes between normal mode and geardown mode via the **dfi_geardown_en** signal.

The signal associated with the geardown interface is listed in Table 34, “Geardown Mode Signal”.

TABLE 34. *Geardown Mode Signal*

Signal	From	Width	Default	Description
dfi_geardown_en	MC	1 bit	0x0	DFI geardown enable. When de-asserted, the MC and PHY operate normally. When asserted, the MC and PHY are in geardown mode. The MC can change CA signals only every other DFI PHY clock as defined by the synchronization pulse from the PHY.

Table 35, “Disconnect Protocol Timing Parameter” defines the geardown mode timing parameter.

TABLE 35. *Disconnect Protocol Timing Parameter*

Parameter	Defined By	Min	Max	Unit	Description
$t_{geardown\_delay}$	PHY	0x0	-[a]	PHY DFI clock cycles	The delay from **dfi_geardown_en** assertion to the time that the PHY is ready to receive commands.

[a] The minimum supportable value is 1. The DFI does not specify a maximum value. The range of values supported is implementation-specific.

3.14 Channels for LPDDR4 memories

LPDDR4 memories define the DRAM signaling in terms of channels; for example, a single LPDDR4 device will have two channels. A system can organize the memory channels as two “independent” 16-bit memory interfaces, or “combined” as a single 32-bit memory. It might be desirable to support both options in the MC and PHY. The DFI bus will be considered a single channel, and therefore a single LPDDR4 device would either connect to a single DFI bus for combined channels or to two DFI buses for independent channels. These DFI buses will operate independently except for the reset.

As a combined interface, the two DFI buses will operate in lock-step. The read and write data interfaces will be used together for transferring two channels’ worth of data. For all other interfaces, the PHY can optionally connect to a single interface or to both interfaces.

The $\mathbf{phy}_{\mathbf{channel\_en}}$ programmable parameter indicates whether the PHY connects to a single channel or both channels when it operates in combined mode. For independent mode, both channels must always be enabled. It is assumed that support for combined and independent modes, and current mode of operation, are defined outside of DFI. Table 36, “DFI Data Channel Programmable Parameters” defines the new programming parameter.

TABLE 36. *DFI Data Channel Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{phy}_{\mathbf{channel\_en}}$	System / PHY	Defines which DFI channels are enabled. • ’b00 = Reserved • ’b01 = Channel 0 enabled, channel 1 disabled. • ’b10 = Channel 1 enabled, channel 0 disabled. • ’b11 = Channel 0 and channel 1 enabled.

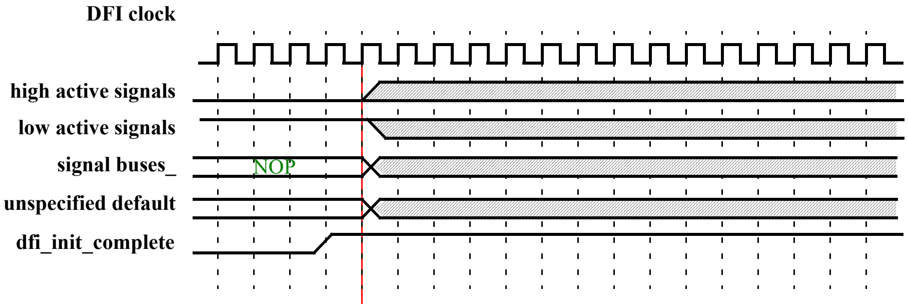
4.0 Functional Use

While some of the diagrams illustrate DFI PHY signals, these signals are only an interpretive example of internal PHY signals, they are not signals on the DFI.

4.1 Initialization

The DFI signals that communicate commands or status, shown in Figure 4, “Dependency on dfi_init_complete”, must maintain their default value until the **dfi_init_complete** signal is asserted.

FIGURE 4. *Dependency on **dfi_init_complete***



The default value for each signal is listed in the corresponding interface table. For example, the default for control signals are listed in the default column of Table 5, “Control Signals”.

1. High active signals have a default value of 0 (e.g., **dfi_odt, dfi_wrdata_en**).
2. Low active signals have a default value of 1 (e.g., **dfi_we_n**).
3. Signal buses have an unspecified default value (e.g., **dfi_address**, **dfi_cid**). The **dfi_address** signal does not have a default value in most cases. However, for LPDDR2, LPDDR3 and LPDDR4 DRAMs, the **dfi_address** bus must drive a NOP command until the dfi_init_complete signal is asserted.
4. State-specific signals have state-specific values (e.g. **dfi_dram_clk_disable**, **dfi_parity_in**).

The **dfi_init_start** signal is used for indicating that the MC is driving valid values on the DFI signals and that the optional **dfi_freq_ratio** signal of the status interface is valid. The **dfi_freq_ratio** signal identifies the MC:PHY frequency ratio. The PHY may use the **dfi_init_start** signal assertion to know that this status signal is valid.

Figure 5, “System Setting Signals - dfi_init_start Asserts Before dfi_init_complete” shows that during initialization, if the frequency ratio protocol is implemented, the **dfi_init_start** signal should only be asserted after the **dfi_freq_ratio** signal has been defined. On the initial assertion of **dfi_init_start**, the user should also define the initial frequency for the system on the **dfi_frequency** signal. If the PHY requires this information for proper initialization, the PHY should wait for the **dfi_init_start** assertion before asserting the **dfi_init_complete** signal. When both **dfi_init_start** and **dfi_init_complete** signals are asserted for at least one DFI clock cycle, the MC can de-assert the **dfi_init_start** signal or continue to hold it asserted.

FIGURE 5. *System Setting Signals - **dfi_init_start** Asserts Before **dfi_init_complete***

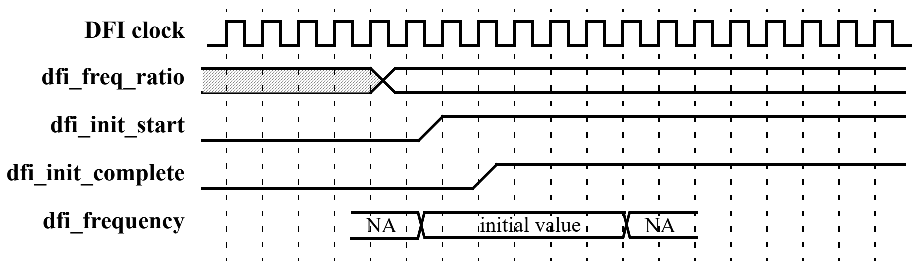
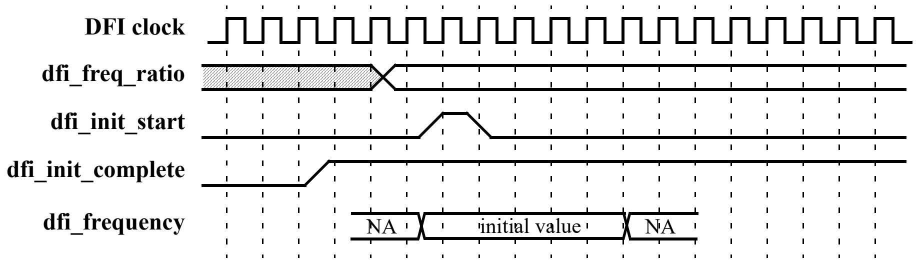


Figure 6, “System Setting Signals - dfi_init_start Asserts After dfi_init_complete” shows that the **dfi_init_complete** signal may be asserted before the assertion of the **dfi_init_start** signal If the frequency ratio protocol is not implemented, or the PHY does not require this or initial frequency information for proper initialization, the PHY may assert the **dfi_init_complete** signal without regard to the assertion of the **dfi_init_start** signal. Note that the **dfi_init_start** signal must be asserted during initialization, even if the PHY is not requiring this information. When both **dfi_init_start** and **dfi_init_complete** signals are asserted for at least one DFI clock cycle, the MC can de-assert the **dfi_init_start** signal or continue to hold it asserted.

FIGURE 6. *System Setting Signals - **dfi_init_start** Asserts After **dfi_init_complete***



The DFI specification does not impose or dictate a reset sequence or any type of signal training for either the PHY or the MC prior to DFI signal assertion. However, the assertion of the **dfi_init_complete** signal signifies that the PHY is ready to respond to any assertions on the DFI by the MC and ensures appropriate responses on the DFI. The PHY must guarantee the integrity of the address and control interface to the DRAMs prior to asserting the **dfi_init_complete** signal.

For LPDDR2, LPDDR3 and LPDDR4 DRAMs, the **dfi_address** bus must drive a NOP command until the **dfi_init_complete** signal is asserted and the signals **dfi_act_n**, **dfi_bank**, **dfi_bg**, **dfi_cid**, **dfi_cas_n**, **dfi_ras_n** and **dfi_we_n** are unused and must remain at a constant value when the DFI bus is being used.

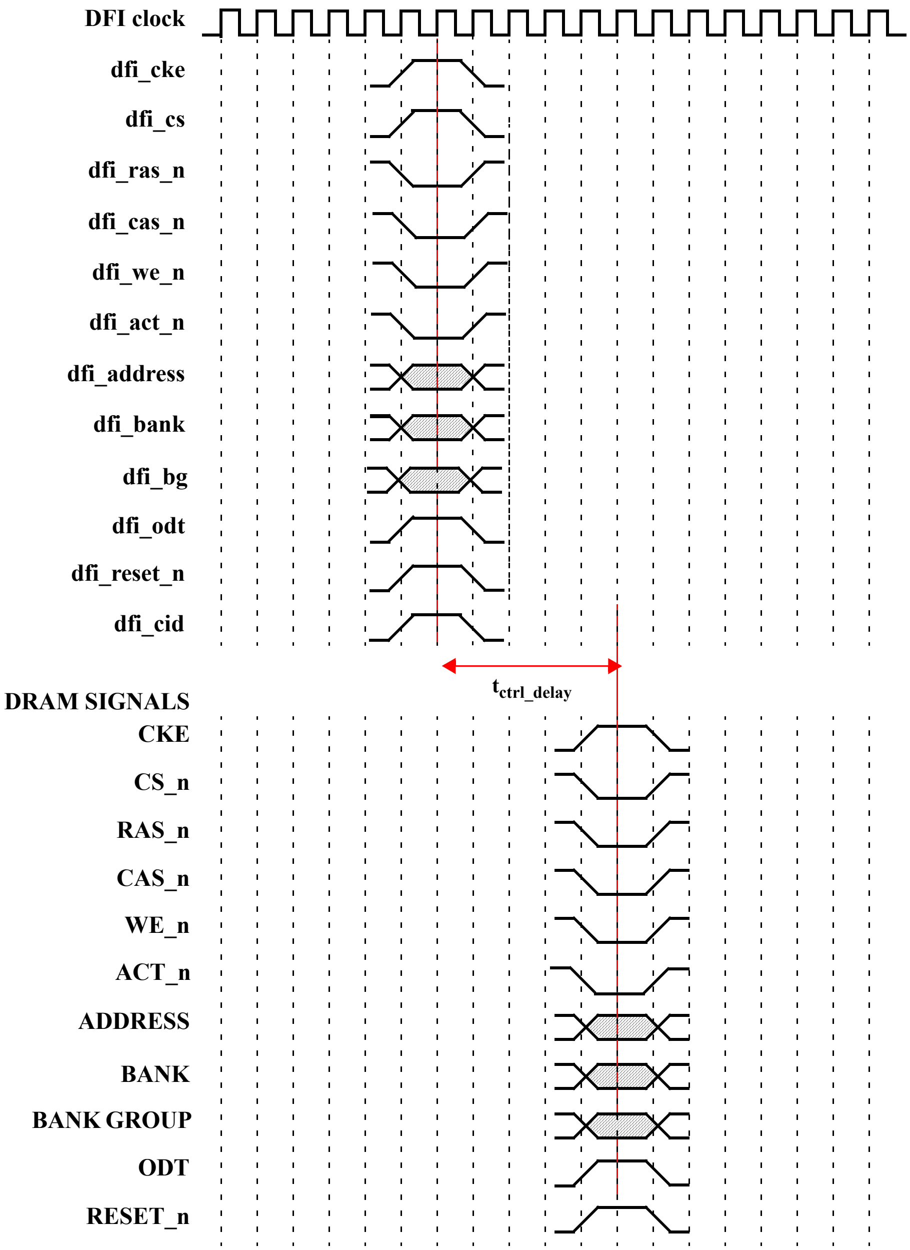
Some of the training interface signals must remain at default until after the assertion of the **dfi_init_complete** signal. No default value must be maintained for the following signals: **dfi_calvl_resp**, **dfi_wrdata**, **dfi_wrdata_mask**, **dfi_rddata**, **dfi_phyupd_type**, **dfi_rdlvl_resp**, **dfi_wdqlvl_resp**, **dfi_wrlvl_resp** and **dfi_lp_wakeup**. The **dfi_address** signal also has no default value except for LPDDR2, LPDDR3 and LPDDR4 DRAMs.

4.2 Control Signals

The DFI control signals consist of the Command Address (CA) signals **dfi_act_n**, **dfi_address**, **dfi_bank**, **dfi_bg**, **dfi_cas_n**, **dfi_ras_n**, **dfi_we_n**, **dfi_cid**, **dfi_cke**, **dfi_cs**, **dfi_odt** and **dfi_reset_n**. The DFI control signals correlate to the DRAM control signals, and are driven according to the timing parameters $\mathbf{t_{ctrl\_delay}}$ and $\mathbf{t_{cmd\_lat}}$. For more information on control signals and parameters, refer to Section 3.1, “Control Interface”.

Figure 7, “DFI Control Interface Signal Relationships” shows that the control signals are driven to the DRAMs and the DFI relationship of the control signals is expected to be maintained at the PHY-DRAM boundary so any delays should be consistent across all signals and defined through the timing parameter $\mathbf{t_{ctrl\_delay}}$. All control signals are illustrated but might not all be required, depending on system requirements. Refer to Table 3, “DFI Signal Requirements” to determine the signals that are relevant for a specific system.

FIGURE 7. *DFI Control Interface Signal Relationships*



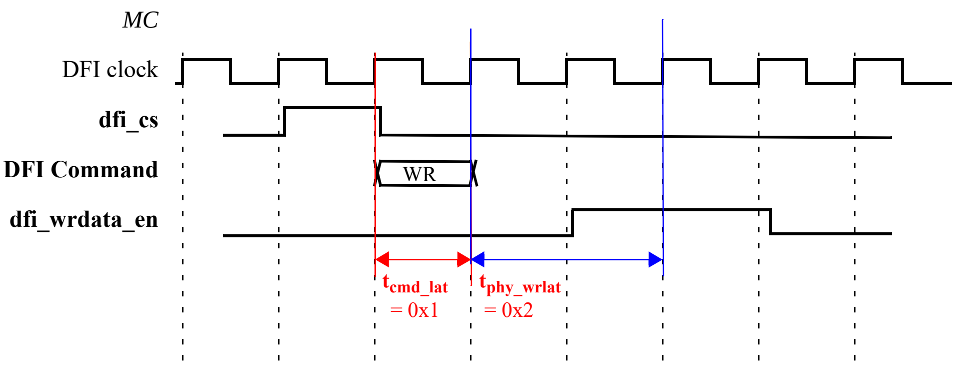
NOTE: The system might not use all of the pins on the DRAM interface, such as additional bank and chip selects. If these signals are never used in a system, they are not required on the DFI. However, if multiple memory sub-systems are supported, the union of the required signals must exist on DFI. In this case, signals unused for a specific topology must be driven through the DFI and must not be left floating.

Based on current DRAM settings, the MC defines **t_{cmd_lat}** with the parameter value defining when the CA bus is driven for each command. The timing of the CA bus assertion is relative to the assertion of **dfi_cs**.

Figure 8, "Example of tcmd_lat (tcmd_lat = 1, tphy_wrlat = 2)" illustrates how the **t_{cmd_lat}** parameter relates the DFI command to the DFI chip select. Furthermore, Figure 8, "Example of tcmd_lat (tcmd_lat = 1, tphy_wrlat = 2)" shows how DFI timing parameters, such as **t_{phy_wrlat}**, relate to the DFI command when **t_{cmd_lat}** is defined as a non-zero value.

For more information on **t_{cmd_lat}**, refer to Table 6, "Control Timing Parameters".

FIGURE 8. *Example of t_{cmd_lat} (t_{cmd_lat} = 1, t_{phy_wrlat} = 2)*



4.3 3DS Stack Support

The DFI specification supports 3D stack support for DDR3 and DDR4 devices. The signaling is dependent on memory type.

4.3.1 3DS Addressing with dfi_cid and dfi_cs for DDR3

While addressing 3DS devices with two and four logical ranks, the MC can use dedicated **dfi_cs** inputs to select these logical ranks. For 3DS SDRAMs with eight logical ranks, the MC requires the additional **dfi_cid** (Chip ID) address input to select logical ranks 4 through 7.

TABLE 37. *DDR3 Configuration with 8 Logical and 1 Physical Rank*

Physical Rank	Logical Rank	dfi_cs [3:0]	dfi_cid [0]
0	0	1110	0
0	1	1101	0
0	2	1011	0
0	3	0111	0
0	4	1110	1
0	5	1101	1
0	6	1011	1
0	7	0111	1

4.3.2 3DS Addressing with dfi_cid and dfi_cs for DDR4

For DDR4, the 3DS package is organized into two, four, or eight logical ranks. For DDR4 3DS devices, the MC can select the logical ranks by using the **dfi_cid** [2:0] signal bits. The **dfi_cs** signal selects the physical ranks.

TABLE 38. *DDR4 Configuration with 8 Logical and 2 Physical Ranks*

Physical Rank	Logical Rank	dfi_cs [1:0]	dfi_cid [2:0]
0	0	10	000
0	1	10	001
0	2	10	010
0	3	10	011
0	4	10	100
0	5	10	101
0	6	10	110
0	7	10	111
1	0	01	000
1	1	01	001
1	2	01	010
1	3	01	011
1	4	01	100
1	5	01	101
1	6	01	110
1	7	01	111

4.4 Data Bus Inversion

DBI is an optional DFI feature used with write and read data transmissions to generate write DBI data and invert both write and read data for DBI as required. When DBI is enabled, the $\mathbf{phy}_{\mathbf{dbi\_mode}}$ parameter is applicable.

If $\mathbf{phy}_{\mathbf{dbi\_mode}}$ = 0, the MC controls the DBI functionality and the **dfi_rddata_dbi** signal is required. If $\mathbf{phy}_{\mathbf{dbi\_mode}}$ = 1, the PHY performs DBI generation and data inversion. If DBI is required in a system, either the MC or the PHY can generate the DBI output to DRAM and the output must be used to selectively invert data for write commands.

The PHY defines the $\mathbf{phy}_{\mathbf{dbi\_mode}}$ parameter value to determine how DBI is handled, as defined in Table 9, “Write Data Programmable Parameters”.

If the MC supports DBI on DFI, the MC must support both settings of the $\mathbf{phy}_{\mathbf{dbi\_mode}}$ parameter; the MC must be able to generate DBI and invert the write data and read data as needed for DBI, and interface with a PHY that generates and handles the data inversion for DBI. The PHY can optionally support either setting of the $\mathbf{phy}_{\mathbf{dbi\_mode}}$ parameter; the PHY can optionally generate DBI and invert the write data and read data as required. If the PHY does not generate DBI and invert the corresponding write/read data, the PHY must be able to interface to an MC that has DBI support. When both CRC and DBI are enabled in a system, special care needs to be taken regarding where CRC and DBI are performed in order to operate correctly.

When $\mathbf{phy}_{\mathbf{dbi\_mode}}$ = 0, the PHY transfers the read DBI data on **dfi_rddata_dbi** coincident with **dfi_rddata**. The MC transfers the write DBI data on **dfi_wrdata_mask** coincident with **dfi_wrdata**.

4.5 Write Transactions

The write data transaction handles the transmission of write data across the DFI bus from the MC to the PHY.

The DFI write transaction includes the signals for write data (**dfi_wrdata**), write data mask (**dfi_wrdata_mask**), write data enable (**dfi_wrdata_en**), write data chip select (**dfi_wrdata_cs**), the associated timing parameters $t_{phy\_crcmax\_lat}$, $t_{phy\_crcmin\_lat}$, t_{phy_wrcgap}, $t_{phy\_wrcslat}$, t_{phy_wrdata}, $\mathbf{t}_{\mathbf{phy\_wrlat}}$ and $\mathbf{t}_{\mathbf{phy\_wrdata}}$ and the programmable parameters $\mathbf{phy}_{\mathbf{crc\_mode}}$ and $\mathbf{phy}_{\mathbf{dbi\_mode}}$.

The **dfi_wrdata_en** signal must correlate with the number of data transfers executed on the DRAM bus; one continuous assertion of the **dfi_wrdata_en** signal may encompass data for multiple write commands.

If the chip select is enabled, the **dfi_wrdata_cs** signal provides the target data path chip select value to each of the PHY data slices. In a 3DS solution control, MRW and training activities are to be limited to physical ranks. For more information on these signals, refer to Section 3.2, “Write Data Interface”.

4.5.1 Write Transaction Sequence

The sequence for DFI write transactions is listed below; the effect of using the optional write data chip select, CRC, or DBI is shown within brackets.

1. The write command is issued.
   a. $\mathbf{t}_{\mathbf{phy\_wrlat}}$ cycles elapse ($\mathbf{t}_{\mathbf{phy\_wrlat}}$ can be zero). (Change: was step 2, now a)

   The $\mathbf{t}_{\mathbf{phy\_wrlat}}$ parameter defines the number of cycles between when the write command is driven on the DFI to assertion of the **dfi_wrdata_en** signal. The $\mathbf{t}_{\mathbf{phy\_wrlat}}$ parameter is PHY-defined but may be specified in terms of other fixed system values.

   The write timing parameters ($\mathbf{t}_{\mathbf{phy\_wrdata}}$ and $\mathbf{t}_{\mathbf{phy\_wrlat}}$) must be held constant while commands are being executed on the DFI bus; however, if necessary, the write timing parameter values may be changed when the bus is in the idle state. The $\mathbf{t}_{\mathbf{phy\_wrdata}}$ and $\mathbf{t}_{\mathbf{phy\_wrlat}}$ timing parameters work together to define the number of cycles from the assertion of a write command on the DFI control interface to when write data is driven on the DFI bus and must be consistent with the write latency timing that correlates to the DRAM timing.

   b. $t_{phy\_wrcslat}$ cycles elapse.

   [If chip select is enabled, the PHY defines the $\mathbf{t}_{\mathbf{phy\_wrcslat}}$ timing parameter to specify the desired alignment of the command to the **dfi_wrdata_cs** signal; the PHY defines the $\mathbf{t}_{\mathbf{phy\_wrcsgap}}$ timing parameter to specify the additional delay it requires between two consecutive commands that are targeting different chip selects. The $\mathbf{t}_{\mathbf{phy\_wrcslat}}$ timing parameter must be held constant while commands are being executed on the DFI bus; however, if necessary, the $\mathbf{t}_{\mathbf{phy\_wrcslat}}$ parameter value may be changed when the bus is in the idle state. The PHY may require the MC to add delay beyond other system timing requirements to account for PHY-specific adjustments transitioning between data path chip selects.]

   [The gap timing requirement may only be applicable to certain chip-select-to-chip-select transitions and not be applicable to other data path chip select transitions where the PHY is not required to make an internal adjustment on the transition; this gap timing requirement is system-specific. For example, a system may not require additional delay on transitions between chip select 0 and chip select 1, but may require additional delay when transitioning from chip select 0 to chip select 2. Accordingly, the interface does not require the gap timing to be applied to every chip select transition.]

2. The MC drives **dfi_wrdata_cs** a minimum of the DFI data transfer width ($\mathbf{dfi}_{\mathbf{rw\_length}}$) plus the gap timing ($\mathbf{t}_{\mathbf{phy\_wrcsgap}}$). The minimum time the chip select is guaranteed to remain driven to the PHY relative to the write command is defined by $\mathbf{t}_{\mathbf{phy\_wrcslat}} + \mathbf{dfi}_{\mathbf{rw\_length}} + \mathbf{t}_{\mathbf{phy\_wrcsgap}}$.-

   The maximum delay that can be achieved between the assertion of a new chip select value on **dfi_wrdata_cs** and the corresponding **dfi_wrdata_en** is limited to the maximum time the PHY has from changing the target chip select to receiving the write data transfer ($\mathbf{t}_{\mathbf{phy\_wrlat}}$ - $\mathbf{t}_{\mathbf{phy\_wrcslat}}$).

   The PHY must define the $\mathbf{t}_{\mathbf{phy\_wrcslat}}$ and $\mathbf{t}_{\mathbf{phy\_wrcsgap}}$ timing parameters to allocate the time between transactions to different chip selects necessary for PHY-specific adjustments.

3. For non-contiguous write commands, the **dfi_wrdata_en** signal is asserted on the DFI after $\mathbf{t}_{\mathbf{phy\_wrlat}}$ is met and remains asserted for the number of cycles required to complete the write data transfer sent on the DFI control interface; $\mathbf{t}_{\mathbf{phy\_wrlat}}$ can be zero.

   [If CRC is enabled, the MC extends the **dfi_wrdata_en** signal to accommodate the extended DRAM burst length and the signal is asserted for an odd number of cycles per burst.]

4. For contiguous write commands, the **dfi_wrdata_en** signal is asserted after $\mathbf{t}_{\mathbf{phy\_wrlat}}$ is met and remains asserted for the entire length of the data stream; $\mathbf{t}_{\mathbf{phy\_wrlat}}$ can be zero. [If CRC is enabled, the MC extends the **dfi_wrdata_en** signal to accommodate the extended DRAM burst length and the signal may be asserted for an odd number of cycles per burst.]

5. $\mathbf{t}_{\mathbf{phy\_wrdata}}$ cycles elapse.

6. The associated write data (**dfi_wrdata**) and masking (**dfi_wrdata_mask**) signals are sent.

   [If CRC is enabled, the MC utilizes the **dfi_wrdata** bus to send CRC data to the PHY; the MC utilizes the **dfi_wrdata_pN** outputs for frequency ratio systems.]

   [If DBI is enabled, **dfi_wrdata_mask** transfers the write data inversion information instead of the write data mask.]

   The MC must always drive **dfi_wrdata** and associated signals with the correct timing relative to the write command as defined by the timing parameters $\mathbf{t}_{\mathbf{phy\_wrdata}}$ and $\mathbf{t}_{\mathbf{phy\_wrlat}}$.

   The $\mathbf{t}_{\mathbf{phy\_wrdata}}$ parameter defines the timing requirements between the assertion of the **dfi_wrdata_en** signal assumed and when the write data is driven on the **dfi_wrdata** signal. The exact value of the $\mathbf{t}_{\mathbf{phy\_wrdata}}$ parameter for a particular application is determined by how many cycles the PHY must receive the **dfi_wrdata_en** signal prior to receiving the **dfi_wrdata** signal.

   If the PHY requires notification of pending write data sooner than 1 cycle, the $\mathbf{t}_{\mathbf{phy\_wrdata}}$ parameter may be increased. However, setting $\mathbf{t}_{\mathbf{phy\_wrdata}}$ to a value greater than 1 may restrict the minimum write latency supported by the interface. The DFI specification does not dictate a value for the $\mathbf{t}_{\mathbf{phy\_wrdata}}$ parameter.

7. The **dfi_wrdata_en** signal de-asserts $\mathbf{t}_{\mathbf{phy\_wrdata}}$ cycles before the last valid data is transferred on the **dfi_wrdata** bus.

8. $\mathbf{t}_{\mathbf{phy\_wrdata\_delay}}$ cycles elapse. The DFI bus enters the idle state.

The idle state timing parameter defines the number of DFI clocks from **dfi_wrdata_en** to the completion of the write data transfer on the DRAM bus. The MC drives the next value (any valid chip select or inactive).

Seven situations showing system behavior with two write transactions are presented in Figure 9, Figure 10, Figure 11, Figure 12, Figure 13, Figure 14 and Figure 15. System behavior with three write transactions using **dfi_wrdata_cs** is shown in Figure 16.

Figure 9, “Back-to-Back Writes (DRAM Burst of 4: tphy_wrlat = 0, tphy_wrdata = 1)” shows back-to-back writes for a system with a $\mathbf{t_{phy\_wrlat}}$ of zero and a $\mathbf{t_{phy\_wrdata}}$ of one. The **dfi_wrdata_en** signal is asserted with the write command for this situation, and is asserted for two cycles per command to inform the DFI that two cycles of DFI data are sent for each write command. The timing parameters and the timing of the write commands allow the **dfi_wrdata_en** signal and the **dfi_wrdata** stream to be sent contiguously.

FIGURE 9. *Back-to-Back Writes (DRAM Burst of 4: t_{phy_wrlat} = 0, t_{phy_wrdata} = 1)*

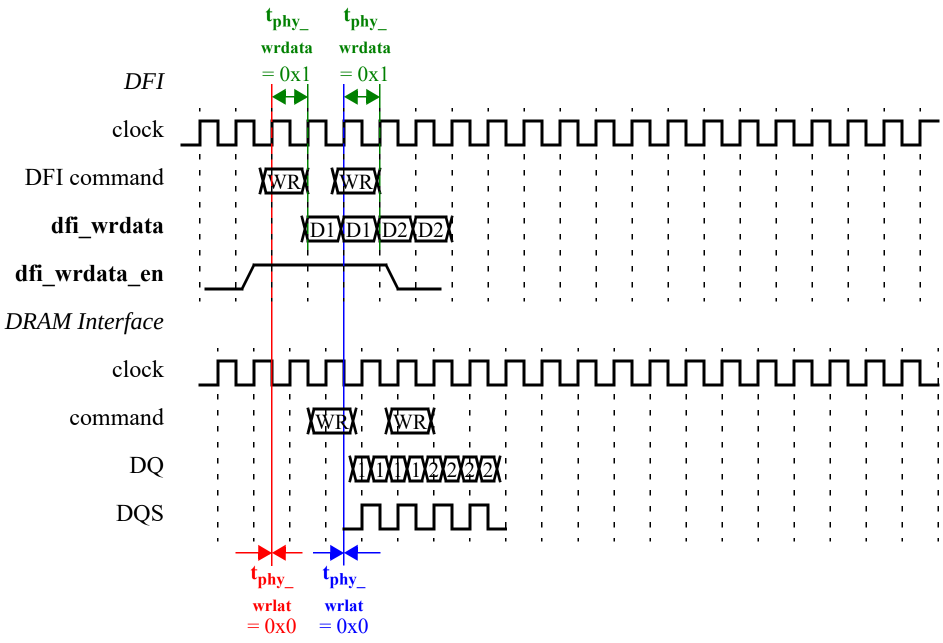


Figure 10, “Back-to-Back Interrupted Contiguous Writes (DRAM Burst of 8: tphy_wrlat = 3, tphy_wrdata = 2)” shows an interrupted write command. The **dfi_wrdata_en** signal should be asserted for 4 cycles for each of these write transactions. However, since the first write is interrupted, the **dfi_wrdata_en** signal is asserted for a portion of the first transaction and the complete second transaction. The **dfi_wrdata_en** signal will not de-assert between write commands, and the **dfi_wrdata** stream will be sent contiguously for a portion of the first command and the complete second command.

FIGURE 10. *Back-to-Back Interrupted Contiguous Writes (DRAM Burst of 8: t_{phy_wrlat} = 3, t_{phy_wrdata} = 2)*

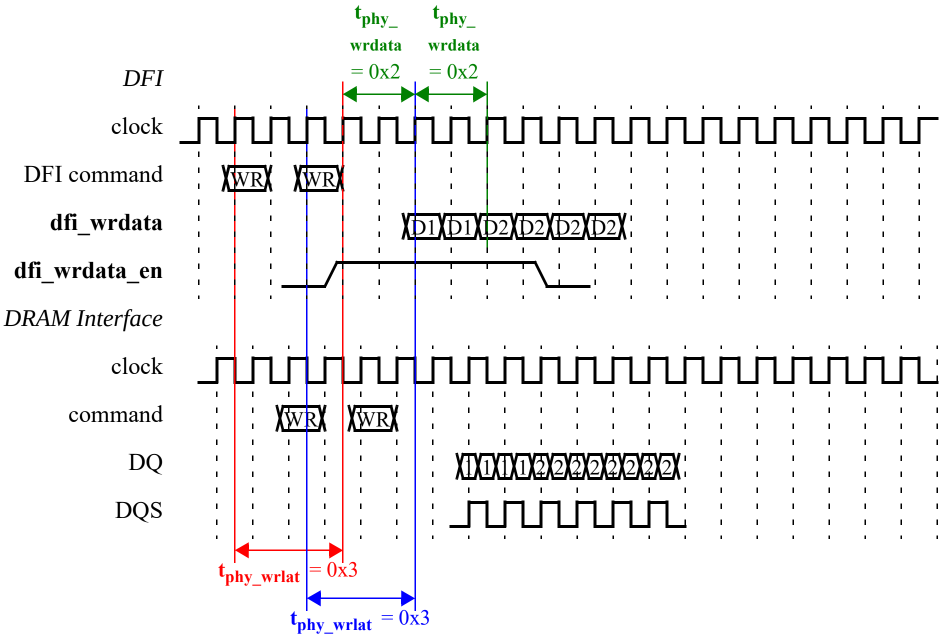


Figure 11, “Back-to-Back Writes (DRAM Burst of 8: tphy_wrlat = 4, tphy_wrdata = 4)” shows back-to-back burst-of-8 writes. The **dfi_wrdata_en** signal must be asserted for 4 cycles for each of these write transactions.

FIGURE 11. *Back-to-Back Writes (DRAM Burst of 8: t_{phy_wrlat} = 4, t_{phy_wrdata} = 4)*

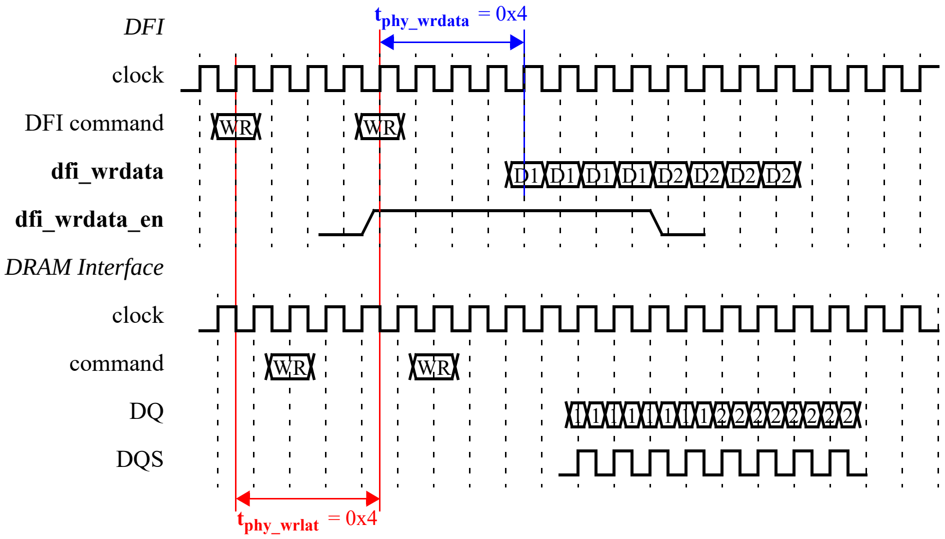


Figure 12, Figure 13, Figure 14 and Figure 15 also show two complete write commands, with different $\mathbf{t_{phy\_wrlat}}$ and $\mathbf{t_{phy\_wrdata}}$ timing parameters and for different DRAM types. The **dfi_wrdata_en** signal is asserted for two cycles for each write transaction. The t_{phy_wrlat} timing and the timing between the write commands causes the **dfi_wrdata_en** signal to be de-asserted between commands. As a result, the **dfi_wrdata** stream is non-contiguous.

FIGURE 12. *Two Independent Writes (DRAM Burst of 4: t_{phy_wrlat} = 0, t_{phy_wrdata} = 1)*

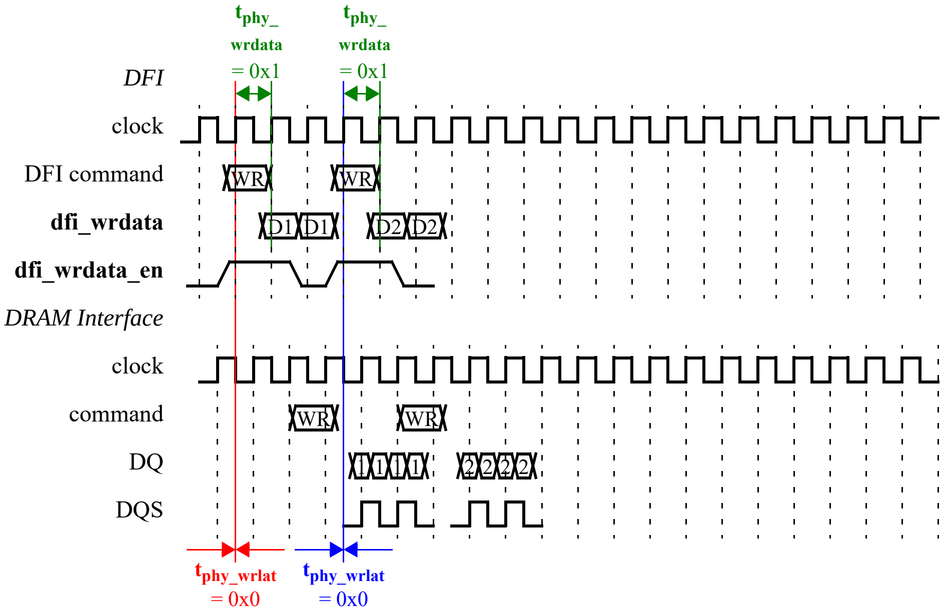


FIGURE 13. *Two Independent Writes (DRAM Burst of 4: t_{phy_wrlat} = 3, t_{phy_wrdata} = 1)*

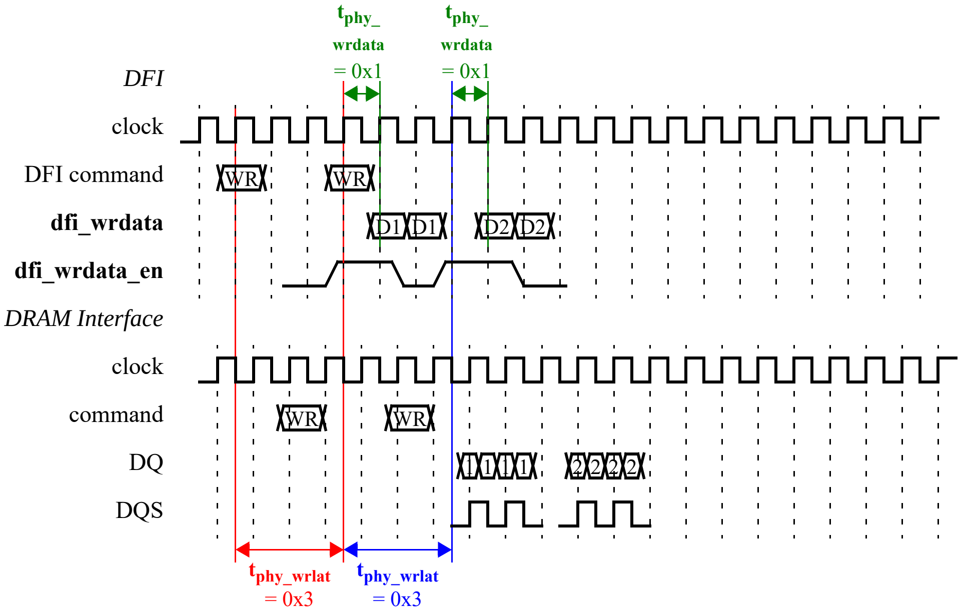


FIGURE 14. *Two Independent Writes (DRAM Burst of 8: t_{phy_wrlat} = 3, t_{phy_wrdata} = 3)*

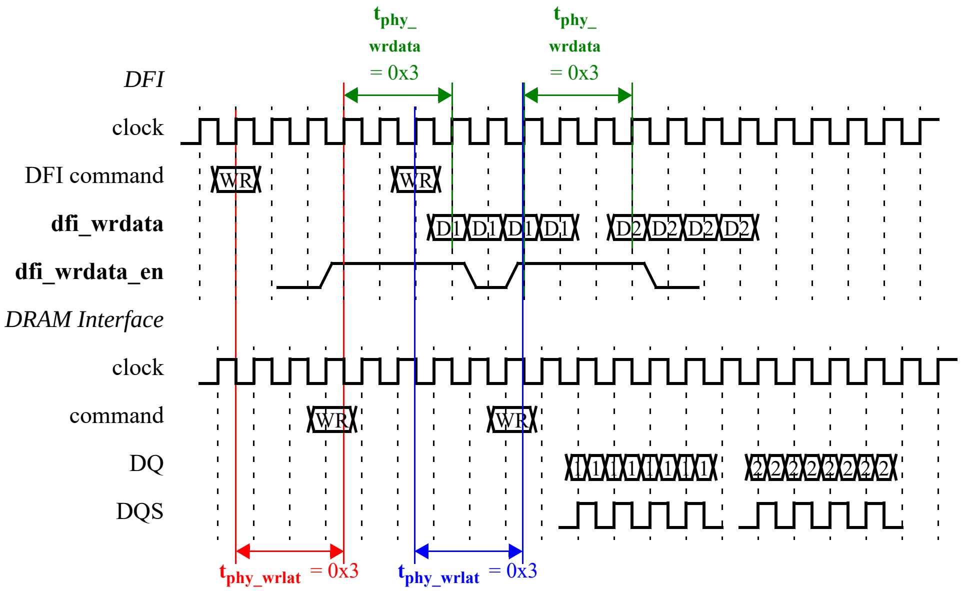


FIGURE 15. *Two Independent Writes (DRAM Burst of 4: t_{phy_wrlat} = 3, t_{phy_wrdata} = 4)*

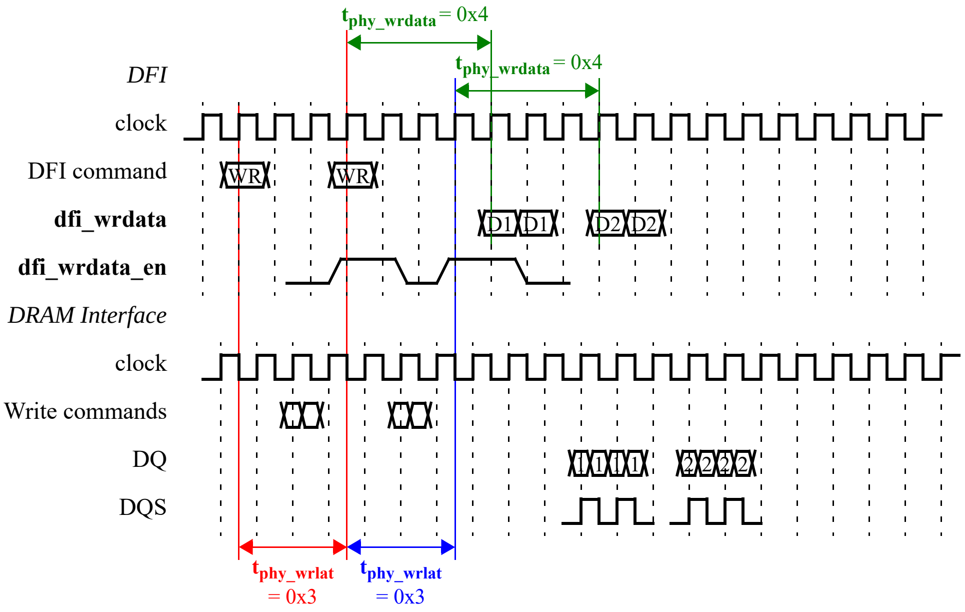
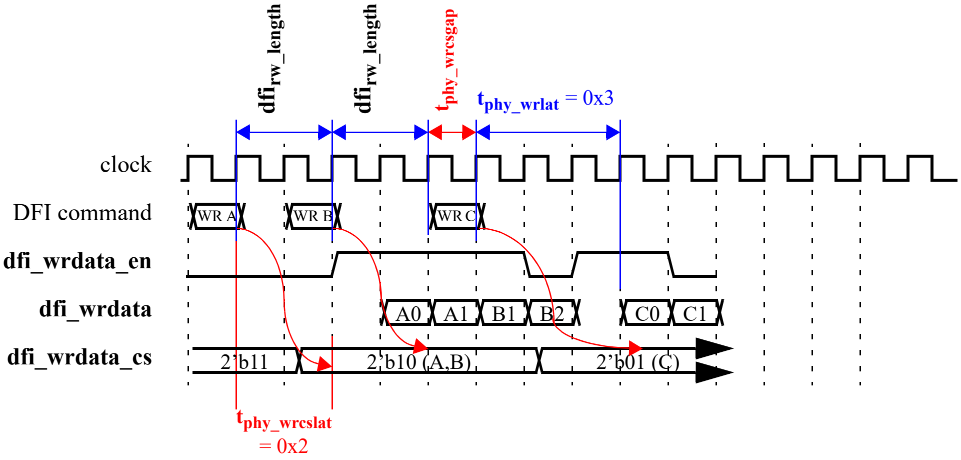


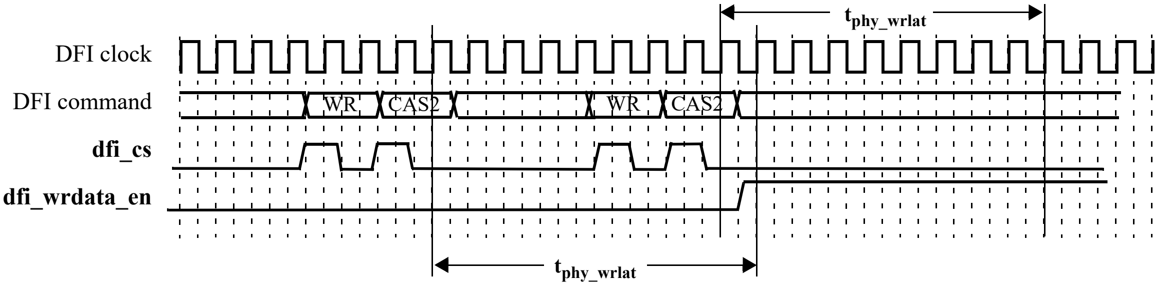
Figure 16, "Write Commands Utilizing dfi_wrdata_cs (DRAM Burst of 4: tphy_wrlat = 3, tphy_wrdat = 4, tphy_wrcslat = 2, tphy_wrcsgap = 1)" shows three write commands, with a gap between the second and third commands. The first two commands (WR A and WR B) address CS0 while the third one (WR C) addresses CS1. The two first write commands are back to back, that is, spaced by **dfi**$_{\textbf{rw\_length}}$. The third command is also back to back, but with the required extra spacing, **t**$_{\textbf{phy\_wrcsgap}}$.

FIGURE 16. *Write Commands Utilizing **dfi_wrdata_cs** (DRAM Burst of 4: **t**$_{phy\_wrlat}$ = 3, **t**$_{phy\_wrdat}$ = 4, **t**$_{phy\_wrcslat}$ = 2, **t**$_{phy\_wrcsgap}$ = 1)*



DFI timing parameters for read and write commands will be referenced from the second tick of the second command in the same manner that JEDEC LPDDR4 references RL and WL.

FIGURE 17. *LPDDR4 Write Command*



4.5.2 DBI - Write

DBI is an optional DFI feature used with write data transmissions. The **phy**$_{\textbf{dbi\_mode}}$ parameter is only needed when DBI is supported in DFI.

For more information on the **phy**$_{\textbf{dbi\_mode}}$ parameter, refer to Table 8, “Write Data Timing Parameters”.

When the MC generates the write DBI data, the MC also inverts write data for DBI as required. The MC transmits the write DBI data across the DFI bus on **dfi_wrdata_mask**. In DBI mode, the PHY is only required to transfer the write DBI data through the PHY. While write data is transmitting, the DBI data transmits simultaneously on the **dfi_wrdata_mask** signal. The **dfi_wrdata_mask** signal is sent coincident with the corresponding **dfi_wrdata** bus.

For frequency ratio systems, the **dfi_wrdata_mask** signal is extended, with a signal defined per phase.

When both DBI and CRC are enabled in a system, special care needs to be taken regarding where DBI and CRC are performed in order to operate correctly. DBI inversion should take effect prior to CRC encoding.

4.5.3 Cyclic Redundancy Check

CRC is an optional DFI feature used with write data transmissions to send CRC data as part of the write data burst. CRC extends a burst of 8 unit intervals (UI) to 10 UIs. When CRC is used, either the MC or the PHY can generate the CRC data; however, the MC must generate the CRC data if the PHY does not generate it. The CRC data is made of the CRC code that is expanded by the needed padding values to reach a full DFI PHY clock cycle data transfer (An example is padding with 1's for DDR4 x8 and x16 devices.)

While either the MC or the PHY can generate the CRC data, the PHY defines the value of the **phy**$_{\mathbf{crc\_mode}}$ programmable parameter. The **phy**$_{\mathbf{crc\_mode}}$ parameter is only needed when CRC is supported in DFI and uses the following definition to determine how CRC is handled:

- **phy**$_{\mathbf{crc\_mode}}$ == 0 → CRC generation is handled in the MC
- **phy**$_{\mathbf{crc\_mode}}$ == 1 → CRC generation is handled in the PHY

When CRC is supported in DFI, the system must be capable of the conditions listed in Table 39, “Systems Requiring CRC Support”.

TABLE 39. *Systems Requiring CRC Support*

Description	MC	PHY
Generates CRC Data	Yes	Optional
Interfaces with CRC Data	Yes	Yes [a]

[a] Required if the PHY does not generate the CRC data.

Regardless of which device generates the CRC, the MC asserts ODT such that it applies to all data sent + CRC data words.

4.5.3.1 MC CRC Support (phy$_{crc\_mode}$ = 0)

When the MC generates the CRC data, the **phy**$_{\mathbf{crc\_mode}}$ == 0 and the MC has the following requirements.

- The MC must assert the **dfi_wrdata_en** signal for the data transmitted across the DFI bus, including CRC data.
- The MC spaces commands to handle an extended burst with CRC.
- The MC generates **dfi_odt** (**dfi_odt_pN** in frequency ratio systems) based on the DRAM burst length including CRC data. With CRC enabled, the MC may need to extend ODT.
- The MC needs to receive and capture error information from the PHY. These CRC write data errors are transmitted on the **dfi_alert_n** signal. In systems that support either command parity or CRC, the MC must support a **dfi_alert_n** input and the PHY must support the **dfi_alert_n** output.

With CRC, the **dfi_wrdata_en** signal could be asserted for an odd number of cycles per burst. Without exception, the PHY must support the odd CRC burst timing.

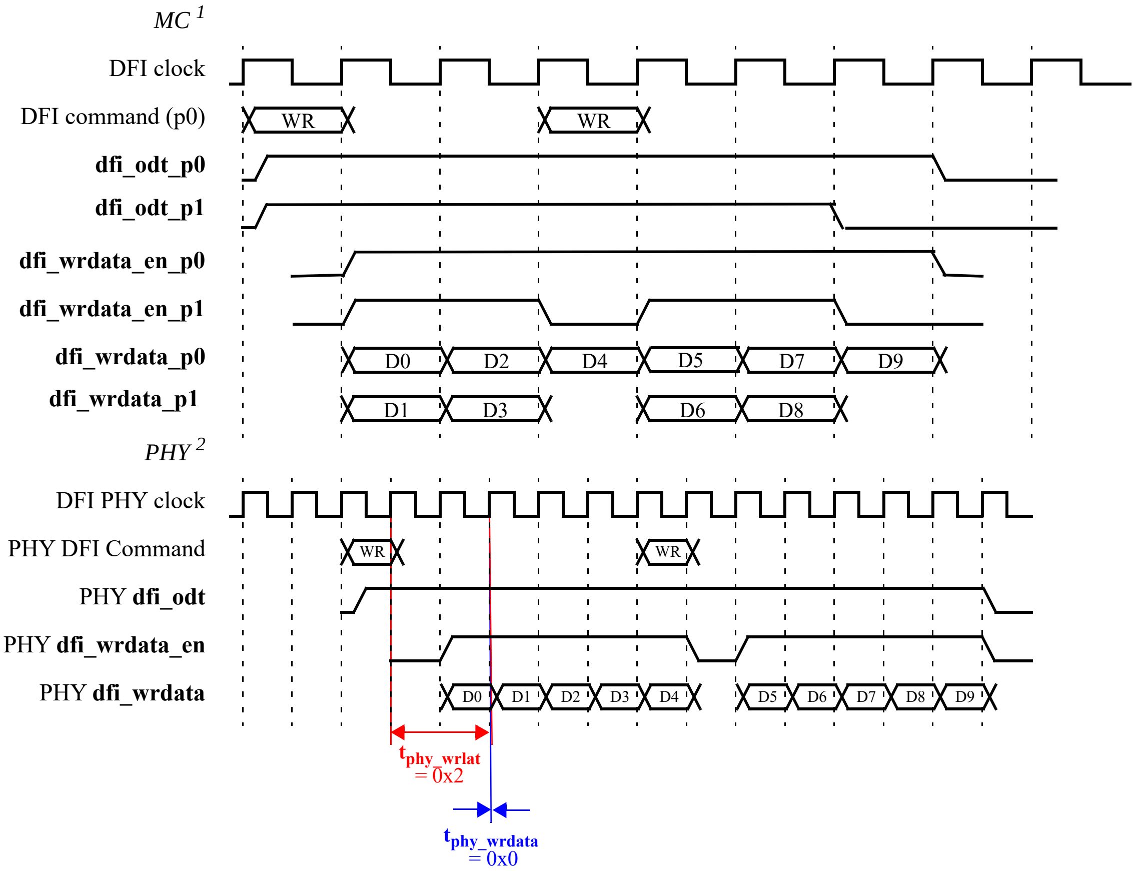
DFI dictates that when CRC is used, the CRC data word must be incorporated in the burst of write data, but DFI does not dictate placement within the burst. The specific ODT timing requirements are dependent on the chip selects accessed and system architecture. The **dfi_odt** assertion and de-assertion are identical in the MC CRC and PHY CRC support modes. The ODT signal timing shown is only an example of one potential solution.

To further compare MC CRC support mode and PHY CRC support mode, DFI ODT signals are included in the following figures:

- Figure 17, “LPDDR4 Write Command”
- Figure 18, “DFI Write Data Bus for MC CRC Support Mode (Two Bursts starting in Phase 0)”
- Figure 19, “DFI Write Data Bus for MC CRC Support Mode (Two Back-to-Back Bursts)”
- Figure 20, “DFI Write Data Bus for PHY CRC Support Mode”

Figure 18 and Figure 19 illustrate the DFI write data bus for a 2:1 frequency ratio system with CRC extending a burst of 8 by1 additional clock DRAM cycle.

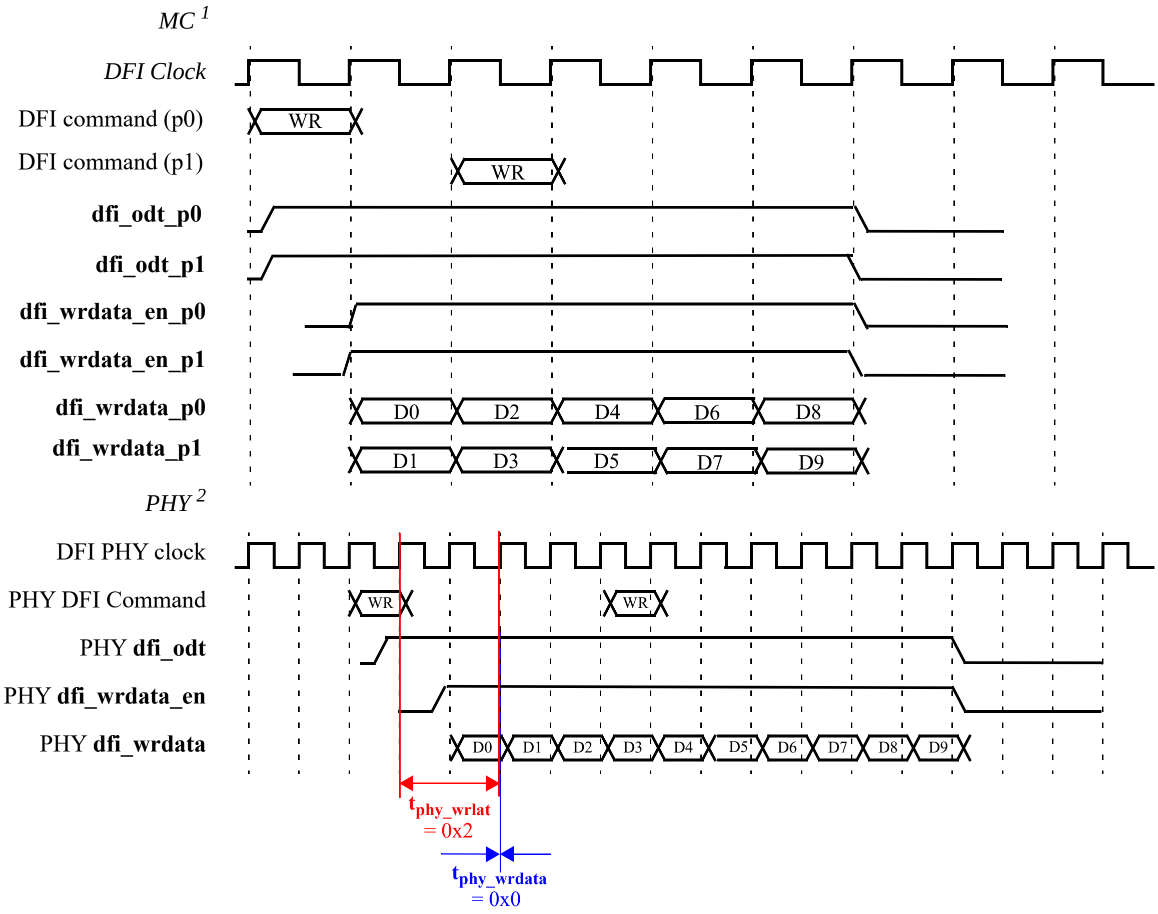
FIGURE 18. *DFI Write Data Bus for MC CRC Support Mode (Two Bursts starting in Phase 0)*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

FIGURE 19. *DFI Write Data Bus for MC CRC Support Mode (Two Back-to-Back Bursts)*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

4.5.3.2 PHY CRC Support ($\mathrm{phy}_{\mathrm{crc\_mode}}$ = 1)

If a PHY is capable of generating CRC data, the $\mathbf{phy}_{\mathbf{crc\_mode}}$ == 1 and the MC has the following requirements.

- The MC must disable its CRC generation logic so that CRC data is not transmitted across the DFI bus for write commands.
- The MC must assert the **dfi_wrdata_en** signal only for the data transmitted across the DFI bus, NOT for CRC.
- The MC spaces commands to handle an extended burst with CRC.
- The MC generates **dfi_odt** (**dfi_odt_pN** in frequency ratio systems) based on the DRAM burst length including CRC data. With CRC enabled, the MC may need to extend ODT.
- The MC needs to receive and capture error information from the PHY. These CRC write data errors are transmitted on the **dfi_alert_n** signal. In systems that support command parity or CRC, the MC must support a **dfi_alert_n** input and the PHY must support a **dfi_alert_n** output.

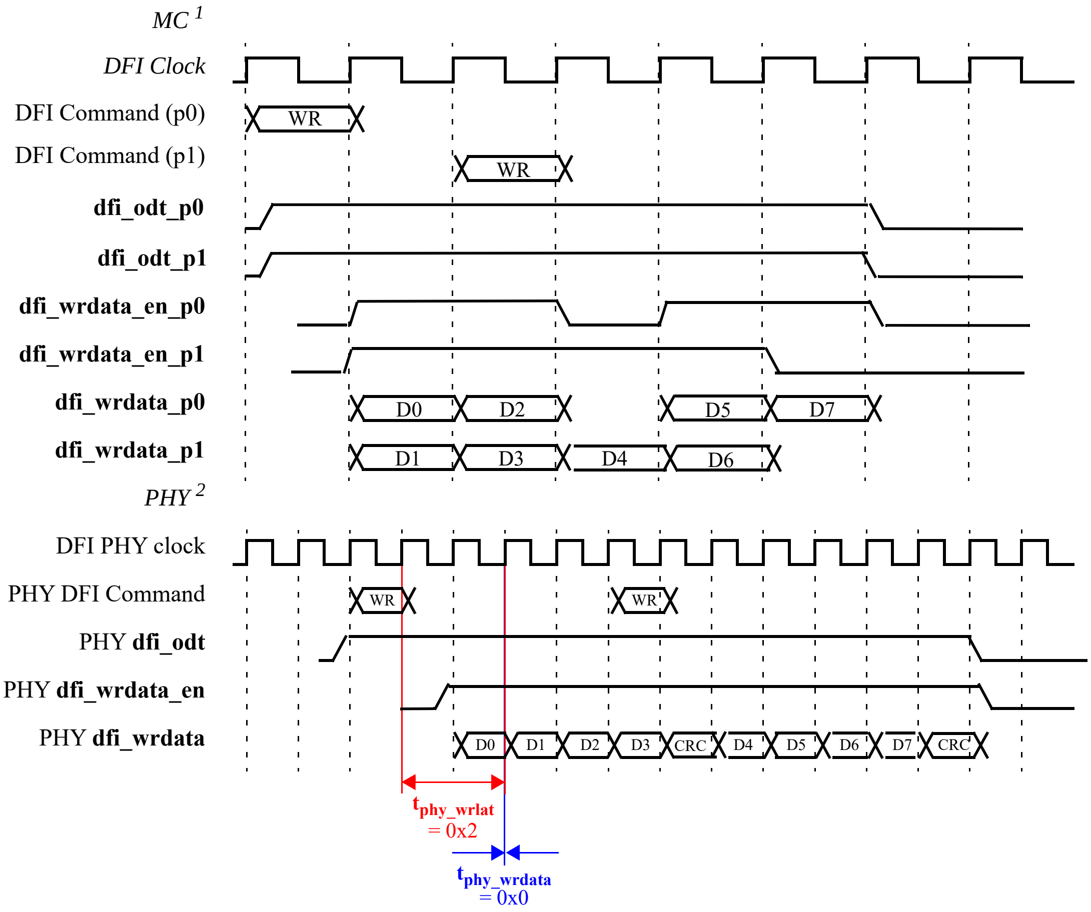
4.5.3.3 Burst Chop 4 with PHY CRC Support

In Burst Chop 4 (BC4) mode, the memory burst is extended for CRC, similar to a burst of 8 requiring a 10 UI transfer. When the PHY generates the CRC data, the controller does not adjust **dfi_wrdata_en_pN** to account for the transfer of CRC data words. In a system supporting dynamic burst lengths and BC4, the PHY can use the width of the **dfi_wrdata_en_pN** signal to determine whether a transfer is a burst of 8 or whether the transfer is a BC4 data transmission. The PHY can utilize this information to determine how to generate CRC and when to send the CRC data.

Figure 20 and Figure 21 show PHY outputs that include a single extra CRC data word, shown at the end of the burst (the location of the CRC data word is DRAM-dependent). The figures also show that **dfi_odt** assertion and de-assertion for PHY CRC support are identical to the MC CRC Support mode in which the **dfi_odt** signals are extended to cover both CRC and write data. The specific ODT timing requirements are dependent on the chip selects accessed and the system topology. The ODT signal timing shown is only an example of one potential solution.

Figure 20, “DFI Write Data Bus for PHY CRC Support Mode” illustrates a write BC4 case. This example shows a BC4 write followed by a burst of 8 write. With BC4, the MC only transmits 2 data words across the DFI bus. For both BC4 and burst of 8 commands, the CRC data is transmitted at the end of the burst, in UI9 and UI10.

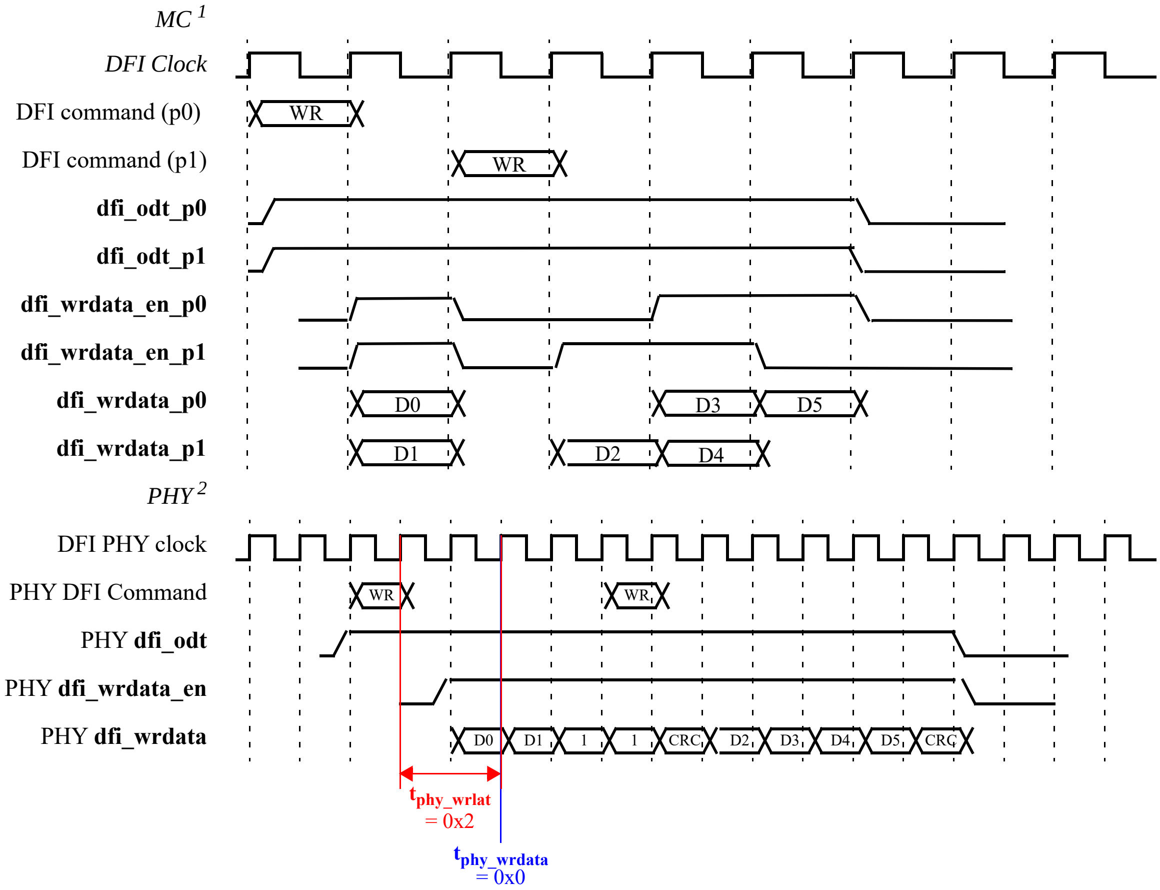
FIGURE 20. *DFI Write Data Bus for PHY CRC Support Mode*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

FIGURE 21. *DFI Write Data Bus for PHY CRC Support Mode with Burst Chop*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock diagram.

1. In the MC clock diagram, the timing shows the DFI bus signaling.
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

4.6 Read Transactions

The read data transaction handles the capture and return of data across the DFI bus.

The DFI read transaction includes the signals for read data enable (**dfi_rddata_en**), read data (**dfi_rddata**), the read data valid (**dfi_rddata_valid**), read data not valid (**dfi_rddata_dnv**) for LPDDR2 DRAMs, read data inversion (**dfi_rddata_dbi**) for DDR4/LPDDR4 systems, an optional target data chip select (**dfi_rddata_cs**) and the timing parameters $\mathbf{t_{rddata\_en}}$ and $\mathbf{t_{phy\_rdlat}}$.

The number of clocks of **dfi_rddata_en** assertion must correlate with the number of data transfers executed on the DRAM bus.

If used, the **dfi_rddata_cs** signal provides the target data path chip select value to each of the PHY data slices. For more information on these signals, refer to Section 3.3, “Read Data Interface”.

DFI read data must be returned from the PHY within a maximum delay defined by the sum of the $\mathbf{t_{rddata\_en}}$ and $\mathbf{t_{phy\_rdlat}}$ timing parameters. The $\mathbf{t_{rddata\_en}}$ is a fixed delay, but the $\mathbf{t_{phy\_rdlat}}$ is defined as a maximum value. The delay can be adjusted as long as both the MC and the PHY coordinate the change such that the DFI specification is maintained. Both parameters may be expressed as equations based on other fixed system parameters.

4.6.1 Read Transaction Sequence

The sequence for DFI read transactions is listed below; the effect of using the optional read data chip select or DBI is shown within brackets.

1. The read command is issued.
2. $\mathbf{t_{rddata\_en}}$ cycles elapse.

   [If using read data chip select, $\mathbf{t_{phy\_rdcslat}}$ cycles elapse.]

   [The PHY defines the $\mathbf{t_{phy\_rdcslat}}$ timing parameter to specify the desired alignment of the command to the **dfi_rddata_cs** signal; the PHY defines the $\mathbf{t_{phy\_rdcsgap}}$ timing parameter to specify the additional delay it requires between two consecutive commands that are targeting different chip selects. The PHY may require the MC to add additional delay beyond other system timing requirements to account for PHY-specific adjustments transitioning between chip selects.]

   [The gap timing requirement is system-specific and may only be applicable to certain chip-select-to-chip-select. It may not be applicable to other chip select transitions where the PHY is not required to make an internal adjustment on the chip select change. For example, a system may not require additional delay on transitions between chip select 0 and chip select 1, but may require additional delay when transitioning from chip select 0 to chip select 2. Accordingly, the interface does not require the gap timing to be applied on every chip select transition.]
3. The $\mathbf{t_{rddata\_en}}$ parameter defines the timing requirements between the read command on the DFI interface and the assertion of the **dfi_rddata_en** signal to maintain synchronization between the MC and the PHY for the start of contiguous read data expected on the DFI interface. The exact value of this parameter for a particular application is determined by memory system components. For non-contiguous read commands, $\mathbf{t_{rddata\_en}}$ cycles elapse, and the **dfi_rddata_en** signal is asserted on the DFI and remains asserted for the number of contiguous cycles that read data is expected.
4. For contiguous read commands, $\mathbf{t_{rddata\_en}}$ cycles after the first read command of the stream, the **dfi_rddata_en** signal is asserted and remains asserted for the entire length of the data stream.
5. One continuous assertion of the **dfi_rddata_en** signal may encompass data for multiple read commands. The **dfi_rddata_en** signal de-asserts to signify there is no more contiguous data expected from the DFI read command(s). Note that the **dfi_rddata_en** signal is not required to be asserted for any fixed number of cycles. The MC continues to drive **dfi_rddata_cs** a minimum of the data transfer width ($\mathbf{dfi_{rw\_length}}$) plus the gap timing ($\mathbf{t_{phy\_rdcsgap}}$); the MC may then drive the next chip select or a valid value (any valid chip select or inactive).

   The minimum time the chip select is guaranteed to remain driven to the PHY relative to the read command is defined by $\mathbf{t_{phy\_rdcslat}} + \mathbf{dfi_{rw\_length}} + \mathbf{t_{phy\_rdcsgap}}$.

   The maximum delay that can be achieved from the assertion of a new chip on **dfi_rddata_cs** and the corresponding **dfi_rddata_en** is limited to the maximum time the PHY has to make internal adjustments associated with changing the target chip select relative to the read data transfer ($\mathbf{t_{rddata\_en}} - \mathbf{t_{phy\_rdcslat}}$).

   The PHY must define the $\mathbf{t_{phy\_rdcslat}}$ and $\mathbf{t_{phy\_rdcsgap}}$ timing parameters to allocate the time between transactions to different chip selects necessary for internal adjustments.

6. The data is returned with the **dfi_rddata_valid** signal asserted.

[For LPDDR2 memory systems, the **dfi_rddata_dnv** signal has the same timing as the **dfi_rddata** signal.]

7. The associated read data signal (**dfi_rddata**) is sent.

[If DBI is enabled, the read data DBI signal (**dfi_rddata_dbi**) is sent coincident with the read data signal.]

Ten situations are presented in Figure 22, Figure 23, Figure 24, Figure 25, Figure 26, Figure 27, Figure 28, Figure 29, Figure 30, Figure 31 and Figure 32.

Figure 22, "Single Read Transaction of 2 Data Words" shows a single read transaction. In this case, the **dfi_rddata_en** signal is asserted for two cycles to inform the DFI that two cycles of DFI data are expected and data is returned $\mathbf{t_{phy\_rdlat}}$ cycles after the **dfi_rddata_en** signal assertion.

FIGURE 22. *Single Read Transaction of 2 Data Words*

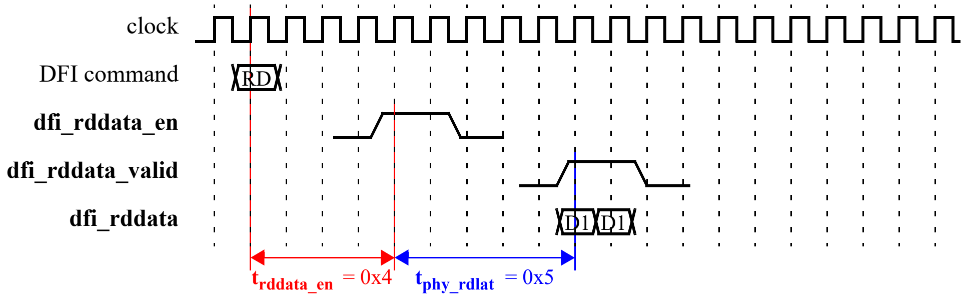


Figure 23, "Single Read Transaction of 4 Data Words" shows a single read transaction where the data is returned in less than the maximum delay. The data returns one cycle less than the maximum PHY read latency.

FIGURE 23. *Single Read Transaction of 4 Data Words*

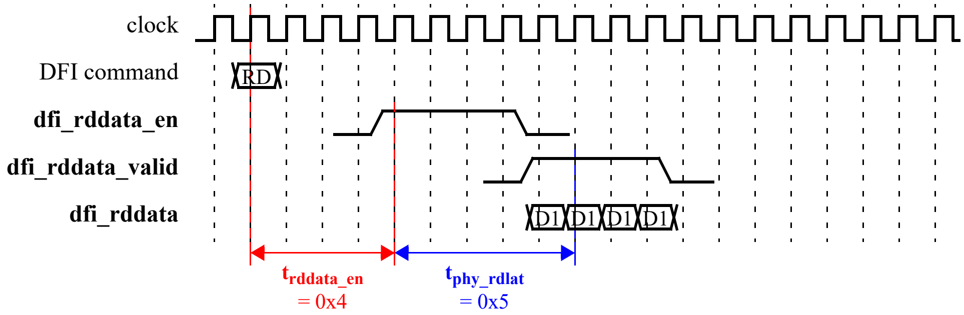


Figure 24, "Back-to-Back Read Transactions with First Read Burst Interrupted (DDR1 Example BL = 8)" shows an interrupted read command. The **dfi_rddata_en** signal must be asserted for 4 cycles for each of these read transactions.

However, since the first read is interrupted, the **dfi_rddata_en** signal is asserted for a portion of the first transaction and the complete second transaction. In this case, the **dfi_rddata_en** signal will not de-assert between read commands.

FIGURE 24. *Back-to-Back Read Transactions with First Read Burst Interrupted (DDR1 Example BL = 8)*

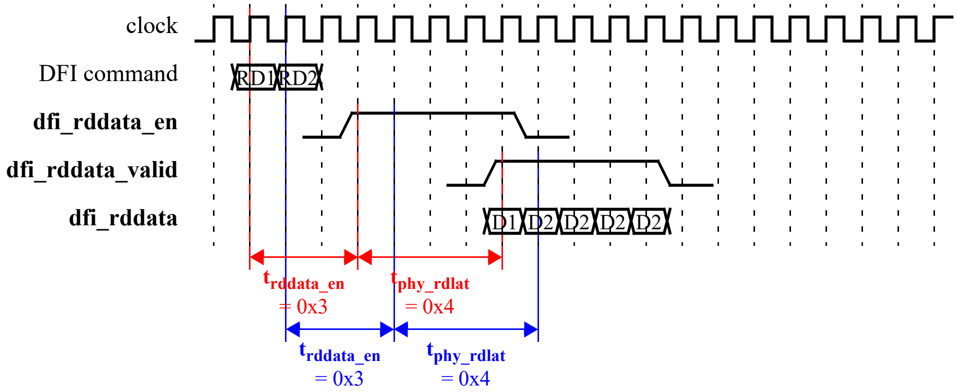
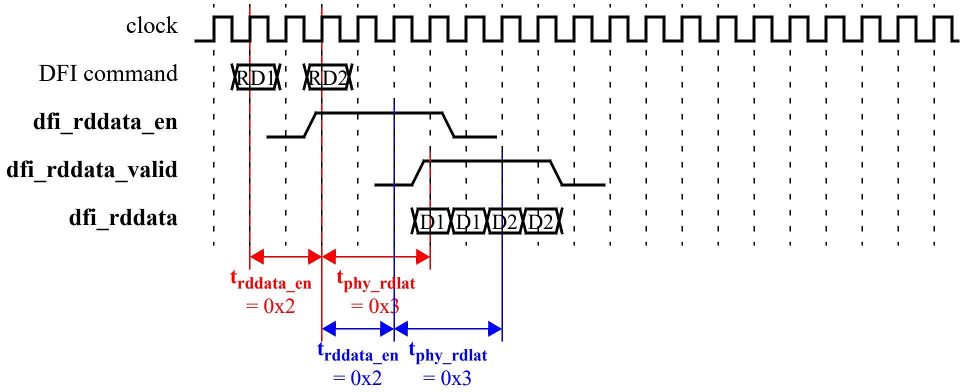


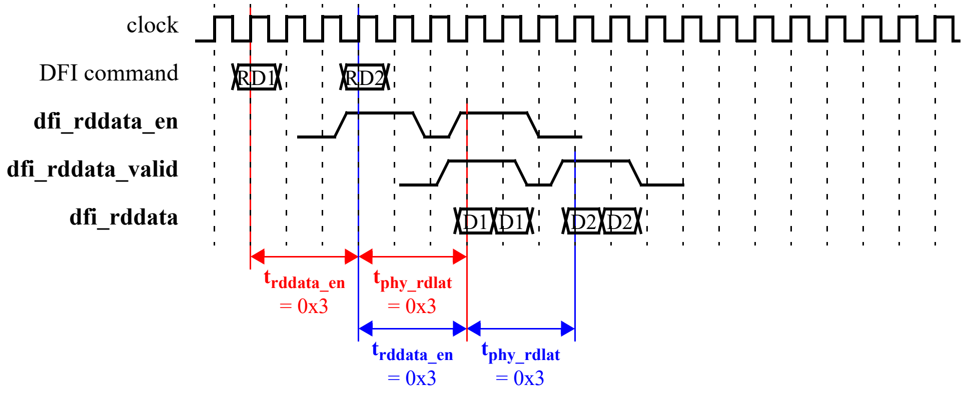
Figure 25, “Two Independent Read Transactions (DDR1 Example)” and Figure 26, “Two Independent Read Transactions (DDR2 Example)” also show two complete read transactions. The **dfi_rddata_en** signal is asserted for two cycles for each read transaction. In Figure 25, “Two Independent Read Transactions (DDR1 Example)”, the values for the timing parameters are such that the read data is returned in a contiguous data stream for both transactions. Therefore, the **dfi_rddata_en** signal and the **dfi_rddata_valid** signal are each asserted for the complete read data stream.

FIGURE 25. *Two Independent Read Transactions (DDR1 Example)*



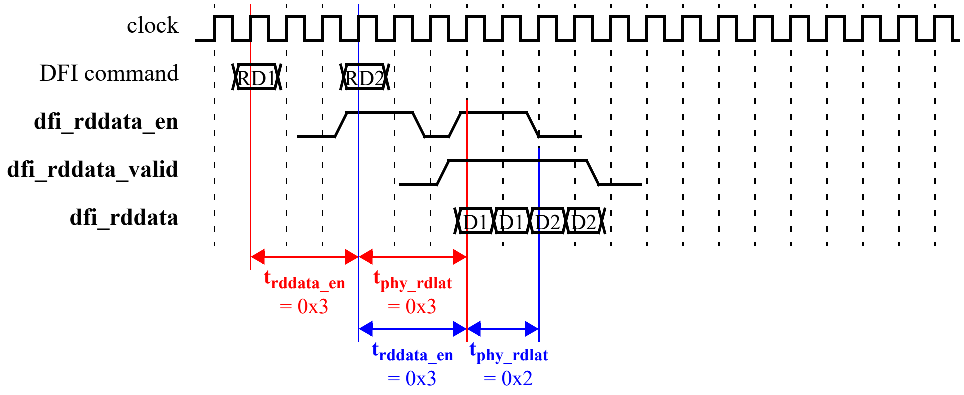
In Figure 26, “Two Independent Read Transactions (DDR2 Example)”, the $\mathbf{t_{rddata\_en}}$ timing and the timing between the read commands causes the **dfi_rddata_en** signal to be de-asserted between commands. As a result, the **dfi_rddata_valid** signal is de-asserted between commands and the **dfi_rddata** stream is non-contiguous.

FIGURE 26. *Two Independent Read Transactions (DDR2 Example)*



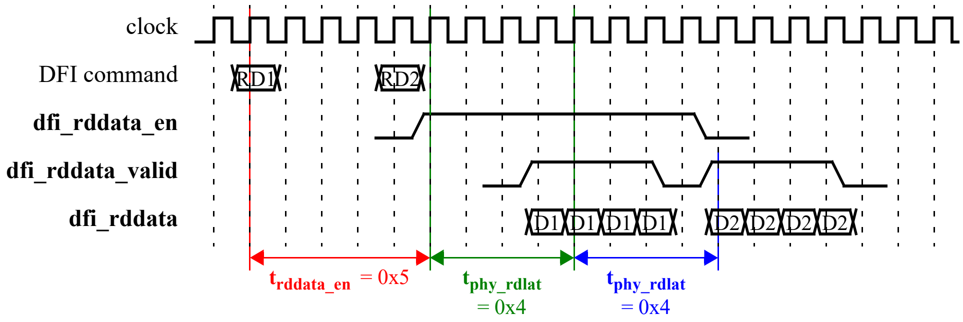
In Figure 27, “Two Independent Read Transactions (DDR2 Example)”, the effective $\mathbf{t_{phy\_rdlat}}$ for the two transactions is different. This results in a situation in which the **dfi_rddata_valid** signal remains asserted across commands and the **dfi_rddata** stream is contiguous.

FIGURE 27. *Two Independent Read Transactions (DDR2 Example)*



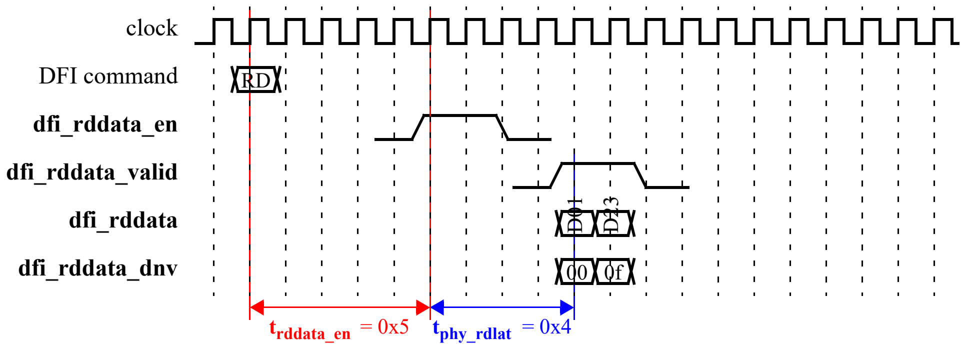
The data may return to the DFI in fewer cycles than maximum delay. In Figure 28, “Two Independent Read Transactions (DDR3 Example)”, the first read data transfer is returned in three cycles, even though the **t_{phy_rdlat}** timing parameter is set to four cycles. The second read data transfer is returned in the maximum of four cycles.

FIGURE 28. *Two Independent Read Transactions (DDR3 Example)*



LPDDR2 and LPDDR3 DRAMs define a new transaction type of mode register read (MRR). From the DFI perspective, a mode register read is handled like any other read command and utilizes the same signals. Figure 29, “Example MRR Transactions with LPDDR2” shows an MRR transaction for a LPDDR2 or LPDDR3 memory device.

FIGURE 29. *Example MRR Transactions with LPDDR2*



In Figure 30, "DFI Read Data Transfer Illustrating dfi_rddata_valid Definition", the **dfi_rddata_valid** signals are transferred independently. This figure shows a one-to-one correspondence between the data words for **dfi_rddata_en** and **dfi_rddata_valid**; the **dfi_rddata_valid** words do not need to be contiguous.

FIGURE 30. *DFI Read Data Transfer Illustrating **dfi_rddata_valid** Definition*

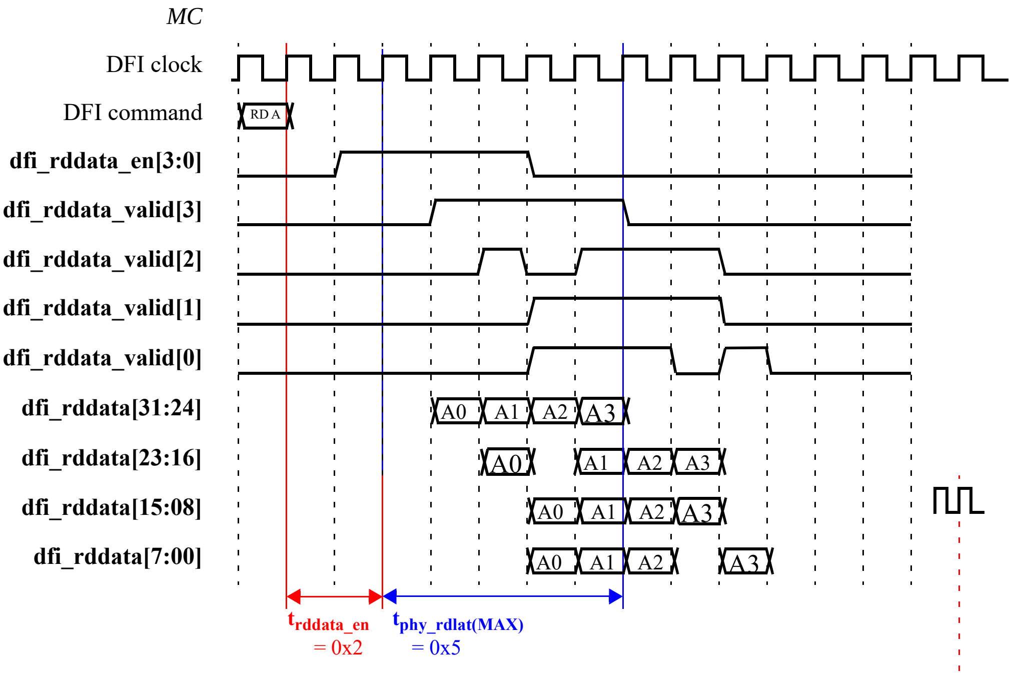
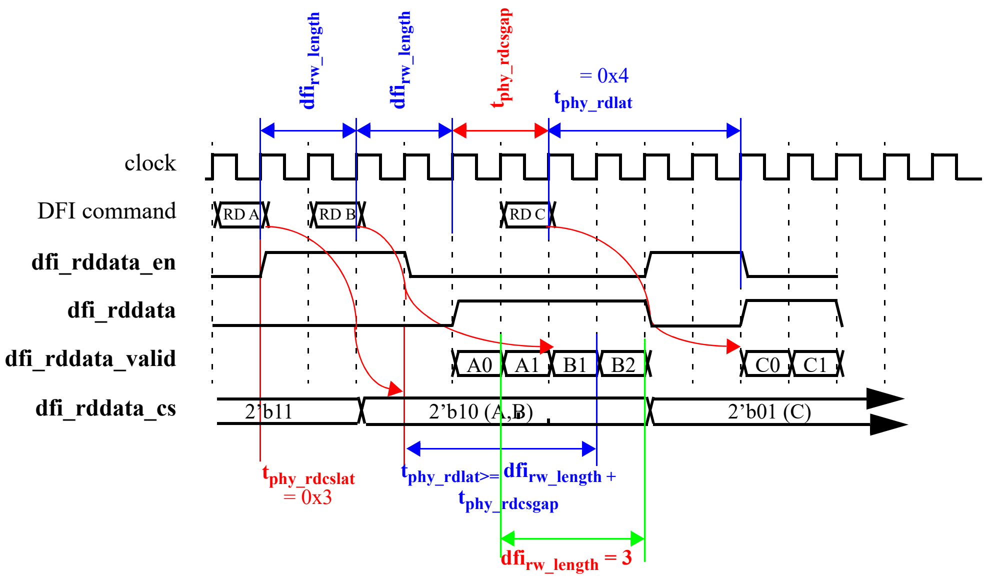


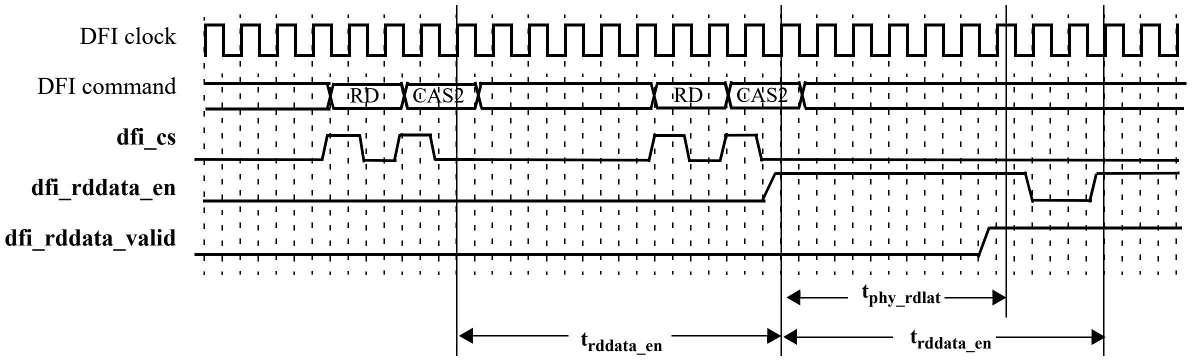
Figure 31, "Read Commands Utilizing dfi_rddata_cs" shows three read commands, with a gap between the second and third commands.

FIGURE 31. *Read Commands Utilizing **dfi_rddata_cs***



DFI timing parameters for read and write commands will be referenced from the second tick of the second command in the same manner that JEDEC LPDDR4 references RL and WL.

FIGURE 32. *LPDDR4 Read Command*



4.6.2 DBI - Read

DBI is an optional DFI feature used with read data transmissions. The $\mathbf{phy_{dbi\_mode}}$ parameter is only needed when DBI is supported in DFI. If DBI is required in a system, DRAM DBI input must be received and used to selectively invert data for read commands.

For more information on the $\mathbf{phy_{dbi\_mode}}$ parameter, refer to Table 9, “Write Data Programmable Parameters”.

4.6.2.1 MC DBI Support ($\mathbf{phy_{dbi\_mode}}$ = 0)

The PHY captures the DRAM read DBI data and transmits the data over the DFI to the MC using the **dfi_rddata_dbi** signals. For more information on the DBI signals, refer to Table 7, “Write Data Signals” and Table 10, “Read Data Signals”. The timing of the **dfi_rddata_dbi** signal is identical to the timing of the **dfi_rddata** signal, and the DBI data is sent coincident with the corresponding **dfi_rddata** bus.

For frequency ratio systems, the **dfi_rddata_dbi** signal is extended, similar to **dfi_rddata**, with a signal defined per phase. For example, with a 4:1 frequency system, the DBI information is transmitted across the **dfi_rddata_dbi_w0**, **dfi_rddata_dbi_w1**, **dfi_rddata_dbi_w2** and **dfi_rddata_dbi_w3** signals.

The timing of the phase outputs for DBI is identical to the **dfi_rddata_wN** outputs, with the data returned in a rolling order.

4.6.2.2 PHY DBI Support ($\mathbf{phy_{dbi\_mode}}$ = 1)

When the PHY generates the write DBI data, the **dfi_rddata_dbi** signal is not needed.

4.7 Update

The DFI contains signals to support MC-initiated and PHY-initiated update processes. The signals used in the update interface are: **dfi_ctrlupd_req**, **dfi_ctrlupd_ack**, **dfi_phyupd_req**, **dfi_phyupd_type** and **dfi_phyupd_ack**. The idle state timing parameters used in the update interface are: $\mathbf{t_{ctrl\_delay}}$ and $\mathbf{t_{wrdata\_delay}}$.

For more information on the signals, refer to Section 3.4, “Update Interface”. For more information on the idle state timing parameters, refer to Table 6, “Control Timing Parameters” and Table 8, “Write Data Timing Parameters”.

4.7.1 MC-Initiated Update

During normal operation, the MC may encounter idle time during which no commands are issued to the DRAMs and all outstanding read and write data have been transferred on the DFI bus and the write data transfer has completed on the memory bus. Assertion of the **dfi_ctrlupd_req** signal indicates the control, read and write interfaces on the DFI are idle. While the **dfi_ctrlupd_ack** signal is asserted, the DFI bus may only be used for commands related to the update process.

The MC guarantees that **dfi_ctrlupd_req** signal is asserted for at least $\mathbf{t_{ctrlupd\_min}}$ cycles, allowing the PHY time to respond. The PHY may respond to or ignore the update request. To acknowledge the request, the **dfi_ctrlupd_ack** signal must be asserted before $\mathbf{t_{ctrlupd\_max}}$ expires and while the **dfi_ctrlupd_req** signal is asserted. The **dfi_ctrlupd_ack** signal must de-assert at least one cycle before $\mathbf{t_{ctrlupd\_max}}$ expires.

The MC should hold the **dfi_ctrlupd_req** signal as long as the **dfi_ctrlupd_ack** signal is asserted, although the MC could de-assert the **dfi_ctrlupd_req** signal to disconnect the handshake through the disconnect protocol. The PHY must de-assert the **dfi_ctrlupd_ack** signal before $\mathbf{t_{ctrlupd\_max}}$ expires.

Figure 33, "MC-Initiated Update Timing Diagram" shows that the DFI does not specify the number of cycles after the **dfi_ctrlupd_ack** signal de-asserts before the **dfi_ctrlupd_req** signal de-asserts.

FIGURE 33. *MC-Initiated Update Timing Diagram*

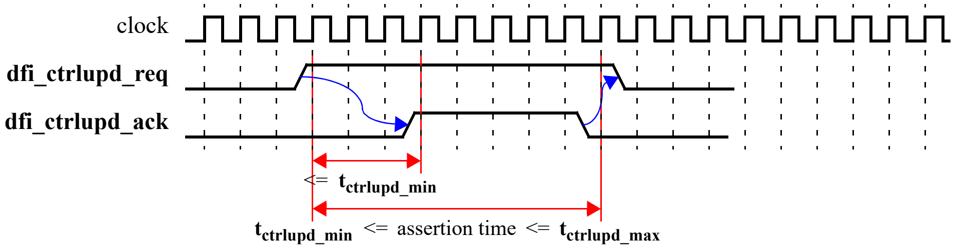
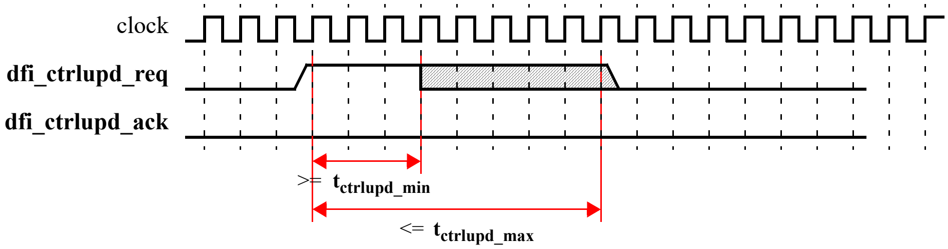


Figure 34, "MC-Initiated Update with No Response" shows the important point that the **dfi_ctrlupd_ack** signal is not required to assert when the **dfi_ctrlupd_req** signal is asserted. The MC must assert the **dfi_ctrlupd_req** signal for at least $\mathbf{t_{ctrlupd\_min}}$ within every $\mathbf{t_{ctrlupd\_interval}}$ cycle, but the total number of cycles that the **dfi_ctrlupd_req** signal is asserted must not exceed $\mathbf{t_{ctrlupd\_max}}$.

FIGURE 34. *MC-Initiated Update with No Response*



4.7.2 PHY-Initiated Update

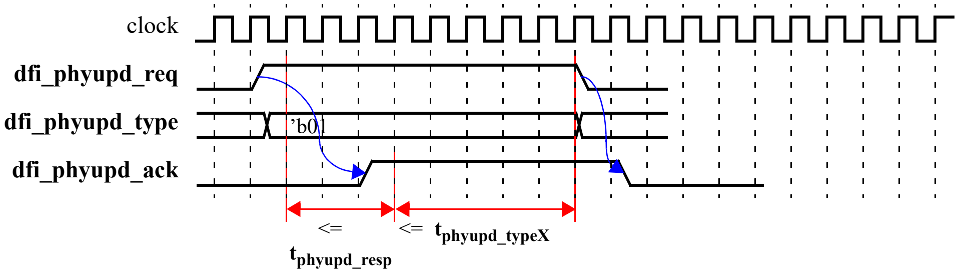
The PHY may also trigger the DFI to enter an idle state. This update process utilizes three signals: **dfi_phyupd_req**, **dfi_phyupd_type** and **dfi_phyupd_ack**. The **dfi_phyupd_req** signal indicates the need for idle time on the DFI, the **dfi_phyupd_type** signal defines the type of update required, and the **dfi_phyupd_ack** signal is the MC's response signal. Four update types are specified by the DFI.

To request an update, the **dfi_phyupd_type** signal must be valid when the **dfi_phyupd_req** signal is asserted. The $\mathbf{t_{phyupd\_typeX}}$ parameters indicate the maximum number of cycles of idle time on the DFI control, read and write data interfaces being requested. The **dfi_phyupd_ack** signal must assert within $\mathbf{t_{phyupd\_resp}}$ cycles after the assertion of the **dfi_phyupd_req** signal.

When the **dfi_phyupd_ack** signal is asserted, it should remain asserted until the **dfi_phyupd_req** signal de-asserts unless the Controller wishes to disconnect the handshake. The **dfi_phyupd_req** signal must de-assert before $\mathbf{t_{phyupd\_typeX}}$ cycles have expired. The **dfi_phyupd_ack** signal must de-assert following the detection of the **dfi_phyupd_req** signal de-assertion. The **dfi_phyupd_ack** signal for the previous transaction must be de-asserted before the **dfi_phyupd_req** signal can re-assert. While the **dfi_phyupd_ack** signal is asserted, the DFI bus may only be used for commands related to the update process.

Figure 35, "PHY-Initiated Update Timing Diagram" shows that the MC must respond to a PHY update request, unlike MC-initiated updates shown in Figure 33, "MC-Initiated Update Timing Diagram" and Figure 34, "MC-Initiated Update with No Response".

FIGURE 35. *PHY-Initiated Update Timing Diagram*



4.7.3 DFI Idle

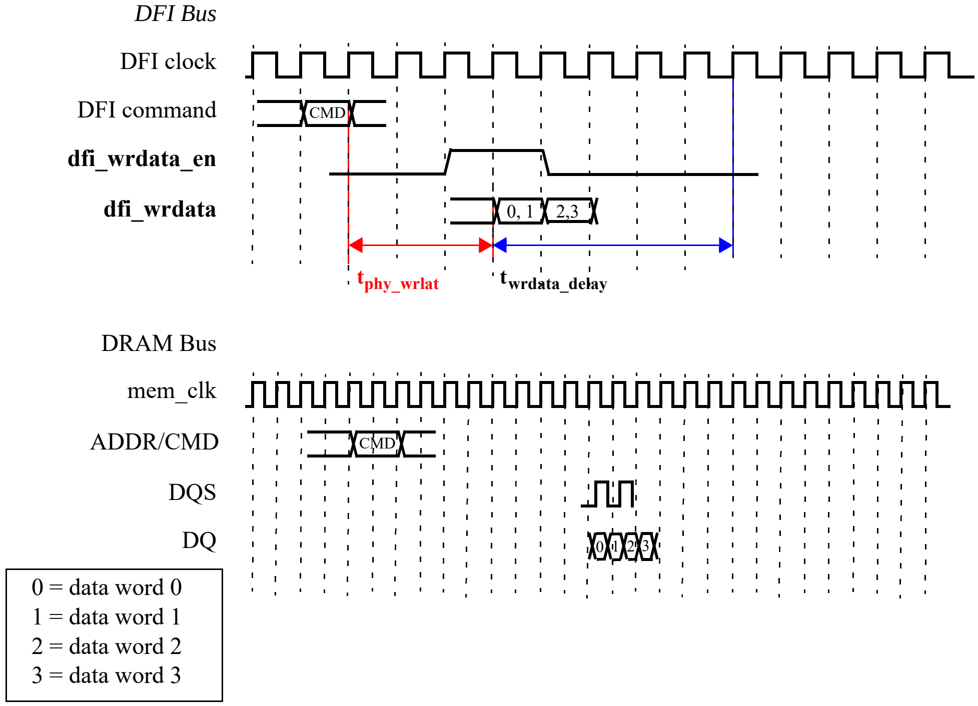
To prevent the condition where a PHY may disrupt the write data transfer on the memory bus, the write data transfer on the memory bus must complete before the DFI idle state is reached.

The $\mathbf{t_{wrdata\_delay}}$ idle state timing parameter defines the number of DFI clocks from **dfi_wrdata_en** to the completion of the write data transfer on the memory bus. Since the requirement for an idle bus state following a write is generally an infrequent event relative to the overall traffic pattern, the accuracy of the setting should not be a performance issue, so it may be set to a larger value.

If a PHY has no dependency between completing a write data transfer on the DFI bus and the idle state, the $\mathbf{t_{wrdata\_delay}}$ parameter can be set to zero.

When a PHY does have a dependency between completing a write data transfer on the DFI bus and the idle state, the $\mathbf{t_{wrdata\_delay}}$ parameter should be set to a sufficiently large value to accommodate the write data width, flight time through the PHY and worst-case timing on the memory bus.

FIGURE 36. *Bus Idle State Timing Parameter -* t_{wrdata_delay}

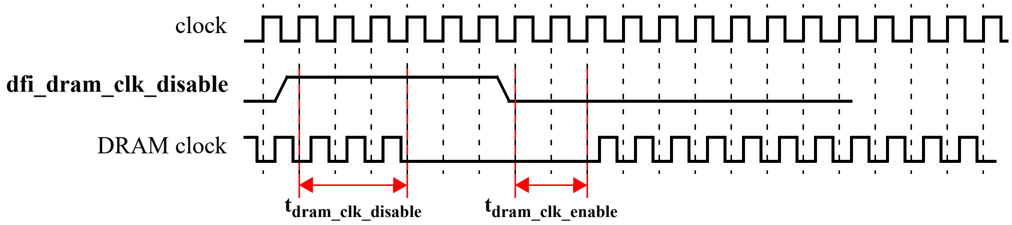


4.8 DFI Clock Disabling

The DFI contains a **dfi_dram_clk_disable** signal which controls the DRAM clock signal to the DRAM device(s). In the default state, the DRAM clock functions normally and the **dfi_dram_clk_disable** bits are all de-asserted. If the system requires the clocks of the memory device(s) to be disabled, the **dfi_dram_clk_disable** signal is asserted. For more information on the **dfi_dram_clk_disable** signal, refer to Section 3.5, “Status Interface”.

Figure 37, “DRAM Clock Disable Behavior” shows that two timing parameters $\mathbf{t_{dram\_clk\_disable}}$ and $\mathbf{t_{dram\_clk\_enable}}$ indicate the number of DFI cycles that the PHY requires to respond to the assertion and de-assertion of the **dfi_dram_clk_disable** signal. The $\mathbf{t_{dram\_clk\_disable}}$ value determines the number of DFI cycles in which a rising edge of the **dfi_dram_clk_disable** signal affects the DRAM clock and $\mathbf{t_{dram\_clk\_enable}}$ sets the number of cycles required for the DRAM clock to be active again.

FIGURE 37. *DRAM Clock Disable Behavior*



4.9 Frequency Ratios Across the DFI

In a DDR memory subsystem, it may be advantageous to operate the PHY at a higher frequency than the MC. If the PHY operates at a multiple of the MC frequency, the PHY transfers data at a higher data rate relative to the DFI clock and the MC has the option to execute multiple commands in a single DFI clock cycle. The DFI is defined at the MC to PHY boundary and therefore operates in the clock frequency domain of the MC.

The MC clock is always the DFI clock and all DFI signals are referenced from the MC clock.

The DFI specification supports a 1:1, 1:2 or 1:4 MC to PHY frequency ratio, defining the relationship of the reference clocks for the MC and the PHY. The DFI DDR PHY clock is always the same frequency as the DRAM clock, which is 1/2 the data rate for the memory.

DFI signals may be sent or received on the DFI PHY clock, provided the signals reference the rising edge of the DFI clock and the clock is phase aligned. The MC communicates frequency ratio settings to the PHY on the **dfi_freq_ratio** signal. This signal is only required for devices using this frequency ratio protocol.

The frequency ratio protocol affects the write data and read data interfaces, including read data rotation and resynchronization. Frequency ratio also affects CA and CRC parity errors.

For information on how frequency ratio affects CA and CRC parity errors, refer to Section 4.11.3, “CA Parity and CRC Errors in Frequency Ratio Systems”.

4.9.1 Frequency Ratio Clock Definition

The DFI clock and the DFI PHY clock must be phase-aligned and at a 1:2 or 1:4 frequency ratio relative to one another. Some DFI signals from the MC to the PHY must communicate information about the signal in reference to the DFI PHY clock to maintain the correct timing information. Therefore, the DFI PHY clock is described in terms of phases, where the number of clock phases for a system is the ratio of the DFI PHY clock to the DFI clock.

Figure 38, “Frequency Ratio 1:2 Phase Definition” and Figure 39, “Frequency Ratio 1:4 Phase Definition” show the clock phase definitions for 1:2 and 1:4 frequency ratio systems.

FIGURE 38. *Frequency Ratio 1:2 Phase Definition*

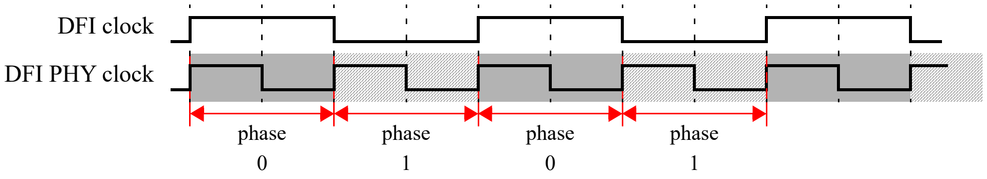
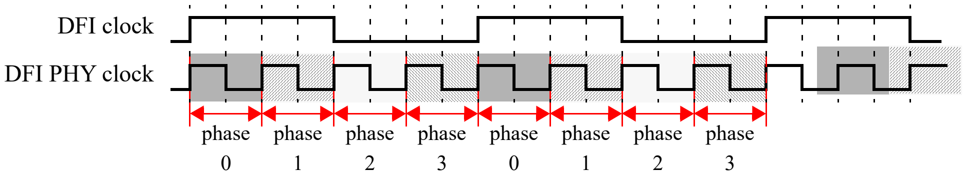


FIGURE 39. *Frequency Ratio 1:4 Phase Definition*



4.9.2 Interface Signals with Frequency Ratio Systems

Write data and read data signals are defined on a per-phase basis and all signal timing is in reference to the DFI clock. The PHY must account for any assertions based on the DFI clock. Any signals driven by the PHY must only change during phase 0 of the DFI PHY clock to allow the MC the full DFI clock to capture the signal change.

The DFI specification supports the ability to send a unique command on each phase of the DFI PHY clock. To communicate this information to the PHY, the DFI specification defines commands for a frequency ratio system in a vectored format. The PHY must maintain this information to preserve the timing relationships between commands and data. Therefore, for frequency ratio systems, the control signal interface, the write data interface and the read data enable signal are all suffixed with a "_pN" where N is the phase number. As an example, for a 1:2 frequency ratio system, instead of a single **dfi_address** signal, there are 2 signals: **dfi_address_p0** and **dfi_address_p1**. The read data signal, read data valid and read data not valid signals are suffixed with a "_wN" where N is the DFI data word. More information on the read data interface for frequency ratio systems is provided in Section 4.9.4, "Read Data Interface in Frequency Ratio Systems". The phase 0 or DFI data word 0 suffixes are not required.

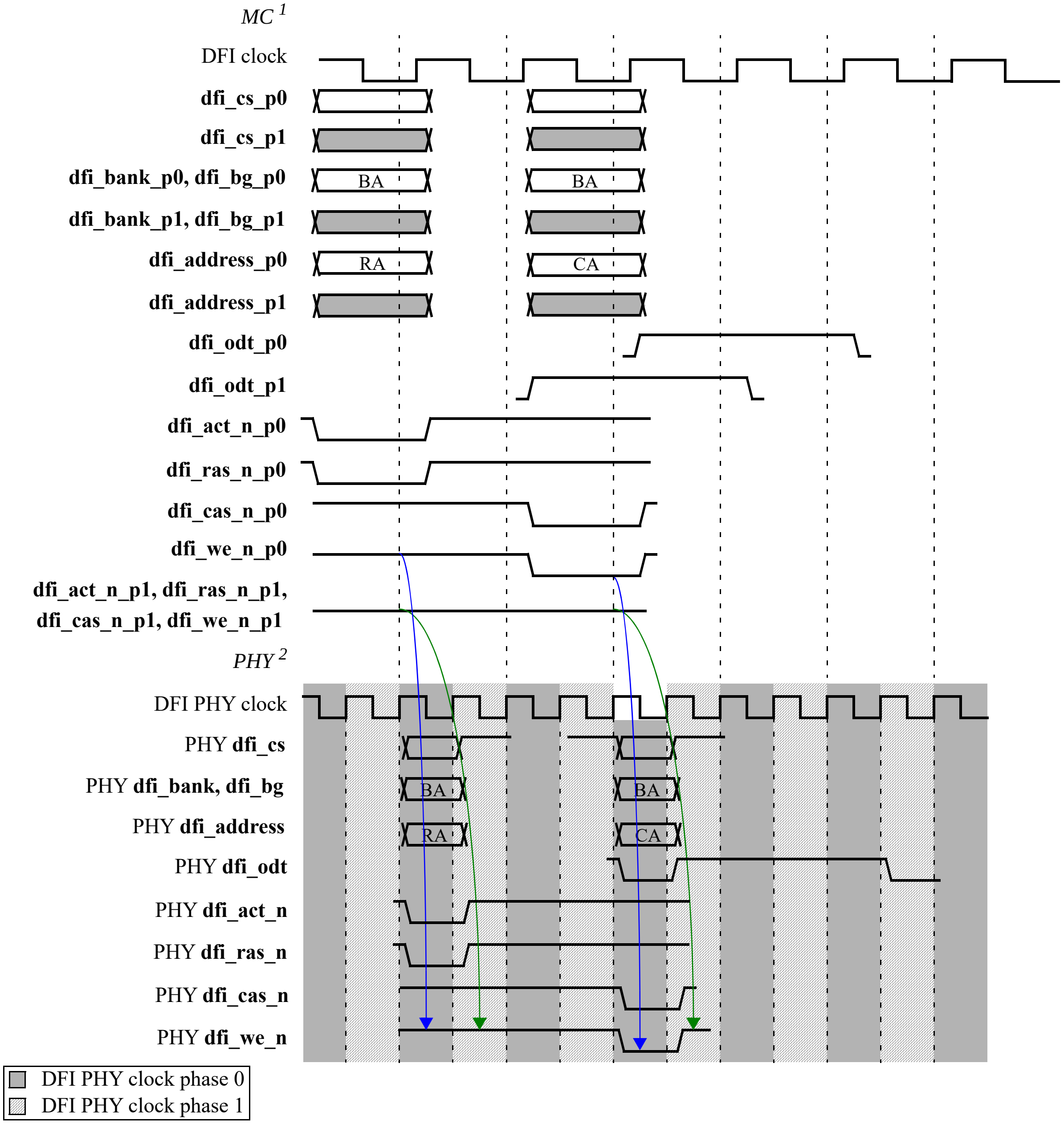
There is flexibility in system setup for frequency ratio systems. The MC may be implemented to support command output on a single phase or on multiple phases. Even if multiple phases are supported, the MC is not required to implement or drive every phase of a signal. Only phases where a command is sent must be implemented and driven. The exceptions to the rule are the **dfi_cke_pN** and **dfi_odt_pN** signals. These two signals are not necessarily driven in the same phase as the rest of the command. Therefore, these signals must be implemented for all phases of the clock to allow flexibility in timing.

The PHY must be able to accept a command on all phases to be DFI compliant. If the MC is only using certain phases, the PHY must be appropriately connected to properly interpret the command stream.

There is no requirement that signals must be implemented in the same way across the interfaces. For example, in a 2T implementation, the **dfi_ras_n_pN**, **dfi_cas_n_pN** and **dfi_we_n_pN** signals may be driven by the MC on all clock phases, but the **dfi_cs_pN** signal may only be driven by the MC on half of the phases.

Figure 40, “Example 1:2 Frequency Ratio Command Stream” illustrates an example command stream for a 1:2 frequency ratio system and how the PHY in this system would interpret the DFI signals. In this example, a command is only sent on phase 0; the values of phase 0 and phase 1 commands will be different. ODT information is provided on both phases. The command bus signals are not always the same value and are not always equal to one.

FIGURE 40. *Example 1:2 Frequency Ratio Command Stream*



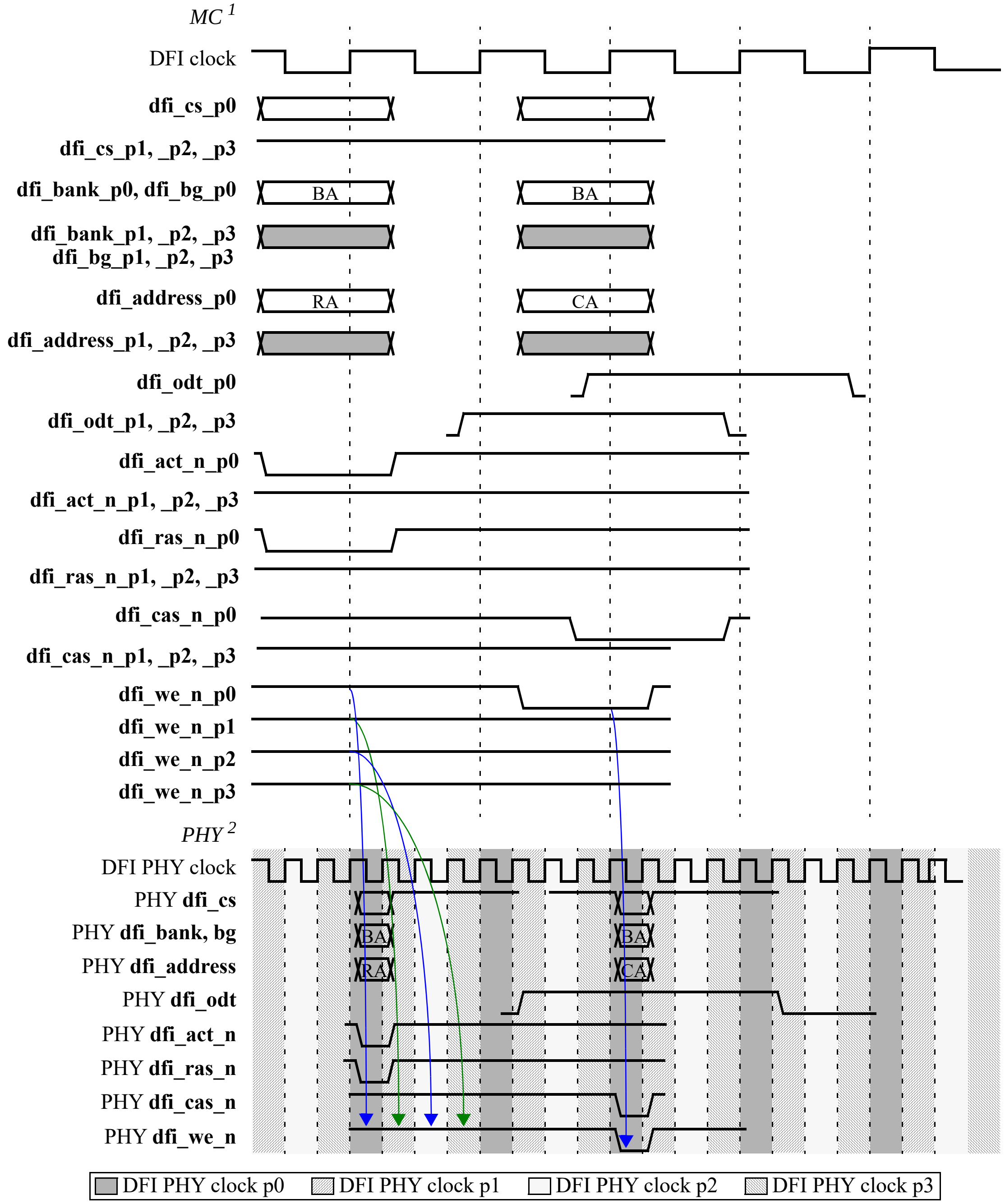
NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

Figure 41, “Example 1:4 Frequency Ratio Command Stream” represents the same example, in a 1:4 frequency ratio system. The command is only sent on phase 0 and ODT information is provided on all phases.

FIGURE 41. *Example 1:4 Frequency Ratio Command Stream*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

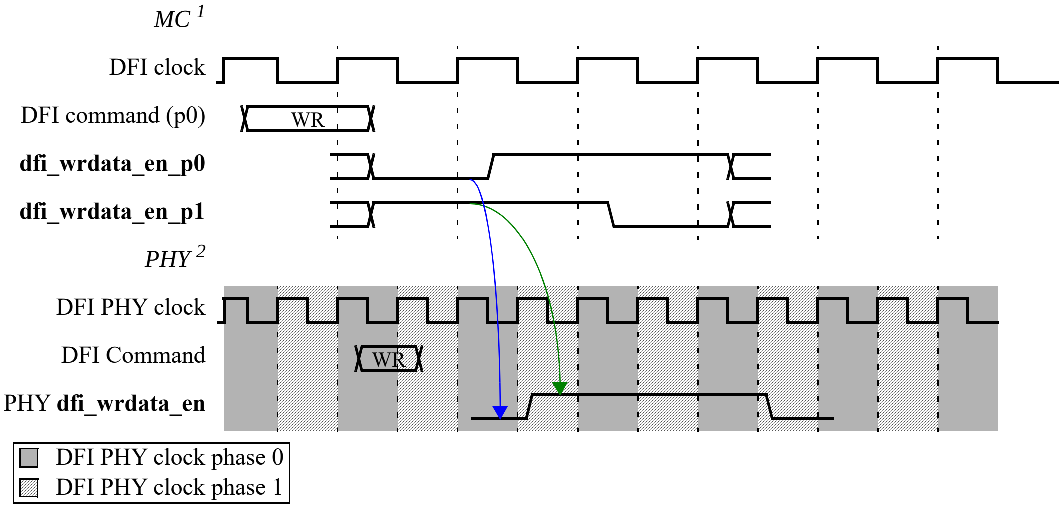
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

4.9.3 Write Data Interface in Frequency Ratio Systems

The write data enable signal (**dfi_wrdata_en_pN**) indicates to the PHY that valid **dfi_wrdata** will be transmitted in $\mathbf{t_{phy\_wrdata}}$ DFI PHY clock cycles and its width defines the number of data phases of the write. In order to communicate this information to the PHY, the phase information must be encoded within the signal. Therefore, this signal is also vectored into multiple signals based on the frequency ratio. Similar to the DFI command, each signal is associated with a phase of the DFI PHY clock.

Figure 42, “1:2 Frequency Ratio Write Data Example” demonstrates how a vectored **dfi_wrdata_en_pN** signal is interpreted by the PHY in a 1:2 frequency ratio system.

FIGURE 42. *1:2 Frequency Ratio Write Data Example*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

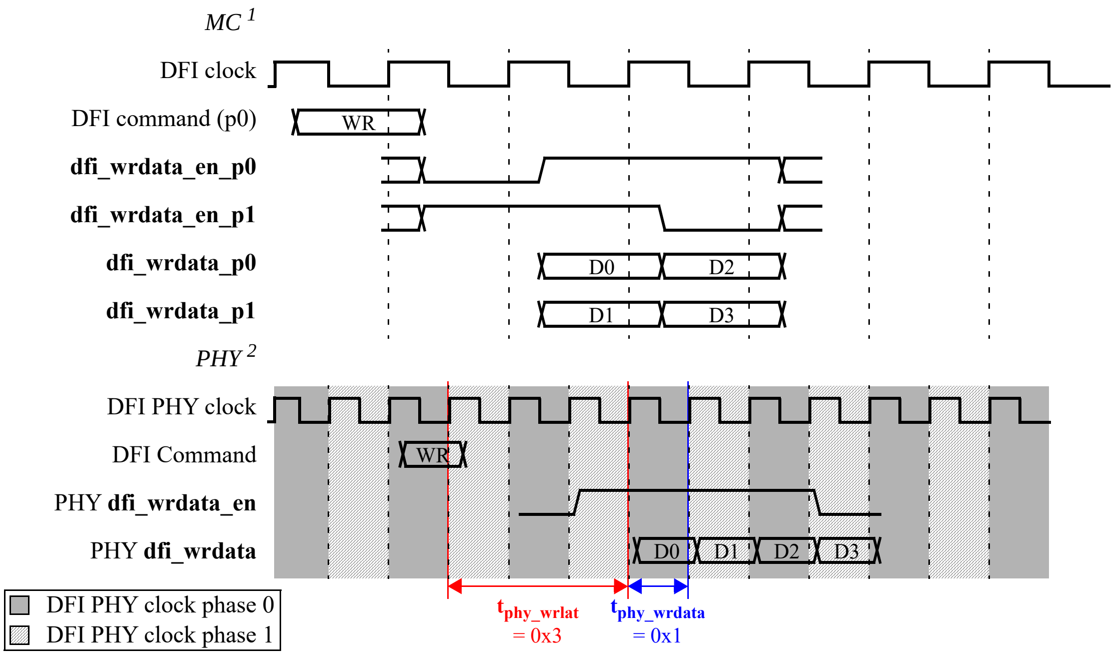
1. In the MC clock diagram, the timing shows the DFI bus signaling.
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

For matched frequency systems, the DFI write data bus width is generally twice the width of the DRAM data bus. For frequency ratio systems, this DFI write data bus width is proportional to the frequency ratio to allow all of the write data that the memory requires to be sent in a single DFI clock cycle. The write data must be delivered with the DFI data words aligned in ascending order.

The timing parameters $\mathbf{t_{phy\_wrlat}}$ and $\mathbf{t_{phy\_wrdata}}$ apply in frequency ratio systems in the same way as in matched frequency systems. The $\mathbf{t_{phy\_wrlat}}$ parameter defines the delay from the write command to the **dfi_wrdata_en_pN** signal. The $\mathbf{t_{phy\_wrdata}}$ parameter defines the delay from the **dfi_wrdata_en_pN** signal to when data is driven on the **dfi_wrdata_pN** signal. These timing parameters are defined in terms of DFI PHY clocks and are measured relative to how the PHY interprets the data.

Figure 43, “1:2 Frequency Ratio Aligned Write Data Example” shows how data is received by the PHY in a situation where the data is sent aligned, but the enable signals are not aligned.

FIGURE 43. *1:2 Frequency Ratio Aligned Write Data Example*



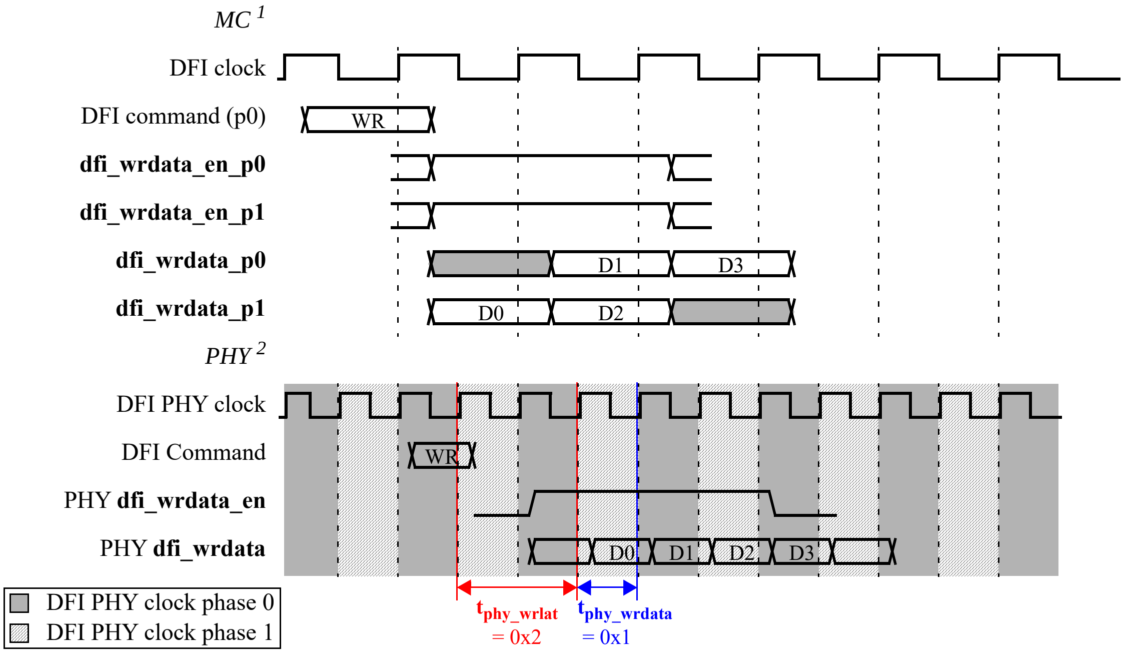
NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

In Figure 44, “1:2 Frequency Ratio Aligned Write Enable Example”, the enable signals are sent aligned, but the data is not aligned. The MC sends the first beat of data on the phase 1 data signal. The write data must be sent un-aligned to achieve the proper relationship between the command and data.

FIGURE 44. *1:2 Frequency Ratio Aligned Write Enable Example*



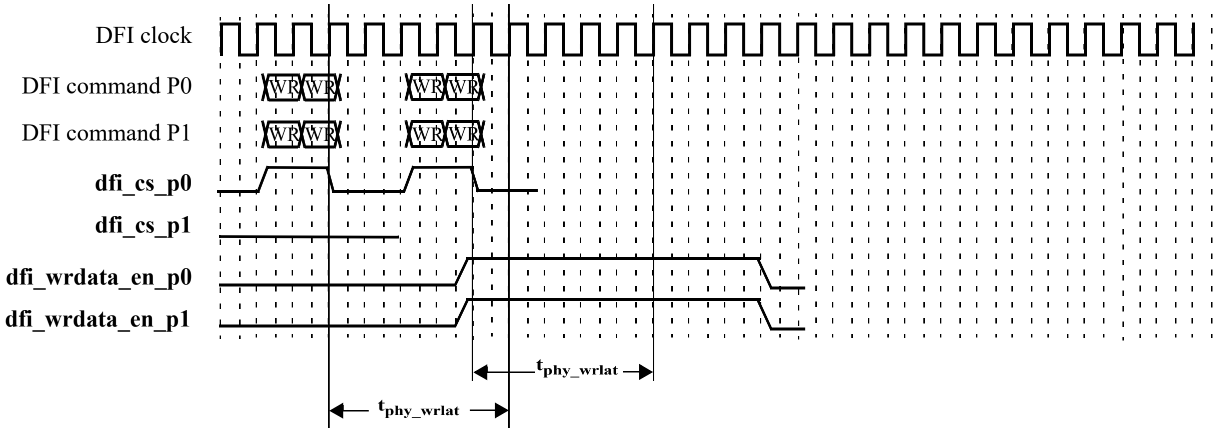
NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock diagram.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

DFI timing parameters for read and write commands will be referenced from the second tick of the second command in the same manner that JEDEC LPDDR4 references RL and WL.

FIGURE 45. *LPDDR4 Write Command, 1:2 Frequency Ratio*



4.9.4 Read Data Interface in Frequency Ratio Systems

Similar to the write data enable signal, the read data enable signal (**dfi_rddata_en_pN**) defines the number of clocks between the read command and the read data, and its width defines the number of data phases of the read. The PHY sends read data to the MC on the **dfi_rddata_wN** buses whenever read data is available, asserting the associated **dfi_rddata_valid_wN** signals to inform the MC which buses contain valid data. Unlike the read data enable signal which correlates to the phase of the DFI PHY clock, the read data, read data valid and read data not valid signals are all vectored with the DFI data word suffix.

For matched frequency systems, the DFI read data bus width is generally twice the width of the DRAM data bus. For frequency ratio systems, this DFI read data bus width is proportional to the frequency ratio to allow all of the read data that the memory returns to be sent in a single DFI clock cycle. The read data must be delivered with the DFI data words aligned in ascending order.

For a 1:2 frequency ratio system, the read data bus is divided into 2 DFI read data words. For a 1:4 frequency ratio system, the read data bus is divided into 4 DFI read data words. Each DFI data word transfers a memory data word, the data associated with one rising and falling DQS. For example, in a 1:4 system with a memory data width of 32 bits, the DFI read data bus would consist of 4 64-bit DFI data words.

On a DFI clock, the PHY is permitted to assert any number of consecutive **dfi_rddata_valid_wN** signals that correspond to valid read data. However, the read data must be returned in a rolling order of DFI data words. For a 1:4 frequency ratio system, if read data is returned on the **dfi_rddata_w0** and **dfi_rddata_w1** buses on one DFI clock cycle, the next transaction must return data starting on the **dfi_rddata_w2** bus, regardless of the number of DFI data words being returned. If that next transaction returned 2 DFI data words, data must be returned on the **dfi_rddata_w2** and **dfi_rddata_w3** buses. If that next transaction returned 4 DFI data words, data must be returned on the **dfi_rddata_w2**, **dfi_rddata_w3**, **dfi_rddata_w0** and **dfi_rddata_w1** buses - in that order.

For a 1:2 frequency ratio system, read data must be returned in the same manner, in a rolling order of DFI data words. In this case, there are only 2 DFI data words in the DFI read data bus - **dfi_rddata_w0** and **dfi_rddata_w1**.

The rolling order rule must be followed regardless of whether the subsequent data transfer occurs on the next DFI clock or several clocks later. For LPDDR2, LPDDR3 and LPDDR4 memory systems, the rolling order rule applies to both reads and mode register reads. The order is critical for the PHY and MC to correctly communicate read data. Each DFI data word must be used prior to sending data on the subsequent DFI data word, requiring data to be sent contiguously. The subsequent read data must be returned on the next DFI data word relative to the previous transaction. If the last transaction ended on **dfi_rddata_w2**, for example, the next transfer must begin on **dfi_rddata_w3**. Similarly, it is not legal to return read data on only the **dfi_rddata_w0** and **dfi_rddata_w2** buses.

Both the MC and the PHY must track which signals were used in the last transfer in order to interpret the data accurately. At initialization, the DFI data word pointer is set to 0, and the first read data returned is expected on the **dfi_rddata_w0** bus. During normal operation, certain procedures may affect the read data rotation, such as frequency changing, data training or gate training. Therefore, any assertion of the **dfi_init_start** signal, or a de-assertion of any bit (or bits) of the **dfi_rdlvl_en** or **dfi_rdlvl_gate_en** signals must trigger a re-initialization of the DFI data word pointer to 0.

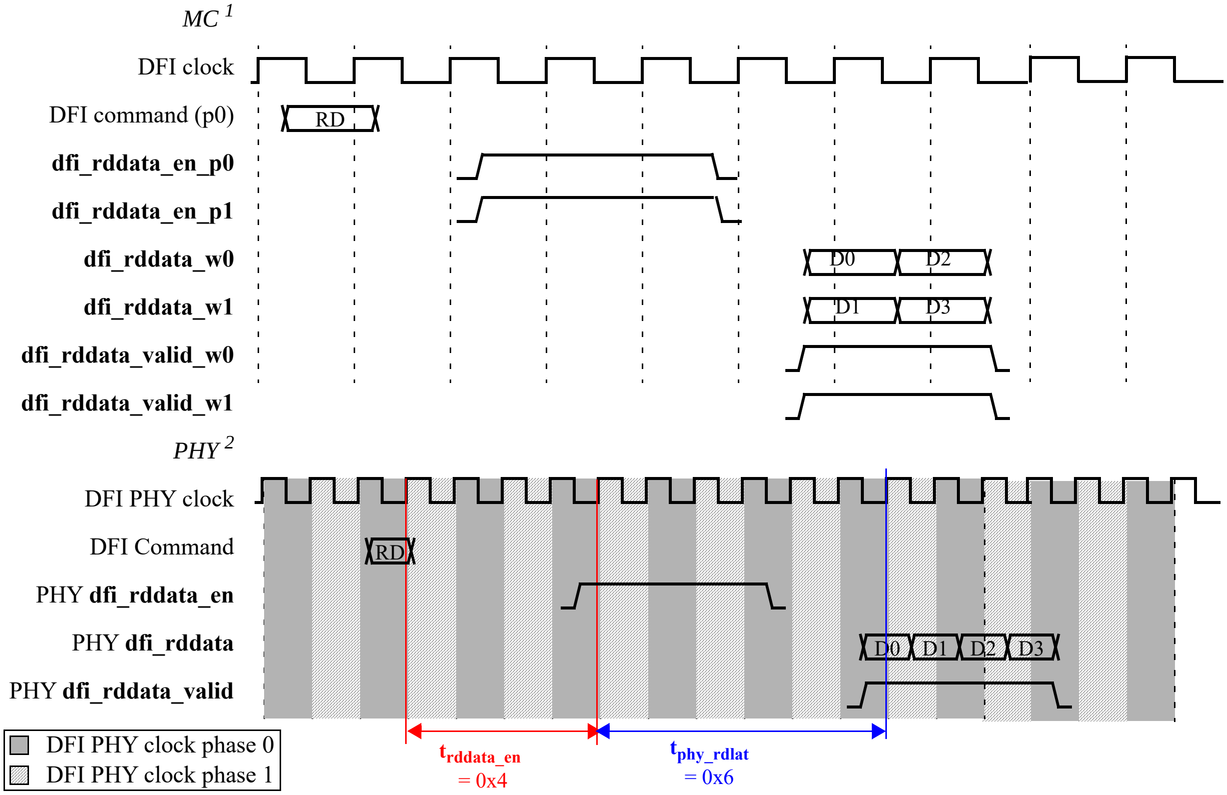
The rotational use of the **dfi_rddata_valid_wN** signals is only required in situations where the system may return less data than the DFI read data bus. If the minimum transfer size is a multiple of the DFI read data bus width, data can always be returned on all DFI data words and the **dfi_rddata_valid_wN** signals are all driven identically. Otherwise, only certain DFI data words of the DFI read data bus are used. In either case, the MC must be able to receive data in a rotating order based on the last transfer to be DFI compliant for frequency ratio. A PHY may optionally be implemented such that it always returns read data on the entire DFI read data bus per transaction.

Regardless of how the signals are vectored, the PHY may only change read data, read data valid and read data not valid signals during phase 0 of the DFI PHY clock to allow the MC the entire DFI clock period to capture the signal and read data.

The timing parameters $\mathbf{t_{rddata\_en}}$ and $\mathbf{t_{phy\_rdlat}}$ apply in frequency ratio systems in the same way as in matched frequency systems. These timing parameters define the delay from the read command to the **dfi_rddata_en_pN** signal, and from the **dfi_rddata_en_pN** signal to when data is returned on the **dfi_rddata_wN** bus, respectively. These timing parameters are defined in terms of DFI PHY clocks and are measured relative to how the PHY interprets the data.

Figure 46, “1:2 Frequency Ratio Single Read Data Example with Even Read Data to Enable Timing” demonstrates how a vectored **dfi_rddata_en_pN** signal is interpreted by the PHY in a 1:2 frequency ratio system with an even value for the $\mathbf{t_{rddata\_en}}$ timing parameter, and where all DFI data words are returned on a DFI clock cycle.

FIGURE 46. *1:2 Frequency Ratio Single Read Data Example with Even Read Data to Enable Timing*



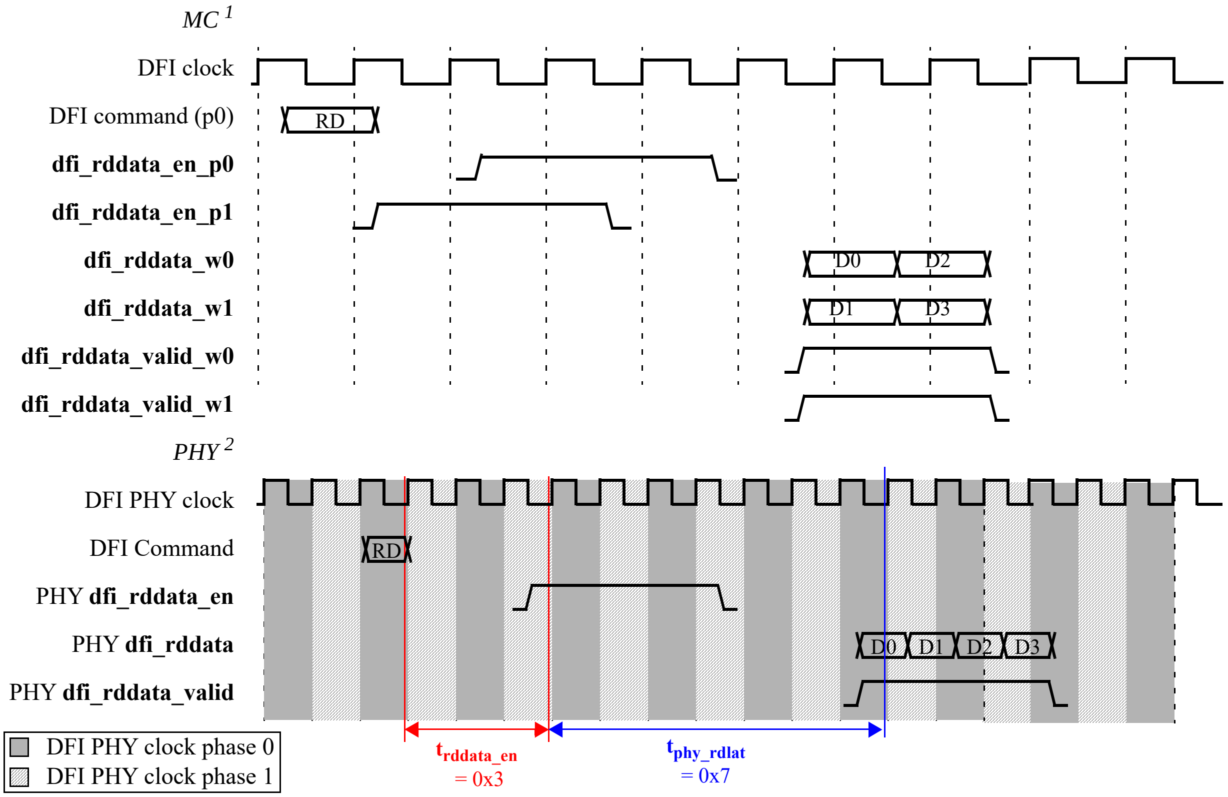
NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

Figure 47, “1:2 Frequency Ratio Single Read Data Example with Odd Read Data to Enable Timing” demonstrates how a vectored **dfi_rddata_en_pN** signal is interpreted by the PHY in a 1:2 frequency ratio system with an odd value for the $\mathbf{t_{rddata\_en}}$ timing parameter, and where all DFI data words are returned on a DFI clock cycle.

FIGURE 47. *1:2 Frequency Ratio Single Read Data Example with Odd Read Data to Enable Timing*

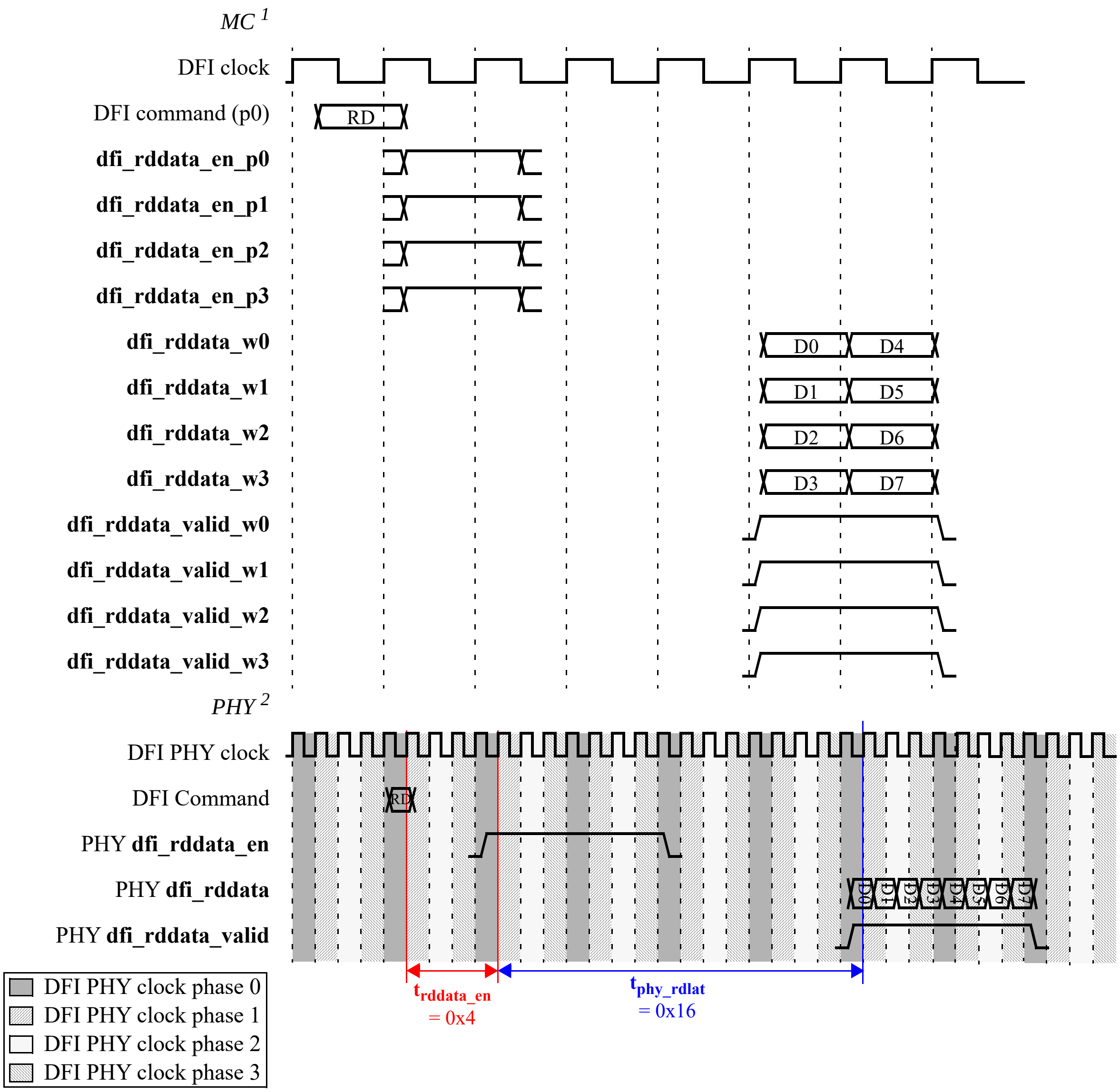


NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

Figure 48, “1:4 Frequency Ratio Single Read Data Example” demonstrates how a vectored **dfi_rddata_en_pN** signal is interpreted by the PHY in a 1:4 frequency ratio system where all DFI data words are being returned on a DFI clock cycle.

FIGURE 48. *1:4 Frequency Ratio Single Read Data Example*



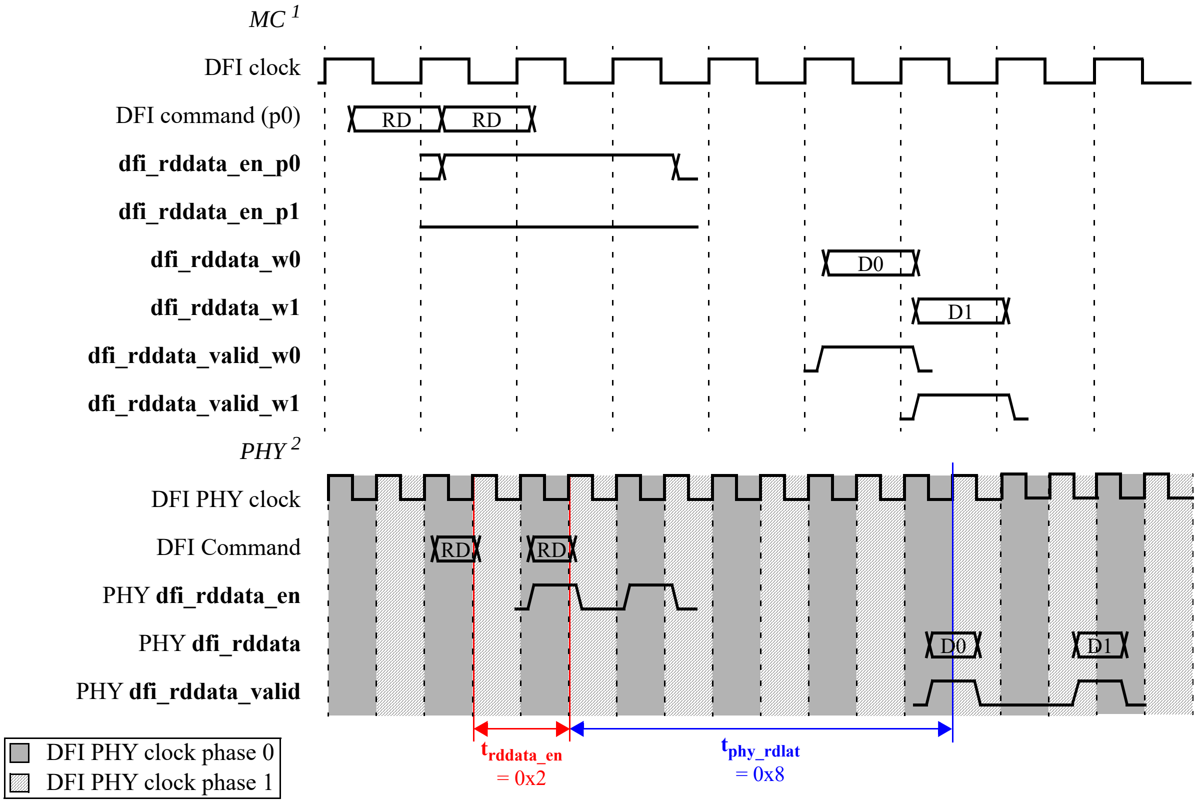
NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

Figure 49, “1:2 Frequency Ratio Multiple Read Data Example” returns a single DFI data word with each command. The data for the second read command is returned on the **dfi_rddata_w1** bus following the rotational order rule.

FIGURE 49. *1:2 Frequency Ratio Multiple Read Data Example*



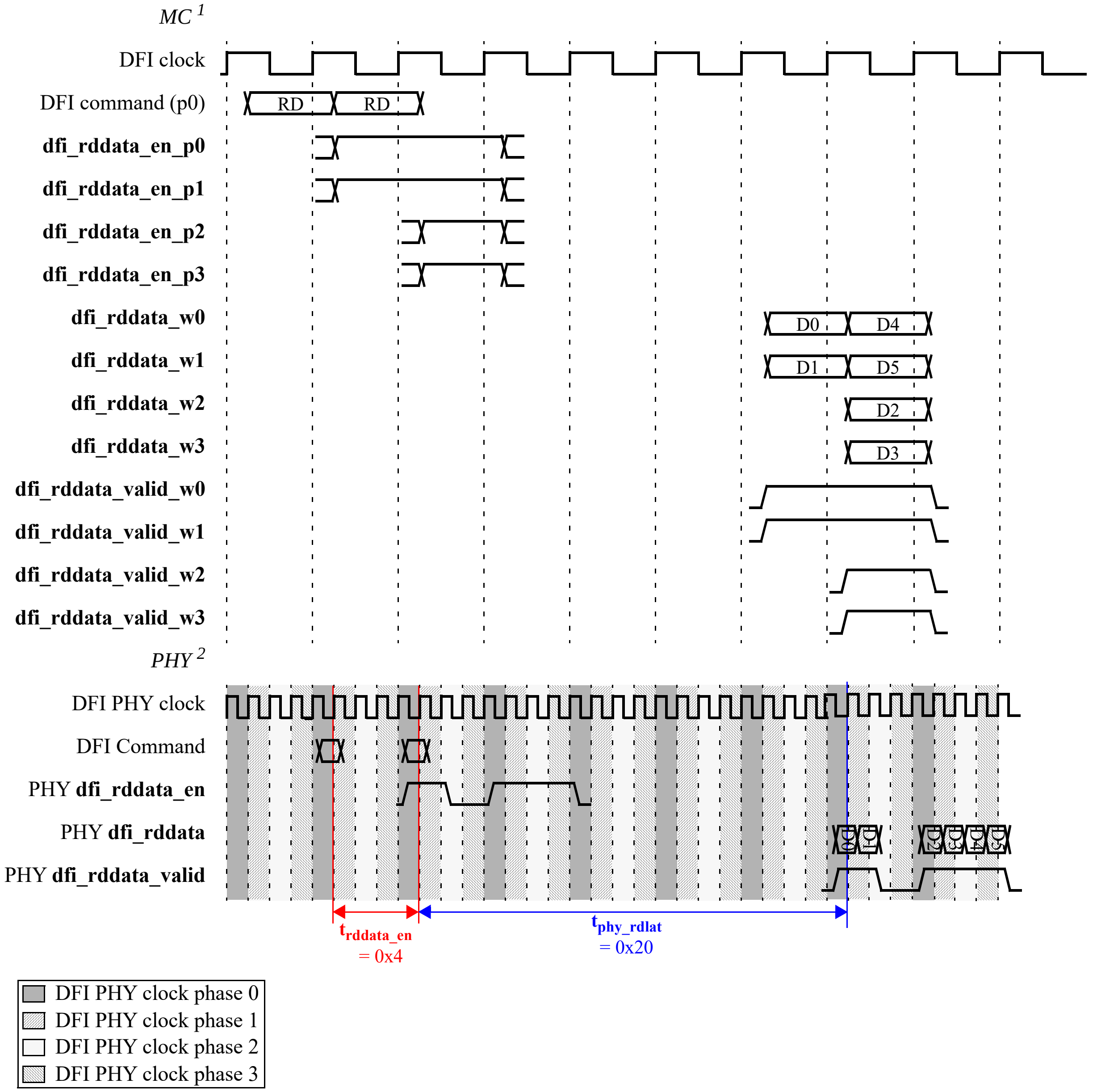
NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

Similar to Figure 49, “1:2 Frequency Ratio Multiple Read Data Example”, Figure 50, “1:4 Frequency Ratio Multiple Read Data Example” shows a burst length 4 followed by a burst length 8 read. The data for the burst length 8 read command is returned starting on the **dfi_rddata_w2** bus following the rotational order rule.

FIGURE 50. *1:4 Frequency Ratio Multiple Read Data Example*



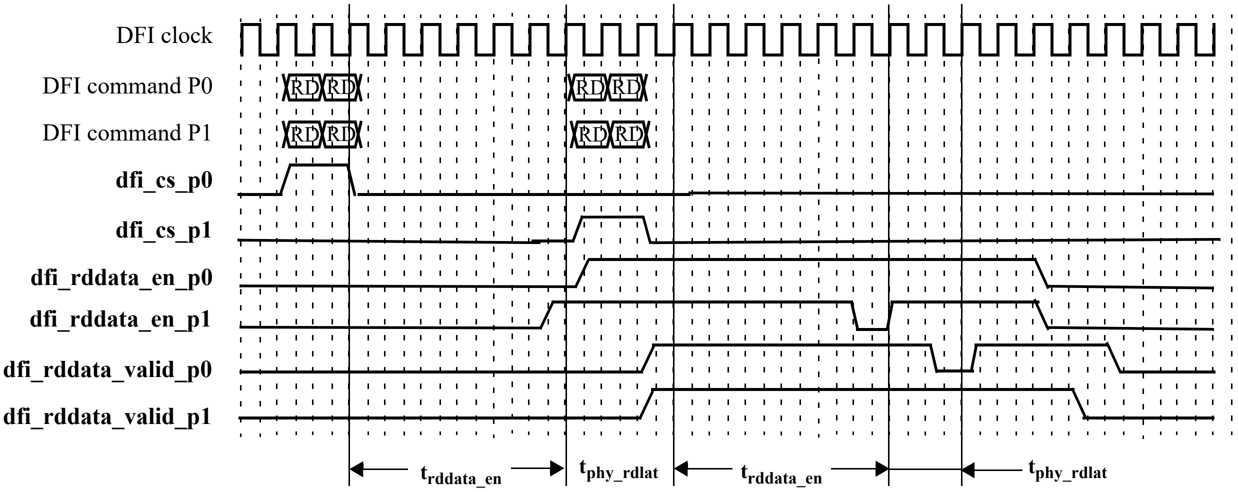
NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

DFI timing parameters for read and write commands will be referenced from the second tick of the second command in the same manner that JEDEC LPDDR4 references RL and WL.

FIGURE 51. *LPDDR4 Read Command, 1:2 Frequency Ratio*



4.9.4.1 DFI Read Data Rotation

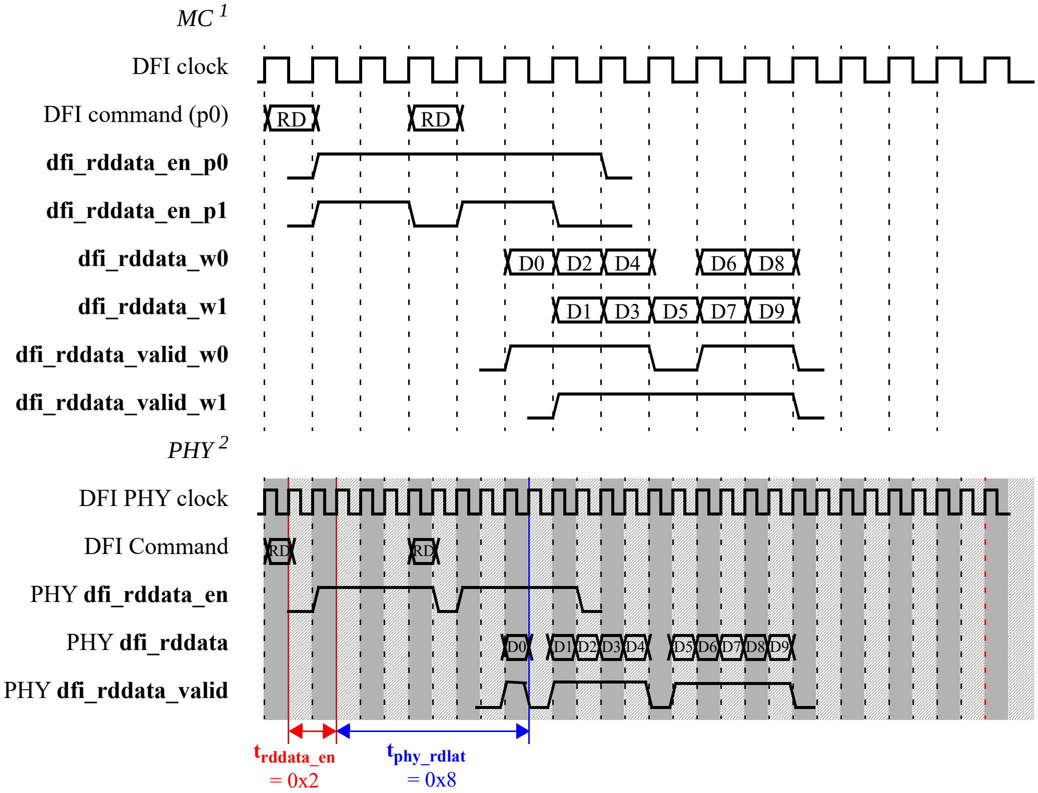
For simplicity, the following four diagrams illustrate different potential timing scenarios with a 1:2 frequency ratio system, transmitting 2 bursts of data, with each burst extended by one DRAM clock cycle.

Many DFI systems do not require support for burst transfers that are not a multiple of the frequency ratio. However, if a system does support these burst transfers, the PHY must transfer the read data in a rotating order as follows:

1. Figure 52, “DFI Read Data Bus for Two 10UI Bursts, each starting in Phase 0”
2. Figure 53, “DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 0 (trddata_en = 2, tphy_rdlat = 8)”
3. Figure 54, “DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 1”
4. Figure 55, “DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 0 (trddata_en = 2, tphy_rdlat = 10)”

Additionally, the MC must be able to capture this data correctly and correlate it to the proper read command.

FIGURE 52. *DFI Read Data Bus for Two 10UI Bursts, each starting in Phase 0*

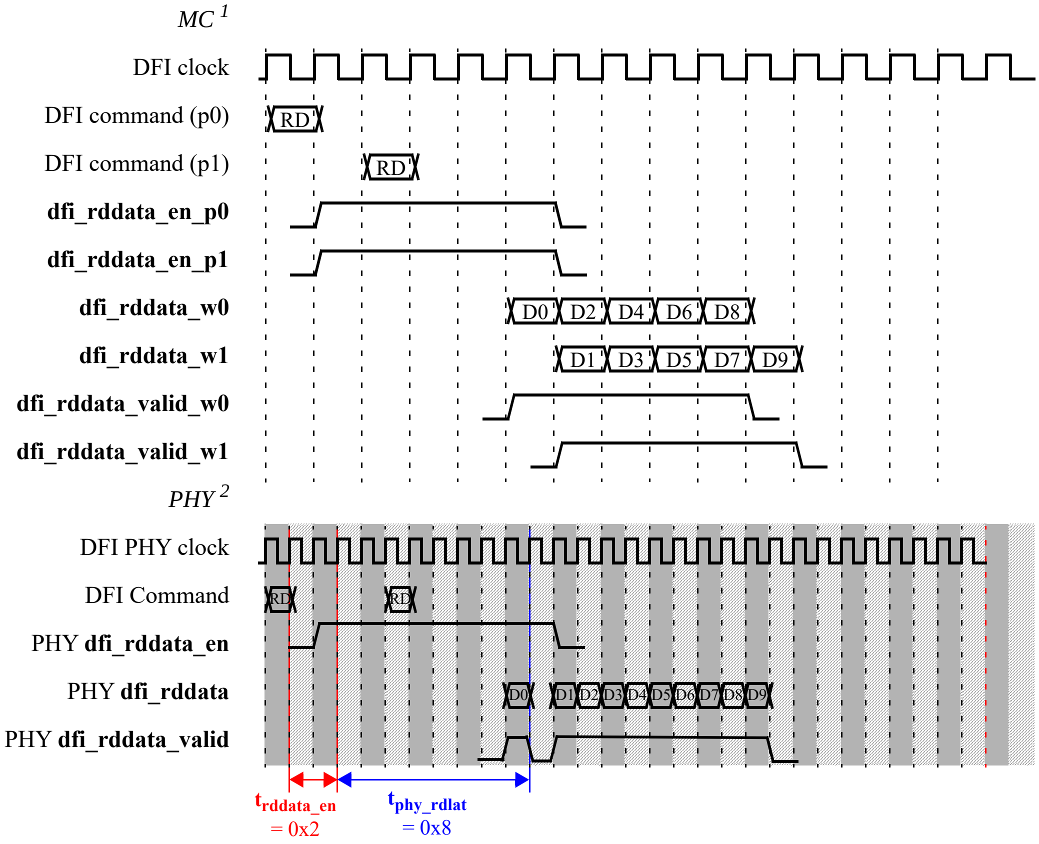


NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain. In this figure, the signals are shown changing on the positive clock edge.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

FIGURE 53. *DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 0 (t_{rddata_en} = 2, t_{phy_rdlat} = 8)*

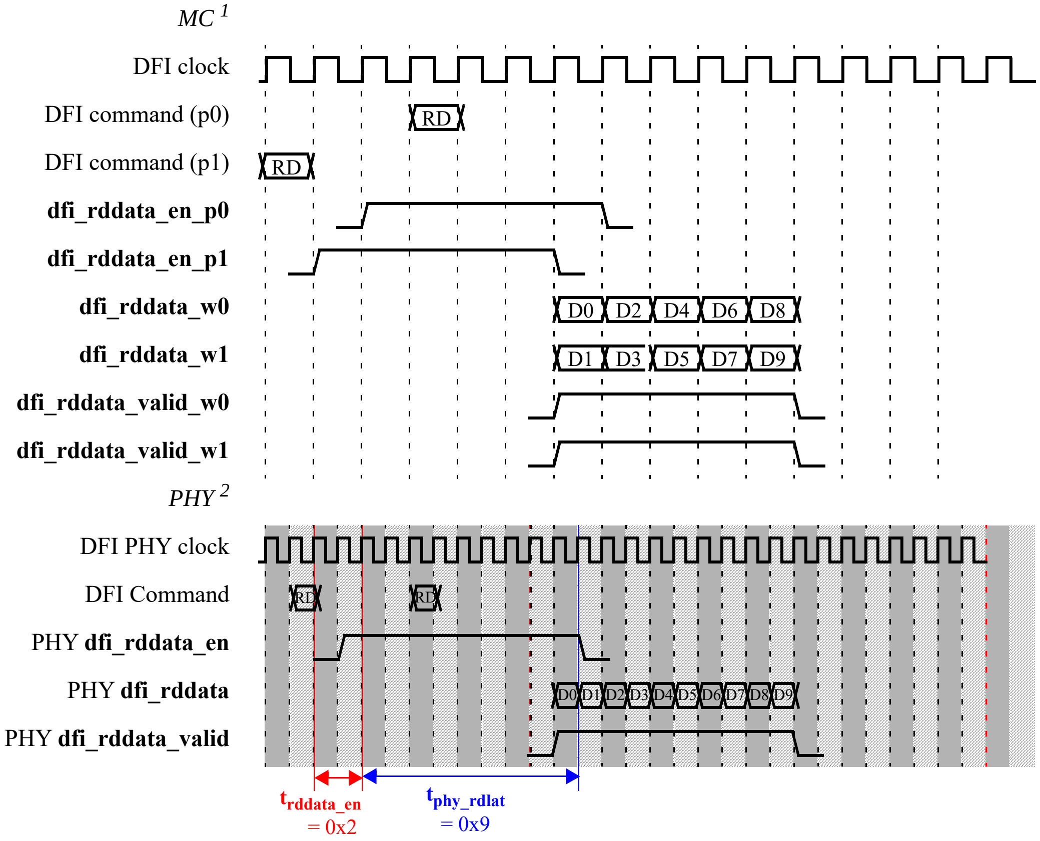


NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain. In this figure, the signals are shown changing on the positive clock edge.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

FIGURE 54. *DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 1*

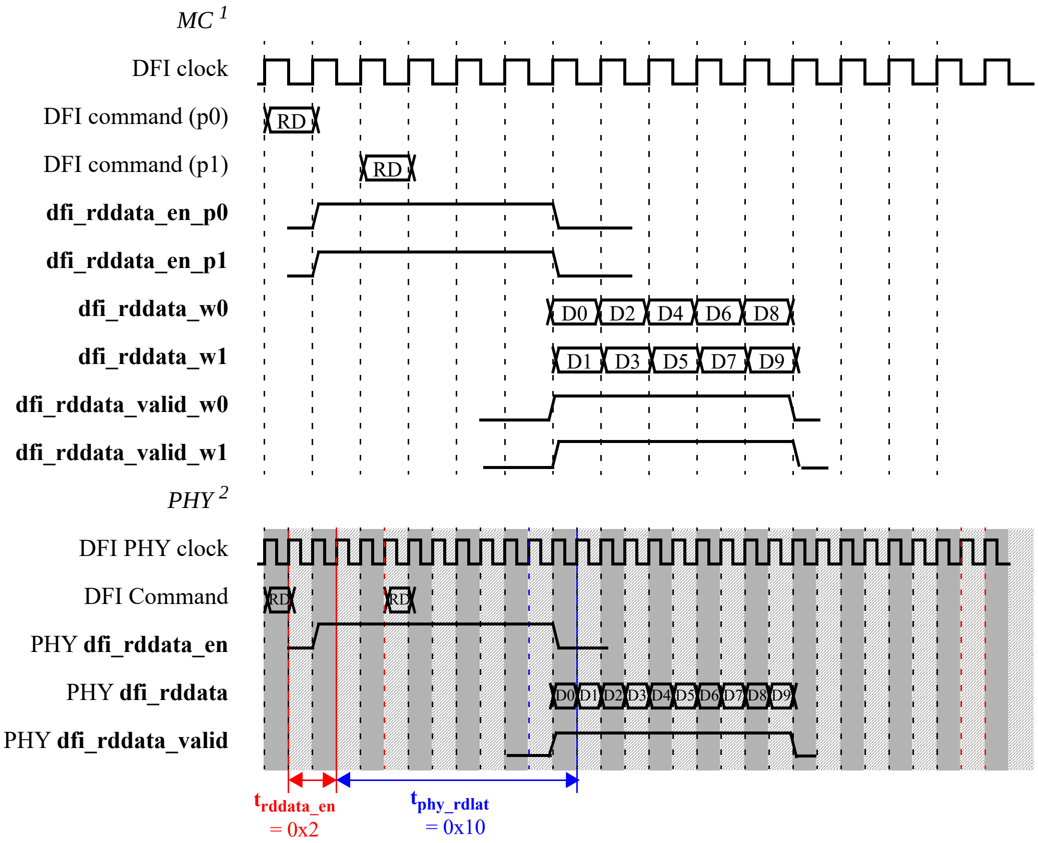


NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain. In this figure, the signals are shown changing on the positive clock edge.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

FIGURE 55. *DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 0 (t_{rddata_en} = 2, t_{phy_rdlat} = 10)*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain. In this figure, the signals are shown changing on the positive clock edge.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

4.9.4.2 Read Data Resynchronization

When the read data rotation order is applicable, the MC and the PHY must maintain read data synchronization to properly interpret the read data order. If a condition occurs where the read data synchronization may be lost, a mechanism is necessary to resynchronize the devices. Conditions may include training or reporting an error. In order to be able to resynchronize when either the **dfi_ctrlupd_req** or **dfi_phyupd_ack** signals are asserted, reset the read data in a Frequency Ratio implementation in both the MC and PHY. Following the assertion of either the **dfi_ctrlupd_req** or **dfi_phyupd_ack** signal, the next read data word is always sent on **dfi_rddata_w0** regardless of the location of the previous data word.

4.10 Frequency Change

The DFI specification defines a frequency change protocol between the MC and the PHY to allow the devices to change the clock frequency of the memory controller and PHY without completely re-setting the system. The memory specifications define various memory states in which the clock frequency can be changed safely. The general procedure is to put the memory in one of these states, modify the clock frequency and re-synchronize the system. When the new clock frequency has been established, the PHY may need to re-initialize various circuits to the new clock frequency prior to resuming normal memory operation. Once complete, the memory system is ready to resume normal operation.

Frequency Change is an optional feature. The system may use a non-DFI frequency change method, or may choose to not support frequency change. However, if both the MC and the PHY intend to use the frequency change protocol, the MC and PHY must comply with the handshake defined by the DFI specification. The handshaking protocol defines the signals through which the MC and the PHY allow a frequency change to occur and also provides a means to abort the process if the PHY does not respond to a frequency change request. When a frequency change occurs, some of the DFI timing parameters may need to be changed.

NOTE: During the frequency change, the DFI clock must remain valid - either operating at a valid frequency or gated high or low.

The signals used in the frequency change protocol are **dfi_init_start**, asserted during initialization and in normal operation, **dfi_init_complete** and **dfi_frequency**. For more information on these signals, refer to Section 3.5, “Status Interface”.

4.10.1 Frequency Change Protocol - Acknowledged

During normal operation, once the **dfi_init_start** and **dfi_init_complete** signals have been asserted, the system may change the DFI clock frequency. The MC asserts the **dfi_init_start** signal to indicate that a clock frequency change is being requested and drives the encoded value of the new frequency on the **dfi_frequency** signal. The PHY should not interpret the initial **dfi_init_start** assertion as a frequency change request. When the **dfi_init_start** signal is asserted, the MC and the PHY must reset their DFI read data word pointers to 0.

The MC guarantees that the **dfi_init_start** signal remains asserted for at least $\mathbf{t_{init\_start}}$ cycles, allowing the PHY time to respond. The PHY may respond or ignore the frequency change request. To acknowledge the request, the **dfi_init_complete** signal must be de-asserted within $\mathbf{t_{init\_start}}$ cycles of the assertion of the **dfi_init_start** signal, or the opportunity for frequency change is withdrawn until the MC re-asserts this signal. The **dfi_init_complete** signal must de-assert at least one cycle before $\mathbf{t_{init\_start}}$ expires.

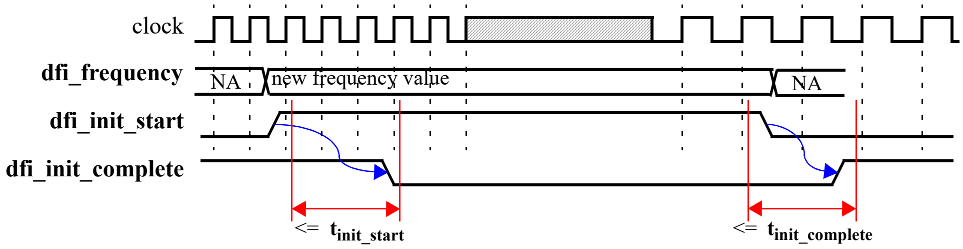
If the frequency change is acknowledged, the MC must hold the **dfi_init_start** signal asserted as long as the frequency change continues. As long as the **dfi_init_start** signal is asserted, the MC can not change the **dfi_frequency** signal. Once the frequency change has completed, the MC de-asserts the **dfi_init_start** signal. The PHY must then complete any re-initialization required for the new clock frequency and re-assert **dfi_init_complete** within $\mathbf{t_{init\_complete}}$ cycles.

During a frequency change operation, the PHY must ensure that the memory interface is maintained at valid and stable levels throughout the operation to ensure that memory protocol is being observed. The MC must also insure that it maintains valid and stable levels on the DFI while **dfi_init_start** is asserted or **dfi_init_complete** is de-asserted.

Note that no maximum number of cycles for the entire cycle to complete is specified by the DFI.

Figure 56, “Frequency Change Acknowledge Timing Diagram” shows the MC holding the **dfi_init_start** signal asserted as long as the frequency change continues and the PHY re-asserting **dfi_init_complete** within $\mathbf{t_{init\_complete}}$ cycles.

FIGURE 56. *Frequency Change Acknowledge Timing Diagram*

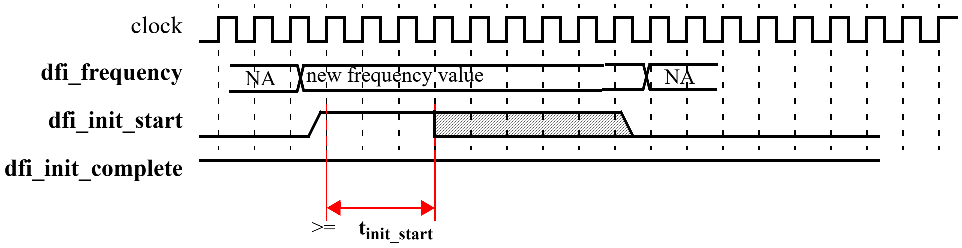


4.10.2 *Frequency Change Protocol - Not Acknowledged*

The MC must assert the **dfi_init_start** signal for at least $\mathbf{t_{init\_start}}$ cycles. It is important to note that the PHY is not required to respond to a frequency change request.

Figure 57, “Frequency Change Request Ignored Timing Diagram” shows the MC asserting the **dfi_init_start** signal for $\mathbf{t_{init\_start}}$ cycles.

FIGURE 57. *Frequency Change Request Ignored Timing Diagram*



4.11 CA Parity Signaling and CA Parity, CRC Errors

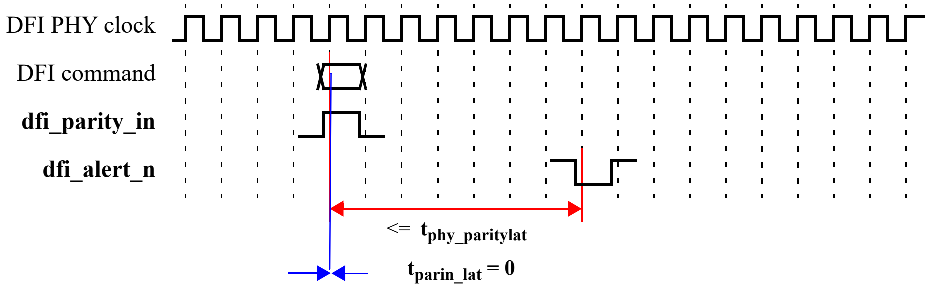
Parity bits are used in command transmission for verifying that the command has been transmitted correctly between master and slave. A single parity bit is sent with each command and identifies if the number of bits set high in the **dfi_address**, **dfi_act_n**, **dfi_bg**, **dfi_bank**, **dfi_cas_n**, **dfi_ras_n** and **dfi_we_n** signals is an even or an odd number. If the DRAM receives a command that the number of bits of these signals that is set to ’b1 does not match the even/odd setting of the **dfi_parity_in** signal, an error occurred during transmission.

The MC sends the **dfi_parity_in** signal, which is valid for one cycle per command. The PHY must delay the **dfi_parity_in** signal identically to the command bus. Consequently, the PHY sends the **dfi_parity_in** signal, along with the command and other DFI signals, to the DRAM interface $\mathbf{t_{ctrl\_delay}}$ cycles after it receives the MC **dfi_parity_in** signal.

The **dfi_parity_in** signal is sent $\mathbf{t_{parin\_lat}}$ cycles after the DFI command; if $\mathbf{t_{parin\_lat}} = 0$, **dfi_parity_in** and the DFI command are sent in the same cycle. The memory systems compute parity on the incoming command and compare the computed value with the value driven on the DFI parity signal. If these values do no match, the memory systems assert a parity error output, which is sent back to the PHY. The **dfi_alert_n** signal is asserted within $\mathbf{t_{phy\_paritylat}}$ cycles of the associated **dfi_parity_in** signal. Since $\mathbf{t_{phy\_paritylat}}$ is a maximum value, the **dfi_alert_n** signal can not be correlated to one specific command, but to any command sent within the last $\mathbf{t_{phy\_paritylat}}$ cycles.

Figure 58, "Odd Command Parity Error Example Timing Diagram" shows the timing between the command and error with an odd command parity example.

FIGURE 58. *Odd Command Parity Error Example Timing Diagram*



4.11.1 CA Parity Timing

The DFI protocol supports CA parity utilizing the **dfi_parity_in** signal to transmit the parity data across the DFI. The MC generates the **dfi_parity_in** signal and sends it across the DFI., The PHY transmits the signal to the memory subsystem, incorporating the same $\mathbf{t_{ctrl\_delay}}$ as the command bus.

In frequency ratio systems, the **dfi_parity_in** signal is defined per phase, similar to the CA bus. For example, in a 1:2 frequency ratio system, the parity data is sent on **dfi_parity_in_p0** and **dfi_parity_in_p1**.

To generalize the parity support for current and future parity implementations, the $\mathbf{t_{parin\_lat}}$ parameter, defined in Table 15, "Status Interface Signals", specifies the timing of the **dfi_parity_in_pN** signal relative to the DFI command. The $\mathbf{t_{parin\_lat}}$ parameter, allows the parity data to be sent coincident with the command or on a subsequent cycle.

4.11.2 CA Parity and CRC Errors

The DRAM generates both CA parity and CRC errors. The error signals are transferred across the DFI bus to the MC. In systems that require CA parity or CRC support, the MC and PHY must both support the **dfi_alert_n** error signal.

The active low **dfi_alert_n** signal transmits both CA parity and write CRC errors. The PHY is not required to distinguish between a CA parity and a CRC error, instead, the PHY transmits the error to the MC for evaluation. The PHY holds the current state of the **dfi_alert_n** signal until the PHY error input transitions to a new value. Consequently, the pulse width of **dfi_alert_n** matches the pulse width of the DRAM subsystem error signal, plus or minus any synchronization cycles.

4.11.3 CA Parity and CRC Errors in Frequency Ratio Systems

With frequency ratio systems, multiple **dfi_alert_n** outputs from the PHY are required to accurately transfer the error pulse width.

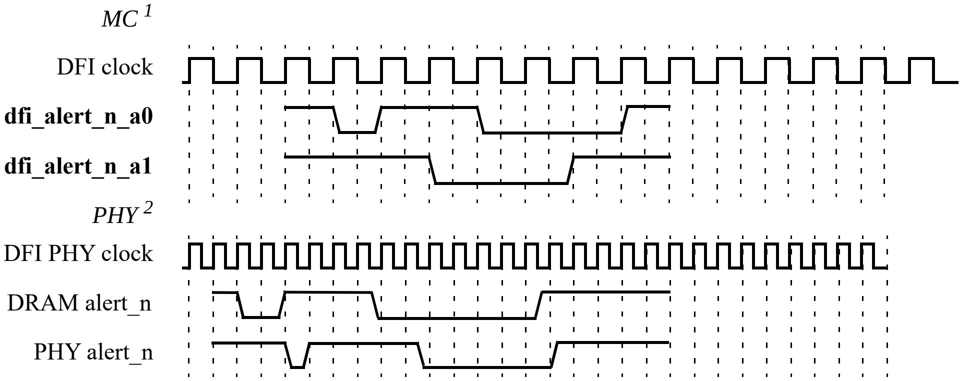
Table 40, “dfi_alert_n Signal With Matched and Frequency Ratio Systems” defines the outputs for matched frequency, 1:2 frequency and 1:4 frequency systems. In all cases, the PHY must synchronize the **dfi_alert_n** outputs to the DFI clock domain. During synchronization, the width of the DRAM error signal may change by a few cycles, and these small changes in the error pulse width are acceptable.

TABLE 40. ***dfi_alert_n*** *Signal With Matched and Frequency Ratio Systems*

System	Alert Outputs	Description
Matched Frequency	**dfi_alert_n**	Single Output Width of output indicates width of error pulse +/- synchronization errors.
1:2 Frequency	**dfi_alert_n_a0** **dfi_alert_n_a1**	2 outputs, one for each phase of the DFI clock. Combination of signals is used for determining error pulse width. For example, **dfi_alert_n_a0** and **dfi_alert_n_a1** are both asserted for one DFI clock period. The error signal width is 2 DRAM clocks +/- synchronization errors.
1:4 Frequency	**dfi_alert_n_a0** **dfi_alert_n_a1** **dfi_alert_n_a2** **dfi_alert_n_a3**	4 outputs, one for each phase of the DFI clock. Combination of signals is used for determining error pulse width. For example, **dfi_alert_n_a0**, **dfi_alert_n_a1**, **dfi_alert_n_a2** and **dfi_alert_n_a3** are all asserted for one DFI clock period. The error signal width is 4 DRAM clocks +/- synchronization errors. As another example, only **dfi_alert_n_a1** and **dfi_alert_n_a2** are asserted for one DFI clock period. The error signal width is 2 DRAM clocks +/- synchronization errors.

In Figure 59, “dfi_alert_n_aN with 1:2 Frequency Ratio”, the 1st assertion of the DRAM error signal (DRAM **alert_n**) is held for 2 DRAM clocks, but due to synchronization, only **dfi_alert_n_a0** is asserted on the DFI bus, indicating a single DRAM clock period pulse. The 2nd assertion of the DRAM error signal is held for multiple cycles and the DFI output indicates this functionality by asserting both **dfi_alert_n_a0** and **dfi_alert_n_a1** as required. The **dfi_alert_n_a0** and **dfi_alert_n_a1** signals correlate to the phase of the DFI clock. However, the assertion of **dfi_alert_n_a0** does not directly relate to the phase of assertion on the DRAM bus and therefore the assertion of **dfi_alert_n_a0** is not directly related to the phase of the write data.

FIGURE 59. ***dfi_alert_n_aN*** *with 1:2 Frequency Ratio*



NOTE: This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.

1. In the MC clock diagram, the timing shows the DFI bus signaling.

2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

4.12 DFI Training Operations

DFI read and write training operations can increase accuracy of signal placement at higher speeds in DDR3, DDR4, LPDDR2, LPDDR3 and LPDDR4 systems. CA training optimizes the CA bus setup and hold times relative to the memory clock.

The DFI has five training operations to support CA training, read training, write DQ training, write leveling. “Read training” collectively refers to two operations - gate training and read data eye training. “CA training” refers to CA data eye training for LPDDR3 memories and to both CA data eye and VREF value training for LPDDR4 memories. For more information on the training operations and to identify operations that correspond to specific devices, refer to Section 3.6, “DFI Training Interface”.

For DFI compliance, the MC and the PHY may support either “ PHY Independent Mode” = or “DFI Training Mode” for the applicable training operations and the mode is set per operation.

Support for the defined state of each programmable parameter must be defined by each device and may be implemented as programmable registers within the device. The parameter is defined as a single enable bit indicating whether or not the corresponding DFI training operation (CA training, gate training, read data eye training, write DQ training or write leveling) is enabled.

4.12.1 Training Operations in DFI Training Mode

In DFI training mode, the MC sets up the DRAM for the training operation and periodically issues read commands, write bursts, write strobes or calibration commands as applicable. The PHY is responsible for determining the correct delay programming for each operation. The PHY evaluates the data returned from the commands, adjusts the delays and evaluates the results to locate the appropriate edges. The MC assists by enabling and disabling the leveling logic in the DRAMs and the PHY and by generating the necessary read commands, mode register reads, write bursts or write strobes.

The PHY informs the MC when it has completed training, which triggers the MC to stop generating commands and to return to normal operation.

The MC must complete all transactions in progress to memory prior to initiating any of the training operations. Once any of the enable signals are asserted, the PHY should immediately enable the associated logic. During training, the MC does not use the memory response from the PHY. Therefore the only relevant DFI timing parameters are:

- CA training: $\mathbf{t_{calvl\_cc}}$ (minimum delay between calibration commands)
- Read training: $\mathbf{t_{rdlvl\_rr}}$ (minimum delay that the MC should wait between issuing reads for DDR3 and DDR4 memory systems or mode register reads for LPDDR2, LPDDR3 and LPDDR4 memory systems)
- Write DQ training: $\mathbf{t_{wdqlvl\_rw}}$ (minimum delay from the last read in a calibration sequence to the first write in the next calibration sequence) and $\mathbf{t_{wdqlvl\_ww}}$ (minimum delay between write commands)
- Write leveling: $\mathbf{t_{wrlvl\_ww}}$ (minimum delay between write strobes)

The MC continues to drive subsequent CA training calibration commands every $\mathbf{t_{calvl\_cc}}$ cycles, subsequent read leveling commands every $\mathbf{t_{rdlvl\_rr}}$ cycles, subsequent write DQS training/leveling commands every $\mathbf{t_{wdqlvl\_ww}}$ cycles, or subsequent write level strobes every $\mathbf{t_{wrlvl\_ww}}$ until the PHY drives all bits of the response signal high.

Multiple training sequences can be defined, with the sequence information supplied by either the MC or the PHY. The MC must provide flexibility to run training sequences required by the PHY and should not dictate a set of sequences. The PHY determines the sequences necessary to accomplish training. Both the MC and the PHY need to be aware of the defined training sequences and the specific sequence required for each training operation.

Whether the MC or the PHY initiates the training sequence, the MC asserts the enable signal to initiate or accept the training sequence, and the MC holds the enable signal asserted until the current training operation completes. The DFI training requires the MC to support the training sequences to the PHY by generating MRW commands, toggling the enable parameter, generating the appropriate strobe signals and evaluating the response. The PHY is responsible for adjusting the DLL delays and evaluating the responses from memory. When the PHY is satisfied with the training sequence, a completion response is sent back to the MC.

For DDR4, training sequence accuracy is increased with a selection of encodings. The MC determines a sequence of up to four unique, non-default Multi-Purpose Register (MPR) values and formats and drives the **dfi_lvl_pattern** signal to communicate the sequence to the PHY to indicate the training pattern that is currently active.

4.12.2 Training Operations in PHY Independent Mode

In PHY Independent mode, the PHY supports training independent of the DFI signaling. In this case, the associated training interface is not used, and the MC should be capable of generating the required MRS commands to enter or exit the test modes of the DRAMs without the training interface signals.

4.12.3 Initiating a Training Operation

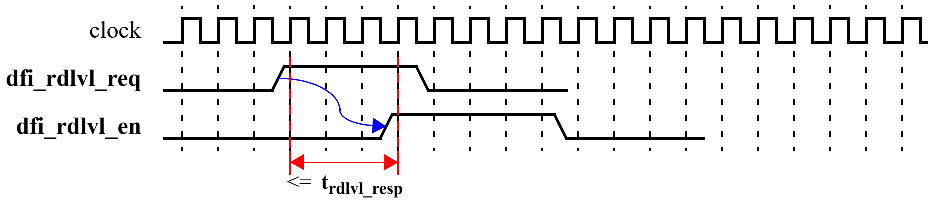
The MC or the PHY may initiate any training operation, depending on which training modes are supported. Training may be executed during initialization, frequency change or during normal operation. In DFI training mode, the MC is responsible for initiating gate training, read data eye training and/or write leveling required during initialization or frequency change. The PHY should not request training or leveling during initialization; if the PHY requests training during a frequency change, the PHY must assert it prior to asserting **dfi_init_complete**. In PHY independent mode, the PHY is responsible for all training/leveling as applicable.

The PHY can request CA training on an individual slice by setting the appropriate bit on the **dfi_calvl_req** signal, gate training by setting a bit on the **dfi_rdlvl_gate_req** signal, read data eye training by setting a bit on the **dfi_rdlvl_req** signal, write DQ training by setting a bit on the **dfi_wdqlvl_req** signal, or write leveling by setting a bit on the **dfi_wrlvl_req** signal. For read training, if the asserted bit is for any of the slices in the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$ and/or $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$, then all the slices in that group should be leveled.

The MC must respond to any of these requests by asserting the appropriate bit (or bits) of the appropriate enable (**dfi_calvl_en**, **dfi_rdlvl_en**, **dfi_rdlvl_gate_en**, **dfi_wdqlvl_en** or **dfi_wrlvl_en**) within the relevant response time ($\mathbf{t}_{\mathbf{calvl\_resp}}$, $\mathbf{t}_{\mathbf{rdlvl\_resp}}$, $\mathbf{t}_{\mathbf{wdqlvl\_resp}}$ or $\mathbf{t}_{\mathbf{wrlvl\_resp}}$) cycles.

Figure 60, “Read Data Eye Training Request Timing” shows this timing relationship for the data eye training process. The timing is similar for CA training, gate training, write DQ training and write leveling.

FIGURE 60. *Read Data Eye Training Request Timing*



The MC waits for the response signal to be asserted before disabling the active training logic. The response is received on the **dfi_calvl_resp**, **dfi_rdlvl_resp**, **dfi_wdqlvl_resp** or **dfi_wrlvl_resp** signals. The DFI specifies maximum times that the system waits for a response as $\mathbf{t}_{\mathbf{calvl\_max}}$, $\mathbf{t}_{\mathbf{rdlvl\_max}}$, $\mathbf{t}_{\mathbf{wdqlvl\_max}}$ or $\mathbf{t}_{\mathbf{wrlvl\_max}}$.

The target chip select is driven on the **dfi_phy_calvl_cs** signal for CA training, the **dfi_rddata_cs** signal for read training and gate training, the **dfi_phy_wdqlvl_cs** signal for write DQ training and the **dfi_wrdata_cs** signal for write leveling.

An MC supporting the DFI training mode in a system that supports independent training per chip select must support the data path chip selects **dfi_rddata_cs** and **dfi_wrdata_cs** and the training chip select signals **dfi_phy_calvl_cs**, **dfi_phy_rdlvl_cs**, **dfi_phy_rdlvl_gate_cs** and **dfi_phy_wrlvl_cs**. A PHY asserting the DFI training mode request optionally may support any combination of these signals. Any unused input signals should be tied inactive at the MC.

For more information on read leveling signals, refer to Figure 62, “Read Training in DFI Training Mode for DDR3 and DDR4 Memory Systems” and Figure 63, “Read Training in DFI Training Mode for LPDDR2 and LPDDR3 Memory Systems”.

For more information on which signals are required and which signals are optional, refer to Table 3, “DFI Signal Requirements”.

4.12.4 Read Training

“Read training” collectively refers to gate training and read data eye training. From the MC perspective, the gate training and read data eye training operations are handled identically so both operations may be issued utilizing a single interface. For more information on the gate training and data eye training operations, refer to Section 3.3, “Read Data Interface” and Section 3.6, “DFI Training Interface”.

The MC can initiate read training by driving the relevant bit (or bits) of the **dfi_rdlvl_en** signal, which is used for enabling the read training logic in the PHY independently for each slice, as Figure 60, “Read Data Eye Training Request Timing” illustrates. The PHY can also initiate read training on a slice by driving the appropriate bit of the **dfi_rdlvl_gate_req** or **dfi_rdlvl_req**. The MC will need to respond to this read training request signal assertion by asserting appropriate bit (or bits) of the **dfi_rdlvl_en** signal. If multiple slices must be enabled together, the user can specify the grouping in the $\mathbf{mc}_{\mathbf{rdlvl\_slice\_group[n]}}$ and/or $\mathbf{phy}_{\mathbf{rdlvl\_slice\_group[n]}}$ parameter.

During read training, the **dfi_rddata_valid** and **dfi_rddata** signals are ignored. All evaluations and delay changes are handled within the PHY. When the PHY finds the necessary edges and completes the read training, the PHY should drive the corresponding **dfi_rdlvl_resp** signal to ’b01, which informs the MC that the procedure is complete for that slice. The MC should then de-assert the set bit (or bits) of the **dfi_rdlvl_en** signal. When the **dfi_rdlvl_en** signal bit (or bits) de-asserts (de-assert), the PHY should stop driving the **dfi_rdlvl_resp** signal.

4.12.4.1 Gate Training

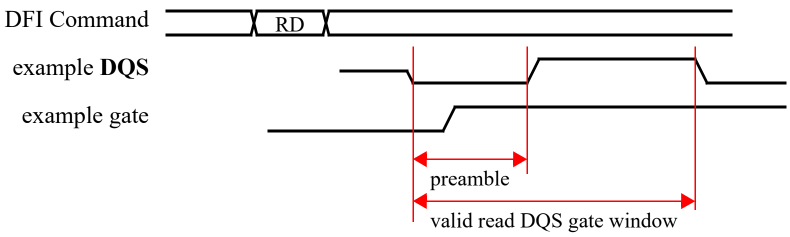
The goal of gate training is to locate the setting at which the initial read DQS rising edge aligns with the rising edge of the read DQS gate. Once this setting is identified, the read DQS gate can be adjusted to the approximate midpoint of the read DQS preamble prior to the DQS.

To indicate proper alignment of the gate to the first read DQS, the gate training operation requires that the read DQS gate be placed within the bounds of the beginning of the read DQS preamble and the falling edge of the first read DQS. Placing the gate within the bounds of the timing window may require another alignment method or may require running gate training iteratively.

The gate training signals are: **dfi_rddata_en**, **dfi_rdlvl_gate_en**, **dfi_rdlvl_gate_req**, **dfi_rdlvl_resp** and the DFI command bus. The corresponding mode parameter is $\mathbf{phy}_{\mathbf{rdlvl\_gate\_en}}$. When the PHY initiates gate training, the PHY can optionally send the **dfi_phy_rdlvl_gate_cs** signal to indicate the targeted chip select. The MC transfers the **dfi_rddata_cs** signal to identify the chip select currently being trained.

Figure 61, “Gate Leading DQS Timing” is an example of gate training.

FIGURE 61. *Gate Leading DQS Timing*



4.12.4.2 Data Eye Training

The goal of data eye training is to identify the delay at which the read DQS rising edge aligns with the beginning and end transitions of the associated DQ data eye. By identifying these delays, the system can calculate the midpoint between the delays and accurately center the read DQS within the DQ data eye.

The read data eye training signals are: **dfi_rddata_en**, **dfi_rdlvl_en**, **dfi_rdlvl_req**, **dfi_rdlvl_resp**, **dfi_lvl_pattern** and the DFI command bus. The corresponding mode parameter is $\mathbf{phy}_{\mathbf{rdlvl\_en}}$. When the PHY initiates data eye training, the

PHY can optionally send the **dfi_phy_rdlvl_cs** signal to indicate the targeted chip select. The MC responds by executing training and transfers **dfi_rddata_cs** to indicate the chip select being trained.

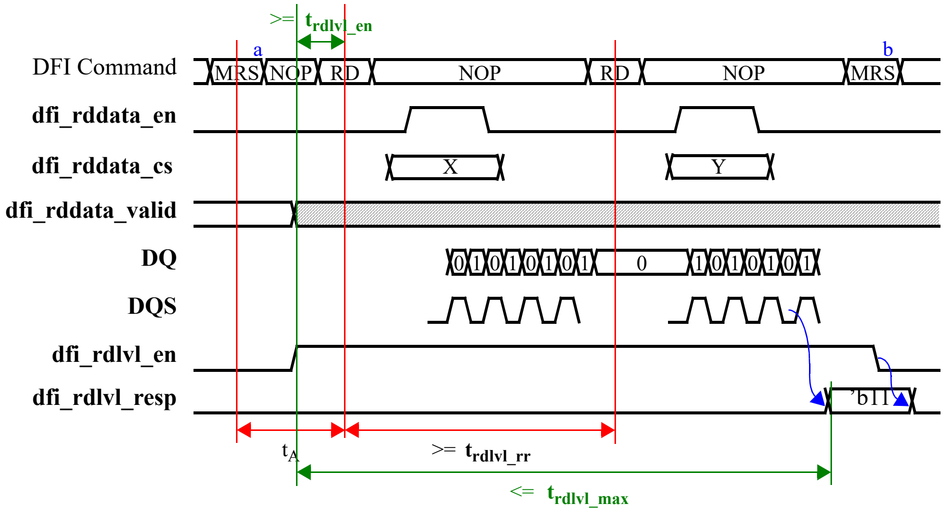
For gate training, when the PHY has found the necessary DQS gate placement, the PHY drives the **dfi_rdlvl_resp** signal to ’b11, which informs the MC that the procedure is done. For read data eye training, when the sequence is complete, the PHY responds with a value of ’b01 or ’b11 on the **dfi_rdlvl_resp** signal. If the PHY returns a ’b01 response, the MC will de-assert the bit (or bits) of the **dfi_rdlvl_en** signal, set up the next sequence, and then re-assert the bit (or bits) of the **dfi_rdlvl_en** signal and repeat the process. If the PHY returns a ’b11 result, the process is complete. All training must also meet the timing defined in the $\mathbf{t_{rdlvl\_max}}$ timing parameter.

The MC then de-asserts the bit (or bits) of the **dfi_rdlvl_en** signal. Once the **dfi_rdlvl_en** signal bit (or bits) has de-asserted (have de-asserted), the PHY may stop driving a value on the **dfi_rdlvl_resp** signal.

For DDR3 and DDR4, MRS commands are used to enable and disable the read training logic in the DRAMs. For LPDDR2, LPDDR3, LPDDR4, no MRS commands are required to enable/disable the leveling mode in the DRAMs.

Figure 62, “Read Training in DFI Training Mode for DDR3 and DDR4 Memory Systems” demonstrates read training in DDR3 and DDR4 memory systems. Figure 63, “Read Training in DFI Training Mode for LPDDR2 and LPDDR3 Memory Systems” shows read training in LPDDR2 and LPDDR3 memory systems.

FIGURE 62. *Read Training in DFI Training Mode for DDR3 and DDR4 Memory Systems*



a = Enables DRAM read training logic.
b = Disables DRAM read training logic.
t_A = Timing delays required by the DDR3 and DDR4 specifications.
X, Y = Target chip select.
NOTE: When **dfi_rddata_en** is asserted, **dfi_rddata_cs** must be set to the target chip select.

FIGURE 63. *Read Training in DFI Training Mode for LPDDR2 and LPDDR3 Memory Systems*

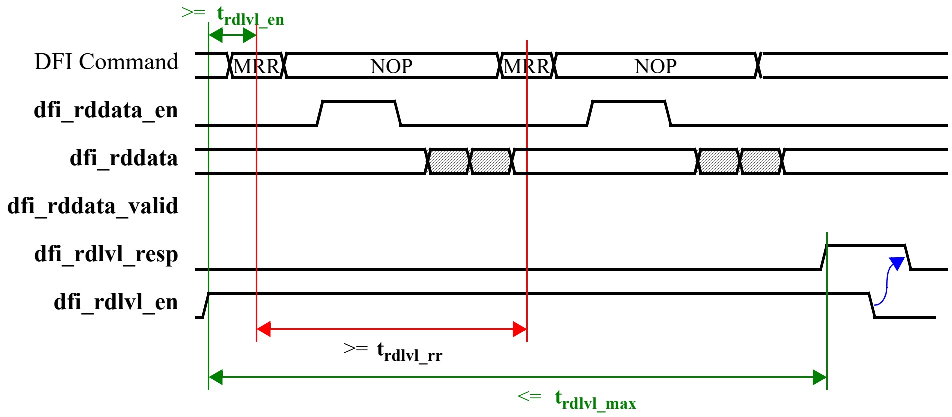


Table 41, “Read Training Testing Results” provides more clarity on the use of the **dfi_rdlvl_resp** and **dfi_rdlvl_done** signals.

TABLE 41. *Read Training Testing Results*

dfi_rdlvl_resp	dfi_rdlvl_done	Result Summary	Value to Program on dfi_lvl_pattern [a]
’b01	’b0	The current test is complete, but the entire operation is not. Re-program the **dfi_lvl_pattern** signal and repeat the test.	Next data value to test. The **dfi_rdlvl_resp** signal provides feedback on how the last two tests fared.
’b01	’b1	The current test is complete, and the MC has sent all data values. Re-program the **dfi_lvl_pattern** signal and repeat the test.	First data value to test
’b11	’b0	The current test is complete, but the entire operation is not. Re-program the **dfi_lvl_pattern** signal and repeat the test.	Next value to test. The **dfi_rdlvl_resp** signal provides feedback on how the last two tests fared.
’b11	’b1	All data patterns have been tested.	None

[a] Please note that the PHY may ignore the **dfi_lvl_pattern** input and drive a different value on the appropriate DQ outputs.

Figure 64, “Read Training in DFI Training Mode for LPDDR4 Memory Systems, Beginning of the Operation” and Figure 65, “Read Training in DFI Training Mode for LPDDR4 Memory Systems, End of the Operation for CS=1; Start of Operation of CS=2” illustrate the new signaling for LPDDR4 memories. Alternatively, LPDDR4 also supports read training using a scratch pad FIFO within the DRAM. For more information on this functionality, refer to Section 4.12.8, “Write DQ Training”.

FIGURE 64. *Read Training in DFI Training Mode for LPDDR4 Memory Systems, Beginning of the Operation*

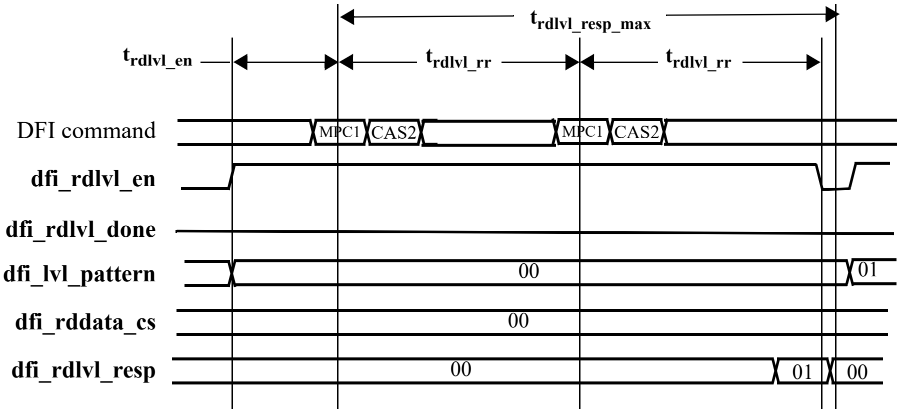
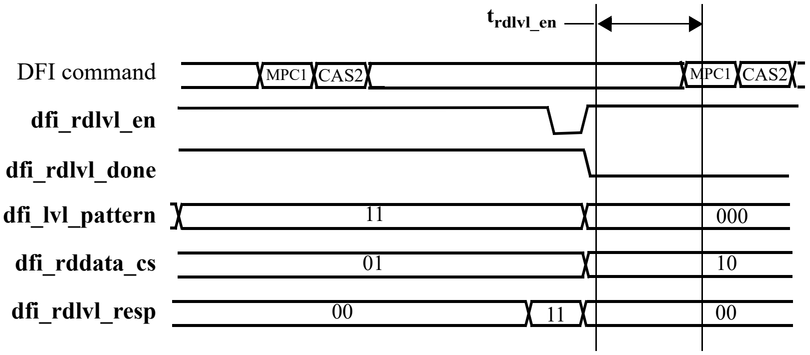


FIGURE 65. *Read Training in DFI Training Mode for LPDDR4 Memory Systems, End of the Operation for CS=1; Start of Operation of CS=2*



4.12.4.2.1 Per-Slice MC-Driven Training Request

To initiate data eye training, the MC will drive the **dfi_rdlvl_en** signal, which is used for enabling the read training logic in the PHY independently for each slice, as shown in Figure 66, “Per-Slice Read Training in DFI Training Mode, MCrdlvl_slice_group = 0x0F”. During data eye training, the **dfi_rddata_valid** and **dfi_rddata** signals will be ignored. All evaluations and delay changes are handled within the PHY. When the PHY finds the necessary edges and completes data eye training for a slice under training, the PHY drives the corresponding **dfi_rdlvl_resp** signal high, which informs the MC that the procedure is complete for the slice under training. The MC then de-asserts the **dfi_rdlvl_en** signal for the respective slice. This de-assertion should cause the PHY to stop driving the **dfi_rdlvl_resp** signal.

4.12.4.2.2 Per-Slice PHY-Driven Training Request

The PHY can request gate training by driving the **dfi_rdlvl_gate_req** signal or read data eye training by driving the_ **dfi_rdlvl_req** signal. To support an independent request to level per slice, the **dfi_rdlvl_req** and **dfi_rdlvl_gate_req** signals can be driven individually by each slice as Figure 66, “Per-Slice Read Training in DFI Training Mode, MCrdlvl_slice_group = 0x0F” and Figure 67, “Per-Slice Read Training in DFI Training Mode, MCrdlvl_slice_group = 0x00” show.

FIGURE 66. *Per-Slice Read Training in DFI Training Mode, $MC_{rdlvl\_slice\_group}$ = 0x0F*

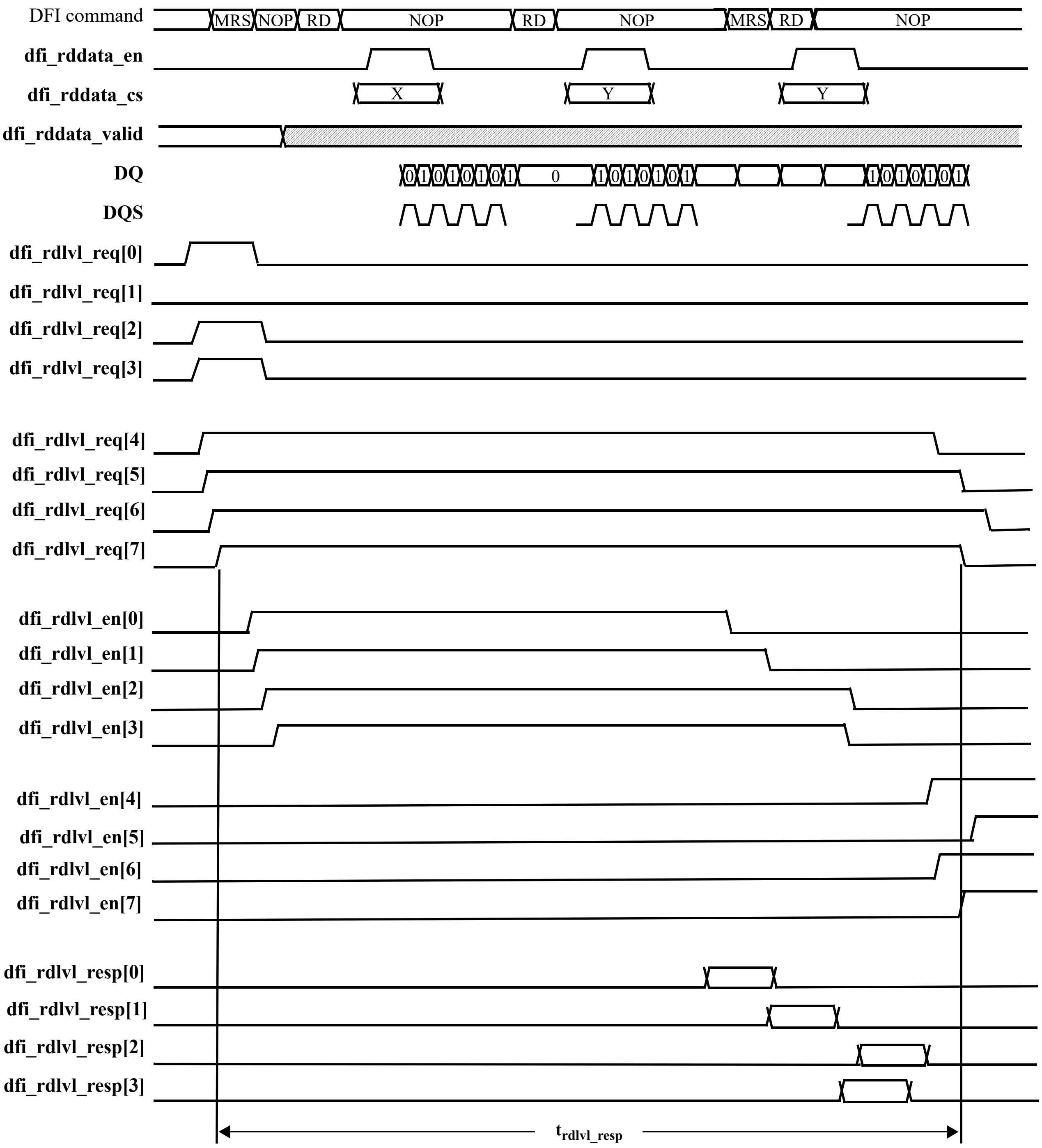
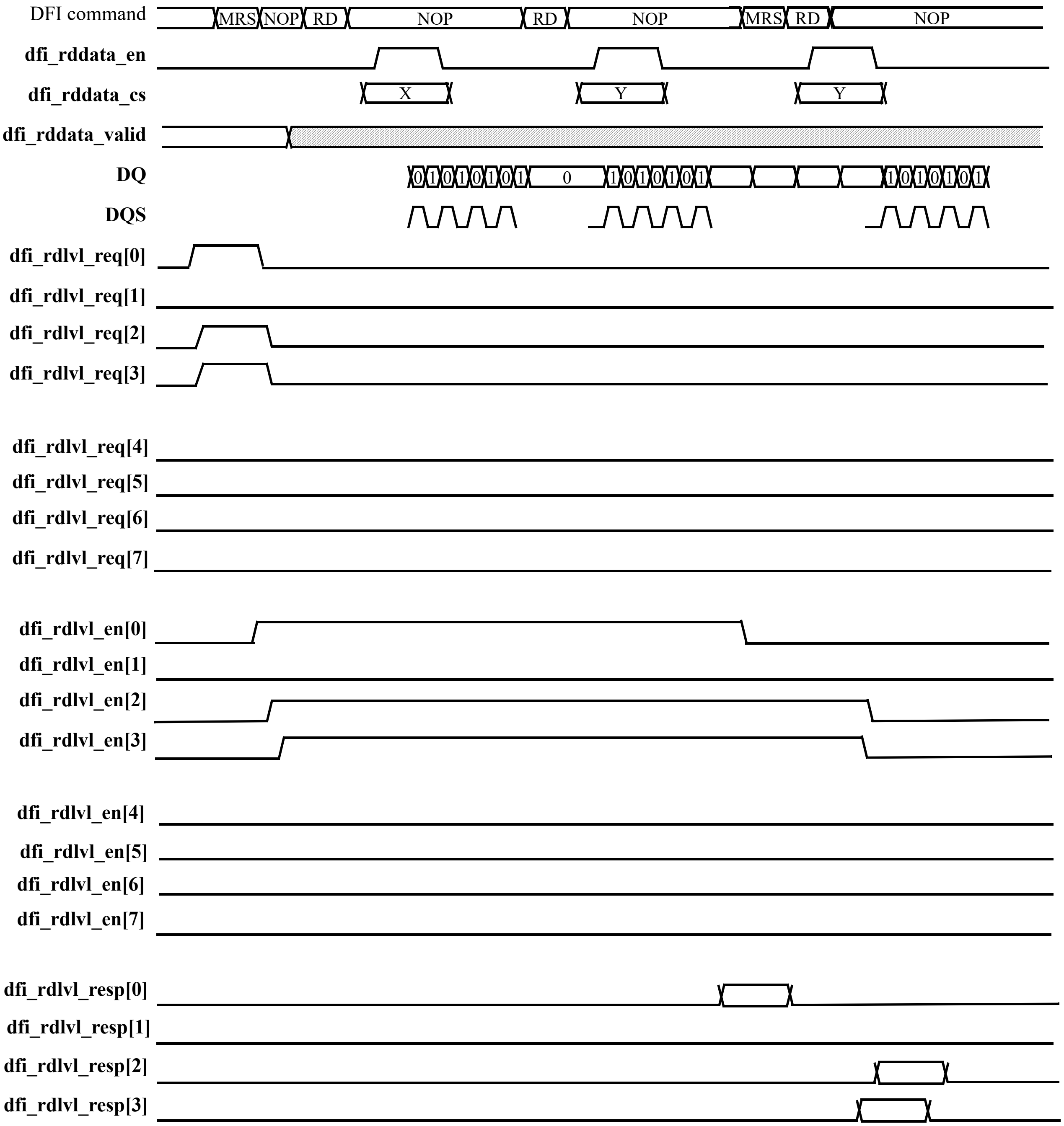


FIGURE 67. *Per-Slice Read Training in DFI Training Mode, $MC_{rdlvl\_slice\_group}$ = 0x00*



4.12.4.2.3 Use of the DFI Training Programmable Parameters

The MC and PHY can use the $\mathbf{mc_{rdlvl\_slice\_group[n]}}$ or $\mathbf{phy_{rdlvl\_slice\_group[n]}}$ parameters to group slices together in a system that supports per-slice training. Slices that are included in one parameter together will assert their **dfi_rdlvl_en** signals together. If the PHY asserts the **dfi_rdlvl_req** or **dfi_rdlvl_gate_req** signal for any of the slices that are set in $\mathbf{mc_{rdlvl\_slice\_group[n]}}$, the MC drives the **dfi_rdlvl_en** or **dfi_rdlvl_gate_en** signal, respectively, for all the slices that are set in that parameter. Using the $\mathbf{mc_{rdlvl\_slice\_group[n]}}$ parameter in this way allows the MC to assert the **dfi_rdlvl_en** signal and complete read training for all slices in multiple sets.

To keep leveling purely independent per slice, the $\mathbf{mc_{rdlvl\_slice\_group[n]}}$ should be cleared to 0 which allows the **dfi_rdlvl_req** and **dfi_rdlvl_gate_req** signals to be driven individually by each slice. To revert back to DFI 3.1 behavior which did not support per-slice leveling, each slice in $\mathbf{mc_{rdlvl\_slice\_group[n]}}$ should be set to 1.

If necessary, the PHY can use one or more parameters $\mathbf{phy_{rdlvl\_slice\_group[n]}}$ for specifying pre-defined sets of **dfi_rdlvl_en** signals that are needed to assert as a group by the MC when the PHY asserts the **dfi_rdlvl_req** or **dfi_rdlvl_gate_req** signals. As with the MC parameters, each slice in the $\mathbf{phy_{rdlvl\_slice\_group[n]}}$ can be cleared to 0 to support an independent request to level per slice, or each slice can be set to 1 to revert back to DFI 3.1 behavior.

NOTE: $\mathbf{mc_{rdlvl\_slice\_group[n]}}$ and $\mathbf{phy_{rdlvl\_slice\_group[n]}}$ values might or might not be compatible. The PHY and MC have independent parameters for the purpose of identifying compatibility and training requirements for both the PHY and MC. The MC values indicate desired MC behavior while the PHY values indicate actual PHY connections.

The MC is not required to respond to each individual per-slice **dfi_rdlvl_req** signal concurrently. In addition, the PHY must initiate training early enough for the following situations:

- For allowing for larger read training intervals in LRDIMM/RDIMM systems
- In cases where the MC does not respond to each individual per-slice **dfi_rdlvl_req** signal concurrently in LRDIMM/RDIMM systems
- In cases where the MC does not respond to each individual per-slice **dfi_rdlvl_req** signal concurrently

NOTE: NOTE: The MC must not assert a **dfi_rddata_en** signal for slices that do not assert **dfi_rdlvl_en** signals

4.12.4.3 dfi_lvl_pattern Encoding

In DDR4, LPDDR2 and LPDDR3 systems executing read training, the MC drives the **dfi_lvl_pattern** signal to define the required training sequence. In LPDDR4 systems, the **dfi_lvl_pattern** defines the index to the pattern. The **dfi_lvl_pattern** signal must be valid when the enable signal is asserted for gate training and data eye training. Similar to the enable signal, the **dfi_lvl_pattern** signal may transition when training completes.

In DDR4 systems, the **dfi_lvl_pattern** signal encodes the required training sequences. The width of **dfi_lvl_pattern** is (4 x DFI Read Leveling PHY I/F Width), with a 4-bit encoding defined per slice. The DDR4 encoding determines which MPR is accessed, which DRAM-defined MPR data format is used and if a unique, non-default pattern is utilized. The DFI bus does not define the unique patterns; the MPR signals contain a set of encodings which can be used to define unique patterns, with the MC and the PHY programming determining the specific patterns.

NOTE: The **dfi_lvl_pattern** signal must hold its value and not transition throughout the training sequence.

In Figure 68, “dfi_lvl_pattern Timing”, “Sequence X” can be any valid sequence defined by the valid **dfi_lvl_pattern** encodings. This value will indicate the pattern returned by the DRAM and evaluated by the PHY.

NOTE: If multiple sequences are required, the MC can assert any bit (or bits) of the **dfi_rdlvl_en** and/or **dfi_rdlvl_gate_en** signal multiple times with different values driven on **dfi_lvl_pattern**.

The **dfi_lvl_pattern** signal encoding supports both serial and staggered MPR formats without specific support for the parallel format. The non-default MPR value and format encodings can be used to select a parallel format if required. The **dfi_lvl_pattern** encodings support up to 4 unique, non-default MPR values and formats for DDR4. These encodings, ’b1000 to ’b1011, can select an MPR number, value, and format to create a unique training sequence. For example, the ’b1000 encoding, defined as Sequence 0, could select the following training sequence:

- MPR Number: MPR2
- MPR Value: 0x1111_1110
- MPR Format: Serial

A setting of ’b0000 will select a DDR3-like MPR setting.

FIGURE 68. ***dfi_lvl_pattern*** *Timing*

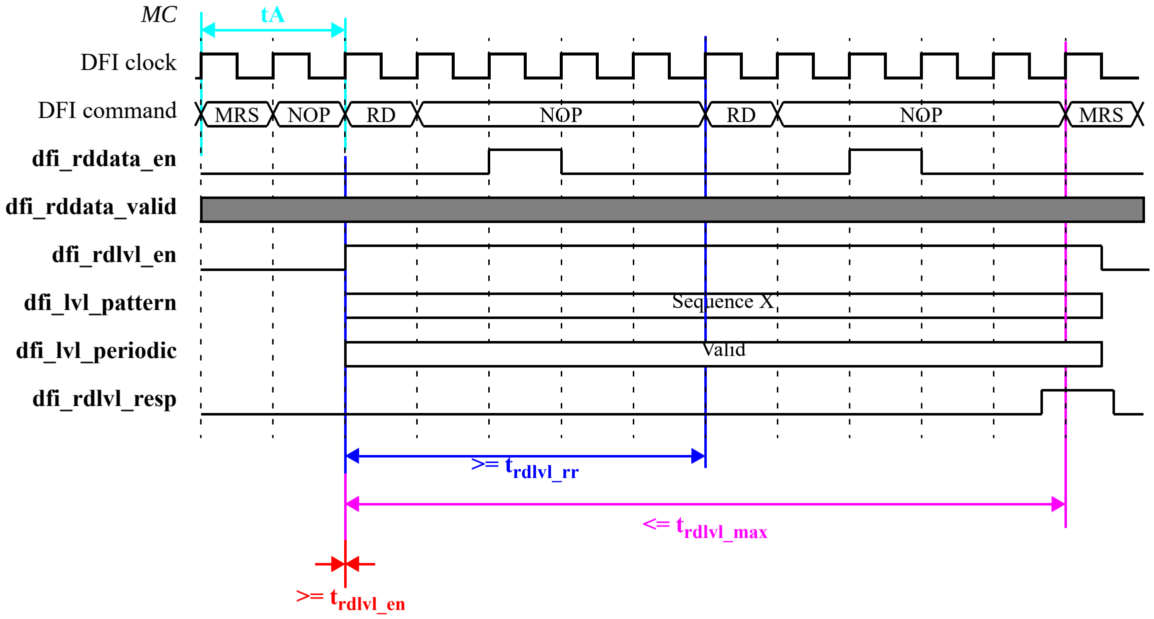


Table 42, “DDR4 Encoding of dfi_lvl_pattern” is used for gate training and data eye training for DDR4 DRAMs and Table 43, “LPDDR3 / LPDDR4 Encoding of dfi_lvl_pattern” is used for gate training and data eye training for LPDDR3 and LPDDR4 DRAMs.

TABLE 42. *DDR4 Encoding of dfi_lvl_pattern*

dfi_lvl_pattern				**Description**	
Non-default	**Format**	**MPR**			
[3]	**[2]**	**[1]**	**[0]**		
0	0	0	0	Serial format Default MPR Value	Access MPR0
0	0	0	1		Access MPR1
0	0	1	0		Access MPR2
0	0	1	1		Access MPR3
0	1	0	0	Staggered format Default MPR Value	Access starting with MPR0
0	1	0	1		Access starting with MPR1
0	1	1	0		Access starting with MPR2
0	1	1	1		Access starting with MPR3

TABLE 42. *DDR4 Encoding of dfi_lvl_pattern (Continued)*

dfi_lvl_pattern					
Non-default	Format	MPR			
[3]	[2]	[1]	[0]	Description	
1	0	0	0	Non-default MPR Value Any MPR Any format All defined through programming	Sequence 0
1	0	0	1		Sequence 1
1	0	1	0		Sequence 2
1	0	1	1		Sequence 3
1	1	0	0	Reserved for future use	
1	1	0	1		
1	1	1	0		
1	1	1	1		

TABLE 43. *LPDDR3 / LPDDR4 Encoding of dfi_lvl_pattern*

DFI_RDLVL_MPR_<MC | PHY>_OUT				Description			
Non-default	Format	MPR					
[3]	[2]	[1]	[0]	LPDDR3		LPDDR4	
0	0	0	0	Default	Access MPR32	Default	Access MPR32
0	0	0	1		Access MPR40		Access MPR40
0	0	1	0	Reserved		Reserved	
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0			Non-default MPR Value. Format and value determined through programming	Sequence 0
1	0	0	1				Sequence 1
1	0	1	0				Sequence 2
1	0	1	1				Sequence 3
1	1	0	0				Sequence 4
1	1	0	1				Sequence 5
1	1	1	0				Sequence 6
1	1	1	1				Sequence 7

In LPDDR2, LPDDR3 and LPDDR4 DRAMs, two DQ calibration mode registers are utilized: MR32 and MR40. The mode registers produce two data calibration patterns defined in Table 44, “Data Calibration Pattern”.

TABLE 44. *Data Calibration Pattern*

Pattern	Bit Time 0	Bit Time 1	Bit Time 2	Bit Time 3
"A" (MR32)	1	0	1	0
"B" (MR40)	0	0	1	1

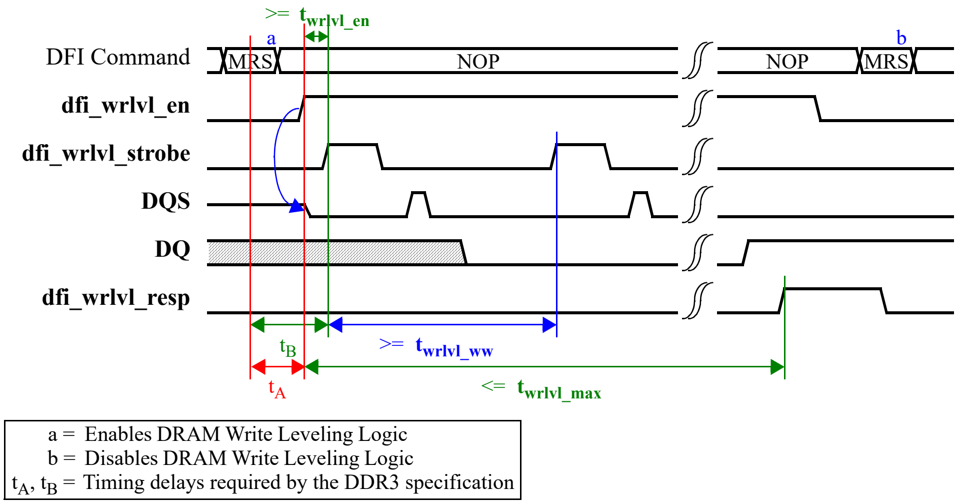
4.12.5 Write Leveling

The goal of write leveling is to locate the delay at which the write DQS rising edge aligns with the rising edge of the memory clock. By identifying this delay, the system can accurately align the write DQS within the memory clock.

The signals used in write leveling are: **dfi_wrlvl_en**, **dfi_wrlvl_req** and **dfi_wrlvl_strobe**. The corresponding parameter is $\mathbf{t}_{\mathbf{wrlvl\_en}}$. When the PHY initiates write leveling, the PHY can optionally send the **dfi_phy_wrlvl_cs** signal to indicate the targeted chip select. For more information on write leveling signals, refer to Section 3.2, "Write Data Interface" and for more information on the **dfi_phy_wrlvl_cs** signal, refer to Table 18, "Training Interface Signals".

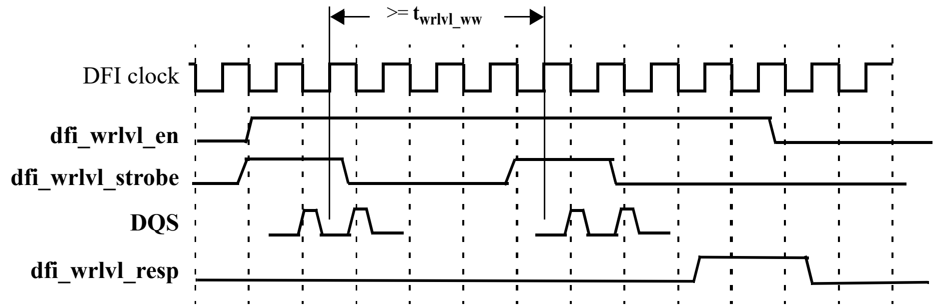
Figure 69, "Write Leveling in DFI Training Mode" demonstrates write leveling in DFI training mode. The MRS commands are used to enable and disable the write leveling logic in the DRAMs and the **dfi_wrlvl_en** signal is used for enabling or disabling the write leveling logic in the PHY.

FIGURE 69. *Write Leveling in DFI Training Mode*


>= t_{wrlvl_en}
a
b
DFI Command
MRS
NOP
NOP
MRS
dfi_wrlvl_en
dfi_wrlvl_strobe
DQS
DQ
dfi_wrlvl_resp
t_B
>= t_{wrlvl_ww}
t_A
<= t_{wrlvl_max}
a = Enables DRAM Write Leveling Logic
b = Disables DRAM Write Leveling Logic
t_A, t_B = Timing delays required by the DDR3 specification

LPDDR4 requires two DQS pulses. To allow two or more DQS pulses to be driven to the DRAM, during DFI write leveling, **dfi_wrlvl_strobe** can be driven for multiple clocks. For DFI4.0, each cycle that the **dfi_wrlvl_strobe** signal is driven generates a single DQS pulse. To support frequency ratio, the **dfi_wrlvl_strobe** signal is a phased signal. Figure 70, "DFI Write Leveling Interface with LPDDR4 Support" demonstrates write leveling in DFI training mode with LPDDR4 support.

FIGURE 70. *DFI Write Leveling Interface with LPDDR4 Support*



4.12.6 Support for Non-data Commands During Training

A PHY training sequence that exceeds the time the MC allows between refreshes or other required commands requires a method for executing these commands. This sequence must be supported by all MCs and PHYs supporting the DFI training interface for all supported DFI training sequences.

For DDR3 and DDR4, read training and write leveling require an MRW command to be issued to place memory in a data eye training (MPR) or write leveling operating mode. In either case, the memory specification does not permit any commands other than NOP and MRW while memory is in the special operating mode. Therefore, execution of a refresh will require issuance of an MRW to exit the special operating mode before executing the refresh command.

The sequence required to execute a refresh or other non-data command while running a read or write leveling sequence during training is:

1. Wait for the current command-to-command timing to complete ($\mathbf{t_{rdlvl\_rr}}$ or $\mathbf{t_{wrlvl\_ww}}$).
2. If required, execute an MRW to exit the training mode in the DRAM. In all cases, maintain the assertion of the bit (or bits) of the relevant training enable signal (**dfi_rdlvl_en**, **dfi_rdlvl_gate_en** or **dfi_wrlvl_en**).
3. Execute a refresh or other non-data command.
4. If required, execute an MRW to re-enter the identical training state previously occupied.
5. Continue training sequence after meeting all timing requirements associated with MRW-to-command timing as defined at the beginning of the sequence.

Maintaining the assertion of the bit (or bits) of the enable signal (**dfi_rdlvl_en**, **dfi_rdlvl_gate_en** or **dfi_wrlvl_en)** allows the PHY to distinguish the exit portion of this command sequence from an aborted training sequence and the enter portion of this command sequence from the start of a new sequence.

Figure 71, “Refresh Execution During Read Training Operation” is a read training example that shows that the PHY should continue the training from the point where it was interrupted. Since the PHY did not assert the ’b11 response signal on the **dfi_rdlvl_resp** signal, the sequence is not complete.

Figure 72, “Refresh Execution During Read Training Operation With Completion Response” shows that if the response is asserted any time during the refresh sequence, the training sequence should be completed as previously defined.

The training sequences may require the MC to maintain certain signal states while training is in progress. For example, during write leveling the MC may drive ODT. When exiting the training sequence to execute a non-data command such as a refresh, the MC may need to change the state of such signals as appropriate for the commands being executed.

FIGURE 71. *Refresh Execution During Read Training Operation*

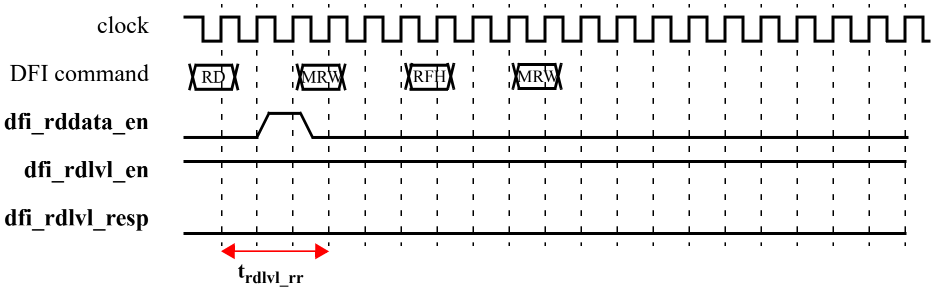
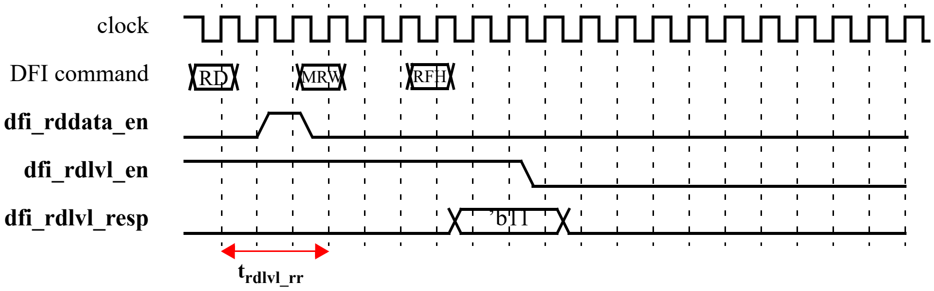


FIGURE 72. *Refresh Execution During Read Training Operation With Completion Response*



4.12.7 CA Training

CA training optimizes the CA bus setup and hold times relative to the memory clock. A mode register write (MRW) enables the CA training mode in the DRAM. Two mode registers are defined to enter CA training to allow testing across the entire CA bus, and one mode register is defined to exit CA training. The DFI specification does not dictate the CA bus values driven during testing. “CA training” refers to CA data eye training for LPDDR3 memories and to both CA data eye and VREF value training for LPDDR4 memories.

The DFI 4.0 specification supports enhanced LPDDR4 CA training features including adjustment to train the VREF value, frequency changes to train at various frequency set points, and additional training pattern options. The LPDDR4 devices define a sequence that incorporates frequency change events during CA training, which are used for training higher, terminated speeds.

CA training expands the functionality of the **dfi_cke** signal (driven LOW to enable the memory to drive the CA value on the DQ bus) and the **dfi_cs** signal (used to generate a calibration command transferring the CA bus pattern to the memory). The CA pattern can be any value defined by the system for the purpose of training. The PHY can vary the delay settings of the command bus relative to the memory clock to determine the maximum setup and hold time settings.

The PHY may request CA training by asserting a bit on the **dfi_calvl_req** signal. The MC will respond to an assertion of a bit of the **dfi_calvl_req** signal by asserting the corresponding bit of the **dfi_calvl_en** signal within $\mathbf{t_{dfi\_calvl\_resp}}$ cycles. The **dfi_calvl_en** signal enables the CA training logic in the PHY for a slice and begins the CA training sequence. During initialization or a frequency change operation, the MC should be configured to initiate CA training when necessary, but the PHY is not required to assert any bits of the **dfi_calvl_req** signal. If the PHY manages CA training during a frequency change, a bit of the **dfi_calvl_req** signal may be asserted while the **dfi_init_complete** signal is de-asserted, but must be asserted before or coincident with the assertion of the **dfi_init_complete** signal in order to guarantee that the training occurs prior to restarting normal command traffic.

4.12.7.1 CA Training Sequence

During CA training, the PHY is responsible for generating the CA patterns to be driven on the memory bus, adjusting the timing of the CA bus relative to the memory clock, and checking the CA pattern captured from the DQ bus against the expected CA pattern.

The CA training sequence can be initiated by either the PHY or the MC. If the PHY requests CA training, the CA training sequence begins with step 1. If the MC initiates training, the process begins with step 3. The CA training sequence is as follows:

1. The PHY asserts a bit on the **dfi_calvl_req** signal.

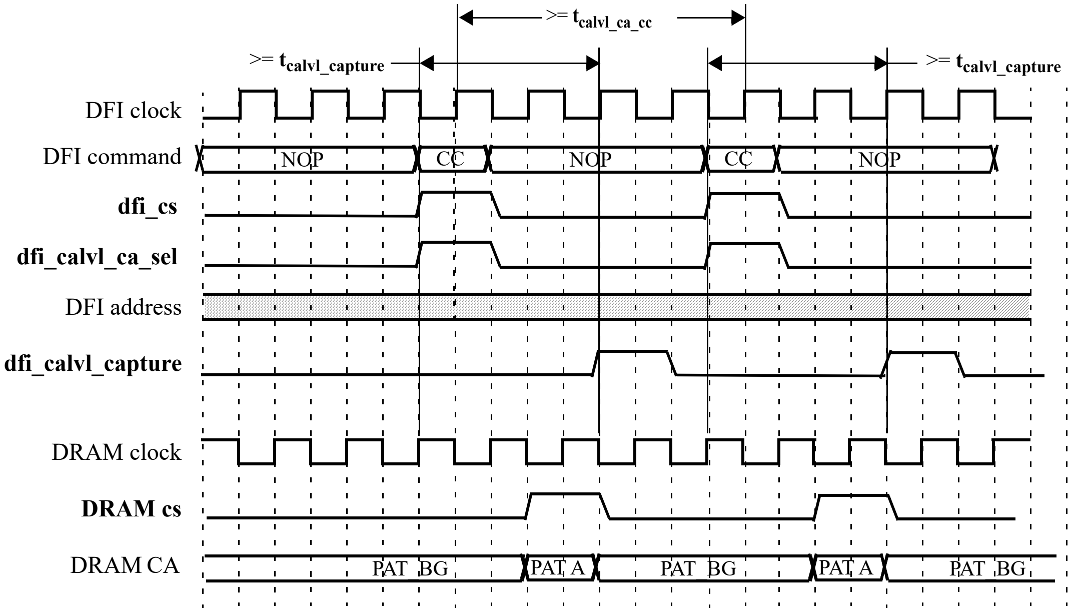
   NOTE: This is an optional signal for the PHY since a PHY is not required to initiate CA training.
2. Within $\mathbf{t_{calvl\_resp}}$ DFI clock cycles, the MC is required to begin training.
3. The MC executes the MRW write commands to each memory to put the targeted chip select into the CA training state.
4. When the memory is in the CA training state, the MC asserts a bit of the **dfi_calvl_en** signal to indicate that the PHY should begin operating in the CA training mode for that slice.
5. The $\mathbf{t_{calvl\_en}}$ timing parameter is met.
6. The MC must de-assert the **dfi_cke** signal for the target chip select.
7. The PHY sends the background pattern to the DRAM (for CA DQ training).
8. For LPDDR4 devices, set the VREF value for the CA training operation on the **dfi_calvl_data** signal.
9. For LPDDR4 devices, once both the $\mathbf{t_{calvl\_strobe}}$ and the $\mathbf{t_{calvl\_data}}$ are met, assert the **dfi_calvl_strobe** signal.
10. The MC drives periodic CA training calibration commands, as defined by the $\mathbf{t_{calvl\_cc}}$ timing parameter. The MC uses the **dfi_calvl_ca_sel** to define whether the foreground or background pattern is to be sent to the DRAMs.
11. At least $\mathbf{t_{calvl\_capture}}$ cycles after the calibration command, the **dfi_calvl_capture** signal can be pulsed.
12. Before the $\mathbf{t_{calvl\_max}}$ timer expires, the **dfi_calvl_resp** signal (and the **dfi_calvl_result signal for LPDDR4)**must change and reflect the status of the operation.
13. The MC de-asserts the set bit of the **dfi_calvl_en** signal and asserts the **dfi_cke** signal.

For CA training, the PHY will drive background and foreground training patterns to the DRAM and is responsible for specifying all of the patterns utilized on the CA bus during CA training. The DFI interface does not define the CA values to be driven during CA training or their means of communication. The CA patterns are generated and checked within the PHY.

Figure 73, “dfi_calvl_ca_sel Timing (Matched Frequency), tcalvl_cs_ca = 0; tcalvl_ca_sel = 1” and Figure 74, “dfi_calvl_ca_sel Timing (Matched Frequency), tcalvl_cs_ca = 1; tcalvl_ca_sel = 3” illustrate the functionality of the

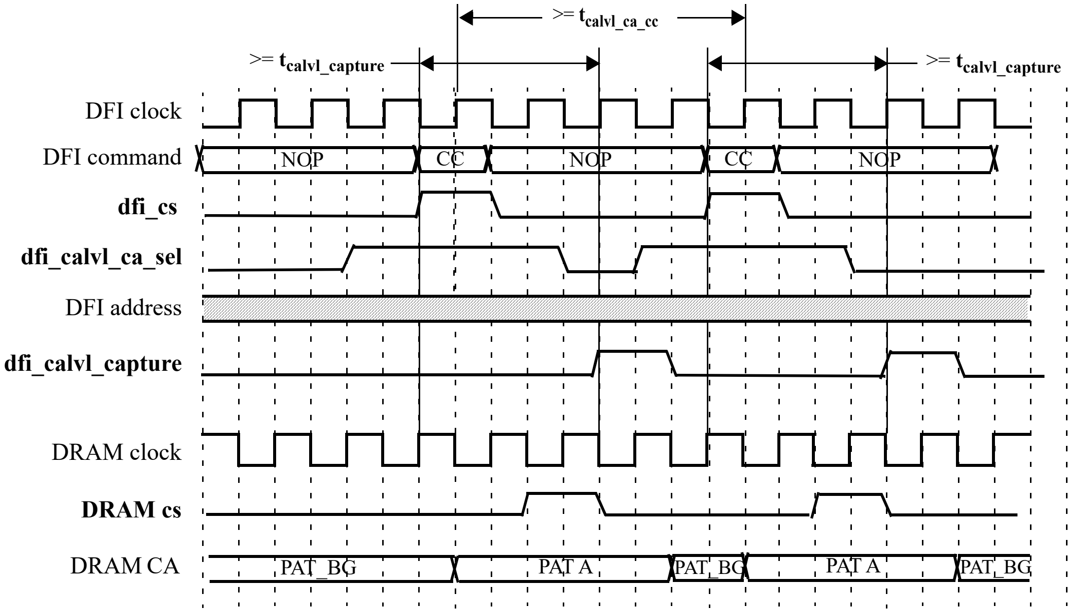
dfi_calvl_ca_sel signal. In both diagrams, each system uses matched frequency ratio. The diagrams show two different sets of timing values.

FIGURE 73. ***dfi_calvl_ca_sel*** *Timing (Matched Frequency),* $t_{calvl\_cs\_ca} = 0$; $t_{calvl\_ca\_sel} = 1$



NOTE: The polarity of the chip select signal is defined by the DRAM. Applies to **dfi_cs** and **DRAM_cs**.

FIGURE 74. *dfi_calvl_ca_sel Timing (Matched Frequency),* $t_{calvl\_cs\_ca} = 1$; $t_{calvl\_ca\_sel} = 3$



4.12.7.2 CA VREF Training

LPDDR4 memories support CA VREF training. The purpose of CA VREF training is to locate the CAVREF with the largest data eye. To perform CA VREF training during CA training, the MC will drive different VREF values on the 7-bit **dfi_calvl_data** bus for each test. DQ data bit reordering between the ASIC and DRAM must be taken into account for correct operation.

After each test, the PHY will report the status of the operation on the **dfi_calvl_resp** and **dfi_calvl_result** signals. The MC will also report status on the **dfi_calvl_done** signal. The value of these signals will determine if more testing is required, or if the testing is complete and the ideal VREF value has been determined.

Timing diagrams illustrating CA training with VREF training and frequency change events are included as Figure 75, “CA Training with CA VREF Set” and Figure 76, “VREF Change During CA Training”.

The procedure for CA VREF training is as follows:

1. The Controller performs a CA training sequence as detailed in Section 4.12.7.1, “CA Training Sequence” with the first VREF value to test on the **dfi_calvl_data**.
2. Once the procedure completes, the system will report the status on the **dfi_calvl_resp**, **dfi_calvl_result** and **dfi_calvl_done** signals.

TABLE 45. *VREF Testing Results*

dfi_calvl_resp	dfi_calvl_done	Result Summary	Value to Program on dfi_calvl_data [a]
’b01	’b0	The current test is complete, but the entire operation is not. Repeat the procedure in Section 4.12.7.1, “CA Training Sequence” with a new value.	Next data value to test. The **dfi_calvl_result** signal provides feedback on how the last two test compared to previous tests.
’b01	’b1	The current test is complete, and the MC has sent all data values. Re-program the **dfi_calvl_data** signal with the first value and repeat the procedure in Section 4.12.7.1, “CA Training Sequence”.	First data value to test
’b11	’b0	The current test is complete, but the entire operation is not. Re-program the **dfi_calvl_data** signal and repeat the procedure in Section 4.12.7.1, “CA Training Sequence”.	Next value to test. The **dfi_calvl_result** signal provides feedback on how the last two test compared to previous tests.
’b11	’b1	All data patterns have been tested.	None

[a] Please note that the PHY may ignore the **dfi_calvl_data** input and drive a different value on the appropriate DQ outputs.

3. The Controller will repeat the CA training sequence with the next valid value for VREF on the **dfi_calvl_data** signal.
4. When the Controller has tested ALL valid VREF values, then it will assert the **dfi_calvl_done** signal.
5. Then VREF training is complete.

When training completes, training values that the MC (VREF, for example) and the PHY (delay settings) control must be in sync so that the DRAM VREF and PHY delays are set to corresponding values.

NOTE: If the CA VREF values are programmed by using MRS DRAM commands, the explicit VREF CA training signals **dfi_calvl_data** and **dfi_calvl_strobe** are not used.

4.12.7.3 Frequency Change Events During CA Training

The defined LPDDR4 CA training procedure could include multiple frequency changes. The existing frequency change signals of the status interface using **dfi_init_start** and **dfi_init_complete** is applicable during CA training. The timing parameters $\mathbf{t}_{\mathbf{init\_start\_min}}$ and $\mathbf{t}_{\mathbf{init\_complete\_min}}$ define the relationship between the frequency change signals and the CA training interface.

Figure 75 and Figure 76 illustrate the CA training DFI bus. Figure 76, “VREF Change During CA Training” demonstrates the timing and control when a value of ’b01 is returned on the **dfi_calvl_resp** signal. In this case, the **dfi_cke** signal is not required to re-assert.

FIGURE 75. *CA Training with CA VREF Set*

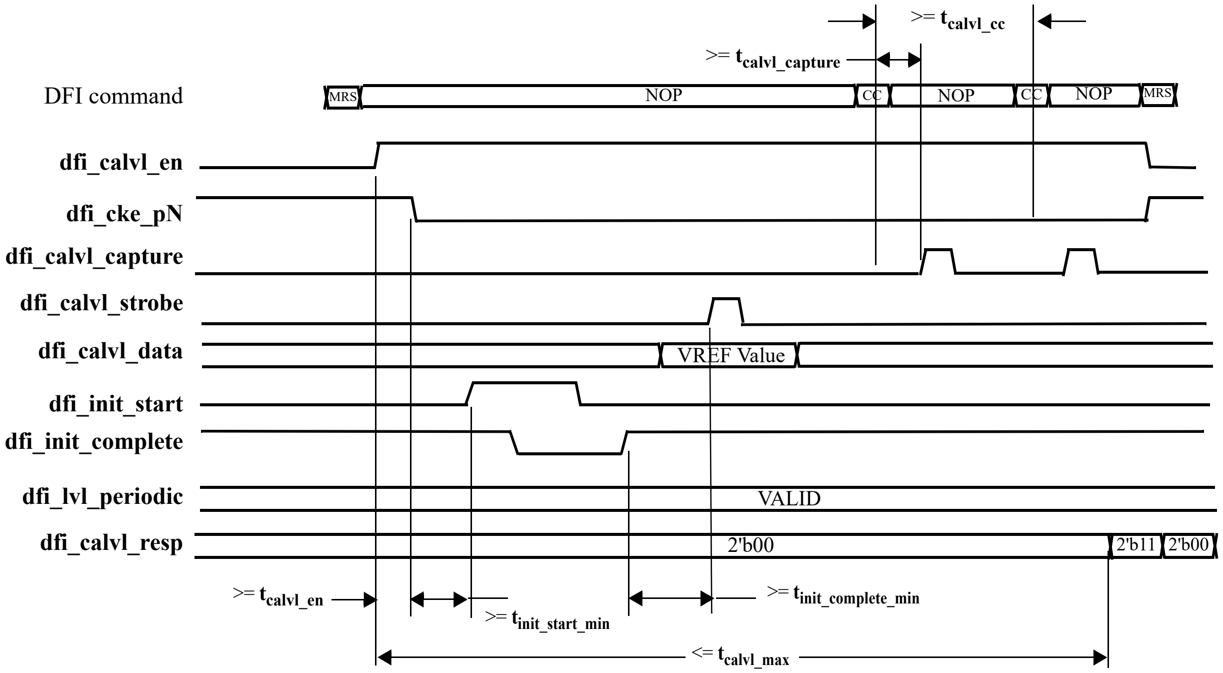
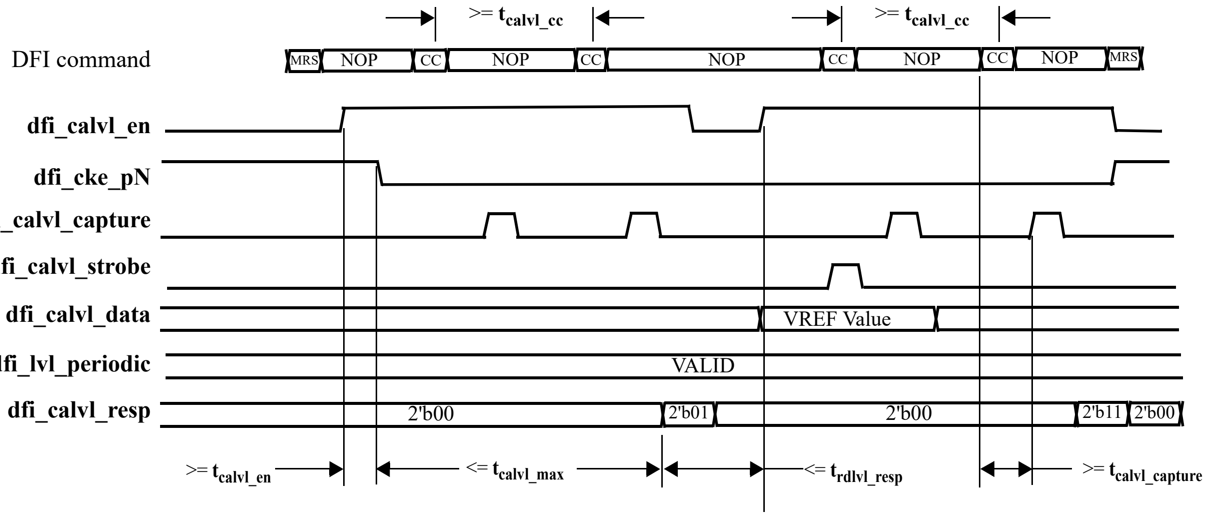


FIGURE 76. *VREF Change During CA Training*



4.12.8 Write DQ Training

The goal of write DQ training is to locate the delay at which the write DQ rising edge aligns with the rising edge of the DRAM clock. By identifying this delay, the system can accurately align the write DQ within the DRAM clock.

The signals used in write DQ training are: **dfi_wdqlvl_done**, **dfi_wdqlvl_en**, **dfi_wdqlvl_req**, **dfi_wdqlvl_resp** and **dfi_wdqlvl_result**. The programmable parameter is **phy**$_{\mathbf{wdqlvl\_bst}}$. When the PHY initiates write DQ training, the PHY can optionally send the **dfi_phy_wdqlvl_cs** signal to indicate the targeted chip select. For more information on write DQ training signals, refer to Section 3.2, “Write Data Interface” and for more information on the **dfi_phy_wdqlvl_cs** signal, refer to Table 18, “Training Interface Signals”.

Because write DQ training might require iterating through multiple variables, including VREF values, write data patterns, delay settings, etc.; sequence control is required from both the MC and the PHY. The **dfi_wdqlvl_done** signal indicates when the MC has set the final VREF value associated with the training sequence. If training is complete at the end of the write DQ training sequence in which the MC asserts the **dfi_wdqlvl_done** signal, the PHY must assert **dfi_wdqlvl_resp** = ’b11 as the response at the end of the sequence. Alternatively, if the PHY requires additional variables, the PHY should assert **dfi_wdqlvl_resp** = ’b01 and the training routine would continue. In this case, the MC should loop back to the first VREF value that was driven at the beginning of training, and iterate through the VREF values again. The MC is to continue looping through the values until the PHY responds with **dfi_wdqlvl_resp = ’**b11 and training is complete.

The **dfi_wdqlvl_result** signal communicates the status of the operation to the MC. This signal indicates how the results of the current value compare with the best previous value, either better or worse.

NOTE: The **dfi_wrdata** bus is not used during write DQ training. The DRAM write data is generated in the PHY.

When training completes, training values controlled by the MC (VREF for example) and the PHY (delay settings) must be in sync so that the DRAM VREF and PHY delays are set to final, trained values.

The write DQ training sequence can be initiated by either the PHY or the MC.

If the PHY requests write DQ training:

1. The PHY asserts a bit of the **dfi_wdqlvl_req** signal.

   NOTE: This is an optional signal for the PHY since a PHY is not required to initiate write DQ training.
2. Within $\mathbf{t}_{\mathbf{wdqlvl\_resp}}$ DFI clock cycles, the MC is required to begin training.

The write DQ training sequence is followed regardless of the requester:

1. The MC asserts the **dfi_wdqlvl_en** parameter. This could be a response to the PHY request or the MC could assert it on its own.
2. The MC issues write bursts, followed by read bursts on the DFI bus. Commands either access a scratch pad FIFO within the DRAM (LPDDR4) or the DRAM array. The number of write bursts issued in succession before the same number of read bursts is controlled by the $\mathbf{phy}_{\mathbf{wdqlvl\_bst}}$ parameter. The write to write command timing can be modified to add delay between write commands by using the t_{wdqlvl_ww} timing parameter. The read-to-write command timing is also controllable through the $\mathbf{t}_{\mathbf{wdqlvl\_rw}}$ timing parameter.
3. The MC controls data timing by asserting the **dfi_wrdata_en** and **dfi_rddata_en** signals, but the data is generated and verified internally in the PHY. If necessary,
4. The PHY transfers the commands to the DRAM, generating the required data on the DQ outputs for the write that is based on **dfi_wrdata_en** timing.
5. The PHY receives the read data from the DRAM and compares it to the expected data. The PHY delays are to be adjusted as needed.
6. When the sequence is complete, the PHY responds with a value of ’b01 or ’b11 on the appropriate **dfi_wdqlvl_resp** signal bit. If the PHY returns a ’b01 response, the MC will de-assert the per-slice **dfi_wdqlvl_en** signal, set up the next value, and then re-assert the **dfi_wdqlvl_en** signal and repeat the process, starting from step 2. If the PHY returns a ’b11 result, the process is complete. All training must also meet the timing defined in the $\mathbf{t}_{\mathbf{wdqlvl\_max}}$ timing parameter.

Figure 77, “WR DQ Training with phywdqlvl_bst = 1” and Figure 78, “WR DQ Training with phywdqlvl_bst = 2” illustrate the timing of the write DQ training interface.

FIGURE 77. *WR DQ Training with* $\mathbf{phy}_{\mathbf{wdqlvl\_bst}}$ *= 1*

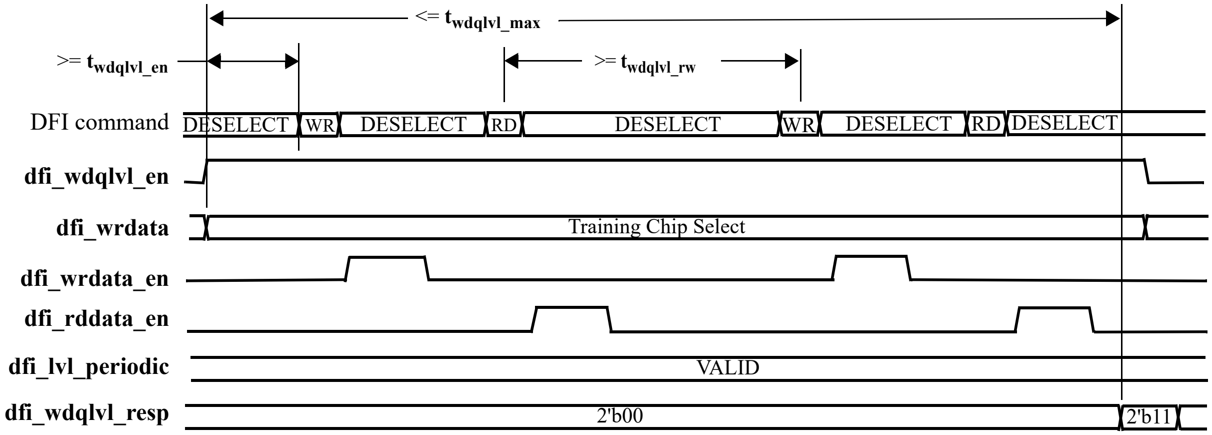


FIGURE 78. *WR DQ Training with* **phy**$_{wdqlvl\_bst}$ *= 2*

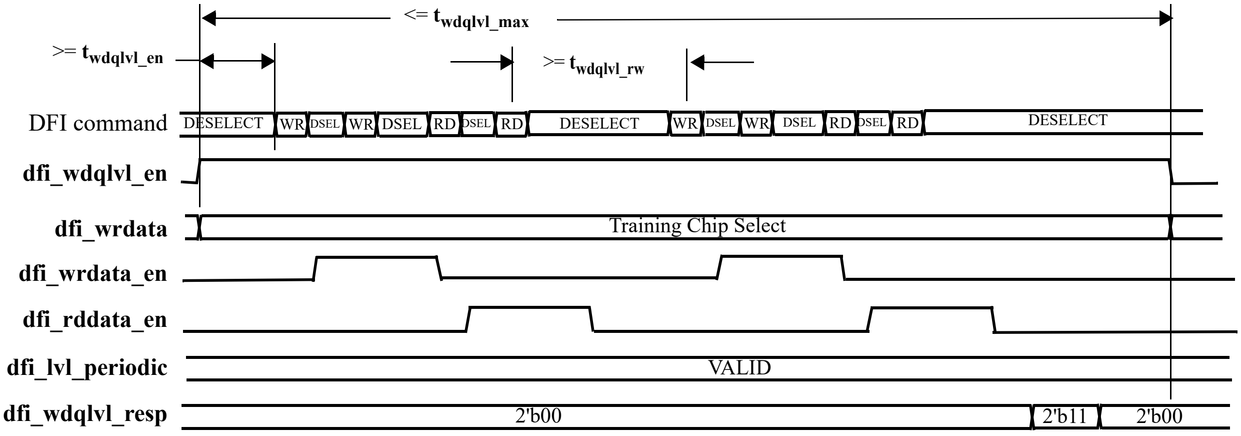
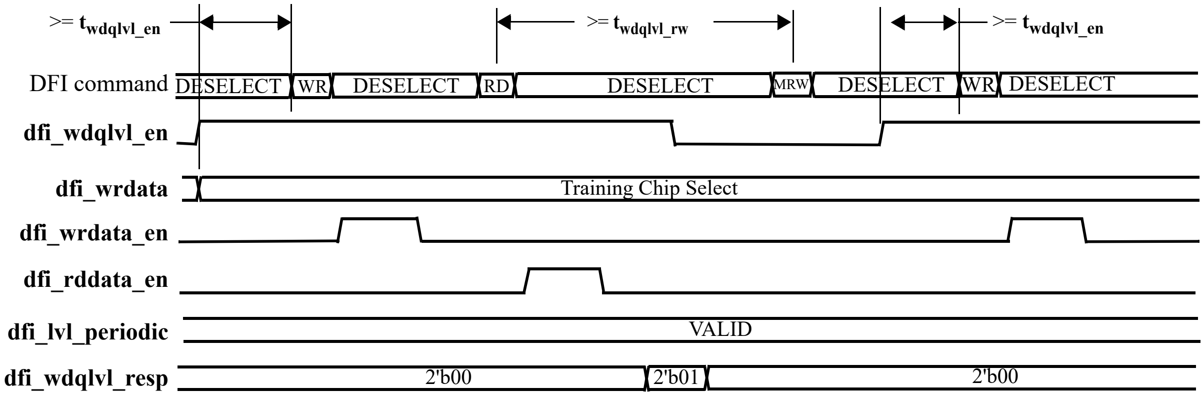


Figure 79, "WR DQ Training with phywdqlvl_bst = 'b1, dfi_wdqlvl_resp = 'b10" illustrates write DQ training with **phy**$_{wdqlvl\_bst}$ = 1, **dfi_wdqlvl_resp** = 2'b01, and an MRW issued to set new VREF value by MC for the next sequence.

FIGURE 79. *WR DQ Training with* **phy**$_{wdqlvl\_bst}$ *= 'b1,* ***dfi_wdqlvl_resp*** *= 'b10*



NOTE: While the MC encodes the MRW command on the **dfi_address** bus for the next VREF value, the PHY can disregard the controller VREF setting in the DFI command. It can set the appropriate address bits to an alternate value.

4.13 Low Power Control Handshaking

If the PHY has knowledge that the DFI will be idle for a period of time, the PHY may be able to enter an MC-initiated low power state. During low power handshaking, the DFI clock must maintain a valid and constant clock operating frequency until **dfi_lp_data_req**, **dfi_lp_ack** and **dfi_lp_wakeup** have reached a constant state.

When the MC detects an idle time, the MC asserts the **dfi_lp_data_req** signal to the PHY and the **dfi_lp_wakeup** signal with the wakeup time required. The PHY can acknowledge the request and go into low power mode based on the wakeup time required and remain in low power mode as long as the request and acknowledge are both asserted, or the PHY can disregard the request and not change power states even if a low power opportunity request was acknowledged.

If the request is acknowledged through the assertion of the **dfi_lp_ack** signal, the PHY may enter a low power mode as long as the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal remains asserted. Once the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is de-asserted, the PHY must return to normal operating mode within the number of cycles indicated by the **dfi_lp_wakeup** signal.

When **dfi_lp_ctrl_req** or **dfi_lp_data_req** and **dfi_lp_ack** have asserted, the MC will not de-assert **dfi_lp_ctrl_req** or **dfi_lp_data_req** to increase the wakeup time. In order for the PHY to recognize that the MC has increased the wakeup time, the PHY must monitor the **dfi_lp_wakeup** signal.

Wakeup time is a specific number of cycles ($\mathbf{t_{lp\_wakeup}}$ cycles) in which the PHY is expected to respond to a signal change (the de-assertion of either the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal) on the DFI. If **dfi_lp_ctrl_req** and **dfi_lp_data_req** signals are both asserted, both must be de-asserted at the same time. The DFI specification defines up to 16 different wakeup times; neither the MC nor the PHY are required to support all of the defined wakeup times. Generally, the PHY should enter the lowest supported power state that allows low power exit within the required wakeup time. The wakeup time may be an average or estimated delay; therefore, exceeding the wakeup time should not be treated as an error condition.

The MC guarantees that **dfi_lp_ctrl_req** or **dfi_lp_data_req** will be asserted and the **dfi_lp_wakeup** signal will be constant for at least $\mathbf{t_{lp\_resp}}$ cycles, allowing the PHY time to respond. The PHY may respond or ignore the low power mode request. To acknowledge the request, the PHY must assert the **dfi_lp_ack** signal within $\mathbf{t_{lp\_resp}}$ clock cycles of the request signal assertion, during which time the MC must hold the **dfi_lp_wakeup** signal constant. Once the request has been acknowledged by the PHY, the MC may de-assert the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal. The PHY is expected to de-assert the **dfi_lp_ack** signal within $\mathbf{t_{lp\_wakeup}}$ clock cycles after the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is de-asserted and be ready for normal operation.

Figure 80, “Low Power Control Handshaking Timing Diagram” shows a sequence in which the request is acknowledged.

FIGURE 80. *Low Power Control Handshaking Timing Diagram*

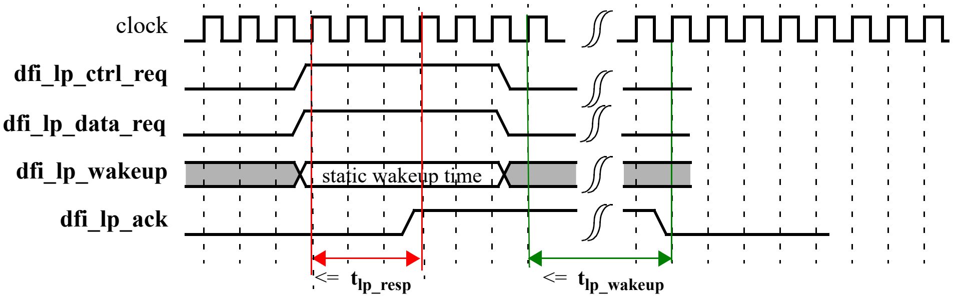
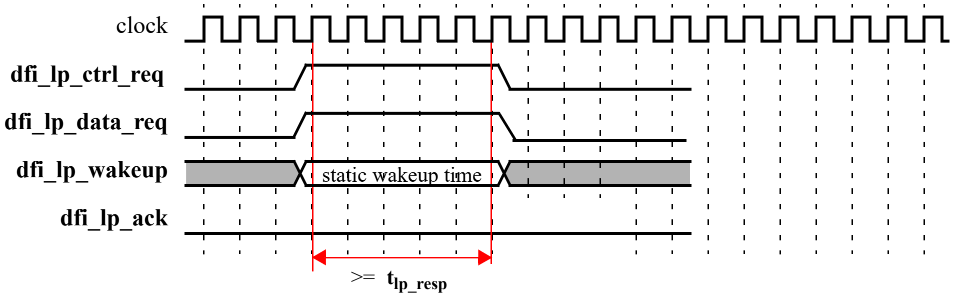


Figure 81, “Low Power Control Request with No Response” shows that the **dfi_lp_ack** signal is not required to assert when the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is asserted. The MC must assert the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal for at least $\mathbf{t_{lp\_resp}}$ cycles. If the **dfi_lp_ack** signal is not asserted within $\mathbf{t_{lp\_resp}}$ cycles, the PHY must not assert the acknowledge for the current request. The **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal should be de-asserted after $\mathbf{t_{lp\_resp}}$ cycles have elapsed without an acknowledge.

FIGURE 81. *Low Power Control Request with No Response*



After the request has been acknowledged, the MC may increase the time that the PHY has to respond beyond the time that was initially defined. The MC is allowed to change the **dfi_lp_wakeup** signal to a larger value as long as both the **dfi_lp_ack** and **dfi_lp_ctrl_req** or **dfi_lp_data_req** signals are asserted. This results in a longer $\mathbf{t_{lp\_wakeup}}$ time for the PHY. The value of the **dfi_lp_wakeup** signal when the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is de-asserted will be used to define the $\mathbf{t_{lp\_wakeup}}$ time.

Figure 82, “Low Power Control Handshaking Timing Diagram with Multiple Wakeup Times” shows a scenario with the assumption that the wakeup time is increased with each change.

FIGURE 82. *Low Power Control Handshaking Timing Diagram with Multiple Wakeup Times*

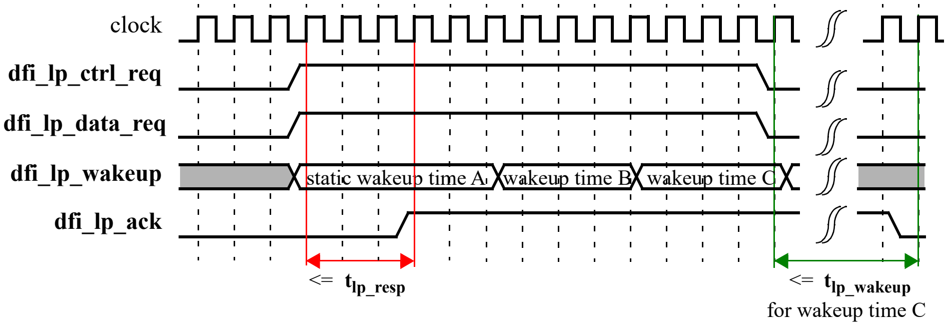


Figure 83, “Low Power State Progressing From an Active Control Interface to Inactive” illustrates **dfi_lp_ctrl_req** and **dfi_lp_data_req** when progressing from a low power state that requires use of the control interface into a low power state that can operate when the Control Interface is inactive.

FIGURE 83. *Low Power State Progressing From an Active Control Interface to Inactive*

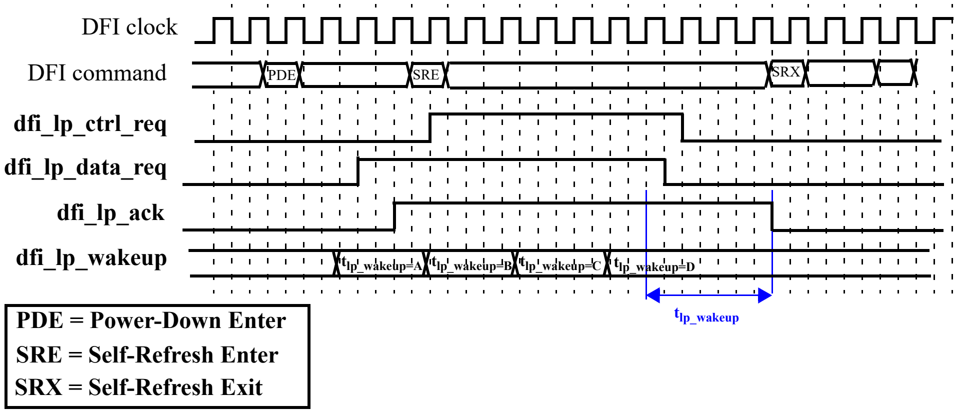
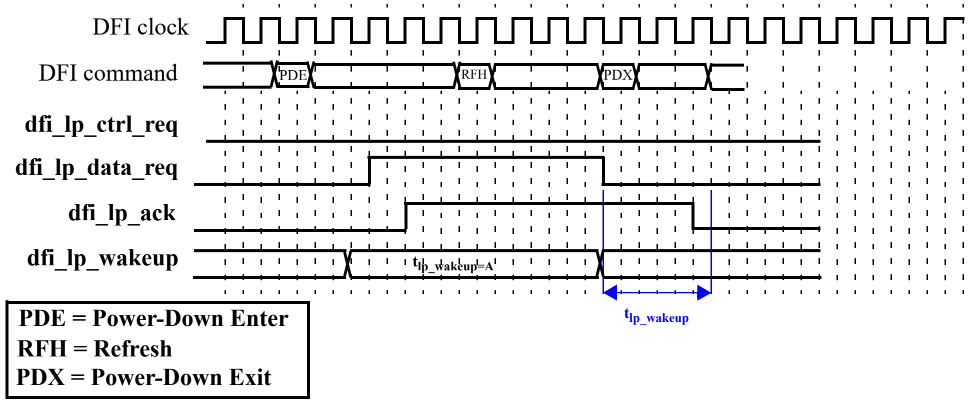


Figure 84, “Low Power State Requiring Ongoing Use of the Control Interface” illustrates **dfi_lp_data_req** when entering a low power state that must continue to execute memory commands.

FIGURE 84. *Low Power State Requiring Ongoing Use of the Control Interface*



4.14 Error Signaling

The optional DFI error interface enables the PHY to communicate error information from the PHY to the MC.

For data errors, the timing parameter is defined as the max delay from **dfi_wrdata_en** or **dfi_rddata_en** to the assertion of the **dfi_error** signal.

For command errors, the timing parameter is defined as the max delay from command to the assertion of the **dfi_error** signal. Since the timing parameter is a maximum delay, it is not always possible to correlate the error with a specific command.

Figure 85, "Example of Error Condition" shows the error interface for the control signal interface of the PHY. The error condition does not affect RD A and RD B, and may affect RD C and/or RD D.

FIGURE 85. *Example of Error Condition*

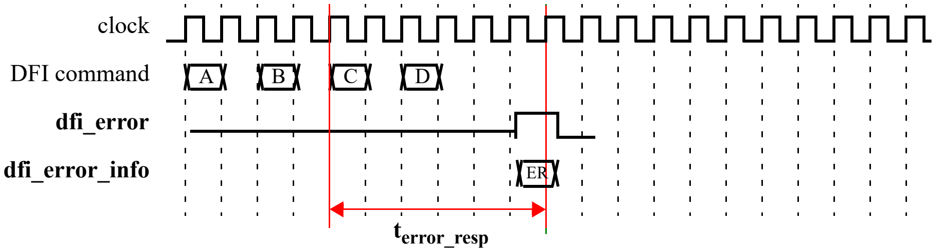


Table 46, "Error Codes" defines specific error codes, error codes reserved for future use, and error codes available for design-specific definition. Not all defined error codes apply to all systems and all error types are optional unless required as part of another interface.

If multiple errors occur in a single clock cycle, the PHY is responsible for resolving communication through error prioritization, using multiple clocks, or through another means.

TABLE 46. *Error Codes*

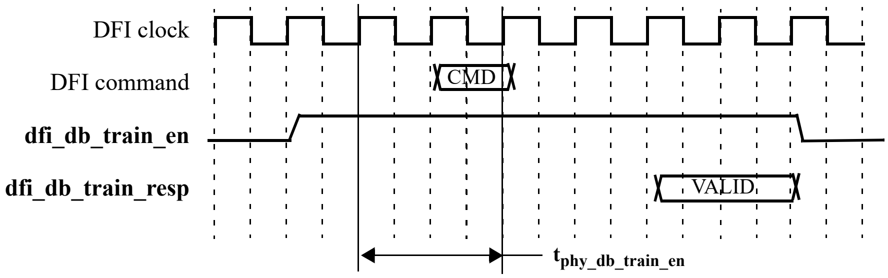
Error Response Code	Name	Description
0000	General purpose	The PHY indicates a general purpose error.
0001	Internal PHY	The PHY indicates some internal error condition.
0010 - 0111	Reserved	Reserved for a future definition.
1000 - 1111	User-defined	These error codes are available for user definition.

4.15 DB Training

The DB training interface is only used in PHY evaluation mode. In this case, the MC sets up training the data buffer of the memory and then receives the response. These signals are not used in PHY independent mode.

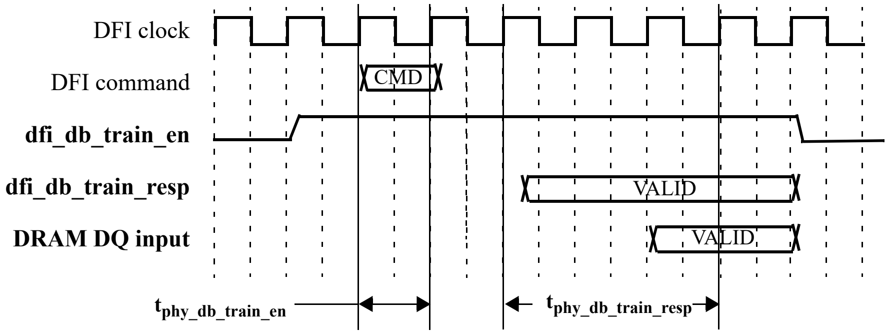
Figure 86, "DB Training" shows a DB training example interface, used for multiple DB-to-DRAM training modes. Not all modes require commands from the MC, and round trip times are not consistent.

FIGURE 86. *DB Training*



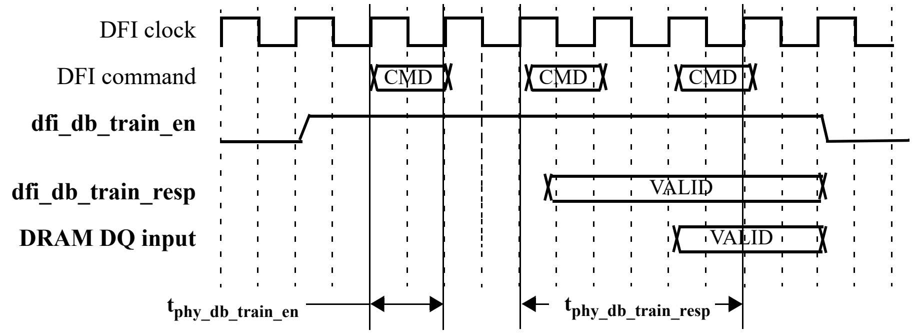
No explicit timing relationship has been defined from the **dfi_db_train_en** signal or DFI CMD to **dfi_db_train_resp** signal. Instead, the MC is to use the $\mathbf{t_{phy\_db\_train\_resp}}$ parameter for determining the delay that is required for transferring the response through the PHY. This information is to be combined with round-trip delay information that are based on JEDEC and board delays. Figure 87, "DB Training with Single MPR Read Command" illustrates the $\mathbf{t_{phy\_db\_train\_resp}}$ timing.

FIGURE 87. *DB Training with* ***Single MPR Read Command***



DB training can involve one or multiple training sequences, which occur while **dfi_db_train_en** is asserted. Figure 88, "DB Training with multiple MPR Read Commands" illustrates the case where multiple MPR read commands are issued to the DRAM/DB for read receiver enable training.

FIGURE 88. *DB Training with multiple MPR Read Commands*



NOTE: No DFI timing parameters are required for CMD-to-CMD delays. These requirements are determined by JEDEC/ Vendor specifications of the memory and DIMM devices.

The DRAM DQ input is to continuously return to the MC on the **dfi_db_train_resp** bus. Depending on the training algorithm, this DQ input might contain periods of invalid data. The MC is to use the $t_{phy\_db\_train\_resp}$ timing parameter. along with information about the command stream, for determining when to capture valid information from DFI.

A DFI 4.0 compliant PHY that implements DB training interface will need to tie the **dfi_db_train_en** input to 0 when the PHY interfaces with an older DFI version of the MC. A DFI 4.0 MC interfacing to an older DFI version PHY will need to disable the DB training interface.

4.16 PHY Control of the DFI Bus

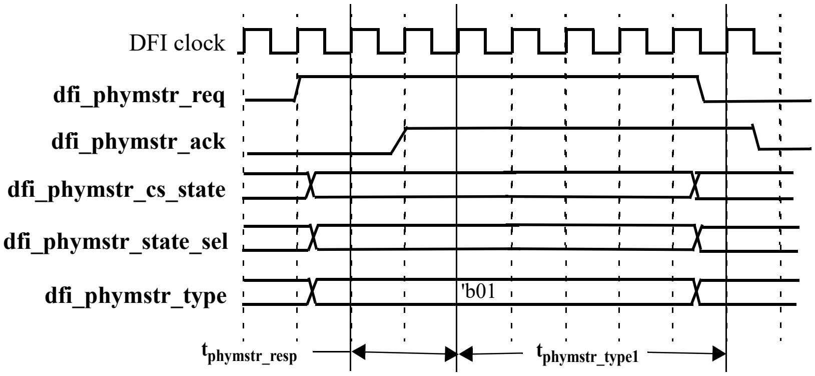
The PHY Master Interface enables the PHY to request control of the DFI bus. The procedure to use this interface is as follows:

1. The PHY asserts the **dfi_phymstr_req** signal along with the **dfi_phymstr_cs_state** and **dfi_phymstr_state_sel** signals to indicate to the MC the state of the memory required before handing off the bus. The PHY drives **dfi_phymstr_type** to indicate to the MC the length of time that it is requesting control.
2. If the **dfi_phymstr_cs_state** signal is cleared, the MC places the DRAM in the state specified in the **dfi_phymstr_state_sel** signal.
3. The MC asserts **dfi_phymstr_ack** within $t_{phymstr\_resp}$ cycles.
4. The MC de-asserts the **dfi_phymstr_ack** signal upon sampling the **dfi_phymstr_req** signal low.

   If the PHY does not de-assert the **dfi_phymstr_req** signal within $t_{phymstr\_typeX}$ cycles (where X was the value of the **dfi_phymstr_type** signal) following the assertion of the **dfi_phymstr_ack** signal, a DFI protocol violation occurs.
5. The PHY must return control of the DFI bus to the MC with DRAM in the identical state that it received it, including the state of memory. Also, all memory pages must be closed, and the refresh count must correspond with the MC refresh count.

Figure 89, “Master Interface Timing” illustrates the new master interface timing.

FIGURE 89. *Master Interface Timing*



If refreshes are required, the MC is responsible for maintaining all refresh timing requirements and sending out refresh commands. This is the only commands that may be sent on the DFI bus while the **dfi_phymstr_req** and **dfi_phymstr_ack** signals are asserted. If the MC passes the memory to the PHY in the self refresh state, no refreshes are issued during the PHY master mode.

To meet DRAM refresh timing requirements the MC and the PHY perform the following steps:

1. The MC performs following actions:
   a. The MC sends refresh commands as required on the DFI bus.
   b. The MC must maintain all refresh timing requirements relative to a refresh command that is issued during PHY master mode.
   c. The MC takes into account the PHY delay ($\mathbf{t_{phymstr\_rfsh}}$ parameter) for meeting all DRAM refresh timing.
2. The PHY performs following actions:

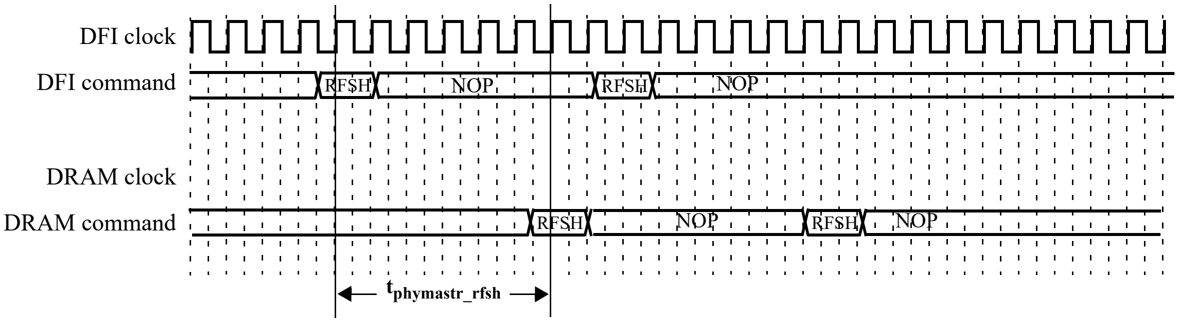
a. The PHY generates a refresh command to the DRAM within $\mathbf{t_{phymstr\_rfsh}}$ cycles.

b. The PHY ensures that all DRAM refresh timings are met.

c. The PHY ensures that all DRAM refresh timings have expired before it returns control to the MC

Figure 90, “Master Refresh Timing” illustrates the new master refresh timing.

FIGURE 90. *Master Refresh Timing*



4.17 DFI Disconnect Protocol

The DFI disconnect protocol can be used to break up a handshake between two DFI signals. There are eight affected interfaces. When the disconnect is issued, the user will specify if the disconnect will leave the system in a fully operational state (QOS), or if the system is no longer guaranteed to be operational (error condition). This information is conveyed through the **dfi_disconnect_error** signal. The **dfi_disconnect_error** signal must be valid on the same clock as the disconnect and remain unchanged until the disconnecting device completes the disconnection. The signal is not meaningful at any other time.

4.17.1 Update Interface

There are two handshakes in the update interface: controller-initiated updates and PHY-initiated updates. For both, when the request and acknowledge signals are asserted, only the PHY can disconnect the handshake. Although there are timing parameters to define the maximum handshake time ($\mathbf{t_{phyupd\_typeX}}$, $\mathbf{t_{ctrlupd\_min}}$), the delay might be unacceptably long in some cases.

To break either handshake, the MC will violate the protocol and de-assert the **dfi_ctrlupd_req** or the **dfi_phyupd_ack** signal. The de-assertion of either signal indicates to the PHY that the MC intends to disconnect the handshake. For a controller-initiated update disconnect, the MC must assume that the PHY did not complete the requested operation, and that a subsequent request should be scheduled. This requires meeting $\mathbf{t_{ctrlupd\_interval}}$ timing.

The **dfi_disconnect_error** signal defines if the disconnect is an error condition or a QOS event. The value of this signal when the disconnect occurs (the de-assertion of the **dfi_ctrlupd_req** or **dfi_phyupd_ack** signal) determines the expected system behavior:

- If the **dfi_disconnect_error** signal is de-asserted (QOS), the PHY must de-assert the corresponding signal (**dfi_ctrlupd_ack** or **dfi_phyupd_req**) within $\mathbf{t_{ctrlupd\_disconnect}}$ or $\mathbf{t_{phyupd\_disconnect}}$ clocks. At that point, the PHY must be fully operational.

- If the **dfi_disconnect_error** signal is asserted (error), the PHY must de-assert the corresponding signal (**dfi_ctrlupd_ack** or **dfi_phyupd_req**) within $\mathbf{t_{ctrlupd\_disconnect\_error}}$ or $\mathbf{t_{phyupd\_disconnect\_error}}$ clocks. The state of the PHY that follows the disconnect error condition is defined by the PHY and should be stated in the PHY specification. Ideally, the PHY disconnects on a boundary that preserves the most memory service possible.

NOTE: In previous versions of the DFI interface, the signaling that is defined for disconnect is an illegal operation.

Figure 91, “PHY Update Request, QOS Disconnect Protocol” and Figure 92, “PHY Update Request, Error Disconnect Protocol” illustrate the new disconnect signaling protocol.

FIGURE 91. *PHY Update Request, QOS Disconnect Protocol*

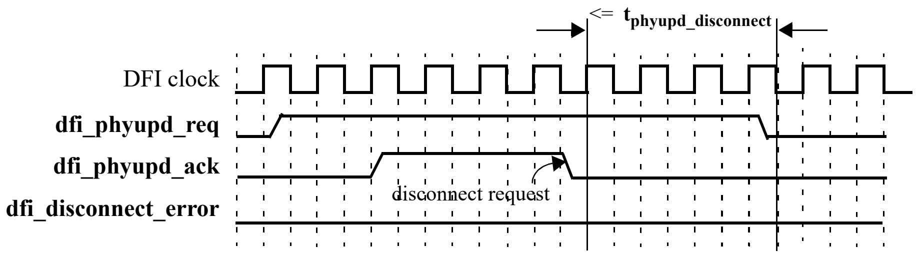
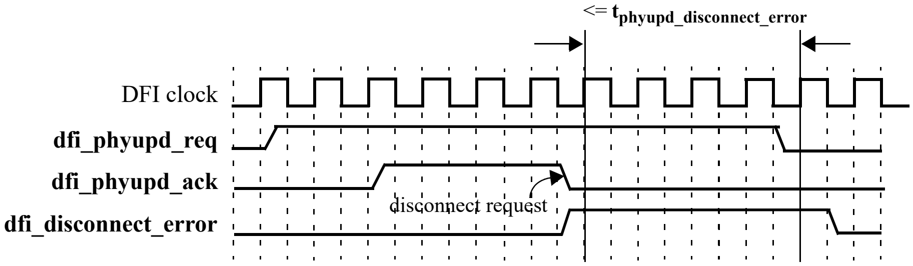


FIGURE 92. *PHY Update Request, Error Disconnect Protocol*



4.17.2 PHY Master Interface

The PHY Master Interface includes a request and acknowledge signal, with the PHY driving the request signal to the MC and the MC responding with the acknowledge. Although there are timing parameters to define the maximum handshake time ($\mathbf{t_{phymstr\_typeX}}$), the delay might be unacceptably long in some cases.

To break the handshake, the MC will de-assert the **dfi_phymstr_ack** signal. The de-assertion of this signal indicates to the PHY that the MC intends to disconnect the handshake.

The **dfi_disconnect_error** signal defines if the disconnect is an error condition or a QOS event. The value of this signal when the disconnect occurs (the de-assertion of the **dfi_phymstr_ack** signal) determines the expected system behavior:

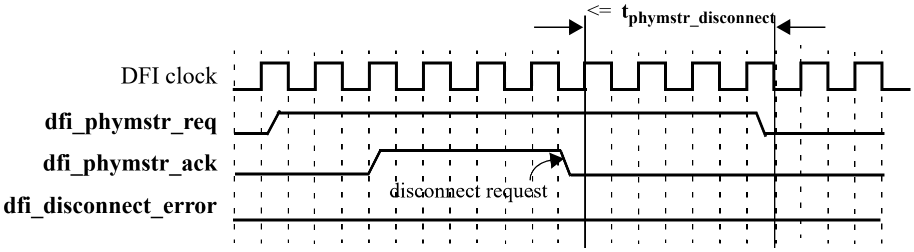
- If the **dfi_disconnect_error** signal is de-asserted (QOS), the PHY must de-assert the **dfi_phymstr_req** signal within $\mathbf{t_{phymstr\_disconnect}}$ clocks. At that point, the PHY must be fully operational.

- If the **dfi_disconnect_error** signal is asserted (error), the PHY must de-assert the **dfi_phymstr_req** signal within $\mathbf{t_{phymstr\_disconnect\_error}}$ clocks. The state of the PHY that follows the disconnect error condition is defined by the PHY and should be stated in the PHY specification. Ideally, the PHY disconnects on a boundary that preserves the most memory service possible.

NOTE: In previous versions of the DFI interface, the signaling that is defined for disconnect is an illegal operation.

Figure 93, "PHY Master Interface, QOS Disconnect Protocol" illustrates the new disconnect signaling protocol.

FIGURE 93. *PHY Master Interface, QOS Disconnect Protocol*



4.17.3 DFI Training Interface

The DFI training interface includes 5 handshakes: CA training, read data eye training, read gate training,write leveling and write DQ training. When the MC asserts a bit (or bits) of a training enable signal (**dfi_calvl_en**, **dfi_rdlvl_en**, **dfi_rdlvl_gate_en**, **dfi_wdqlvl_en** or **dfi_wrlvl_en**), only the PHY can disconnect the handshake by driving the associated response (**dfi_calvl_resp**, **dfi_rdlvl_resp**, **dfi_wdqlvl_resp** or **dfi_wrlvl_resp**).

To break any of these handshakes, the MC will violate the protocol and de-assert the associated training enable signal. The de-assertion of any of these signals indicates to the PHY that the MC intends to disconnect the handshake. For the disconnect, the MC must assume that the PHY did not complete the requested training operation, and that a subsequent training request should be scheduled.

The PHY will also respond to the disconnect by asserting a done response on the corresponding training done signal. Only a training complete response of 'b11 indicates that training has terminated. If one of these training operations responds with a 'b01, then training should continue. This provides a way for the PHY to report that it is ready for regular operation.

The **dfi_disconnect_error** signal defines if the disconnect is an error condition or a QOS event. The value of this signal when the disconnect occurs (the de-assertion of the set bit (or bits) of the **dfi_calvl_en**, **dfi_rdlvl_en**, **dfi_rdlvl_gate_en**, **dfi_wdqlvl_en** or **dfi_wrlvl_en** signals) determines the expected system behavior:

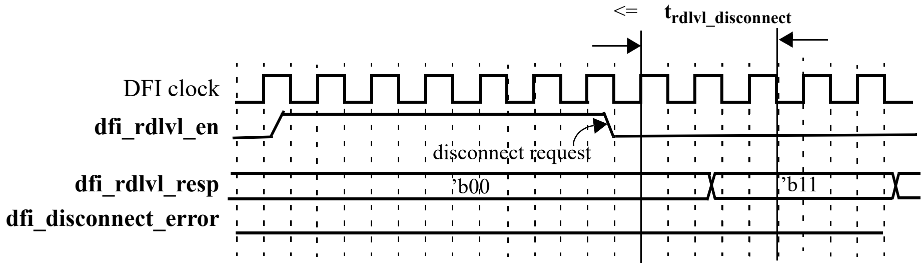
- If the **dfi_disconnect_error** signal is de-asserted (QOS), the PHY must assert the corresponding response signal (**dfi_calvl_resp**, **dfi_rdlvl_resp**, **dfi_wdqlvl_resp** or **dfi_wrlvl_resp**) within the disconnect time ($\mathbf{t_{calvl\_disconnect}}$, $\mathbf{t_{rdlvl\_disconnect}}$, $\mathbf{t_{rdlvl\_gate\_disconnect}}$, $\mathbf{t_{wdqlvl\_disconnect}}$ or $\mathbf{t_{wrlvl\_disconnect}}$). At that point, the PHY must be fully operational. The state of system after the disconnect should be the same as if the state would be if the disconnect had not occurred. For CA and write training, the response signal should report 'b11 to indicate that training has terminated.
- If the **dfi_disconnect_error** signal is asserted (error), the PHY must de-assert the corresponding signal (**dfi_calvl_resp**, **dfi_rdlvl_resp**, **dfi_wdqlvl_resp** or **dfi_wrlvl_resp**) within the disconnect time ($\mathbf{t_{calvl\_disconnect\_error}}$, $\mathbf{t_{rdlvl\_disconnect\_error}}$, $\mathbf{t_{rdlvl\_gate\_disconnect\_error}}$, $\mathbf{t_{wdqlvl\_disconnect\_error}}$ or $\mathbf{t_{wrlvl\_disconnect\_error}}$). The

state of the PHY that follows the disconnect error condition is defined by the PHY and should be stated in the PHY specification. Ideally, the PHY disconnects on a boundary that preserves the most memory service possible.

NOTE: In previous versions of the DFI interface, the signaling that is defined for disconnect is an illegal operation.

Figure 94, “Read Data Eye Training Request, QOS Disconnect Protocol” illustrates the new disconnect signaling protocol.

FIGURE 94. *Read Data Eye Training Request, QOS Disconnect Protocol*



4.18 Use of the Geardown Mode

DDR4 memories support a geardown mode in which the command’s alignment with the memory clock is altered and the command should be aligned to the command's first clock rising edge to center on the command's second clock rising edge. The DFI spec includes a **dfi_geardown_en** signal to support this behavior.

To enter geardown mode on DFI, the MC enters the DRAM geardown mode. When the $\mathbf{t_{ctrl\_delay}}$ time elapses from the defined sync pulse, the MC will assert the **dfi_geardown_en** signal. The CA bus signals must be maintained in an idle or de-select state or both for the time defined in the $\mathbf{t_{geardown\_delay}}$ parameter. The PHY time makes adjustments in CA timing.

When **dfi_geardown_en** = ’b1, the MC must drive commands for two DFI PHY clocks - for two DFI clocks in a matched frequency configuration, or for two phases in a Frequency Ratio configuration. These commands must be aligned to the proper clock edge as defined by the sync pulse.

Figure 95, “Self Refresh Exit” shows that the MC must de-assert the **dfi_geardown_en** signal at least $\mathbf{t_{geardown\_delay}}$ cycles before sending a self-refresh exit command.

FIGURE 95. *Self Refresh Exit*

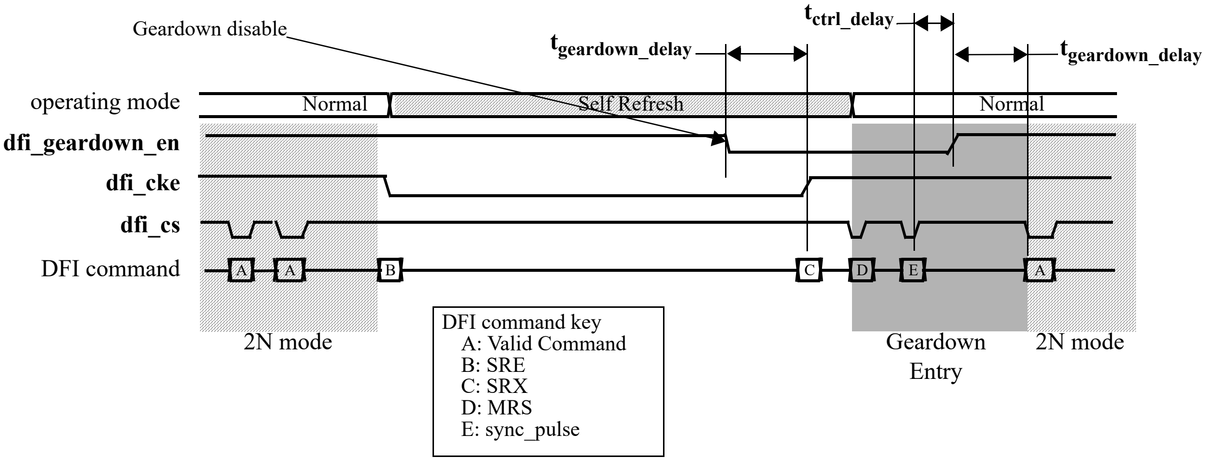
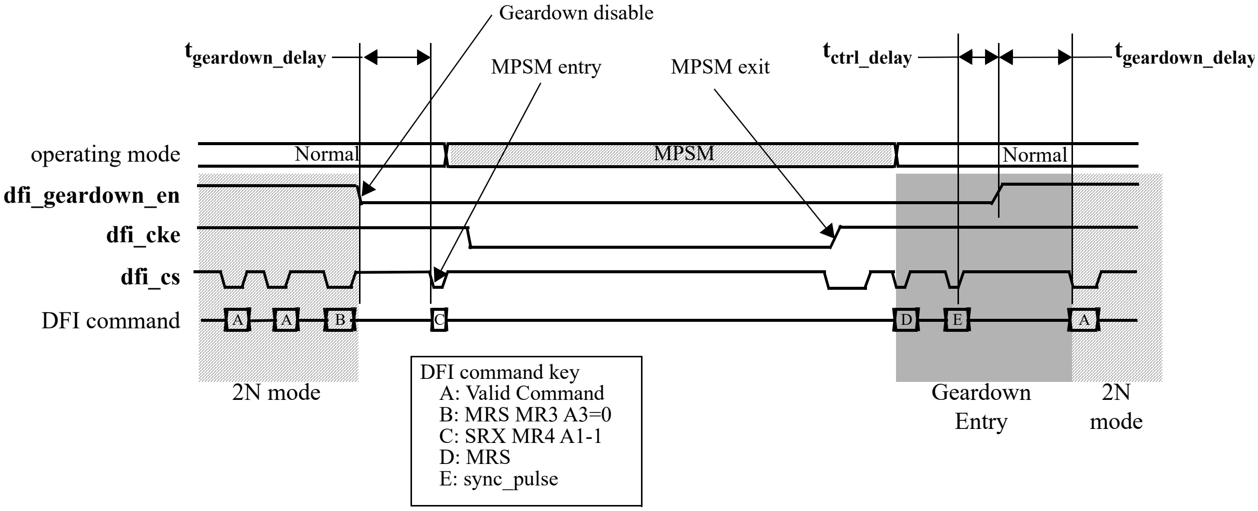


Figure 96, "MPSM Entry and Exit" shows that the MC must de-assert the **dfi_geardown_en** signal at least $\mathbf{t}_{\mathbf{geardown\_delay}}$ cycles before sending a Maximum Power Savings Mode (MPSM) entry MRS command.

FIGURE 96. *MPSM Entry and Exit*



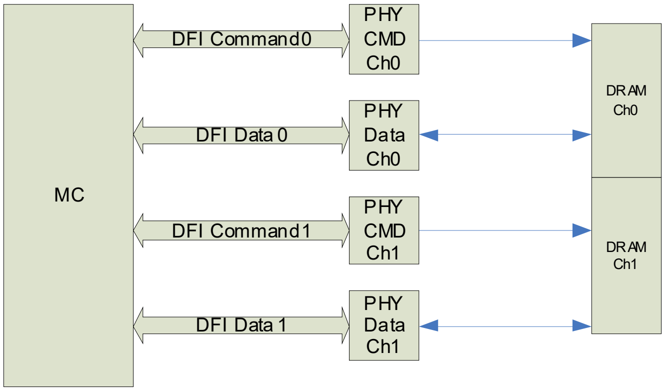
4.19 Channel Operation for LPDDR4 memories

LPDDR4 memories can operate with independent or combine channels. If used in combined mode, the setting of the $\mathbf{phy}_{\mathbf{channel\_en}}$ programmable parameter indicates whether the PHY connects to a single channel or both channels when it operates in combined mode.

4.19.1 Independent Operation

Figure 97, “LPDDR4 Independent Channels” illustrates how an LPDDR4 device could be connected for independent channels. The MC can represent one or two memory controllers. The memory reset is shared, so the MC must ensure that all DFI channels change their **dfi_reset_n** signals to the PHY simultaneously. The system can use any reset signal.

FIGURE 97. *LPDDR4 Independent Channels*



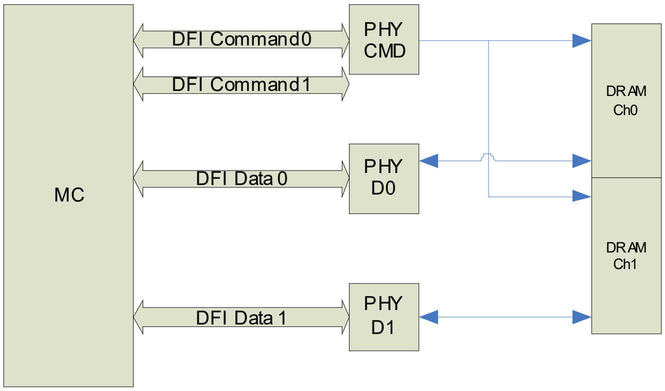
4.19.2 Combined Operation

For combined operation, the PHY will be connected to two channels of read and write data interfaces. All signals for both channels must be present and operational as defined for the interface. For all other interfaces (control, update, status, training, low power) the PHY can connect to one channel's interfaces or to both channels’ interfaces. The connectivity is defined by a programmable parameter, $\mathbf{phy}_{\mathbf{channel\_en}}$. If the PHY is connected to a single interface, the two channel lock-step operation is easily maintained. Furthermore, if the PHY connects to both channels, but always generates the same responses to both channels (for example, if the same signal is fanned out to both channels), the lock-step operation is maintained. However, if the PHY connects to both channels, and the PHY driven signals on the two channels operate independently (and might be different), special handling is required for the various interfaces.

For combined operation with the PHY connected to both command channels, Figure 97 would still apply. The command signals from the MC to the PHY are identical for both channels. The signals from the PHY to the MC may or may not be identical.

For combined operation with the PHY connected to a single command channel, Figure 98 applies. The MC can represent one or two memory controllers.

FIGURE 98. *LPDDR4 Combined Channels*



For combined operations, the control interface, read data interface control signals, and write data interface control signals that the MC drives operate in lock-step. In general, other MC-driven signals also operate in lock-step, unless defined otherwise. Some signals, such as **dfi_freq_ratio**, must drive the same value. Other interfaces require some special considerations in a combined configuration when using the interfaces from both channels, if the channels are not required to be driven identically. Using both channels for some interfaces, and only a single channel for other interfaces, is permitted. Unused interfaces should be tied inactive.

NOTE: A channel can be disabled, but a single interface within a channel cannot be disabled.

4.19.2.1 Update Interface

The update interface can be driven uniquely by the two channels. The controller update must be driven identically to both channels. For example, if the PHY on one channel asserts the acknowledge and the other channel does not respond, or if the acknowledges from the channels are de-asserted at different times, both DFI channels remain idle until the update is completed on both channels. If the PHY update request is asserted differently on the two channels, when the MC acknowledges, both channels must remain idle. When the MC acknowledges the PHY update request, the acknowledge will be asserted to one or both channels, depending on whether one or both requests are asserted. When the acknowledge is asserted, no additional PHY update requests can be serviced until the update in progress completes.

4.19.2.2 Status Interface

For the status interface, initialization does not complete until both channels assert **dfi_init_complete**. For frequency change, the MC requests a frequency change in lock-step by asserting **dfi_init_start** to both channels. If one channel acknowledges and the other channel does not acknowledge, the MC terminates the frequency change on the channel that acknowledged by de-asserting **dfi_init_start** without changing the clock frequency. Both channels must wait for the completion of the frequency change handshake on the acknowledging channel before running additional commands. The **dfi_freq_ratio** signal must be identical for both channels.

4.19.2.3 DFI Training Interface

The training interface will work as currently defined. Training is done on a per-slice basis, and slices should align within channels. In addition, independent slice training supports dual channel operation.

4.19.2.4 Low Power Interface

The low power interface can receive unique responses from the different channels. If one channel acknowledges a low power request, and the other channel does not, both channels must maintain the same operation and adhere to the wakeup time that is associated with the channel that acknowledged the low power request. The MC can assert additional low power requests to the channel that is not in low power state. However, the MC cannot de-assert the acknowledged request when the other channel request is still pending. If the two channels are not in the same low power state, the larger wakeup time applies to both interfaces.

NOTE: The timing parameters for all interfaces operating in lock-step must be programmed identically.

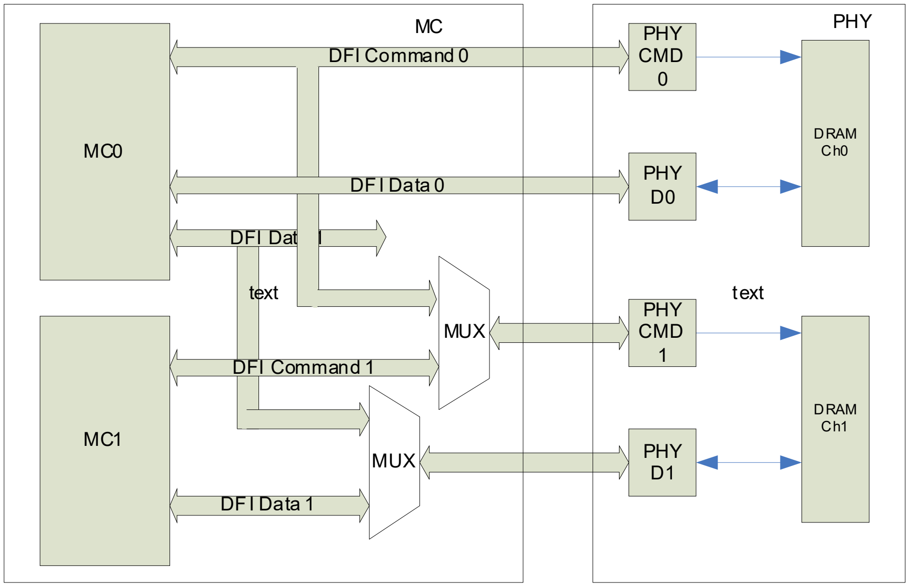
4.19.3 Multi-Configuration Support

It might be desirable for MCs and PHYs to be able to operate in both an independent and combined mode of operation. For interoperability, it would be preferable to define the device, MC or PHY, that is responsible for multiplexing the bus when supporting both independent and combined operations. The recommendation is placing the burden for multiplexing within the MC rather than the PHY.

NOTE: It is also possible to achieve the same system by using external multiplexing on the DFI interface.

Figure 99, "Example of LPDDR4 Multi-Configuration" illustrates the MC multiplexing the DFI bus for operating in both modes. Combined operation means a single MC (MC0), and independent operation means two MCs (MC0 and MC1). The MC block internally multiplexes the command and data for MC0 and MC1, and no multiplexing of the command and data is required within the PHY.

FIGURE 99. *Example of LPDDR4 Multi-Configuration*



When operating in combined mode, the DFI channels operate in lock-step. In this mode, the PHY can either connect to the interfaces for both channels (as Figure 97 and Figure 98 show), or connect to a single interface. When connecting to a single interface, the other channel can be disabled. When using only a single channel, channel 0 should be used and channel 1 disabled.

5.0 Signal Timing

The DFI specification does not specify timing values for signaling between the MC and the PHY. The only requirement is that a DFI clock must exist, and all DFI-related signals must be driven by registers referenced to the rising edge of the DFI clock. There are no restrictions on how these signals are received, nor any rules on the source of the DFI clock. Compatibility between the MC and the PHY at given frequencies is dependent on the specification of both the output timing for signals driven, and the setup and hold requirements for reception of these signals on the DFI.

However, the DFI signals are categorized into three signal groups that place restrictions on how signals may be driven and captured by DFI devices. All DFI signals may be driven from the DFI clock and captured on the following rising edge of the DFI clock. However, some signals may allow less restrictive timing which may alleviate timing restrictions within the design.

For frequency ratio systems, the PHY must send and receive DFI signals with respect to the rising edge of the DFI clock, even if the signals are driven and captured by registers clocked by the higher frequency DFI PHY clock.

There are three signal groups:

1. **Standard Signals**

   Standard signals contain timing critical information on a cycle-by-cycle basis, and therefore, must be sent and received on every DFI clock. The Control Interface signals fall into the standard signal category because the Control Interface signals must be cycle-accurate to properly communicate memory commands.

   Standard signals must be sent on the DFI clock and must be received on every DFI clock period for proper operation. All standard signals are required to meet setup and hold time to the DFI clock at the destination. Neither device should have a dependency on the source clock of the other device. The source and destination clock should always be assumed to be the DFI clock.

2. **State-Retaining Signals**

   State-retaining signals contain information that is not single cycle-critical because state-retaining signals retain a state change until either a signal acknowledge is received or a timing parameter has been satisfied. The update interface signals fall into the standard signal category because all signal state changes are defined in terms of signal responses and timing parameters.

   State-retaining signals may be sent and received in the same way on every DFI clock period. They may also be sent and/or received by a divided frequency clock, provided that the lower frequency clock is an even multiple (½, ¼, etc.) and is phase aligned to the DFI clock.

   All state-retaining signals are required to meet setup and hold time to the DFI clock at the destination regardless of whether the signal is generated from the DFI clock or from a lower frequency, phase-aligned clock source. Neither device should have a dependency on the source clock of the other device; the source and destination clock should always be assumed to be the DFI clock. If a lower frequency clock is used, the associated timing parameters must be set to appropriately account for the timing effects of using a lower frequency clock at either the source or destination.

The timing parameters are always defined in terms of the DFI clock regardless of the source and destination clock frequency.

3. **Timer-Based Signals**

Training signals such as the **dfi_freq_ratio** and **dfi_rdlvl_resp** are classified as timer-based signals because the training signals are not valid until the associated timing parameters are met.

Timer-based signals do not have a clock-edge dependency because timer-based signals are either not required to be valid until a timing parameter has been met, another signal has been asserted, or they are expected to be static during normal operation (static signals may be changed during idle times).

Timer-based signals do not have to meet setup and hold time to the DFI clock except on the cycle after meeting the associated timing parameter. Timer-based signals may also be changed during idle times in which case the setup and hold times are irrelevant. For timing analysis, timer-based signals may be treated as multi-cycle paths.

Table 47, “Signal Group Divisions” shows that each DFI signal is categorized into a signal group.

TABLE 47. *Signal Group Divisions*

Signal	Signal Group
dfi_act_n **(or dfi_act_n_pN)**	Standard
dfi_alert_n (or **dfi_alert_n_aN**)	Standard
dfi_address (or **dfi_address_pN**)	Standard
dfi_bank (or **dfi_bank_pN**)	Standard
dfi_bg **(or dfi_bg_pN)**	Standard
dfi_cas_n (or **dfi_cs_pN**)	Standard
dfi_cid **(or dfi_cid_pN)**	Standard
dfi_cke (or **dfi_cke_pN**)	Standard
dfi_cs (or **dfi_cs_pN**)	Standard
dfi_ctrlupd_ack	State-Retaining
dfi_ctrlupd_req	State-Retaining
dfi_dram_clk_disable	Standard
dfi_error	Standard
dfi_error_info	Standard
dfi_freq_ratio	Timer-Based
dfi_init_complete	State-Retaining
dfi_init_start	State-Retaining
dfi_lp_ack	State-Retaining

TABLE 47. *Signal Group Divisions (Continued)*

Signal	Signal Group
dfi_lp_ctrl_req	State-Retaining
dfi_lp_data_req	State-Retaining
dfi_lp_wakeup	State-Retaining
dfi_lvl_pattern	State-Retaining
dfi_lvl_periodic	State-Retaining
dfi_odt (or **dfi_odt_pN**)	Standard
dfi_alert_n	Standard
dfi_parity_in (or **dfi_parity_in_pN**)	Standard
dfi_phy_rdlvl_cs	State-Retaining
dfi_phy_rdlvl_gate_cs	State-Retaining
dfi_phyupd_ack	State-Retaining
dfi_phyupd_req	State-Retaining
dfi_phyupd_type	State-Retaining
dfi_phy_wrlvl_cs	State-Retaining
dfi_ras_n (or **dfi_ras_n_pN**)	Standard
dfi_rddata (or **dfi_rddata_wN**)	Standard
dfi_rddata_cs_pN	State-Retaining
dfi_rddata_dbi_wN	Standard
dfi_rddata_dnv (or **dfi_rddata_dnv_wN**)	Standard
dfi_rddata_en (or **dfi_rddata_en_pN**)	Standard
dfi_rddata_valid (or **dfi_rddata_valid_wN**)	Standard
dfi_rdlvl_en	State-Retaining
dfi_rdlvl_gate_en	State-Retaining
dfi_rdlvl_gate_req	State-Retaining
dfi_rdlvl_req	State-Retaining
dfi_rdlvl_resp	Timer-Based
dfi_reset_n (or **dfi_reset_n_pN**)	Standard
dfi_we_n (or **dfi_we_n_pN**)	Standard

TABLE 47. *Signal Group Divisions (Continued)*

Signal	Signal Group
dfi_wrdata (or **dfi_wrdata_pN**)	Standard
dfi_wrdata_cs_pN	Standard
dfi_wrdata_en (or **dfi_wrdata_en_pN**)	Standard
dfi_wrdata_mask (or **dfi_wrdata_mask_pN**)	Standard
dfi_wrlvl_en	State-Retaining
dfi_wrlvl_req	State-Retaining
dfi_wrlvl_resp	Timer-Based
dfi_wrlvl_strobe	Standard

6.0 Glossary

TABLE 48. *Glossary of Terms*

Term	Definition
CA Bus	JEDEC defines the CA bus as a bus containing command, address and bank/row buffer information. The CA bus operates at double data rate for LPDDR2 and LPDDR3 systems, and single data rate for LPDDR4 systems.
CA Training	A mechanism provided by the LPDDR3 and LPDDR4 DRAMs to optimize the CA timing relative to the memory clock. CA training is specific to LPDDR3 and LPDDR4 systems.
Column Selection (CAS)	Column address strobe.
Command Address (CA) Signals	Includes the following control interface signals: **dfi_act_n, dfi_address**, **dfi_bank**, **dfi_bg, dfi_ras_n, dfi_cas_n**, and **dfi_we_n.** The DRAM class determines the signals applicable to a specific system.
Command Bus	Includes the following control interface signals: **dfi_act_n, dfi_cas_n**, **dfi_ras_n** and **dfi_we_n.** The DRAM class determines the signals applicable to a specific system.
Data Bus Inversion (DBI)	A feature of the DRAM that allows read and write data to be inverted selectively before the data is transferred across the DDR data bus.
DFI Address Width	The number of address bits on the DFI interface. This is generally the same number of bits as the number of address bits on the DRAM device.
DFI Alert Width	The width of the alert signal on the DFI interface. Typically the PHY would drive an alert signal per slice and the alert is typically 1 bit.
DFI Bank Group Width	The number of bank group bits on the DFI interface. This is generally the same number of bits as the number of BG bits on the DRAM device.
DFI Bank Width	The number of bank bits on the DFI interface. This is generally the same number of bits as the number of bank pins on the DRAM device.
DFI CA Training MC I/F Width	The CA training MC interface width on the DFI interface. This is typically one bit per PHY command slice and typically one command slice.
DFI CA Training PHY I/F Width	The CA training PHY interface width on the DFI interface. This is typically one bit per PHY command slice and typically one command slice.
DFI Chip ID Width	The CID signals are used with 3D stacks. For DDR4, the MAX width is 3 bits (stack of 8). There are also stacks with 1 bit for CID (stack of 2).
DFI Chip Select Width	The number of chip select bits on the DFI interface. This is generally the same number of bits as the number of chip select pins on the DRAM bus. Rank width and chip select width are equivalent.
DFI CKE Width	Defines the width of the **dfi_cke** signal. The MC and the PHY should support the same number of bits on this signal. The MC must detect CKE signals that are valid. It could send more signals than the PHY can use, transmit, or both. However, whatever is dropped should not be asserted by the MC.
DFI clock cycle	The DFI clock has the same phase and frequency as the MC clock.
DFI clock frequency	Defines the clock frequency of the MC.
DFI Control Width	The number of bits required to control the DRAMs, usually a single bit.
DFI Data Enable Width	The width of the datapath enable signals on the DFI interface. For PHYs with an 8-bit slice, this is generally 1/16th of the DFI Data Width to provide a single enable bit per memory data slice, but may be 1/4, 1/8, 1/32, or any other ratio. Bit zero corresponds to the lowest segment.

TABLE 48. *Glossary of Terms (Continued)*

Term	Definition
DFI Data Slice Count	The number of data slices in the PHY.
DFI Data Width	The width of the datapath on the DFI interface. This is generally twice the DRAM data width.
DFI Data Word	One phase of read or write data passed between the MC and the PHY. A DFI data word is twice the width of the bus between the DRAM and the PHY and corresponds to a single memory word transfer across the DFI bus.
DFI DBI Width	The DBI width on the DFI interface. Typically DFI Data Width/8. See the description of **dfi_wrdata_mask** for more details.
DFI DRAM Clk Disable Width	Defines the width of the **dfi_dram_clk_disable** signal.
DFI Error Width	Typically, 1 bit per data slice plus 1 bit for control. The PHY may implement **dfi_error** as a single bit or any other width not exceeding the sum of the data slices + 1 bit for control. The MC should accept 1 bit per instance as defined by the sum of data slices plus 1 bit for control.
DFI Leveling PHY I/F Width	The leveling PHY interface width on the DFI interface. Typically one bit per PHY data and command slice that support training.
DFI ODT Width	Defines the width of the **dfi_odt** signal. The MC must detect ODT signals that are valid. It could send more signals than the PHY can use, transmit, or both. However, whatever is dropped should not be asserted by the MC.
DFI PHY Clock Cycle	The DFI PHY clock cycle is the DFI clock cycle divided by the frequency ratio. For a 1:1 frequency ratio, the DFI clock cycle and the DFI PHY clock cycle are equivalent.
DFI PHY Clock Frequency	Defines the period of the clock frequency of the PHY. For matched systems, this is the same as the period of the DFI clock frequency. For frequency ratio systems, the period of the PHY DFI clock frequency must be 1/2 or 1/4 the period of the DFI clock frequency. These clocks must be aligned in phase.
DFI Physical Rank Width	Defines the number of chip selects. For 3DS, this is referred to as the “Physical Rank.”
DFI Read Data Valid Width	The width of the datapath valid signals on the DFI interface. Equivalent to the number of PHY data slices, the same width definition as the **dfi_rddata_en** signal. Bit zero corresponds to the lowest segment.
DFI Read Leveling MC I/F Width	The read leveling MC interface width on the DFI interface. Typically one bit per PHY data slice.
DFI Read Leveling PHY I/F Width	The read leveling PHY interface width on the DFI interface. Typically one bit per PHY data slice.
DFI Read Leveling Response Width	The read leveling response width on the DFI interface. Typically one bit per PHY data slice.
DFI Read Training PHY I/F Width	The number of bits used to control the read training interface from the PHY perspective. The PHY may drive a signal from each memory data slice or combine the signals into a single signal.
DFI Read Training Response Width	The number of bits used to communicate read training status to the MC. The PHY Read Training response may be 2 bits per memory data slice or 2 bits per bit on the memory data bus. If this width is the same width as the memory data bus, gate training information should be returned on the lowest bit of each data slice.
DFI Reset Width	Defines the width of the **dfi_reset_n** signal. The MC and the PHY should support the same number of bits on this signal.

TABLE 48. *Glossary of Terms (Continued)*

Term	Definition
DFI Training Interface	Utilized when the DFI training mode is enabled. The DFI training interface has five operations: CA training, gate training, read data eye training, write leveling and write DQ training. Gate training and read data eye training are referred to collectively as "read training". The training operation used is determined by the memory type and whether the system is using read signals or write signals.
DFI Write DQ Training Response Width	The number of bits used to communicate write DQ training status to the MC. The PHY should drive 2 bits per memory data slice.
DFI Write DQ Training MC I/F Width	The number of bits used to control the write DQ training interface from the MC perspective. The MC write leveling signals are generally fanned out such that a copy of the signal can be sent to each PHY memory data slice.
DFI Write DQ Training PHY I/F Width	The number of bits used to control the write DQ training interface from the PHY perspective. The PHY may drive a signal from each memory data slice or combine the signals into a single signal.
DFI Write Leveling MC I/F Width	The number of bits used to control the write leveling interface from the MC perspective. The MC write leveling signals are generally fanned out such that a copy of the signal can be sent to each PHY memory data slice.
DFI Write Leveling PHY I/F Width	The number of bits used to control the write leveling interface from the PHY perspective. The PHY may drive a signal from each memory data slice or combine the signals into a single signal.
DFI Write Leveling Response Width	The number of bits used to communicate write leveling status to the MC. The PHY should drive a single bit per memory data slice.
dfi_{rw_length}	The value of the total number of DFI clocks required to transfer one DFI read or write command worth of data. • For a matched frequency system: $\mathbf{dfi}_{\mathbf{rw\_length}}$ would typically equal (burst length/2). • For a frequency ratio system: $\mathbf{dfi}_{\mathbf{rw\_length}}$ is defined in terms of DFI PHY clocks and would typically equal (burst length/2). Additional DFI clock (or DFI PHY clock) cycles must be added for the CRC data transfer.
DQ	The bi-directional bus that transfers read and write data to the DRAM.
DQS	The bi-directional data strobe bus transmitted to and from the DRAM.
Frequency Ratio	In a frequency ratio system, the MC and the PHY operate at a common frequency ratio of 1:2 or 1:4; the PHY must be able to accept a command on any and all phases. The frequency ratio depends on the relationship of the reference clocks for the MC and the PHY. Phase-specific signals with a suffix of "_pN" with the phase number N (e.g., **dfi_wrdata_pN**) replace the matched frequency control, write data, read data and status interface enable signals. Phase-specific signals allow the MC to drive multiple commands in a single clock cycle. Data word-specific signals with a suffix of "_wN" with the DFI data word number N (e.g., **dfi_rddata_wN**) replace the matched frequency read data interface signals to distinguish how memory words are transferred across the DFI bus. Variable pulse width-specific signals with a suffix of "aN", with the PHY clock cycle N (e.g., **dfi_alert_n_aN)**, replace the matched frequency status interface signals to maintain the pulse width during transmission of error signals from the memory system to the PHY. For all signal types, the suffix for phase 0/data word 0/clock cycle 0 is optional. For more information on frequency ratios, refer to Section 4.9, "Frequency Ratios Across the DFI".

TABLE 48. *Glossary of Terms (Continued)*

Term	Definition
Gate Training	An operation to locate the DQS gate in the read data preamble.
Idle	The DFI bus is considered idle when the control interface is not sending any commands and all read and write data has transferred on the DFI bus, reached its destination (DRAM or MC), and the write data transfer has completed on the DRAM bus. No pages are open.
Matched Frequency	In a matched frequency system, the MC and the PHY operate at a common frequency ratio of 1:1.
MC	DDR Memory Controller logic
PHY	DDR Physical Interface logic
PHY Data Word	One phase of read or write data passed between the PHY and the DRAM. A PHY data word is the width of the bus between the PHY and the DRAM and corresponds to a single memory word transfer across the DFI bus. A PHY data word is half the width of a DFI data word.
Read data eye training	An operation utilizing pattern registers of the DRAM to center the DQS in the read data eye.
Read training	May refer to either gate training, data eye training, or both.
Row Selection (RAS)	Row address strobe.
Unit interval (UI)	Half of a data word; the number of DRAM data words in one DRAM burst. NOTE: With DDR DRAM devices, the number of DRAM data words is 2x the number of DFI data words.
Variable pulse width-specific signals	Differentiated with a suffix of “aN”, with the PHY clock cycle N (e.g., **dfi_alert_n_aN)**, variable pulse width-specific signals replace status interface matched frequency signals to maintain the pulse width during transmission of error signals from the memory system to the PHY.
Write DQ training	An operation to center the DQS in the write data eye at the DRAM.
Write leveling	A feature of the DRAM used to adjust the relationship between the DQS and the DRAM clock.