DFI

DDR PHY Interface

DFI 5.1 Specification

MAY 21, 2021

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.
5.0	27 Apr 2018	Corrected formatting, spelling and content errors from 4.0 version
5.1	26 Mar 2021	Incorporated new proposals for DFI 5.1, standardized look and feel of all figures, added new signals and updated tables as a result.

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

AMD ARM Broadcom Cadence Intel Samsung ST Synopsys Uniquify

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 command information and data across the DFI and between the DDR memory controller (MC) and the DDR PHY (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, DDR5, DDR5 LRDIMM, DDR5 RDIMM, LPDDR1, LPDDR2, LPDDR3, LPDDR4 and LPDDR5 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
Command	Required to drive the address and command 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.
PHY Master	Used to allow the PHY to control the DFI bus.
Disconnect Protocol	Allows an ongoing handshake to be broken.
Error	Used to communicate error information from the PHY to the MC.
2N Mode	Uses the 2N function (also referred to as the geardown mode) of the DRAM
Low Power Control	Allows the PHY to enter power-saving modes.
MC to PHY Message	Used to send defined messages from the MC to the PHY.
WCK Control	Controls the WCK on, off and synchronization timing.

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 49, “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 4, “DFI Signal Requirements”.

2.1 Clocking

The DFI bus does not include a clock signal; The DFI clock is generally assumed to be the same clock as the MC clock. The MC may operate at the same or different clock frequency than the memory. If the DFI clock and the memory clock are the same frequency, the system is defined as a matched frequency system. If the MC operates at a lower frequency, either 1:2 or 1:4 ratio to the memory clock, the system is defined as a frequency ratio system. For DRAMs utilizing a single memory clock, the clock ratio applies to the ratio between the DFI clock and the memory clock. For DRAMs utilizing a separate command and data clock, the clock ratio applies to the ratio between the DFI clock and the data clock; the DFI clock and the command clock operate at the same frequency.

DFI defines three clock domains - the control clock domain, the command clock domain, and the data clock domain. For a matched frequency system, these clock domains operate at the same frequency. For a frequency ratio system with a single memory clock, the control clock domain operates at the DFI clock frequency, the command and data clock domains operate at the higher clock ratio; generally referred to as the DFI PHY clock frequency. For a frequency ratio system with separate command and data clock frequencies, the control clock domain and the command clock domain operate at the DFI clock frequency and the data clock domain operates at the higher clock ratio. Timing parameters associated with each clock domain are also defined in reference to the corresponding clock domain.

Table 2, “Signals by Clock Domain” correlates each DFI signal with its clock domain.

TABLE 2. *Signals by Clock Domain*

Signal	Interface	Clock Domain
dfi_ctrlupd_ack	Update	DFI control clock
dfi_ctrlupd_req		
dfi_phyupd_ack		
dfi_phyupd_req		
dfi_phyupd_type		
dfi_freq_fsp	Status	
dfi_freq_ratio		
dfi_frequency		
dfi_init_complete		
dfi_init_start		
dfi_phymstr_ack	PHY Master	
dfi_phymstr_cs_state		
dfi_phymstr_req		
dfi_phymstr_state_sel		
dfi_phymstr_type		
dfi_disconnect_error	Disconnect Protocol	
dfi_error	Error	
dfi_error_info		
dfi_lp_ctrl_ack	Low Power Control	
dfi_lp_ctrl_req		
dfi_lp_ctrl_wakeup		
dfi_lp_data_ack		
dfi_lp_data_req		
dfi_lp_data_wakeup		
dfi_ctrlmsg	MC to PHY Message	
dfi_ctrlmsg_ack		
dfi_ctrlmsg_data		
dfi_ctrlmsg_req		

TABLE 2. *Signals by Clock Domain*

Signal	Interface	Clock Domain
dfi_2n_mode	Command	DFI command clock
dfi_act_n		
dfi_address		
dfi_alert_n		
dfi_bank		
dfi_bg		
dfi_cas_n		
dfi_cid		
dfi_cke		
dfi_cs		
dfi_dram_clk_disable		
dfi_odt		
dfi_parity_in		
dfi_ras_n		
dfi_reset_n		
dfi_we_n		
dfi_wrdata	Write Data	DFI data clock
dfi_wrdata_cs		
dfi_wrdata_ecc		
dfi_wrdata_en		
dfi_wrdata_mask		
dfi_rddata	Read Data	
dfi_rddata_cs		
dfi_rddata_dbi		
dfi_rddata_dnv		
dfi_rddata_en		
dfi_rddata_valid		
dfi_wck_cs	WCK Control	
dfi_wck_en		
dfi_wck_toggle		

For LPDDR5, the DFI clock domain and DFI command clock domain operate at the DFI clock frequency and the DFI data clock domain is 2:1 or 4:1 ratio to DFI clock domain. For all other memories, the DFI clock domain is at DFI clock frequency and the Command and DFI data clock domains are at the same frequency and at a 1:1, 1:2, or 1:4 ratio to the DFI clock.

The MC and the PHY must operate at the same frequency ratio. The frequency ratio applies to the command and DFI data clock domain (PHY frequency ratio) or to the DFI data clock domain (data frequency ratio) only. DFI clock domain signals do not operate at a clock ratio, they are always DFI clock based. Refer to Table 2, “Signals by Clock Domain” for the signal clock domains.

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

For a matched frequency DFI command clock domain system, a single DFI command is issued per DFI clock as required for a single memory clock of DRAM command; for DDR commands this includes both edges. For a frequency ratio DFI command clock domain system, a DFI command is issued per phase where the number of command phases is equivalent to the clock ratio.

- Operating at a matched frequency system, the MC and the PHY operate with a 1:1 ratio.
- Operating at a frequency ratio, 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. For a PHY frequency ratio system, the frequency ratio is defined between the DFI clock and the memory clock. For a data frequency ratio system, the command frequency ratio is matched between the DFI clock and the memory command clock and the data frequency ratio is between the DFI clock and the memory data clock.
- For signals operating at a frequency ratio, 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 command (if PHY frequency ratio), write data, read data and WCK control 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 4.10, “Frequency Ratios Across the DFI”.

2.2 Optional Protocols

Optional protocols handle data bus inversion (DBI), link ECC, 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 4.6, “Data Bus Inversion”, Section 4.7.4, “Cyclic Redundancy Check”, Section 4.11, “Frequency Change”, Section 4.12, “CA Parity Signaling and CA Parity, CRC Errors”, Section 4.14, “Error Signaling”, Section 4.17, “Use of the 2N Mode” and Section 4.19, “Use of the WCK Control Interface”.

2.3 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.

2.3.1 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
- MC to PHY message

For specific signal requirements, refer to Table 4, “DFI Signal Requirements”.

2.3.2 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 3, “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 3, “Features by Memory Topology”.

TABLE 3. *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	DFI disconnect during training
	Registered DIMM	• All DDR3 discrete, unbuffered DIMM features • CA parity • CA parity errors

TABLE 3. *Features by Memory Topology*

Memory Class	Topologies	Applicable Features [a,b]
DDR4	Discrete, Unbuffered DIMM	• DFI disconnect during training • Read data bus inversion • Write data bus inversion • Write data CRC • Write data CRC errors • CA parity • CA parity errors • 2N mode
	Registered DIMM	All DDR4 discrete and unbuffered DIMM features
	Load Reduced DIMM	• All DDR4 discrete and unbuffered DIMM features
DDR5	Discrete, Unbuffered DIMM	• DFI disconnect during training • Read data bus inversion • Write data bus inversion • Read data CRC • Write data CRC • Write data CRC errors • CA parity • CA parity errors • 2N mode
	Registered DIMM	All DDR5 discrete and unbuffered DIMM features
	Load Reduced DIMM	• All DDR5 discrete and unbuffered DIMM features
LPDDR1	Any	No additional features beyond global features
LPDDR2	S2, S4	DFI disconnect during training
	NVM	• All LPDDR2 S2, S4 features • Data not valid
LPDDR3	Any	DFI disconnect during training
LPDDR4	Any	• Read data bus inversion • Write data bus inversion • Combined and multi-configuration channel support • DFI disconnect during training
LPDDR5	Any	• Read data bus inversion • Write data bus inversion • Combined and multi-configuration channel support • DFI disconnect during training • WCK Control

[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

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**
  - 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.

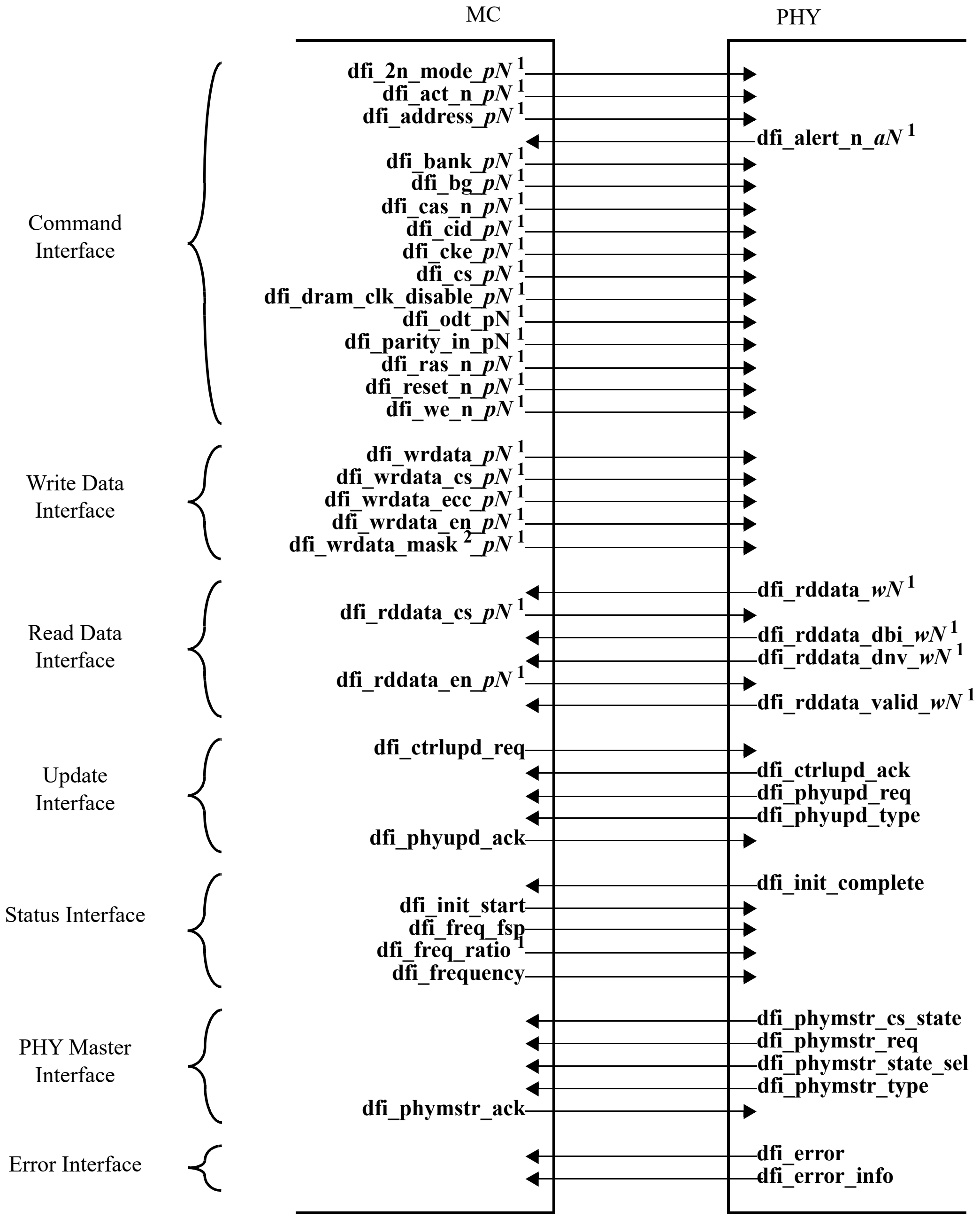
DRAM features supported but not explicitly covered by DFI are not included in the matrix. These include:

- Per DRAM addressability

2.3.3 DFI Signals

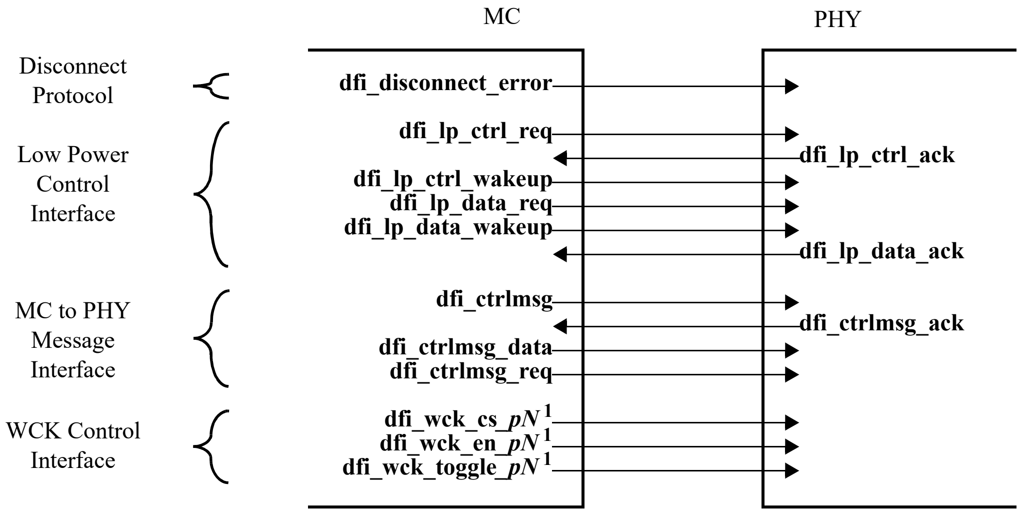
The DFI signals associated with each interface group, and the device originating the signal, are shown in Figure 1, “Block Diagram of Interface Signals: Command, Write Data, Read Data, Update, Status, PHY Master, Disconnect and Error” and Figure 2, “Block Diagram of Interface Signals: Low Power Control, MC to PHY Message and WCK Control”. The signal requirements and parameters associated with each signal are listed in Table 4, “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: Command, Write Data, Read Data, Update, Status, PHY Master, Disconnect and Error*



1. Optional suffix for frequency ratio systems.
2. Dual-function signal. In DDR4/LPDDR4 systems with write DBI enabled, the signal transforms from a mask to a write DBI signal.
Italicized text indicates that the phase/word/cycle is optional.

FIGURE 2. *Block Diagram of Interface Signals: Low Power Control, MC to PHY Message and WCK Control*



1. Optional suffix for frequency ratio systems.

2. Dual-function signal. In DDR4/LPDDR4 systems with write DBI enabled, the signal transforms from a mask to a write DBI signal.

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

To determine which signals are required for a specific configuration, review Table 4, “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., $\mathbf{t_{xxxxx\_xxxxx}}$). 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 4. *DFI Signal Requirements*

Command Interface			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_2n_mode_*pN*	$t_{2n\_mode\_delay}$	Optional for DDR4 DRAMs. Required for DDR5 DRAMs.	Optional for DDR4 DRAMs. Required for DDR5 DRAMs.
dfi_act_n_*pN*	t_{ctrl_delay}	Required for DDR4 and DDR5 DRAMs. [b]	Required for DDR4 and DDR5 DRAMs. [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]

TABLE 4. *DFI Signal Requirements*

dfi_alert_n_*aN*	$\mathbf{phy}_{\mathbf{crc\_mode}}$ $\mathbf{t}_{\mathbf{parin\_lat}}$ $\mathbf{t}_{\mathbf{phy\_crcmax\_lat}}$ $\mathbf{t}_{\mathbf{phy\_crcmin\_lat}}$ $\mathbf{t}_{\mathbf{phy\_paritylat}}$	Required for the following systems: • DDR3 RDIMM systems • DDR4 and DDR5 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 $\mathbf{phy}_{\mathbf{crc\_mode}}$ 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 and DDR5 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 $\mathbf{phy}_{\mathbf{crc\_mode}}$ 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_bank_*pN*	$\mathbf{t}_{\mathbf{ctrl\_delay}}$	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*	$\mathbf{t}_{\mathbf{ctrl\_delay}}$	Required for DDR4 DRAMs. [b]	Required for DDR4 DRAMs. [b]
dfi_cas_n_*pN*	$\mathbf{t}_{\mathbf{ctrl\_delay}}$	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*	$\mathbf{t}_{\mathbf{ctrl\_delay}}$	Required for DDR3, DDR4, and LPDDR5 3D stack devices. [b]	Required for DDR3, DDR4, and LPDDR5 3D stack devices. [b]
dfi_cke_*pN*	$\mathbf{t}_{\mathbf{ctrl\_delay}}$	Required for DDR1, DDR2, DDR3, DDR4, LPDDR1, LPDDR2, LPDDR3 and LPDDR4 DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR1, DDR2, DDR3, DDR4, LPDDR1, LPDDR2, LPDDR3 and LPDDR4 DRAMs. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]

TABLE 4. *DFI Signal Requirements*

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_dram_clk_disable_*pN*	$t_{dram\_clk\_disable}$ $t_{dram\_clk\_enable}$	Required for all DRAMs. Phased signaling is required for all DRAMs.	Required for all DRAMs. Phased signaling is required for all DRAMs.
dfi_odt_*pN*	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]
dfi_parity_in_*pN*	t_{parin_lat} $t_{phy\_paritylat}$	Required for the following systems: • DDR3 RDIMM systems • DDR4 and DDR5 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 and DDR5 systems that support CA parity. Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]
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, DDR5, LPDDR4 and LPDDR5 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]	Required for DDR3, DDR4, DDR5, LPDDR4 and LPDDR5 DRAMs. [b] Suffix (_pN) required for frequency ratio systems to replicate information across the phases. [a]

TABLE 4. *DFI Signal Requirements*

dfi_we_n_*pN*	$\mathbf{t_{ctrl\_delay}}$	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 Data Interface			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_wrdata_*pN*	$\mathbf{phy_{crc\_mode}}$ $\mathbf{t_{phy\_wrdata}}$ $\mathbf{t_{phy\_wrlat}}$	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*	$\mathbf{t_{phy\_wrcsgap}}$ $\mathbf{t_{phy\_wrcslat}}$	Required for all DRAMs if there is more than 1 chip select. 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_ecc_*pN*	$\mathbf{phy_{linkecc\_mode}}$	Required for LPDDR5 DRAMs when $\mathbf{phy_{linkecc\_mode}}$ = ’b1	Optional. Is a pass-through signal if the MC supports Link ECC, or is required if the $\mathbf{phy_{linkecc\_mode}}$ = ’b0 and the PHY is handling link ECC
dfi_wrdata_en_*pN*	$\mathbf{dfi_{rw\_length}}$ $\mathbf{phy_{crc\_mode}}$ $\mathbf{t_{phy\_crcmax\_lat}}$ $\mathbf{t_{phy\_crcmin\_lat}}$ $\mathbf{t_{phy\_wrdata}}$ $\mathbf{t_{phy\_wrlat}}$ $\mathbf{t_{wrdata\_delay}}$	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]
dfi_wrdata_mask_*pN*	$\mathbf{phy_{crc\_mode}}$ $\mathbf{t_{phy\_wrdata}}$ $\mathbf{t_{phy\_wrlat}}$	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]
Read Data Interface			
Signal	**Associated Parameters**	**MC**	**PHY**

TABLE 4. *DFI Signal Requirements*

dfi_rddata_*wN*	t_{phy_rdlat} t_{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*	$t_{phy\_rdcsgap}$ $t_{phy\_rdcslat}$	Required for all DRAMs if there is more than 1 chip select. 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*	phy_{dbi_mode} $phy_{linkecc\_mode}$ t_{phy_rdlat} t_{rddata_en}	Applicable for DDR4, DDR5, LPDDR4 and LPDDR5 DRAMs only and required when MC DBI support is enabled and $phy_{dbi\_mode} = 0$. Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]	Applicable for DDR4, DDR5, LPDDR4 and LPDDR5 DRAMs only and required when MC DBI support is enabled and $phy_{dbi\_mode} = 0$. Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]
dfi_rddata_dnv_*wN*	t_{phy_rdlat} t_{rddata_en}	Required for LPDDR2 DRAMs. [b] Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]	Required for LPDDR2 DRAMs. [b] Suffix (_wN) required for frequency ratio systems to replicate information across the word. [a]
dfi_rddata_en_*pN*	t_{phy_rdlat} t_{rddata_en} dfi_{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_{phy_rdlat} t_{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]
Update Interface			
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.

TABLE 4. *DFI Signal Requirements*

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			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_freq_fsp	dfi_{fspx_freq}	Required if the DRAM supports FSP and the MC supports frequency change.	Required if the DRAM supports FSP and the PHY supports frequency change.
dfi_freq_ratio	dfi_{freq_ratio}	Required if the MC supports changing the frequency ratio on a frequency change. [b] Not applicable if the system supports a single ratio.	Required if the PHY supports changing the frequency ratio on a frequency change. Not applicable if the system supports a single ratio.
dfi_frequency	phy_{freq_range}	Required for all DRAMs if the MC supports frequency change.	Required if the PHY supports frequency change.
dfi_init_complete	$t_{init\_complete}$ t_{init_start}	Required for all DRAMs.	Required for all DRAMs.
dfi_init_start	$t_{init\_complete}$ t_{init_start} $t_{init\_start\_min}$	Required for all DRAMs.	Required for all DRAMs.
PHY Master Interface			
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.
Disconnect Protocol			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_disconnect_error	$t_{ctrlupd\_disconnect}$ $t_{ctrlupd\_disconnect\_error}$ $t_{phymstr\_disconnect}$ $t_{phymstr\_disconnect\_error}$ $t_{phyupd\_disconnect}$ $t_{phyupd\_disconnect\_error}$	Optional for all DRAMs.	Optional.

TABLE 4. *DFI Signal Requirements*

Error Interface (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.
Low Power Control Interface (optional)			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_lp_ctrl_ack	t_{lp_resp} $t_{lp\_ctrl\_wakeup}$	Supported for all DRAM types. Required when low power is supported.	Supported for all DRAM types. Required when low power is supported.
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_ctrl_wakeup	$t_{lp\_ctrl\_wakeup}$	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_ack	t_{lp_resp} $t_{lp\_data\_wakeup}$	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_data_wakeup	$t_{lp\_data\_wakeup}$	Supported for all DRAM types. Required when low power is supported.	Supported for all DRAM types. Required when low power is supported.
MC to PHY Message Interface			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_ctrlmsg		Optional.	Optional.
dfi_ctrlmsg_ack	$t_{ctrlmsg\_resp}$ $t_{ctrlmsg\_max}$	Optional.	Optional.
dfi_ctrlmsg_data		Optional.	Optional.
dfi_ctrlmsg_req		Optional.	Optional.
WCK Control Interface			
Signal	**Associated Parameters**	**MC**	**PHY**
dfi_wck_cs_*pN*	$t_{wck\_toggle\_cs}$ $t_{wck\_toggle\_post}$	Required for LPDDR5 DRAMs.	Required for LPDDR5 DRAMs.

TABLE 4. *DFI Signal Requirements*

dfi_wck_en_*pN*	t_{wck_dis} $t_{wck\_en\_fs}$ $t_{wck\_en\_rd}$ $t_{wck\_en\_wr}$ $t_{wck\_en\_wrx}$	Required for LPDDR5 DRAMs.	Required for LPDDR5 DRAMs.
dfi_wck_toggle_*pN*	$t_{wck\_fast\_toggle}$ t_{wck_toggle} $t_{wck\_toggle\_cs}$ $t_{wck\_toggle\_post}$	Required for LPDDR5 DRAMs.	Required for LPDDR5 DRAMs.

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.

2.4 Definition of a Slice

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 $\mathbf{phy_{data\_slice\_width}}$ 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 5, “DFI Data Slice Definition Programmable Parameters”.

TABLE 5. *DFI Data Slice Definition Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{phy_{data\_slice\_width}}$	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 $\mathbf{dfi_{data\_bit\_enable}}$ parameter.

The number of PHY slices is determined by dividing the DFI Data Width by the $\mathbf{phy_{data\_slice\_width}}$ 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 $\mathbf{dfi_{data\_bit\_enable}}$ parameter. Otherwise, the number of slices might not be deterministic.

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.

3.0 Interface Signal Groups

3.1 Command Interface

The command interface handles the transmission of signals required to drive the address and command 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.

3.1.1 Mapping the CA Bus to the dfi_address Bus

For newer memories, a CA bus exists to the memory. This is not relevant for DDR1, LPDDR1, DDR2, DDR3 or DDR4 memories.

For low power memories and the latest DDR DRAMs, the CA bus is mapped onto to the **dfi_address** bus. The following signals must be held at constant values when present in any of these implementations: **dfi_cid**, **dfi_bank**, **dfi_bg**, **dfi_act_n**, **dfi_ras_n**, **dfi_cas_n**, and **dfi_we_n**. Special mapping is required for LPDDR2, LPDDR3, LPDDR5 and DDR5 memories.

For LPDDR2 and LPDDR3 memories, the **dfi_address** bus must have a minimum of 20 bits to hold the 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 6, "Bit Definitions of the dfi_address bus for LPDDR2 and LPDDR3".

TABLE 6. *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; therefore, the **dfi_address** bus is 6 bits wide.

For LPDDR5, the CA bus is a 7-bit DDR (double data rate) interface. Since the **dfi_address** is a single phase signal for LPDDR5, the CA bus phases are concatenated such that the rising edge CA is on the lower 7-bits and the falling edge CA is on the upper 7-bits.

TABLE 7. *Bit Definitions of the **dfi_address** bus for LPDDR5*

dfi_address	⎍	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CA Bus	↑								LPDDR5						
									6	5	4	3	2	1	0
	↓	LPDDR5													
		6	5	4	3	2	1	0							

For DDR5, the CA bus is an SDR (single data rate) CA bus; therefore, the **dfi_address** bus width is matched to the DDR5 CA bus width. For 1N mode, the CA bus values are driven on the **dfi_address** bus for 1 DFI clock in matched frequency or 1 clock phase for frequency ratio. For 2N mode, the CA bus values are driven on the **dfi_address** bus for 2 DFI clocks in matched frequency or 2 clock phases for frequency ratio. The command can start on any clock phase and for multi-clock commands, must continue for consecutive clocks or phases until the command is complete.

3.1.2 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 4.10, “Frequency Ratios Across the DFI”.

3.1.3 Frequency Ratio Systems

For frequency ratio systems, the buses/signals of the command 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.

3.1.4 CRC and CA Parity

The CRC and CA parity support signals are provided specifically when this feature is supported. In these cases, additional IO pins are included 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 3.2.3, “Write Data CRC”.

The signals and parameters associated with the command interface are listed in Table 8, “Command Interface Signals”, Table 9, “Command Interface Timing Parameters”, and Table 10, “2N Mode Programmable Parameter”. For more information on the command interface, refer to Section 4.3, “Command Interface Signals”. For more information on which signals are required and which signals are optional, refer to Table 4, “DFI Signal Requirements”.

TABLE 8. *Command Interface Signals*

Signal	From	Width	Default	Description
dfi_2n_mode or **dfi_2n_mode_pN** [a]	MC	1 bit	0x0	DFI 2N Mode. When de-asserted, the MC and PHY operate normally. When asserted, the MC and PHY operate in 2N mode. For DDR4, the MC can change CA and CS signals only every other DFI PHY clock as defined by the synchronization pulse from the PHY. For DDR5, the MC must hold the CA signals for at least 1 DFI PHY clock cycle after the CS is active. There is no alignment requirement between commands.

TABLE 8. *Command Interface 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 memories that use a CA bus, the CA bus maps to the **dfi_address** bus as described in Section 3.1.1, "Mapping the CA Bus to the dfi_address Bus".
dfi_alert_n or **dfi_alert_n_aN** [d]	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_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 [e]	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	0 [f]	DFI chip ID. This signal defines the chip ID. This signal is required for 3D stacked solutions.

TABLE 8. *Command Interface Signals*

Signal	From	Width	Default	Description
dfi_cke or **dfi_cke_pN** [a]	MC	DFI CKE Width	- [g]	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.
dfi_cs or **dfi_cs_pN** [a]	MC	DFI Chip Select Width	0x1	DFI chip select. This signal defines the chip select. 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_dram_clk_disable or **dfi_dram_clk_disable_pN** [a]	MC	DFI DRAM Clk Disable Width	0x0 [h]	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_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_parity_in or **dfi_parity_in_pN** [a]	MC	1 bit	0x0 [i] 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.
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	- [j]	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 memory systems that use a CA bus, the **dfi_address** bus must be driven with a DES until **dfi_init_complete** is asserted.

[d] 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.

e. 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.

f. This signal initializes to 0 during initialization; a default value of 0 is required.

g. The MC is expected to drive the **dfi_cke** signal with appropriate value as per the state of the memory defined in the $\mathbf{dfi_{boot\_state}}$ programmable parameter.

h. 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.

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

j. The MC is expected to drive the **dfi_reset_n** signal with appropriate value as per the state of the memory defined in the $\mathbf{dfi_{boot\_state}}$ programmable parameter.

The timing parameters associated with the command interface are listed in Table 9, “Command Interface Timing Parameters”.

TABLE 9. *Command Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{2n\_mode\_delay}$	PHY	0x0	- [a]	DFI command clock cycles	The delay from **dfi_2n_mode** assertion to the time that the PHY is ready to receive commands.
t_{cmd_lat}	MC	0x0	- [a]	DFI command clock cycles	Specifies the number of DFI clocks after the **dfi_cs** signal is asserted until the associated CA signals are driven.
t_{ctrl_delay}	PHY	0x0	- [a]	DFI command clock cycles	Specifies the number of DFI clock cycles from the time that any command 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.
$t_{dram\_clk\_disable}$ [b]	PHY	0x1	- [b, c]	DFI command clock cycles	Specifies the number of DFI clock cycles from the assertion of the first phase of the **dfi_dram_clk_disable_pN** 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.
$t_{dram\_clk\_enable}$ [b]	PHY	0x1	- [c]	DFI command clock cycles	Specifies the number of DFI clock cycles from the de-assertion of the first phase of the **dfi_dram_clk_disable_pN** 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.

TABLE 9. *Command Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
t_{parin_lat}	MC	0x0	-	DFI command clock cycles	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	-	DFI command clock cycles	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. The minimum supportable value is 0; the DFI specification does not specify a maximum value. The range of values supported is implementation-specific.

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

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

3.1.5 2N Mode Interface

Some DRAMs include a 2N mode (also referred to as the 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. With 2N mode, all command/address (CA) bus signals can benefit from larger setup and hold times. In addition, for DDR4 memories operating in Geardown mode, the CS signals can also 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. This is not a requirement for all memory types. For DDR4 operating in Geardown mode, the MC can change CA and CS signals only every other DFI PHY clock as defined by the synchronization pulse from the PHY. For DDR5 operating in 2N mode, the MC can change the CA signals only every other DFI PHY clock, however there is no synchronization pulse and no restrictions on the alignment.

For 2N mode, the PHY might need to change the alignment of the memory command's CA and/or CS 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 2N 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 2N mode via the **dfi_2n_mode** signal.

The signal associated with the 2N mode interface is listed in Table 8, "Command Interface Signals".

Table 9, "Command Interface Timing Parameters" defines a 2N mode timing parameter.

The programmable parameter associated with the 2N mode is listed in Table 10, "2N Mode Programmable Parameter".

TABLE 10. *2N Mode Programmable Parameter*

Parameter	Defined By	Description
dfi$_{\mathbf{2n\_mode}}$	SYS	Indicates the initial operating state following the boot sequence when the PHY asserts **dfi_init_complete**. • ’b0 = 1N Mode • ’b1 = 2N Mode

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 11, “Write Data Interface 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 **t$_{\mathbf{phy\_wrdata}}$** cycles.

3.2.1 Write Data Mask/Write DBI

The **dfi_wrdata_mask** signal has two functions. If the DBI feature (described in Section 4.6, “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 **phy$_{\mathbf{dbi\_mode}}$** = 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 and MRW 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 command signal timing to handle the additional data word.

Table 12, “Write Data Interface Timing Parameters” describes the write timing parameters **t$_{\mathbf{phy\_wrcsgap}}$**, **t$_{\mathbf{phy\_wrcslat}}$**, **t$_{\mathbf{phy\_wrdata}}$**, **t$_{\mathbf{phy\_wrlat}}$**, **t$_{\mathbf{wrdata\_delay}}$**, **t$_{\mathbf{phy\_crcmax\_lat}}$** and **t$_{\mathbf{phy\_crcmin\_lat}}$**.

The **t$_{\mathbf{phy\_wrcsgap}}$** 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 **t$_{\mathbf{phy\_wrcslat}}$** parameter specifies the number of DFI PHY clocks between when a write command is sent on the DFI command 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 **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 **t_{phy_wrlat}** parameter specifies the number of DFI PHY clock cycles between when a write command is sent on the DFI command interface and when the **dfi_wrdata_en** signal is asserted. The **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 **$t_{phy\_crcmax\_lat}$** and **$t_{phy\_crcmin\_lat}$** 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 13, “Write Data Interface 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 **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 associated with the write data interface are listed in Table 11, “Write Data Interface Signals”, Table 12, “Write Data Interface Timing Parameters”, and Table 13, “Write Data Interface Programmable Parameters”.

For more information on the write data interface, refer to Section 4.7, “Write Transactions”. For more information on which signals are required and which signals are optional, refer to Table 4, “DFI Signal Requirements”.

TABLE 11. *Write Data Interface 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 **t_{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 indicates the chip select that is accessed or targeted for associated write data.

TABLE 11. *Write Data Interface Signals*

Signal	From	Width	Default	Description
dfi_wrdata_ecc or **dfi_wrdata_ecc_pN** [a]	MC	DFI Data Width / 8	-[b]	Write Data Link ECC. This signal is sent with **dfi_wrdata** bus indicating link ECC functionality. This signal is used by the MC only when $\mathbf{phy}_{\mathbf{linkecc\_mode}}$ = 1. The polarity of this signal is defined by the polarity of the corresponding memory signal. When the **dfi_wrdata_ecc** signal is used, it is sent with the **dfi_wrdata** signal.
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 and MRW 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 12, "Write Data Interface Timing Parameters".

TABLE 12. *Write Data Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{phy\_crcmax\_lat}$	System	0x1	- [a]	DFI data 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 data 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 data 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 data 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 command interface and when the associated **dfi_wrdata_cs** signal is asserted.

TABLE 12. *Write Data Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
t_{phy_wrdata}	PHY	0x0 [a]	- [a]	DFI data 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.
t_{phy_wrlat}	PHY	0x0 [a]	- [a]	DFI data 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 command 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 data 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 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, dfi_{rw_length} would typically equal (burst length/2). For a frequency ratio system, 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 13, "Write Data Interface Programmable Parameters".

TABLE 13. *Write Data Interface Programmable Parameters*

Parameter	Defined By	Description
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.
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 14, “Read Data Interface Signals” lists the signals in the read data interface.

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 and MRW 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 15, “Read Data Interface 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 command 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.

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 associated with the read data interface are listed in Table 14, "Read Data Interface Signals", Table 15, "Read Data Interface Timing Parameters", and Table 16, "Read Data Interface Programmable Parameter". The programmable parameter $\mathbf{phy_{dbi\_mode}}$ applies to both write and read signals.

For more information on the read data interface, refer to Section 4.8, "Read Transactions". For more information on which signals are required and which signals are optional, refer to Table 4, "DFI Signal Requirements".

TABLE 14. *Read Data Interface 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 $\mathbf{t_{phy\_rdlat}}$ 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 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 or Link ECC. This signal is sent with **dfi_rddata** bus indicating DBI functionality or link ECC information. This signal is used by the MC only when $\mathbf{phy_{dbi\_mode}}$ = 0 or when $\mathbf{phy_{linkecc\_mode}}$ = 1 and the read data copy function is disabled. In LPDDR5, link ECC and the read data copy function, or the link ECC and read DBI functions are mutually exclusive. The polarity of this signal is defined by the polarity of the corresponding memory signal. When the **dfi_rddata_dbi** signal is used, it is sent with the **dfi_rddata** signal.

TABLE 14. *Read Data Interface Signals*

Signal	From	Width	Default	Description
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. This must be sent with the read data signal **dfi_rddata** when the **dfi_rddata_valid** signal is asserted.
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 $\mathbf{t_{rddata\_en}}$ cycles after the assertion of a read command on the DFI command 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.
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 and MRW 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 15, “Read Data Interface Timing Parameters”.

TABLE 15. *Read Data Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{phy\_rdcsgap}}$	PHY	0x0	-[a]	DFI data 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 data clock cycles [b]	Specifies the number of DFI PHY clocks between when a read command is sent on the DFI command interface and when the associated **dfi_rddata_cs** signal is asserted.
$\mathbf{t_{phy\_rdlat}}$	PHY	0x0	- [a]	DFI data 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 data 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 $\mathbf{dfi_{rw\_length}}$ 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 parameter associated with the read data interface is listed in Table 16, “Read Data Interface Programmable Parameter”.

TABLE 16. *Read Data Interface Programmable Parameter*

Parameter	Defined By	Description
$\mathbf{phy_{dbi\_mode}}$	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.

3.4 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, 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. The update interface does not define a required state of memory.

When the DFI bus is in an idle state, the command 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 4.9, “Update”.

The $\mathbf{t_{ctrlupd\_interval}}$ parameter defines the maximum interval at which the MC can assert **dfi_ctrlupd_req** signals. A MC-initiated update request via **dfi_ctrlupd_req** 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 unless the PHY de-asserts the **dfi_phyupd_req** signal. The PHY is permitted to de-assert this request when **dfi_ctrlupd_req** is asserted or the **dfi_init_start** signal is asserted. 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 4.9.2, “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. 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 and parameters associated with the update interface are listed in Table 17, “Update Interface Signals”, and Table 18, “Update Interface Timing Parameters”. For more information on the update interface, refer to Section 4.9, “Update”. For more information on which signals are required and which signals are optional, refer to Table 4, “DFI Signal Requirements”.

TABLE 17. *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.
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.

TABLE 17. *Update Interface Signals*

Signal	From	Width	Default	Description
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. In most cases, the MC must assert the **dfi_phyupd_ack** signal within $\mathbf{t_{phyupd\_resp}}$ cycles of the **dfi_phyupd_req** signal; exceptions are granted when the **dfi_phymstr_req** signal is also asserted (refer to Section 4.21, "DFI Interactions" for details). When the **dfi_phyupd_ack** signal is asserted, it should remain 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. The $\mathbf{t_{phyupd\_resp}}$ duration should be sufficiently large to allow for all cases, including a Controller Update occurring before the PHY Update is acknowledged.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.
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 $\mathbf{t_{phyupd\_typeX}}$ parameter associated with the **dfi_phyupd_type** signal. Once asserted, the **dfi_phyupd_req** signal must generally remain asserted until the request is acknowledged by the assertion of the **dfi_phyupd_ack** signal and the PHY's internal update procedure has been completed. Exceptions are granted if the **dfi_ctrlupd_req**, **dfi_phymstr_req** or **dfi_init_start** signals are also asserted - refer to Section 4.21, "DFI Interactions" for details. 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.

TABLE 17. *Update Interface Signals*

Signal	From	Width	Default	Description
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 ($\mathbf{t_{phyupd\_type0}}$, $\mathbf{t_{phyupd\_type1}}$, $\mathbf{t_{phyupd\_type2}}$, $\mathbf{t_{phyupd\_type3}}$) 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 18, “Update Interface Timing Parameters”.

TABLE 18. *Update Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{ctrlupd\_interval}}$	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.
$\mathbf{t_{ctrlupd\_max}}$	MC	- [a]	- [a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles that the **dfi_ctrlupd_req** signal can assert.
$\mathbf{t_{ctrlupd\_min}}$	MC	0x1	- [a]	DFI clock cycles	Specifies the minimum number of DFI clock cycles that the **dfi_ctrlupd_req** signal must be asserted.
$\mathbf{t_{phyupd\_resp}}$	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. Exceptions are granted if **dfi_init_start**, **dfi_ctrlupd_req**, or **dfi_phymstr_req** are active along with **dfi_phyupd_req**. Refer to Section 4.9.2, “PHY-Initiated Update” for details. This timing parameter should be greater than $\mathbf{t_{init\_resp}}$ and much greater than $\mathbf{t_{ctrlupd\_resp}}$.
$\mathbf{t_{phyupd\_type0}}$	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.
$\mathbf{t_{phyupd\_type1}}$	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.

TABLE 18. *Update Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
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 Change" describes in detail. At initialization, the **dfi_init_start** signal indicates to the PHY that the optional DFI frequency signals (**dfi_frequency, dfi_freq_ratio and dfi_freq_fsp)** 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. This information is conveyed through the $\mathbf{dfi_{data\_bit\_enable}}$ programmable parameter. 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 Frequency Ratio

The **dfi_freq_ratio** signal conveys frequency ratio information to the PHY. This signal indicates the ratio between the MC and the PHY for the signals operating at a ratio across the DFI. This is an optional signal which is required if the MC and/or PHY supports changing frequency ratio on a frequency change. For more information on the frequency ratio protocol, refer to Section 4.10, "Frequency Ratios Across the DFI".

3.5.4 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 and frequency ratio of the MC and the PHY during operation.

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 de-asserting **dfi_init_complete**. The associated timing parameters are $\mathbf{t_{init\_start}}$ and $\mathbf{t_{init\_complete}}$.

During a frequency change, the frequency ratio between the MC and the PHY can be changed using the **dfi_freq_ratio** signal. For memories that have multiple FSPs, the **dfi_freq_fsp** signal should be driven indicating the FSP associated with the new target frequency. The **dfi_frequency** signal reports the operating frequency for the system. The timing of the **dfi_freq_fsp** signal is similar to the **dfi_frequency** signal while the **dfi_init_start** signal is asserted during a frequency change operation.

For more information on the frequency change protocol, refer to Section 4.11, “Frequency Change”.

3.5.5 Frequency Indicator

The MC initiates a frequency change operation and reports the new frequency through the encoding of the **dfi_frequency** signal, the new frequency set point through the **dfi_freq_fsp** signal and the new frequency ratio through the encoding of the **dfi_freq_ratio** signal to the PHY. 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 recommended 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**, **dfi_freq_fsp** or **dfi_freq_ratio** signals; the system must determine the initial frequency, FSP and frequency ratio and the MC drives those initial values when the it asserts the **dfi_init_start** signal during initialization. Similar to existing signals, the PHY must wait for the **dfi_init_start** signal to assert.

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.

3.5.6 Status Interface Signals and Parameters

The signals and parameters in the status interface are listed in Table 19, “Status Interface Signals”, Table 20, “Status Interface Timing Parameters”, and Table 21, “Status Interface Programmable Parameters”.

For more information on which signals are required and which signals are optional, refer to Table 4, “DFI Signal Requirements”.

TABLE 19. *Status Interface Signals*

Signal	From	Width	Default	Description
dfi_freq_fsp	MC	DFI FSP Width	?	DFI frequency set point. Indicates the operating frequency set point for the system. This signal should change only at initialization or during a DFI frequency change operation. This signal is required for MCs and PHYs that support multiple frequency set points. This signal is required only for the DRAMs that support FSP. This signal is optional for MCs and PHYs that support only a single frequency. This signal is valid when **dfi_init_start** is asserted during initialization and during frequency change operations. • For a DFI FSP Width of 1 bit: • ’b0 = FSP0 • ’b1 = FSP1 • For a DFI FSP Width of 2 bits: • ’b00 = FSP0 • ’b01 = FSP1 • ’b10 = FSP2 • ’b11 = Reserved
dfi_freq_ratio	MC	2 bits	?	DFI frequency ratio indicator. This signal defines the frequency ratio for the 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 is only valid when the **dfi_init_start** signal is asserted during initialization and frequency changes. • ’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 For memories that support a frequency ratio only for data, the signal defines the frequency ratio for the data interface.
dfi_frequency	MC	5 bits	?	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.

TABLE 19. *Status Interface Signals*

Signal	From	Width	Default	Description
dfi_init_complete	PHY	1 bit	0x0 [a]	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 $\mathbf{t_{init\_complete}}$ 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 [a] 0x1	DFI setup stabilization or frequency change initiation. This signal can perform two functions. When **dfi_init_start** is asserted during initialization, the MC is indicating that the **dfi_frequency**, **dfi_freq_fsp** and **dfi_freq_ratio** signals are driven with valid values. 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 $\mathbf{t_{init\_start}}$ 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 $\mathbf{t_{init\_start}}$ cycles.

[a] The PHY should wait for the **dfi_init_start** signal assertion before asserting the **dfi_init_complete** signal.

The timing parameters associated with the status interface are listed in Table 20, “Status Interface Timing Parameters”.

TABLE 20. *Status Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{init\_complete}}$	PHY	0x0	- [b]	DFI clock cycles [c]	During a frequency change operation, 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.
$\mathbf{t_{init\_start}}$	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. After $\mathbf{t_{init\_start}}$ expires, the PHY must not de-assert the **dfi_init_complete** signal. The MC can re-assert **dfi_init_start** at a later point. The PHY may complete memory setup before de-asserting the **dfi_init_complete** signal. In this case, $\mathbf{t_{init\_start}}$ should be programmed to larger values.
$\mathbf{t_{init\_start\_min}}$	MC	0x1	- [a]	DFI clock cycles	Minimum number of DFI clocks before **dfi_init_start** can be driven after a previous command.

The programmable parameters associated with the status interface are listed in Table 21, "Status Interface Programmable Parameters".

TABLE 21. *Status Interface Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{dfi_{boot\_state}}$	PHY	Defines the state of memory when the DFI bus ownership is passed to the controller after PHY initialization/training. • 'b0 = Memory is in Idle • 'b1 = Memory is in Self-Refresh

TABLE 21. *Status Interface Programmable Parameters*

Parameter	Defined By	Description
dfi$_{\mathbf{data\_bit\_enable}}$	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. Previous DFI specifications included a **dfi_data_byte_disable** signal. That signal has been removed for lack of usefulness and been replaced by this 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.
dfi$_{\mathbf{freq\_change\_state}}$	PHY	Defines the state of memory when the MC initiates a frequency change operation by asserting the **dfi_init_start** signal or when PHY asserts the **dfi_init_complete** signal after completing re-initialization and re-training during a frequency change operation. This is the state when the DFI bus ownership is passed back to the controller. • ’b0 = Memory is in Idle • ’b1 = Memory is in Self refresh
dfi$_{\mathbf{freq\_ratio}}$	SYS	Defines the ratio of the DFI Clock and the clock in the PHY. This is referred to as the frequency ratio in previous DFI specifications and is generally associated with the ratio of the controller clock to the memory clock. This is the state when the DFI bus ownership is passed back to the controller. • ’b000 = 1:1 • ’b001 = 1:2 • ’b010 = 1:4 • All other settings reserved For memories other than LPDDR5, this is the ratio between the DFI clock and the Memory clock. For LPDDR5 memories, this is the ratio between the DFI Clock and WCK. The DFI Clock to CK clock is always a 1:1 ratio. This value will not necessarily be correct post-initialization; the **dfi_freq_ratio** signal is used for frequency change.
dfi$_{\mathbf{fspx\_freq}}$	SYS	FSPx Frequency, where x is defined by the number of FSPs supported in the DRAM. Defines the frequency for each FSP. This is an encoded value as defined by the **dfi_frequency** signal and the **phy$_{\mathbf{freq\_range}}$** programmable parameter. These values will not necessarily be correct post-initialization; the **dfi_freq_fsp** signal is used for frequency change.
dfi$_{\mathbf{init\_freq}}$	SYS	Defines the DFI clock frequency during initialization with the **dfi_init_start** signal. This is an encoded value as defined by the **dfi_frequency** signal and the **phy$_{\mathbf{freq\_range}}$** programmable parameter.
phy$_{\mathbf{freq\_range}}$	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.

The status interface provides multiple programmable parameters to convey information between the MC and PHY. The **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.

3.6 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 22, “Low Power Control Interface Signals” describes the signals used in the low power control interface: **dfi_lp_ctrl_req** (control low power opportunity request), **dfi_lp_data_req** (data low power opportunity request), **dfi_lp_ctrl_ack** (control low power acknowledge), **dfi_lp_data_ack** (data low power acknowledge), **dfi_lp_ctrl_wakeup** (control low power wakeup time) and **dfi_lp_data_wakeup** (data low power wakeup time).

The low power requests (control and data) are independent interfaces which can be acknowledged without any dependency between them. Once asserted, a low power (control/data) request cannot be de-asserted unless the corresponding acknowledge is not asserted within $\mathbf{t_{lp\_resp}}$. Table 23, “Low Power Control Interface Timing Parameters” describes the timing parameters: $\mathbf{t_{lp\_resp}}$ and $\mathbf{t_{lp\_ctrl\_wakeup}}$ and $\mathbf{t_{lp\_data\_wakeup}}$.

The signals and parameters associated with the low power interface are listed in Table 22, “Low Power Control Interface Signals”, Table 23, “Low Power Control Interface Timing Parameters”, and Table 24, “Low Power Control Interface Programmable Parameters”. 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 4, “DFI Signal Requirements”.

TABLE 22. *Low Power Control Interface Signals*

Signal	From	Width	Default	Description
dfi_lp_ctrl_ack	PHY	1 bit	0x0	Control low power acknowledge. The **dfi_lp_ctrl_ack** signal is asserted to acknowledge the MC control low power opportunity request. The PHY is not required to acknowledge this request. If the PHY acknowledges the request, the **dfi_lp_ctrl_ack** signal must be asserted within $\mathbf{t_{lp\_resp}}$ cycles after the **dfi_lp_ctrl_req** signal assertion. Once asserted, this signal remains asserted until the **dfi_lp_ctrl_req** signal de-asserts. The signal de-asserts within $\mathbf{t_{lp\_ctrl\_wakeup}}$ cycles after the **dfi_lp_ctrl_req** signal de-asserts, indicating that the PHY is able to resume normal operation. If the PHY ignores the request, the **dfi_lp_ctrl_ack** signal must remain de-asserted for the remainder of the low power mode opportunity. The **dfi_lp_ctrl_req** signal is asserted for at least $\mathbf{t_{lp\_resp}}$ cycles.
dfi_lp_ctrl_req	MC	1 bit	0x0	Control low power opportunity 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 command interface. The MC must assert a constant value on the **dfi_lp_ctrl_wakeup** signal while this signal is asserted before the request is acknowledged by the PHY through the assertion of the **dfi_lp_ctrl_ack** signal or until $\mathbf{t_{lp\_resp}}$ cycles have elapsed. The MC may increase the value of the **dfi_lp_ctrl_wakeup** signal while the **dfi_lp_ctrl_req** signal is asserted. Following the de-assertion of the **dfi_lp_ctrl_req** signal, the PHY has $\mathbf{t_{lp\_ctrl\_wakeup}}$ cycles to resume normal operation and de-assert the **dfi_lp_ctrl_ack** signal.

TABLE 22. *Low Power Control Interface Signals*

Signal	From	Width	Default	Description
dfi_lp_ctrl_wakeup	MC	4 or 6 bits depending on **$dfi_{lp\_version}$**	-[a]	Control low power wakeup time. The **dfi_lp_ctrl_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** signal is asserted. The **dfi_lp_ctrl_wakeup** signal must remain constant until the **dfi_lp_ctrl_ack** signal is asserted. Once the request has been acknowledged, the MC may increase the **dfi_lp_ctrl_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_ctrl_wakeup** signal at the time that the **dfi_lp_ctrl_req** signal is de-asserted sets the **$t_{lp\_ctrl\_wakeup}$** time. • If **$dfi_{lp\_version}$** = ’b0: • ’b000000 = **$t_{lp\_ctrl\_wakeup}$** is 1 cycle • ’b000001 = **$t_{lp\_ctrl\_wakeup}$** is 2 cycles • ’b000010 = **$t_{lp\_ctrl\_wakeup}$** is 4 cycles • ’b000011 = **$t_{lp\_ctrl\_wakeup}$** is 8 cycles • ’b000100 = **$t_{lp\_ctrl\_wakeup}$** is 16 cycles • ’b000101 = **$t_{lp\_ctrl\_wakeup}$** is 32 cycles • ’b000110 = **$t_{lp\_ctrl\_wakeup}$** is 64 cycles • ’b000111 = **$t_{lp\_ctrl\_wakeup}$** is 128 cycles • ’b001000 = **$t_{lp\_ctrl\_wakeup}$** is 256 cycles • ’b001001 = **$t_{lp\_ctrl\_wakeup}$** is 512 cycles • ’b001010 = **$t_{lp\_ctrl\_wakeup}$** is 1024 cycles • ’b001011 = **$t_{lp\_ctrl\_wakeup}$** is 2048 cycles • ’b001100 = **$t_{lp\_ctrl\_wakeup}$** is 4096 cycles • ’b001101 = **$t_{lp\_ctrl\_wakeup}$** is 8192 cycles • ’b001110 = **$t_{lp\_ctrl\_wakeup}$** is 16384 cycles • ’b001111 = **$t_{lp\_ctrl\_wakeup}$** is 32768 cycles • ’b010000 = **$t_{lp\_ctrl\_wakeup}$** is 65536 cycles • ’b010001 = **$t_{lp\_ctrl\_wakeup}$** is 131072 cycles • ’b010010 = **$t_{lp\_ctrl\_wakeup}$** is 262144 cycles • ’b010011 = **$t_{lp\_ctrl\_wakeup}$** is unlimited • If **$dfi_{lp\_version}$** = ’b1: • ’b0000 = **$t_{lp\_ctrl\_wakeup}$** is 16 cycles • ’b0001 = **$t_{lp\_ctrl\_wakeup}$** is 32 cycles • ’b0010 = **$t_{lp\_ctrl\_wakeup}$** is 64 cycles • ’b0011 = **$t_{lp\_ctrl\_wakeup}$** is 128 cycles • ’b0100 = **$t_{lp\_ctrl\_wakeup}$** is 256 cycles • ’b0101 = **$t_{lp\_ctrl\_wakeup}$** is 512 cycles • ’b0110 = **$t_{lp\_ctrl\_wakeup}$** is 1024 cycles • ’b0111 = **$t_{lp\_ctrl\_wakeup}$** is 2048 cycles • ’b1000 = **$t_{lp\_ctrl\_wakeup}$** is 4096 cycles • ’b1001 = **$t_{lp\_ctrl\_wakeup}$** is 8192 cycles • ’b1010 = **$t_{lp\_ctrl\_wakeup}$** is 16384 cycles • ’b1011 = **$t_{lp\_ctrl\_wakeup}$** is 32768 cycles • ’b1100 = **$t_{lp\_ctrl\_wakeup}$** is 65536 cycles • ’b1101 = **$t_{lp\_ctrl\_wakeup}$** is 131072 cycles • ’b1110 = **$t_{lp\_ctrl\_wakeup}$** is 262144 cycles • ’b1111 = **$t_{lp\_ctrl\_wakeup}$** is unlimited

TABLE 22. *Low Power Control Interface Signals*

Signal	From	Width	Default	Description
dfi_lp_data_ack	PHY	1 bit	0x0	Data low power acknowledge. The **dfi_lp_data_ack** signal is asserted to acknowledge the MC data low power opportunity request. The PHY is not required to acknowledge this request. If the PHY acknowledges the request, the **dfi_lp_data_ack** signal must be asserted within $\mathbf{t_{lp\_resp}}$ cycles after the **dfi_lp_data_req** signal assertion. Once asserted, this signal remains asserted until the **dfi_lp_data_req** signal de-asserts. The signal de-asserts within $\mathbf{t_{lp\_data\_wakeup}}$ cycles after the **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_data_ack** signal must remain de-asserted for the remainder of the low power mode opportunity. The **dfi_lp_data_req** signal is asserted for at least $\mathbf{t_{lp\_resp}}$ cycles.
dfi_lp_data_req	MC	1 bit	0x0	Data low power opportunity 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_data_wakeup** signal while this signal is asserted before the request is acknowledged by the PHY through the assertion of the **dfi_lp_data_ack** signal or until $\mathbf{t_{lp\_resp}}$ cycles have elapsed. The MC may increase the value of the **dfi_lp_data_wakeup** signal while the **dfi_lp_data_req** signal is asserted. Following the de-assertion of the **dfi_lp_data_req** signal, the PHY has $\mathbf{t_{lp\_data\_wakeup}}$ cycles to resume normal operation and de-assert the **dfi_lp_data_ack** signal.

TABLE 22. *Low Power Control Interface Signals*

Signal	From	Width	Default	Description
dfi_lp_data_wakeup	MC	4 or 6 bits depending on $\mathbf{dfi}_{\mathbf{lp\_version}}$	- [b]	Data low power wakeup time. The **dfi_lp_data_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_data_req** signal is asserted. The **dfi_lp_data_wakeup** signal must remain constant until the **dfi_lp_data_ack** signal is asserted. Once the request has been acknowledged, the MC may increase the **dfi_lp_data_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_data_wakeup** signal at the time that the **dfi_lp_data_req** signal is de-asserted sets the $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ time. • If $\mathbf{dfi}_{\mathbf{lp\_version}}$ = ’b0: • ’b000000 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 1 cycle • ’b000001 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 2 cycles • ’b000010 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 4 cycles • ’b000011 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 8 cycles • ’b000100 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 16 cycles • ’b000101 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 32 cycles • ’b000110 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 64 cycles • ’b000111 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 128 cycles • ’b001000 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 256 cycles • ’b001001 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 512 cycles • ’b001010 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 1024 cycles • ’b001011 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 2048 cycles • ’b001100 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 4096 cycles • ’b001101 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 8192 cycles • ’b001110 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 16384 cycles • ’b001111 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 32768 cycles • ’b010000 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 65536 cycles • ’b010001 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 131072 cycles • ’b010010 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 262144 cycles • ’b010011 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is unlimited • If $\mathbf{dfi}_{\mathbf{lp\_version}}$ = ’b1: • ’b0000 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 16 cycles • ’b0001 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 32 cycles • ’b0010 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 64 cycles • ’b0011 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 128 cycles • ’b0100 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 256 cycles • ’b0101 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 512 cycles • ’b0110 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 1024 cycles • ’b0111 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 2048 cycles • ’b1000 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 4096 cycles • ’b1001 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 8192 cycles • ’b1010 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 16384 cycles • ’b1011 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 32768 cycles • ’b1100 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 65536 cycles • ’b1101 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 131072 cycles • ’b1110 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is 262144 cycles • ’b1111 = $\mathbf{t}_{\mathbf{lp\_data\_wakeup}}$ is unlimited

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

[b] 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 23, “Low Power Control Interface Timing Parameters”.

TABLE 23. *Low Power Control Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{lp\_ctrl\_wakeup}$	MC	1 (0x1)	262144 (0x4000)	DFI clock cycles	Specifies the target maximum number of DFI clock cycles that the **dfi_lp_ctrl_ack** signal may remain asserted after the de-assertion of the **dfi_lp_ctrl_req** signal. The **dfi_lp_ctrl_ack** signal may de-assert at any cycle after the de-assertion of the **dfi_lp_ctrl_req** signal. Exceeding the maximum is not considered an error condition.
$t_{lp\_data\_wakeup}$	MC	1 (0x1)	262144 (0x4000)	DFI clock cycles	Specifies the target maximum number of DFI clock cycles that the **dfi_lp_data_ack** signal may remain asserted after the de-assertion of the **dfi_lp_data_req** signal. The **dfi_lp_data_ack** signal may de-assert at any cycle after the de-assertion of the **dfi_lp_data_req** signal. Exceeding the maximum is not considered an error condition.
t_{lp_resp}	MC	0x1 [a]	16	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 associated **dfi_lp_ctrl_ack** or **dfi_lp_data_ack** signal.

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

The programmable parameters associated with the low power control interface are listed in Table 24, “Low Power Control Interface Programmable Parameters”. The $\mathbf{dfi}_{lp\_version}$ parameter defines which wakeup time definition is used by both the MC and PHY, either the DFI 5 definition or the DFI 4.0 and earlier definition. The $\mathbf{dfi}_{lp\_wakeup\_threshold}$ defines the wakeup time at which the PHY is in a low power state where it can no longer respond to DFI interface requests from the controller - this includes controller updates, controller messages, and frequency change requests. Once the wakeup time meets or exceeds this value, no more commands should be sent on these interfaces. If any of these command requests are pending or currently acknowledged, the command should be allowed to complete as usual. If the PHY has no limitation, the $\mathbf{dfi}_{lp\_wakeup\_threshold}$ can be set to 0 indicating that the parameter is not applicable.

TABLE 24. *Low Power Control Interface Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{dfi}_{lp\_version}$	MC/PHY	DFI low power version parameter for connecting a DFI 5 MC/PHY with an older version DFI MC/PHY. • ’b0 = Connecting with DFI 5 (and above) Low Power Interface. • ’b1 = Connecting with older version DFI 5 Low Power Interface.

TABLE 24. *Low Power Control Interface Programmable Parameters*

Parameter	Defined By	Description
$\textbf{dfi}_{\textbf{lp\_wakeup\_threshold}}$	MC	Defines the low power wakeup value at which (or above) the PHY is no longer able to respond to commands including controller updates, controller messages, and frequency change requests. • ’b0 = PHY has not limitations based on wakeup time for these commands. • All Other Values = PHY defines a wakeup threshold for accepting these commands.

3.7 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 and parameter associated with the error interface are listed in Table 25, “Error Interface Signals”, and Table 26, “Error Interface Timing Parameter”. 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 4, “DFI Signal Requirements”.

TABLE 25. *Error Interface 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, $\textbf{t}_{\textbf{error\_resp}}$. 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 26, “Error Interface Timing Parameter”.

TABLE 26. *Error Interface 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.

3.8 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.

The DFI specification does not dictate any training methodology, however training may be accomplished using the PHY master interface through PHY Independent Mode. In this mode, the controller temporarily relinquishes control of the DFI bus, except for refresh commands (if necessary) and the PHY can execute any training sequences independent of the Controller.

3.8.1 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 $\mathbf{t_{phymstr\_resp}}$ violation is system-dependent and out of the scope of this document.

3.8.2 PHY Master Interface Signals and Parameters

The signals and parameters associated with 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 4, “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.
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 memories that support self-refresh without being in the power-down state, this state must be self-refresh 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 **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.

TABLE 27. *PHY Master Interface Signals*

Signal	From	Width	Default	Description
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 $\textbf{sys}_{\textbf{cs\_state}}$ 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. For memories that support self-refresh without being in the power-down state, this state must be self-refresh 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.
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 ($t_{phymstr\_type0}$, $t_{phymstr\_type1}$, $t_{phymstr\_type2}$, $t_{phymstr\_type3}$) 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 $\textbf{sys}_{\textbf{cs\_state}}$ set to ’b0 (inactive chip selects). The MC can continue keeping the chip selects with $\textbf{sys}_{\textbf{cs\_state}}$ 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
$t_{phymstr\_resp}$	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. Exceptions are granted if **dfi_init_start**, **dfi_lp_ctrl_req** or **dfi_lp_data_req** is active along with **dfi_phymstr_req**. Refer to Section 4.21, “DFI Interactions” for details. This timing parameter should be greater than t_{init_resp}.
$t_{phymstr\_rfsh}$	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.
$t_{phymstr\_type0}$	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.
$t_{phymstr\_type1}$	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.
$t_{phymstr\_type2}$	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.
$t_{phymstr\_type3}$	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
$\textbf{sys}_{\textbf{cs\_state}}$	MC/PHY	Global parameter that defines the state of each chip select in the system. The width of $\textbf{sys}_{\textbf{cs\_state}}$ is DFI Chip Select Width, with each bit corresponding to a chip select in ascending order. • $\textbf{sys}_{\textbf{cs\_state}}$[0]: State of chip select 0 • $\textbf{sys}_{\textbf{cs\_state}}$[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 $\textbf{sys}_{\textbf{cs\_state}}$ between the PHY and MC for ensuring synchronization between the MC and PHY.

3.9 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 30, “MC / PHY Handshaking Interfaces and Signals”.

TABLE 30. *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
Low Power Control	Low Power Control	**dfi_lp_ctrl_req** / **dfi_lp_ctrl_ack** **dfi_lp_data_req** / **dfi_lp_data_ack**	No
PHY Master	PHY Master	**dfi_phymstr_req** / **dfi_phymstr_ack**	Yes
Status	Frequency Change	**dfi_init_start** / **dfi_init_complete**	No

The signal associated with the disconnect protocol is listed in Table 31, “Disconnect Protocol Signal”.

TABLE 31. *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 $\mathbf{t_{*\_disconnect}}$ and $\mathbf{t_{*\_disconnect\_error}}$ timing parameter. The $\mathbf{t_{*\_disconnect}}$ timing parameter is associated with **dfi_disconnect_error** = ’b0 and the $\mathbf{t_{*\_disconnect\_error}}$ 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 32, “Disconnect Protocol Timing Parameters”.

TABLE 32. *Disconnect Protocol Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{ctrlupd\_disconnect}}$	PHY	0x0	- [a]	DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a controller update sequence, from the de-assertion of **dfi_ctrlupd_req** to the de-assertion of **dfi_ctrlupd_ack** when **dfi_disconnect_error** = ’b0.

TABLE 32. *Disconnect Protocol Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{ctrlupd\_disconnect\_error}$	PHY	0x0	-[a]	DFI clock cycles	Defines the maximum number of clocks that are required to disconnect the PHY during a controller update sequence, from the de-assertion of **dfi_ctrlupd_req** to the de-assertion of **dfi_ctrlupd_ack** when **dfi_disconnect_error** = 'b1.
$t_{phymstr\_disconnect}$	PHY	0x0	-[a]	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]	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.
$t_{phyupd\_disconnect}$	PHY	0x0	-[a]	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]	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.

[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 MC to PHY Message Interface

The MC to PHY message interface handles the transmission of messages from the MC to the PHY; the interface includes signals and timing parameters. The communication interface is an optional feature for both the MC and the PHY.

In a DDR memory sub-system, the memory subsystem functions may be supported in the controller, PHY, or both. In some instances, a function executed by the controller may result in the need to send a message to the PHY. The MC to PHY message interface encodes messages from the MC to the PHY. The messaging includes both pre-defined messages and device-specific messages. The controller and PHY should support the same encodings.

The signals and parameters associated with the MC to PHY Message interface are listed in Table 33, "MC to PHY Message Interface Signals" and Table 34, "MC to PHY Message Interface Timing Parameters".

TABLE 33. *MC to PHY Message Interface Signals*

Signal	From	Width	Default	Description
dfi_ctrlmsg	MC	8 bits	0x00	DFI Controller Message Command. Valid only when the **dfi_ctrlmsg_req** signal is asserted. Encodes messages from the MC to the PHY.
dfi_ctrlmsg_ack	PHY	1 bit	0x0	DFI Controller Message Acknowledge. When asserted, indicates that the PHY received the MC message. If the message is to be acknowledged, the **dfi_ctrlmsg_ack** signal must assert by within $\mathbf{t_{ctrlmsg\_resp}}$ clock cycles. Once asserted, the **dfi_ctrlmsg_ack** signal must de-assert within $\mathbf{t_{ctrlmsg\_max}}$ clock cycles.
dfi_ctrlmsg_data	MC	16 bits	0x0000	DFI Controller Message Data. Valid only when the **dfi_ctrlmsg_req** signal is asserted. Data associated with the info command from the MC to the PHY.
dfi_ctrlmsg_req	MC	1 bit	0x0	DFI Controller Message Request. When asserted, indicates a valid MC to PHY message. If acknowledged, the request must remain asserted until the **dfi_ctrlmsg_ack** signal is de-asserted. If not acknowledged within $\mathbf{t_{ctrlmsg\_resp}}$, the request should be de-asserted.

The timing parameters associated with the MC to PHY message interface are listed in Table 34, “MC to PHY Message Interface Timing Parameters”.

TABLE 34. *MC to PHY Message Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$\mathbf{t_{ctrlmsg\_max}}$	MC	N/A	-[a]	DFI clock cycles	Specifies the maximum number of DFI clocks that the **dfi_ctrlmsg_ack** signal can remain asserted.
$\mathbf{t_{ctrlmsg\_resp}}$	MC	1	-[a]	DFI clock cycles	Specifies the maximum number of DFI clock cycles between the assertion of the **dfi_ctrlmsg_req** signal to the assertion of the **dfi_ctrlmsg_ack** signal.

[a] There is no maximum value defined by the specification. The range of values supported is implementation-specific.

3.11 WCK Control Interface

For DRAM devices that utilize a Write Clock (WCK) clocking system, signals are defined to control the WCK synchronization sequence turning the WCK on, the toggle modes, the static states, and turning the WCK off. The signals are sent from the controller to the PHY data slices and are phased signals defined by the data interface clock frequency ratio. The **dfi_wck_en** signal defines when the clock is enabled and disabled. The **dfi_wck_toggle** signal is a 2-bit encoded value that communicates the states of the WCK: STATIC_LOW, STATIC_HIGH, TOGGLE and FAST_TOGGLE. Timing parameters are defined in the data clock domain to allow PHY-specific timing requirements to be reflected in the signal timing across the interface.

The signals and parameters associated with the WCK control interface are listed in Table 35, “WCK Control Interface Signals” and Table 36, “WCK Control Interface Timing Parameters”. These are always phased signals supporting the

WCK to CK ratio of 2:1, 4:1 or both. The number of instances of the signals is defined by the term DFI WCK Width which is generally defined by the number of WCK's on the memory bus. If there are multiple WCK's to the memory bus, the entire WCK control signal group will be replicated per WCK instance. All WCK control signals are replicated per data slice.

TABLE 35. *WCK Control Interface Signals*

Signal	From	Width	Default	Description
dfi_wck_cs or **dfi_wck_cs_pN** [a]	MC	DFI WCK Width x DFI CS Width x DFI Data Slice Count (3X)	0x0	WCK chip select. This signal indicates which chip selects currently have the WCK active. More than one chip select can be active at a time. There is one bit per chip select. This signal is only valid when the **dfi_wck_en** signal is asserted.
dfi_wck_en or **dfi_wck_en_pN** [a]	MC	DFI WCK Width x Data Slice Count	0x0	WCK clock enable. This signal defines when the WCK clock is driven or disabled (tri-state). • 'b0 = WCK disabled 'b1 = WCK enabled
dfi_wck_toggle or **dfi_wck_toggle_pN** [a]	MC	DFI WCK Width x 2 x Data Slice Count	0x0	WCK toggle. This is a 2-bit encoded value defining the state of the WCK clock. This signal is only valid when the **dfi_wck_en** signal is asserted. • 'b00 = WCK static low • 'b01 = WCK static high • 'b10 = WCK toggle • 'b11 = WCK fast-toggle

[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.

The timing parameters associated with the WCK control interface are listed in Table 36, "WCK Control Interface Timing Parameters".

TABLE 36. *WCK Control Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
t_{wck_dis}	PHY	?	- [a]	DFI data clock cycles	Defines the number of clock cycles between the last command (LAST CMD) without a WCK synchronization required (assuming no command issued) or any command that disables the WCK to when the **dfi_wck_en** signal is disabled.
$t_{wck\_en\_fs}$	PHY	?	- [a]	DFI data clock cycles	Defines the number of clocks between the CAS_WS_FS command to when the **dfi_wck_en** signal is driven.
$t_{wck\_en\_rd}$	PHY	?	- [a]	DFI data clock cycles	Defines the number of clocks between the CAS_WS_RD command to when the **dfi_wck_en** signal is driven.

TABLE 36. *WCK Control Interface Timing Parameters*

Parameter	Defined By	Min	Max	Unit	Description
$t_{wck\_en\_wr}$	PHY	0	-[a]	DFI data clock cycles	Defines the number of clocks between the CAS_WS_WR command to when the **dfi_wck_en** signal is driven.
$t_{wck\_en\_wrX}$	PHY	0	-[a]	PHY clock cycles	Defines the number of clocks between the CAS_WS_WRX command to when the **dfi_wck_en** signal is driven.
$t_{wck\_fast\_toggle}$	PHY	1	-[a]	DFI data clock cycles	Defines the number of clock cycles between the **dfi_wck_toggle** signal being driven to TOGGLE to when the **dfi_wck_toggle** signal is driven to FAST_TOGGLE. This timing is only applicable when the WCK transitions from the slow to fast toggle. Otherwise, this timing parameter should be set to 0x0.
t_{wck_toggle}	PHY	1	-[a]	DFI data clock cycles	Defines the number of clock cycles between the **dfi_wck_en** signal being enabled to when the **dfi_wck_toggle** signal is driven to TOGGLE.
$t_{wck\_toggle\_cs}$	PHY	0	-[a]	DFI data clock cycles	Defines the number of clock cycles between a read or write command to when the **dfi_wck_cs** signal must be stable. This timing is applicable when the WCK is synchronized for multiple CS's and commands are to different CS's. During WCK synchronization, the CS should be static from the CAS command to the completion of the synchronization sequence.
$t_{wck\_toggle\_post}$	PHY	0	-[a]	DFI data clock cycles	Defines the number of clock cycles after a read or write command data burst completion during which the WCK must remain in the current toggle state. During this time, the **dfi_wck_cs** signal must also remain stable.

[a] There is no maximum value defined by the specification. The range of values supported is implementation-specific.

3.12 Channels for Multi-Channel Memories

Certain memories define the DRAM signaling in terms of channels. For example, a single DRAM device may have two channels and the 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 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 37, “DFI Data Channel Programmable Parameters” defines the new programming parameter.

TABLE 37. *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.

3.13 Link ECC

LPDDR5 DRAM memories support Link ECC between a DRAM and the controller/SOC. During a write burst, ECC will be generated and checked across 128 data bits and sent to the DRAM on the RDQS_t pin. During a read burst, ECC will be generated and checked across 128 data bits and sent to the controller on the DMI (DBI) pin. In LPDDR5, link ECC and Read DBI & link ECC and the read data copy functions are mutually exclusive. Link ECC is an optional feature used with write and read bursts transmissions to generate ECC.

If the programmable parameter $\mathbf{phy}_{\mathbf{linkecc\_mode}}$ = 1, the MC controls the link ECC functionality and the **dfi_rddata_dbi** signal is required for read transactions and the **dfi_wrdata_ecc** signal is required for write transactions. If the programmable parameter $\mathbf{phy}_{\mathbf{linkecc\_mode}}$ = 0, the PHY controls link ECC functionality or link ECC is not supported. If link ECC is required in a system, either the MC or the PHY is able to generate ECC outputs for write commands and check EC C inputs for read commands.

The PHY defines the programmable parameter $\mathbf{phy}_{\mathbf{linkecc\_mode}}$ value to determine how link ECC is handled, as defined in Table 38, “DFI Link ECC Programmable Parameters”.

TABLE 38. *DFI Link ECC Programmable Parameters*

Parameter	Defined By	Description
$\mathbf{phy}_{\mathbf{linkecc\_mode}}$	PHY	PHY Link ECC Responsibility. Defines which device controls link ECC. • ’b0 = Link ECC control is handled in the PHY or not supported. • ’b1 = Link ECC control is handled in the MC.

If the MC supports link ECC on the DFI, the MC must support both settings of the $\mathbf{phy}_{\mathbf{linkecc\_mode}}$ parameter; the MC must be able to generate ECC for the write data, check and correct ECC for the read data. If the memory subsystem supports link ECC on the DFI, the PHY can optionally support either setting of the $\mathbf{phy}_{\mathbf{linkecc\_mode}}$ parameter; the PHY can optionally generate ECC for write data, check and correct ECC for the read data as required or not support the function at all. If the PHY does not support Link ECC, then PHY must be able to interface to a MC that has link ECC support. In LPDDR5, link ECC and read DBI & link ECC and the Read Data Copy functions are mutually exclusive.

When the $\mathbf{phy}_{\mathbf{linkecc\_mode}}$ = 1, the PHY transfers the read ECC on the **dfi_rddata_dbi** signal coincident with the **dfi_rddata** signal. The MC transfers the write ECC on the **dfi_wrdata_ecc** signal coincident with the **dfi_wrdata** signal.

The link ECC information transferred on the **dfi_rddata_dbi** and **dfi_wrdata_ecc** signals in relationship with data follows the JEDEC LPDDR5 specification.

The **dfi_rddata_dbi** signal has two mutually exclusive functions. If the DBI is enabled, **dfi_rddata_dbi** defines the DBI of the **dfi_rddata** signal. If DBI is disabled and Link ECC is enabled, **dfi_rddata_dbi** defines ECC of the **dfi_rddata** signal.

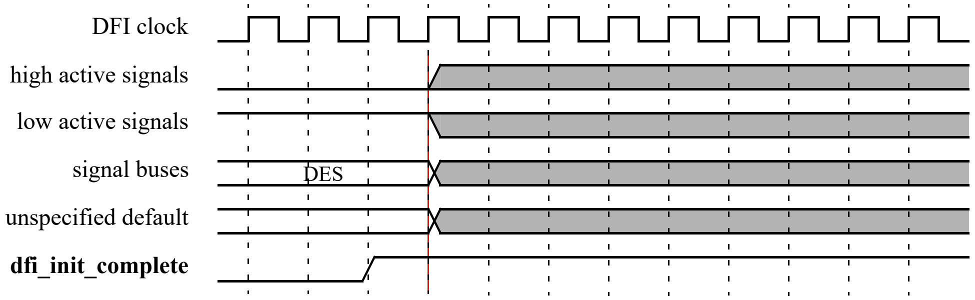
4.0 Functional Use

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 3, “Dependency on dfi_init_complete”, must maintain their default value until the **dfi_init_complete** signal is asserted.

FIGURE 3. *Dependency on **dfi_init_complete***

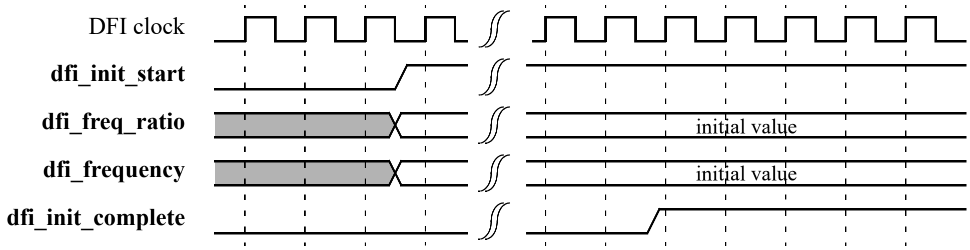


The default value for each signal is listed in the corresponding interface table. For example, the default for command signals are listed in the default column of Table 8, “Command Interface Signals”.
1. High active signals have a default value of 0 (e.g., **dfi_odt, dfi_wrdata_en**).
2. Low active signals 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**). For protocols that use a CA bus, the **dfi_address** signal does not have a default value. For protocols that use a CA bus, the **dfi_address** bus must drive a DES command until the **dfi_init_complete** signal is asserted
4. State-specific signals have state-specific values (e.g. **dfi_dram_clk_disable_pN**, **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 must use the **dfi_init_start** signal assertion to know that this status signal is valid.

Figure 4, “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. The PHY must 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. On the initial run of **dfi_init_start**, the user should also define the initial frequency for the system on the **dfi_frequency** signal and define the value for the $\mathbf{dfi_{init\_freq}}$ timing parameter.

FIGURE 4. *System Setting Signals - **dfi_init_start** Asserts Before **dfi_init_complete***



Note: "Initial Value" is the expected value at which the PHY initially transfers DFI bus ownership to the MC with the **dfi_init_complete** assertion.

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 command interface to the DRAMs prior to asserting the **dfi_init_complete** signal.

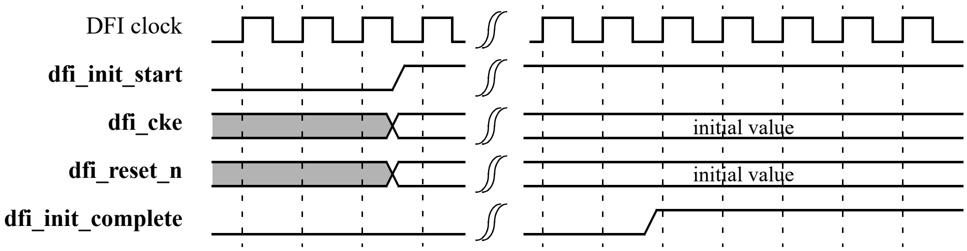
During initialization, the MC and PHY must ensure that all the memory and DFI timing requirements are met prior to transitioning corresponding **dfi_init_start** and **dfi_init_complete** signals. For DRAMs that use a CA bus, the signals of the command interface must drive a DES (de-select) command until the **dfi_init_complete** signal is asserted

4.2 PHY Independent Training Boot Sequence

The PHY executes the memory boot sequence by programming memory, executing training sequences, etc. before asserting the **dfi_init_complete** signal to the MC. The **dfi_phymstr_req** signal is not asserted as part of the boot sequence for training. However, when the MC asserts the **dfi_init_start** signal, the MC is also expected to de-assert the **dfi_reset_n** signal and/or drive the **dfi_cke** signal with an appropriate value in the $\mathbf{dfi_{boot\_state}}$ programmable parameter.

Figure 5, "MC De-Asserting dfi_reset_n and/or Asserting dfi_cke along with dfi_init_start" illustrates the MC changing the state of the **dfi_reset_n** signal and/or **dfi_cke** signal on the same cycle as the **dfi_init_start** signal is asserted.

FIGURE 5. *MC De-Asserting **dfi_reset_n** and/or Asserting **dfi_cke** along with **dfi_init_start***



Note: The MC is expected to drive the **dfi_cke** signal with appropriate value as per the state of the memory defined in the $\mathbf{dfi_{boot\_state}}$ programmable parameter.
Note: The PHY is responsible for driving the RESET and/or CKE IOs on the memory with appropriate values until the **dfi_init_complete** signal is asserted.

When the PHY asserts the **dfi_init_complete** signal, the MC may execute any command including MRR, MRW, ZQ, etc. The controller is NOT required to execute any of the commands. Any boot sequence commands that are not required prior to training are not defined by the DFI specification. Memory state information must be known including refresh state, bank states, low power state, etc. The memory must be given to the MC as defined in the $\mathbf{dfi_{boot\_state}}$ programmable parameter. Refer to Section 3.11, "WCK Control Interface" for the WCK state that must be maintained before the PHY asserts the **dfi_init_complete** signal during initialization for DRAMs which support WCK.

4.2.1 Refresh Requirements

Following the PHY boot sequence, the MC will assume the following regarding refresh:

- Refresh is enabled and must be maintained.
- The refresh count is not at a credit or deficit.
- The next refresh can be issued after tREFIab; if using Per Bank Refresh, a new sequence can be started and the first refresh can be issued after tREFIpb.
- All DFI timers, such as $\mathbf{t_{ctrlupd\_interval}}$, should start counting; no DFI commands should be required prior to the associated timer expiring.

The PHY must give the DFI to the controller without any refresh requirements - for example, the PHY cannot be in a refresh deficit.

4.2.2 Boot Into Self-Refresh

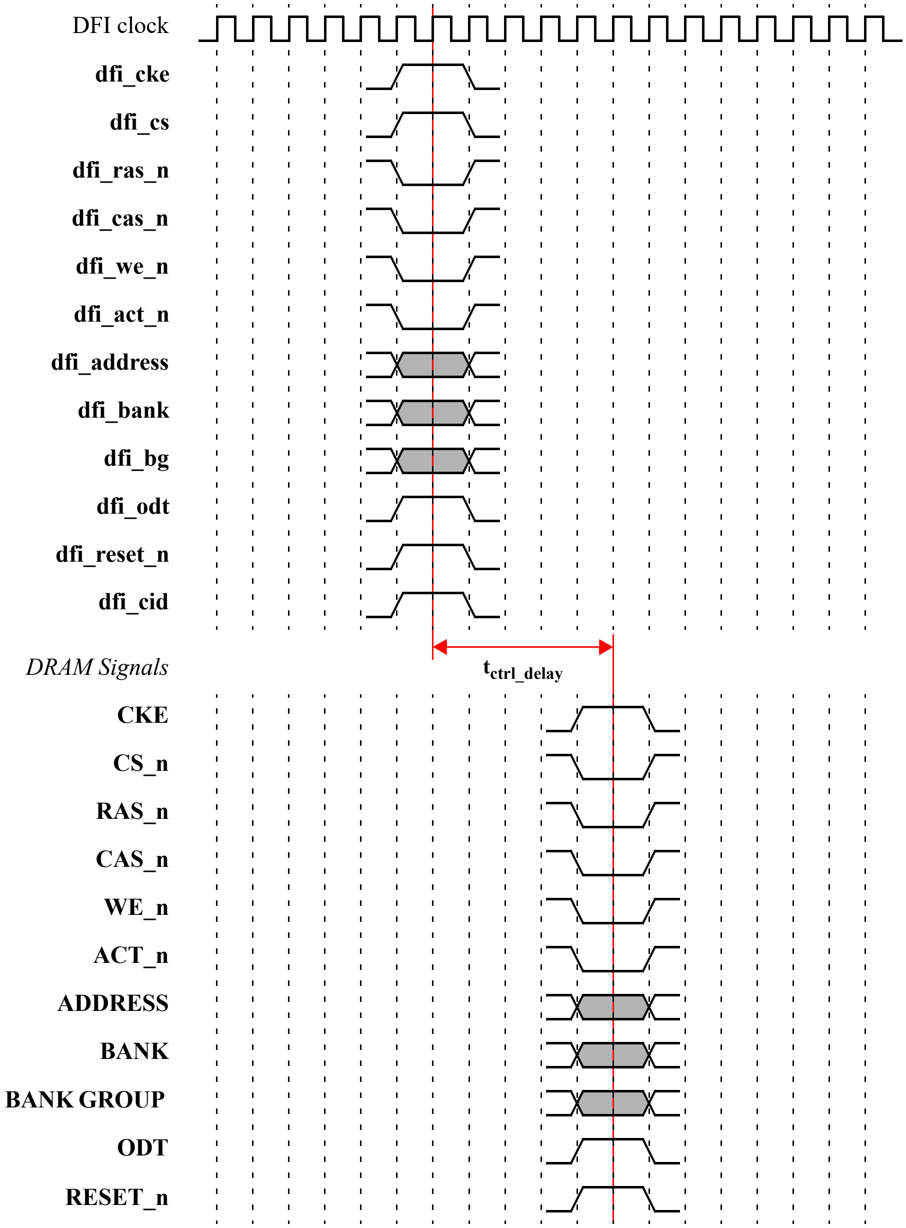
The $\mathbf{dfi_{boot\_state}}$ programmable parameter defines the state of memory as either Idle or Self-Refresh (but not Self-Refresh Power-down) when the PHY completes its boot sequence. The same refresh rules apply even when passing the memory back in the Self-Refresh state.

4.3 Command Interface Signals

The DFI command 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 command interface signals correlate to the DRAM command signals, and are driven according to the timing parameters $\mathbf{t_{ctrl\_delay}}$ and $\mathbf{t_{cmd\_lat}}$. For more information on command signals and parameters, refer to Section 3.1, "Command Interface".

Figure 6, "DFI Command Interface Signal Relationships" shows that the command signals are driven to the DRAMs and the DFI relationship of the command 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 command signals are illustrated but might not all be required, depending on system requirements. Refer to Table 4, “DFI Signal Requirements” to determine the signals that are relevant for a specific system.

FIGURE 6. *DFI Command Interface Signal Relationships*



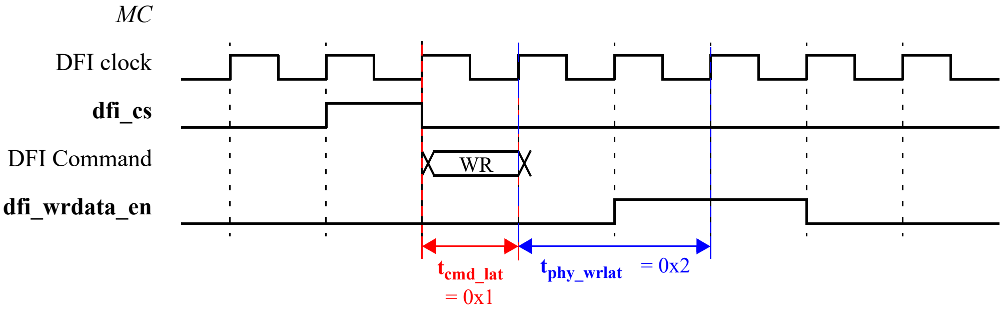
NOTE: The system might not use all of the pins on the DRAM interface, such as additional bank and chip selects. If these signals are never used in a system, they are not required on the DFI. However, if multiple memory sub-systems are supported, the union of the required signals must exist on DFI. In 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 7, "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. This figure also 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 9, "Command Interface Timing Parameters".

FIGURE 7. *Example of t_{cmd_lat} (t_{cmd_lat} = 1, t_{phy_wrlat} = 2)*

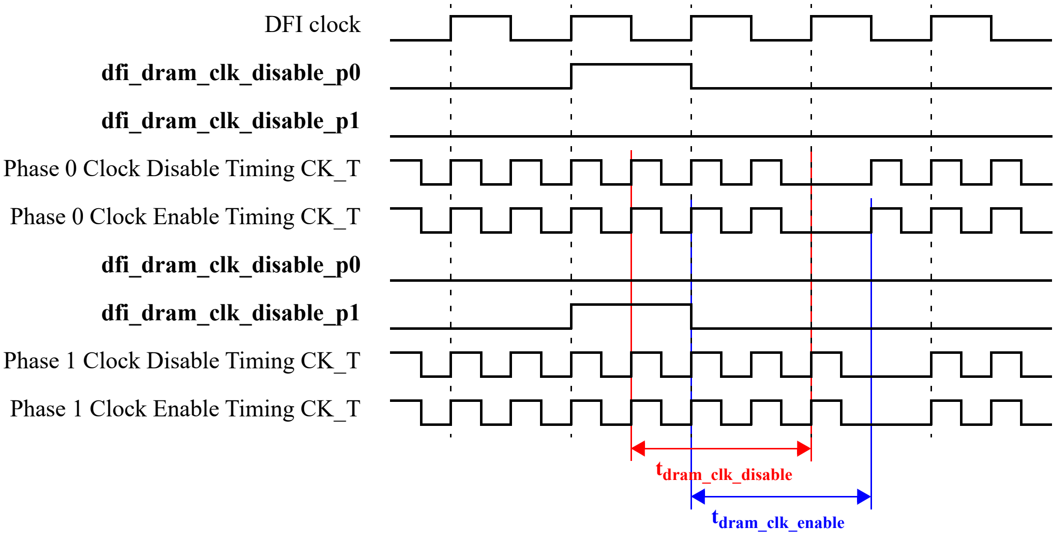


4.4 DFI Clock Disabling

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 8, "DRAM Clock Disable Behavior" shows that two timing parameters **$t_{dram\_clk\_disable}$** and **$t_{dram\_clk\_enable}$** 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 **$t_{dram\_clk\_disable}$** 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 8. *DRAM Clock Disable Behavior*



4.5 3DS Stack Support

The DFI specification supports 3D stacks. The signaling is dependent on memory type.

4.5.1 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 39. *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

4.5.2 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 40. *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

4.6 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 13, “Write Data Interface 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 **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**.

4.7 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}, **t$_{\mathbf{phy\_wrlat}}$** and **t$_{\mathbf{phy\_wrdata}}$** and the programmable parameters **phy$_{\mathbf{crc\_mode}}$** and **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 and MRW activities are to be limited to physical ranks. For more information on these signals, refer to Section 3.2, “Write Data Interface”.

4.7.1 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. **t$_{\mathbf{phy\_wrlat}}$** cycles elapse (**t$_{\mathbf{phy\_wrlat}}$** can be zero). (Change: was step 2, now a)

      The **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 **t$_{\mathbf{phy\_wrlat}}$** parameter is PHY-defined but may be specified in terms of other fixed system values.

      The write timing parameters (**t$_{\mathbf{phy\_wrdata}}$** and **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 **t$_{\mathbf{phy\_wrdata}}$** and **t$_{\mathbf{phy\_wrlat}}$** timing parameters work together to define the number of cycles from the assertion of a write command on the DFI command 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 **t$_{\mathbf{phy\_wrcslat}}$** timing parameter to specify the desired alignment of the command to the **dfi_wrdata_cs** signal; the PHY defines the **t$_{\mathbf{phy\_wrcsgap}}$** timing parameter to specify the additional delay it requires between two consecutive commands that are targeting different chip selects. The **t$_{\mathbf{phy\_wrcslat}}$** timing parameter must be held constant while commands are being executed on the DFI bus; however, if necessary, the **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 (**$\mathrm{dfi}_{\mathrm{rw\_length}}$**) plus the gap timing (**$t_{\mathrm{phy\_wrcsgap}}$**). The minimum time the chip select is guaranteed to remain driven to the PHY relative to the write command is defined by **$t_{\mathrm{phy\_wrcslat}}$** + **$\mathrm{dfi}_{\mathrm{rw\_length}}$** + **$t_{\mathrm{phy\_wrcsgap}}$**.

   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 (**$t_{\mathrm{phy\_wrlat}}$** - **$t_{\mathrm{phy\_wrcslat}}$**).

   The PHY must define the **$t_{\mathrm{phy\_wrcslat}}$** and **$t_{\mathrm{phy\_wrcsgap}}$** 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 **$t_{\mathrm{phy\_wrlat}}$** is met and remains asserted for the number of cycles required to complete the write data transfer sent on the DFI command interface; **$t_{\mathrm{phy\_wrlat}}$** 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 **$t_{\mathrm{phy\_wrlat}}$** is met and remains asserted for the entire length of the data stream; **$t_{\mathrm{phy\_wrlat}}$** 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. **$t_{\mathrm{phy\_wrdata}}$** 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 **$t_{\mathrm{phy\_wrdata}}$** and **$t_{\mathrm{phy\_wrlat}}$**.

   The **$t_{\mathrm{phy\_wrdata}}$** 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 **$t_{\mathrm{phy\_wrdata}}$** 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 **$t_{\mathrm{phy\_wrdata}}$** parameter may be increased. However, setting **$t_{\mathrm{phy\_wrdata}}$** 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 **$t_{\mathrm{phy\_wrdata}}$** parameter.

7. The **dfi_wrdata_en** signal de-asserts **$t_{\mathrm{phy\_wrdata}}$** cycles before the last valid data is transferred on the **dfi_wrdata** bus.
8. **$t_{\mathrm{phy\_wrdata\_delay}}$** 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)*

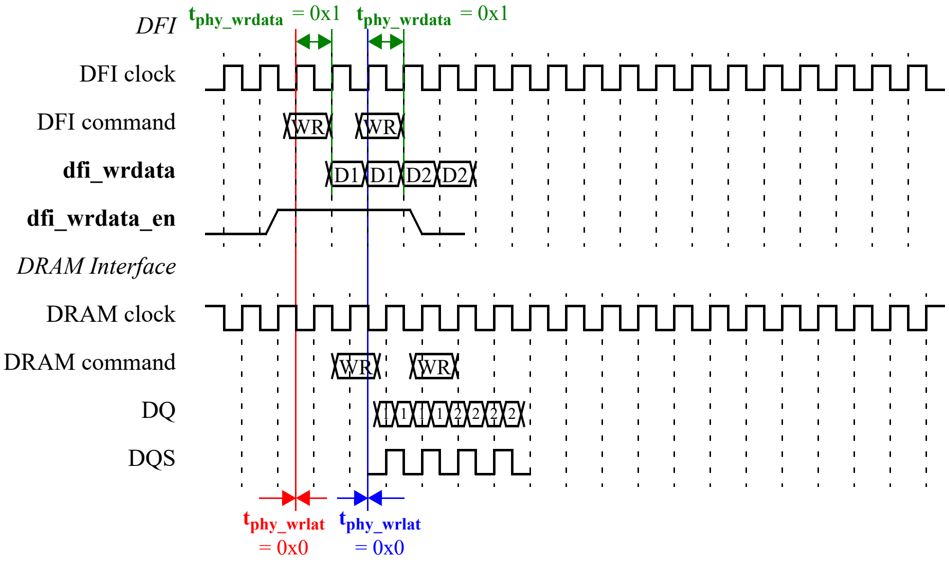


Figure 10, “Back-to-Back Interrupted Contiguous Writes (DRAM Burst of 8: tphy_wrlat = 3, tphy_wrdata = 2)” shows an interrupted write command. The **dfi_wrdata_en** signal should be asserted for 4 cycles for each of these write transactions. However, since the first write is interrupted, the **dfi_wrdata_en** 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)*

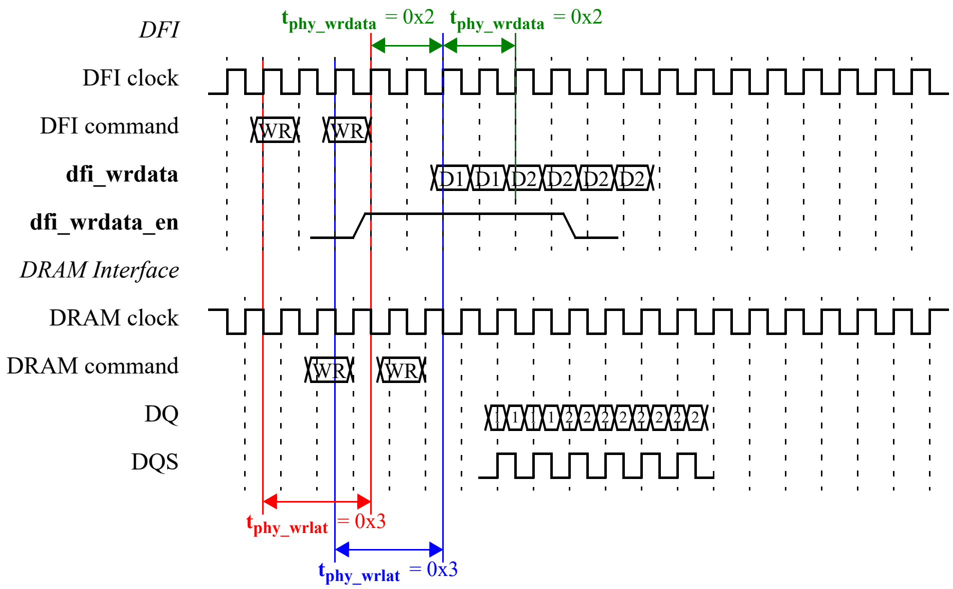


Figure 11, “Back-to-Back Writes (DRAM Burst of 8: tphy_wrlat = 4, tphy_wrdata = 4)” shows back-to-back burst-of-8 writes. The **dfi_wrdata_en** signal must be asserted for 4 cycles for each of these write transactions.

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

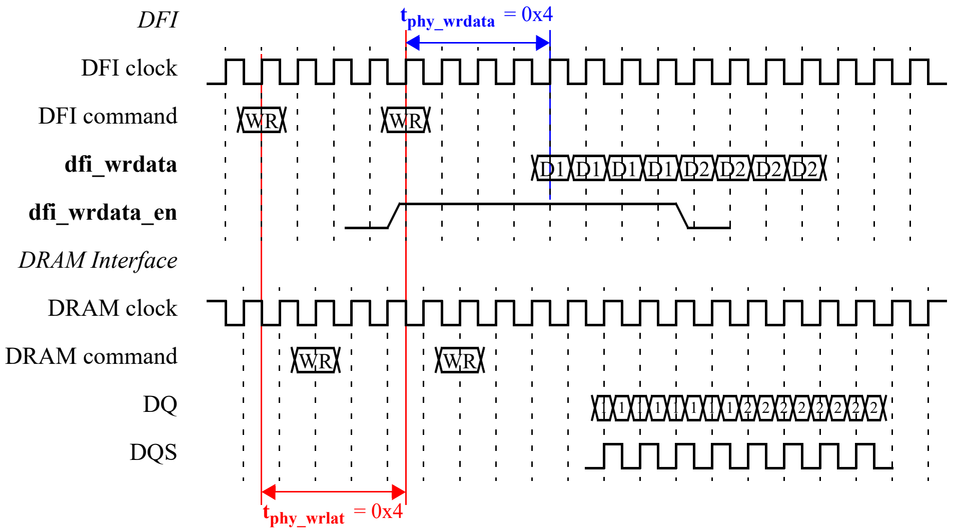


Figure 12, Figure 13, Figure 14 and Figure 15 also show two complete write commands, with different $\mathbf{t_{phy\_wrlat}}$ and $\mathbf{t_{phy\_wrdata}}$ timing parameters and for different DRAM types. The **dfi_wrdata_en** signal is asserted for two cycles for each write transaction. The t_{phy_wrlat} 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)*

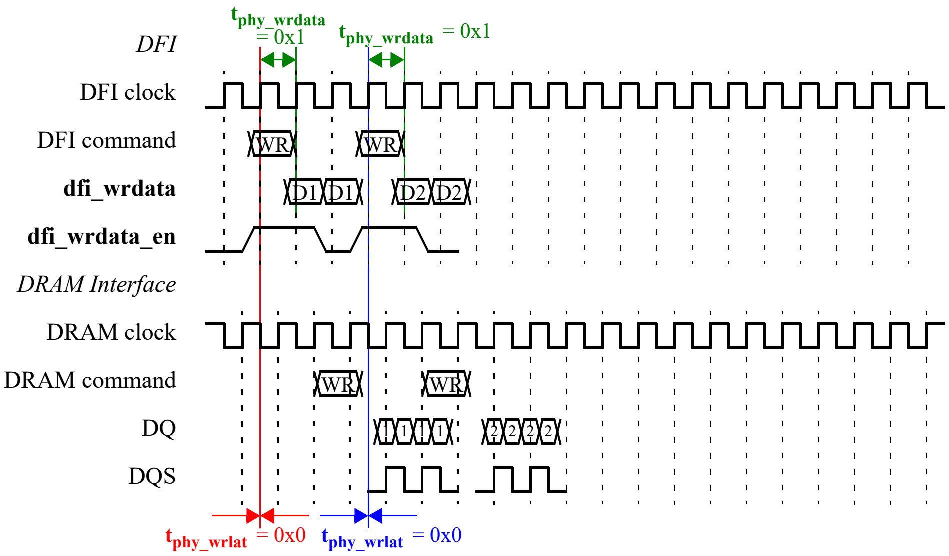


FIGURE 13. *Two Independent Writes (DRAM Burst of 4: t_{phy_wrlat} = 3, t_{phy_wrdata} = 1)*

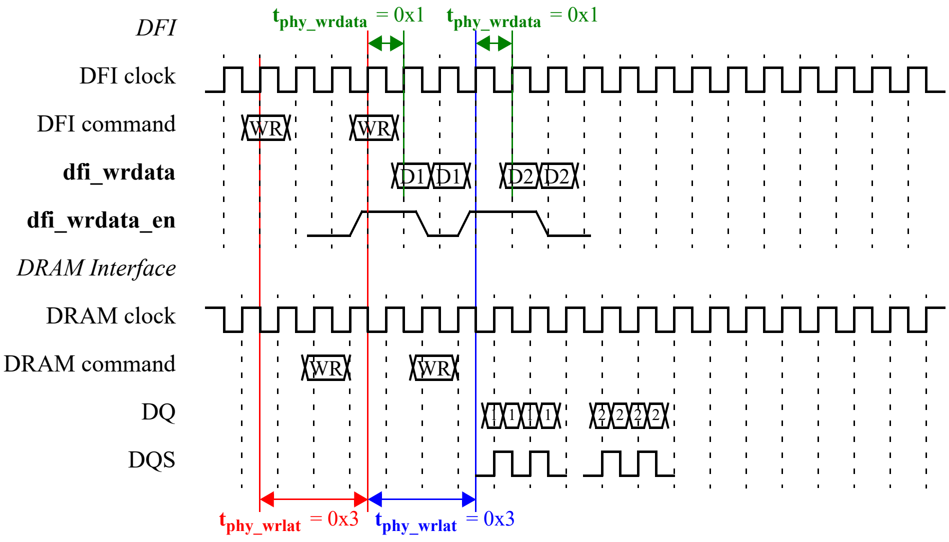


FIGURE 14. *Two Independent Writes (DRAM Burst of 8: t_{phy_wrlat} = 3, t_{phy_wrdata} = 3)*

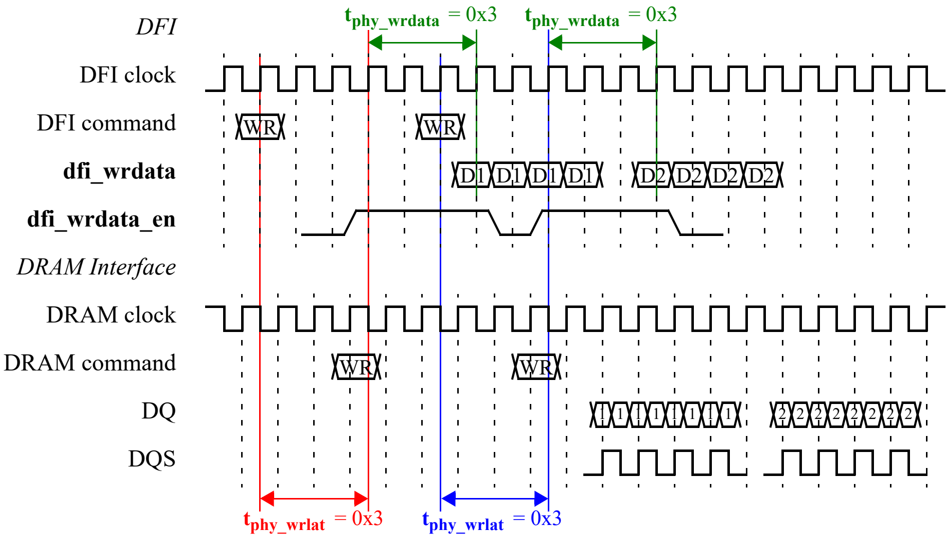


FIGURE 15. *Two Independent Writes (DRAM Burst of 4: t_{phy_wrlat} = 3, t_{phy_wrdata} = 4)*

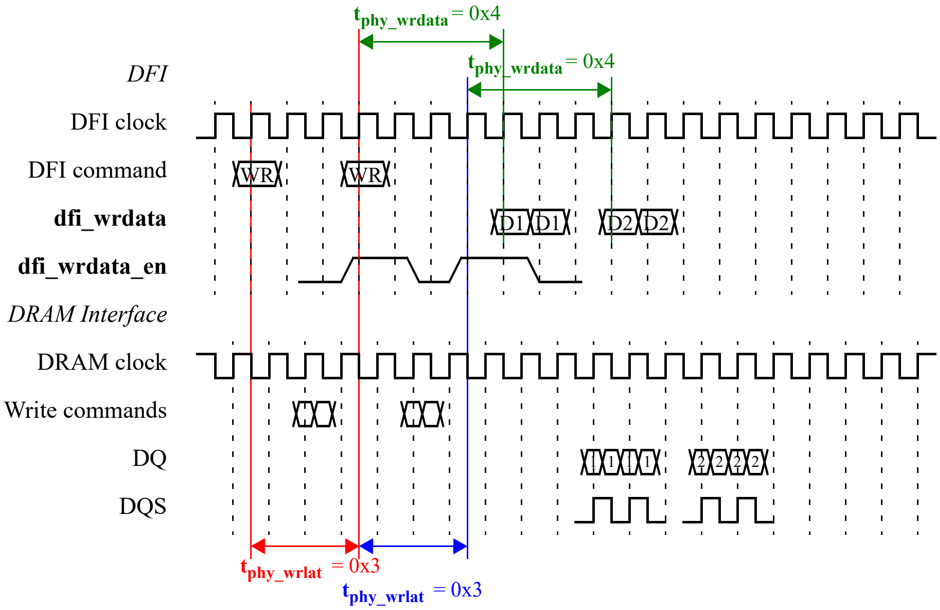
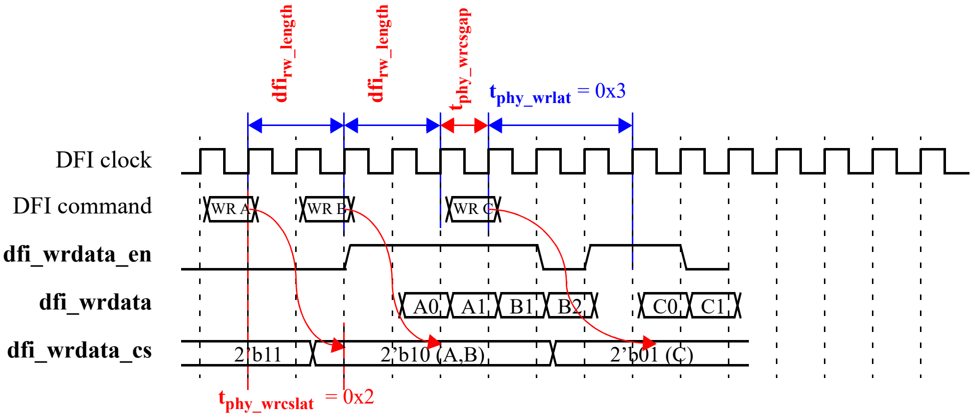


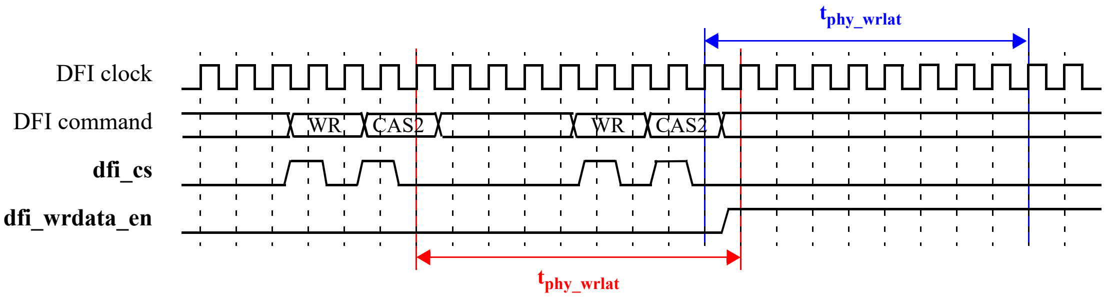
Figure 16, “Write Commands Utilizing dfi_wrdata_cs (DRAM Burst of 4: tphy_wrlat = 3, tphy_wrdat = 4, tphy_wrcslat = 2, tphy_wrcsgap = 1)” shows three write commands, with a gap between the second and third commands. The first two commands (WR A and WR B) address CS0 while the third one (WR C) addresses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

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*



4.7.2 DBI - Write

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

For more information on the **phy**$_{\mathbf{dbi\_mode}}$ parameter, refer to Table 12, “Write Data Interface 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.

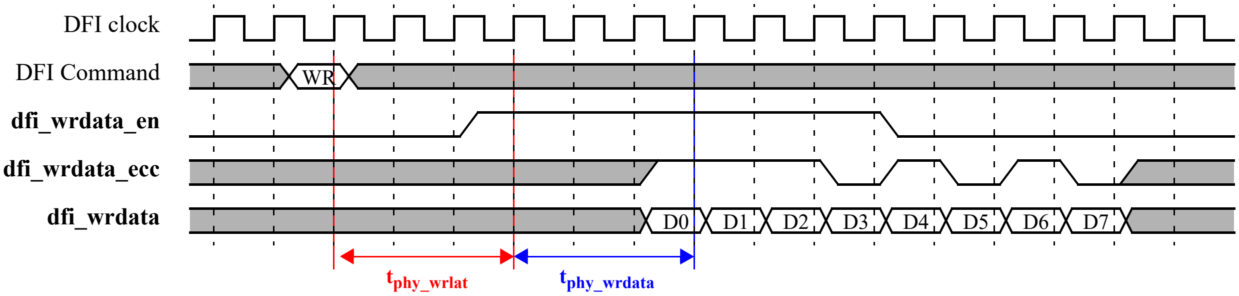
When both DBI and CRC are enabled in a system, special care needs to be taken regarding where DBI and CRC are performed in order to operate correctly. DBI inversion should take effect prior to CRC encoding.

4.7.3 Link ECC - Write

Link ECC is an optional DFI feature used with write data transmissions. The $\textbf{phy}_{\textbf{linkecc\_mode}}$ parameter is needed when Link ECC is supported in DFI. If the MC is handling link ECC operations, the MC will transmit the write ECC across the DFI bus on the **dfi_wrdata_ecc** signal. In Link ECC mode, the PHY is only required to transfer the write Link ECC through the PHY.

For more information on the $\textbf{phy}_{\textbf{linkecc\_mode}}$ parameter, refer to Section 3.13, “Link ECC”.

FIGURE 18. *Link ECC with Write Commands*



4.7.4 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 data 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 $\textbf{phy}_{\textbf{crc\_mode}}$ programmable parameter. The $\textbf{phy}_{\textbf{crc\_mode}}$ parameter is only needed when CRC is supported in DFI and uses the following definition to determine how CRC is handled:

- $\textbf{phy}_{\textbf{crc\_mode}}$ == 0 → CRC generation is handled in the MC
- $\textbf{phy}_{\textbf{crc\_mode}}$ == 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 41, “Systems Requiring CRC Support”.

TABLE 41. *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.

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

When the MC generates the CRC data, the $\mathbf{phy}_{\mathbf{crc\_mode}}$ == 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.

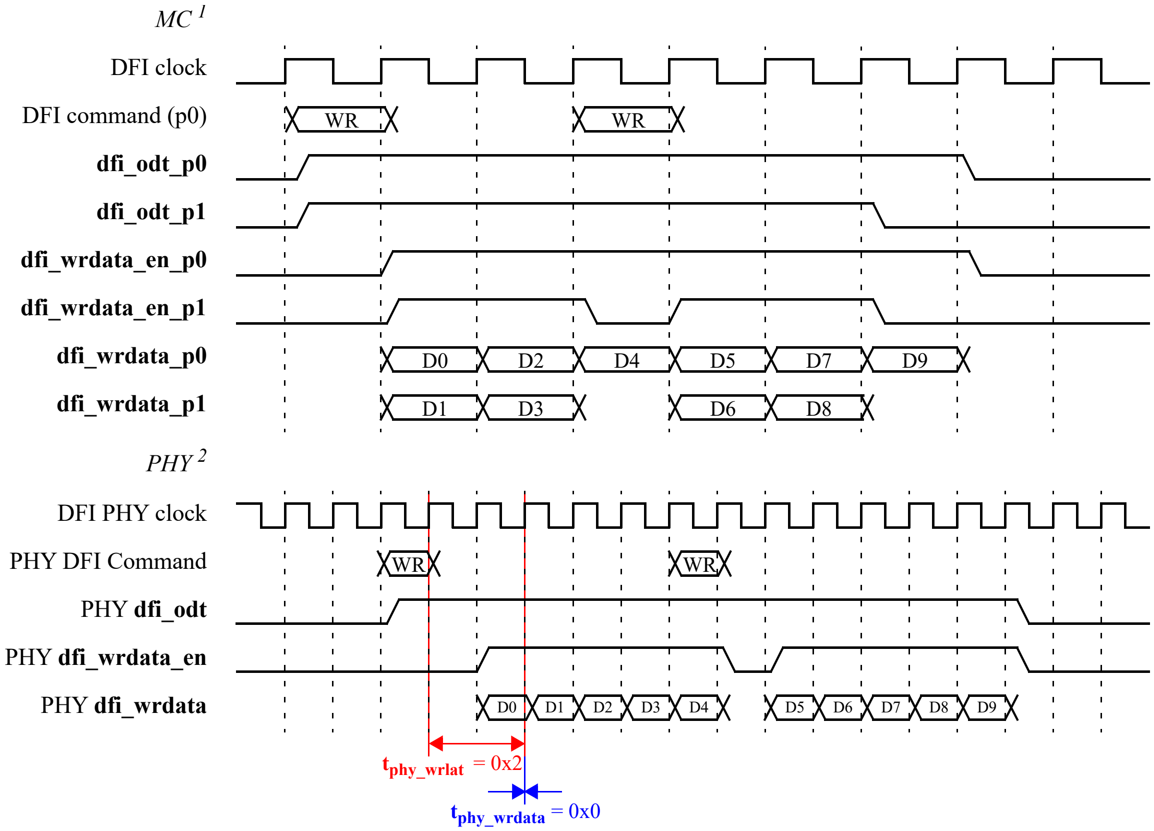
DFI dictates that when CRC is used, the CRC data word must be incorporated in the burst of write data, but DFI does not dictate placement within the burst. The specific ODT timing requirements are dependent on the chip selects accessed and system architecture. The **dfi_odt** assertion and de-assertion 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 19, “DFI Write Data Bus for MC CRC Support Mode (Two Bursts starting in Phase 0)”
- Figure 20, “DFI Write Data Bus for MC CRC Support Mode (Two Back-to-Back Bursts)”
- Figure 21, “DFI Write Data Bus for PHY CRC Support Mode”

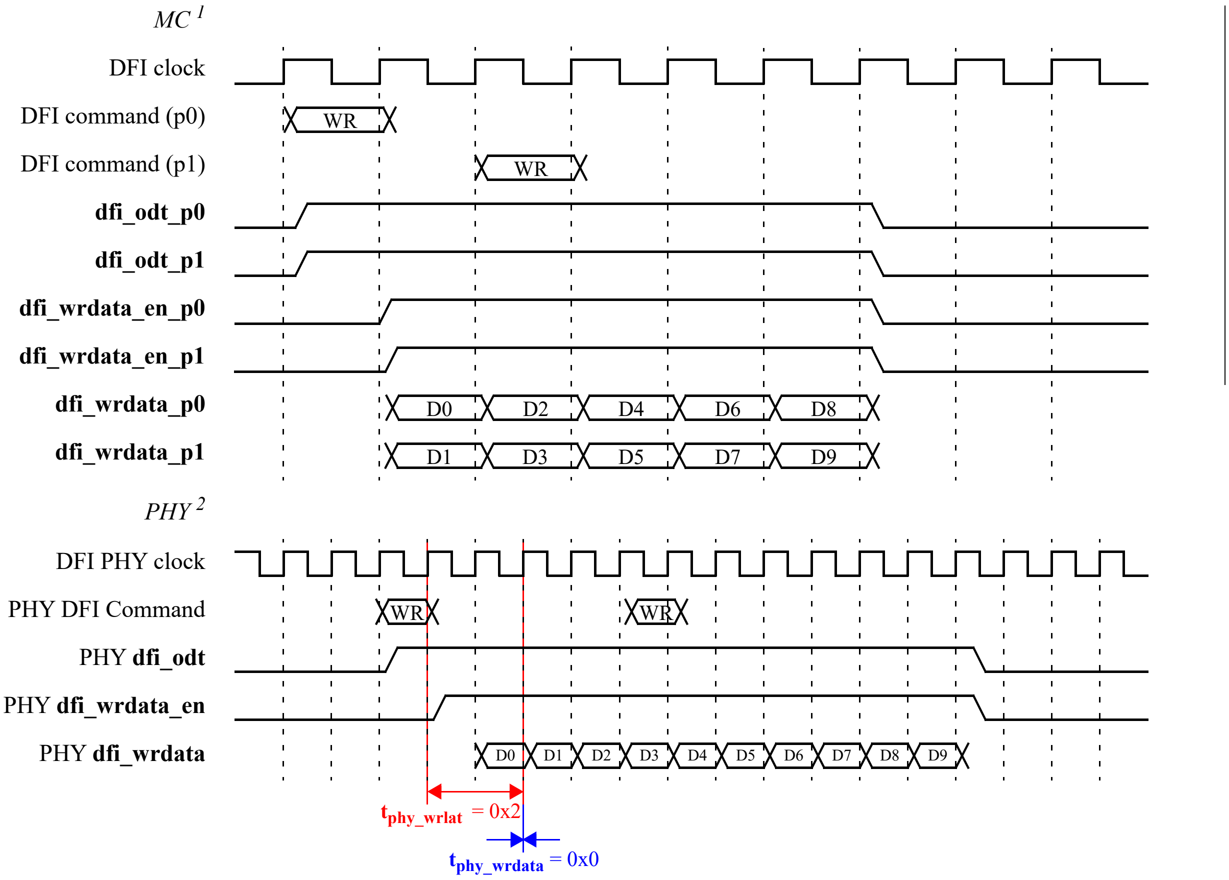
Figure 19 and Figure 20 illustrate the DFI write data bus for a 2:1 frequency ratio system with CRC extending a burst of 8 by1 additional clock DRAM cycle.

FIGURE 19. *DFI Write Data Bus for MC CRC Support Mode (Two Bursts starting in Phase 0)*



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



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.

4.7.4.2 PHY CRC Support ($\mathbf{phy}_{\mathbf{crc\_mode}}$ = 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.

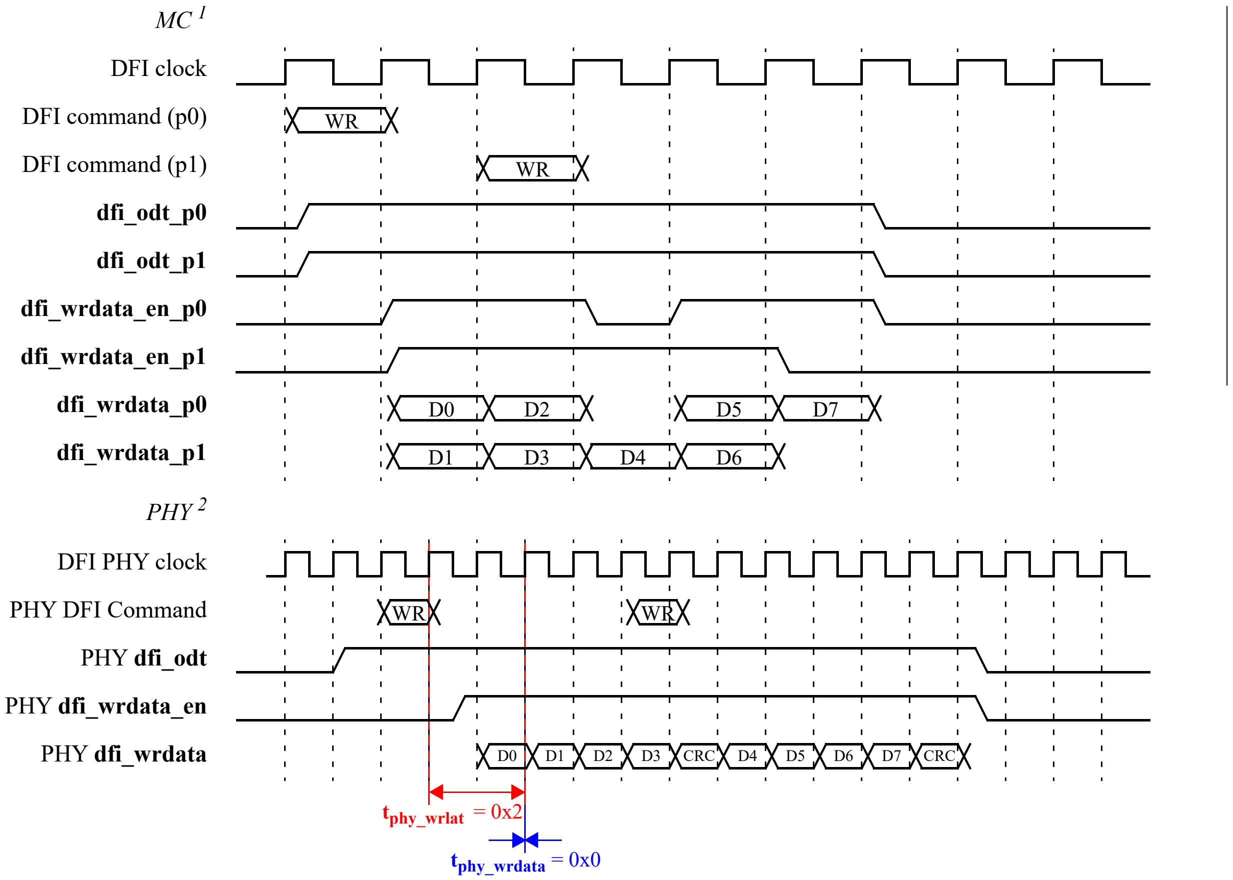
4.7.4.3 Burst Chop 4 with PHY CRC Support

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 21 and Figure 22 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.

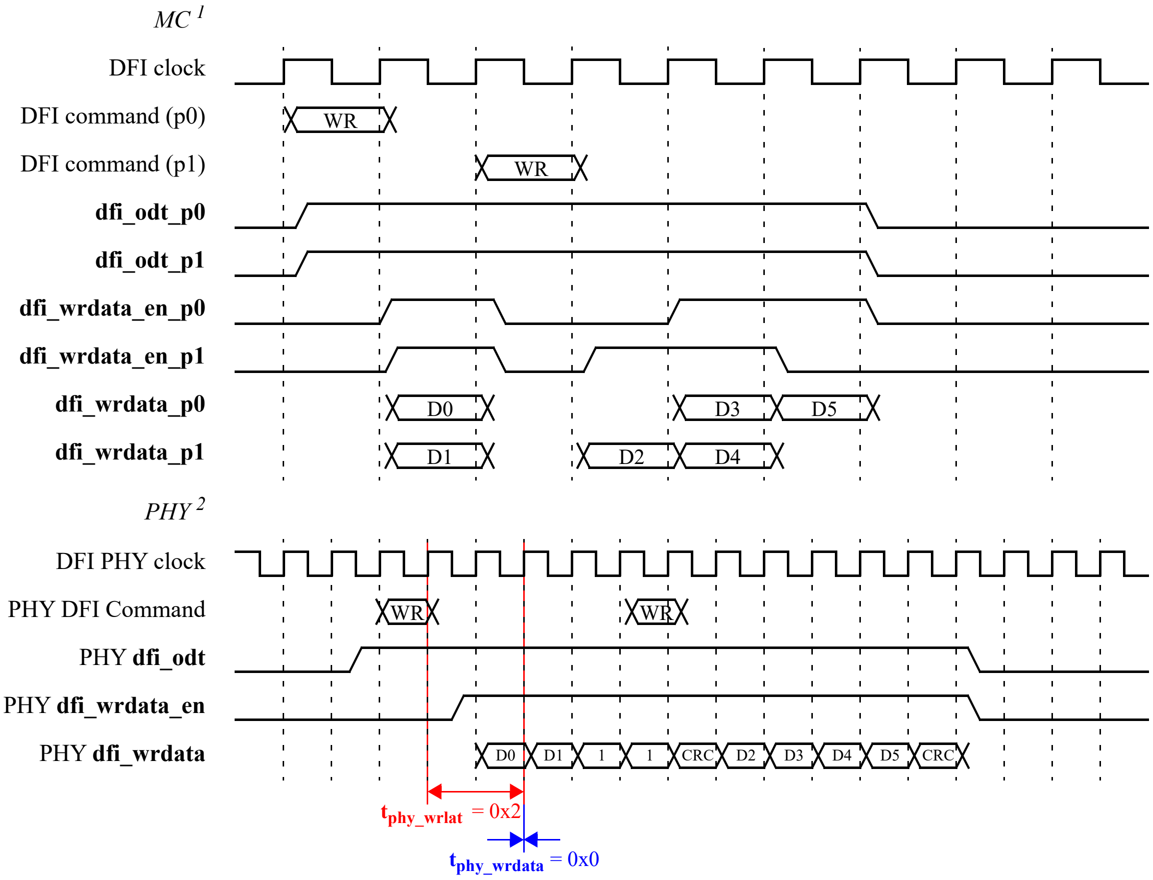
Figure 21, “DFI Write Data Bus for PHY CRC Support Mode” illustrates a write BC4 case. This example shows a BC4 write followed by a burst of 8 write. With BC4, the MC only transmits 2 data words across the DFI bus. For both BC4 and burst of 8 commands, the CRC data is transmitted at the end of the burst, in UI9 and UI10.

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



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



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.

4.7.5 CA Bus Timing

For memories with a CA bus including LPDDR2, LPDDR3, LPDDR4, LPDDR5, and DDR5, the timing parameters referencing the command are referenced to the corresponding clock on the DFI bus as defined by JEDEC as the timing reference point for the command to the memory clock on the memory bus. The following table summarizes the timing reference points for these memories and the corresponding DFI point of reference. The same rule applies to all memories including those not listed in Table 42, "CA Bus Timing Summary".

TABLE 42. *CA Bus Timing Summary*

DRAM Type	CA Command Definition	Memory Command Timing Reference	DFI Command Timing Reference
LPDDR2 / LPDDR3	1 clock DDR command [a]	Rising edge of the memory clock of the command	Rising edge of the DFI command clock of the command

TABLE 42. *CA Bus Timing Summary*

DRAM Type	CA Command Definition	Memory Command Timing Reference	DFI Command Timing Reference
LPDDR4	4 clock SDR command: {CMD-1, CMD-1, CAS-2, CAS2} where CMD is RD or WR	Rising edge of the memory clock of the 2nd CAS-2 command when CS is de-asserted	Rising edge of the DFI command clock of the 2nd CAS-2 command
LPDDR5	1 clock DDR command1: {CMD_r, CMD_f} where CMD is RD or WR Note: _r, _f refer to the rising and falling edges of the memory clock	Rising edge of the memory command clock of the command	Rising edge of the DFI command clock of the command
DDR5 - 1N	2 clock SDR command: {CMD_L,CMD_H} where CMD is RD or WR Note: _L, _H refer to CS_n	Rising edge of 2nd memory clock of the command when CS is de-asserted	2nd rising edge of the DFI command clock of the command
DDR5 - 2N	4 clock SDR command: {CMD1_L,CMD1_H,CMD2_H,CMD2_H} where CMD1 and CMD2 are RD or WR Note: L, H refer to CS_n	Rising edge of the 1st memory clock of the 2nd command	Rising edge of the DFI command clock of the 2nd command

a. For DDR commands, the rising and falling edge memory command is concatenated and sent in a single clock or phase on the interface.

The following diagrams illustrate the timing reference relative to the command. For all timing diagrams, the command from which the timing parameter is referenced is highlighted. In the timing diagrams, the timing parameters are referenced from the rising edge of the clock when the command is sampled.

FIGURE 23. *LPDDR2 / LPDDR3 Read Timing with a 1:1 Frequency Ratio*

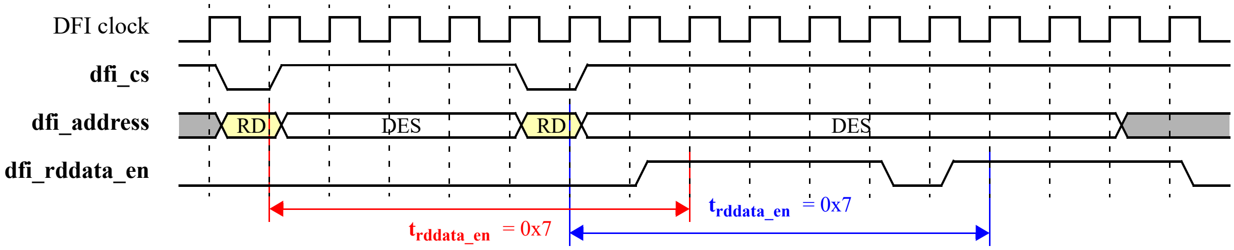


FIGURE 24. *LPDDR4 Read Timing with a 1:2 Frequency Ratio*

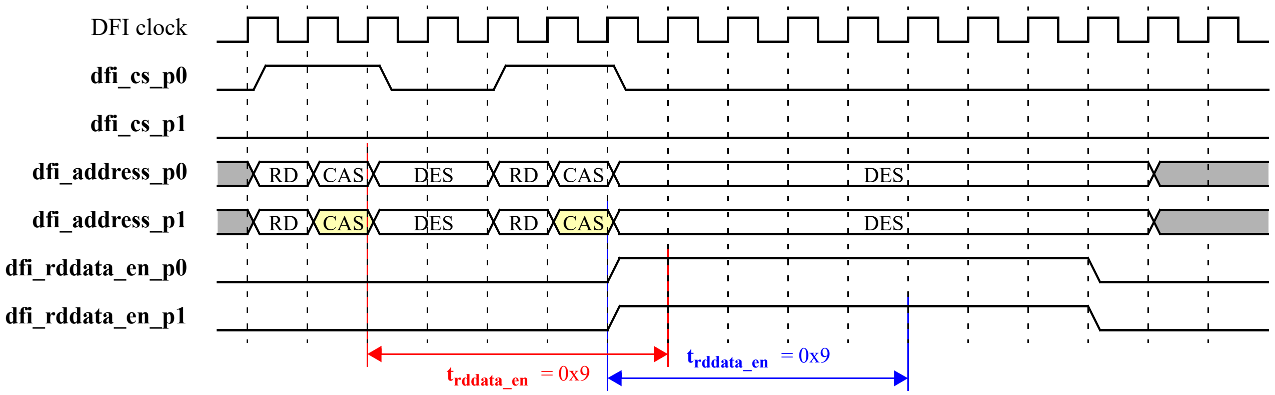
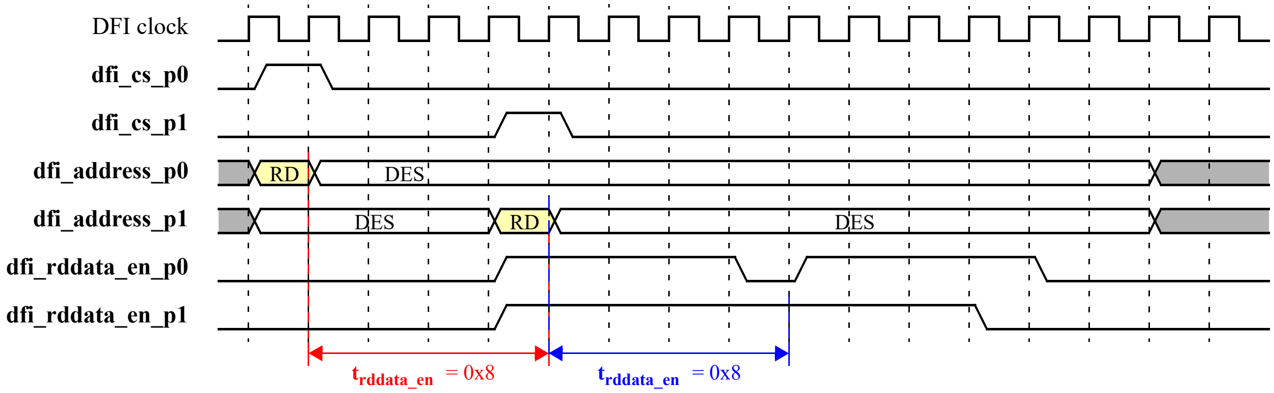
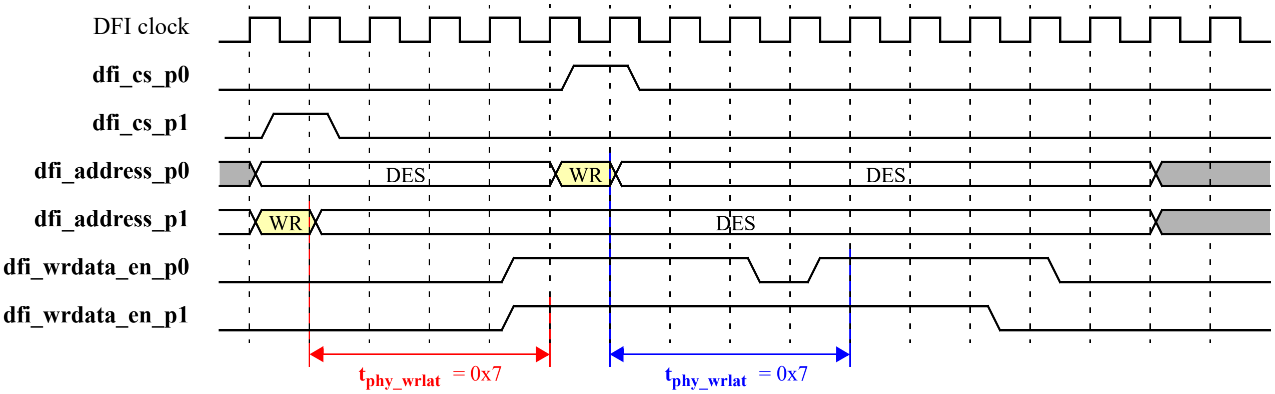


FIGURE 25. *LPDDR5 Read Timing*



RD includes both the rising and falling edge read command for the DDR read command.

FIGURE 26. *LPDDR5 Write Timing*



WR includes both the rising and falling edge write command for the DDR write command.

FIGURE 27. *DDR5 1N Read Timing*

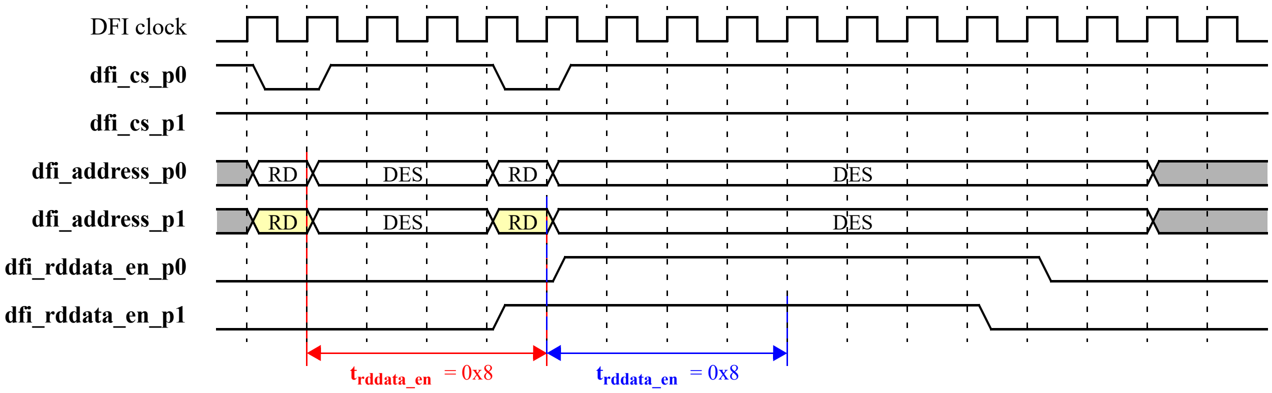


FIGURE 28. *DDR5 1N Write Timing*

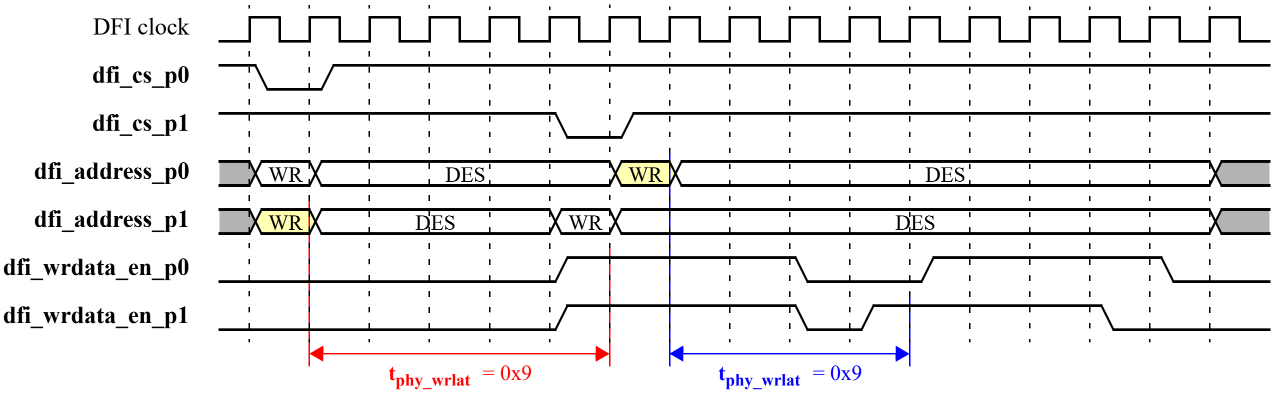


FIGURE 29. *DDR5 2N Read Timing*

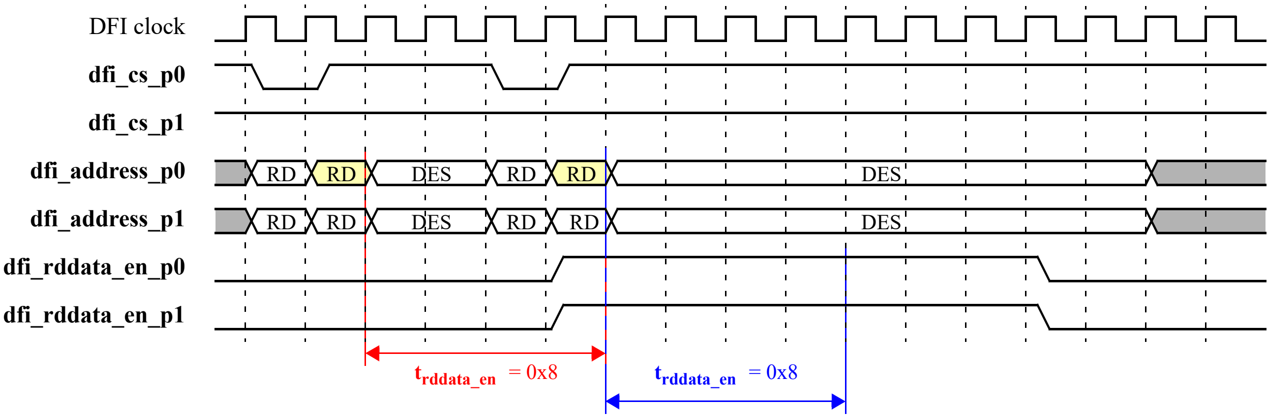
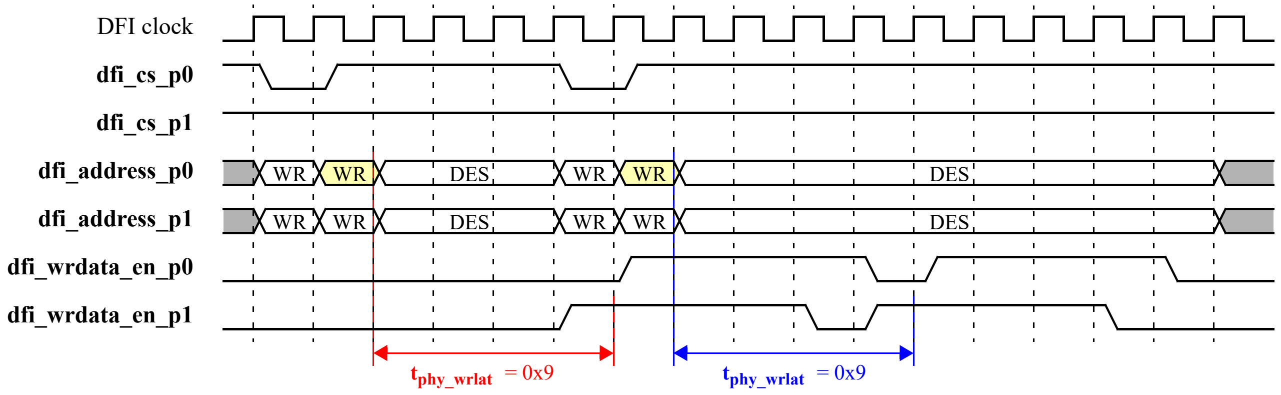


FIGURE 30. *DDR5 2N Write Timing*



4.8 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**), read data inversion (**dfi_rddata_dbi**), 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.

4.8.1 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 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, 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, 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).

   The minimum time that the MC will continue to drive the **dfi_rddata_cs** signal to the PHY relative to the read command is defined by $t_{phy\_rdcslat} + dfi_{rw\_length} + t_{phy\_rdcsgap}$. The additional time needed by the PHY between read commands to different chip selects to make internal adjustments due to changing the target chip select is defined by the $t_{phy\_rdcsgap}$ timing parameter. The PHY must define the $t_{phy\_rdcslat}$ and $t_{phy\_rdcsgap}$ 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.
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 31, Figure 32, Figure 33, Figure 34, Figure 35, Figure 36, Figure 37, Figure 38, Figure 39, Figure 40 and Figure 41.

Figure 31, "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 t_{phy_rdlat} cycles after the **dfi_rddata_en** signal assertion.

FIGURE 31. *Single Read Transaction of 2 Data Words*

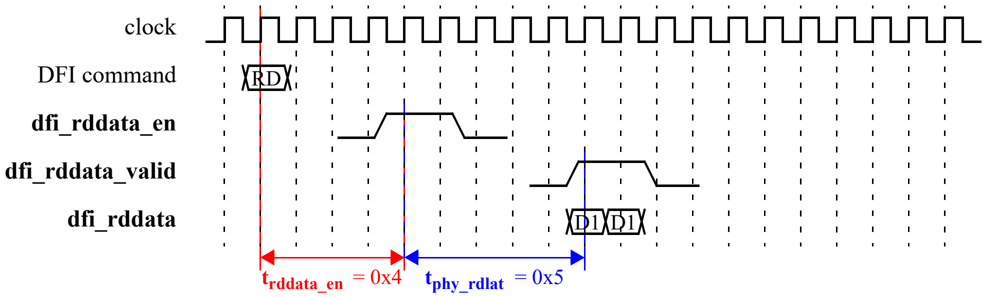


Figure 32, “Single Read Transaction of 4 Data Words” shows a single read transaction where the data is returned in less than the maximum delay. The data returns one cycle less than the maximum PHY read latency.

FIGURE 32. *Single Read Transaction of 4 Data Words*

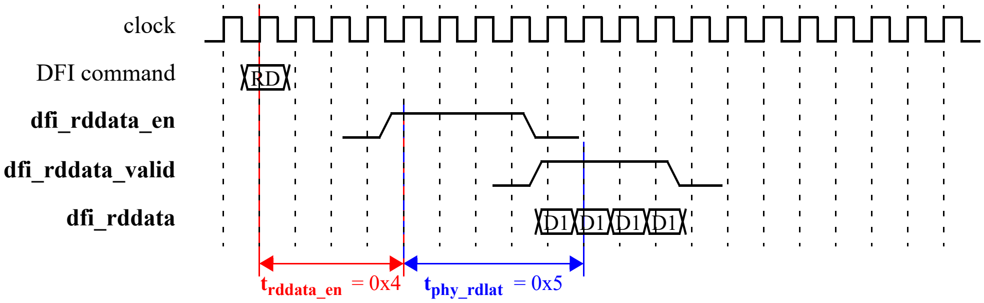


Figure 33, “Back-to-Back Read Transactions with First Read Burst Interrupted (DDR1 Example BL = 8)” shows an interrupted read command. The **dfi_rddata_en** signal must be asserted for 4 cycles for each of these read transactions. However, since the first read is interrupted, the **dfi_rddata_en** signal 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 33. *Back-to-Back Read Transactions with First Read Burst Interrupted (DDR1 Example BL = 8)*

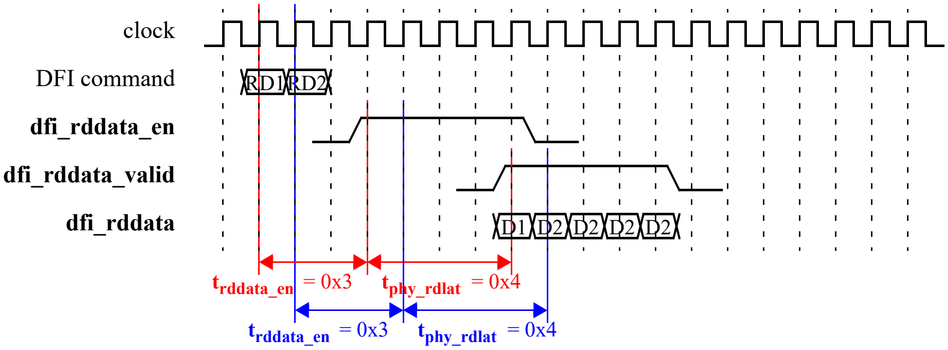
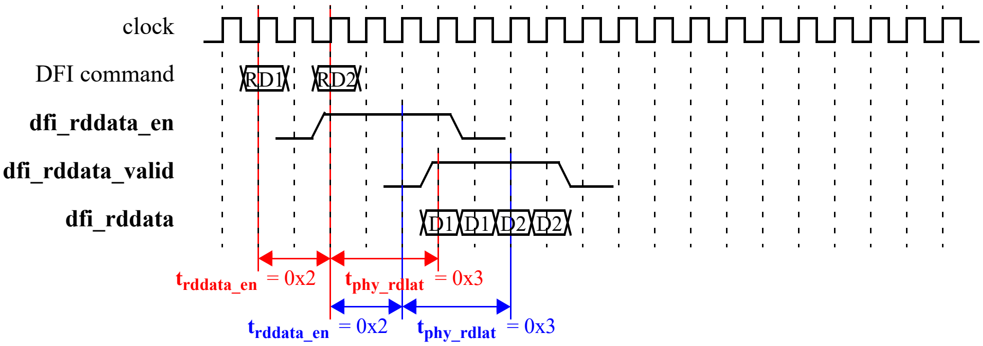


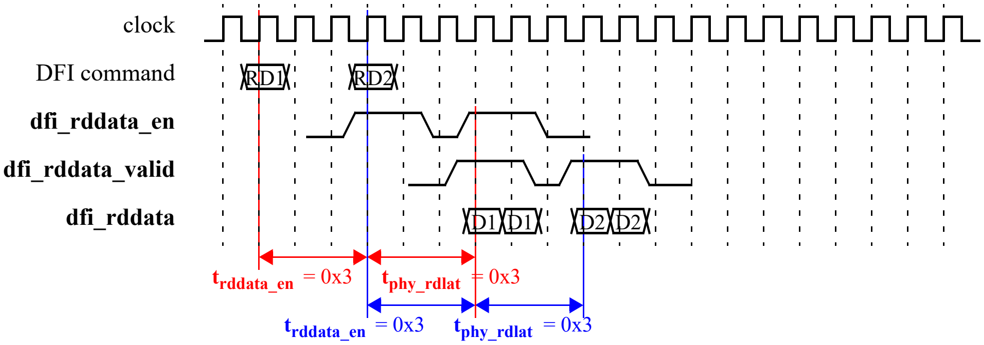
Figure 34, “Two Independent Read Transactions (DDR1 Example)” and Figure 35, “Two Independent Read Transactions (DDR2 Example)” also show two complete read transactions. The **dfi_rddata_en** signal is asserted for two cycles for each read transaction. In Figure 34, “Two Independent Read Transactions (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 34. *Two Independent Read Transactions (DDR1 Example)*



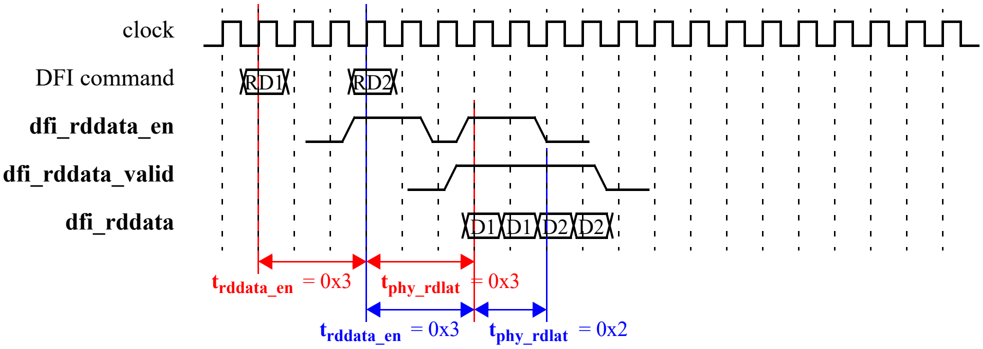
In Figure 35, “Two Independent Read Transactions (DDR2 Example)”, the t_{rddata_en} timing and the timing between the read commands causes the **dfi_rddata_en** signal to be de-asserted between commands. As a result, the **dfi_rddata_valid** signal is de-asserted between commands and the **dfi_rddata** stream is non-contiguous.

FIGURE 35. *Two Independent Read Transactions (DDR2 Example)*



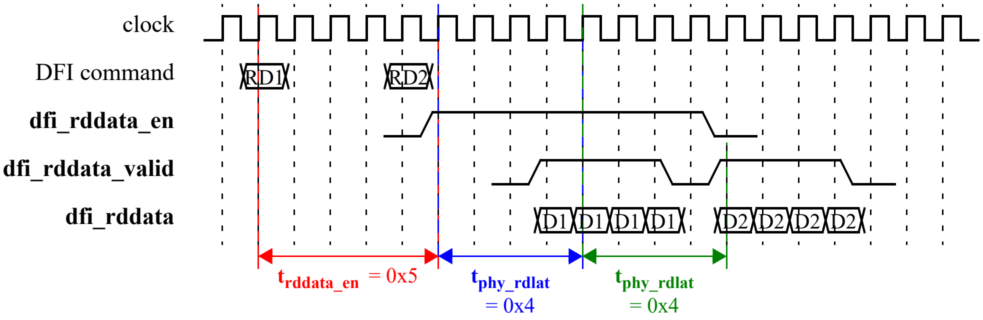
In Figure 36, “Two Independent Read Transactions (DDR2 Example)”, the effective **t_{phy_rdlat}** for the two transactions is different. This results in a situation in which the **dfi_rddata_valid** signal remains asserted across commands and the **dfi_rddata** stream is contiguous.

FIGURE 36. *Two Independent Read Transactions (DDR2 Example)*



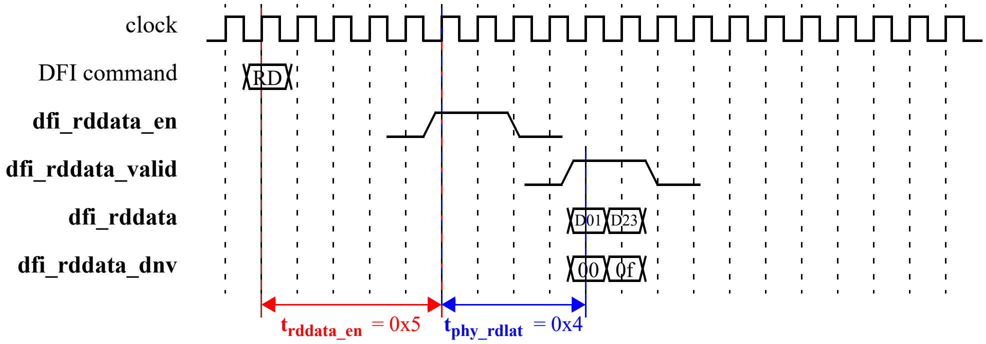
The data may return to the DFI in fewer cycles than maximum delay. In Figure 37, “Two Independent Read Transactions (DDR3 Example)”, the first read data transfer is returned in three cycles, even though the **t_{phy_rdlat}** timing parameter is set to four cycles. The second read data transfer is returned in the maximum of four cycles.

FIGURE 37. *Two Independent Read Transactions (DDR3 Example)*



Some DRAMs define a 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 38, “Example MRR Transactions with LPDDR2” shows an MRR transaction for a LPDDR2 memory device.

FIGURE 38. *Example MRR Transactions with LPDDR2*



In Figure 39, “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 39. *DFI Read Data Transfer Illustrating **dfi_rddata_valid** Definition*

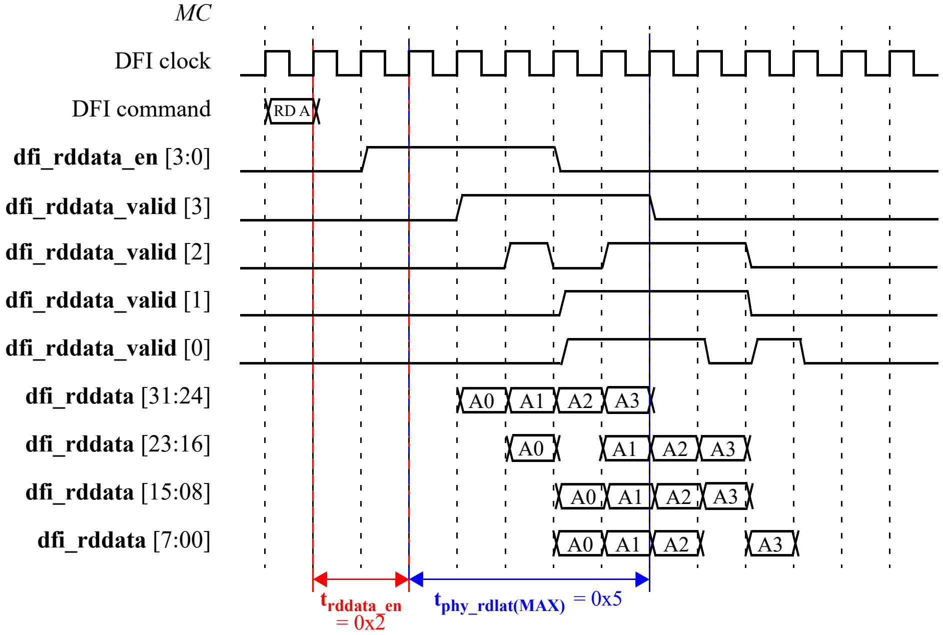
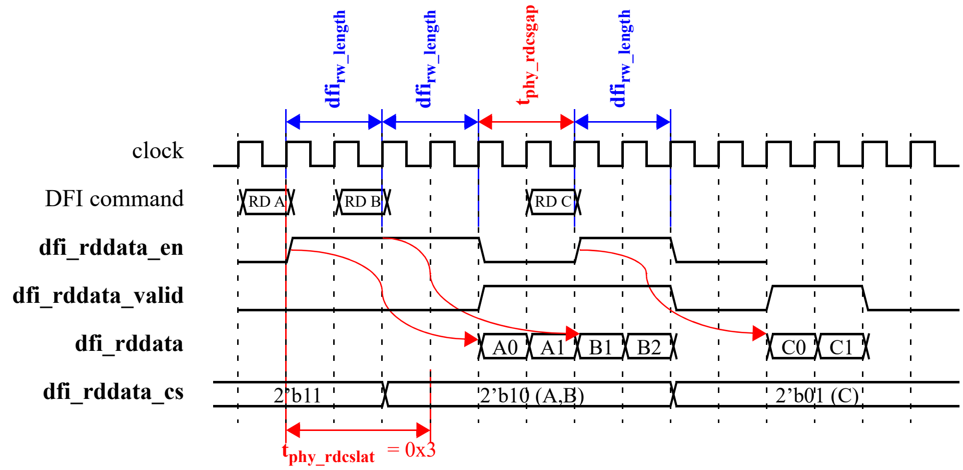


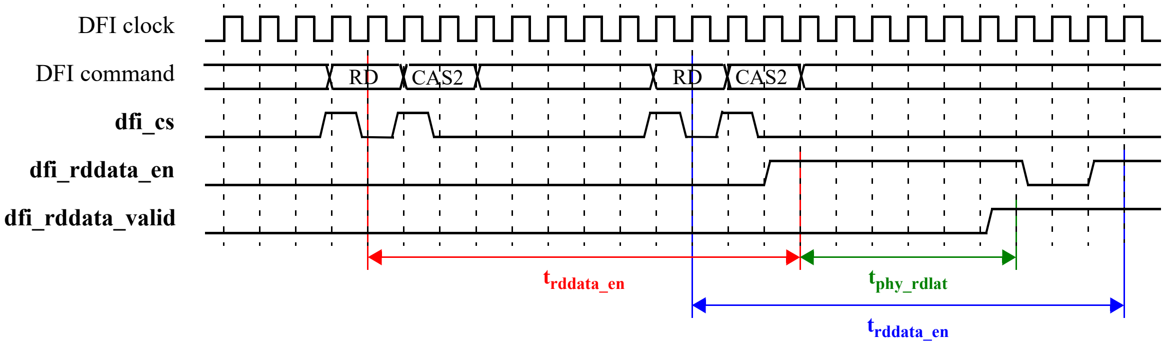
Figure 40, “Read Commands Utilizing dfi_rddata_cs” shows three read commands, with a gap between the second and third commands.

FIGURE 40. *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 41. *LPDDR4 Read Command*



4.8.2 DBI - Read

DBI is an optional DFI feature used with read data transmissions. The **phy**$_{\mathbf{dbi\_mode}}$ 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 **phy**$_{\mathbf{dbi\_mode}}$ parameter, refer to Table 13, “Write Data Interface Programmable Parameters”.

4.8.2.1 MC DBI Support (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 11, “Write Data Interface Signals” and Table 14, “Read

Data Interface 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.

4.8.2.2 PHY DBI Support (phy$_{\mathrm{dbi\_mode}}$ = 1)

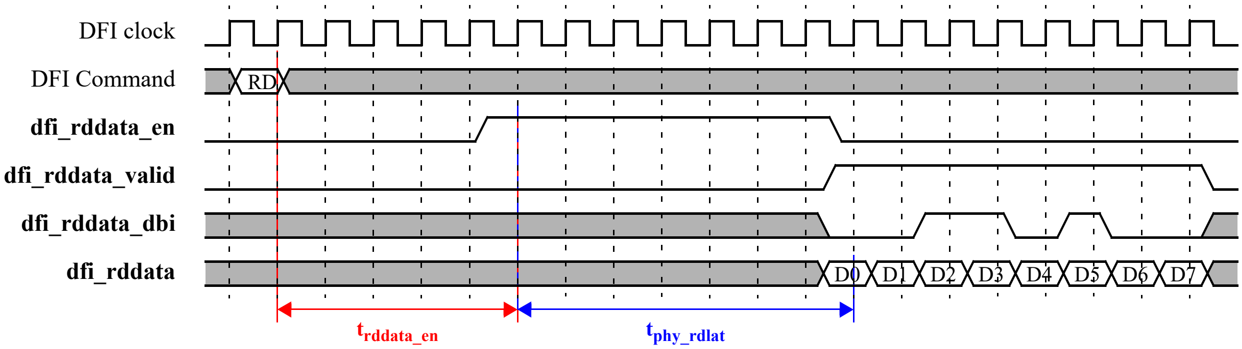
When the PHY generates the write DBI data, the **dfi_rddata_dbi** signal is not needed.

4.8.3 Link ECC - Read

Link ECC is an optional DFI feature used with read data transmissions. The **phy**$_{\mathbf{linkecc\_mode}}$ parameter is needed when Link ECC is supported in DFI. If the MC is handling link ECC operations, the DRAM will transmit the write ECC across the DFI bus on the **dfi_rddata_dbi** signal. In Link ECC mode, the PHY is only required to transfer the read Link ECC through the PHY.

For more information on the **phy**$_{\mathbf{linkecc\_mode}}$ parameter, refer to Section 3.13, “Link ECC”.

FIGURE 42. *Link ECC with Read Commands*



4.9 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 9, “Command Interface Timing Parameters” and Table 12, “Write Data Interface Timing Parameters”.

4.9.1 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_req** 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 43, "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 43. *MC-Initiated Update Timing Diagram*

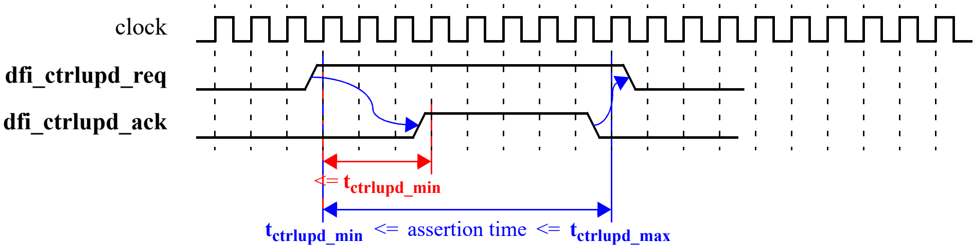
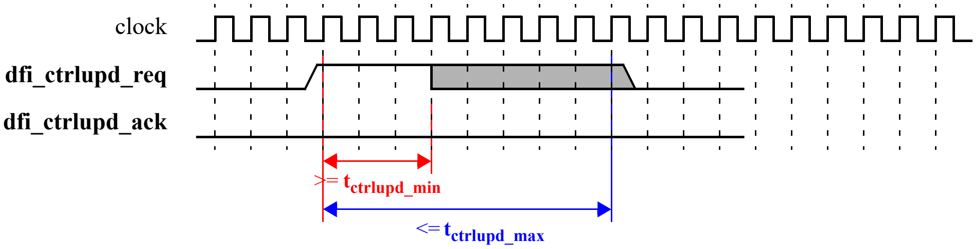


Figure 44, "MC-Initiated Update with No Response" shows the important point that the **dfi_ctrlupd_ack** signal is not required to assert when the **dfi_ctrlupd_req** signal is asserted. The MC must assert the **dfi_ctrlupd_req** signal for at least $\mathbf{t_{ctrlupd\_min}}$ within every $\mathbf{t_{ctrlupd\_interval}}$ cycle, but the total number of cycles that the **dfi_ctrlupd_req** signal is asserted must not exceed $\mathbf{t_{ctrlupd\_max}}$.

FIGURE 44. *MC-Initiated Update with No Response*



4.9.2 PHY-Initiated Update

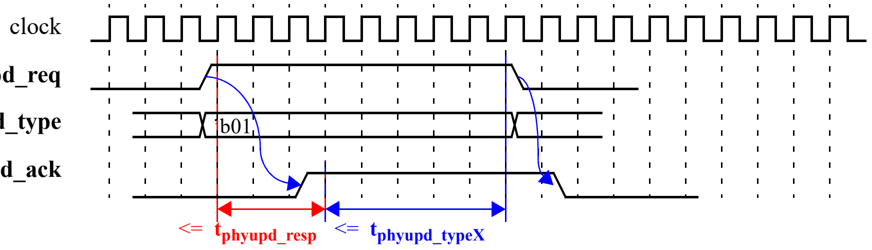
The PHY may also trigger the DFI to enter an idle state. This update process utilizes three signals: **dfi_phyupd_req**, **dfi_phyupd_type** and **dfi_phyupd_ack**. The **dfi_phyupd_req** signal indicates the need for idle time on the DFI, the **dfi_phyupd_type** signal defines the type of update required, 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 command, read data 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 in most cases. Exceptions are granted if **dfi_init_start**, **dfi_phymstr_req** or **dfi_ctrlupd_req** are active along with **dfi_phyupd_req**. Refer to Section 4.21, "DFI Interactions"

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 45, "PHY-Initiated Update Timing Diagram" shows that the MC must respond to a PHY update request, unlike MC-initiated updates shown in Figure 43, "MC-Initiated Update Timing Diagram" and Figure 44, "MC-Initiated Update with No Response".

FIGURE 45. *PHY-Initiated Update Timing Diagram*



4.9.3 DFI Bus Idle

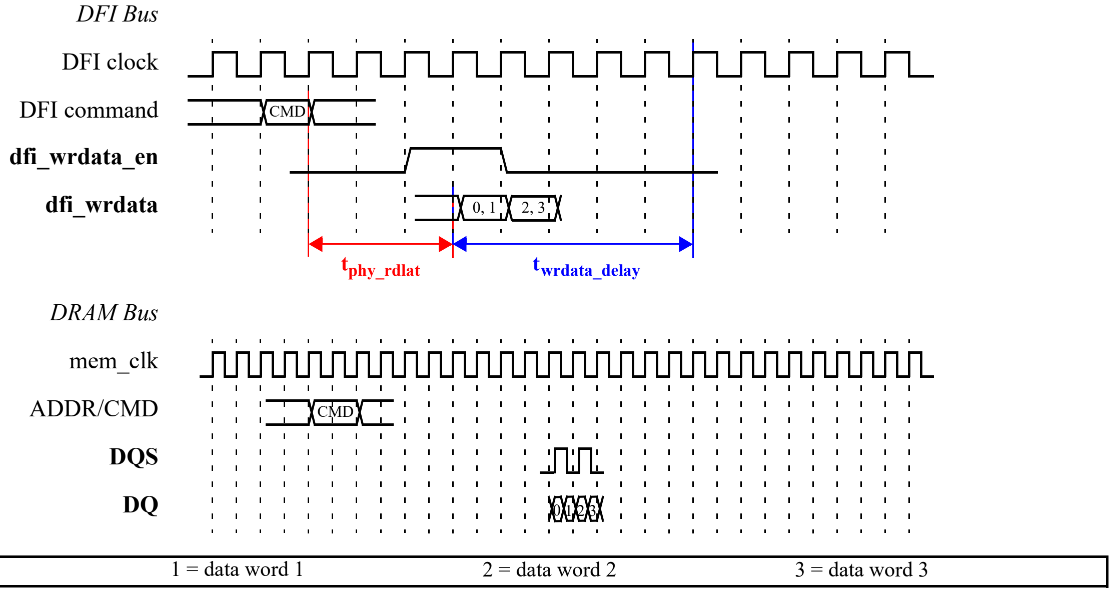
To prevent the condition where a PHY may disrupt the write data transfer on the memory bus, the write data transfer on the memory bus must complete before the DFI bus idle state is reached.

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 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 46. *DFI Bus Idle State Timing Parameter - t_{wrdata_delay}*



4.10 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 data interface operates at a multiple of the MC frequency, the PHY transfers data at a higher data rate relative to the DFI clock. If the PHY command interface operates at a multiple of the MC frequency, the PHY 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 DFI clock. DFI defines 3 clock domains: the DFI clock domain, the DFI command clock domain and the DFI data clock domain. For DRAMs that operate command and data from the same memory clock frequency, the command and data clocks are the same. For DRAMs with separate command and data clocks, these clocks are different.

The DFI specification supports a 1:1, 1:2 or 1:4 MC to PHY clock frequency ratio, defining the relationship of the reference clocks for the MC and the PHY. For DRAMs with a common command and data clock, these clocks may operate at any one of the frequency ratios. When operating at a frequency ratio, the PHY address and data clock are commonly referred to as the DFI PHY clock. For DRAMs with different address and data clocks, the command clock is always the same as the DFI clock and the data clock may operate at any one of the frequency ratios. The DFI data clock is always the same frequency as the DRAM clock, which is 1/2 the data rate for the memory. For DRAMs with a common command and data clock, the DFI PHY clock is the same frequency as the DRAM clock. In the following sections defining frequency ratios across DFI, the PHY clock is commonly referred to as the DFI PHY clock.

DFI signals may be sent or received by the PHY from any of the defined clock domains, 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 4.12.3, “CA Parity and CRC Errors in Frequency Ratio Systems”.

4.10.1 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 47, “Frequency Ratio 1:2 Phase Definition” and Figure 48, “Frequency Ratio 1:4 Phase Definition” show the clock phase definitions for 1:2 and 1:4 frequency ratio systems.

FIGURE 47. *Frequency Ratio 1:2 Phase Definition*

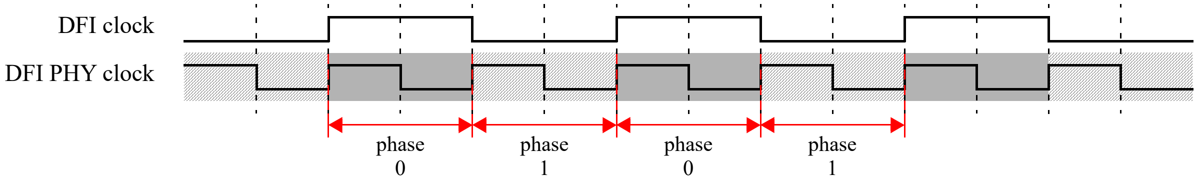
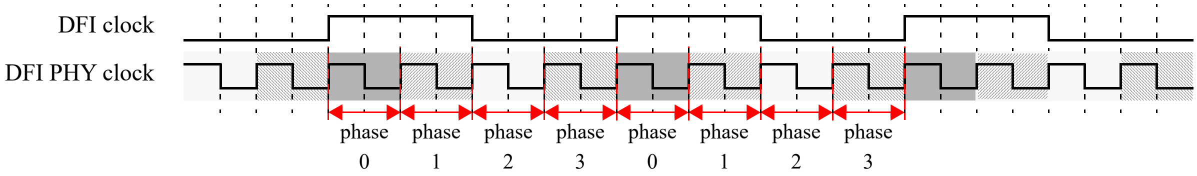


FIGURE 48. *Frequency Ratio 1:4 Phase Definition*



4.10.2 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 command 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 4.10.4, "Read Data Interface in Frequency Ratio Systems". The phase 0 or DFI data word 0 suffixes are not required.

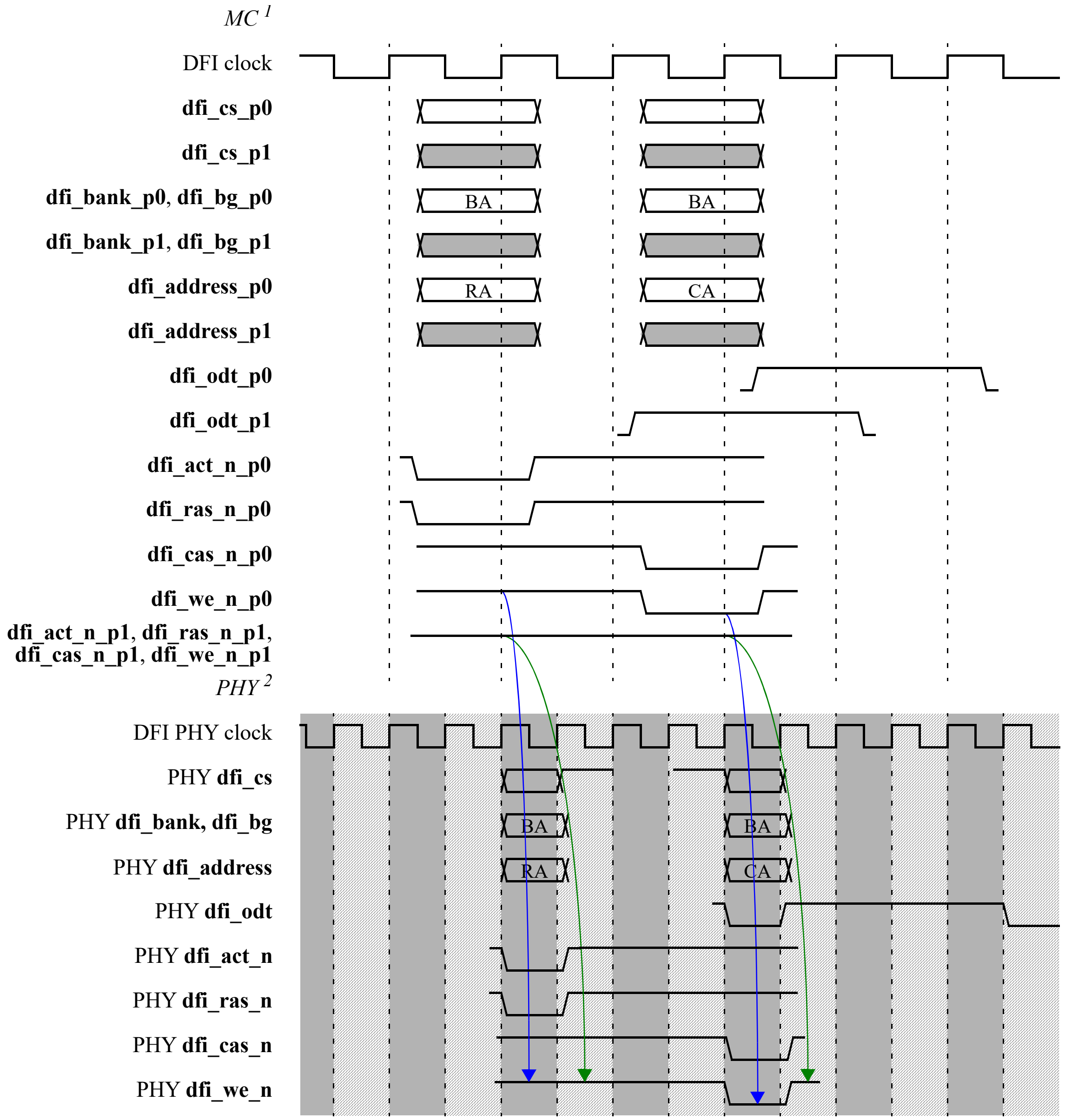
There is flexibility in system setup for frequency ratio systems. The MC may be implemented to support command output on a single phase or on multiple phases. Even if multiple phases are supported, the MC is not required to implement or drive every phase of a signal. Only phases where a command is sent 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 49, "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 49. *Example 1:2 Frequency Ratio Command Stream*

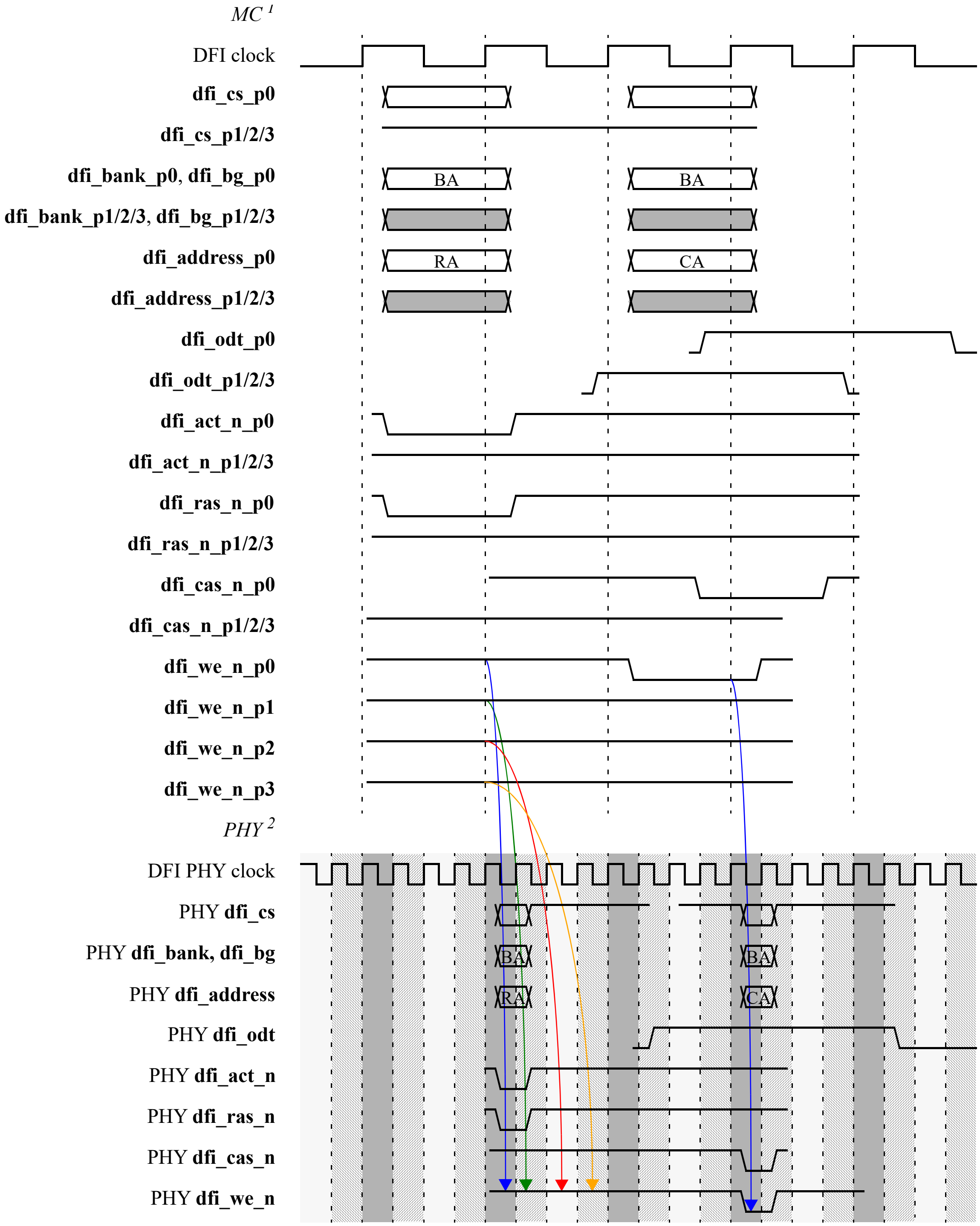


■ DFI PHY clock phase 0 ▨ DFI PHY clock phase 1

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 50, "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 50. *Example 1:4 Frequency Ratio Command Stream*



▩ DFI PHY clock phase p0 ▨ DFI PHY clock phase p1 □ DFI PHY clock phase p2 ▦ DFI PHY clock phase p3

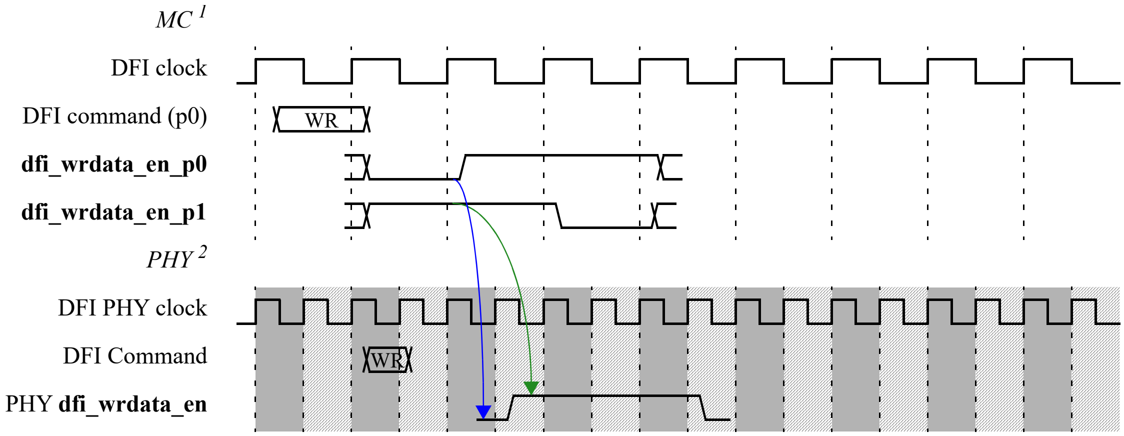
This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.
1. In the MC clock diagram, the timing shows the DFI bus signaling.
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is shown for illustrative purposes only.

4.10.3 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 51, "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 51. *1:2 Frequency Ratio Write Data Example*



■ DFI PHY clock phase p0	▨ DFI PHY clock phase p1

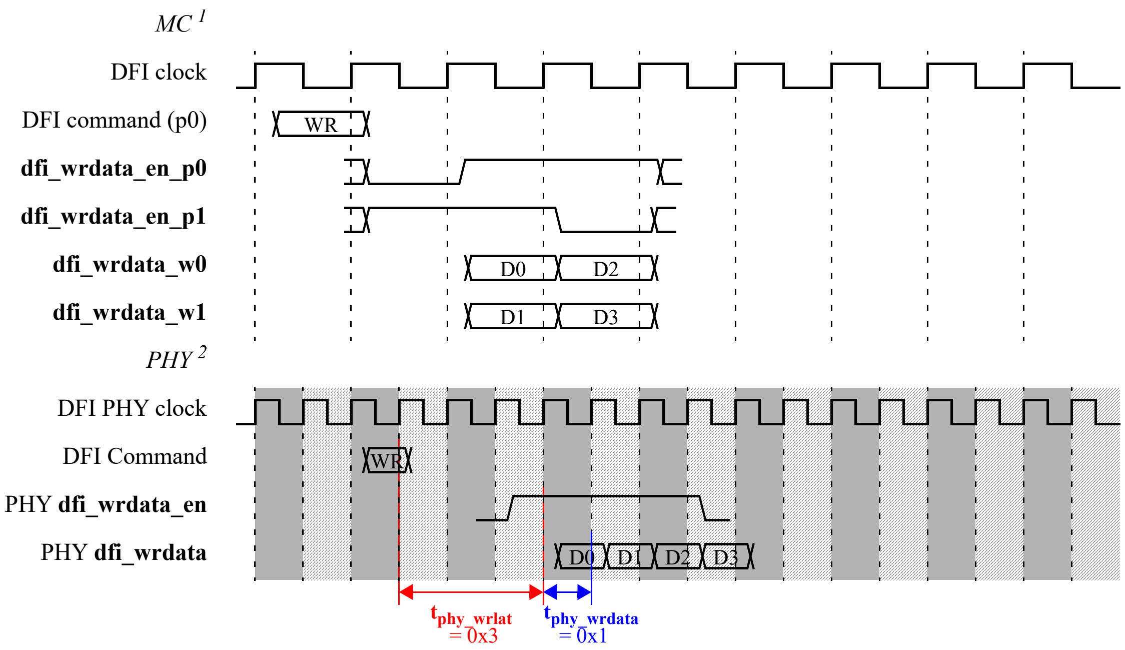
This timing diagram includes both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain.
1. In the MC clock diagram, the timing shows the DFI bus signaling.
2. In the PHY clock diagram, the timing illustrates how the PHY interprets the DFI bus. PHY timing is 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 52, “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 52. *1:2 Frequency Ratio Aligned Write Data Example*

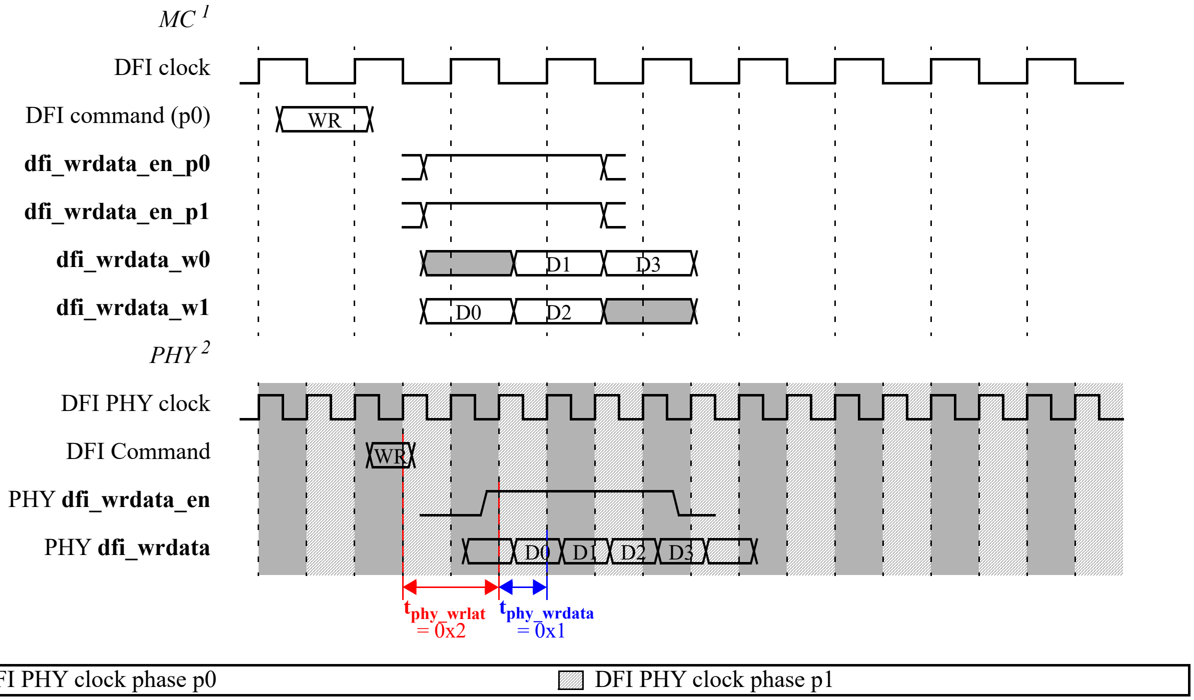


■ DFI PHY clock phase p0 ▨ DFI PHY clock phase p1

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 53, “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.

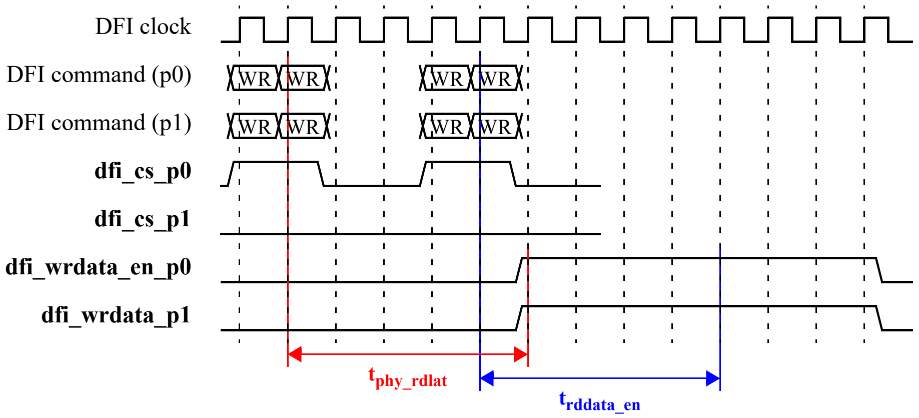
FIGURE 53. *1:2 Frequency Ratio Aligned Write Enable Example*



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 54. *LPDDR4 Write Command, 1:2 Frequency Ratio*



4.10.4 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. 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. Therefore, any assertion of the **dfi_init_start** signal must trigger a re-initialization of the DFI data word pointer to 0.

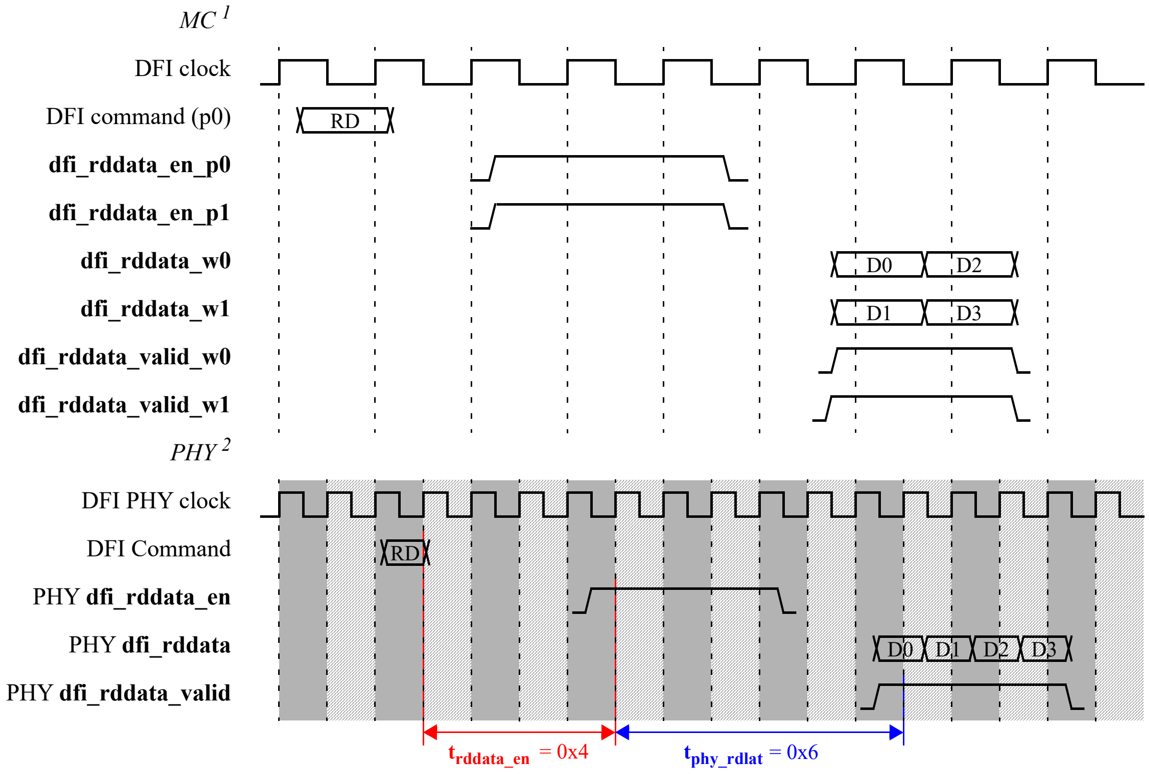
The rotational use of the **dfi_rddata_valid_wN** signals is only required in situations where the system may return less data than the DFI read data bus. If the minimum transfer size is a multiple of the DFI read data bus width, data can always be returned on all DFI data words and the **dfi_rddata_valid_wN** 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 55, “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 55. *1:2 Frequency Ratio Single Read Data Example with Even Read Data to Enable Timing*

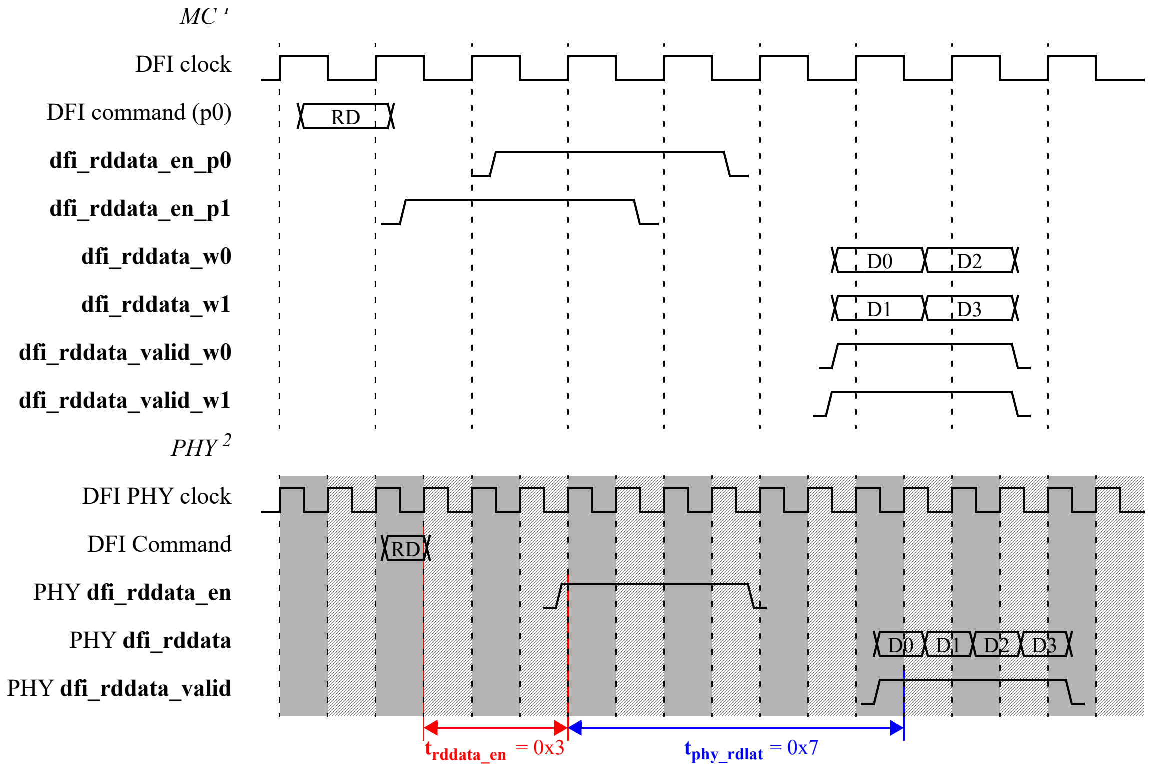


■ DFI PHY clock phase p0 ▨ DFI PHY clock phase p1

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 56, “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 56. *1:2 Frequency Ratio Single Read Data Example with Odd Read Data to Enable Timing*

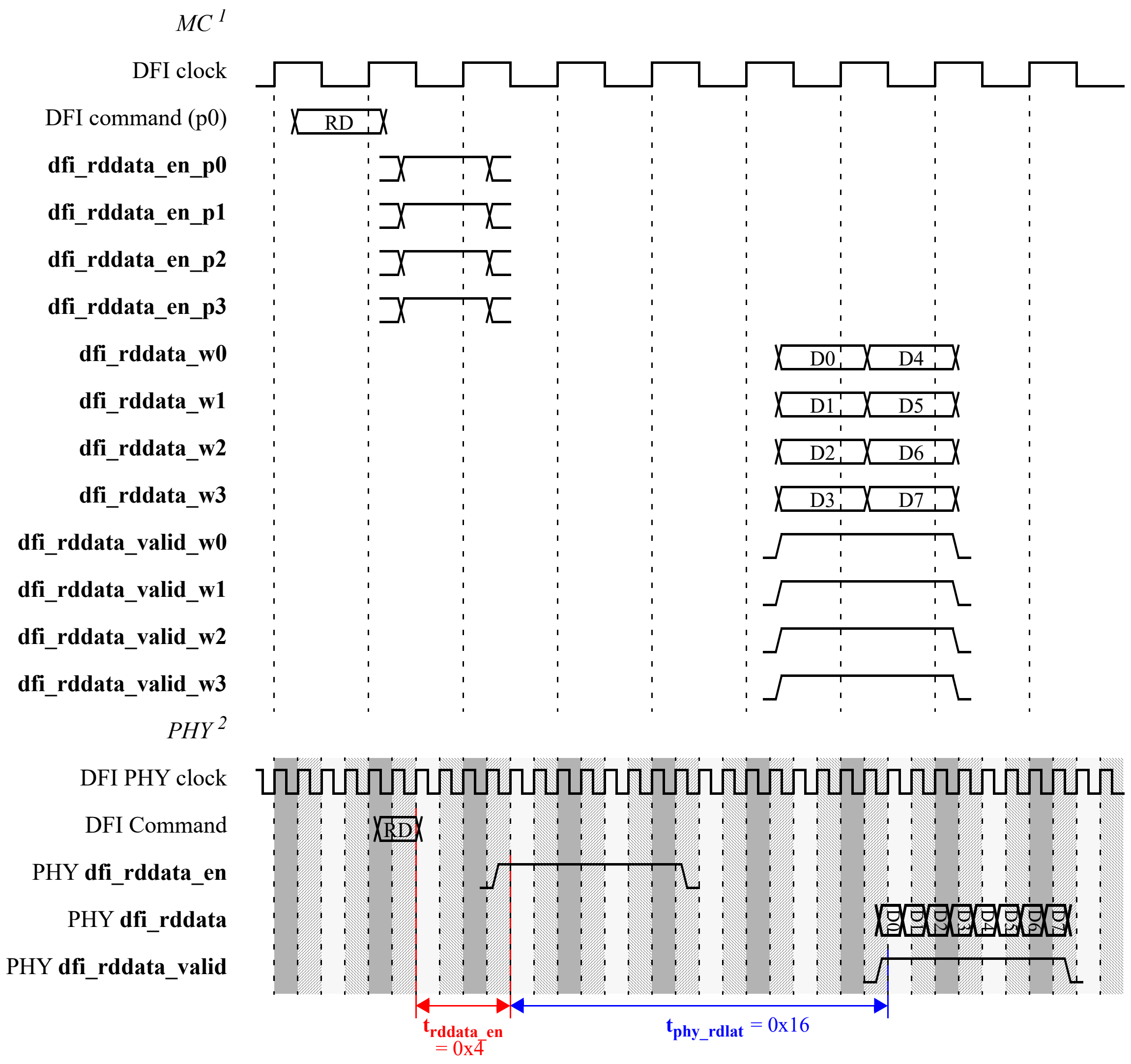


▨ DFI PHY clock phase p0	▨ DFI PHY clock phase p1

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 57, “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 57. *1:4 Frequency Ratio Single Read Data Example*

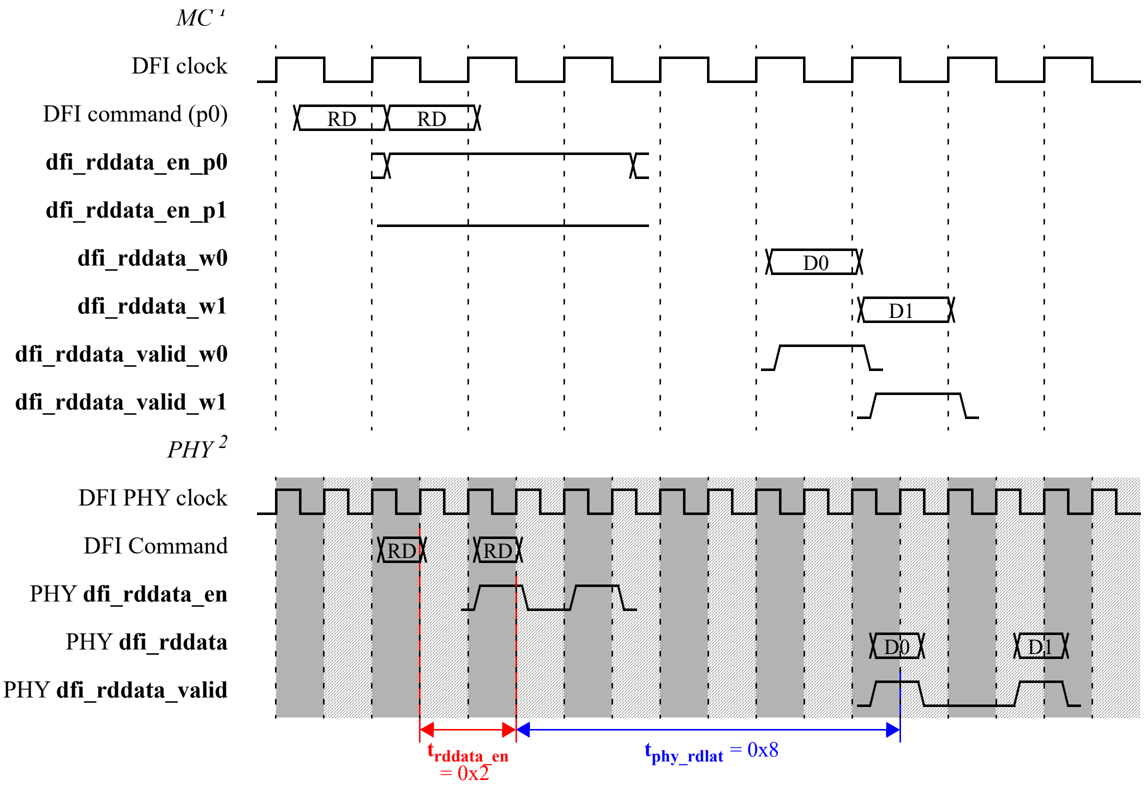


■ DFI PHY clock phase 0 ▨ DFI PHY clock phase 1 □ DFI PHY clock phase 2 ▧ DFI PHY clock phase 3

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 58, “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 58. *1:2 Frequency Ratio Multiple Read Data Example*

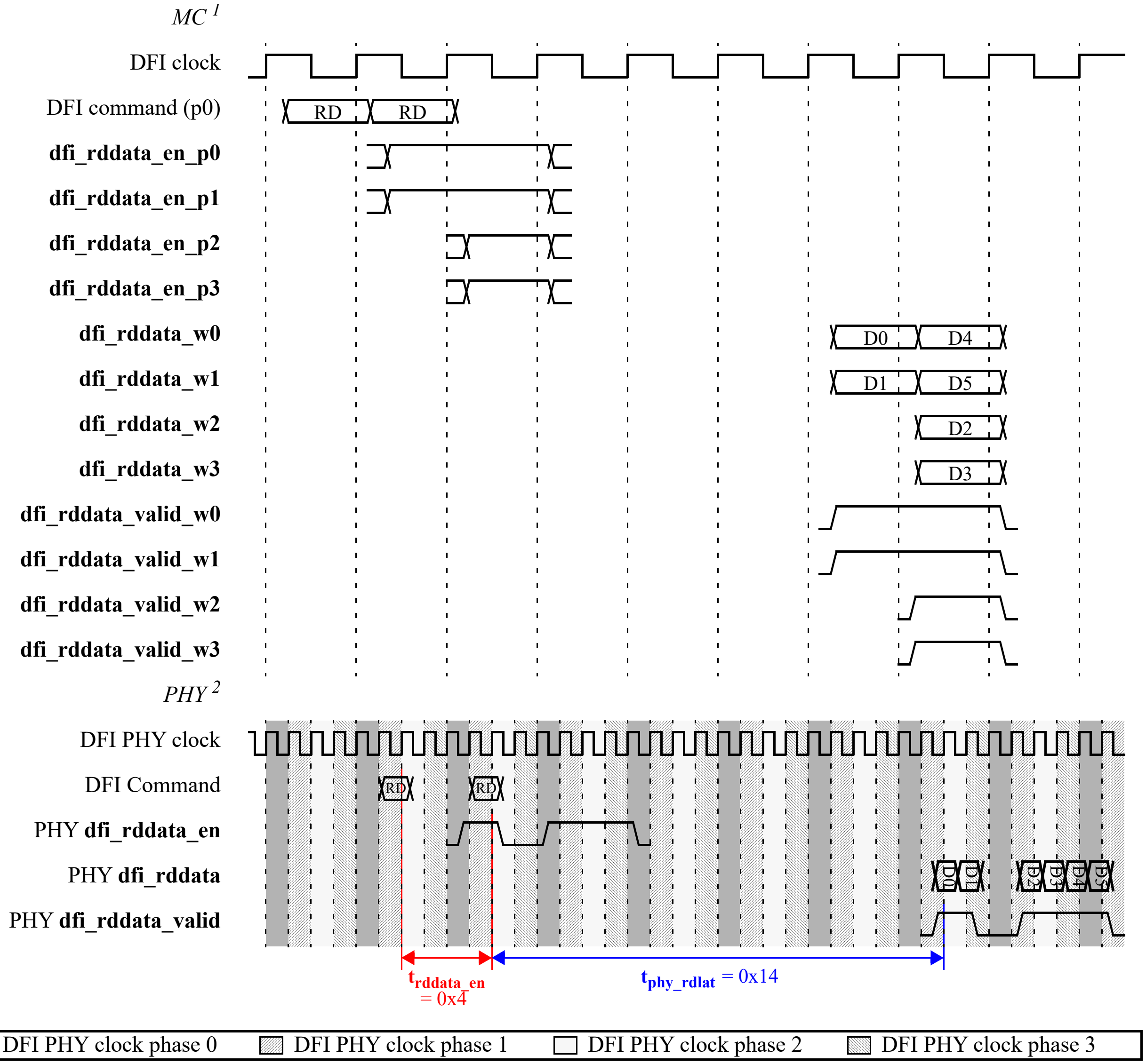


▩ DFI PHY clock phase p0 ▨ DFI PHY clock phase p1

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 58, “1:2 Frequency Ratio Multiple Read Data Example”, Figure 59, “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.

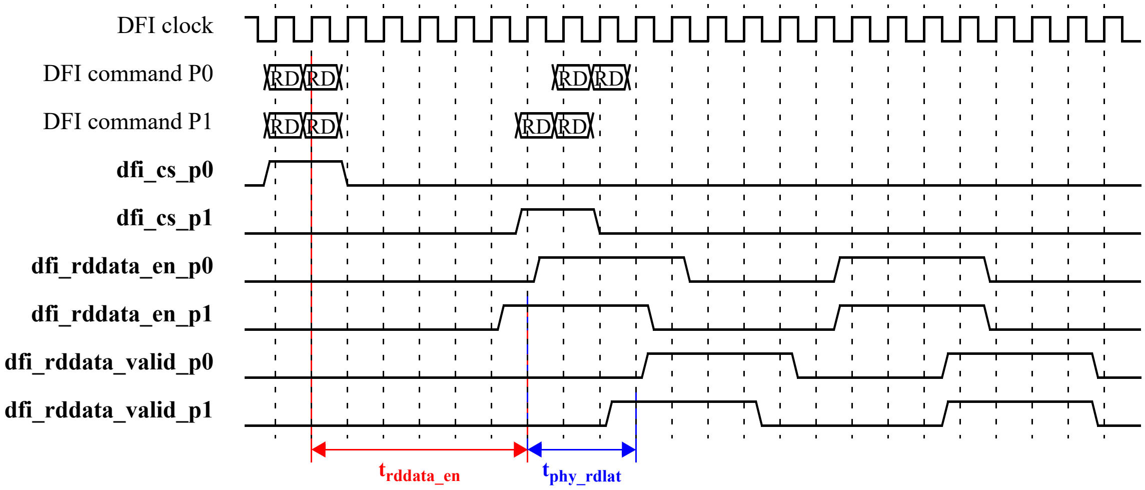
FIGURE 59. *1:4 Frequency Ratio Multiple Read Data Example*



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 60. *LPDDR4 Read Command, 1:2 Frequency Ratio*



4.10.4.1 DFI Read Data Rotation

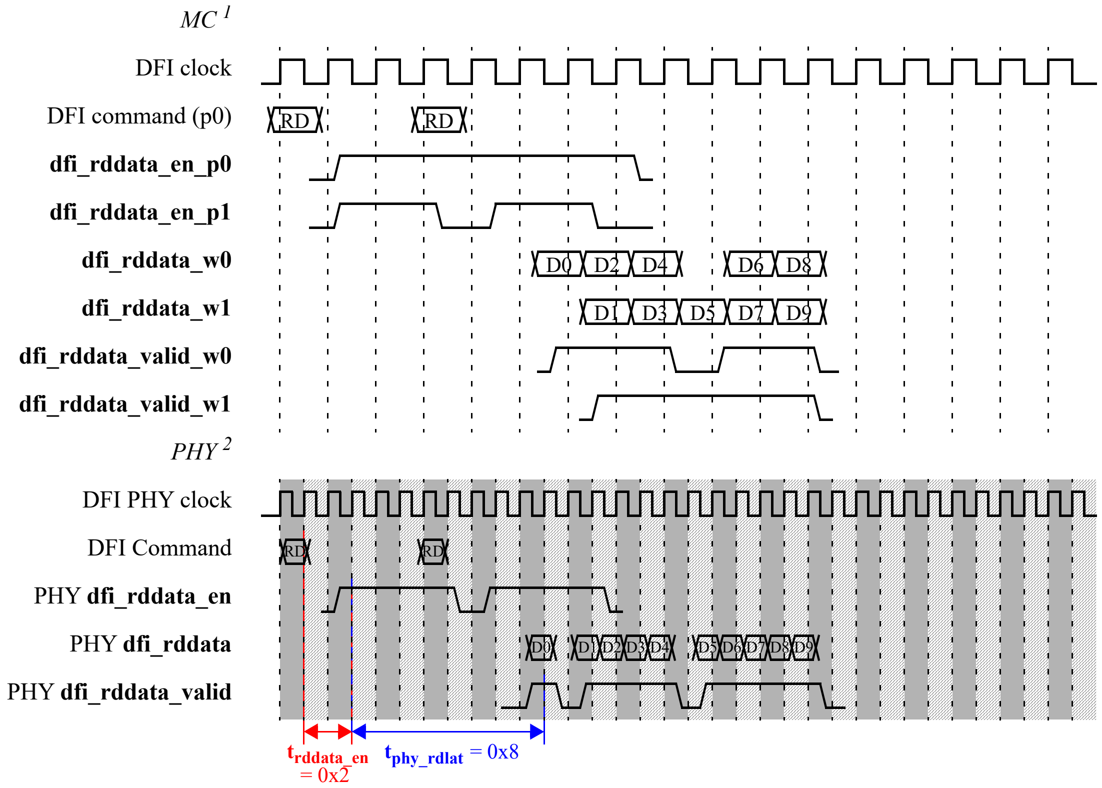
For simplicity, the following four diagrams illustrate different potential timing scenarios with a 1:2 frequency ratio system, transmitting 2 bursts of data, with each burst extended by one DRAM clock cycle.

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 61, “DFI Read Data Bus for Two 10UI Bursts, each starting in Phase 0”
2. Figure 62, “DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 0 (trddata_en = 2, tphy_rdlat = 8)”
3. Figure 63, “DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 1”
4. Figure 64, “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.

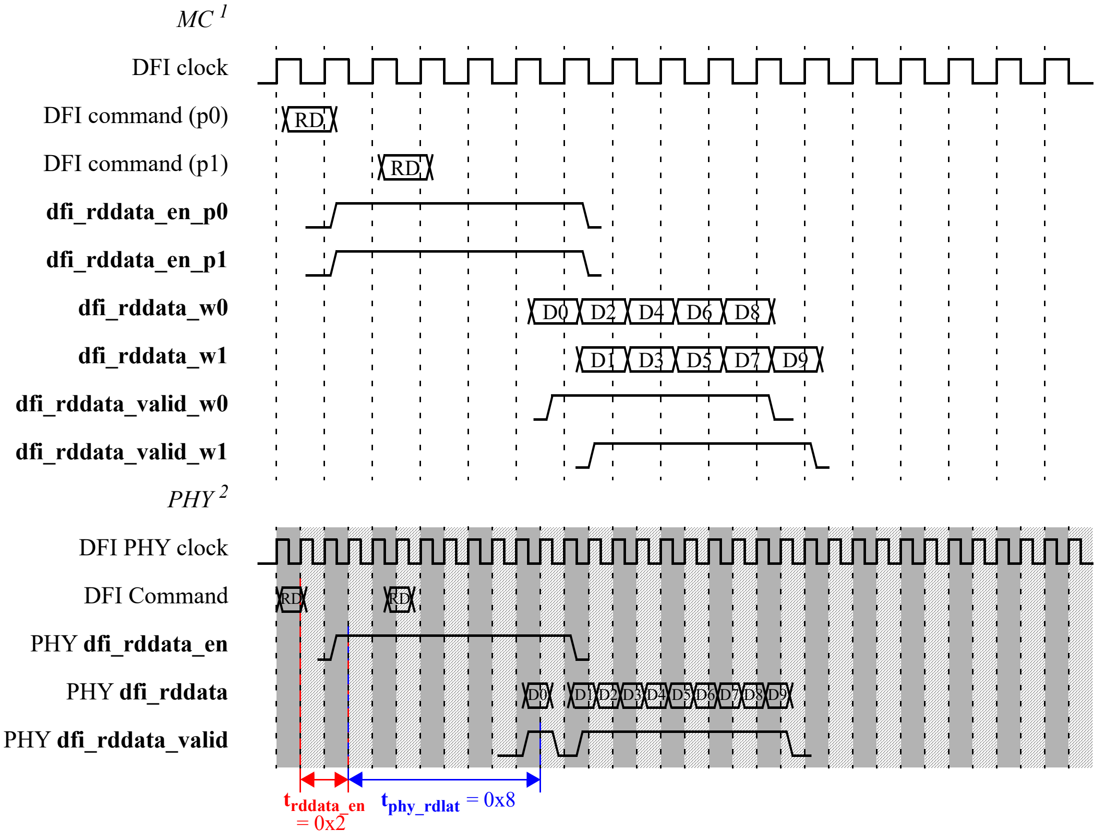
FIGURE 61. *DFI Read Data Bus for Two 10UI Bursts, each starting in Phase 0*



■ DFI PHY clock phase 0 ▨ DFI PHY clock phase 1

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.

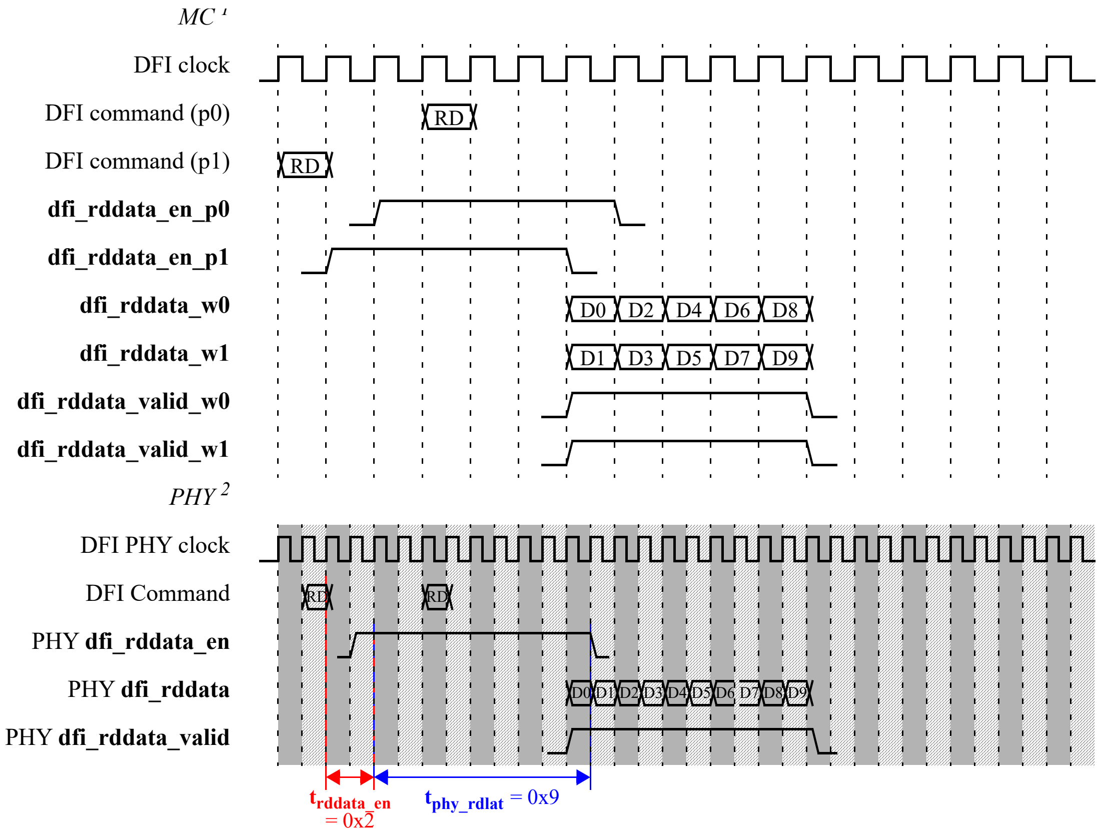
FIGURE 62. *DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 0 (**t_{rddata_en}** = 2, **t_{phy_rdlat}** = 8)*



■ DFI PHY clock phase 0 ▨ DFI PHY clock phase 1

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.

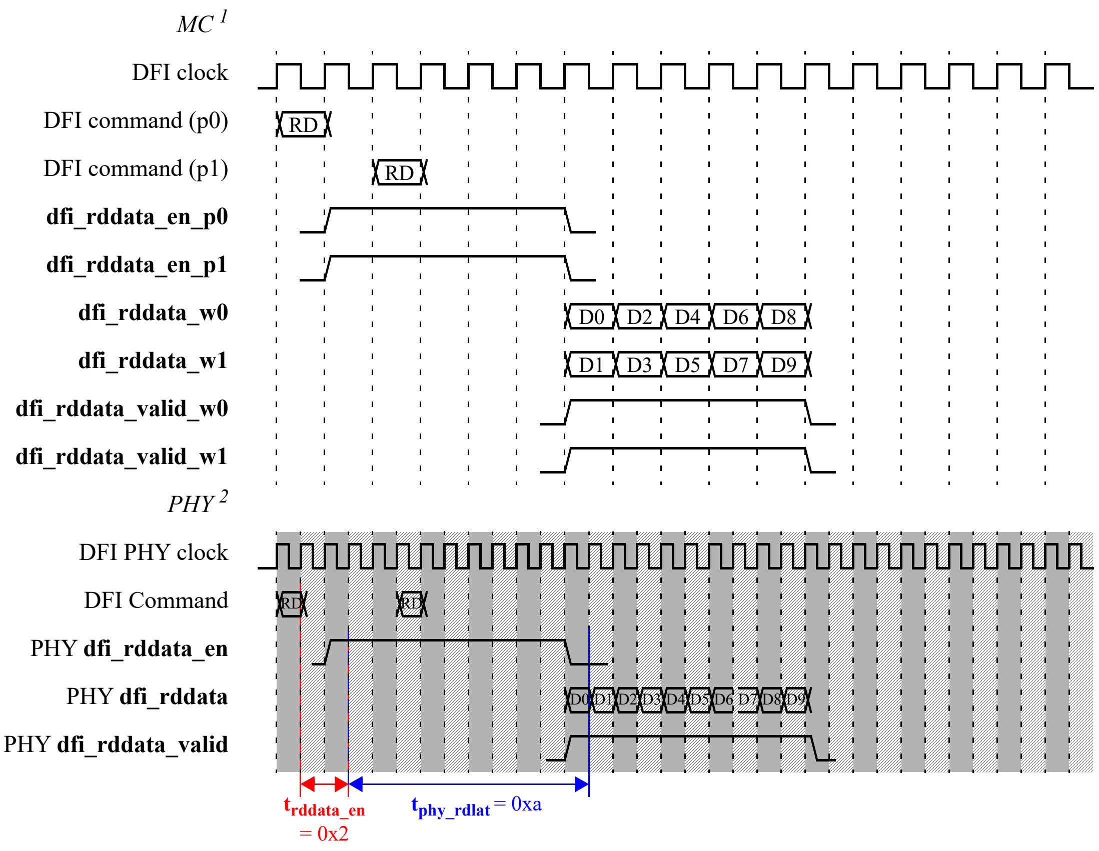
FIGURE 63. *DFI Read Data Bus for 10UI Back-to-Back Bursts Starting in Phase 1*



■ DFI PHY clock phase 0	▨ DFI PHY clock phase 1

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



■ DFI PHY clock phase 0	▨ DFI PHY clock phase 1

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.

4.10.4.2 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 (such as if an error must be reported), a mechanism is necessary to resynchronize the devices. 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.

4.11 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. 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.

The DFI frequency change protocol is an optional feature. The system may use a non-DFI frequency change method, or may choose to not support a frequency change option. However, if both the MC and the PHY intend to use the DFI 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 operation 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 operation occurs, some of the DFI timing parameters may need to be changed.

NOTE: During the frequency change protocol, 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**, **dfi_init_complete**, **dfi_frequency**, **dfi_freq_ratio** and **dfi_freq_fsp**. If supported, the **dfi_freq_ratio** signal indicates the clock ratio between the DFI clock and the memory clock or memory data clock depending on DDR technology. If the memory supports multiple FSPs, the **dfi_freq_fsp** signal indicates the target FSP for the frequency change. For more information on these signals, refer to Section 3.5, “Status Interface”.

4.11.1 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 request has been made and drives the encoded value of the new frequency on the **dfi_frequency** signal. The **dfi_freq_ratio** and **dfi_freq_fsp** are also driven if applicable. The memory must be given to the PHY as defined in the $\mathbf{dfi_{freq\_change\_state}}$ timing parameter during a frequency change operation before asserting the **dfi_init_start** signal. The PHY should not interpret the initial **dfi_init_start** assertion (during initialization) 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 a DFI frequency change operation 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 necessary, when the PHY acknowledges the frequency change request, the PHY will complete memory setup before de-asserting the **dfi_init_complete** signal. In this case, $\mathbf{t_{init\_start}}$ should be programmed to larger values, so that the PHY can complete memory setup with the new frequency settings.

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

The PHY must also perform the following actions before asserting the **dfi_init_complete** signal during a frequency change operation:

- Ensure that all the DRAM timings associated with re-initialization (PHY and memory setup) and re-training are met.
- Ensure that memory is in the state defined by the $\mathbf{dfi_{freq\_change\_state}}$.

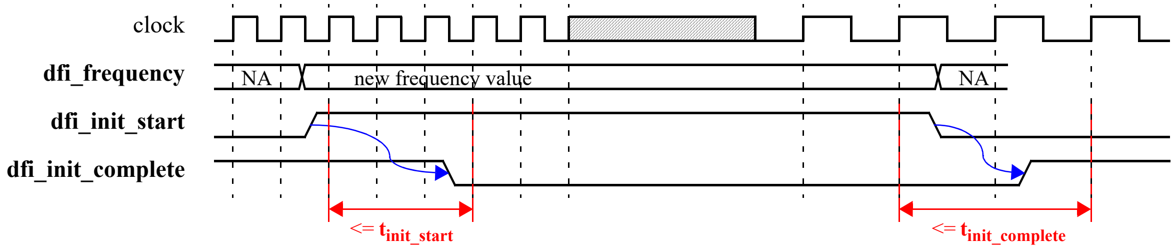
- Ensure that WCK is turned off for the DRAMs which support WCK.

During a frequency change operation, the PHY must ensure that the memory interface is maintained at valid and stable levels to ensure that memory protocol is being observed. The MC must also ensure that it maintains valid and stable levels on the DFI while **dfi_init_start** is asserted or **dfi_init_complete** is de-asserted. During a frequency change operation, the MC and the PHY must meet all the memory and DFI timing requirements prior to transitioning the corresponding **dfi_init_start** and **dfi_init_complete** signals. WCK needs to be disabled during a frequency change operation when the MC initiates with assertion of the **dfi_init_start** signal and while the PHY asserts the **dfi_init_complete** signal for the DRAMs which support WCK. For more information, refer to Section 3.11, “WCK Control Interface”.

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

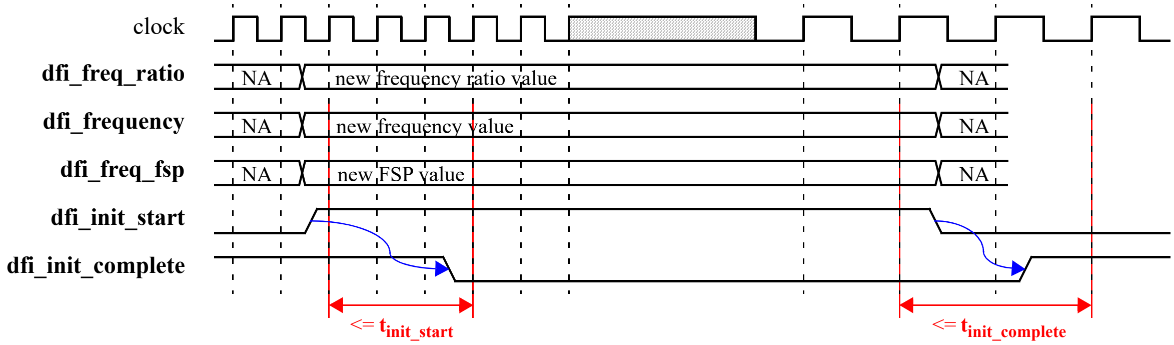
Figure 65, “Frequency Change Request Acknowledge Timing Diagram” shows the MC holding the **dfi_init_start** signal asserted as long as the frequency change operation continues and the PHY re-asserting **dfi_init_complete** within **$t_{init\_complete}$** cycles.

FIGURE 65. *Frequency Change Request Acknowledge Timing Diagram*



Note: “New frequency value” is the encoded value of the new frequency after the MC completes its MC-initiated frequency change operation.

FIGURE 66. *Frequency Change Request Acknowledge Timing Diagram, with Optional **dfi_freq_fsp** and **dfi_freq_ratio** Signals*

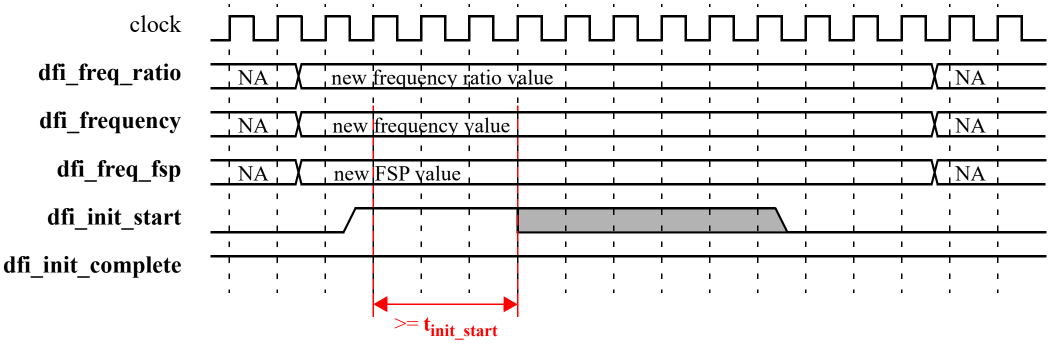


Note: “New frequency value” is the encoded value of the new frequency after the MC completes its MC-initiated frequency change operation, “new FSP value” is the value of the new frequency set point and “new frequency ratio value” is the encoded value of the new frequency ratio.

4.11.2 Frequency Change Request Protocol - Not Acknowledged

When asserted, 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 67, "Frequency Change Request Ignored Timing Diagram" shows the MC asserting the **dfi_init_start** signal for $\mathbf{t_{init\_start}}$ cycles.

FIGURE 67. *Frequency Change Request Ignored Timing Diagram*



4.12 CA Parity Signaling and CA Parity, CRC Errors

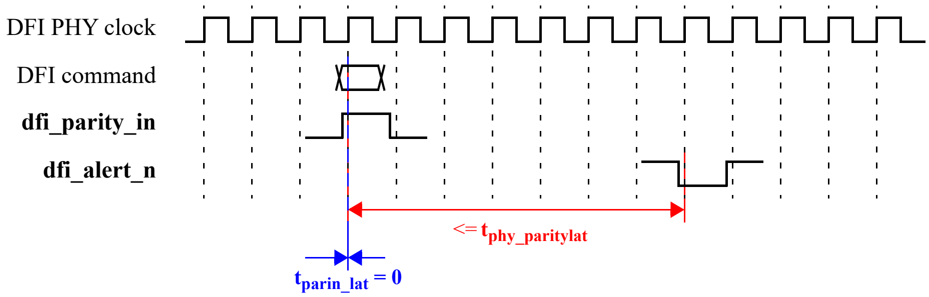
Parity bits are used in command transmission for verifying that the command has been transmitted correctly between master and slave. A single parity bit is sent with each command and identifies if the number of bits set high in the **dfi_address**, **dfi_act_n**, **dfi_bg**, **dfi_bank**, **dfi_cas_n**, **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 68, "Odd Command Parity Error Example Timing Diagram" shows the timing between the command and error with an odd command parity example.

FIGURE 68. *Odd Command Parity Error Example Timing Diagram*



4.12.1 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 **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 **t_{parin_lat}** parameter, defined in Table 19, “Status Interface Signals”, specifies the timing of the **dfi_parity_in_pN** signal relative to the DFI command. The **t_{parin_lat}** parameter, allows the parity data to be sent coincident with the command or on a subsequent cycle.

4.12.2 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.

4.12.3 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.

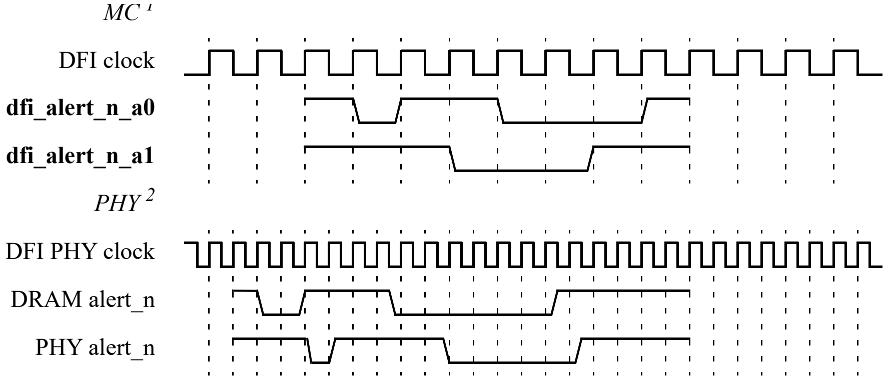
Table 43, “dfi_alert_n Signal With Matched and Frequency Ratio Systems” defines the outputs for matched frequency, 1:2 frequency and 1:4 frequency systems. In all cases, the PHY must synchronize the **dfi_alert_n** outputs to the DFI clock domain. During synchronization, the width of the DRAM error signal may change by a few cycles, and these small changes in the error pulse width are acceptable.

TABLE 43. ***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 69, “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 69. ***dfi_alert_n_aN*** *with 1:2 Frequency Ratio*



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.

4.13 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_ctrl_req**, **dfi_lp_ctrl_ack** and **dfi_lp_ctrl_wakeup** OR **dfi_lp_data_req**, **dfi_lp_data_ack** and **dfi_lp_data_wakeup** have reached a constant state.

When the MC detects an idle time, the MC asserts low power requests (control and/or data) to the PHY and wakeup signals (control and/or data) 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_ctrl_ack** or **dfi_lp_data_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 corresponding **dfi_lp_ctrl_wakeup** or **dfi_lp_data_wakeup** signal.

When **dfi_lp_ctrl_req** and **dfi_lp_ctrl_ack** have asserted, the MC will not de-assert **dfi_lp_ctrl_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_ctrl_wakeup** signal. Similarly, when the **dfi_lp_data_req** and **dfi_lp_data_ack** signals have asserted, the MC will not de-assert the **dfi_lp_data_req** signal to increase the wakeup time and the PHY will need to monitor the **dfi_lp_data_wakeup** signal.

Wakeup time is a specific number of cycles ($\mathbf{t_{lp\_ctrl\_wakeup}}$ or $\mathbf{t_{lp\_data\_wakeup}}$ 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. The DFI specification defines up to 20 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_ctrl_wakeup** or **dfi_lp_data_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_ctrl_ack** or **dfi_lp_data_ack** signal within $\mathbf{t_{lp\_resp}}$ clock cycles of the request (control and/or data) signal assertion, during which time the MC must hold the **dfi_lp_ctrl_wakeup** or **dfi_lp_data_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_ctrl_ack** or **dfi_lp_data_ack** signal within the appropriate $\mathbf{t_{lp\_ctrl\_wakeup}}$ or $\mathbf{t_{lp\_data\_wakeup}}$ clock cycles after the **dfi_lp_ctrl_req** or **dfi_lp_data_req** signal is de-asserted and be ready for normal operation.

Figure 70, “Low Power Control Handshaking Timing Diagram” shows a sequence in which the request is acknowledged. The same diagram applies to the data low power interface.

FIGURE 70. *Low Power Control Handshaking Timing Diagram*

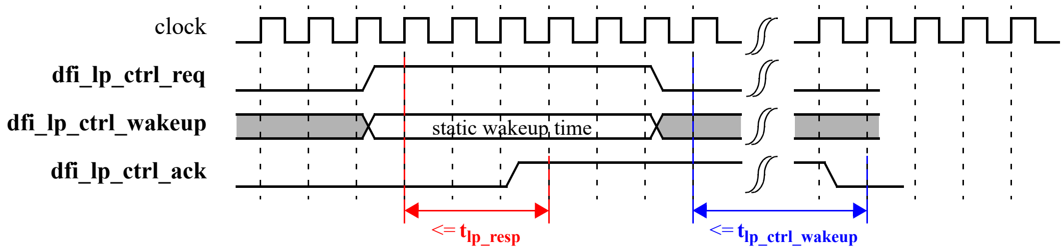
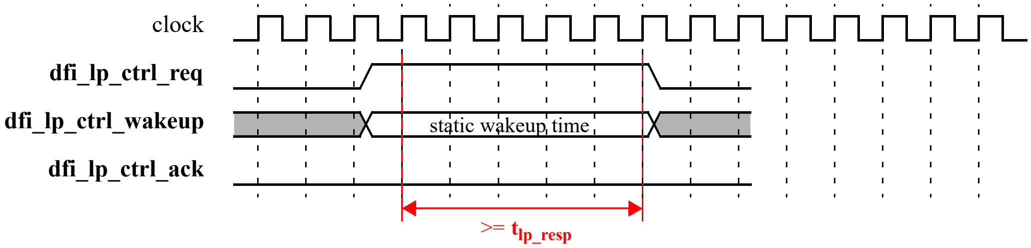


Figure 71, “Low Power Control Request with No Response” shows that the **dfi_lp_ctrl_ack** signal is not required to assert when the **dfi_lp_ctrl_req** signal is asserted. The MC must assert the **dfi_lp_ctrl_req** signal for at least $\mathbf{t_{lp\_resp}}$ cycles. If the **dfi_lp_ctrl_ack** signal 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** signal should be de-asserted after $\mathbf{t_{lp\_resp}}$ cycles have elapsed without an acknowledge. The same timing diagram would apply for the data low power interface.

FIGURE 71. *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_ctrl_wakeup** (or **dfi_lp_data_wakeup**) signal to a larger value as long as both the **dfi_lp_ctrl_ack** and **dfi_lp_ctrl_req** (or **dfi_lp_data_ack/dfi_lp_data_req)** signals are asserted. This results in a longer $\mathbf{t_{lp\_ctrl\_wakeup}}$ (or $\mathbf{t_{lp\_data\_wakeup}}$) time for the PHY. The value of the

dfi_lp_ctrl_wakeup / dfi_lp_data_wakeup signal when the **dfi_lp_ctrl_req / dfi_lp_data_req** signal is de-asserted will be used to define the $\mathbf{t_{lp\_ctrl\_wakeup}}$ / $\mathbf{t_{lp\_data\_wakeup}}$ time.

Figure 72, “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 72. *Low Power Control Handshaking Timing Diagram with Multiple Wakeup Times*

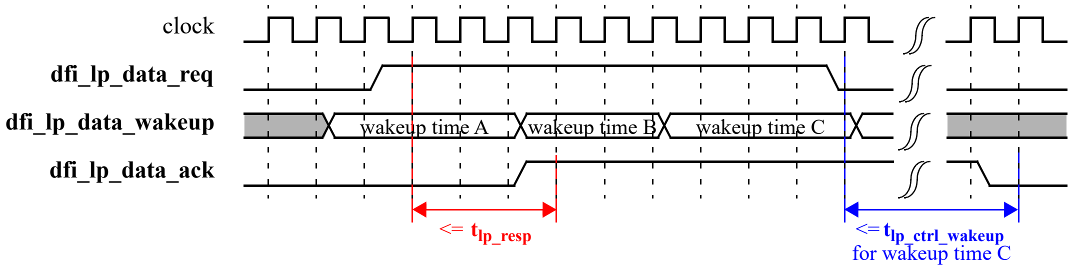
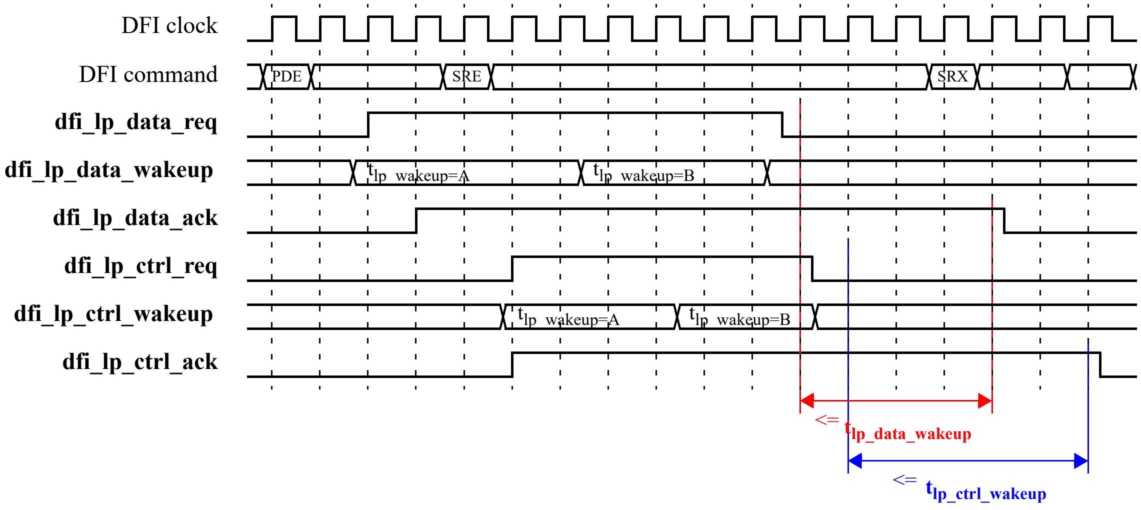


Figure 73, “Low Power State Progressing From an Active Command Interface to Inactive” illustrates **dfi_lp_ctrl_req** and **dfi_lp_data_req** when progressing from a low power state that requires use of the command interface into a low power state that can operate when the command interface is inactive.

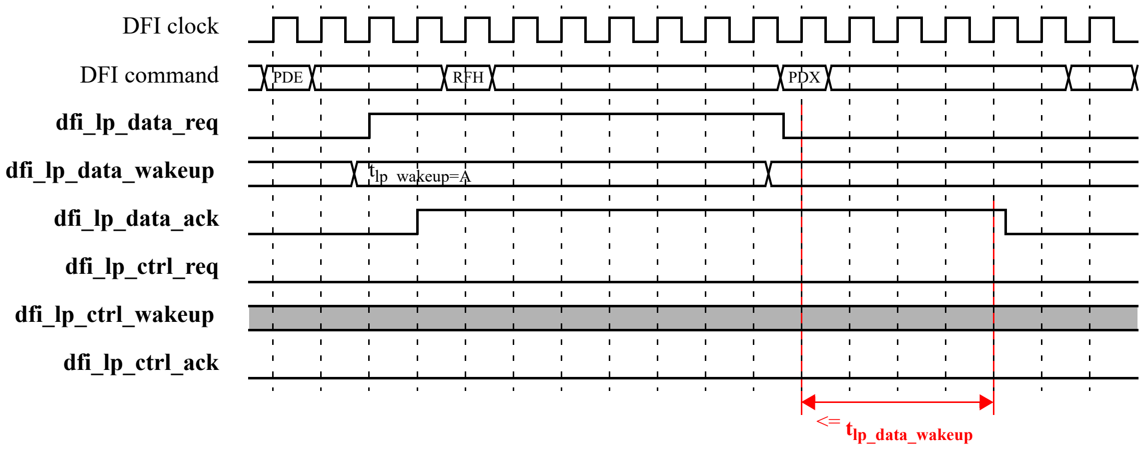
FIGURE 73. *Low Power State Progressing From an Active Command Interface to Inactive*



PDE = Power-Down Enter SRE = Self-Refresh Enter SRX = Self-Refresh Exit

Figure 74, “Low Power State Requiring Ongoing Use of the Command Interface” illustrates **dfi_lp_data_req** when entering a low power state that must continue to execute memory commands.

FIGURE 74. *Low Power State Requiring Ongoing Use of the Command Interface*



PDE = Power-Down Enter RFH = Refresh Enter PDX = Power-Down Exit

Figure 75, “Data Low Power Request Acknowledged Only”, Figure 76, “Both Data and Control Low Power Requests Acknowledged Simultaneously” and Figure 77, “Both Data and Control Low Power Requests Acknowledged, At Different Times” show four illustrations of both the control and data low power requests asserting. The requests are independent and may assert in either order or simultaneously, and the PHY may acknowledge one, both or neither - and at any point in time. There is no requirement that the PHY has to respond in either order.

FIGURE 75. *Data Low Power Request Acknowledged Only*

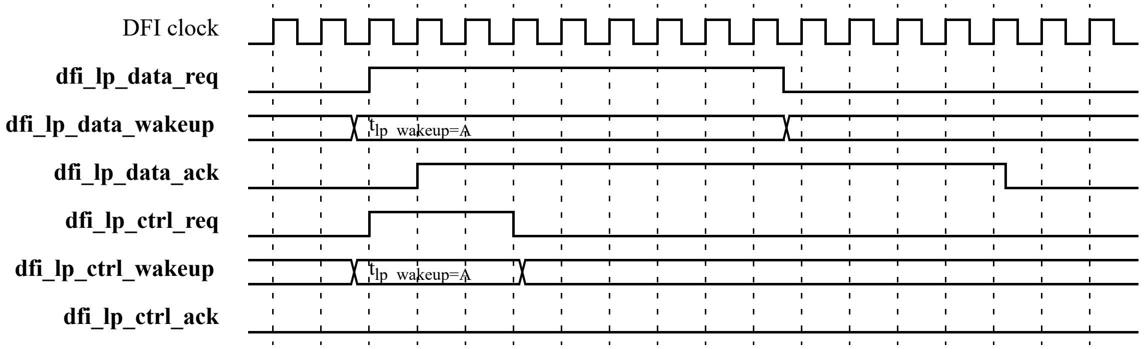


FIGURE 76. *Both Data and Control Low Power Requests Acknowledged Simultaneously*

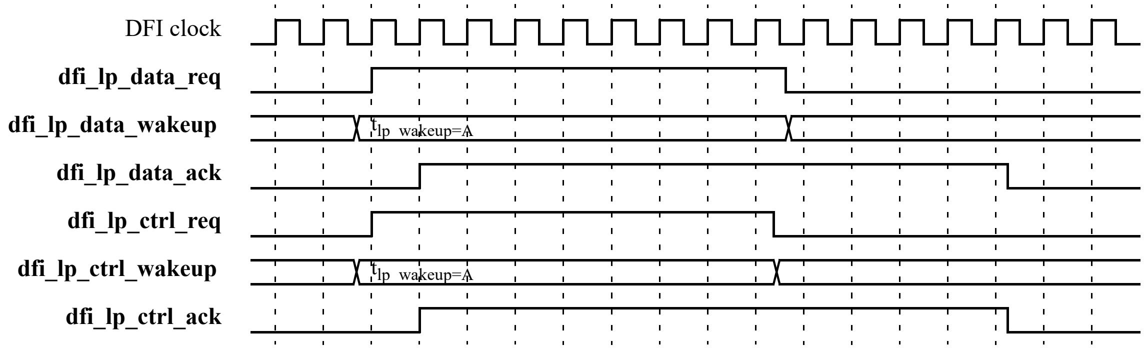
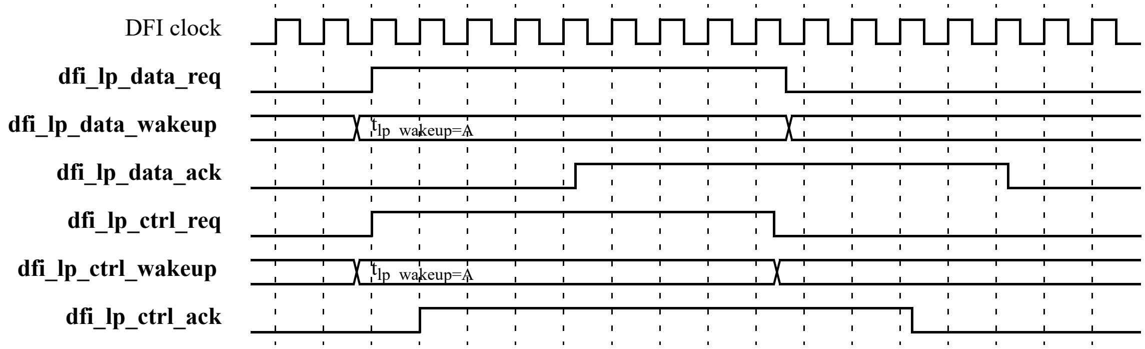


FIGURE 77. *Both Data and Control Low Power Requests Acknowledged, At Different Times*



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 78, “Example of Error Condition” shows the error interface. The error condition does not affect RD A and RD B, and may affect RD C and/or RD D.

FIGURE 78. *Example of Error Condition*

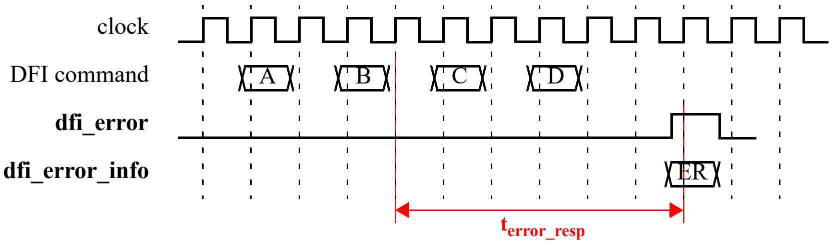


Table 44, “Error Codes” defines specific error codes, error codes reserved for future use, and error codes available for design-specific definition. Not all defined error codes apply to all systems and all error types are optional unless required as part of another interface.

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 44. *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 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:

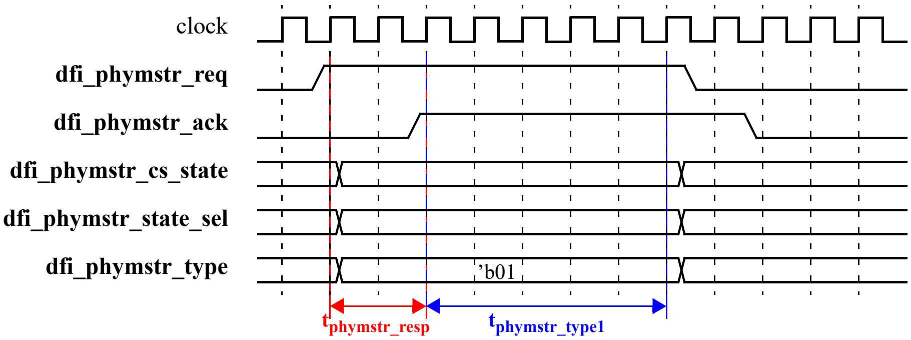
1. The PHY asserts the **dfi_phymstr_req** signal along with the **dfi_phymstr_cs_state** and **dfi_phymstr_state_sel** signals to indicate to the MC the state of the memory required before handing off the bus. The PHY drives **dfi_phymstr_type** to indicate to the MC the length of time that it is requesting 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 of the assertion of **dfi_phymstr_req** in most cases. Exceptions are granted if **dfi_phyupd_ack**, **dfi_lp_ctrl_req** or **dfi_lp_data_req** are active along with **dfi_phyupd_req**. Refer to Section 4.21, “DFI Interactions” for details.
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 $\mathbf{t}_{\mathbf{phymstr\_typeX}}$ 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 79, "Master Interface Timing" illustrates the new master interface timing.

FIGURE 79. *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.

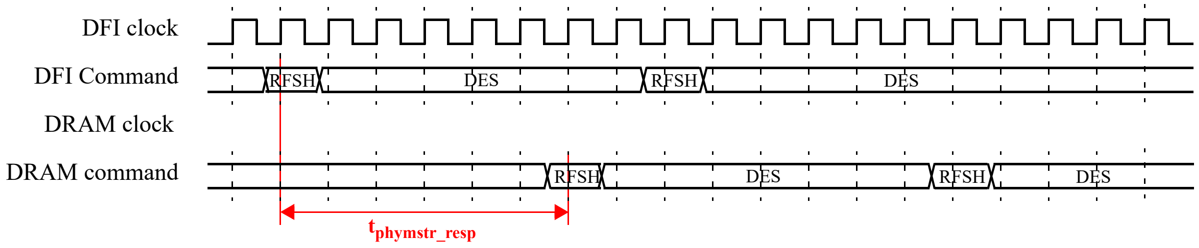
To meet DRAM refresh timing requirements the MC and the PHY perform the following steps:

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 ($\mathbf{t}_{\mathbf{phymstr\_rfsh}}$ parameter) for meeting all DRAM refresh timing.
2. The PHY performs following actions:

a. The PHY generates a refresh command to the DRAM within $\mathbf{t}_{\mathbf{phymstr\_rfsh}}$ 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 80, “Master Refresh Timing” illustrates the new master refresh timing.

FIGURE 80. *Master Refresh Timing*



4.16 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.

4.16.1 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 ($\mathbf{t}_{\mathbf{phyupd\_typeX}}$, $\mathbf{t}_{\mathbf{ctrlupd\_min}}$), 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 $\mathbf{t}_{\mathbf{ctrlupd\_interval}}$ 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 $\mathbf{t}_{\mathbf{ctrlupd\_disconnect}}$ or $\mathbf{t}_{\mathbf{phyupd\_disconnect}}$ 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 $\mathbf{t}_{\mathbf{ctrlupd\_disconnect\_error}}$ or $\mathbf{t}_{\mathbf{phyupd\_disconnect\_error}}$ 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 81, “PHY Update Request, QOS Disconnect Protocol” and Figure 82, “PHY Update Request, Error Disconnect Protocol” illustrate the new disconnect signaling protocol.

FIGURE 81. *PHY Update Request, QOS Disconnect Protocol*

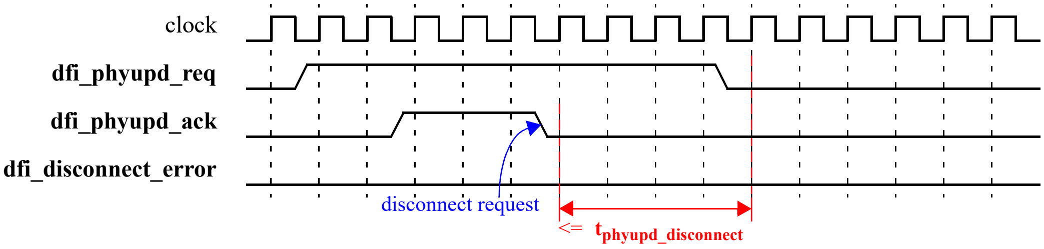
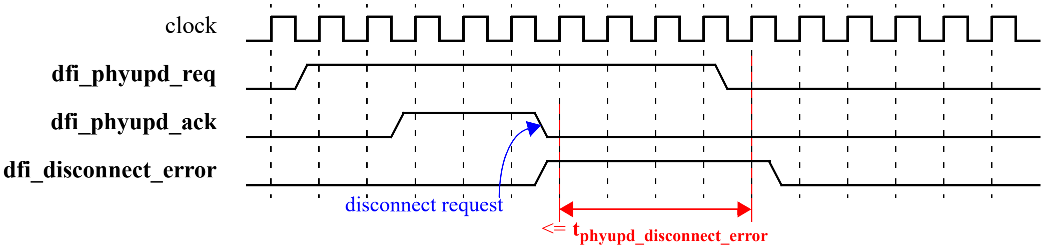


FIGURE 82. *PHY Update Request, Error Disconnect Protocol*



4.16.2 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.

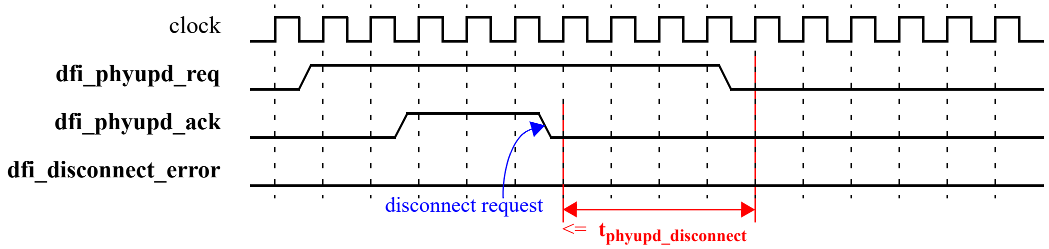
The **dfi_disconnect_error** signal defines if the disconnect is an error condition or a QOS event. The value of this signal when the disconnect occurs (the de-assertion of the **dfi_phymstr_ack** signal) determines the expected system behavior:

- 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 83, “PHY Master Interface, QOS Disconnect Protocol” illustrates the new disconnect signaling protocol.

FIGURE 83. *PHY Master Interface, QOS Disconnect Protocol*



4.17 Use of the 2N Mode

Some memories support a 2N mode (also referred to as 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 specification includes a **dfi_2n_mode** signal to support this behavior.

To enter 2N mode on DFI, the MC enters the DRAM 2N mode. If synchronization is required, when the $\mathbf{t_{ctrl\_delay}}$ time elapses from the defined sync pulse, the MC will assert the **dfi_2n_mode** signal. If synchronization is not required, the MC will assert **dfi_2n_mode** following the command to place the DRAM in 2N mode. The CA bus signals must be maintained in an idle or de-select state or both for the time defined in the $\mathbf{t_{2n\_mode\_delay}}$ parameter. During this time, the PHY makes adjustments in CA and/or CS timing. To exit 2N mode, the same timing applies, no synchronization is required.

For DDR4, when **dfi_2n_mode** = ’b1, the MC must drive command CA and CS signals for two DFI PHY clocks - for two DFI clocks in a matched frequency configuration, or for two phases in a Frequency Ratio configuration.

For DDR5, when **dfi_2n_mode** = ’b1, the MC must drive CA signals for two DFI PHY clocks - for two DFI clocks in a matched frequency configuration, or for two phases in a Frequency Ratio Configuration.

Several timing diagrams follow. These timing diagrams include both the MC timing and an illustration of how the PHY interprets the timing in the Memory Clock domain. In these figures, the signals are shown changing on the positive clock edge. In the MC clock portion, the timing shows the DFI bus signaling. In the Memory portion, the timing illustrates how the PHY interprets the DFI bus. Memory timing is shown for illustrative purposes only.

Figure 84, “Illustration of DDR4 read in 1N Mode” and Figure 85, “Illustration of DDR4 read in 2N Mode” illustrate the differences between 1N and 2N mode commands for DDR4 on DFI.

FIGURE 84. *Illustration of DDR4 read in 1N Mode*

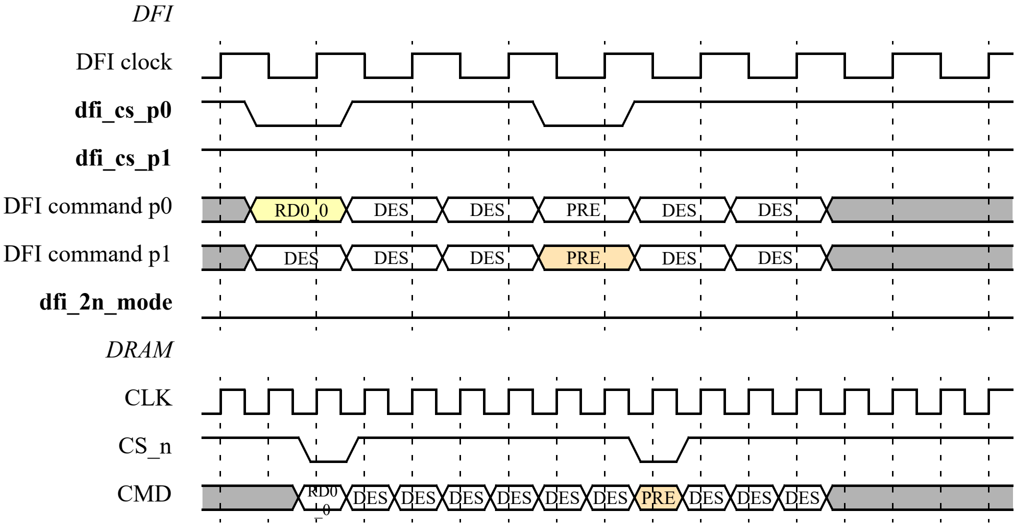


FIGURE 85. *Illustration of DDR4 read in 2N Mode*

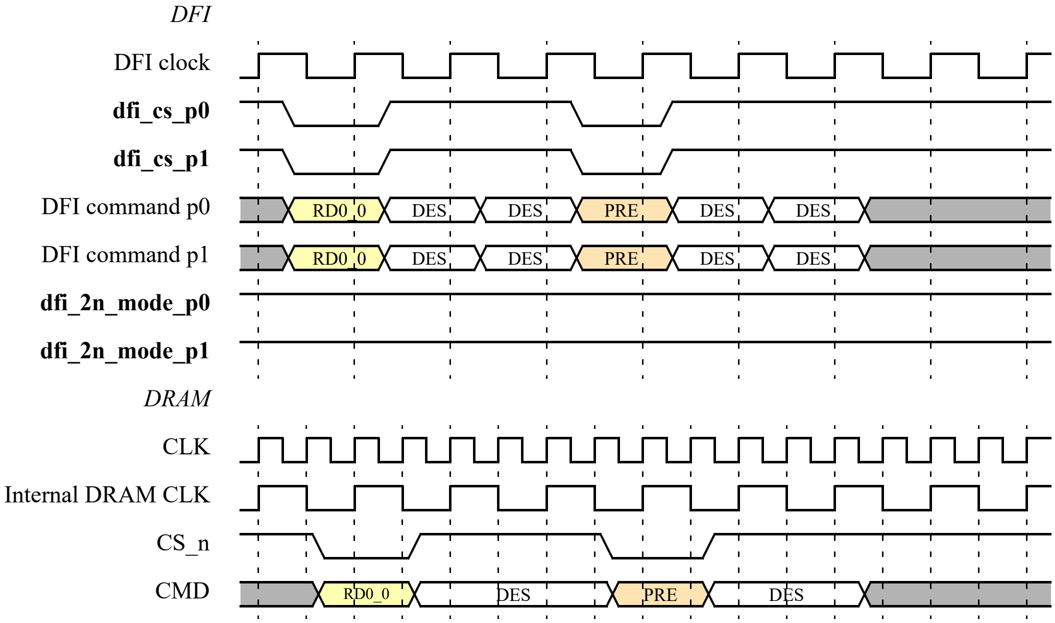


Figure 86, “Illustration of DDR5 read in 1N Mode” and Figure 87, “Illustration of DDR5 read in 2N Mode” illustrate the differences between the modes for DDR5 on DFI.

FIGURE 86. *Illustration of DDR5 read in 1N Mode*

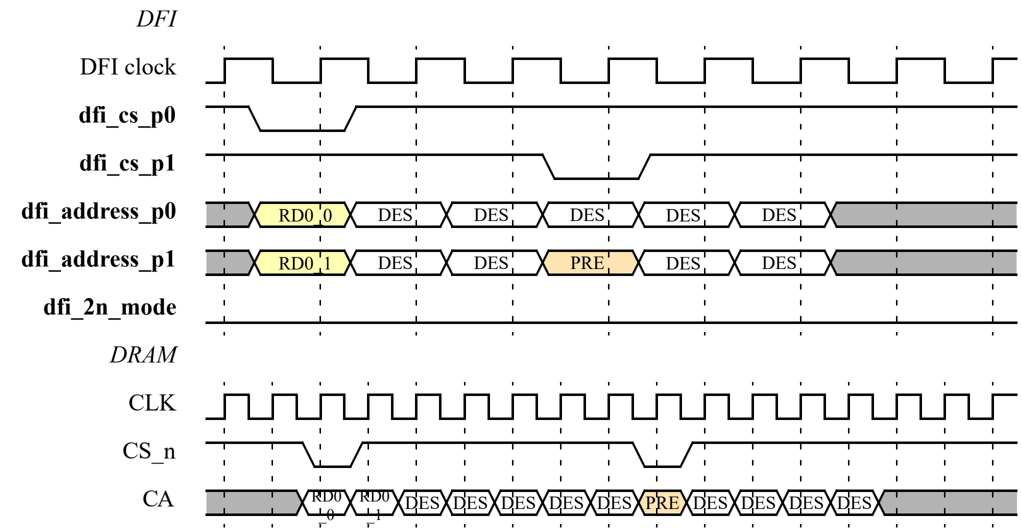


FIGURE 87. *Illustration of DDR5 read in 2N Mode*

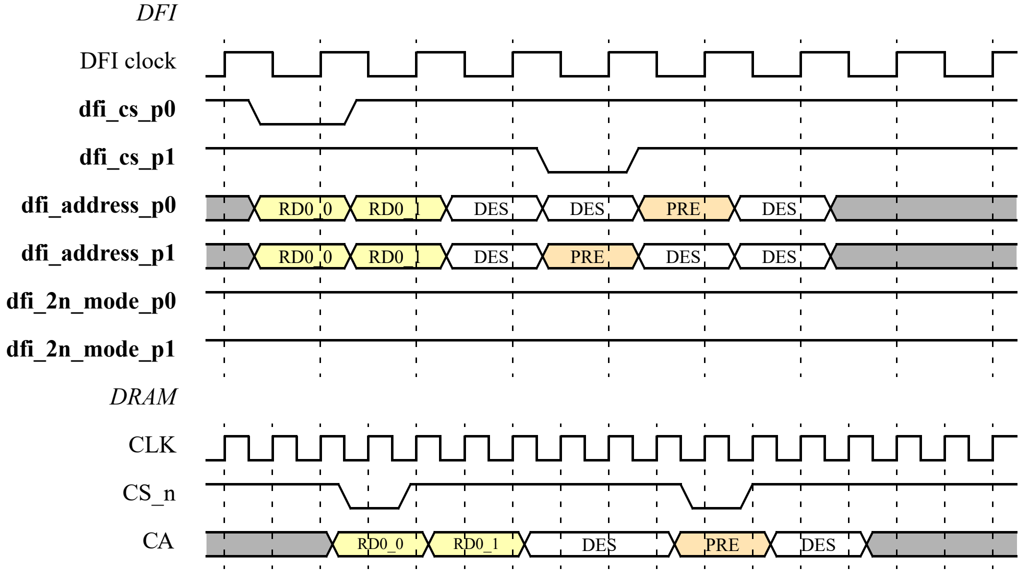


Figure 88, “Entering 2N Mode with a Sync Command; DDR4 with Matched Frequency” and Figure 89, “Entering 2N Mode with a MPC command; DDR5, 1:2 Frequency Ratio” illustrate the difference in entering 2N mode with and without issuing a sync command.

FIGURE 88. *Entering 2N Mode with a Sync Command; DDR4 with Matched Frequency*

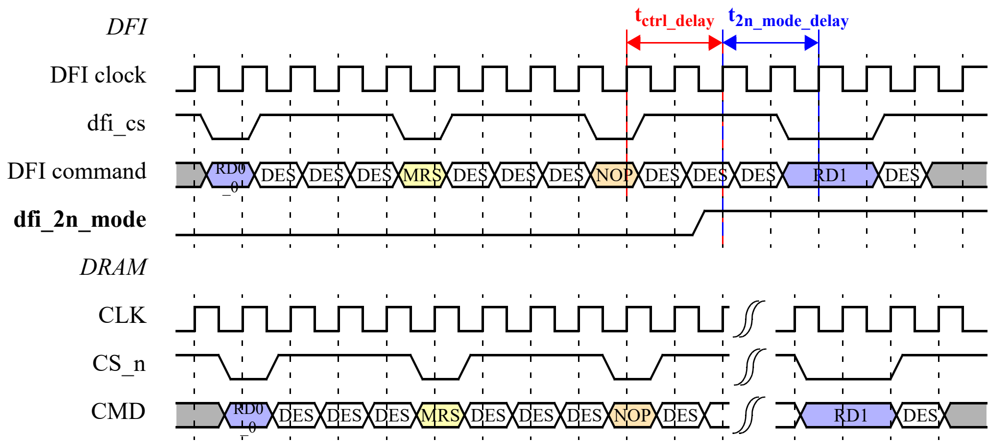
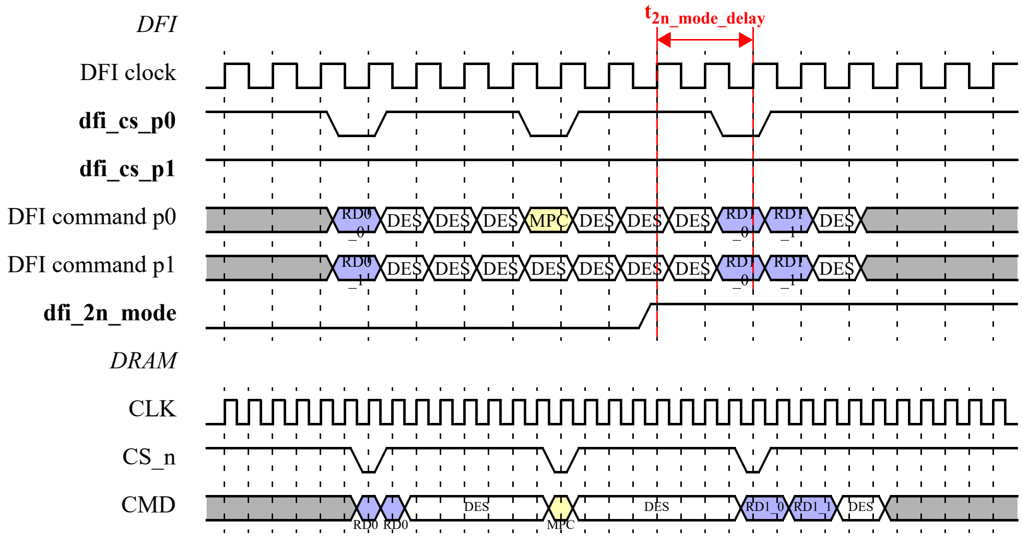


FIGURE 89. *Entering 2N Mode with a MPC command; DDR5, 1:2 Frequency Ratio*



4.18 Use of the MC to PHY Message Interface

The MC to PHY message interface handles the transmission of encoded messages from the MC to the PHY. Messages may be pre-defined or device-specific messages, Both the controller and PHY should support encodings for all messages that are supported by the device.

When the **dfi_ctrlmsg_req** signal is asserted, the PHY can ignore the request, or if it chooses to respond, it must assert the **dfi_ctrlmsg_ack** within $\mathbf{t_{ctrlmsg\_resp}}$ DFI clocks. Once asserted, the **dfi_ctrlmsg_ack** may de-assert at any time. If the PHY wants to prevent additional messages, it may continue to drive the **dfi_ctrlmsg_ack** signal for up to $\mathbf{t_{ctrlmsg\_max}}$ DFI clocks. As long as **dfi_ctrlmsg_ack** is asserted, the **dfi_ctrlmsg_req** signal must remain asserted and should de-assert immediately if the **dfi_ctrlmsg_ack** signal is de-asserted; sufficient delay is allowed to account for pipelining and clock domains and no specific timing requirement is defined.

If the PHY does not acknowledge the request within $\mathbf{t_{ctrlmsg\_resp}}$ cycles, the PHY must not acknowledge the request at all. In this case, the controller should de-assert the **dfi_ctrlmsg_req** signal. The **dfi_ctrlmsg_req** signal must remain de-asserted for at least one DFI clock before issuing another request.

While the **dfi_ctrlmsg_req** signal is asserted, the **dfi_ctrlmsg** and **dfi_ctrlmsg_data** signals must remain static. When the **dfi_ctrlmsg_req** signal is de-asserted, the **dfi_ctrlmsg** and **dfi_ctrlmsg_data** signals are in a "don't_care" state. If the PHY asserts the **dfi_ctrlmsg_ack** signal, the PHY has received the message. If the PHY does not assert the **dfi_ctrlmsg_ack** signal, the PHY may or may not have received the message.

There is not a defined response from the MC or PHY regarding messages sent and no implied timing for a response or any action in response to the message. The message bus is informational and the corresponding activity is implementation-specific.

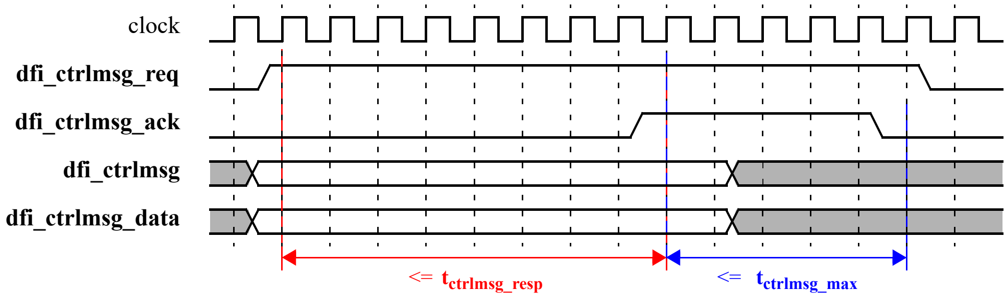
The assertion of the **dfi_ctrlmsg_req** and/or **dfi_ctrlmsg_ack** signals cannot block the operation of the other interfaces on the DFI bus; messages should be sent in parallel to other DFI operations. The specific behavior of the MC to PHY message interface relative to other interfaces is detailed as follows:

- PHY Master and Update Interface: DFI messages are permitted during update or PHY master interface activity.
- Low Power Interface: The MC may send messages in a low power state if the wakeup time is below the $\mathbf{dfi_{lp\_wakeup\_threshold}}$ value. If the PHY is in the defined low power state corresponding to the wakeup time or a larger wakeup time, the MC should not issue a message request.
- Frequency Change Protocol: When a frequency change is acknowledged, any pending messages can be canceled. The PHY should not acknowledge a message once a frequency change has been acknowledged until the frequency change handshake has completed.

4.18.1 Timing Diagram

Figure 90, "MC to PHY Message Handshaking Timing Diagram" shows the timing relationship between the MC to PHY message interface signals.

FIGURE 90. *MC to PHY Message Handshaking Timing Diagram*



4.18.2 PHY Messages

For compatibility between a DFI controller and a DFI PHY, a standard set of messages is defined. Additional message encodings will be defined by the device, or those settings will be reserved. Table 45, "Controller Message Codes" defines the command encodings.

TABLE 45. *Controller Message Codes*

Message Code	Name	Description
0x00	Command Bus Training	Training the CA bus relative to the memory clock.
0x01	Read Data Eye Training	Training the DQ bus to the DQS
0x02	Read Gate Training	Training the read DQS to the DQS gate
0x03	CS Training	Training the chip select timing
0x04	Write Leveling	Training the write DQ to the DQS
0x05	Write Data Eye Training	Training the write DQ to the DQS
0x06	DB Bus Training	Training the Data Buffer
0x07	ZQ Calibration	Output driver and termination calibration
0x08	DQS Interval Oscillator	Evaluation of the DQS timing
0x09	WCK2CK Internal Oscillator Input	Evaluation of WCK input timing
0x0a	WCK2CK Interval Oscillator Output	Evaluation of WCK output timing
0x0b - 0x1f	Reserved	Future expansion
0x20-0x3f	Device-specific	Defined by the MC and the PHY. These should be programmatically mapped
0x40 - 0xff	Reserved	Future expansion

4.18.3 Information Data Signal

Additional information may be required for each message. The **dfi_ctrlmsg_data** signal provides additional information, if required, for the message. The data field may be uniquely defined per message. The message data signal width is 16 bits. The lower bits define whether the message is to a single rank or all ranks. Other encodings, such as 2 ranks, are not supported. The physical and logic ranks are encoded; these encodings are ignored if the corresponding all rank bit is set.

For device-specific message encodings, the data coding can be customized to the message. Additional fields are available for device-specific usage and reserved for future expansion. Table 46, "Data Signal Codes" defines the data signal for each command.

TABLE 46. *Data Signal Codes*

Field	Data Coding	Description
Bit [0]	0, 1	• ’b0 = Signal physical rank as defined by bits [7:4] • ’b1 = All physical ranks In a single-rank system, this bit should be set to ’b1.
Bit [1]	0, 1	• ’b0 = Signal logical rank as defined by bits [11:8] • ’b1 = All logical ranks If a 3DS device is not being used, this bit should be set to ’b1.

TABLE 46. *Data Signal Codes*

Field	Data Coding	Description
Bits [3:2]	Reserved	
Bits [7:4]	0x0 - 0xf	Target physical rank; this field is ignored if Bit [0] = 1
Bits [11:8]	0x0 - 0xf	Target logical rank; this field is ignored if Bit [1] 1
Bits [15:12]	0x0 - 0x7	Defined by device
	0x8-0xf	Reserved

4.19 Use of the WCK Control Interface

For WCK free running operation, the **dfi_wck_en** signal and timing parameters are the same. The WCK must be synchronized initially and following commands that cause the WCK to stop. When the DFI interface changes ownership between the MC and PHY, the WCK clock must be in the DISABLED state. This applies to both the normal and free running mode and to the following interfaces: initialization, frequency change, PHY master, PHY update, and Controller update.

The timing diagrams that follow include both the MC timing and an illustration of how the PHY interprets the MC timing in the PHY clock domain. In these figures, the signals are shown changing on the positive clock edge. In the MC clock diagrams, the timing shows the DFI bus signaling. In the DRAM clock diagrams, the timing illustrates how the DRAM interprets the DFI bus. DRAM timing is shown for illustrative purposes only.

Figure 91, "WCK Timing, Read 4:1 Ratio Timing Diagram" shows the read timing relationship in a 4:1 ratio system.

FIGURE 91. *WCK Timing, Read 4:1 Ratio Timing Diagram*

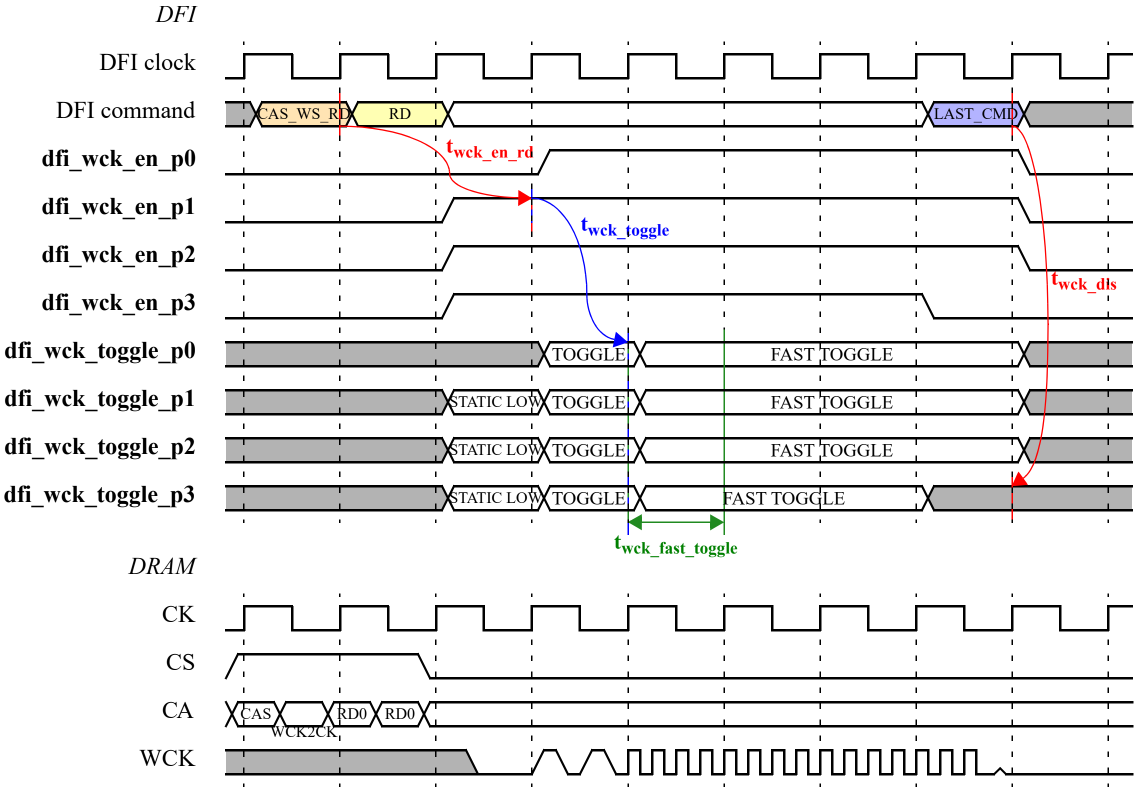
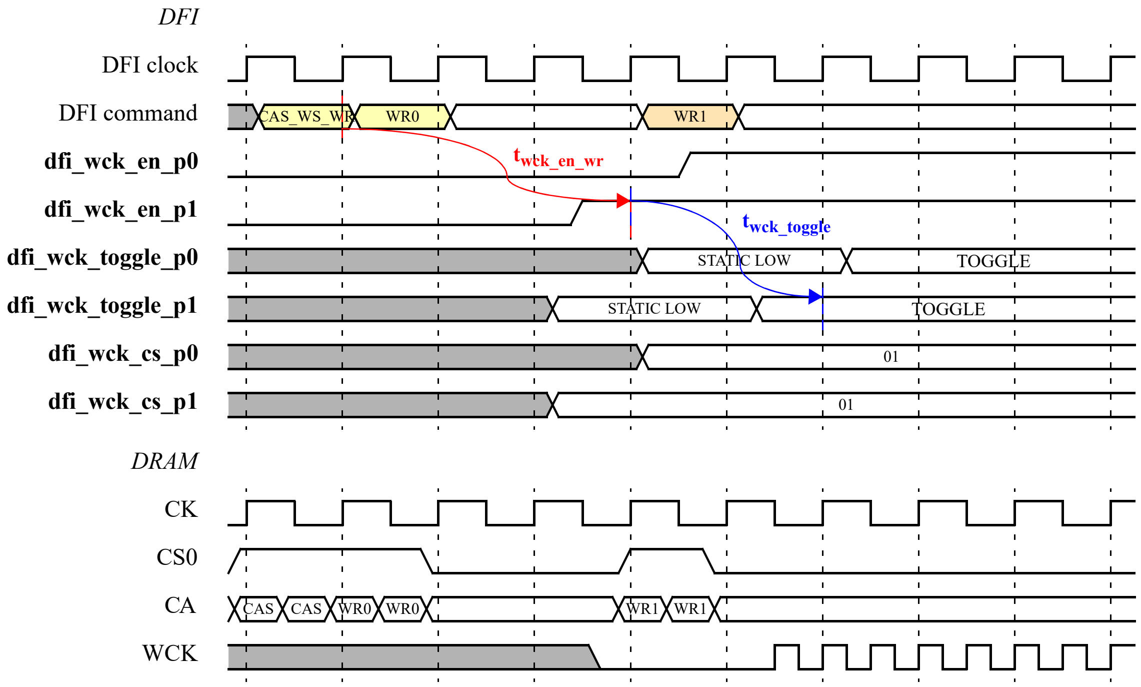


Figure 92, “WCK Timing, Write 2:1 Ratio, 2 Rank Timing Diagram” shows the write timing relationship in a 2:1 system.

FIGURE 92. *WCK Timing, Write 2:1 Ratio, 2 Rank Timing Diagram*



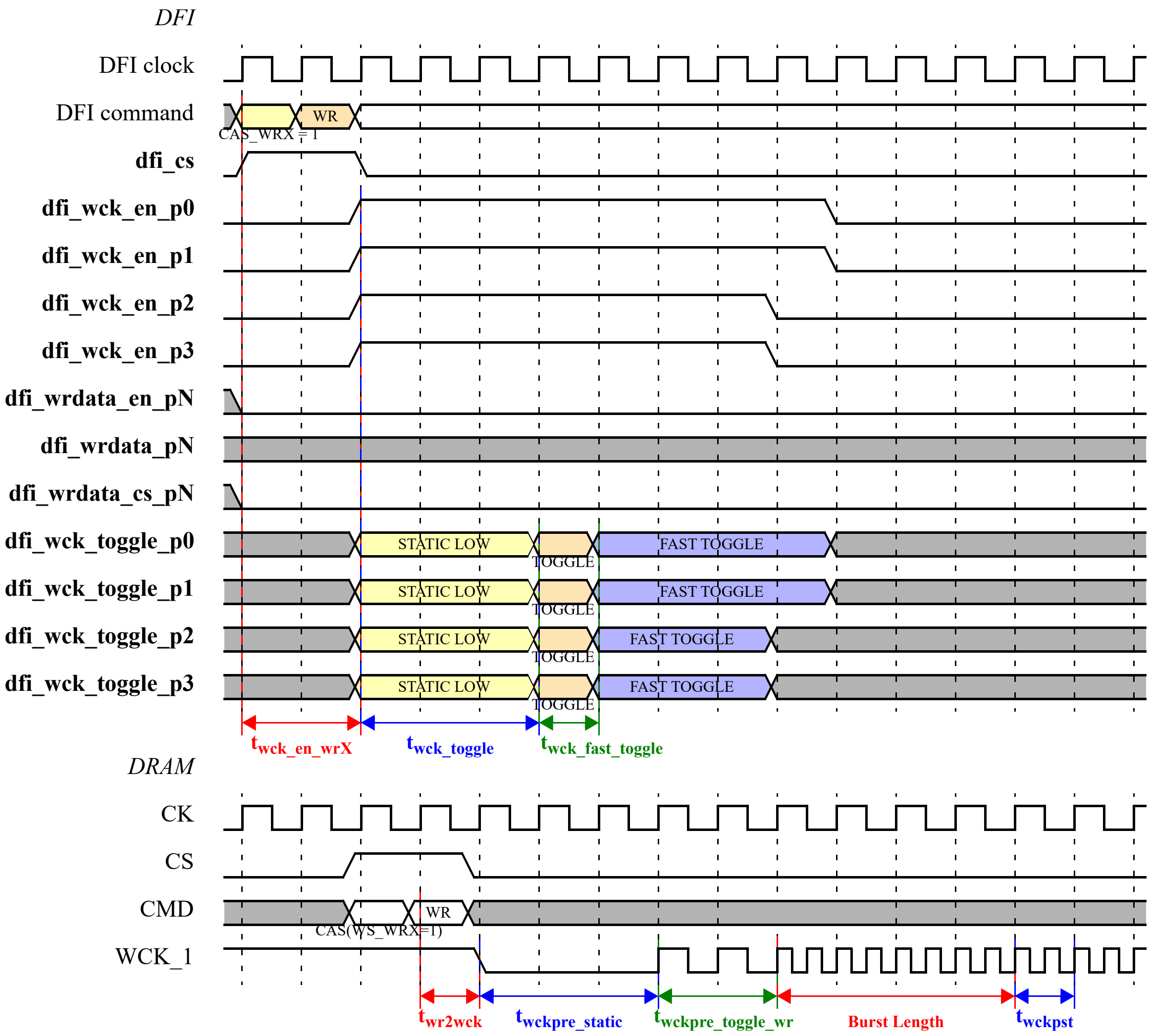
For WCK free-running operation, the WCK is initially synchronized in the same manner as normal operation. In this mode, certain commands will stop the WCK in the DRAM. In these cases, the same signal and timing applies to WCK disable as defined for normal operations; $\mathbf{t_{wck\_dis}}$ timing applies from the issuance of the command that stops the WCK.

When the DFI interface changes ownership between the MC and PHY, the WCK clock must be in the DISABLED state. This applies to both the normal and free running mode and to the following interfaces: initialization, frequency change, PHY Master, PHY update, and Controller update.

The write X function is an optional feature in LPDDR5 devices used for reducing power in write operations of repeated data patterns of all 0s or 1s. With a Write X command, the MC must drive the **dfi_wrdata_en** signal low and the

dfi_wrdata, **dfi_wrdata_mask** / **dfi_wrdata_cs** signals are all in a don't care state. The MC will need to generate the **dfi_wck_en** signal with CAS-WRX as shown in Figure 93, "LPDDR5 WriteX Function".

FIGURE 93. *LPDDR5 WriteX Function*



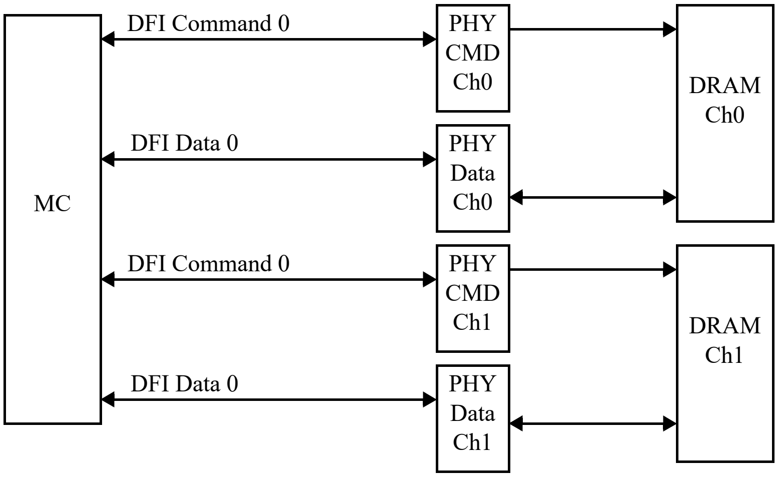
4.20 Channel Operation for Multi-Channel Memories

Certain memories can operate with independent or combine channels. If used in combined mode, the setting of the **phy**$_{\textbf{channel\_en}}$ programmable parameter indicates whether the PHY connects to a single channel or both channels when it operates in combined mode.

4.20.1 Independent Operation

Figure 94, "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 94. *LPDDR4 Independent Channels*



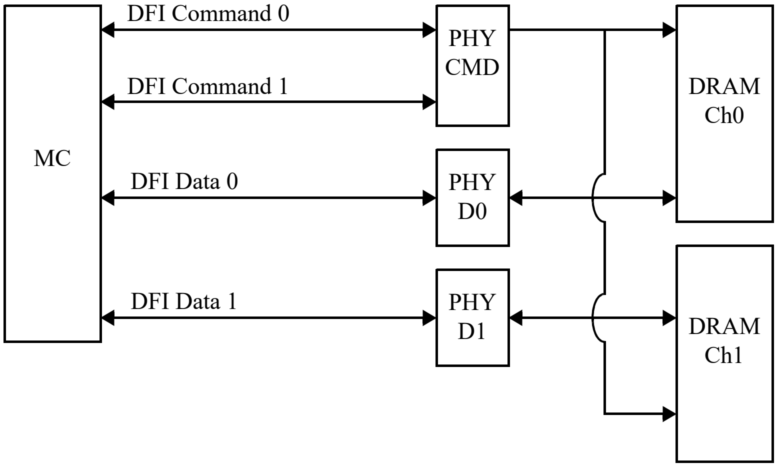
4.20.2 Combined Operation

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, low power) the PHY can connect to one channel's interfaces or to both channels' interfaces. The connectivity is defined by a programmable parameter, $\mathbf{phy_{channel\_en}}$. 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 94 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 95 applies. The MC can represent one or two memory controllers.

FIGURE 95. *LPDDR4 Combined Channels*



For combined operations, the command interface, read data interface signals, and write data interface 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.

4.20.2.1 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.

4.20.2.2 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.

4.20.2.3 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.

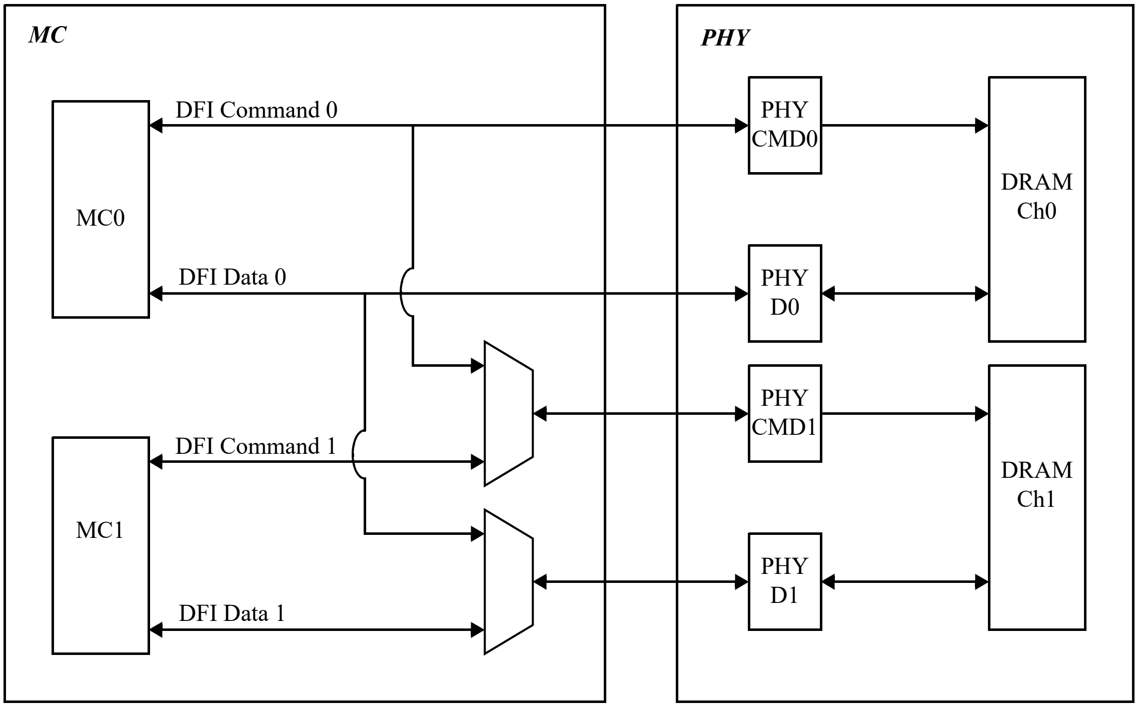
4.20.3 Multi-Configuration Support

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 96, "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 96. *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 94 and Figure 95 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.

4.21 DFI Interactions

This section describes how interactions between the Update, Status, PHY Master and Low Power Interfaces should be handled.

TABLE 47. *DFI Interactions*

Rules	During Interactions Between		Exceptions
t_{phyupd_resp}, $t_{phymstr\_resp}$	**dfi_phyupd_req**	**dfi_phymstr_req**	The PHY is permitted to have both **dfi_phyupd_req** and **dfi_phymstr_req** asserted. If the MC asserts **dfi_phymstr_ack** first, it must assert **dfi_phyupd_ack** within t_{phyupd_resp} of de-asserting **dfi_phymstr_ack**. If the MC asserts the **dfi_phyupd_ack** first, it must assert **dfi_phymstr_ack** within $t_{phymstr\_resp}$ cycles of de-asserting the **dfi_phyupd_ack** signal. The MC must never have both **dfi_phyupd_ack** and **dfi_phymstr_ack** asserted.
MC requirement to assert **dfi_phyupd_ack** in response to **dfi_phyupd_req** PHY requirement to maintain **dfi_phyupd_req** until **dfi_phyupd_ack** asserts	**dfi_phyupd_req**	**dfi_init_start**	If **dfi_init_start** and **dfi_phyupd_req** are both asserted, the MC is permitted to not assert **dfi_phyupd_ack**. In this case, the PHY should de-assert **dfi_init_complete** to initiate the frequency change protocol. In this case it must also de-assert the **dfi_phyupd_req** signal before asserting the **dfi_init_complete** signal again. Alternately, if the **dfi_init_start** and the **dfi_phyupd_req** signals are both asserted, the PHY may not de-assert the **dfi_init_complete** signal. In this case, after t_{init_start}, the frequency change request is not accepted and the MC must assert the **dfi_phyupd_ack** signal before t_{phyupd_resp}. The timing value t_{phyupd_resp} must be defined large enough to accommodate this behavior. The MC must never have both the **dfi_init_start** and the **dfi_phyupd_ack** signals asserted together.
MC requirement to assert **dfi_phyupd_ack** in response to **dfi_phyupd_req**	**dfi_phyupd_req**	**dfi_ctrlupd_req**	If **dfi_phyupd_req** and **dfi_ctrlupd_req** are both asserted, the PHY may assert **dfi_ctrlupd_ack**; the PHY then has the option to keep **dfi_phyupd_req** asserted or to de-assert **dfi_phyupd_req**. If the PHY keeps **dfi_phyupd_req** asserted, the MC should assert **dfi_phyupd_ack** after **dfi_ctrlupd_req** and **dfi_ctrlupd_ack** have both de-asserted. If the PHY de-asserts dfi_phyupd_req, it should do so while **dfi_ctrlupd_ack** is asserted. The MC should never have **dfi_ctrlupd_req** and **dfi_phyupd_ack** asserted together.
MC requirement to assert **dfi_phyupd_ack** in response to **dfi_phyupd_req**	**dfi_phyupd_req**	**dfi_lp_ctrl_req** **dfi_lp_data_req**	If **dfi_phyupd_req** asserts while **dfi_lp_ctrl_req** or **dfi_lp_data_req** are asserted, the MC must first de-assert **dfi_lp_ctrl_req** and **dfi_lp_data_req** and wait (if necessary) for **dfi_lp_ctrl_ack** and **dfi_lp_data_ack** to de-assert. After that, and before sending any data commands, the MC must assert **dfi_phyupd_ack**. Even in these cases, **dfi_lp_ctrl_req** and **dfi_lp_data_req** must meet t_{lp_resp} requirements.

TABLE 47. *DFI Interactions*

Rules	During Interactions Between		Exceptions
t_{phyupd_resp}	**dfi_phyupd_req**	**dfi_ctrlupd_req**	The PHY is permitted to assert **dfi_ctrlupd_ack** while **dfi_phyupd_req** is asserted and keep **dfi_phyupd_req** asserted. The MC must assert **dfi_phyupd_ack** within t_{phyupd_resp} cycles of a **dfi_phyupd_req** signal assertion even if a DFI controller update handshake occurs during that time.
t_{phyupd_resp}	**dfi_phyupd_req**	**dfi_lp_ctrl_req** **dfi_lp_data_req**	If **dfi_phyupd_req** is asserted or asserts while **dfi_lp_ctrl_req** and/or **dfi_lp_data_req** are asserted, t_{phyupd_resp} is counted from the first cycle in which **dfi_lp_ctrl_req**, **dfi_lp_ctrl_ack**, **dfi_lp_data_req**, and **dfi_lp_data_ack** are all de-asserted.
MC requirement to assert **dfi_phymstr_ack** in response to **dfi_phymstr_req**	**dfi_phymstr_req**	**dfi_init_start**	If the **dfi_init_start** and **dfi_phymstr_req** signals are both asserted, the MC cannot assert the **dfi_phymstr_ack** signal. In this case, the PHY may de-assert the **dfi_init_complete** signal to initiate the frequency change protocol and it must also de-assert the **dfi_phymstr_req** signal before asserting the **dfi_init_complete** signal again. Alternately, if the **dfi_init_start** and **dfi_phymstr_req** signals are both asserted, the PHY may not de-assert the **dfi_init_complete** signal. In this case, after t_{init_start}, the frequency change request is not accepted and the MC must assert the **dfi_phymstr_ack** signal before $t_{phymstr\_resp}$. The timing value $t_{phymstr\_resp}$ must be defined large enough to accommodate this behavior. The MC must never have both the **dfi_init_start** and the **dfi_phymstr_ack** signals asserted together.
MC requirement to assert **dfi_phymstr_ack** in response to **dfi_phymstr_req**	**dfi_phymstr_req**	**dfi_lp_ctrl_req**, **dfi_lp_data_req**	If **dfi_phymstr_req** asserts while **dfi_lp_ctrl_req** or **dfi_lp_data_req** are asserted, the MC must first de-assert **dfi_lp_ctrl_req** and **dfi_lp_data_req** and wait (if necessary) for **dfi_lp_ctrl_ack** and **dfi_lp_data_ack** to de-assert. After that, and before sending any data commands, the MC must assert **dfi_phymstr_ack**. Even in these cases, **dfi_lp_ctrl_req** and **dfi_lp_data_req** must meet t_{lp_resp} requirements.
$t_{phymstr\_resp}$	**dfi_phymstr_req**	**dfi_lp_ctrl_req**, **dfi_lp_data_req**	If **dfi_phymstr_req** is asserted or asserts while **dfi_lp_ctrl_req** and/or **dfi_lp_data_req** are asserted, $t_{phymstr\_resp}$ is counted from the first cycle in which **dfi_lp_ctrl_req**, **dfi_lp_ctrl_ack**, **dfi_lp_data_req**, and **dfi_lp_data_ack** are all de-asserted.
Illegal operation	**dfi_ctrlupd_req**	**dfi_init_start**	MC must never have both **dfi_ctrlupd_req** and **dfi_init_start** asserted.
Illegal operation	**dfi_lp_ctrl_req** **dfi_lp_data_req**	**dfi_init_start**	MC must never have both **dfi_lp_ctrl_req** and **dfi_init_start** asserted. MC must never have both **dfi_lp_data_req** and **dfi_init_start** asserted.

TABLE 47. *DFI Interactions*

Rules	During Interactions Between		Exceptions
Illegal operation	**dfi_ctrlupd_ack** **dfi_ctrlupd_req**	All other DFI Interfaces	If **dfi_ctrlupd_req** is asserted, all DFI signals must not change while **dfi_ctrlupd_req** remains asserted with the following exceptions: **dfi_phymstr_req**, **dfi_phymstr_cs_state**, **dfi_phymstr_state_sel**, **dfi_phymstr_type**, **dfi_ctrlmsg_req**, **dfi_ctrlmsg**, **dfi_ctrlmsg_data**, **dfi_ctrlmsg_ack**, **dfi_phyupd_req**, **dfi_phyupd_type**, **dfi_ctrlmsg_req**, **dfi_ctrlmsg**, **dfi_ctrlmsg_data**, **dfi_ctrlmsg_ack**, **dfi_error**, and **dfi_error_info**
Illegal operation	**dfi_phymstr_ack**	**dfi_lp_ctrl_req** **dfi_lp_data_req**	If **dfi_phymstr_ack** is asserted and **dfi_lp_ctrl_req** and **dfi_lp_data_req** are not asserted, the MC must not assert **dfi_lp_ctrl_req** or **dfi_lp_data_req** while **dfi_phymstr_ack** remains asserted.
Illegal operation	**dfi_phymstr_ack**	**dfi_ctrlupd_req**	If **dfi_phymstr_ack** is asserted and **dfi_ctrlupd_req** is not asserted, the MC must not assert **dfi_ctrlupd_req** while **dfi_phymstr_ack** remains asserted.
Illegal operation	**dfi_phymstr_ack**	DFI command, data and WCK interfaces	If **dfi_phymstr_ack** is asserted, the DFI bus must be idle, no commands may be issued on the command bus, no data may be transferred on the data buses, and read and write data must have reached their destinations. The **dfi_dram_clk_disable** signal is defined as part of the command and cannot change. The WCK interface can not change. The exception is that the controller can send refresh commands.
Illegal operation	**dfi_phyupd_ack**	**dfi_lp_ctrl_req** **dfi_lp_data_req**	If **dfi_phyupd_ack** is asserted and **dfi_lp_ctrl_req** and **dfi_lp_data_req** are not asserted, the MC must not assert **dfi_lp_ctrl_req** or **dfi_lp_data_req** while **dfi_phyupd_ack** remains asserted.
Illegal operation	**dfi_phyupd_ack**	DFI command, data and WCK interfaces	If **dfi_phyupd_ack** is asserted, the DFI bus must be idle, no commands may be issued on the command bus, no data may be transferred on the data buses, and read and write data must have reached their destinations. The **dfi_dram_clk_disable** signal is defined as part of the command and cannot change. The WCK interface cannot change.
Illegal operation	**dfi_init_complete** **dfi_init_start**	**dfi_ctrlmsg_req**	If **dfi_init_start** is asserted or **dfi_init_complete** is de-asserted and **dfi_ctrlmsg_req** is not asserted, the MC must not assert **dfi_ctrlmsg_req** while **dfi_init_start** remains asserted or **dfi_init_complete** remains de-asserted.
Illegal operation	**dfi_init_complete** **dfi_init_start**	**dfi_dram_clk_disable**	If **dfi_init_start** is asserted or **dfi_init_complete** is de-asserted during a frequency change, **dfi_dram_clk_disable** must not change while **dfi_init_start** remains asserted or **dfi_init_complete** remains de-asserted.

TABLE 47. *DFI Interactions*

Rules	During Interactions Between		Exceptions
Illegal operation	**dfi_lp_ctrl_ack** **dfi_lp_ctrl_req** **dfi_lp_ctrl_wakeup** **dfi_lp_data_ack** **dfi_lp_data_req** **dfi_lp_data_wakeup**	**dfi_ctrlmsg_req**	If **dfi_lp_ctrl_req**, **dfi_lp_ctrl_ack**, **dfi_lp_data_req** or **dfi_lp_data_ack** are asserted and the wakeup time for either interface is greater than or equal to **dfi**$_{\mathbf{lp\_wakeup\_threshold}}$ and **dfi_ctrlmsg_req** is not asserted, the MC must not assert **dfi_ctrlmsg_req** while these signals remain asserted. If **dfi_ctrlmsg_req** is asserted and **dfi_lp_ctrl_req** or **dfi_lp_data_req** are not asserted, **dfi_lp_ctrl_req** or **dfi_lp_data_req** cannot be asserted with a wakeup time that is greater than or equal to **dfi**$_{\mathbf{lp\_wakeup\_threshold}}$ while **dfi_ctrlmsg_req** remains asserted.
Illegal operation	**dfi_lp_ctrl_ack** **dfi_lp_ctrl_req**	**dfi_dram_clk_disable**	If **dfi_lp_ctrl_req** and/or **dfi_lp_ctrl_ack** is asserted, **dfi_dram_clk_disable** must not change while **dfi_lp_ctrl_req** or **dfi_lp_ctrl_ack** remains asserted.

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 Command Interface signals fall into the standard signal category because the Command 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**

   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 48, "Signal Group Divisions" shows that each DFI signal is categorized into a signal group.

TABLE 48. *Signal Group Divisions*

Signal	Signal Group
dfi_2n_mode (or dfi_2n_mode_pN)	Standard
dfi_act_n (or dfi_act_n_pN)	Standard
dfi_address (or **dfi_address_pN**)	Standard
dfi_alert_n (or **dfi_alert_n_aN**)	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_ctrlmsg	State-Retaining
dfi_ctrlmsg_ack	State-Retaining
dfi_ctrlmsg_data	State-Retaining
dfi_ctrlmsg_req	State-Retaining
dfi_ctrlupd_ack	State-Retaining
dfi_ctrlupd_req	State-Retaining
dfi_disconnect_error	Standard
dfi_dram_clk_disable (or **dfi_dram_clk_disable_pN**)	Standard
dfi_error	Standard
dfi_error_info	Standard
dfi_freq_fsp	Timer-Based
dfi_freq_ratio	Timer-Based
dfi_frequency	Timer-Based
dfi_init_complete	State-Retaining
dfi_init_start	State-Retaining
dfi_lp_ctrl_ack	State-Retaining
dfi_lp_ctrl_req	State-Retaining
dfi_lp_ctrl_wakeup	State-Retaining

TABLE 48. *Signal Group Divisions*

Signal	Signal Group
dfi_lp_data_ack	State-Retaining
dfi_lp_data_req	State-Retaining
dfi_lp_data_wakeup	State-Retaining
dfi_odt (or **dfi_odt_pN**)	Standard
dfi_parity_in (or **dfi_parity_in_pN**)	Standard
dfi_phymstr_ack	State-Retaining
dfi_phymstr_cs_state	State-Retaining
dfi_phymstr_req	State-Retaining
dfi_phymstr_state_sel	State-Retaining
dfi_phymstr_type	State-Retaining
dfi_phyupd_ack	State-Retaining
dfi_phyupd_req	State-Retaining
dfi_phyupd_type	State-Retaining
dfi_ras_n (or **dfi_ras_n_pN**)	Standard
dfi_rddata (or **dfi_rddata_wN**)	Standard
dfi_rddata_cs (or **dfi_rddata_cs_pN**)	State-Retaining
dfi_rddata_dbi (or **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_reset_n (or **dfi_reset_n_pN**)	Standard
dfi_wck_cs (or **dfi_wck_cs_pN**)	Standard
dfi_wck_en (or **dfi_wck_en_pN**)	Standard
dfi_wck_toggle (or **dfi_wck_toggle_pN**)	Standard
dfi_we_n (or **dfi_we_n_pN**)	Standard
dfi_wrdata (or **dfi_wrdata_pN**)	Standard
dfi_wrdata_cs (or **dfi_wrdata_cs_pN**)	Standard
dfi_wrdata_ecc (or **dfi_wrdata_ecc_pN**)	Standard
dfi_wrdata_en (or **dfi_wrdata_en_pN**)	Standard
dfi_wrdata_mask (or **dfi_wrdata_mask_pN**)	Standard

6.0 Glossary

TABLE 49. *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, DDR5 and LPDDR5 systems.
Column Selection (CAS)	Column address strobe.
Command Address (CA) Signals	Includes the following command 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 command 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 Bus Idle	The DFI bus is considered idle when the command 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.
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. For 3DS, this is referred to as the “Physical Rank.
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.
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.

TABLE 49. *Glossary of Terms*

Term	Definition
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 FSP Width	For LPDDR4, this is 1 bit. For LPDDR5, this is 2 bits.
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 (or 1/1 for LPDDR5 memory systems) the period of the DFI clock frequency. These clocks must be aligned in phase.
DFI PHY Data Frequency	Defines the period of the of the data frequency of the PHY. For memory systems other than LPDDR5, this is the same as the clock frequency. For LPDDR5 memory systems, the period of the DFI PHY data is 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 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.
DFI WCK Width	The number of individual WCK control signal interfaces driven on the DFI bus. This is generally the number of WCKs supported on the memory interface.
$\mathrm{dfi}_{\mathrm{rw\_length}}$	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.

TABLE 49. *Glossary of Terms*

Term	Definition
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. The frequency ratio applies to the command and DFI data clock domain (PHY frequency ratio) or to the DFI data clock domain (data frequency ratio) only. DFI clock domain signals do not operate at a clock ratio, they are always DFI clock based. Refer to Section 2.1, “Clocking” for more details. 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 4.10, “Frequency Ratios Across the DFI”.
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.
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.