ZeBu DFI LPDDR5 Interface User Manual

Version V-2024.03-SP1, February 2025

Feedback

Contents

Feedback

1

About This Manual

This manual describes how to use the ZeBu DFI interface for easier integration between Memory Controller (MC) and PHY.

Related Documentation

For more relevant information about each memory protocol associated with the zDFI, please refer to the related JEDEC specifications.

For more relevant information about the DFI protocol, please refer to the latest MIPI DFI specification.

Synopsys Statement on Inclusivity and Diversity

Synopsys is committed to creating an inclusive environment where every employee, customer, and partner feels welcomed. We are reviewing and removing exclusionary language from our products and supporting customer-facing collateral. Our effort also includes internal initiatives to remove biased language from our engineering and working environment, including terms that are embedded in our software and IPs. At the same time, we are working to ensure that our web content and software applications are usable to people of varying abilities. You may still find examples of non-inclusive language in our software or documentation as our IPs implement industry-standard specifications that are currently under review to remove exclusionary language.

2 Introduction

DDR PHY Interface is an industry-standard interface protocol for easier integration between Memory Controller (MC) and PHY. DFI defines the signals and timing parameters required to transfer control information and data. These parameters are constrained by the MC and/or the PHY. Current version is DFI 5.1. It supports DDR3, DDR4, DDR5, GDDR6, HBM2/2E, LPDDR1, LPDDR2, LPDDR3, LPDDR4, LPDDR5.

This section explains the following topics:

- Overview
- Features
- Limitations

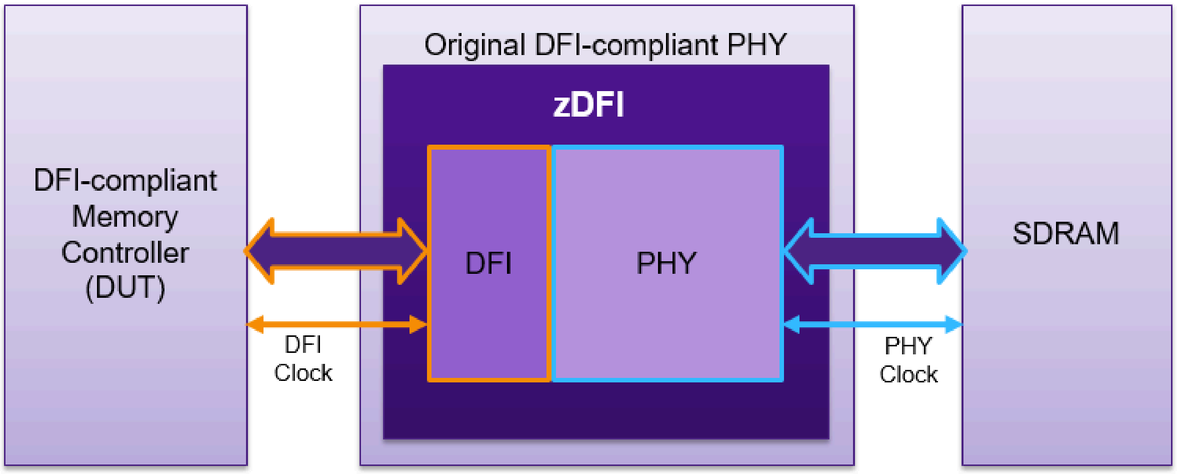
Overview

zDFI is a PHY model, compliant with the DFI standard, for ZeBu DDR Memory Models. It is delivered as an encrypted SystemVerilog file, including a SystemVerilog module called zdfi. One package is delivered for all DDR families.

zDFI supports integration of a custom DFI-compliant Memory Controller (DUT). Replace your current PHY with the zdfi module assigning proper values to the signal width and timing parameters.

The following figure illustrates the DFI Interface:

Figure 1 DFI Interface


DFI-compliant Memory Controller (DUT)
DFI Clock
Original DFI-compliant PHY
zDFI
DFI
PHY
PHY Clock
SDRAM

Features

The following are the features supported by DFI:

Table 1 Supported Features

SDRAM Family	**Comment**
Read transactions	Transmitted to the DDR model.
Write transactions	Transmitted to the DDR model.
Configuration operations (MRW/MRR)	Transmitted to the DDR model.
Configuration operations detection (MRW/MRR)	The zDFI detects these accesses and extracts useful information (Burst length, Write latency, read latency).
Status Interface	Only dfi_init_start/complete
Data Bus Inversion (DBI/DMI)	

Table 1 Supported Features (Continued)

SDRAM Family	Comment
1:2 clock ratio1:4 clock ratio	LPDDR5 devices can switch from 1:2 to 1:4 mode. Unlike other zDFI, zDFI_LPDDR5_HDR supports both 1:2 and 1:4.*Note:* In order to switch between clock ratio 1:2 and 1:4, no specific action on the zDFI is required. When a clock ratio switch is sent to the memory by the controller (accessing bit MR18[7]), the zDFI will catch this access on the fly and switch to the new clk ratio at the same time as the memory.Also note that the idfi_freq_ratio input port is not used internally, i.e., the values input to idfi_freq_ratio will be ignored.
Single channel support	
WriteX support	
Write32 bubble mode support	
Frequency Set Points	
Link ECC	

Limitations

The following are the limitations of the DFI interface:

- DFI update interface.
- DFI optional protocols:
  - Cyclic redundancy check (CRC)
  - System frequency change
  - DFI low power interface
  - Error interface
  - Training interface
- Multiple channel support.
- The zDFI does not support any change of tWCK2DQO/tWC2DQI/tWCK2CK in the zLPDDR5 device. They must remain at their default values.

3

Integration Methodology

This section explains how to integrate Memory Controller (MC) and PHY under the following topics:

- Integration Process
- Connecting Clocks
- Design Topologies

Integration Process

To integrate a Memory Controller (MC) and PHY, perform the following steps:

1. Gather the following information for parameterization of the zDFI model (bus sizes and timing parameters):

   If the timing parameters are not available, simulate your design using the zDFI model and the memory simulation model.

   Use the default timing parameters in the first stage of the integration process.

   Extract the timing parameters from the waveforms. See zDFI Timing Parameters for a methodology to compute DFI_T_PHY_WRDATA and DFI_T_PHY_WRLAT.

   Use the memory probe vector to extract MEM_WRITE_LATENCY and MEM_READ_LATENCY. See *ZeBu LPDDR5 Memory Models User Manual* for more information.

   Once the value of DFI_T_PHY_WRDATA, DFI_T_PHY_WRLAT and MEM_WRITE_LATENCY is computed, compute DFI_T_CTRL_DELAY as detailed in zDFI Timing Parameters.

   Modify your simulation with the new values of the timing parameters, relaunch simulation and check that write and read access are correct.

2. Emulate your design using the zDFI model and the VS DDR model with the same parameter values as above.

Note:

The zDFI is intended to work together with the zLPDDR5 model IP. These are simplified models designed for emulation on ZeBu. Trying to run the zDFI with a third-party memory simulation model may be difficult or impossible

Connecting Clocks

The following are the two clock inputs on zDFI:

- *dfi_clk*: Generated by the MC (DFI controller).
- *mem_wck_1x4*: is user-generated.

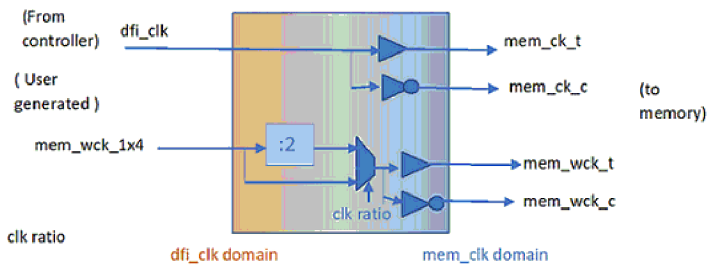
The *mem_wck_1x4* 4x is faster than *dfi_clk*. In clock ratio 1:4, 4-phase zDFI, *mem_wck_t* is a copy of *mem_wck_1x4*. In clock ratio 1:2, 2-phase zDFI, *mem_wck_t* is *mem_wck_1x4* divided by 2.

The clock domain for mem_wck_1x4 should be the same as that for dfi_clk (derived from the same primary clock). Also, the two clocks must have their rising edges aligned.

In clock ratio 1:4, 4-phase zDFI, mem_wck_t is a copy of mem_wck_1x4. In clock ratio 1:2, 2-phase zDFI, mem_wck_t is mem_wck_1x4 divided by 2.

The following is the clock structure inside the zDFI:

Figure 2 Clock Structure



As shown in the above figure, the following are the configuration requirements of the clocks:

- The *mem_wck_t* output is the positive side of the differential clock. Ensure that it is connected to the positive clock input of the memory. On the memory model, the input is named WCK_t.
- The *mem_wck_c* output is the negative side of the differential clock. Ensure that it is connected to the negative clock input of the memory. On the memory model, the input is named WCK_c.
- The *mem_ck_t* output is the positive side of the differential clock. Ensure that it is connected to the positive clock input of the memory. On the memory model, the input is named CK_t. The mem_ck_t signal is a copy of dfi_clk.
- The *mem_ck_c* output is the negative side of the differential clock. Ensure that it is connected to the negative clock input of the memory. On the memory model, the input is named CK_c.

Design Topologies

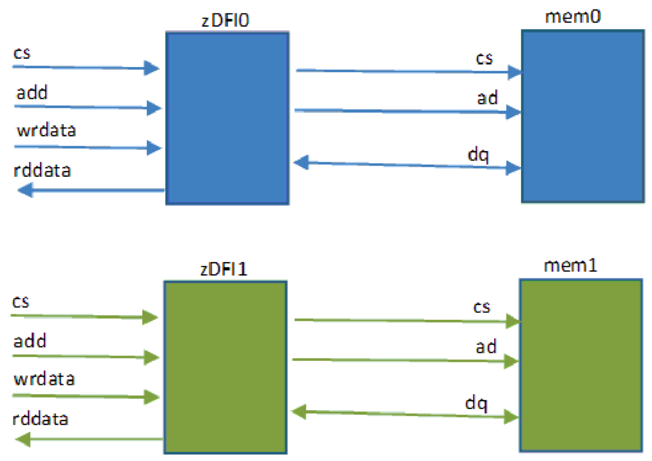
The following design topologies are available in the zDFI Interface for LPDDR5:

- Independent Channels / Independent Memories
- Parallel Channels

Independent Channels / Independent Memories

In this topology, two channels or two memories do not share the same address bus. In this case, instantiate one DFI per channel.

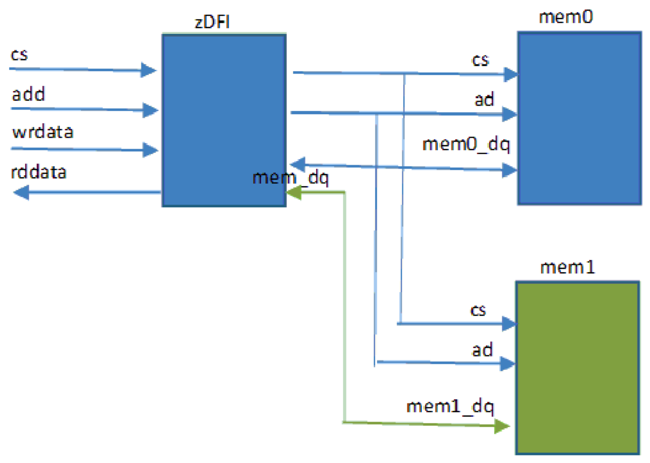
Figure 3 2 Independent Channels / 2 Independent Memories



Parallel Channels

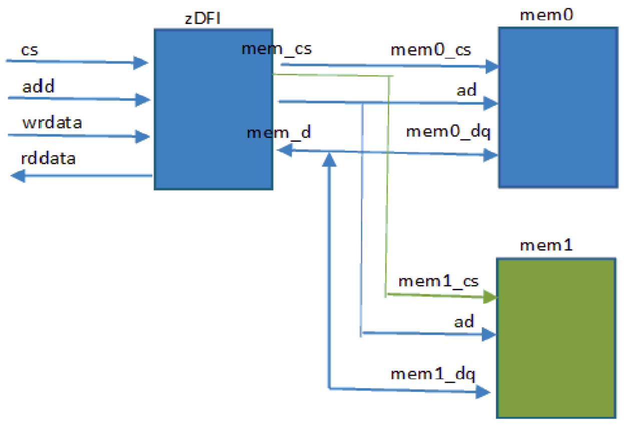
In this topology, two channels or two memories share the same address buses. They also share the same clock enable. The mem_dq bus of the zDFI is split in 2 to/from the 2 memories (mem_dq = {mem1_dq,mem0_dq}). In this case, instantiate one DFI with double data size.

Figure 4 Instantiate One DFI With Double Data Size



However, in this topology, two channels or two memories share the same address bus but not the same chip select (mem_cs = {mem1_cs,mem0_cs}). Also, the data bus is shared (mem_dq = mem0_dq = mem1_dq). In this case, instantiate one DFI with double chip select size.

Figure 5 Instantiate one DFI with Double Chip Select Size



4

zDFI Parameters

This section explains the following topics:

- zDFI Timing Parameters
- zDFI Vector Width Parameters

zDFI Timing Parameters

The zDFI memory timing parameters can also be defined at runtime using the zebu_TCFG register bank.

The following table describes the zDFI timing parameters and corresponding mapping of the zebu_TCFG register bank:

Table 2 zDFI Timing Parameters

Parameter	zebu_TCFG Register Mapping	Description
DFI_T_CTRL_DELAY	Zebu_TCFG[7:0]	Delay between command on the DFI interface and command reaching the memory. Minimum value is 3.
DFI_T_PHY_RDLAT	Zebu_TCFG[15:8]	Maximum delay between idfi_rddata_en_pN and odfi_rddata_valid_wN.
DFI_T_PHY_WRLAT	Zebu_TCFG[23:16]	Delay between DFI write command and DFI write_en.
DFI_T_PHY_WRDATA	Zebu_TCFG[31:24]	Delay between DFI write_en and DFI wrdata.

DFI_T_PHY_WRDATA, DFI_T_PHY_WRLAT, DFI_T_PHY_RDLAT: All delay value is calculated based on mem_wck clock unit. This mem_wck clock is the same frequency as the PHY clock.

DFI_T_CTRL_DELAY: Delay is calculated based on mem_ck unit.

MEM_WRITE_LATENCY and MEM_READ_LATENCY is the lpddr5 protocol latency.

Since, mem_ck and mem_wck are generated by zDFI, user doesn't need to worry about it. Details are provided in the section 2.2.

All of the above values are used as default values.

The zDFI memory timing parameters can also be defined at runtime using the zebu_TCFG register bank. The mapping of this register is as follows:

DFI_T_CTRL_DELAY	zebu_TCFG[7:0]
DFI_T_PHY_RDLAT	Zebu_TCFG[15:8]
DFI_T_PHY_WRLAT	Zebu_TCFG[23:16]
DFI_T_PHY_WRDATA	Zebu_TCFG[31:24]
USE_RDDATA_EN: Follow rddata_en shape and consider DFI_T_PHY_RDLAT as strict value number in DFI clock cycle (USE_RDLAT_EN must be set to 1).	Zebu_TCFG[32]
USE_RDLAT_EN: DFI_T_PHY_RDLAT as strict value number in DFI clock cycle, if USE_RDDATA_EN is not enabled, it will use the rddata_en_p0 as reference.	Zebu_TCFG[33]
Backdoor configuration to select clock ratio. Default value: 1 (1:1x4, 0:1x2)	Zebu_TCFG[34]

Each timing parameter is accessible with a backdoor access. The path to the zebu_TCFG register is the following.

```
<path_to_zDFI_instance>.zebu_TCFG.zebuReg
```

Any update on any timing parameter at runtime is overwriting the parameter value defined at compilation time.

Note: Usage of USE_RDDATA_EN and USE_RDLAT_EN is very restricted and reserved to a very specific use-case. By default the DFI_T_PHY_RDLAT value is don't care.

This section explains the following topics:

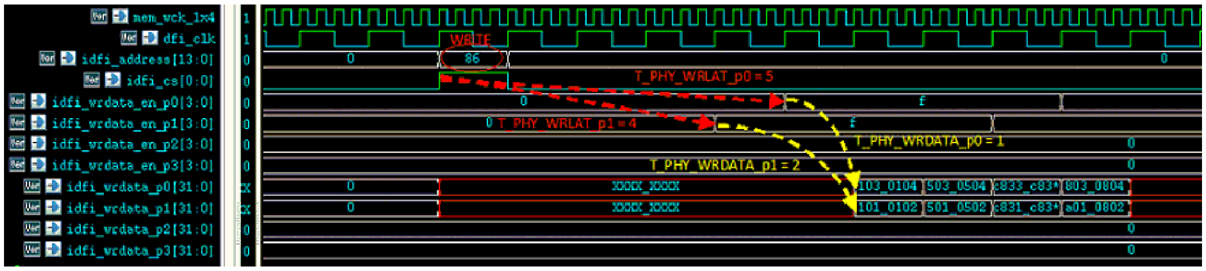
- DFI Timing Parameters for 1:2 Clock Ratio
- DFI Timing Parameters for 1:4 Clock Ratio
- zDFI Memory Timing Parameters
- Timing Constraints

DFI Timing Parameters for 1:2 Clock Ratio

The write latency is computed from the WRITE command. As per the LPDDR5 command truth table, a WRITE is recognized by values 0x2 or 0xA (MASK WRITE), 0x6 or 0xE (WRITE), 0x4 (WRITE32) on the idfi_address bus.

The following figures illustrates how to compute DFI_T_PHY_WRLAT and DFI_T_PHY_WRDATA from a reference simulation waveform.

Figure 6 Timing Parameters with Clock Ratio 1:2



In the above example:

```
DFI_T_PHY_WRLAT  = T_PHY_WRLAT_p0+T_PHY_WRLAT_p1  = 5+4  = 9
DFI_T_PHY_WRDATA = T_PHY_WRDATA_p0+T_PHY_WRDATA_p1 = 1+2 = 3
```

For the above example, calculate the following values:

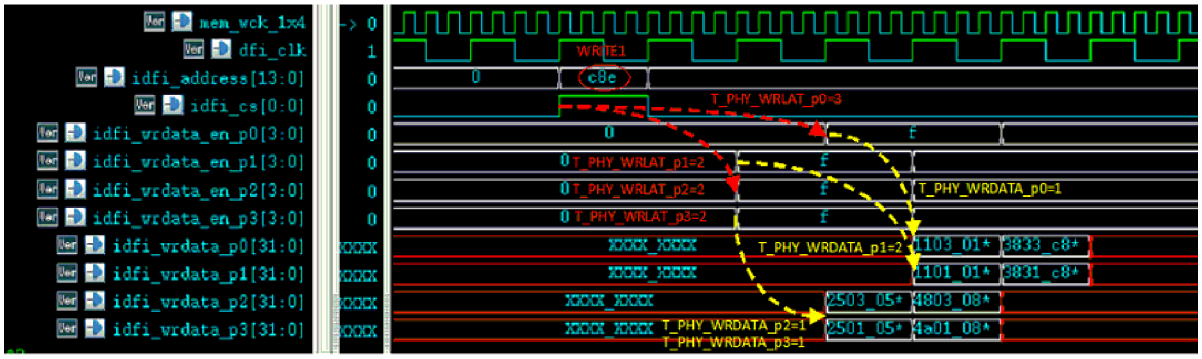
- T_PHY_WRLAT_p0: number of dfi_clk cycles between WRITE command and idfi_wrdata_en_p0.
- T_PHY_WRLAT_p1: number of dfi_clk cycles between WRITE command and idfi_wrdata_en_p1.
- T_PHY_WRDATA_p0: number of dfi_clk cycles between idfi_wrdata_en_p0 and idfi_wrdata_p0.
- T_PHY_WRDATA_p1: number of dfi_clk cycles between idfi_wrdata_en_p1 and idfi_wrdata_p1.

DFI Timing Parameters for 1:4 Clock Ratio

The write latency is computed from the WRITE command. Per the LPDDR5 command truth table, WRITE is recognized by values 0x2 or 0xA (MASK WRITE), 0x6 or 0xE (WRITE), 0x4 (WRITE32) on the idfi_address bus.

The following figures illustrates how to compute DFI_T_PHY_WRLAT and DFI_T_PHY_WRDATA from a reference simulation waveform:

Figure 7 Timing parameters with clock ratio 1:4



In the above example:

```
DFI_T_PHY_WRLAT =
 T_PHY_WRLAT_p0+T_PHY_WRLAT_p1+T_PHY_WRLAT_p2+T_PHY_WRLAT_p3 = 2+2+2+3 =
 9
DFI_T_PHY_WRDATA =
 T_PHY_WRDATA_p0+T_PHY_WRDATA_p1+T_PHY_WRDATA_p2+T_PHY_WRDATA_p3 =
 1+2+1+1 = 5
```

For the above example, calculate the following values:

- T_PHY_WRLAT_p0: number of dfi_clk cycles between WRITE command and idfi_wrdata_en_p0
- T_PHY_WRLAT_p1: number of dfi_clk cycles between WRITE command and idfi_wrdata_en_p1
- T_PHY_WRLAT_p2: number of dfi_clk cycles between WRITE command and idfi_wrdata_en_p2
- T_PHY_WRLAT_p3: number of dfi_clk cycles between WRITE command and idfi_wrdata_en_p3
- T_PHY_WRDATA_p0: number of dfi_clk cycles between idfi_wrdata_en_p0 and idfi_wrdata_p0
- T_PHY_WRDATA_p1: number of dfi_clk cycles between idfi_wrdata_en_p1 and idfi_wrdata_p1
- T_PHY_WRDATA_p2: number of dfi_clk cycles between idfi_wrdata_en_p2 and idfi_wrdata_p2)
- T_PHY_WRDATA_p3: number of dfi_clk cycles between idfi_wrdata_en_p3 and idfi_wrdata_p3)

zDFI Memory Timing Parameters

You can define the zDFI memory timing parameters at compile time.

The memory timing parameter values are default values. They are overridden by MRW commands sent to the memory through the DFI.This helps the controller perform the MRW command to set the latency.

Any mismatch between the actual latencies programmed in the LPDDR5 memory and the related parameters of the zDFI lead to an incorrect behavior.

The following table describes the parameters for zDFI memory timing:

Table 3 Memory Timing Parameters

Parameter	**Description**
MEM_WRITE_LATENCY	Must match the LPPDDR5 settings (lowest value).
MEM_READ_LATENCY	Must match the LPDDR5 settings (lowest value).
MEM_BURST_LENGTH	Must match the LPDDR5 settings.0= on-the-fly BL.

Timing Constraints

The following lists the general timing constraint:

```
CKR*(DFI_T_CTRL_DELAY+MEM_WRITE_LATENCY)≥DFI_T_PHY_WRLAT+DFI_T_PHY_WRDATA
 + zDFI_LATENCY
DFI_T_CTRL_DELAY ≥ 3
1:2 clock ratio: CKR=2, zDFI_LATENCY=4
1:4 clock ratio: CKR=4, zDFI_LATENCY=5
```

For more details about how this equation is computed, see Appendix: Calculating Timing Constraints.

zDFI_LATENCY: It is an internal delay of zDFI model and user doesn't need to know in which clock unit.

It is advised to use the smallest possible write latency of the memory (according to the JEDEC specification) as MEM_WRITE_LATENCY. For LPDDR5, this value is 4 in 1:2 clock ratio and 2 in 1:4 clock ratio.

The simplified equations are:

For 1:2 clock ratio

```
2*DFI_T_CTRL_DELAY ≥ DFI_T_PHY_WRLAT+DFI_T_PHY_WRDATA -4
DFI_T_CTRL_DELAY ≥ 3
```

For 1:4 clock ratio

```
4*DFI_T_CTRL_DELAY ≥ DFI_T_PHY_WRLAT+DFI_T_PHY_WRDATA-3
DFI_T_CTRL_DELAY ≥ 3
```

In the majority of cases, due to the CKR multiplying factor, the constraint on 1:2 clock ratio is stronger than on 1:4. It is not required to meet the constraint on 1:2, if only 1:4 write access operations are performed during the run.

zDFI Vector Width Parameters

The following table describes the zDFI width parameters and their default value:

Table 4 Width Parameters

Parameter	Default Value (1:2 Clock Ratio)
DFI_ADDRESS_WIDTH	14
DFI_CHIP_SELECT_WIDTH	2
DFI_DATA_ENABLE_WIDTH	4
DFI_DATA_WIDTH	32
DFI_READ_DATA_VALID_WIDTH	1

The default values are suitable for a zDFI connected to a dual channel LPDDR5 device with 16 data bits per channel.

The following are the constraints regarding bus sizes:

- idfi_cs, idfi_reset_n / mem_cs and mem_reset_n all have the same size parameterized by DFI_CHIP_SELECT_WIDTH.
- mem_ca is a DDR signal. Therefore, it has half the size of idfi_address, parameterized by DFI_ADDRESS_WIDTH.
- mem_dq is a DDR signal. Therefore, it has half the size of idfi_wrdata_pN/ odfi_rddata_wN, parameterized by DFI_DATA_WIDTH
- mem_rdqs_t, mem_rdqs_c, mem_wck_t, mem_wck_c and mem_dmi are 8 times smaller than mem_dq (1 bit per mem_dq byte).

- idfi_wrdata_mask_pN and odfi_rddata_dbi_wN are 8 times smaller than idfi_wrdata_pN/odfi_rddata_wN(1 bit per idfi_wrdata_pN/odfi_rddata_wN).
- All of them are parameterized by DFI_DATA_WIDTH.

5
DFI Ports

When a port is not used with a given memory type, input ports should be tied to 0 and output ports should be left unconnected.

This section describes the following DFI interfaces:

- DFI Control Interface
- DFI Write Data Interface
- DFI Read Data Interface
- DFI Status Interface
- zDFI Memory Interface
- DFI Probe Vector

DFI Control Interface

The following table lists the DFI control interface:

Table 5 DFI Control Interface

Interface Type	**Size**	**Signal Name**	**Description**
input	1	rst_n	global hardware reset, active low
input	1	dfi_clk	dfi clock input
input	1	mem_wck_1x4	memory clock input
input	DFI_ROW_WIDTH	idfi_address	row address input
input	DFI_CHIP_SELECT_WIDTH	idfi_cs	chip select input

WCK Control Interface

The following table lists the WCK control interface:

Table 6 DFI Control Interface

Interface Type	Size	Signal Name	Description
input	1	idfi_wck_cs_pN	WCK select per phase. Unused internally
input	1	idfi_wck_en_pN	WCK enable per phase. Unused internally
input	MEM_DQS_WIDTH*2	idfi_wck_toggle_pN	WCK enable toggling per phase

DFI Write Data Interface

The following table lists the DFI write data interface:

Table 7 DFI Write Data Interface

Interface Type	Size	Signal Name	Description
input	DFI_DATA_ENABLE_WIDTH	idfi_wrdata_en_pN	write data enable input per phase.
input	DFI_DATA_WIDTH	idfi_wrdata_pN	write data input per phase
input	DFI_DATA_WIDTH/8	idfi_wrdata_mask_pN	write data mask input per phase. Also used for DBI.
input	DFI_CHIP_SELECT_WIDTH * DFI_DATA_ENABLE_WIDTH	idfi_wrdata_cs_pN	Unused internally

DFI Read Data Interface

The following table lists the DFI read data interface:

Table 8 DFI Read Data Interface

Interface Type	Size	Signal Name	Description
input	DFI_DATA_ENABLE_WIDTH	idfi_rddata_en_pN	read data enable input per phase.
output	DFI_DATA_WIDTH	odfi_rddata_wN	read data output per phase
output	DFI_READ_DATA_VALID_WIDTH	odfi_rddata_valid_wN	read data valid output per phase. All bits are identical
Output	DFI_DATA_WIDTH /8	odfi_rddata_dbi_wN	read data bus inversion per phase
Input	DFI_CHIP_SELECT_WIDTH * DFI_DATA_ENABLE_WIDTH	idfi_rddata_cs_pN	Unused internally

DFI Status Interface

The following table lists the DFI status interface:

Table 9 DFI Status Interface

Interface Type	Size	Signal Name	Description
Input	1	idfi_dram_clk_disable	clock disable. Not used in zDFI. Expect optimization at synthesis
output	1	odfi_init_complete	init complete output
input	1	idfi_init_start	init start input
input	2	idfi_freq_ratio	unused internally
input	2	idfi_freq_fsp	unused internally
input	2	idfi_freq_fsp	unused internally

zDFI Memory Interface

The following table lists the interface of zDFI on the memory side:

Table 10 zDFI Memory Interface

Interface Type	Size	Signal Name	Description
output	MEM_CK_WIDTH	mem_ck_t	memory clock differential input, positive side of the pair
output	MEM_CK_WIDTH	mem_ck_c	memory clock differential input, negative side of the pair
output	DFI_CHIP_SELECT_WIDTH	mem_cs	memory chip select
output	DFI_COL_WIDTH/2	mem_ca	memory address.
inout tri0	DFI_DATA_WIDTH /2	mem_dq	memory data width (pull down).
output	DFI_DATA_WIDTH/64	mem_wck_t	memory write clock differential output, positive side of the pair.
output	DFI_DATA_WIDTH/64	mem_wck_c	memory write clock differential output, negative side of the pair.
input	DFI_DATA_WIDTH/64	mem_rdqs_t	memory data strobe differential input, positive side of the pair.
input	DFI_DATA_WIDTH/64	mem_rdqs_c	memory data strobe differential input, negative side of the pair.
inout	DFI_DATA_WIDTH / 16	mem_dmi	memory data mask / data bus inversion.
output	DFI_CHIP_SELECT_WIDTH	mem_reset_n	memory reset. Active low.

DFI Probe Vector

The following table describes the complete mapping of the debug vector.

Table 11 DFI Probe Vector

Probe Vector	Signal Name
dfi_probe[0]	dfi_write_command

Table 11 DFI Probe Vector (Continued)

Probe Vector	Signal Name
dfi_probe[1]	dfi_read_command
dfi_probe[2]	dfi_mrw1_command
dfi_probe[3]	dfi_mrw2_command
dfi_probe[4]	mem_write_command
dfi_probe[5]	mem_write32_command
dfi_probe[6]	mem_read_command
dfi_probe[7]	mem_read32_command
dfi_probe[15: 8]	burst_length_otf[7:0]
dfi_probe[23: 16]	burst_length_otf_delay[7:0]
dfi_probe[31: 24]	burst_length_actual[7:0]
dfi_probe[39: 32]	write_latency_detected[7:0]
dfi_probe[47: 40]	read_latency_detected[7:0]
dfi_probe[48]	wrecc_en
dfi_probe[49]	rdecc_en
dfi_probe[50]	mem_wrdata_out_cnt_start
dfi_probe[59: 51]	mem_wrdata_out_cnt
dfi_probe[60]	write_in_progress[-2]
dfi_probe[61]	write_in_progress[-1]
dfi_probe[62]	write_in_progress[0]
dfi_probe[63]	write_in_progress[1]
dfi_probe[64]	mem_dq_enable
dfi_probe[65]	mem_dqs_enable
dfi_probe[66]	|phy_wrdata_en_del_r
dfi_probe[67]	mem_wrdata_enable

Table 11 DFI Probe Vector (Continued)

Probe Vector	Signal Name
dfi_probe[68]	~empty_resynch_data_fifo_1_2
dfi_probe[69]	~empty_resynch_data_en_fifo_1_4
dfi_probe[70]	~empty_resynch_data_fifo_1_2
dfi_probe[71]	~empty_resynch_data_en_fifo_1_4
dfi_probe[73:72]	FSP_OP
dfi_probe[74]	request_read
dfi_probe[75]	ckratio_1_2_detected (0:1x4, 1:1x2)
dfi_probe[76]	rddata_fifo_en_1_2
dfi_probe[77]	rddata_fifo_en_1_4
dfi_probe[78]	rddata_p1_first_out
dfi_probe[79]	rddata_p2_first_out
dfi_probe[80]	rddata_p3_first_out
dfi_probe[88: 81]	min_wl[7:0]
dfi_probe[91: 89]	param_error[2:0]
dfi_probe[97: 92]	suggested_ctrl_delay_min[5:0]
dfi_probe[99: 98]	Reserve
dfi_probe[163:100]	mem_wrdata
dfi_probe[291:164]	mem_rddata
dfi_probe[299:292]	Reserve
dfi_probe[303:300]	Mem read DBI
dfi_probe[304]	Read DBI enabled
dfi_probe[329:305]	Reserve
dfi_probe[330]	MRR Detected
dfi_probe[331]	Reserve

Table 11 DFI Probe Vector (Continued)

Probe Vector	Signal Name
dfi_probe[332]	MRW-1 (Mode register write detected on MR3)
dfi_probe[333]	MRW-2 (Mode register write detected on MR3)
probe[482:334]	Reserve
dfi_probe [488:483]	DFI_T_PHY_WRDATA
dfi_probe [494:489]	DFI_T_PHY_WRLAT
dfi_probe [500:495]	DFI_T_PHY_RDLAT
dfi_probe [506:501]	DFI_T_CTRL_DELAY
dfi_probe[507]	Reserve
dfi_probe[508]	zebuCFG[32]
dfi_probe[509]	zebuCFG[33]
dfi_probe[511:510]	FSP_WR

Feedback

6

Debug

This section provides information on debugging and troubleshooting the following:

- Timing Debug
- Memory Interface
- DFI Interface

Timing Debug

The block that checks the timing parameters in the zDFI interface has the following outputs:

- min_wl
- param_error
- suggested_ctrl_delay_min

The min_wl Value

The min_wl output value is always equal to MEM_WRITE_LATENCY.

As defined in section 3.1, the MEM_WRITE_LATENCY parameter defines the write latency used in the memory. This parameter is important to compute DFI_T_CTRL_DELAY.

The param_error Value

The param_error is a 3-bit vector. Possible error codes are:

- 3'b000: No error
- 3'b001: The zDFI timing condition is not met. Parameters must be recomputed.

- 3'b010 / 3'b011: The zDFI timing condition is not met. DFI_T_CTRL_DELAY is too small.
- 3'b101: DFI_T_CTRL_DELAY is lower than 3.

The suggested_ctrl_delay_min Value

The suggested_ctrl_delay_min is a suggested value for T_CTRL_DELAY in case of error (param_error not equal to 3'b000).

If no parameter error is found, suggested_ctrl_delay_min is 0 and the T_CTRL_DELAY is correct.

If no error is detected and the data is not written/read correctly, then DFI_T_PHY_WRLAT and/or DFI_T_PHY_WRDATA might be erroneous.

Memory Interface

To check the correct behavior of the memory, LPDDR5 memory interface provides a debug vector, as well as an alias file to apply in Verdi.

Also, a part of the probe displays the zrm interface. It is used to understand if the write or the read is failing. None of the mem_* signals on the zDFI should be left unconnected.

For more information, see ZeBu LPDDR5 Memory Models Manual.

DFI Interface

Compared to older revisions, all signals that do not apply to LPDDR5 and the non-supported interfaces have been removed from the zDFI interface. As a result, all signals (idfi_* / odfi_*) of the DFI interface are connected to the controller.

Note:

To identify the commands received by the zDFI, see the Command truth table of LPDDR5 memory, either from the JEDEC specification or from the data sheet provided by memory vendors.

Per this truth table, a valid command is identified by CS high.

Table 12 ROW Command Truth Table

Command	Value on idfi_address_pN
NOP	0x**00 or 0x**80

Table 12 ROW Command Truth Table (Continued)

Command	Value on idfi_address_pN
POWER DOWN ENTRY	0x**40 or 0x**C0
ACTIVATE 1	0x***7 or 0x***F
ACTIVATE 2	0x***3 or 0x***B
PRECHARGE	0x**78 or 0x**F8
REFRESH	0x**38 or 0x**B8
MASK WRITE	0x***2 or 0x***A
WRITE	0x***6 or 0x***E
WRITE32	0x***4
READ	0x***1 or 0x***9
READ32	0x***5 or 0x***D
CAS	0x***C
MUTLI PURPOSE COMMAND	0x**30 or 0x**70 or 0x**B0 or 0x**F0
SELF REFRESH ENTRY	0x**68 or 0x**E8
SELF REFRESH EXIT	0x**28 or 0x**A8
MODE REGISTER WRITE 1	0x**58 or 0x**D8
MODE REGISTER WRITE 2	0x**08 or 0x**48 or 0x**88 or 0x**C8
MODE REGISTER READ	0x**18 or 0x**98
WRITE FIFO	0x**60 or 0x**E0
READ FIFO	0x**20 or 0x**A0
READ DQ CALIBRATION	0x**50 or 0x**D0

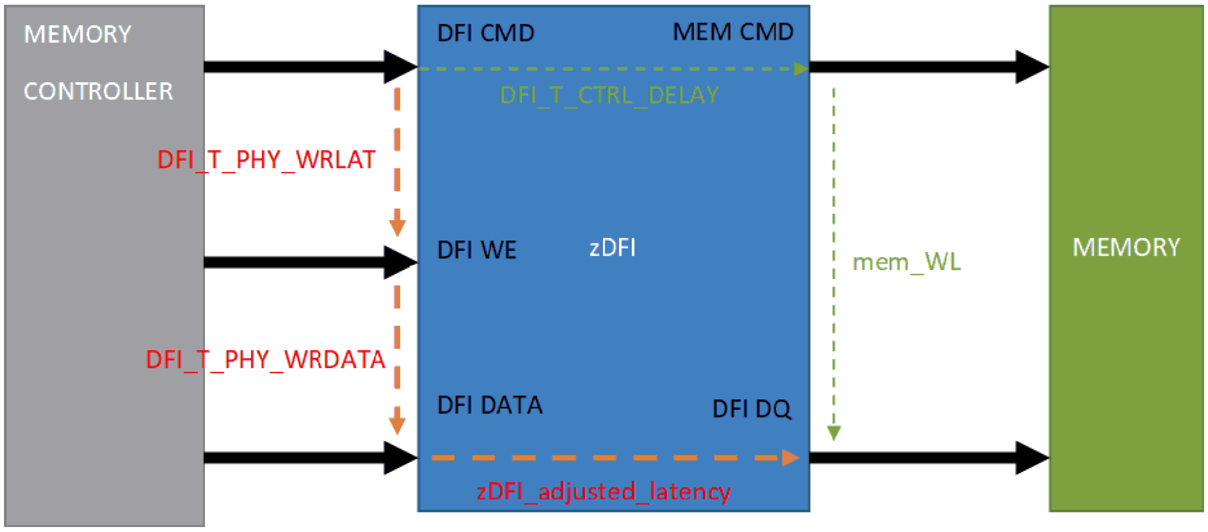
Feedback

7

Appendix: Calculating Timing Constraints

You can compute the time between a DFI write command and the data reaching the memory interface as described in the following figure:

Figure 8 Timing constraints in the zDFI



Method 1 (T1; Green Path)

Per the MIPI DFI spec, the write command reaches the memory interface, DFI_T_CTRL_DELAY, after the command reaches the DFI interface.

In addition, per the LPDDR5 JEDEC specification, the data reaches the memory interface, mem_WL (as set in mode register), after the command.

Therefore, the timing constraint is calculated as:

```
T1 = DFI_T_CTRL_DELAY + mem_WL
```

Note that mem_WL can change during the run or from one run to another.

Method 2 (T2; Orange Path)

By definition in MIPI DFI Spec, the data reaches the DFI interface (DFI_T_PHY_WRLAT + DFI_T_PHY_WRDATA) after the command reached the DFI interface.

In addition, by construction (zDFI), the data uses zDFI_adjusted_latency to go through the zDFI (that is, from the DFI interface to the memory interface).

Therefore, the timing constraint is calculated as:

```
T2 = DFI_T_PHY_WRLAT + DFI_T_PHT_WRDATA + zDFI_adjusted_latency
```

T1=T2 ó DFI_T_CTRL_DELAY + mem_WL = DFI_T_PHY_WRLAT + DFI_T_PHT_WRDATA + zDFI_adjusted_latency

Note:

Mem_WL can change, from run to run, or even during the same run.

The zDFI detects the MEM_WRITE_LATENCY as mem_WL value (when a Mode register write is performed), and automatically adapts the zDFI_adjusted_latency accordingly, in order to comply with T1=T2.

Points to Consider

- If the mem_WL decreases, then the zDFI_adjusted_latency decreases accordingly so that T1 and T2 remain equal. If mem_WL is small enough, the zDFI_adjusted_latency reaches the minimum possible value, called zDFI_LATENCY. For LPDDR5, this value is 4 or 5 depending on the clock ratio. Ensure that the following is true:

  ```
  DFI_T_CTRL_DELAY + MEM_WRITE_LATENCY = DFI_T_PHY_WRLAT +
   DFI_T_PHT_WRDATA + zDFI_LATENCY
  ```

- If MEM_WRITE_LATENCY is too small, modify the value of the DFI_T_CTRL_DELAY parameter to ensure that T1=T2.

Increasing the DFI_T_CTRL_DELAY reduces the performance (the zDFI is holding the command instead of transmitting it to the memory ASAP).

This results in the following equation:

```
DFI_T_CTRL_DELAY + MEM_WRITE_LATENCY ≥ DFI_T_PHY_WRLAT + DFI_T_PHY_WRDATA
 + zDFI_LATENCY, with zDFI_LATENCY=4 or 5
```

- The minimum time between command detection on the zDFI interface and command transmission to the memory interface is used by the zDFI to decode the DFI command. This minimum time is 3 CK. That is, DFI_T_CTRL_DELAY(min)=3
- DFI_T_CTRL_DELAY and MEM_WRITE_LATENCY signify number of memory clock cycles, mem_ck. Also, DFI_T_PHY_WRLAT, DFI_T_PHY_WRDATA and zDFI_LATENCY signify number of PHY clock, which is synchronous with mem_wck. There is a difference in clock ratio for both mem_ck and mem_wck.