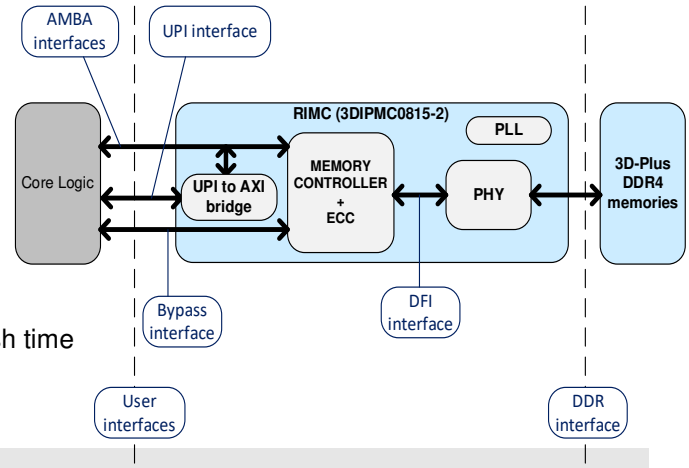
1 FEATURES

- Provides rank and bank management algorithms
- Configurable DDR4 ranks to increase memory capacity
- Selectable Hamming or Reed-Solomon Error Correction Code (ECC)
- Configurable via AMBA user interface - compliant AXI/AHB/APB
- DDR PHY interface compliant DFI 3.1
- Dynamically configurable via an 8-bit APB slave interface
- Controller Bypass mode provides direct access to memories
- Clock & ODT settings compatible with 3D PLUS modules
- Fixed Burst-of-8 accesses
- Selectable Enable/Disable of DDR4 scrubbing
- User-definable scrubbing frequency
- Selectable and/or auto-configurable DRAM refresh time
- Configurable user data width from 8 to 72 bits



2 OVERVIEW

The Radiation Intelligent Memory Controller (RIMC DDR4) is a fully configurable DDR4 SDRAM memory controller IP core designed tocontrol 3D PLUS DDR4 memory modules and to achieve radiation tolerance improvement. The RIMC contains all standard functions of a DDR memory controller for data width applications from 8b up to 72b. It also includesError Correction Code ECC , such as Single Event Upset (SEU) mitigation,Single Event Functional Interrupt (SEFI) protection (described in paragraph 4.2) and weak bit mitigation to be able to work in radiation environments.

In addition to the memory controller IP within the RIMC IP core, the user can either select PHY IPs provided in the IP core or develop its own PHY IP.

The RIMC has two primary interfaces, the user interface (UIF) and the DDR memory interface (DIF).

The UIF is comprised of at least one AHB bus, AXI bus or UPI, and also contains an APB bus for user dynamic configuration:

- Slave APB interface dedicated to internal registers
- Optional slave AHB/AXI interface
- Optional UPI interface (interface compatible with MIG from Xilinx)
- Optional Bypass interface

The Memory Controller has the previously described UIF and the DIF, compliant to DFI 3.1, to send commands and data to the DDR memory components through the DDR PHY (which is uniquely configured to work with specific FPGAs/ASICs).

Table Of Content

TABLES and FIGURES

3 RIMC CONFIGURATION

In order to use the RIMC to control 3D PLUS DDR4 Memory modules: first of all, dedicated RIMC configuration registers must be set up. This paragraph describes how to configure the RIMC through registers and generics.

Memory registers Setting

The Memory Setting configuration data are dependent of the 3D PLUS memory modules characteristics, which can be found within 3D PLUS's memory module Detail Specifications. Below are the items of the RIMC that should be configured.

- The user data bus width (DATA_WIDTH generic) that should be selected by the application,
- The ECC width (ECC_WIDTH generic)that should be selected in accordance with paragraph 4.2.1;
- The number of control signals, ranks, banks and rows that should be configured as per Table 33.

The other configuration data are dependent on the end-user's application.

Clock Frequency

The Clock should be configured in accordance with paragraph 4.1.

Error Management

Error Management should be configured in accordance with paragraph 4.2. Below are the items that should be configured.

- ECC setting (see §4.2.1).
- Scrubbing setting (see §4.2.2).
- Refresh Calibration setting (see §4.2.3).

Interface Setting

Interface Setting should be configured in accordance with paragraph 4.3. Below are the items that should be configured.

- Core logic Interface AXI/AHB/APB (see §4.3.1).
- DFI Interface (see §4.3.3).
- Bypass mode (Direct DDR4 Access, see §4.3.2).
- Uncorrectable errors Interruption (see §4.2.1).

4 MEMORY CONTROLLER

The RIMC IP is responsible of the following standard functions of a Synchronous DRAM Memory Controller::

- Bank management
- Conversion of user AMBA commands to DFI compatible DDR commands.

In addition to the standard functions mentioned above, the RIMC IP also embeds the following additional functions that are detailed in § 4.2:

- Data error management including an ECC and a scrubbing mechanism;
- SEFI protection mechanisms;
- Weak bit mitigation

4.1 CLOCK, RESET AND INITIALIZATION

4.1.1 RIMC CLOCK

The RIMC needs 2 internal clocks. The first clock, SysClk, is dedicated to the Memory Controller as well as the AMBA user interfaces. The second clock, DFIClk, is dedicated to the DFI interface. These 2 clocks should be synchronous with a frequency ratio of 1:1. DFIClk and SysClk are generated from ClkIn by a PLL embedded in the RIMC.

Note 1: For Ultrascale and Ultrascale+ targets, DFIClk and SysClk are phase-aligned and have the same frequency.

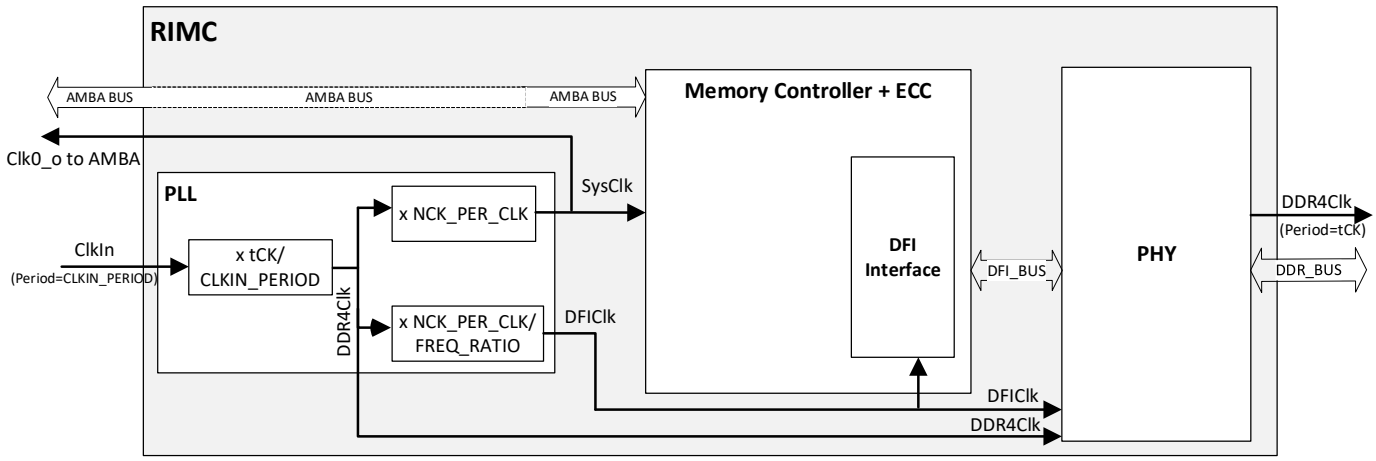


Figure 1: Embedded clock management

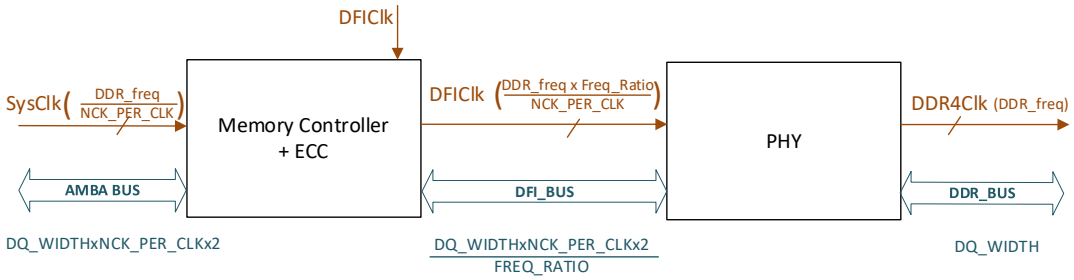


Figure 2: Clocks frequency ratio configuration

FPGA Type	FREQ_RATIO	nCK_PER_CLK
Ultrascale	1	4
Ultrascale+	1	4

Table 1: Frequency ratio constants configuration

4.1.2 RIMC RESET

One asynchronous input reset (Rst_N) must be provided as input to the RIMC

After deassertion of the reset input port (Rst_N), the RIMC automatically configures itself by reading the following configuration input pins:

- SCRUB_EN: This input port can enable or disable the scrubbing mechanism. When activated, the scrubbing performs read and write accesses across the entire memory array after the initialization phase is completed.
- REF_MODE: This input port is used to bypass internal refresh time calibration and select between 8ms, 16ms, 32ms or 64ms refresh time. For further details, please refer to §**4.2.3**.

After this automatic configuration process, the user may dynamically update the internal configuration using the APB slave interface.

4.1.3 RIMC INITIALIZATION

A memory initialization sequence is recommended after powering up the User Memory. This initialization sequence can be enabled through the INIT_Pad APB register. When enabled, the RIMC IP Core initializes the entire memory array with a fixed value as soon as the Physical layer is ready (after the completion of the Power-up sequence and calibration phase). The Byte value to be written on the entire memory array is contained in the Init_Reg register.

If the initialization phase is not performed and a Read is requested by the user to an address prior to a Write at the same address, the RIMC IP Core may detect potential ECC errors.

4.2 ERROR MANAGEMENT

4.2.1 ERROR CORRECTION CODE (ECC)

The RIMC SEU mitigation scheme uses different configurable ECCs (Hamming or Reed-Solomon) and scrubbing to correct SEUs and Single Event Row Errors (SERE). For example, using a Reed-Solomon code for 32b data and 50% overhead [2 * RS(6;4), m=4, Global Bus = 48bits], the RIMC can correct up to 8 bits of error (row error) in one die per 48b, and 2 SEUs in the same address of different dice per 48b. In case where scrubbing is applied, the worst case (in which one particle creates 2 upsets in 2 dice) error rate will be 3.8E-9 upset/day/module.

The RIMC implements an Error Correction Code (ECC) with additional DDR4 components. The supported configurations are described in Table 2.

Two different ECC types are handled by the RIMC:

- Hamming, which is able to mitigate SEU
- Reed-Solomon, which is able to mitigate SEU, MBU and SERE.

There are several types or errors that the RIMC can detect:

- Correctable errors: detected and corrected.
- Uncorrectable errors: detected but not correctable.

Both correctableand uncorrectable errors have their own register to count the cumulated number of errors (the registers are incremented when an error occur even when srubbing is on). The errors are counted on 16 bits. Thus, there are 2 registers of 8 bits for correctable errors (address 0x02 for MSB and 0x03 for LSB) and 2 registers of 8 bits for uncorrectable errors (address 0x00 for MSB and 0x01 for LSB).

Register	Address
ECC_DErr_Reg0	0x00
ECC_DErr_Reg1	0x01
ECC_SErr_Reg0	0x02
ECC_SErr_Reg1	0x03

When an **uncorrectable** error is detected and in addition to the register that will be incremented, an error flag is raised through the AMBA interface.

- For AXI, a **SLVERR** is generated.
- For AHB, a **HRESP_ERROR** is generated.

This flag can be used by the user to take the right action and address the situation.

DDR4 Component Width	Data Bus Width (DATA_WIDTH)	(ECC_WIDTH)	Global Bus Width (DQ_WIDTH)	ECC Type	Correction capability
8	8	0	8	No ECC	none
8	8	8	16	RS(4;2) with m=4	Correct a 4 bits word
8	16	0	16	No ECC	none
8	16	8	24	RS(6;4) with m=4	Correct a 4 bits word
8	16	16	32	2*RS(4;2) with m=4	Correct a 8 bits DDR4 component
8	32	0	0	No ECC	none
8	32	8	40	RS(10;8) with m=4	Correct a 4 bits word
8	32	16	48	2*RS(6;4) with m=4	Correct a 8 bits DDR4 component
8	64	0	64	No ECC	none
16	16	0	16	No ECC	none
16	32	0	32	No ECC	none
16	32	32	64	4*RS(4;2) with m=4	Correct a 16 bits DDR4 component
16	64	8	72	Hamming 64+8	Correct 1 bit in error (Note 2)

Table 2: Supported ECC codes

Note 2: This ECC utlilizes a Hamming code. This code CANNOT correct for a DDR4 component failure but it is able to correct a SEU.

Below is a list of available 3D PLUS DDR4 modules with their configurations:

DDR4 Module P/N	Configuration	Access/Clock	Package
3D4D16G72LB2832	256M x 72	620-1200Mhz	BGA259
3D4D24G48LB2833	512M x 48	620-1200Mhz	BGA259
3D4D32G72LB2805	512M x 72	620-1200Mhz	BGA259
3D4D48G48LB2825	1G x 48	620-1200Mhz	BGA259

Table 3: 3D PLUS DDR4 modules

4.2.2 SCRUBBING FUNCTIONALITY

A scrubbing mechanism can be dynamically enabled either by the SCRUB_EN input pin or by the ScrubEn_Reg register.

When enabled, the RIMC IP core periodically reads the entire memory array and then writes the corrected data back into the memory array. The frequency of the scrubbing is user-defined through the Scrub_Reg register.

An Example of Scrub_Reg configuration:

- The RIMC is connected to a single 3D4D16G72LB2832 (BGA package)
- User wants to scrub the full memory array in 60 seconds (t).
- ddrClk frequency (F_{DDR}) is 620 MHz
- Burst length is set to 8

Scrub_Reg = BURST_LENGTH * t * F_{DFI} / 256M = 562 clock cycles

In this example the RIMC performs one Read/Write Burst every 562 clock cycles.

4.2.3 REFRESH CALIBRATION FUNCTIONALITY

The RIMC provides a refresh calibration functionality. It auto-calibrates the refresh time. Calibrating the refresh time can lower the weak bits ratio.

Note: Weak bits are bits that tend to shift their value to ‘0’ or ‘1’ faster than normal bits because their capacitor is deteriored. A faster refresh rate can lower weak bits consequencies.

A refresh calibration can be launched to automatically calibrate the refresh time. To start the calibration, the START_CAL bit of the Init_Pad_Ctl shall be set to 1. The user also needs to write a value in the Init_Pad_Threshold register in order to define the acceptable number of weak bits accordingly to the user needs and ECC configuration.

The refresh time value can be bypassed with the REF_TIME pin.

- "0XX": No bypass.
- "111": Bypass, 64ms.
- "110": Bypass, 32ms.
- "101": Bypass, 16ms.
- "100": Bypass, 8ms.

Warning: reducing the refresh time value will lower the weak bits ratio but it will also lower the user's bandwidth.

Warning: No accesses shall be performed during refresh calibration. Refresh calibration and Padding shall not be performed simultaneously.

4.2.4 SEFI PROTECTION

The RIMC provides SEFI protection throught Reed-Solomon when 1 component error detection and correction is provided. This is true until the next SEU which would trigger the uncorrectable error flag.

4.3 RIMC INTERFACES

The RIMC is defined by three interfaces (see Figure 3):

- The AMBA interface
- The Direct DDR Access interface
- The DDR PHY interface, compliant to DFI 3.1

These three interfaces are described in the following paragraphs.

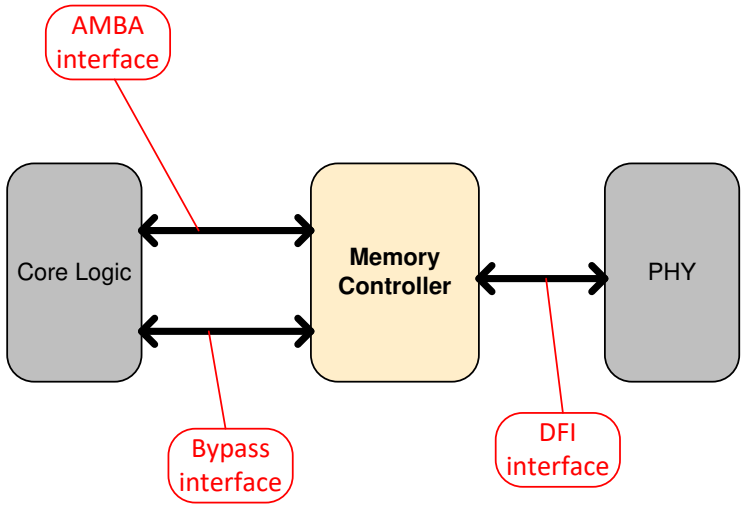


Figure 3: Interfaces with DDR controller

4.3.1 AMBA INTERFACES

Three AMBA interfaces are available for connection to the Memory Controller:

- AMBA AXI: 0 to 8 ports can be instantiated through the NB_AXI_SLV generic.
- AMBA AHB: 0 to 8 ports can be instantiated through the NB_AHB_SLV generic
- AMBA APB: This interface is used to configure the IP.

At least one port shall be instantiated to be synthesizable, i.e. NB_AXI_SLV >= 1 or NB_AHB_SLV >= 1

The Memory Controller can be configured by the core logic using the slave APB interface dedicated to internal registers.

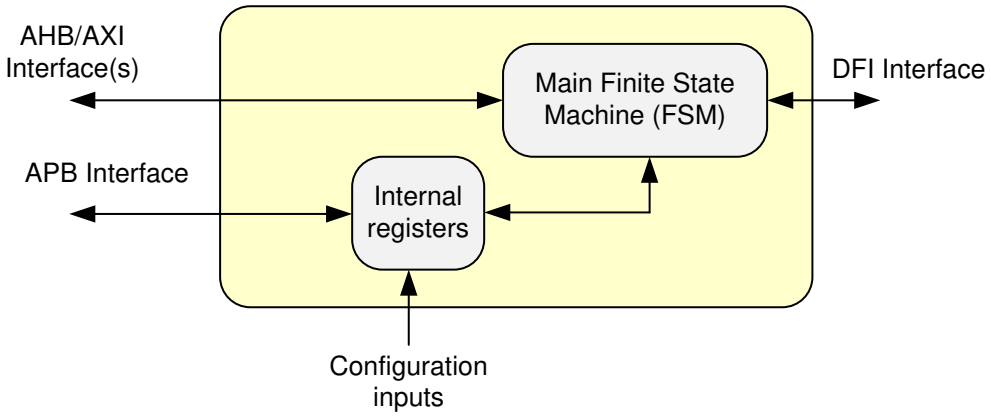


Figure 4: Configuration management

4.3.1.1 AXI interface

The Memory Controller is compliant to the AMBA Advanced eXtensible Interface (AXI) protocol in version 4.0. Up to 8 slaves AXI interfaces can be instantiated inside the RIMC through the NB_AXI_SLV generic.

The data bus width of both the write channel and the read channel can be configured to 16, 32, 64, 128, 256 or 512 bits.

*Note: To realize optimal bandwidth, AXI data width should be greater than or equal to "DDR data bus" width * 2 * nCK_PER_CLK.*

The key features of the AXI protocol are:

- separate address/control and data phases
- support for unaligned data transfers using byte strobes
- burst-based transactions with only start address issued
- separate read and write data channels to enable Direct Memory Access (DMA)
- ability to issue multiple outstanding addresses
- out-of-order transaction completion
- easy addition of register stages to provide timing closure.
- Fastest AMBA interfaces for RIMC

4.3.1.2 AHB interface

The Memory Controller is compliant with AHB slave interface data widths of 16, 32, 64, 128, 256 and 512 bits. RIMC is compatible of AHB version 3.0

*Note: To realize optimal bandwidth, AHB data width should be greater than or equal to "DDR data bus" width * 2 * FREQ_RATIO.*

The AHB slave interface provides the following input signals for each AHB port:

- HSEL: Slave select
- HADDR: Address bus
- HWRITE: Transfer direction (0 for Read / 1 for Write)
- HTRANS: Transfer type
- HSIZE: Transfer size
- HBURST: Burst type
- HWDATA: Write data bus
- HPROT: Protection control. This signal is not used by RIMC IP Core
- HREADY: Transfer done
- HMASTER: Master number. This signal is not used by RIMC IP Core
- HMASTLOCK: Locked sequence

The AHB slave interface provides the following output signals for each AHB port:

- HREADY: Transfer done
- HRESP: Transfer response
- HRDATA: Read data bus
- HSPLIT: Split completion request. The RIMC does not implement the SPLIT functionality so this output is driven LOW.

The slave AHB accepts the following burst types:

- SINGLE
- INCR4
- INCR8
- INCR16

The Memory Controller implements neither the RETRY nor the SPLIT functions. The Memory Controller does not implement the BURST with an unknown number of beats.

An HRESP_ERROR is generated by the RIMC when an uncorrectable error is detected during a Read operation.

4.3.1.3 APB interface

An APB bus is instantiated inside the Memory Controller.

The Memory Controller is compliant with the slave APB interface data width of 8 bits. The Memory Controller is compatible of APB version 2.0 .

The APB interface can be used to configure the Memory Controller internal registers (the complete list of registers is provided in paragraph 4.5).

4.3.2 BYPASS INTERFACE

In addition to the AHB/AXI interfaces the Memory Controller provides a direct path to the DDR4 memory array in a *Controller Bypass* mode.

The following signals are provided at the user interface:

- user_ras_n
- user_cas_n
- user_we_n
- user_address
- user_act_n
- user_ba
- user_bg
- user_cs_n
- user_cke
- user_odt

- user_reset_n
- user_dm
- user_dqo
- user_rddata_en
- user_dqi
- user_dqiv
- user_wdata_en

When using this interface, signals are transferred directly to the DDR4 memory array without any reformatting inside the Memory Controller, therefore it is the user's responsibility to insure that the signals are in conformance with the applicable DDR4 memory datasheets/Detail Specification.

4.3.3 DFI INTERFACE

The Memory Controller implements a DDR PHY Interface (DFI) compatible to DFI 3.1. The DFI is a standardized interface that defines the connectivity between a DDR memory controller and a DDR physical interface (PHY) for DDR1, DDR2, LPDDR2, DDR3, DDR3L, DDR4, DDR4L.

PHY IP is available from 3D PLUS, supporting Ultrascale and Ultrascale+.

4.4 MEMORY CONTROLLER GENERICS/PARAMETERS AND PORTS

The Memory Controller contains the ports defined in

Name	Width	Direction	Functionality	Reset Value
SysClk	1	I	Main input clock used for all the RIMC except DFI interface	-
DFIClk	1	I	Input clock used for DFI interface. Can be SysClk, 2* SysClk or 4* SysClk frequency	-
Rst_N	1	I	Input reset synchronous to SysClk, active LOW	-
DFIRst_N	1	I	Input reset synchronous to DFIClk, active LOW	-
SCRUB_EN	1	I	0: Scrubbing is disabled (or enabled through the ScrubEn_Reg register) 1: Scrubbing is enabled	-
REF_MODE	3	I	0XX: Refresh mode is set according to the RIMC internal register 100: 64ms 101: 32ms 110: 16ms 111: 8ms	-
TESTEN	1	I	0: Normal mode 1: Test mode activate	-

Name	Width	Direction	Functionality	Reset Value
Ahbsi	NB_AHB_SLV records	I	AHB slave input array of record type	-
Ahbso	NB_AHB_SLV records	O	AHB slave output array of record type	-
Apbi	1 record	I	APB slave input	-
Apbo	1 record	O	APB slave output	-
AXI interface				
AXI_aw_out	NB_AXI_SLV records	O	Outputs relative to Address Write Channel	-
AXI_aw_in	NB_AXI_SLV records	I	Inputs relative to Address Write Channel	-
AXI_w_out	NB_AXI_SLV records	O	Outputs relative to Write Data Channel	-
AXI_w_in	NB_AXI_SLV records	I	Inputs relative to Write Data Channel	-
AXI_b_out	NB_AXI_SLV records	O	Outputs relative to Write Response Channel	-
AXI_b_in	NB_AXI_SLV records	I	Inputs relative to Write Response Channel	-
AXI_ar_out	NB_AXI_SLV records	O	Outputs relative to Address Read Channel	-
AXI_ar_in	NB_AXI_SLV records	I	Inputs relative to Address Read Channel	-
AXI_r_out	NB_AXI_SLV records	O	Outputs relative to Read Data Channel	-
AXI_r_in	NB_AXI_SLV records	I	Inputs relative to Read Data Channel	-
DFI interface				
dfi_address	4*ROW_WIDTH	O	DFI address bus	0
dfi_act_n	4	O	DFI activate	‘1111’
dfi_bank	4*BANK_WIDTH	O	DFI bank bus	0
dfi_bg	4*BANK_GROUP_WIDTH	O	DFI bank group bus	0
dfi_cas_n	4	O	DFI column address strobe bus	‘1111’
dfi_cke	4*CS_NUM	O	DFI clock enable bus	‘1’ x Width
dfi_cs_n	4*CS_NUM	O	DFI chip select bus	‘1’ x Widh
dfi_odt	4*CS_NUM	O	DFI on-die-termination control bus	0
dfi_ras_n	4	O	DFI row address strobe bus	‘1111’

Name	Width	Direction	Functionality	Reset Value
dfi_reset_n	4	O	DFI reset bus	‘1111’
dfi_we_n	4	O	DFI write enable bus	‘1111’
dfi_cid	1	O	Not used	‘0’
dfi_wrdata	4*2*DQ_WIDTH	O	DFI Write data bus	0
dfi_wrdata_en	4	O	Write data and data mask enable	0
dfi_wrdata_mask	4*2*DQ_WIDTH/8	O	Write data Byte mask	1 1 1 1 1 1 0 1
dfi_rddata_en	4	O	Read data enable	0
dfi_rddata	4*2*DQ_WIDTH	I	Read data bus	-
dfi_rddata_dnv	4*2*DQ_WIDTH/8	I	DFI data not valid. Not used by RIMC.	-
dfi_rddata_valid	4	I	Read data valid indicator	-
dfi_ctrlupd_req	1	O	MC-inititiated update request	‘0’
dfi_ctrlupd_ack	1	I	MC-initiated update acknowledge	-
dfi_phyupd_req	1	I	PHY-inititiated update request	-
dfi_phyupd_type	2	I	PHY-initiated update select	-
dfi_phyupd_ack	1	O	PHY-initiated update acknowledge	‘0’
dfi_dram_clk_disable	4*CS_NUM	O	DRAM clock disable.	‘0’ x Width
dfi_freq_ratio	2	O	This signal is the image of FREQ_RATIO generic: b00: FREQ_RATIO=1 b01: FREQ_RATIO=2 b10: FREQ_RATIO=4 b11: Reserved	‘00’
dfi_init_complete	1	I	PHY initialization complete	-
dfi_rdlvl_req	1	I	Not used	-
dfi_phy_rdlvl_cs_n	4*CS_NUM	I	Not used	-
dfi_rdlvl_en	1	O	Not used	‘0’
dfi_rdlvl_resp	1	I	Not used	-
dfi_rdlvl_gate_req	1	I	Not used	-
dfi_phy_rdlvl_gate_cs_n	4*CS_NUM	I	Not used	-

Name	Width	Direction	Functionality	Reset Value
dfi_rdlvl_gate_en	1	O	Not used	'0'
dfi_wrlvl_req	1	I	Not used	-
dfi_phy_wrlvl_cs_n	4*CS_NUM	I	Not used	-
dfi_wrlvl_en	1	O	Not used	'0'
dfi_wrlvl_strobe	1	O	Not used	'0'
dfi_wrlvl_resp	1	I	Not used	-
dfi_calvl_req	1	I	Not used	-
dfi_phy_calvl_cs_n	4*CS_NUM	I	Not used	-
dfi_calvl_en	1	O	Not used	'0'
dfi_calvl_capture	1	O	Not used	'0'
dfi_calvl_resp	1	I	Not used	-
dfi_lvl_pattern	4	O	Not used	'0000'
dfi_lvl_periodic	1	O	Not used	'0'
dfi_phylvl_req_cs_n	4*CS_NUM	I	Not used	-
dfi_phylvl_ack_cs_n	4*CS_NUM	O	Not used	'0' x Width
Bypass interface				
user_ras_n	4	I	Row Address Select	-
user_cas_n	4	I	Column Address Select	-
user_we_n	4	I	Write Enable, active LOW	-
user_address	4*ROW_WIDTH	I	Address bus	-
user_act_n	4		Activate	-
user_bank	4*BANK_WIDTH	I	Bank Address bus	-
user_bg	4*BANK_GROUP_WIDTH	I	Bank Group bus	-
user_cs_n	4*CS_NUM	I	Chip Select, active LOW	-
user_cke	4*CS_NUM	I	Clock Enable	-
user_odt	4*CS_NUM	I	On Die Termination	-
user_reset_n	4	I	Memory reset. Used only for DDR4	-
user_wdata_en	4	I	Memory write data enable	-
user_dm	4*2*DQ_WIDTH/8	I	Memory data mask	-
user_dqo	4*2*DQ_WIDTH	I	Memory write data	-
user_rdata_en	4	I	Memory read data Enable	-

Name	Width	Direction	Functionality	Reset Value
user_dqi	4*2*DQ_WIDTH	O	Memory Read Data	0
user_dqiv	1	O	Memory Read data valid	'0'
MEM_CTRL_DEBUG	record	O	Debug signals	-

Table 4: Memory controller pinout

The Memory Controller contains the generics/parameters defined in

Name	Functionality	Possible values
RTT_NOM	Nominal ODT termination value	"OFF", "60", "120", "180", "240", "48", "80", "34"
RTT_WR	Dynamic ODT	"OFF", "120', "240", "80"
OUTPUT_DRV	Memory output drive. For DDR4, "HIGH" is 34 Ohm and "LOW" is 40 Ohm	"HIGH", "LOW"
HDMAX	Data width of AHB and AXI busses	16, 32, 64, 128, 256, 512
GSYNCRST	Type of internal reset	0: Asynchronous reset 1: Synchronous reset
USE_DFF	Type of internal memories	0 : Use normal RAM blocks 1 : Use DFF
ENDIANNESS	Endianness of AHB, APB and AXI busses	0: Little endian 1: Big endian
SIMU	Activated only for simulation. Used to accelerate calibration.	0, 1
NB_AHB_SLV	Number of slave AHB busses.	0 to 8. *Note: NB_AHB_SLV+NB_AXI_SLV shall be between 0 and 8* *If the sum is null, then the UPI interface is used.*
NB_AXI_SLV	Number of slave AXI busses	0 to 8. *Note: NB_AHB_SLV+NB_AXI_SLV shall be between 0 and 8* *If the sum is null, then the UPI interface is used.*
TWO_T_TIME_EN	Mode 2T	0, 1
DFI parameters		

Name	Functionality	Possible values
tphy_wrdata	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** signal	Positive integer
tphy_wrlat	Number of DFI clock cycles between a write command and the **dfi_wrdata_en** signal	Positive integer
trddata_en	Number of DFI clock cycles from the assertion of a read command to the assertion of the **dfi_rddata_en** signal	Positive integer
CLK_PERIOD	SysClk clock period, in picosecond	
DATA_WIDTH	Full data width. Please refer to **Table 2**.	8, 16, 32, 64
ECC_WIDTH	Ful ECC width. Please refer to **Table 2**.	8, 16, 32
DQ_WIDTH	Full memory width, including DATA and ECC DQ_WIDTH shall be DATA_WIDTH+ECC_WIDTH, rounded up to the next multiple of 8	8, 16, 24, 32, 40, 48, 64, 72
Memory generics		
BURST_TYPE	Sequential or interleaved 0: Sequential 1: Interleaved	0, 1
CS_NUM	Number of ranks	1, 2
BANK_WIDTH	Number of banks	2, 3
BANK_GROUP_WIDTH	Number of bank groups	1, 2
ROW_WIDTH	Number of rows	17
COL_WIDTH	Number of columns	10
TWR_PS	Write recovery time, in picosecond	Depending on the DDR4 component datasheet
TWTR_PS	Write to read timing, in ps	Depending on the DDR4 component datasheet
TRTP_PS	Read to Precharge timing, in ps	Depending on the DDR4 component datasheet

Name	Functionality	Possible values
TRFC_PS	Refresh to active or refresh to refresh command interval	Depending on the DDR component datasheet
CAS_LAT	Read CAS latency	Depending on the DDR4 component datasheet
CWL_LAT	Write CAS latency	Depending on the DDR4 component datasheet **Only even values are supported.**

Table 5 : Memory Controller Generic/Parameter list

4.5 MEMORY CONTROLLER REGISTERS

The Memory Controller contains the registers defined in Table 6 . Details are given from table 7 to 31.

Registers	Address
ECC_DErr_Reg0	0x00
ECC_DErr_Reg1	0x01
ECC_SErr_Reg0	0x02
ECC_SErr_Reg1	0x03
ECC_DErr_Clr	0x04
ECC_SErr_Clr	0x05
Bypass_Reg_AD	0x06
Init_Pad_Reg	0x07
Init_Pad_Sts	0x08
Init_Pad_Ctl	0x09
Init_Pad_Threshold_Reg0	0x0A
Init_Pad_Threshold_Reg1	0x0B
Ref_Time_Reg	0x0C
Scrubbing_Reg0	0x0D
Scrubbing_Reg1	0x0E
ScrubEn_Reg	0x0F
Scrub_Ctl	0x10
CorEn_Reg	0x11
CorBeat_Reg	0x12
CorVec_Reg0	0x13
CorVec_Reg1	0x14
CorVec_Reg2	0x15
CorVec_Reg3	0x16
CorVec_Reg4	0x17
CorVec_Reg5	0x18
CorVec_Reg6	0x19
CorVec_Reg7	0x1A
CorVec_Reg8	0x1B
CorVec_Reg9	0x1C
CorVec_Reg10	0x1D
CorVec_Reg11	0x1E
ScrubAd_Sts0	0x1F
ScrubAd_Sts1	0x20
ScrubAd_Sts2	0x21

Registers	Address
ScrubAd_Sts3	0x22
ScrubAd_Sts4	0x23
Version_Sts	0x24
Reset_ZQ_Ctl	0x25
Reset_ZQ_Sts	0x26
Selftest_Ctl	0x27
Selftest_Start_Add_Reg0	0x28
Selftest_Start_Add_Reg1	0x29
Selftest_Start_Add_Reg2	0x2A
Selftest_Start_Add_Reg3	0x2B
Selftest_Start_Add_Reg4	0x2C
Selftest_Stop_Add_Reg0	0x2D
Selftest_Stop_Add_Reg1	0x2E
Selftest_Stop_Add_Reg2	0x2F
Selftest_Stop_Add_Reg3	0x30
Selftest_Stop_Add_Reg4	0x31
Selftest_Sts	0x32
Selftest_Err_Cnt_Sts0	0x33
Selftest_Err_Cnt _Sts1	0x34
Selftest_Err_Cnt _Sts2	0x35
Selftest_Err_Cnt _Sts3	0x36
Selftest_Err_Cnt _Sts4	0x37
Selftest_Err_Add_Sts0	0x40
Selftest_Err_Add_Sts1	0x41
Selftest_Err_Add_Sts2	0x42
Selftest_Err_Add_Sts3	0x43
Selftest_Err_Add_Sts4	0x44
Selftest_Err_Add_Sts5	0x45
Selftest_Err_Add_Sts6	0x46
Selftest_Err_Add_Sts7	0x47
Selftest_Err_Add_Sts8	0x48
Selftest_Err_Add_Sts9	0x49
Selftest_Err_Add_Sts10	0x4A
Selftest_Err_Add_Sts11	0x4B
Selftest_Err_Add_Sts12	0x4C
Selftest_Err_Add_Sts13	0x4D
Selftest_Err_Add_Sts14	0x4E
Selftest_Err_Add_Sts15	0x4F
Selftest_Err_Add_Sts16	0x50
Selftest_Err_Add_Sts17	0x51
Selftest_Err_Add_Sts18	0x52
Selftest_Err_Add_Sts19	0x53

Table 6: RIMC registers list

Bits	Field Name	Comments	R/W	Reset value
7-5		Reserved	R	0
4-0	BITERR	Number of the bit in error-1. 0 for no error. This register is used only for EDAC testability, i.e. when TESTEN input pin is high.	R/W	0

Table 7: EDACConf_Reg register

Bits	Field Name	Comments	R/W	Reset value
15-0	CNT	Uncorrectable ECC counter	R	0

Table 8: ECC_DErr_Reg0 and ECC_DErr_Reg1

Bits	Field Name	Comments	R/W	Reset value
15-0	CNT	Correctable ECC counter	R	0

Table 9: ECC_SErr_Reg0 and ECC_SErr_Reg1

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	R	0
0	CLR	0: No effect 1: Clear ECC_DErr_Reg registers	W	0

Table 10: ECC_DErr_Clr register

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	R	0
0	CLR	0: No effect 1: Clear ECC_SErr_Reg registers	W	0

Table 11: ECC_SErr_Clr register

Bits	Field Name	Comments	R/W	Reset value
7-4	-	Reserved	R	0
0	BYP	0: Normal mode 1: Bypass mode. In this mode the only available user interface is the bypass interface	R/W	0

Table 12: Bypass_Reg_AD register

Bits	Field Name	Comments	R/W	Reset value
7-0	VAL	Byte value used for initialization of the whole memory array.	R/W	0

Table 13: Init_Pad_Reg register

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	R	0
0	STS	0: Padding or Refresh Calibration is in IDLE 1: Padding or Refresh Calibration is in progress	R	0

Table 14: Init_Pad_Sts register

Bits	Field Name	Comments	R/W	Reset value
7-2	-	Reserved	R	0
1	START_CAL	0: No Effect 1: Start the Refresh Calibration	W	0
0	START_PAD	0: No Effect 1: Start the Padding	W	0

Table 15: Init_Pad_Ctl register

Bits	Field Name	Comments	R/W	Reset value
15-0	THRESH	Defines the maximum number or weak bits allowed	R/W	0

Table 16: Init_Pad_Threshold_Reg0 and Init_Pad_Threshold_Reg1 registers

Bits	Field Name	Comments	R/W	Reset value
7-2	-	Reserved	R	0
1-0	STS	Register used to configure the refresh time 11: 64ms 10: 32ms 01: 16ms 00: 8ms	R/W	0

Table 17: Ref_Time_Reg

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	R	0
0	EN	0: Scrubbing disabled 1: Scrubbing Enabled	R/W	0

Table 18: ScrubEn_Reg register

Bits	Field Name	Comments	R/W	Reset value
15-0	FREQ	Register used to select the frequency of the scrubbing. 1 DDR4 Burst performed each FREQ SysClk clock cycles. *Note: This register shall have a value above or equal to 128.*	R/W	256

Table 19: Scrubbing_Reg0 and Scrubbing_Reg1 registers

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	R	0
0	RST	0: No effect 1: Reset ScrubAd_Sts0 to ScrubAd_Sts4 registers	W	0

Table 20: Scrub_Ctl register

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	R	0
0	EN	Enable the corruption of data. Used to test the correct behavior of the ECC mechanism	R/W	0

Table 21: CorEn_Reg register

Bits	Field Name	Comments	R/W	Reset value
95-0	COR_BIT	Bit to corrupt (used only when CorEn_Reg register = 0x01)	R/W	0

Table 22: CorVec_Reg0 to CorVec_Reg11 registers

Bits	Field Name	Comments	R/W	Reset value
7-3	-	Reserved	R	0
2-0	COR_BEAT	Beat to corrupt (used only when CorEn_Reg register = 0x01)	R/W	0

Table 23: CorBeat_Reg register

Bits	Field Name	Comments	R/W	Reset value
39-0	AD	Address of the memory in which the scrubbing detected the 1st error	R	0

Table 24: ScrubAd_Sts0 to ScrubAd_Sts4 registers

Bits	Field Name	Comments	R/W	Reset value
7-0	VERSION	Version of the RTL code	R	(1)

Table 25: Version_Sts register

Bits	Field Name	Comments	R/W	Reset value
7-3	-	Reserved	W	
2	PHY_RST_CTL	1: Start phy reset of DDR4 components	W	
1	ZQ_LG_CTL	1: Start ZQ long calibration	W	
0	ZQ_SH_CTL	1: Start ZQ short calibration	W	

Table 26: Reset_ZQ_Ctl register

Bits	Field Name	Comments	R/W	Reset value
7-3	-	Reserved	R	0
2	PHY_RST_STS	1: indicate when phy reset of DDR4 components is completed	R	0
1	ZQ_LG_STS	1: indicate ZQ long calibration is completed	R	0
0	ZQ_SH_STS	1: indicate ZQ short calibration is completed	R	0

Table 27: Reset_ZQ_Sts register

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	W	
0	ST_SELF_TEST	1: start self test	W	

Table 28: Selftest_Ctl register

Bits	Field Name	Comments	R/W	Reset value
39-0	START_ADD	Address to start the selftest	R/W	0

Table 29: Selftest_Start_Add_Reg0 to Selftest_Start_Add_Reg4

Bits	Field Name	Comments	R/W	Reset value
39-0	STOP_ADD	Address to stop the selftest	R/W	0

Table 30: Selftest_Stop_Add_Reg0 to Selftest_Stop_Add_Reg4

Bits	Field Name	Comments	R/W	Reset value
7-1	-	Reserved	R	0
0	STS_SELF_TEST	1: indicate self test is completed	R	0

Table 31: Selftest_Sts register

Bits	Field Name	Comments	R/W	Reset value
39-0	ERR_CNT_STS	Indicate number of stuckbits detected	R	0

Table 32: Selftest_Err_Cnt_Sts0 to Selftest_Err_Cnt_Sts4 registers

Bits	Field Name	Comments	R/W	Reset value
159-120	ERR_ADD3_STS	Report fourth address in error	R	0
119-80	ERR_ADD2_STS	Report third address in error	R	0
79-40	ERR_ADD1_STS	Report second address in error	R	0
39-0	ERR_ADD0_STS	Report first address in error	R	0

Table 33: Selftest_Err_Add_Sts0 to Selftest_Err_Add_Sts19 registers

5 RIMC (MEMORY CONTROLLER + PHY)

The Physical layer IP (PHY) is dependant on the FPGA/ASIC technology being used. The following PHY are available for use inside the RIMC:

- Ultrascale FPGA **
- Ultrascale+ FPGA **

** Under Space qualification

N.B: Those FPGAs are validated targets running with our IP. For all information regarding FPGA Space qualification status, please contact directly FPGA supplier.

5.1 RIMC INTERFACES

The RIMC provides the following interfaces:
- The AMBA interfaces (already described in paragraph 4.3.1)
- The Bypass interface (already described in paragraph 4.3.2)
- The DDR4 memory interface
- The UPI interface

5.1.1 DDR4 INTERFACE

The RIMC is compliant to the JEDEC DDR4 memory interface specification.

5.1.2 UPI INTERFACE

An UPI interface is present at the user interface of the RIMC to allow a direct replacement of a MIG DDR controller from Xilinx with the RIMC.
A bridge is provided inside of the RIMC to convert the MIG into a standard AXI interface.

5.1 RIMC FOR ULTRASCALE/ULTRASCALE+ GENERICS/PARAMETERS AND PORTS

The RIMC contains the ports defined in Table 34.

The RIMC contains the generics defined in Table 35.

Name	Width	Direction	Functionality	Reset Value
ClkIn	1	I	Input clock used to generate other clock used by the RIMC. Used when SYSCLK_TYPE = “SINGLE_ENDED” or “NO_BUFFER”	-
ClkIn_p	1	I	Input clock used to generate other clock used by the RIMC. Used when SYSCLK_TYPE = “DIFFERENTIAL”	-
ClkIn_n	1	I	Input clock used to generate other clock used by the RIMC. Used when SYSCLK_TYPE = “DIFFERENTIAL”	-
Rst_N	1	I	Input asynchronous reset	-
Clk0_o	1	O	Output clock, to be used for all user interfaces	-
rst0_n_o	1	O	Output reset, synchronous to Clk0_o	‘1’
TESTEN	1	I	When set the signal enable functionality to test the design	-
Mmcm_lock_user	1	I	Mmcm_lock generated by the user when GEN_CLK =0	-
Div_clk_user	1	I	Div_clk generated by the user when GEN_CLK =0	-
Riu_clk_user	1	I	Riu_clk generated by the user when GEN_CLK =0	-
Div_clk_rst_user	1	I	Div_clk_rst generated by the user when GEN_CLK =0	-
Riu_clk_rst_user	1	I	Riu_clk_rst generated by the user when GEN_CLK =0	-
Reset_ub_user	1	I	Reset_ub generated by the user when GEN_CLK =0	-

Name	Width	Direction	Functionality	Reset Value
PllGate_user	1	I	PllGate generated by the user when GEN_CLK =0	-
Sys_clk_in_user	1	O	Sys_clk_in output to the user when GEN_CLK =0	-
Mmcm_clk_in_user	1	O	Mmcm_clk_in output to the user when GEN_CLK =0	-
Pll_lock_user	1	O	Pll_lock output to the user when GEN_CLK =0	-
SCRUB_EN	1	I	0: Scrubbing is disabled (or enabled through the ScrubEn_Reg register) 1: Scrubbing is enabled	-
REF_MODE	3	I	0XX: Refresh mode is set according to the RIMC internal register 100: 64ms 101: 32ms 110: 16ms 111: 8ms	-
Ahbsi	NB_AHB_SLV records	I	AHB slave input array of record type	-
Ahbso	NB_AHB_SLV records	O	AHB slave output array of record type	-
Apbi	1 record	I	APB slave input	-
Apbo	1 record	O	APB slave output	-
AXI interface				
AXI_aw_out	NB_AXI_SLV records	I	Outputs relative to Address Write Channel	-
AXI_aw_in	NB_AXI_SLV records	O	Inputs relative to Address Write Channel	-
AXI_w_out	NB_AXI_SLV records	I	Outputs relative to Write Data Channel	-
AXI_w_in	NB_AXI_SLV records	O	Inputs relative to Write Data Channel	-
AXI_b_out	NB_AXI_SLV records	O	Outputs relative to Write Response Channel	-
AXI_b_in	NB_AXI_SLV records	I	Inputs relative to Write	-

Name	Width	Direction	Functionality	Reset Value
			Response Channel	
AXI_ar_out	NB_AXI_SLV records	I	Outputs relative to Address Read Channel	-
AXI_ar_in	NB_AXI_SLV records	O	Inputs relative to Address Read Channel	-
AXI_r_out	NB_AXI_SLV records	O	Outputs relative to Read Data Channel	-
AXI_r_in	NB_AXI_SLV records	I	Inputs relative to Read Data Channel	-
DDR4 interface				
ddr4_act_n	1	O	DDR activate	'1'
ddr4_ck_t	CLK_WIDTH	O	DDR differential clock	-
ddr4_ck_c	CLK_WIDTH	O	DDR differential clock	-
ddr4_adr	ROW_WIDTH	O	DDR address bus	'1110 … 0'
ddr4_ba	BANK_WIDTH	O	DDR bank bus	0
ddr4_bg	BANK_GROUP_WIDTH	O	DDR bank group bus	0
ddr4_cke	CKE_WIDTH	O	DDR clock enable bus	0
ddr4_cs_n	CS_WIDTH	O	DDR chip select bus	'1' x Width
ddr4_reset_n	1	O	DDR reset bus.	'0'
ddr4_dm_dbi_n	DM_WIDTH	O	DDR data mask	'Z' x Width
ddr4_dq	DQ_WIDTH	IO	DDR data	'Z' x Width
ddr4_dqs_t	DQS_WIDTH	IO	DDR data strobe	'Z' x Width
ddr4_dqs_c	DQS_WIDTH	IO	DDR data strobe	'Z' x Width
ddr4_odt	ODT_WIDTH	O	DDR on die termination	0
UPI Interface				
app_cmd	3	I	b000: Write b001: Read Others: Reserved	-
app_addr	38	I	Gives information about the address of the memory location to be accessed. This bus contains the bank address, the row address and the column address	-
app_en	1	I	Indicate that the command on signal app_cmd and app_addr is valid	-

Name	Width	Direction	Functionality	Reset Value
app_rdy	1	O	Indicate that RIMC accept the command	'1'
app_wdf_data	2*4*DATA_WIDTH	I	User input data	-
app_wdf_mask_data	2*4*DATA_WIDTH /8	I	User mask data	-
app_wdf_wren	1	I	Write data to be written in memory	-
app_wdf_end	1	I	End of access on UPI interface	-
app_wdf_rdy	1	O	Indicate that RIMC accept the data to be written in memory	'1'
app_rd_data_valid	1	O	Status signal indicating read data is valid on the app_rd_data bus	'0'
app_rd_data_end	1	O	Status signal indicating end of read data access	'0'
app_rd_data	2*4*DATA_WIDTH	O	Read data from the memory	0
Bypass interface				
user_ras_n	4	I	Row Address Select	-
user_cas_n	4	I	Column Address Select	-
user_we_n	4	I	Write Enable, active LOW	-
user_address	4*ROW_WIDTH	I	Address bus	-
user_bank	4*BANK_WIDTH	I	Bank Address bus	-
user_bg	4*BANK_GROUP_WIDTH	I	Bank Goup Bus	-
user_cs_n	4*CS_NUM	I	Chip Select, active LOW	-
uer_cke	4*CS_NUM	I	Clock Enable	-
user_odt	4*CS_NUM	I	On Die Termination	-
user_reset_n	4	I	Memory reset. Used only for DDR4	-
user_wdata_en	4	I	Memory write data enable	-
user_dm	4*2*DQ_WIDTH/8	I	Memory data mask	-
user_dqo	4*2*DQ_WIDTH	I	Memory write data	-
user_rdata_en	4	I	Memory read data Enable	-
user_dqi	4*2*DQ_WIDTH	O	Memory Read Data	0

Name	Width	Direction	Functionality	Reset Value
user_dqiv	1	O	Memory Read data valid	'0'
rimc_dbg	record	O	Debug signals	-

Table 34: RIMC Ultrascale/Ultrascale+ Core Ports

Name	Functionality	Possible values
GEN_UPI	Define if the UPI interface can be used by the user	0, 1
GEN_CLK	Define if clock are generated inside the RIMC or provided by the user	0, 1
SYSCLK_TYPE	Define if a clock buffer shall be added to the SysClk input clock	"SINGLE_ENDED", "NO_BUFFER", "DIFFERENTIAL"
BANK_TYPE	Used only for Xilinx 7-series PHY.	"HP_IO", "HR_IO"
HDMAX	Data width of AHB and AXI busses	16, 32, 64, 128, 256, 512
GSYNCRST	Type of internal reset	0: Asynchronous reset 1: Synchronous reset
USE_DFF	0 : Internal flip-flops 1 : Embedded RAM	Type of internal memories used for the FIFO functions
ENDIANNESS	Endianness of AHB, APB and AXI busses	0: Little endian 1: Big endian
NB_AHB_SLV	Number of slave AHB busses.	0 to 8. *Note: NB_AHB_SLV+NB_AXI_SLV shall be between 0 and 8* *If the sum is null, then the UPI interface is used.*
NB_AXI_SLV	Number of slave AXI busses	0 to 8. *Note: NB_AHB_SLV+NB_AXI_SLV shall be between 0 and 8* *If the sum is null, then the UPI interface is used.*
SIMU	Activated only for simulation. Used to accelerate calibration.	0, 1
phy_config	Physical layer configuration	Generated with scripts
RTT_NOM	Nominal ODT termination value	"OFF", "60", "120", "180", "240", "48", "80", "34"
RTT_WR		"OFF", "120", "240", "80"

Name	Functionality	Possible values
OUTPUT_DRV	Memory output drive. For DDR4, “HIGH” is 34 Ohm and “LOW” is 40 Ohm	“HIGH”, “LOW”
CLKIN_PERIOD	SysClk clock period, in picosecond	
tCK	DDR4_ck clock period, in picosecond	
TWO_T_TIME_EN	Mode 2T	0, 1
DM_WIDTH	Data mask width	1, 2, 3, 4, 5, 6, 8, 9
DQS_WIDTH	Data Strobe width	1, 2, 3, 4, 5, 6, 8, 9
DATA_WIDTH	Full data width.	8, 16, 32, 64
ECC_WIDTH	Ful ECC width.	5, 6, 8, 16
DQ_WIDTH	Full memory width, including DATA and ECC DQ_WIDTH shall be DATA_WIDTH+ECC_WIDTH, rounded up to the next multiple of 8	8, 16, 24, 32, 40, 48, 64, 72
Memory generics		
CS_WIDTH	Width of memory chip select bus	1 to 4
CKE_WIDTH	Width of memory clock enable bus	1 to 12
ODT_WIDTH	Width of memory ODT bus	1 to 12
CS_NUM	Number of ranks	1, 2
ODT_NUM	Number of rank for ODT	1, 2
BANK_WIDTH	Number of banks	2, 3
BANK_GROUP_WIDTH	Number of bank groups	1
ROW_WIDTH	Number of rows	17
COL_WIDTH	Number of columns	10
CLK_WIDTH	Number of clock outputs	1, 2, 3, 4, 5, 6, 7, 8, 9, 12
TWR_PS	Write recovery time, in picosecond	Depending on the DDR component datasheet
TWTR_PS	Write to read timing, in ps	Depending on the DDR component datasheet

Name	Functionality	Possible values
TRTP_PS	Read to Precharge timing, in ps	Depending on the DDR component datasheet
TRFC_PS	Refresh to active or refresh to refresh command interval	Depending on the DDR component datasheet
CAS_LAT	Read CAS latency	According to the DDR4 clock frequency
CWL_LAT	Write CAS latency	According to the DDR4 clock frequency **Only even values are supported.**

Table 35: RIMC Ultrascale/Ultrascale+ generics list

6 RIMC PERFORMANCES

6.1 MAXIMUM FREQUENCY

	Memory Controller	Physical Layer
Xilinx Ultrascale	300 MHz	1200 MHz
Xilinx Ultrascale+	300 MHz	1200 MHz

Table 36 : Frequency result of RIMC

6.2 IP SIZE

The following sizes are given for the complete RIMC IP composed of both the Memory Controller and the PHY for a given FPGA target.

Table 37 gives the estimated sizes for different FPGA platforms based on the following Hypothesis :

- DDR4 48 bits (32 data + 16 ECC)
- 2 user AMBA interfaces

The precise IP Size depends on the configuration.

FPGA	Slice LUTs	Slice FFs	Slice BRAMs
Xilinx Ultrascale	14 521 / 355 200 (4.1%)	17 800 / 484 800 (3.7%)	58 / 600 (9.7%)
Xilinx Ultrascale+	9 944 / 316 800 (3.1%)	14 463 / 433 920 (3.3%)	35 / 480 (7.3%)

Table 37: Xilinx resource usage

7 DELIVERABLES

The RIMC IP is delivered with the following directory structure:

- 3DIPMC0815-2:
  - Tech_independant: This directory contains all files for the Memory Controller IP.
    - source: This directory contains all of the VHDL file.
      - Memory_ctrl: This directory contains all of the VHDL files of the Memory Controller core
  - Tech_dependant: This directory contains all source files specific to a technology.
    - Ultrascale This directory contains all files specific to the Ultrascale
      - Source contain all hdl source files
        - rimc: This directory contains all of the VHDL files of the Ultrascale RIMC
        - ddr4_phy: This directory contains all of the VHDL files Ultrascale physical layer
    - UltrascalePlus This directory contains all files specific to the Ultrascale+
      - Source contain all hdl source files
        - rimc: This directory contains all of the VHDL files of the Ultrascale+ RIMC
        - ddr4_phy: This directory contains all of the VHDL files Ultrascale+ physical layer
  - doc:
    - Contains the datasheet of the RIMC DDR4
    - Contains the RIMC DDR4 User Guide

8 TARGET COMPATIBILITY

The DDR4 memory controller IP core is delivered as high level VHDL source code. The memory controller is technology independent HDL code and can be implemented on any targeted technology. The IP provided physical layer and connection between memory controller and physical layer for following targets: Ultrascale and Ultrascale+.

9 PART NUMBER / ORDER INFORMATION

3DIPMC0815-2

Main Sales Office

FRANCE	3D PLUS 408, rue Hélène Boucher ZI. 78532 BUC Cedex	Tel : 33 (0)1 30 83 26 50	Web : www.3d-plus.com e-mail : sales@3d-plus.com	**DISTRIBUTOR**
USA	3D PLUS USA, Inc 151 Callan Avenue, Suite 310 San Leandro, CA 94577	Tel : (510) 824-5591	e-mail : sales@3d-plus.com	

10 REVISION HISTORY

Rev. 2, April 2022

- Revision2: Datasheet release Ultrascale and Ultrascale+

Rev. 1, October 2021

- Initial Datasheet release Ultrascale