PLB Double Data Rate (DDR) Synchronous DRAM (SDRAM) Controller

DS425 December 16, 2005

Product Specification

Introduction

The Xilinx Processor Local Bus Double Data Rate (PLB DDR) Synchronous DRAM (SDRAM) controller for Xilinx FPGAs provides a DDR SDRAM controller which connects to the PLB and provides the control interface for DDR SDRAMs. It is assumed that the reader is familiar with DDR SDRAMs and the IBM PowerPC.

Features

The Xilinx PLB DDR SDRAM Controller is a soft IP core designed for Xilinx FPGAs and contains the following features:

- PLB interface
- Performs device initialization sequence upon power-up and reset conditions for ~200uS. Provides a parameter to adjust this time for simulation purposes only
- Performs auto-refresh cycles
- Supports single-beat and burst transactions
- Supports target-word first cache-line transactions
- Supports CAS latencies of 2 or 3 set by a design parameter
- Supports 32 and 64 DDR data widths set by a design parameter
- Supports indeterminate burst length
- Provides big-endian connections to memory devices. See Connecting to Memory for details on memory connections
- Supports an error correction code (ECC) for 32 bit DDR data width only, set by design parameters
- Supports multiple (up to 4) DDR memory banks

LogiCORE™ Facts		
Core Specifics		
Supported Device Family	Virtex-II Pro™, Virtex-4™	
Version of Core	plb_ddr	v1.11a
Resources Used. See Table 24		
	Min	Max
Slices	712	1970
LUTs	507	2261
FFs	651	1841
Block RAMs		
Provided with Core		
Documentation	Product Specification	
Design File Formats	VHDL	
Constraints File	N/A	
Verification	N/A	
Instantiation Template	N/A	
Reference Designs	None	
Design Tool Requirements		
Xilinx Implementation Tools	ISE 6.1i or later	
Verification	N/A	
Simulation	ModelSim SE/EE 5.8 or later	
Synthesis	XST	
Support		
Support provided by Xilinx, Inc.		

DDR SDRAM Controller Design Parameters

To allow the user to obtain a DDR SDRAM Controller that is uniquely tailored for their system, certain features are parameterizable in the DDR SDRAM Controller design. This allows the user to have a design that only utilizes the resources required by their system and runs at the best possible performance. The features that are parameterizable in the DDR SDRAM Controller are shown in Table 1

Table 1: **DDR SDRAM Controller Design Parameters**

Grouping / Number		Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
DDR SDRAM Controller Features	G1	Include logic to support PLB bursts and cacheline transactions	C_INCLUDE_BURST_CACHELN_SUPPORT	0 = don't include logic to support PLB bursts and cacheline transfers 1 = include logic to support PLB bursts and cacheline transfers	0	integer
	G2	Include support for Registered DIMM	C_REG_DIMM	0 = DDR device is not registered DIMM 1 = DDR device is registered DIMM	0	integer
	G3	Supported number of DDR SDRAM memory banks(1)	C_NUM_BANKS_MEM	1 - 4	1	integer
	G4	Number of generated DDR clock pairs	C_NUM_CLK_PAIRS	1 - 4	1	integer
	G5	Target FPGA family	C_FAMILY	virtex2, virtex2p, virtex4	virtex2p	string
ECC	G6	Include support for ECC logic(2)	C_INCLUDE_ECC_SUPPORT	0=don't include support for ECC 1=support ECC	0	integer
	G7	Enable use of ECC registers including: ECCCR, ECCSR, ECCSEC, ECCDEC, & ECCPEC(3)	C_ENABLE_ECC_REG	0=disables all ECC and IPIF interrupt registers (all register default conditions are implemented) 1=enable all ECC registers	1	integer
	G8	ECC default on or off condition (controls reset condition of ECCCR[30:31]	C_ECC_DEFAULT_ON	0=ECC default OFF (must write to ECCCR to enable ECC) 1=ECC default ON	1	integer

Table 1: **DDR SDRAM Controller Design Parameters *(Continued)***

Grouping / Number		Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
ECC (con't)	G9	Support interrupts in IPIF for ECC error conditions(3,4) (Enables use of IPIF interrupt registers)	C_INCLUDE_ECC_INTR	0=no ECC error interrupt support (no IPIF interrupt registers available) 1=ECC error interrupts supported (IPIF interrrupt registers available)	0	integer
	G10	Support ECC force error testing	C_INCLUDE_ECC_TEST	0=no ECC test support 1=enable ECC test support	0	integer
	G11	Specifies single bit data error interrupt threshold counter value	C_ECC_SEC_THRESHOLD	1 - 4095(5)	1	integer
	G13	Specifies double bit error interrupt threshold counter value	C_ECC_DEC_THRESHOLD	1 - 4095(5)	1	integer
	G14	Specifies parity field bit error interrupt threshold counter value	C_ECC_PEC_THRESHOLD	1 - 4095(5)	1	integer
DDR SDRAM Device Features	G15	Load Mode Register command cycle time (ps)	C_DDR_TMRD		15000	integer
	G16	Write Recovery Time (ps)	C_DDR_TWR		15000	integer
	G17	Write-to-Read Command Delay (Tck)	C_DDR_TWTR		1	integer
	G18	Delay after ACTIVE command before PRECHARGE command (ps)	C_DDR_TRAS		40000	integer
	G19	Delay after ACTIVE command before another ACTIVE or AUTOREFRESH command (ps)	C_DDR_TRC		65000	integer
	G20	Delay after AUTOREFRESH before another command (ps)	C_DDR_TRFC		75000	integer

Table 1: **DDR SDRAM Controller Design Parameters *(Continued)***

Grouping / Number		Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
DDR SDRAM Device Features (con't)	G21	Delay after ACTIVE command before READ/WRITE command (ps)	C_DDR_TRCD		20000	integer
	G22	Delay after ACTIVE command for a row before an ACTIVE command for another row (ps)	C_DDR_TRRD		15000	integer
	G23	Delay after a PRECHARGE command (ps)	C_DDR_TRP		20000	integer
	G24	Average periodic refresh command interval (ps)	C_DDR_TREFI		7800000	integer
	G25	Refresh command interval (ns)	C_DDR_TREFC		70000	
	G26	CAS Latency	C_DDR_CAS_LAT	2,3	2	integer
	G27	Cumulative data width of DDR devices not including ECC check bits(2)	C_DDR_DWIDTH	32, 64	32	integer
	G28	DDR address width	C_DDR_AWIDTH	See note(6)	13	integer
	G29	DDR column address width	C_DDR_COL_AWIDTH	See note(6)	9	integer
	G30	DDR bank address width	C_DDR_BANK_AWIDTH	See note(6)	2	integer
Address Space	G31	Base Address for Memory Bank x (x = 0 to 3)	C_MEMx_BASEADDR	Valid address(7,8)		std_logic_vector
	G32	High Address for Memory Bank x (x = 0 to 3)	C_MEMx_HIGHADDR	Valid address(7,8)		std_logic_vector
	G33	ECC register base address	C_ECC_BASEADDR	Valid address(9)		std_logic_vector
	G34	ECC register high address	C_ECC_HIGHADDR	Valid address(9)		std_logic_vector
PLB Interface	G35	PLB Data bus width	C_PLB_DWIDTH	64	64	integer
	G36	PLB Address bus width	C_PLB_AWIDTH	32	32	integer

Table 1: **DDR SDRAM Controller Design Parameters *(Continued)***

Grouping / Number		Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
	G37	Number of PLB masters	C_PLB_NUM_MASTERS	1 - 16	4	integer
	G38	PLB clock period (ps)	C_PLB_CLK_PERIOD_PS			integer
Simulation Only	G39	DDR Initialization time for simulation(10)	C_SIM_INIT_TIME_PS	Minimum 200 clock periods	2000000000	integer
Auto-calculated parameters(11)	G40	Number of bits required to encode the number of PLB Masters	C_PLB_MID_WIDTH	1 - log2(C_NUM_MASTERS)	2	integer

Notes:
1. C_NUM_BANKS_MEM specifies the number of DDR SDRAM memory banks with identical device characteristics. All the DDR SDRAM device characteristics specified in parameters, G13 through G28, are applicable for all memory banks. The C_NUM_BANKS_MEM parameter specifies the size of the DDR_CSn signal(s).
2. ECC support only applies for 32-bit data width interface of DDR SDRAM.
3. Setting C_ENABLE_ECC_REG=1 will disable all ECC and IPIF interrupt registers. When C_ENABLE_ECC_REG=1, it will override any setting of the C_INCLUDE_ECC_INTR parameter to include interrupt logic.
4. Enabling interrupt support for ECC error conditions (setting C_INCLUDE_ECC_INTR=1) will instantiate the IPIF IP ISC. The IPIF Device ISC will not be instantiated.
5. Threshold values limited by size of error counter registers. Current implementation is12-bit counter registers.
6. C_DDR_AWIDTH + C_DDR_COL_AWIDTH + C_DDR_BANK_AWIDTH + log2(C_DDR_DWIDTH/8) must be < C_PLB_AWIDTH-1.
7. This design can accommodate up to 4 banks of DDR memory. The address range generics are designated as C_MEM0_BASEADDR, C_MEM1_BASEADDR, C_MEM0_HIGHADDR, C_MEM1_HIGHADDR, etc.
8. The range specified by C_MEMx_BASEADDR and C_MEMx_HIGHADDR must comprise a complete, contiguous power of two range such that range = 2^m, and the m least significant bits of C_MEMx_BASEADDR must be zero.
9. The range specified by C_ECC_BASEADDR and C_ECC_HIGHADDR must comprise a complete, contiguous power of two range such that range = 2^n, and the n least significant bits of C_ECC_BASEADDR must be zero. C_BASEADDR must be a multiple of the range, where the range is C_HIGHADDR - C_BASEADDR +1.
10. This parameter adjusts the initialization time of the DDR for simulation only. Must be > 200 clocks
11. These design parameters are automatically calculated by the system generation tool and are not input by user

Allowable Parameter Combinations

The DDR SDRAM Controller supports up to 4 banks of memory. Each bank of memory has its own independent base address and address range. Each address range specified by C_MEMx_BASEADDR and C_MEMx_HIGHADDR must comprise a complete, contiguous power of two range such that range = 2^m, and the m least significant bits of C_MEMx_BASEADDR must be zero. The range specified by these parameters should not exceed the DDR SDRAM memory space.

The address range specified by C_ECC_BASEADDR and C_ECC_HIGHADDR, must comprise a complete, contiguous power of two range such that range = 2^n, and the n least significant bits of C_ECC_BASEADDR must be zero (for 32-bit data interface if ECC is required). The range specified by these parameters should not exceed the DDR SDRAM memory space.

The combination of generics C_INCLUDE_ECC_SUPPORT=1 and C_DDR_DWIDTH=64 is not allowed as this IP Core is not designed to support ECC for 64-bit Data width interface with DDR SDRAM.

DDR SDRAM Controller I/O Signals

Table 2 provides a summary of all DDR SDRAM Controller input/output (I/O) signals, the interfaces under which they are grouped, and a brief description of the signals.

Table 2: **DDR SDRAM Controller Pin Descriptions**

Grouping		Signal Name	Interface	I/O	Initial State	Description
DDR SDRAM Signals	P1	DDR_Clk [0:C_NUM_CLK_PAIRS-1]	DDR	O	0	DDR Clock(1)
	P2	DDR_Clkn [0:C_NUM_CLK_PAIRS-1]	DDR	O	1	DDR inverted clock
	P3	DDR_CKE [0:C_NUM_BANKS_MEM-1]	DDR	O	0	DDR Clock Enable
	P4	DDR_CSn [0:C_NUM_BANKS_MEM-1]	DDR	O	1	Active low DDR chip select(s)
	P5	DDR_RASn	DDR	O	1	Active low DDR row address strobe
	P6	DDR_CASn	DDR	O	1	Active low DDR column address strobe
	P7	DDR_WEn	DDR	O	1	Active low DDR write enable
	P8	DDR_DM[0:C_DDR_DWIDTH/8-1]	DDR	O	0	DDR data mask
	P9	DDR_BankAddr[0:C_DDR_BANK_AWIDTH-1]	DDR	O	0	DDR bank address
	P10	DDR_Addr[0:C_DDR_AWIDTH-1]	DDR	O	0	DDR address
	P11	DDR_DQ_o[0:C_DDR_DWIDTH-1]	DDR	O	0	Output data to DDR
	P12	DDR_DQ_i[0:C_DDR_DWIDTH-1]	DDR	I		Input data from DDR
	P13	DDR_DQ_t[0:C_DDR_DWIDTH-1]	DDR	O	0	3-state control for DDR data buffers
	P14	DDR_DQS_o[0:C_DDR_DWIDTH/8-1]	DDR	O	0	Output data strobe to DDR
	P15	DDR_DQS_i[0:C_DDR_DWIDTH/8-1]	DDR	I		Input data strobe from DDR
	P16	DDR_DQS_t[0:C_DDR_DWIDTH/8-1]	DDR	O	1	3-state control for DDR data strobe buffers
	P17	DDR_DM_ECC	DDR	O	0	DDR ECC data mask
	P18	DDR_DQ_ECC_o[0:NUM_ECC_BITS-1]	DDR	O	0	Output ECC data to DDR
	P19	DDR_DQ_ECC_i[0:NUM_ECC_BITS-1]	DDR	I		Input ECC data from DDR
	P20	DDR_DQ_ECC_t[0:NUM_ECC_BITS-1]	DDR	O	0	3-state control for DDR ECC data buffer
	P21	DDR_DQS_ECC_o	DDR	O	0	Output ECC data strobe to DDR
	P22	DDR_DQS_ECC_i	DDR	I		Input ECC data strobe from DDR
	P23	DDR_DQS_ECC_t	DDR	O	1	3-state control for DDR ECC data strobe buffers
	P24	DDR_Init_done	DDR	O	0	Signals that the DDR initialization is complete

Table 2: **DDR SDRAM Controller Pin Descriptions *(Continued)***

Grouping		Signal Name	Interface	I/O	Initial State	Description
Clock Signals	P25	Clk90_in		I		System bus clock phase shifted by 90 degrees
	P26	Clk90_in_n		I		System bus clock phase shifted by 270 degrees
	P27	DDR_Clk90_in		I		DDR clock feedback shifted by 90 degrees
	P28	DDR_Clk90_in_n		I		DDR clock feedback shifted by 270 degrees
PLB Slave Signals(2)	P29	PLB_PAValid	PLB	I		PLB primary address valid indicator
	P30	PLB_buslock	PLB	I		PLB bus lock
	P31	PLB_masterID[0:C_PLB_NUM_MASTERS-1]	PLB	I		PLB current master indicator
	P32	PLB_RNW	PLB	I		PLB read not write
	P33	PLB_BE[0:C_PLB_DWIDTH/8-1]	PLB	I		PLB byte enables
	P34	PLB_size[0:3]	PLB	I		PLB transfer size
	P35	PLB_type[0:2]	PLB	I		PLB transfer type
	P36	PLB_MSize[0:1]	PLB	I		PLB master data bus size
	P37	PLB_compress	PLB	I		PLB compressed data transfer indicator
	P38	PLB_guarded	PLB	I		PLB guarded transfer indicator
	P39	PLB_ordered	PLB	I		PLB synchronize transfer indicator
	P40	PLB_lockErr	PLB	I		PLB lock error indicator
	P41	PLB_abort	PLB	I		PLB abort bus request indicator
	P42	PLB_ABus[0:C_PLB_AWIDTH-1]	PLB	I		PLB address bus
	P43	PLB_SAValid	PLB	I		PLB secondary address valid indicator
	P44	PLB_rdPrim	PLB	I		PLB secondary to primary read request indicator
	P45	PLB_wrPrim	PLB	I		PLB secondary to primary write request indicator
	P46	PLB_wrDBus[0:C_PLB_DWIDTH-1]	PLB	I		PLB write data bus
	P47	PLB_wrBurst	PLB	I		PLB burst write transfer indicator
	P48	PLB_rdBurst	PLB	I		PLB burst read transfer indicator

Table 2: **DDR SDRAM Controller Pin Descriptions *(Continued)***

Grouping		Signal Name	Interface	I/O	Initial State	Description
PLB Slave Signals[2] (con't)	P49	Sl_addrAck	PLB	O	0	Slave address acknowledge
	P50	Sl_wait	PLB	O	0	Slave wait indicator
	P51	Sl_SSize[0:1]	PLB	O	0	Slave data bus size
	P52	Sl_rearbitrate	PLB	O	0	Slave rearbitrate bus indicator
	P53	Sl_MBusy[0:C_PLB_NUM_MASTERS-1]	PLB	O	0	Slave busy indicator
	P54	Sl_MErr[0:C_PLB_NUM_MASTERS-1]	PLB	O	0	Slave error indicator
	P55	Sl_wrDAck	PLB	O	0	Slave write data acknowledge
	P56	Sl_wrComp	PLB	O	0	Slave write transfer complete indicator
	P57	Sl_wrBTerm	PLB	O	0	Slave terminate write burst transfer
	P58	Sl_rdDBus[0:C_PLB_DWIDTH-1]	PLB	O	0	Slave read bus
	P59	Sl_rdWdAddr[0:3]	PLB	O	0	Slave read word address
	P60	Sl_rdDAck	PLB	O	0	Slave read data acknowledge
	P61	Sl_rdComp	PLB	O	0	Slave read transfer complete indicator
	P62	Sl_rdBTerm	PLB	O	0	Slave terminate read burst transfer
	P63	PLB_Clk	PLB	I		PLB clock
	P64	PLB_Clk_n	PLB	I		PLB clock phase shifted by 180 degrees
	P65	PLB_Rst	PLB	I		PLB reset
System	P66	IP2INTC_Irpt	System Interrupt Controller	O		Device interrupt output to microprocessor interrupt input or system interrupt controller.

Notes:
1. DDR_Clk is the same frequency as PLB_Clk.
2. Please refer to the IBM PLBArchitecture Specification for more detailed information on these signals.

Parameter-Port Dependencies

The dependencies between the DDR SDRAM controller design parameters and I/O signals are shown in Table 3. It gives the information about how the ports and parameters get affected by changing certain parameters.

Table 3 shows when certain features are parameterized away, the related logic will be part of design and signals are unconnected and the related output signals are set to a constant value.

Table 3: **Parameter-Port Dependencies**

Grouping		Parameter	Affects	Depends	Description
Design Parameters	G3	C_NUM_BANKS_MEM	P4		Specifies the number of DDR SDRAM Memory Banks
	G4	C_NUM_CLK_PAIRS	P1, P2		Number of generated DDR clock pairs
	G6	C_INCLUDE_ECC_SUPPORT	P17 - P23	G27	Input signals are unconnected and output signals are set to a constant value.
	G7	C_ENABLE_ECC_REG		G6	Unused when C_INCLUDE_ECC_SUPPORT=0.
	G8	C_ECC_DEFAULT_ON		G6	Unused when C_INCLUDE_ECC_SUPPORT= 0.
	G9	C_INCLUDE_ECC_INTR	P66	G6, G7	Unused when C_INCLUDE_ECC_SUPPORT= 0. Unused when C_ENABLE_ECC_REG= 0.
	G10	C_INCLUDE_ECC_TEST		G6 G7	Unused when C_INCLUDE_ECC_SUPPORT= 0. Unused when C_ENABLE_ECC_REG= 0.
	G11	C_ECC_SEC_THRESHOLD		G6, G7	Unused when C_INCLUDE_ECC_SUPPORT= 0. Unused when C_ENABLE_ECC_REG= 0.
	G12	C_ECC_DEC_THRESHOLD		G6, G7	Unused when C_INCLUDE_ECC_SUPPORT= 0. Unused when C_ENABLE_ECC_REG= 0.
	G13	C_ECC_PEC_THRESHOLD		G6, G7	Unused when C_INCLUDE_ECC_SUPPORT= 0. Unused when C_ENABLE_ECC_REG= 0.
	G33	C_ECC_BASEADDR		G6, G7	Unused when C_INCLUDE_ECC_SUPPORT=0. Unused when C_ENABLE_ECC_REG = 0.
	G34	C_ECC_HIGHADDR		G6, G7	Unused when C_INCLUDE_ECC_SUPPORT=0. Unused when C_ENABLE_ECC_REG = 0.
	G27	C_DDR_DWIDTH	P17, P18, P19, P20, P21, P22, P23, P63	G6	C_DDR_DWIDTH can only be set to 32 when C_INCLUDE_ECC_SUPPORT=1
I/O Signals		DDR_Clk [0:C_NUM_CLK_PAIRS-1]		G4	
		DDR_Clkn [0:C_NUM_CLK_PAIRS-1]		G4	

Table 3: **Parameter-Port Dependencies *(Continued)***

Grouping		Parameter	Affects	Depends	Description
I/O Signals (con't)		DDR_CKE [0:C_NUM_BANKS_MEM-1]		G3	
	P4	DDR_CSn [0:C_NUM_BANKS_MEM-1]		G3	Active low DDR SDRAM Chip Selects
	P17	DDR_DM_ECC		G6, G27	Output driven high when C_INCLUDE_ECC_SUPPORT=0 or C_DDR_DWIDTH=64
	P18	DDR_DQ_ECC_o		G6, G27	Output is grounded when C_INCLUDE_ECC_SUPPORT=0 or C_DDR_DWIDTH=64 and should not be connected.
	P19	DDR_DQ_ECC_i		G6, G27	Input is unused when C_INCLUDE_ECC_SUPPORT=0 or C_DDR_DWIDTH=64.
	P20	DDR_DQ_ECC_t		G6, G27	Output is grounded when C_INCLUDE_ECC_SUPPORT=0 or C_DDR_DWIDTH=64 and should not be connected.
	P21	DDR_DQS_ECC_o		G6, G27	Output is grounded when C_INCLUDE_ECC_SUPPORT=0 or C_DDR_DWIDTH=64 and should not be connected.
	P22	DDR_DQS_ECC_i		G6, G27	Input is unused when C_INCLUDE_ECC_SUPPORT=0 or C_DDR_DWIDTH=64.
	P23	DDR_DQS_ECC_t		G6, G27	Output is grounded when C_INCLUDE_ECC_SUPPORT=0 or C_DDR_DWIDTH=64 and should not be connected.
	P63	IP2INTC_Irpt		G7, G9, G27	Output is grounded when C_INCLUDE_ECC_INTR=0 or when C_ENABLE_ECC_REG = 0 or C_DDR_DWIDTH=64 and should not be connected.

DDR ECC Register Descriptions

Register Summary

If the DDR SDRAM controller design utilizes ECC (if C_INCLUDE_ECC_SUPPORT = 1 and with C_DDR_DWIDTH = 32) and register usage is enable (if C_ENABLE_ECC_REG = 1), the design contains registers in the ECC core logic. If the ECC core logic utilizes interrupts (C_INCLUDE_ECC_INTR=1), the design contains interrupt registers in the IPIF. Table 4 summarizes these registers.

If the DDR SDRAM controller design does not utilize ECC (if C_INCLUDE_ECC_SUPPORT = 0 or C_DDR_DWIDTH = 64) or register usage is not enabled (if C_ENABLE_ECC_REG = 0), then none of the registers shown in Table 4 are included in the design.

Table 4: **ECC Register Summary**

Grouping	Base Address + Offset (hex)	Register Name	Access Type	Default Value (hex)	Description
ECC Core	C_ECC_BASEADDR + 0	ECCCR(1)	R/W	00000003(4)	ECC Control Register.
	C_ECC_BASEADDR + 4	ECCSR(1)	R/ROW(3)	00000000	ECC Status Register.
	C_ECC_BASEADDR + 8	ECCSEC(1)	R/ROW(3)	00000000	ECC Single Error Count Register.
	C_ECC_BASEADDR + C	ECCDEC(1)	R/ROW(3)	00000000	ECC Double Error Count Register.
	C_ECC_BASEADDR + 10	ECCPEC(1)	R/ROW(3)	00000000	ECC Parity Field Error Count Register.
PLB IPIF ISC	C_ECC_BASEADDR + 11C	DGIE(2)	R/W	00000000	Device Global Interrupt Enable Register
	C_ECC_BASEADDR + 120	IPISR(2)	R/TOW(5)	00000000	IP Interrupt Status Register.
	C_ECC_BASEADDR + 128	IPIER(2)	R/W	00000000	IP Interrupt Enable Register.

Notes:
1. Only used if C_INCLUDE_ECC_SUPPORT=1 and C_ENABLE_ECC_REG=1 and C_DDR_DWIDTH=32.
2. Only used if C_INCLUDE_ECC_SUPPORT=1, C_ENABLE_ECC_REG=1, C_INCLUDE_ECC_INTR=1, and C_DDR_DWIDTH=32.
3. ROW = Reset On Write. A write operation will cause all bits in the register to default to the reset value.
4. Reset condition of ECCCR depends on the value of C_ECC_DEFAULT_ON.
5. TOW = Toggle On Write. TOW is defined as Toggle On Write. Writing a '1' to a bit position within the register causes the corresponding bit position in the register to 'toggle' state.

ECC Control Register (ECCCR)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1 and C_ENABLE_ECC_REG=1 and C_DDR_DWIDTH=32.

The ECC Control Register is shown in Figure 1. The ECC Control Register determines if ECC check bits will be generated in a memory write operation and checked during a memory read operation. The parameter level, odd or even can be set in this register. The ECC Control Register also defines testing modes if enabled by the parameter, C_INCLUDE_ECC_TEST. Table 5 defines the bit values for this register.

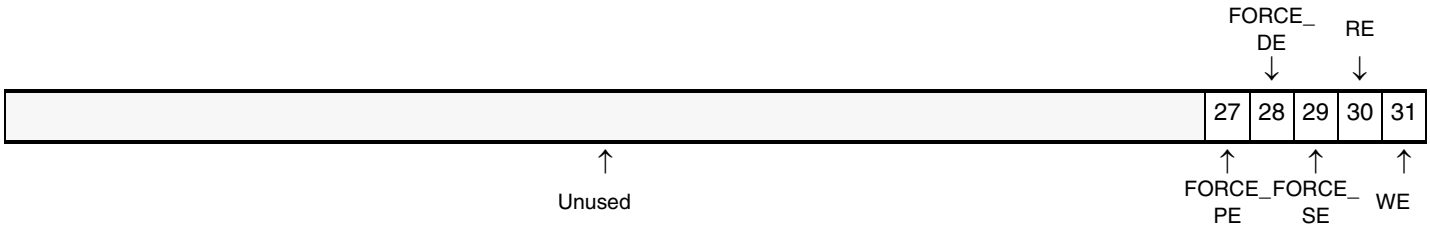


Figure 1: **ECC Control Register**

Table 5: **ECC Control Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-26	Reserved			
27	FORCE_PE	R/W	'0'	**Force Parity Field Bit Error.** Available for testing and determines if parity field bit errors are forced in the data stored in memory. See "ECC Testing" on page 28 for more information. '0' = No parity field bit errors are created '1' = Parity field bit errors are forced in stored data Note: This bit is available only if C_INCLUDE_ECC_TEST=1and C_DDR_DWIDTH=32.
28	FORCE_DE	R/W	'0'	**Force Double Bit Error.** Available for testing and determines if double bit errors are forced in the data stored in memory. See "ECC Testing" on page 28 for more information. '0' = No double bit errors are created '1' = Double bit errors are forced in stored data Note: This bit is available only if C_INCLUDE_ECC_TEST=1 and C_DDR_DWIDTH=32.
29	FORCE_SE	R/W	'0'	**Force Single Bit Error.** Available for testing and determines if single bit errors are forced in the data stored in memory. See "ECC Testing" on page 28 for more information. '0' = No single bit errors are created '1' = Single bit errors are forced in stored data Note: This bit is available only if C_INCLUDE_ECC_TEST=1 and C_DDR_DWIDTH=32.
30	RE	R/W	'1'(1)	**ECC Read Enable.** Determines whether ECC logic is used in memory read operation. '0' = ECC read logic is bypassed '1' = ECC read logic is enabled
31	WE	R/W	'1'(1)	**ECC Write Enable.** Determines whether ECC logic is used in memory write operation. '0' = ECC write logic is bypassed '1' = ECC write logic is enabled

Notes:
1. Reset value is determined by C_ECC_DEFAULT_ON. If C_ECC_DEFAULT_ON=1, then this bit is equal to '1'. If C_ECC_DEFAULT_ON=0, then this bit is equal to '0'.

ECC Status Register (ECCSR)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1 and C_ENABLE_ECC_REG=1 and C_DDR_DWIDTH=32.

The ECC Status Register is illustrated in Figure 2. Table 6 describes the function of each bit in the ECC Status Register.

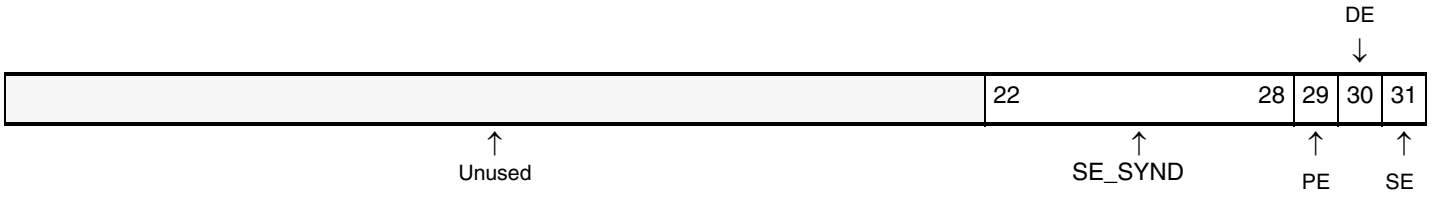


Figure 2: **ECC Status Register**

Table 6: **ECC Status Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-21	Reserved			
22-28	SE_SYND	R/ROW(1)	0000000	**Single Bit Error Syndrome.** Indicates the ECC syndrome value of the most recent memory transaction in which a single bit error was detected. The 7-bit syndrome value indicates the data bit position in which an error was detected and corrected.
29	PE	R/ROW(1)	'0'	**Parity Field Bit Error.** During memory transaction an error was detected in a parity field bit. '0' = No parity bit errors detected '1' = Parity bit error detected and corrected
30	DE	R/ROW(1)	'0'	**Double Bit Error.** During memory transaction a double bit error was detected and is not correctable. '0' = No double bit errors were detected '1' = Double bit error was detected
31	SE	R/ROW(1)	'0'	**Single Bit Error.** During memory transaction a single bit error was detected and corrected. '0' = No single bit errors were detected '1' = Single bit error detected and corrected

Notes:
1. ROW = Reset On Write. Any write operation to the ECCSR will cause all bits in the register to default to the reset value.

ECC Single Error Count Register (ECCSEC)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1 and C_ENABLE_ECC_REG=1 and C_DDR_DWIDTH=32.

The ECC Single Error Count Register records the number of ECC single bit errors that occurred during the memory transaction as shown in Table 7. In the ECC, single bit errors are detected and corrected. This count consumes 12-bits as shown in Figure 3. When the value in this register reaches 4095 (the max count), the next single bit error detected will cause the value in the register to reset to 0.

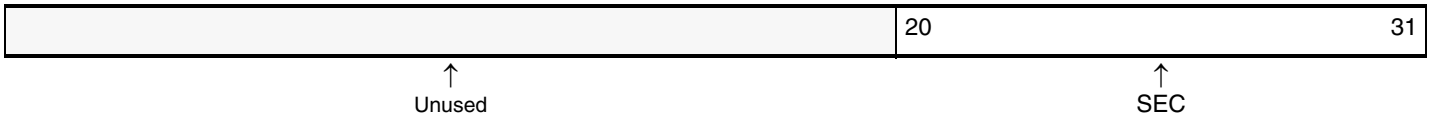


Figure 3: **ECC SEC Register**

Table 7: **ECC SEC Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-19	Reserved			
20-31	SEC	R/ROW(1)	0	**Single Error Count.** Indicates the number of single bit errors that occurred during the last memory transaction. The maximum error count is 4095.

Notes:
1. ROW = Reset On Write. Any write operation to the ECCSEC register will cause all bits to default to the reset value.

ECC Double Error Count Register (ECCDEC)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1 and C_ENABLE_ECC_REG=1 and C_DDR_DWIDTH=32.

The ECC Double Error Count Register records the number of ECC double bit errors that occurred during the last memory transaction shown in Table 8. In the ECC, double bit errors are detected and not correctable. This count consumes 12-bits as shown in Figure 4. When the value in this registe

r reaches 4095 (the max count), the next double bit error detected will cause the value in the register to reset to 0.

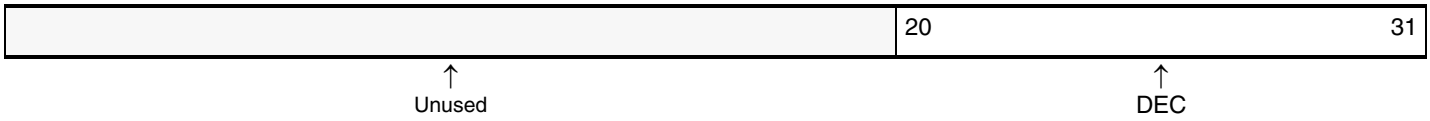


Figure 4: **ECC DEC Register**

Table 8: **ECC DEC Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-19	Reserved			
20-31	DEC	R/ROW(1)	0	**Double Error Count.** Indicates the number of double bit errors that occurred during the last memory transaction. The maximum error count is 4095.

Notes:
1. ROW = Reset On Write. Any write operation to the ECCDEC register will cause all bits to default to the reset value.

ECC Parity Error Count Register (ECCPEC)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1, C_ENABLE_ECC_REG=1, and C_DDR_DWIDTH=32.

The ECC Parity Error Count Register records the number of bit errors that occurred in the ECC parity field during the last memory transaction shown in Table 9. In the ECC, parity field bit error are detected and corrected. This count consumes 12-bits as shown in Figure 5. When the value in this register reaches 4095 (the max count), the next parity field bit error detected will cause the value in the register to reset to 0.

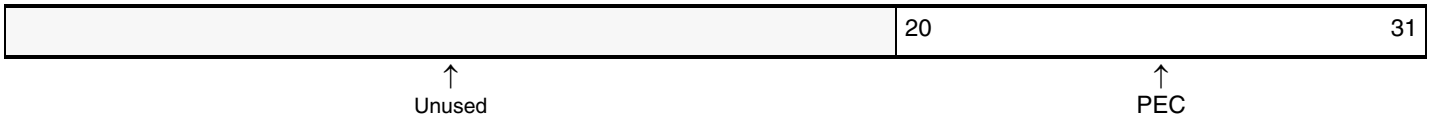


Figure 5: **ECC PEC Register**

Table 9: **ECC PEC Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-19	Reserved			
20-31	PEC	R/ROW(1)	0	**Parity Field Error Count.** Indicates the number of errors that occurred in the parity field bits during the last memory transaction. The maximum error count is 4095.

Notes:
1. ROW = Reset On Write. Any write operation to the ECCPEC register will cause all bits to default to the reset value.

ECC Interrupt Descriptions

Note: The interrupts described here are only available if C_INCLUDE_ECC_SUPPORT=1, C_ENABLE_ECC_REG=1, and C_INCLUDE_ECC_INTR=1 and C_DDR_DWIDTH=32.

The ECC module has 3 distinct interrupts that are sent to the IPIF. The IPIF utilizes the IP ISC and allows each interrupt to be enabled independently (via the IPIER).

Device Global Interrupt Enable Register (DGIE)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1, C_INCLUDE_ECC_INTR=1, and C_DDR_DWIDTH=32.

The Device Global Interrupt Enable Register is used to globally enable the final interrupt output from the IPIF Interrupt service as shown in Figure 6 and described in Table 10.

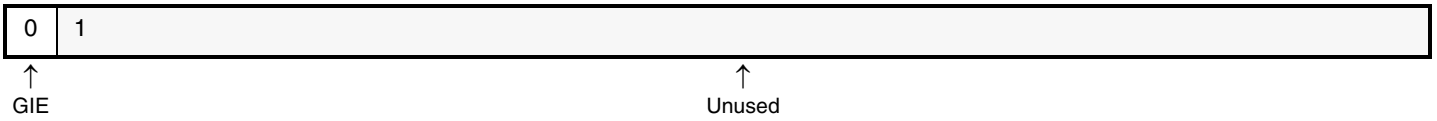


Figure 6: **DGIE Register**

Table 10: **DGIE Register Description**

Bit(s)	Name	Core Access	Reset Value	Description
0	GIE	R/W	'0'	Global Interrupt Enable. '0' = Interrupts disabled '1' = Interrupts enabled
1-31		Read	zeros	Unused

IP Interrupt Status Register (IPISR)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1, C_INCLUDE_ECC_INTR=1, and C_DDR_DWIDTH=32.

The IP Interrupt Status Register is the interrupt capture register for the DDR ECC logic as shown in Figure 7 and described in Table 11.

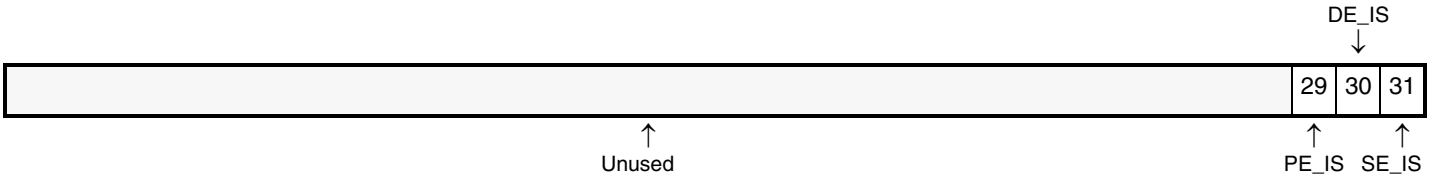


Figure 7: **IPISR Register**

Table 11: **IPISR Description**

Bit(s)	Name	Core Access	Reset Value	Description
0-28	Reserved			
29	PE_IS	R/TOW(1)	'0'	**Parity Field Error Interrupt Status.** Indicates a parity field bit error has occurred during the memory data transaction. In the ECC, parity field bit errors will be corrected as data is read from memory. This interrupt is for system monitoring only and does not indicate corrupt data. '0' = not active '1' = active
30	DE_IS	R/TOW(1)	'0'	**Double Bit Error Interrupt Status.** Indicates a double bit data error has occurred during the memory transaction. In the ECC, double bit errors can be detected, but not corrected. When this interrupt is asserted, the data read from memory is not valid. '0' = not active '1' = active
31	SE_IS	R/TOW(1)	'0'	**Single Bit Error Interrupt Status.** Indicates a single bit error has been detected during the memory transaction. In the ECC, single bit errors will be detected and corrected. This interrupt is for system monitoring only and does not indicate corrupt data. '0' = not active '1' = active

Notes:
1. TOW is defined as Toggle On Write. Writing a '1' to a bit position within the register causes the corresponding bit position in the register to 'toggle' state.

IP Interrupt Enable Register (IPIER)

Note: This register is available only when C_INCLUDE_ECC_SUPPORT=1 and C_ENABLE_ECC_REG=1 and C_INCLUDE_ECC_INTR=1 and C_DDR_DWIDTH=32.

The IP Interrupt Enable Register has an enable bit for each defined bit of the IP Interrupt Status Register as shown in Figure 8 and described in Table 12.

Figure 8: **IPIER Register**

Table 12: **IPIER Description**

Bit(s)	Name	Core Access	Reset Value	Description
0-28	Reserved			
29	PE_IE	R/W	'0'	**Parity Field Error Interrupt Enable.** Enables assertion of the interrupt for indicating parity field bit errors have occurred. '0' = disabled '1' = enabled
30	DE_IE	R/W	'0'	**Double Bit Error Interrupt Enable.** Enables assertion of the interrupt for indicating double bit data errors have occurred. '0' = disabled '1' = enabled
31	SE_IE	R/W	'0'	**Single Bit Error Interrupt Enable.** Enables assertion of the interrupt for indicating single bit data errors have occurred. '0' = disabled '1' = enabled

Connecting to Memory

Memory Data Types and Organization

DDR SDRAM memory can be accessed as byte (8 bits), halfword (2 bytes),,word (4 bytes) or Double word (8 bytes) depending on the size of the bus to which the processor is attached. From the point of view of the PLB, data is organized as big-endian. The bit and byte labeling for the big-endian data types is shown below in Figure 9.

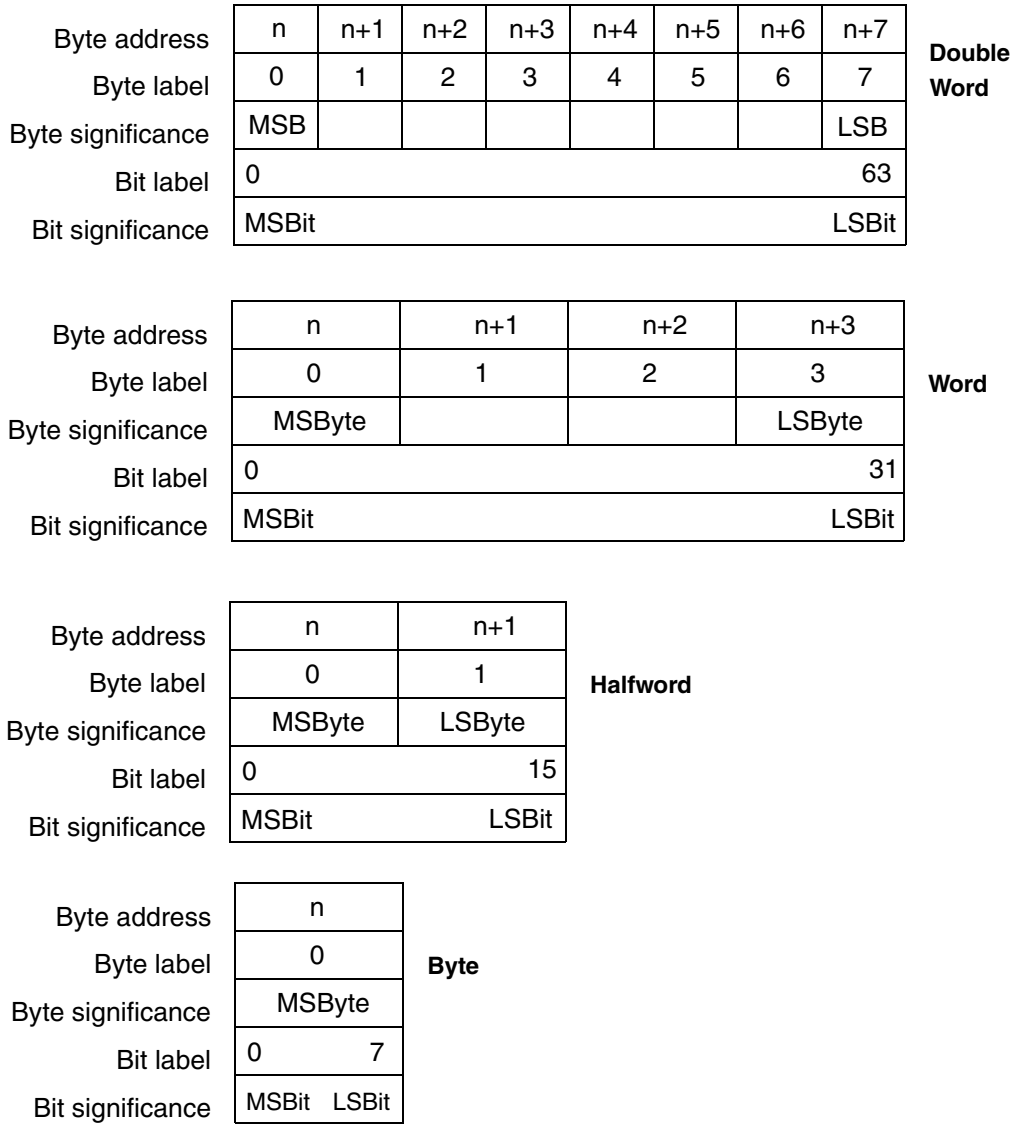


Figure 9: **Big-Endian Data Types**

Memory to DDR SDRAM Controller Connections

The data and address signals at the DDR SDRAM controller are labeled with big-endian bit labeling (for example, D(0:31), D(0) is the MSB), whereas most memory devices are either endian agnostic (they can be connected either way) or little-endian D(31:0) with D(31) as the MSB.

Caution must be exercised with the connections to the external memory devices to avoid incorrect data and address connections.

Table 13 shows the correct mapping of DDR SDRAM controller pins to memory device pins for 32-bit data width. Figure 10 shows the interconnection between DDR SDRAM controller and DDR SDRAM for 32-bit data width.

Table 13: **DDR SDRAM controller to memory interconnect for 32-bit Data width DDR SDRAM Interface**

Description	DDR SDRAM Controller Signal (MSB:LSB)	Memory Device Signal (MSB:LSB)
Data Bus	DDR_DQ(0:C_DDR_DWIDTH-1)	DQ(C_DDR_DWIDTH-1:0)
Bank Address	DDR_BankAddr(0:C_DDR_BANK_AWIDTH-1)	BA(C_DDR_BANK_AWIDTH-1:0)
Address	DDR_Addr(0:C_DDR_AWIDTH-1)	A(C_DDR_AWIDTH-1:0)
Data Strobe	DDR_DQS(0:C_DDR_DWIDTH/8-1)	UDQS, LDQS
Data Mask	DDR_DM(0:C_DDR_DWIDTH/8-1)	UDM, LDM
ECC Check Bits	DDR_DQ_ECC(0:NUM_ECC_BITS-1)	DQ_ECC(NUM_ECC_BITS-1:0)
ECC Data Strobe	DDR_DQS_ECC	DQS_ECC
ECC Data Mask	DDR_DM_ECC	DM_ECC

Notes:
1. NUM_ECC_BITS=7 (includes 6 parity bits plus 1 overall parity bit)

Example

Figure 10 illustrates an example of connecting memory to the DDR SDRAM controller design with ECC logic enabled. The example shown here has the following specified parameters:

- C_INCLUDE_ECC_SUPPORT=1
- C_NUM_BANKS_MEM=1
- C_DDR_DWIDTH = 32
- NUM_ECC_BITS = 7
- C_DDR_BANK_AWIDTH = 2
- C_DDR_AWIDTH = 13

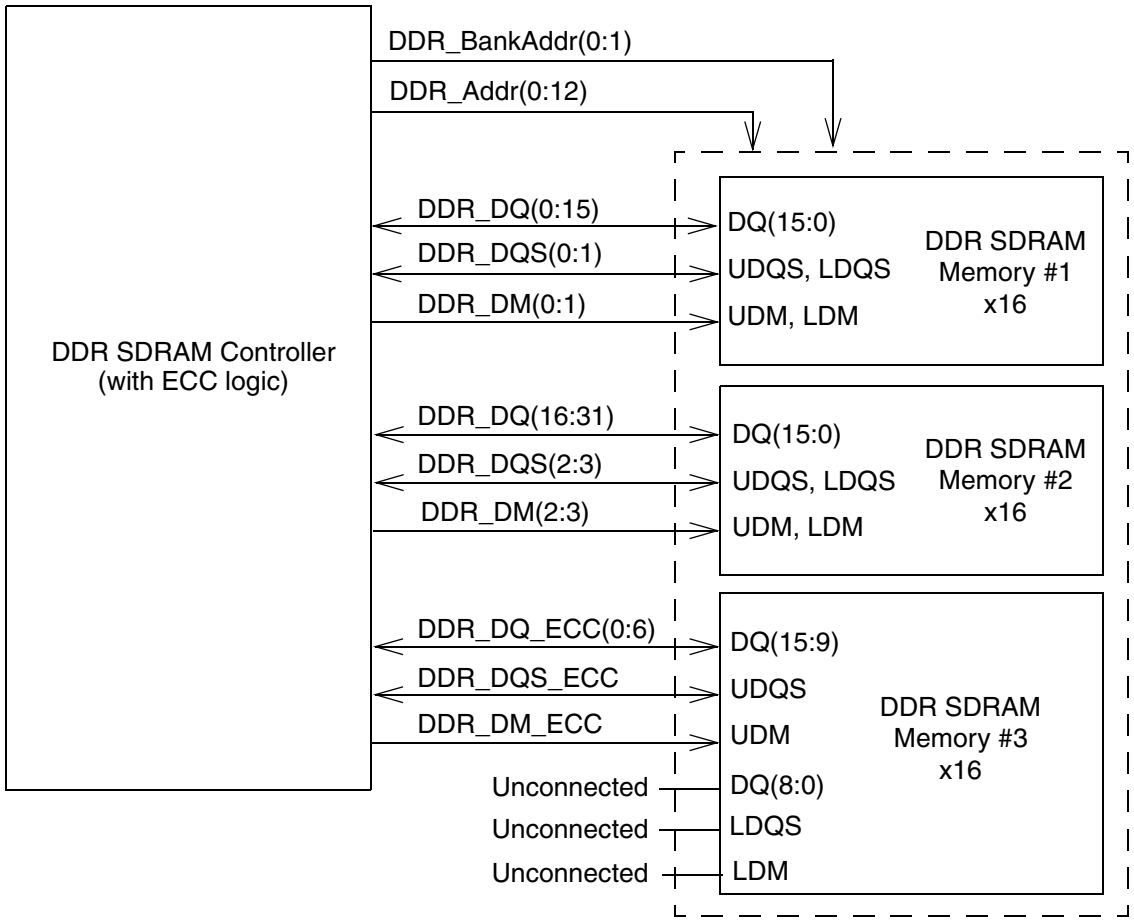


Figure 10: **PLB 32-bit DDR SDRAM Controller to Memory Connection Example Block Diagram**

Table 14 shows the correct mapping of DDR SDRAM controller pins to memory device pins for 62-bit data width. Figure 11 shows the interconnection between DDR SDRAM controller and DDR SDRAM for 62-bit data width.

Table 14: **PLB DDR SDRAM controller to memory interconnect for 64-bit DDR SDRAM Interface**

Description	PLB DDR SDRAM Controller Signal (MSB:LSB)	Memory Device Signal (MSB:LSB)
Data Bus	DDR_DQ(0:C_DDR_DWIDTH-1)	DQ(C_DDR_DWIDTH-1:0)
Bank Address	DDR_BankAddr(0:C_DDR_BANK_AWIDTH-1)	BA(C_DDR_BANK_AWIDTH-1:0)
Address	DDR_Addr(0:C_DDR_AWIDTH-1)	A(C_DDR_AWIDTH-1:0)
Data Strobe	DDR_DQS(0:C_DDR_DWIDTH/8-1)	DQS(C_DDR_DWIDTH/8-1:0)
Data Mask	DDR_DM(0:C_DDR_DWIDTH/8-1)	DM(C_DDR_DWIDTH/8-1:0)

Example

Figure 11 illustrates an example of connecting memory to the DDR SDRAM controller for 64-bit DDR SDRAM. The example shown here has the following specified parameters:

- C_INCLUDE_ECC_SUPPORT=0
- C_NUM_BANKS_MEM=1
- C_DDR_DWIDTH = 64
- C_DDR_BANK_AWIDTH = 2
- C_DDR_AWIDTH = 13

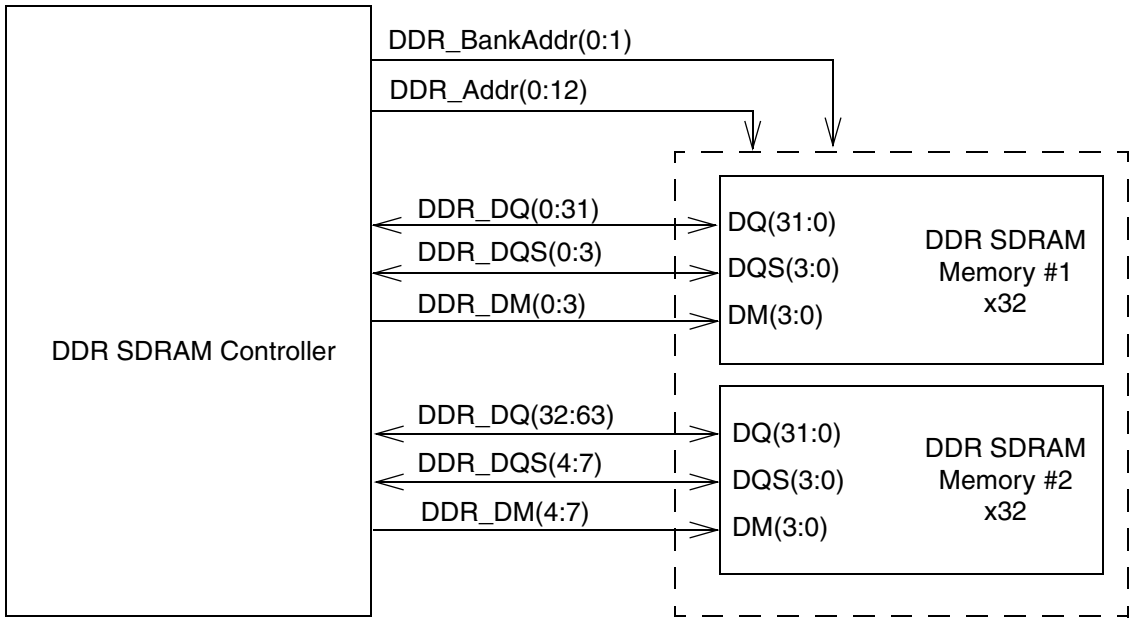


Figure 11: **PLB 64-bit DDR SDRAM Controller to Memory Connection Example Block Diagram**

DDR Address Mapping

An address offset is calculated based on the width of the DDR data bus. The DDR column address is then mapped to the PLB address bus, followed by the row address and bank address.

The PLB address bus bit locations for the DDR column, row, and bank addresses are calculated as shown in Table 15 and Table 16.

Table 15: **DDR Address offset calculations**

Variable	Equation
ADDR_OFFSET	log2(C_DDR_DWIDTH/8)
COLADDR_STARTBIT	C_PLB_AWIDTH - (C_DDR_COL_AWIDTH+ADDR_OFFSET)
COLADDR_ENDBIT	COLADDR_STARTBIT + C_DDR_COL_AWIDTH- 2 (A0 is not used)
ROWADDR_STARTBIT	COLADDR_STARTBIT - C_DDR_AWIDTH

Table 15: **DDR Address offset calculations *(Continued)***

Variable	Equation
ROWADDR_ENDBIT	ROWADDR_STARTBIT + C_DDR_AWIDTH-1
BANKADDR_STARTBIT	ROWADDR_STARTBIT - C_DDR_BANK_AWIDTH
BANKADDR_ENDBIT	BANKADDR_STARTBIT + C_DDR_BANK_AWIDTH-1

Table 16: **DDR - PLB Address Bus Assignments**

DDR Address	PLB Address Bus
Column Address	PLB_ABus(COLADDR_STARTBIT to COLADDR_ENDBIT) & '0'
Row Address	PLB_ABus(ROWADDR_STARTBIT to ROWADDR_ENDBIT)
Bank Address	PLB_ABus(BANKADDR_STARTBIT to BANKADDR_ENDBIT)

Table 17 and Table 18 show an example of the mapping between the PLB address and the DDR address when the data width of the DDR is 32 and the data width of the PLB bus is 64, the column address width is 9, the row address width is 13, and the bank address width is 2.

Table 17: **Example DDR Address offset calculations for C_DDR_DWIDTH=32**

Variable	Value
ADDR_OFFSET	log2(32/8) = 2
COLADDR_STARTBIT	32 - (9+2) = 21
COLADDR_ENDBIT	21 +(9-2) = 28
ROWADDR_STARTBIT	21 - 13 = 8
ROWADDR_ENDBIT	8 + 13 -1 = 20
BANKADDR_STARTBIT	8 - 2 = 6
BANKADDR_ENDBIT	6+ 2 -1 = 7

Table 18: **DDR - PLB Address Bus Assignments for C_DDR_DWIDTH=32**

DDR Address	PLB Address Bus
Column Address	PLB_ABus(21: 28) &'0'
Row Address	PLB_ABus(8:20)
Bank Address	PLB_ABus(6:7)

Table 19 and Table 20 show an example of the mapping between the PLB address and the DDR address when the data width of the DDR is 64 and the data width of the PLB bus is 64, the column address width is 8, the row address width is 13, and the bank address width is 2.

Table 19: **PLB Example DDR Address offset calculations for C_DDR_DWIDTH=64**

Variable	Value
ADDR_OFFSET	log2(64/8) = 3
COLADDR_STARTBIT	32 - (8+3) = 21
COLADDR_ENDBIT	21 +(8-2) = 27
ROWADDR_STARTBIT	21 - 13 = 8
ROWADDR_ENDBIT	8 + 13 -1 = 20
BANKADDR_STARTBIT	8 - 2 = 6
BANKADDR_ENDBIT	6+ 2 -1 = 5

Table 20: **DDR - PLB Address Bus Assignments for C_DDR_DWIDTH=64**

DDR Address	PLB Address Bus
Column Address	PLB_ABus(21:27) &'0'
Row Address	PLB_ABus(8:20)
Bank Address	PLB_ABus(4:5)

IMPORTANT: Virtex-II and Virtex-II Pro IO pairs share input and output clock signals. Since the DDR registers in the IO blocks use both of the input and output clock signals, the ports assigned to the IO pairs MUST use the same input and output clocks. Care should be taken when making port IO assignments that the DDR_DQ and DDR_DM signals use the system clock as the output clock and the DDR_DQS signals use a 90 degree phase shift of the system clock as the output clock. Therefore, a DDR_DQS signal should not be assigned with a DDR_DQ signal or a DDR_DM signal in an IO pair.

Since this DDR controller design utilizes the DDR registers in the Virtex-II and Virtex-II Pro FGPA IO blocks, this controller is not suitable for other FPGA families.

Since the DDR_DQ and DDR_DQS busses are 3-stateable, the user should pullup these signals in the FPGA IO blocks or external to the FPGA in the board design. Note that the DDR controller design will drive the DQS signals to a '1' during the IDLE state so only one DDR controller can be used to control a DDR memory, i.e., two DDR controllers can not share the same DDR memory.

PLB DDR SDRAM Controller Design

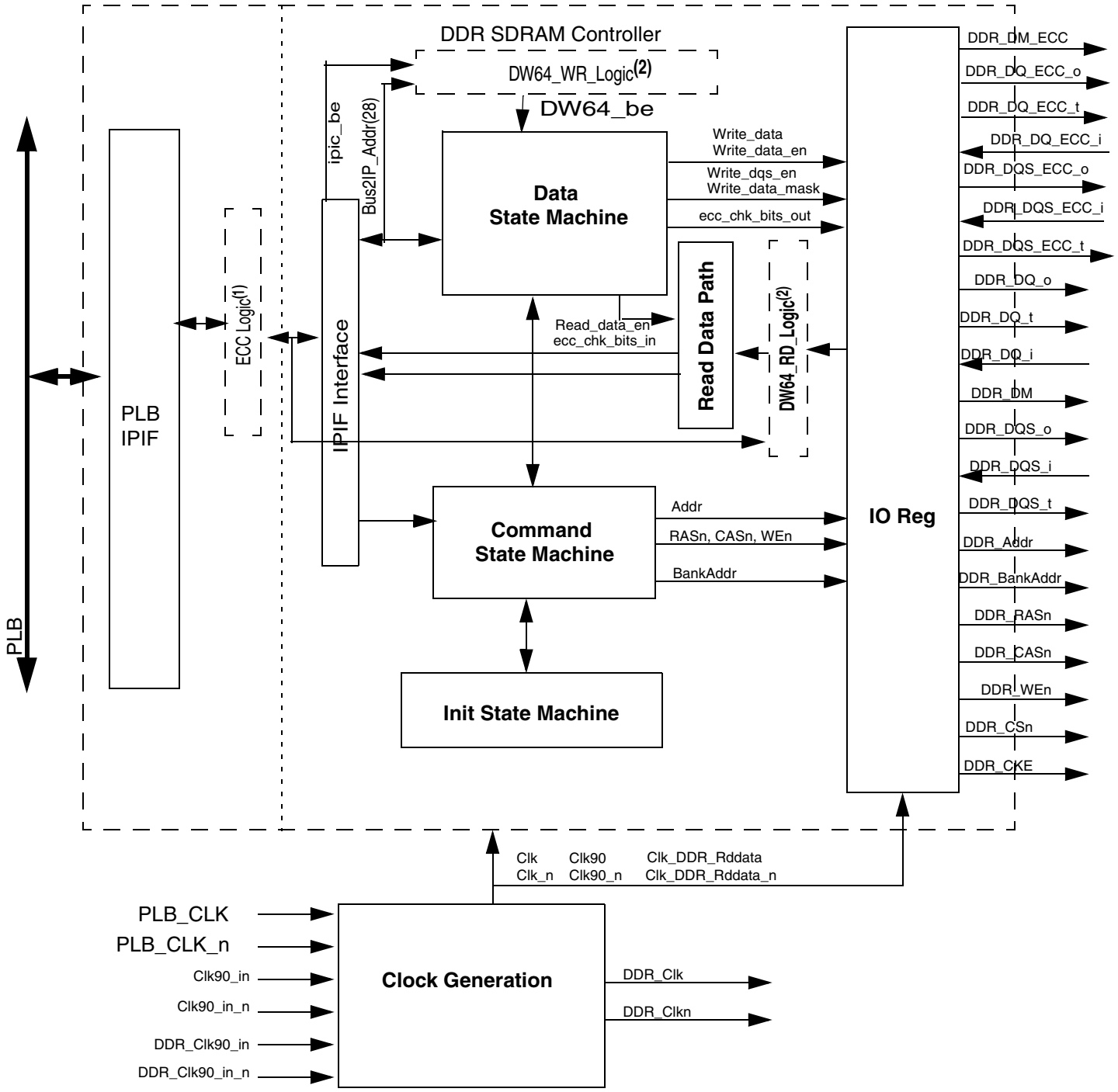
Block Diagram

The block diagram for PLB DDR SDRAM Controller is shown in Figure 12. The PLB DDR SDRAM controller consists of -

1. PLB IPIF interface to provide the bus protocol
2. Data state machine for data transfer to DDR SDRAM
3. Command state machine for execution of DDR command
4. Inititialization state machine for initilization of DDR SDRAM
5. I/O module to instantiate the DDR I/O registers for the DDR data interface
6. A clock generation module
7. ECC logic
8. Read and write logic to support DDR data width of 64 bits (DW64 RD Logic & DW64 WR Logic)

The separation of the Command State Machine and the Data State Machine allows for the application of commands to the DDR while data reception/transmission is in progress. Overlapping the DDR commands with the data transfer when accessing data in the same row of the same bank allows for more optimal DDR operation.

ECC Logic, DW64 RD Logic and DW64 WR Logic are optional design blocks that are included in the design by the design parameters. The PLB DDR Controller is designed to support ECC for 32-bit data width interface of DDR SDRAM, while 64-bit data width interface of DDR SDRAM can not support ECC.



Notes:
1) Included only when C_INCLUDE_ECC_SUPPORT=1 and C_DDR_DWIDTH=32.
2) Included only when C_INCLUDE_ECC_SUPPORT = 0 and C_DDR_DWIDTH=64.

Figure 12: **PLB DDR SDRAM Controller Block Diagram**

ECC Logic

As illustrated in Figure 12, the ECC logic fits between the IPIF Interface and the DDR controller logic when C_INCLUDE_ECC_SUPPORT=1 and C_DDR_DWIDTH=32. Figure 13 illustrates the control and data signals intercepted by the ECC logic from the IPIC to the IPIF Interface of the DDR controller logic. A level of muxing is created when using ECC on these signals to enable or disable the use of ECC read or ECC write logic.

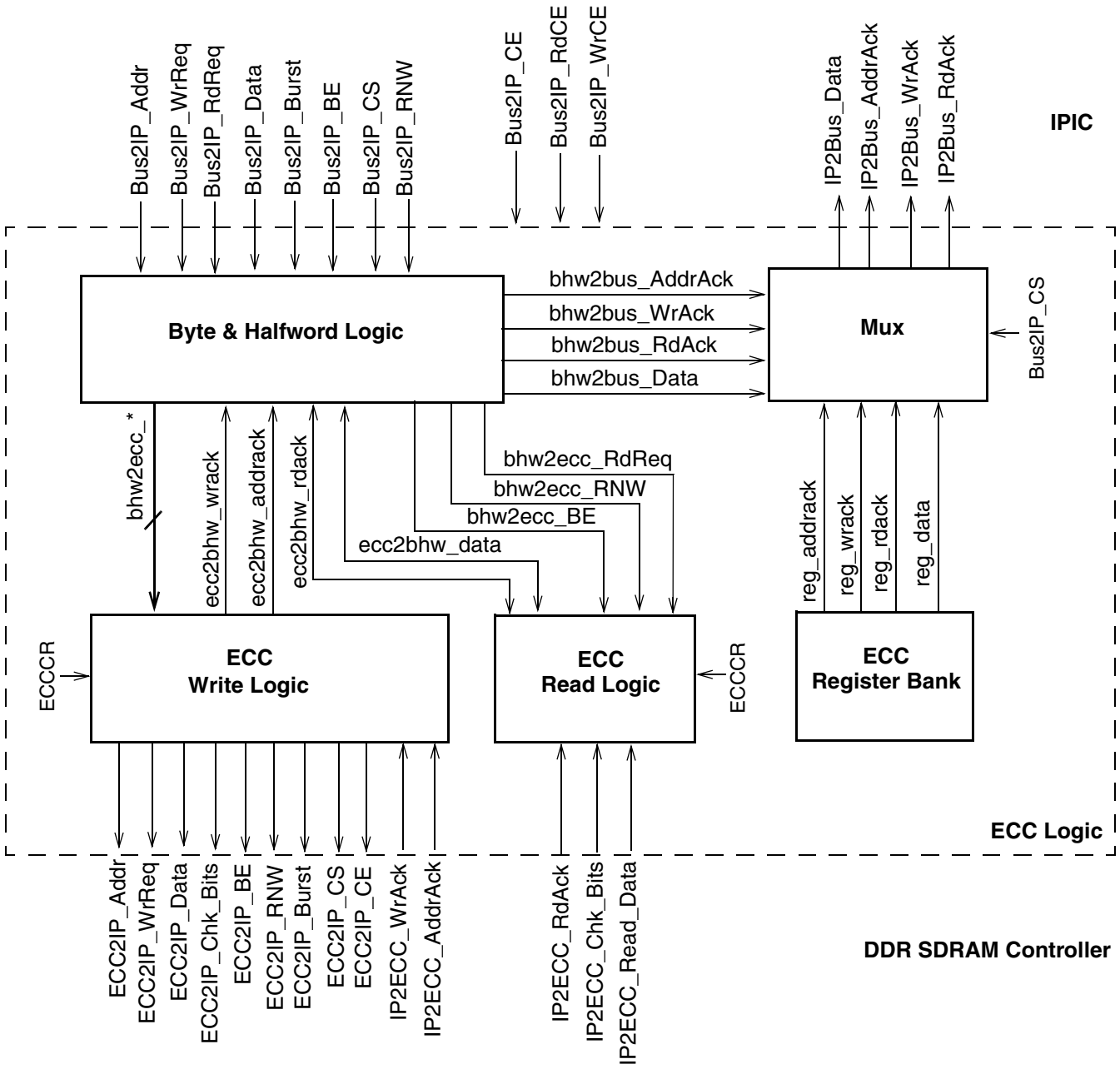


Figure 13: **ECC Logic Block Diagram**

The byte and halfword logic shown in Figure 13 is utilized during an active byte or halfword write operation when ECC write logic is enabled. With the ECC check bit data generated and stored for each 32-bits of DDR data, a read-modify-write operation is required for any PLB byte or halfword write request to DDR memory. The byte or halfword data to write is muxed with

the 32-bit word after reading from the corresponding memory location. The new 32-bit data word is then presented to the ECC write logic and the ECC check bits are re-created for the memory write. Hence an additional latency of a complete read cycle should be expected on any byte or halfword write transaction when ECC write logic is enabled.

In a write cycle, the ECC logic will pipeline WrReq to the DDR Command State Machine until the ECC check bit data is ready to write to memory. When ECC write logic is enabled, the write transaction latency will increase by two clock cycles.

During a memory read, the ECC check bits read from memory, IP2ECC_Chk_Bits, are compared with the check bits calculated on the data word read. The result of this comparison is stored in the syndrome and indicates the data bit or parity bit in error. Table 21 illustrates the error type decoding based on the syndrome.

Table 21: **ECC Error Decoding**

Syndrome (MSB)	Syndrome (MSB-1:0)	Result
0	= 0	No errors in memory read.
1	$\neq 0$	Single bit error. Syndrome holds bit position to correct. If the Syndrome(MSB-1:0) has a single bit = '1', then a parity field bit error is detected. The Syndrome(MSB-1:0) holds the parity field bit to correct. When NUM_ECC_BITS=7, the following values for Syndrome(MSB-1:0) would represent a parity field bit error: • 000000 • 000001 • 000010 • 000100 • 001000 • 010000 • 100000
0	$\neq 0$	Double bit error. Not correctable.

ECC Testing

To enable testing on the ECC core logic, C_INCLUDE_ECC_TEST=1.

The ECC Control Register (ECCCR) described in "ECC Control Register (ECCCR)" on page 11 includes control bits for enabling or disabling the test logic. The following list describes possible forcing errors combinations. If any other combination is attempted, no errors will be forced on the data written to memory.

- No bit error forcing
- Force single bit data errors (ECC Control Register (ECCCR): FORCE_SE = '1')
- Force double bit data errors (ECC Control Register (ECCCR): FORCE_DE = '1')
- Force parity bit errors (ECC Control Register (ECCCR): FORCE_PE = '1')
- Force single bit data and single bit parity errors (ECC Control Register (ECCCR): FORCE_SE = '1' and FORCE_PE = '1')

For single bit error testing, a mask shift register forces single bit errors on the data written to memory. The 64-bit mask shift register has the least significant single bit equal to '1'. When testing is enabled during a memory write, the shift register clocks the '1' towards to most significant bit.

For double bit error testing, a mask shift register forces double bit errors on the data written to memory. The 64-bit mask shift register has two adjacent bits equal to '1' (starting in the two least significant bit positions) and rotates the "11" pattern in the shift register (towards the two most significant bits) on each memory write when testing for double bit errors is enabled.

When parity field bit error testing is enabled, a mask shift register (with a size of NUM_ECC_BITS * 2) forces single bit errors on the check bits stored in memory. The check bit mask shift register has the least significant single bit equal to '1' and rotates the '1' (towards the most significant bit) on each memory write when parity field bit error testing is enabled.

Note: For all ECC testing conditions, note that an error might not be forced into memory if only writing a 32-bit doubleword.

Init State Machine

DDR SDRAMs must be powered-up and initialized in a predefined manner. After power supplies and all the clocks are stable, the DDR SDRAM requires a 200uS delay prior to applying an executable command.

The Init State Machine provides the 200uS delay and the sequencing of the required DDR start-up commands.It instructs the Command State Machine to send the proper commands in the proper sequence to the DDR. This state machine starts execution after Reset and returns to the IDLE state when Reset is applied.

When the initialization sequence has been completed, the INIT_DONE signal asserts.

Note that after Reset has been applied, the 200uS delay is again implemented before any commands are issued to the DDR. The 200uS delay must be accounted for in simulation as well as the delay of the command sequence.

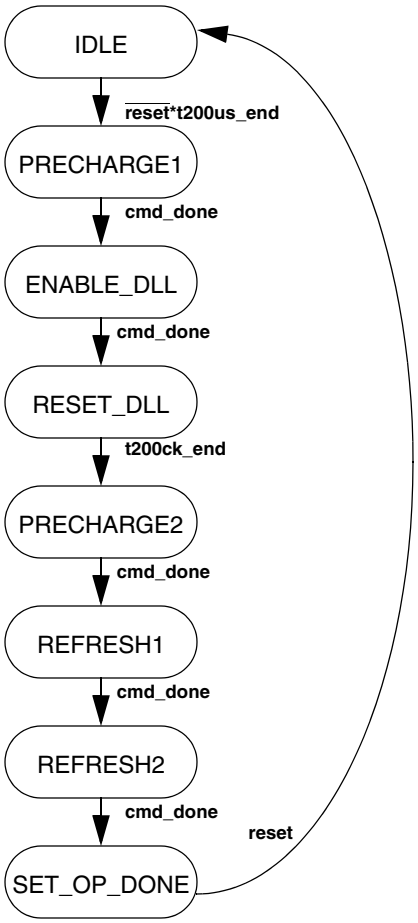


Figure 14: **DDR Init State Machine**

Command State Machine

The Command State Machine provides the address bus and commands signals to the DDR. It sends the DATA_EN signal to the Data State Machine to start the reception/transmission of data. If a burst transaction is in progress or a secondary transaction has been received, the Command State Machine will send the next command to the DDR while data reception/transmission is still in progress to optimize the DDR operation.

A simplified version of the Command State Machine is shown in Figure 15. For readability, only the major state transitions are shown.

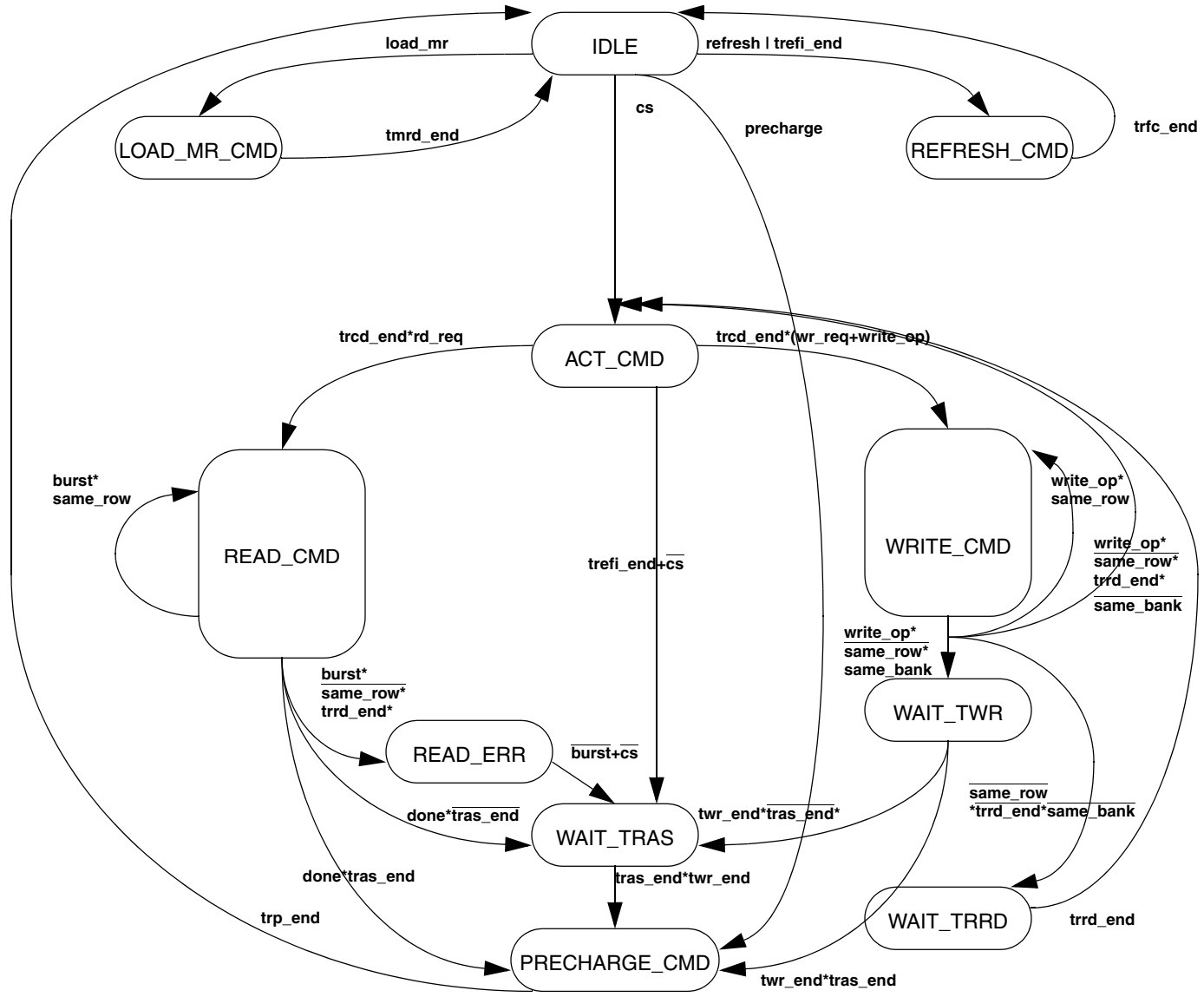


Figure 15: **DDR Command State Machine**

Data State Machine

The Data State Machine transfers the data to/from the DDR and determines when the specified DDR burst is complete. It monitors the PEND_OP signal from the Command State Machine to know if more data transmissions are required. It waits for CAS_LATENCY during read operations and signals when the DDR has completed the data transfer for both read and write operations. It provides the READ_DATA_EN signal to the input DDR registers and read data FIFO.

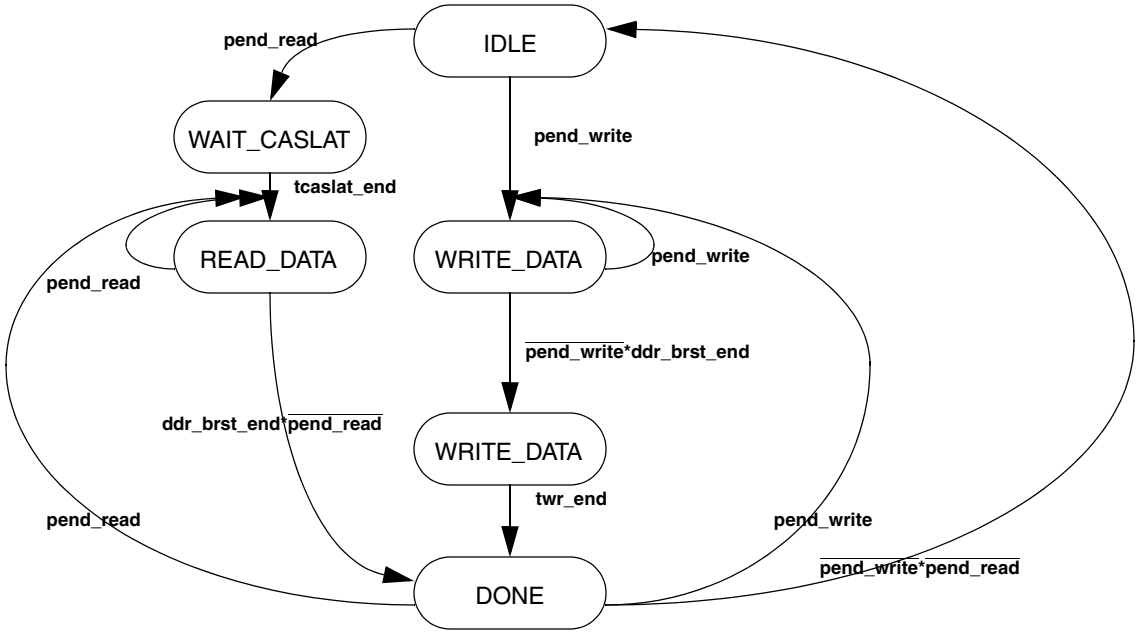


Figure 16: **DDR Data State Machine**

Double Word (64 bit) Writer Logic

As shown in Figure 17, the DW64 WR Logic contains logic to support the writing of 64-bit data to the DDR SDRAM memory bank on each clock edge. It interfaces to the IPIC_IF logic and the Data State Machine, creating a 128-bit data word for presentation to the DDR SDRAM memory. This logic will be instantiated in the design only when C_DDR_DWIDTH=64 and C_INCLUDE_ECC_SUPPORT=0.

Interface detail for DW64 WR Logic is shown in Table 22 and in Figure 17.

Table 22: **DW64 WR Logic Signal Interface detai**

Signal Name	Interface Module	I/O	Description
Bus2ip_Addr(28)	IPIC	I	Controls data steering to upper or lower 64-bit word when presented as 128-bits to the DDR SDRAM memory.
ipic_be(0:7)	IPIC_IF	I	Data enables from the IPIC interface logic
DW64_be(0:15)	Data State Mmachine	O	Generated byte enables presented to the data state machine for the 128-bit data vector presented to the DDR SDRAM memory.

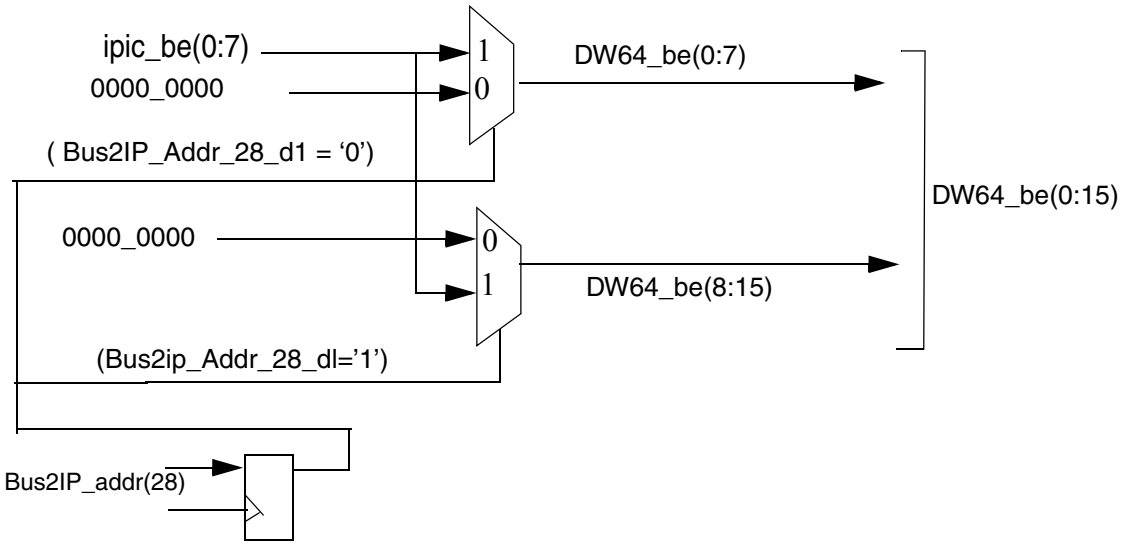


Figure 17: **DW64 WR Logic Detailed Design**

The ipic_be signal of width C_IPIC_DWIDTH/8 are routed onto the DW64_be(0:7) or DW64_be(8:15) depending on the Bus2IP_Addr(28) bit of BusIP_Addr(0:31).

If Bus2IP_Addr(28) is zero then the ipic_be(0:7) signals are driven onto DW64_be(0:7) and DW64_be(8:15) is driven low and if Bus2IP_Addr(28) is high then the ipic_be(0:7) signals are driven onto DW64_be(8:15) and DW64_be(0:7) is driven low as shown in.

DW64_be(0:15) is routed to data state machine module used to generate corresponding Data mask signals. These Data mask signals act as qualifier for the data driven onto the DDR device.

If Bus2IP_Addr(28) is zero then the data will be written on the positive edge of the clock and the data on the negative edge of the clock will be masked Figure 18.

If Bus2IP_Addr(28) is high then the data will be written on the negative edge of the clock and the data on the positive edge of the clock will be masked Figure 19.

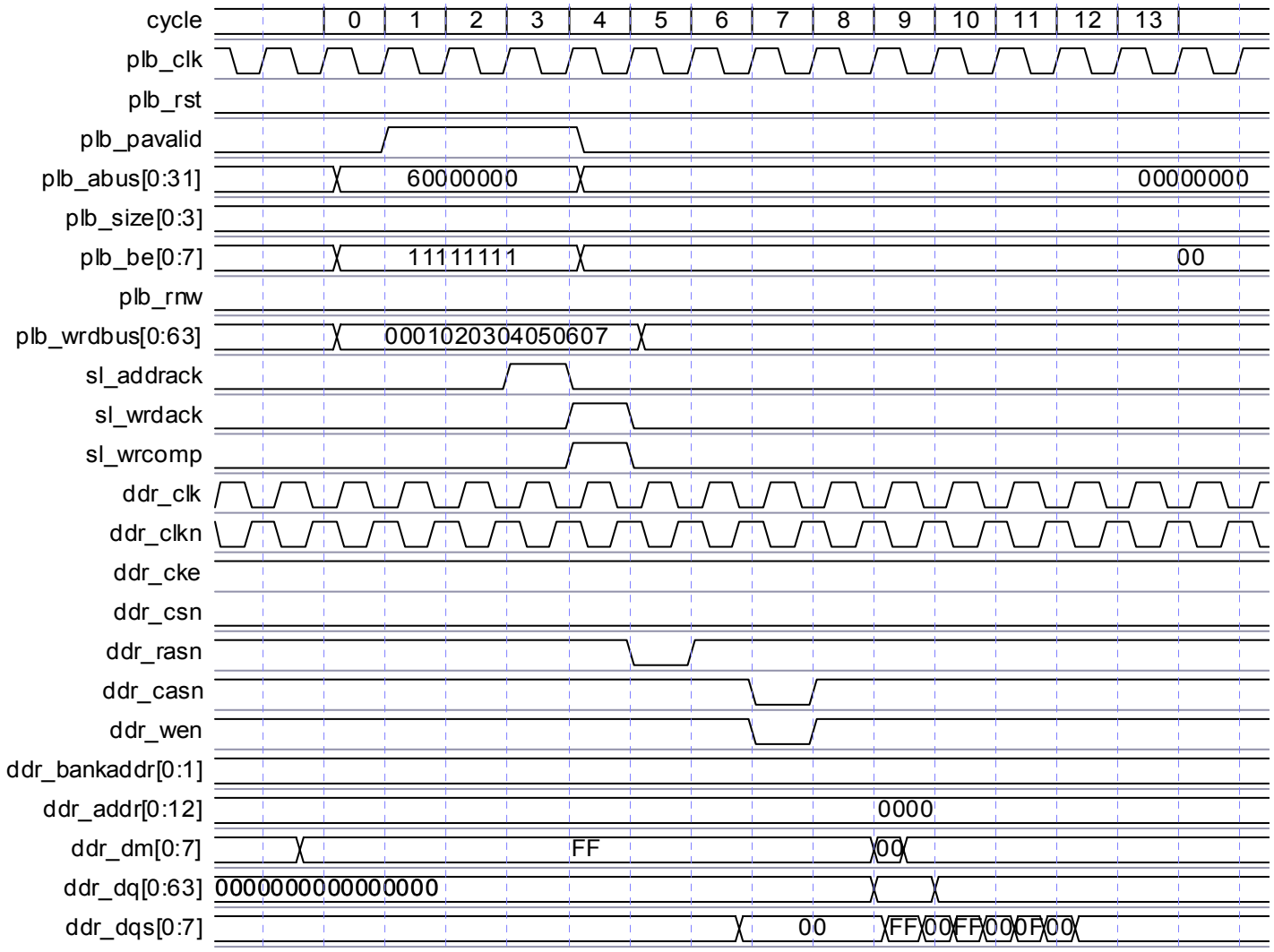


Figure 18: **Single Double Word Write Transaction with Bus2IP_Addr(28)='0'**

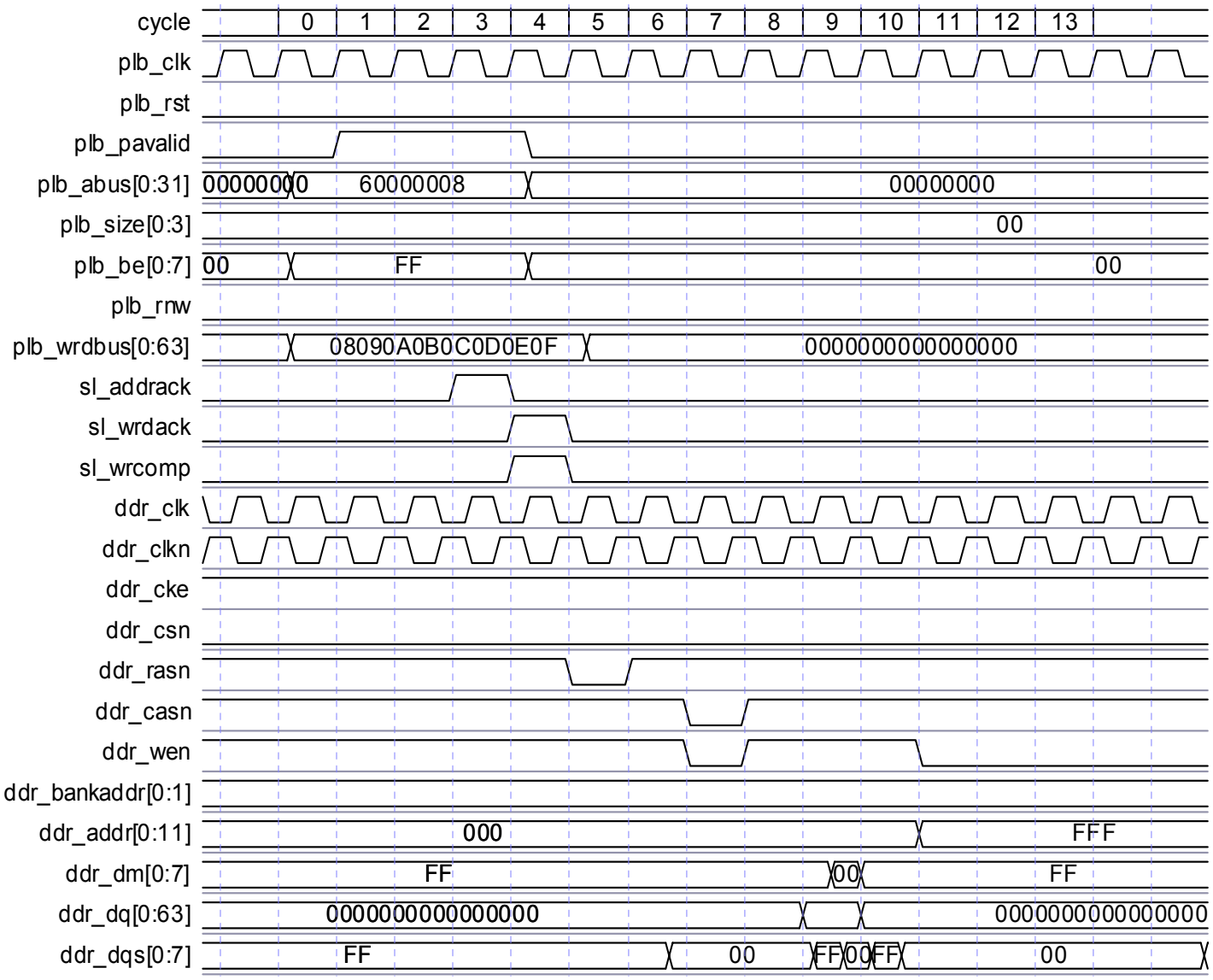


Figure 19: **Single Double Word Write Transaction with Bus2IP_Addr(28)='1'**

I/O Registers

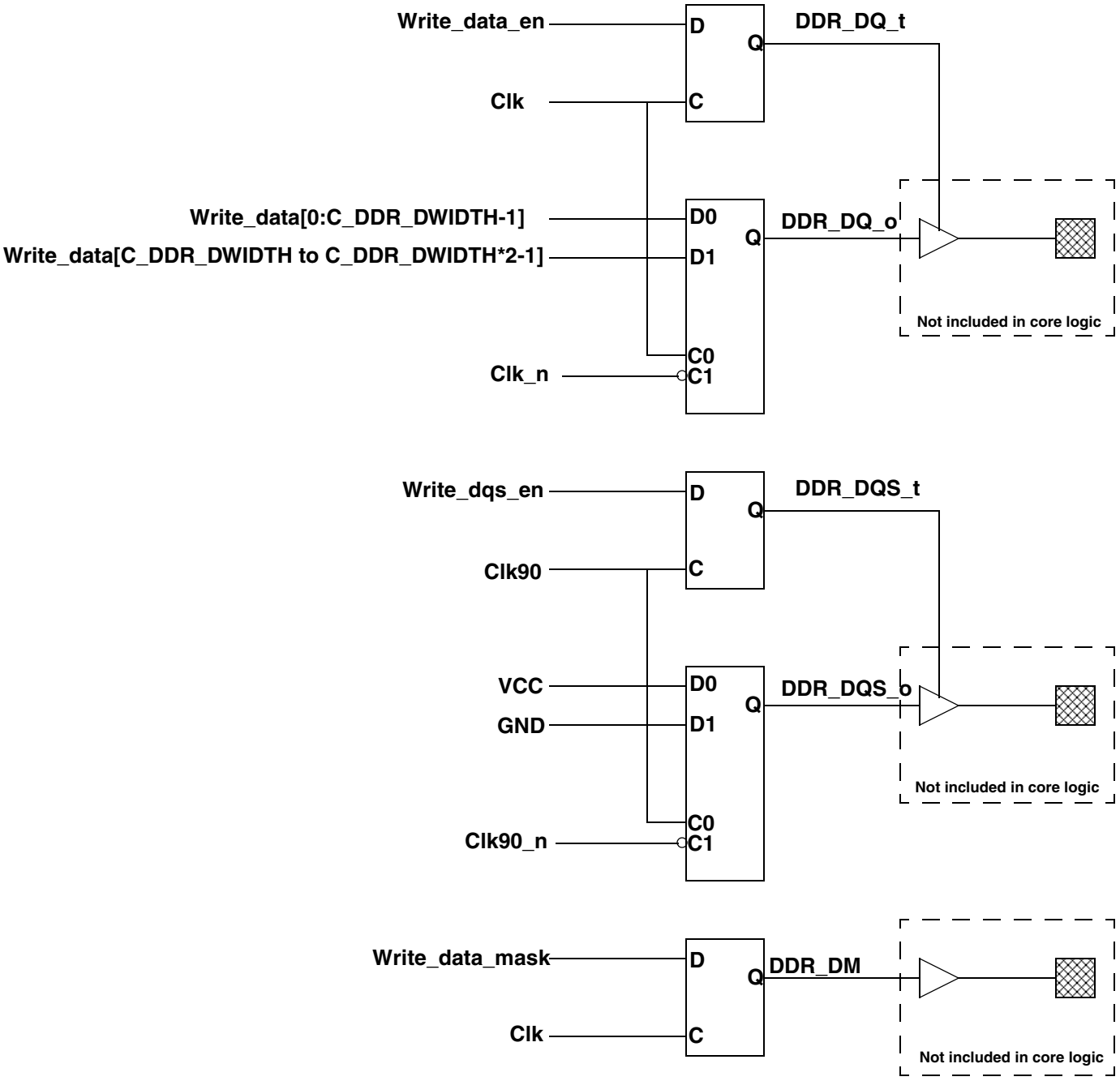
Control Signals

All control signals and the address bus to the DDR are registered in the IOBs of the FPGA.

Write Data

The DDR I/O registers are used to output the write data to the DDR as shown in Figure 20. Since the clock is being generated from the Clk90 output of the DCM, the Clk0 output is used to clock out the data so that the DDR clock is centered in the DDR data. This also allows a full clock period for the data to get to the IOBs.

DQS is generated from the Clk0 output so that it is centered in the data.



Note:
DDR_DQ_ECC (0:NUM_ECC_BITS-1), DDR_DQS_ECC, and DDR_DM_ECC generated in the same manner when C_INCLUDE_ECC_SUPPORT=1 and C_DDR_DWIDTH=32.

Figure 20: **Write Data Path**

Read Data

The DDR I/O registers are used to input data from the DDR as shown in Figure 21. The clock output to the DDR is used to clock the input data. This clock is input to a DCM and generates DDR_Clk90_in.

During a read cycle, the data strobe signal from the DDR (DDR_DQS) is registered on the rising edge only of DDR_Clk90_in so that it is always high while the DDR is transmitting data. This signal will be used by the Read Data Path logic as the write enable into a FIFO.

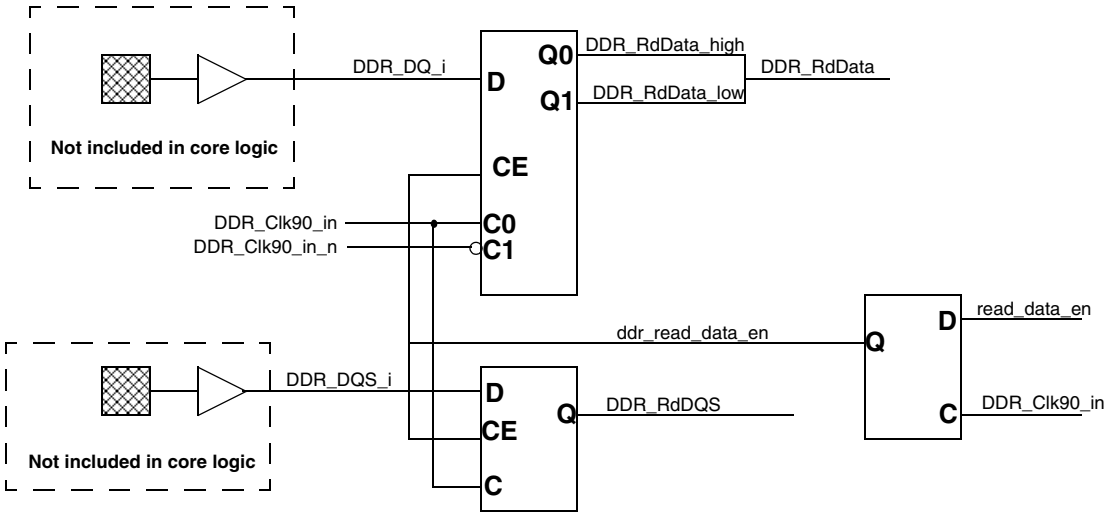


Figure 21: **DDR Input Data Registers**

Double Word (64-bit) Read Logic

As shown in Figure 22, the DW64 RD Logic contains logic to support reading of 64-bit data on each clock edge from the DDR SDRAM memory. It interfaces to the Read Data Path and IO Reg modules and will be instantiated in the design only when C_DDR_DWIDTH=64 and C_INCLUDE_ECC_SUPPORT=0.

The IO Reg module captures 128-bits of data, 64-bits on each clock edge. The Read Data Path module will only store the 64-bits of data as specified by Bus2IP_Addr(28).

Figure 22 shows the detail of the logic implementation for DW64 RD Logic.

A separate FIFO is created to store Bus2IP_Addr(28) so that this address bit is stored for use when the data is read from the DDR memory. Bus2IP_Addr(28) is written into this FIFO when IP2Bus_AddrAck is asserted to advance the address from the PLB IPIF. This address bit is read from the FIFO at the rising edge of the DDR_ReadDQS signal and when DDR_read_data_en is asserted. The Bus2IP_Addr(28) read from this FIFO selects the correct 64-bits of the 128-bits read from the DDR memory for input in the Read Data Path module.

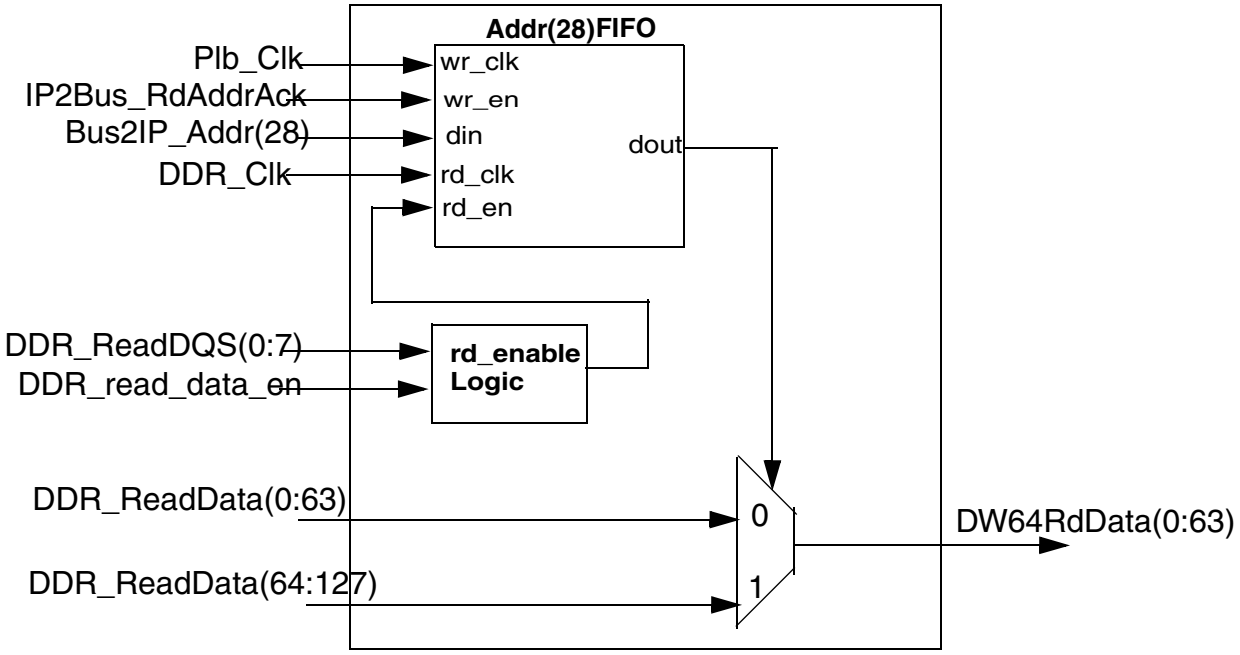


Figure 22: **DW64 RD Logic Detailed Design**

The signal description of the module is as below in Table 23.

Table 23: **DW64 RD Logic signal interface detail**

Signal Name	Interface	I/O	Description
DDR_ReadData(0:127)	IO Reg	I	128-bit data read from DDR memory
DDR_ReadDQS(0:7)	IO Reg	I	Captured DDR_DQS signal
DDR_read_data_en	IO Reg	I	Read data enable
IP2Bus_RdAddrAck	IPIC	I	Write enable of Addr(28)FIFO
Bus2IP_Addr(28)	IPIC	I	Specifies which 64-bit data (upper or lower) of the 128-bit data read from DDR memory to route to the Read Data Path module
PLB_Clk	PLB	I	Write clock for Addr(28)FIFO
DDR_CLK	DDR	I	Read clock for Addr(28)FIFO
DW64RdData[0:63]	Read Data Path	O	Data Out 64 bits

Read Data Path Logic

The Read Data Path logic consists of an asynchronous FIFO in which the DDR input data is written from the DDR_Clk90_in and read from the internal FPGA clock. The write enable to the FIFO is the DDR_RdDQS signal which will be high during DDR data transmission.

Once the FIFO is not empty, the data is read from the FIFO and a read acknowledge is generated. This is shown in Figure 23.

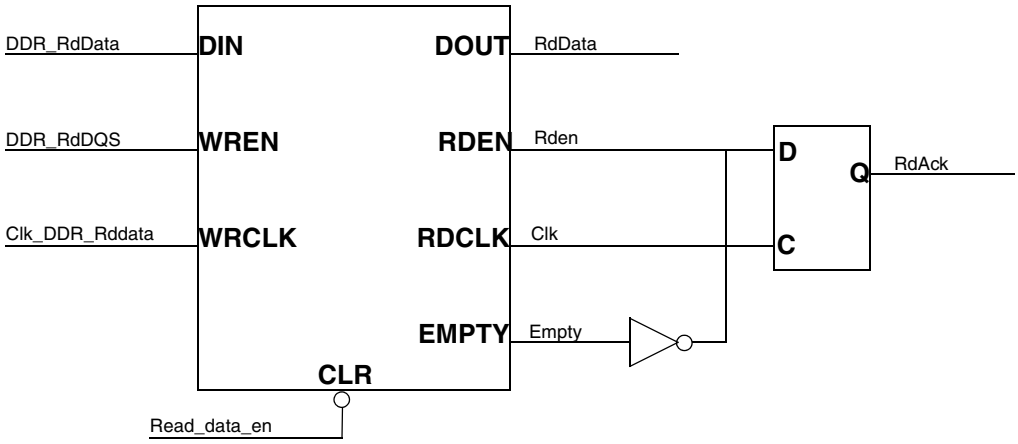


Figure 23: **Read Data Path**

DDR Clocking

Clock Generation

The clocking scheme required in the FGPA and used by the DDR controller core is shown in Figure 24 or Figure 25. An example implementation can be found in the DDR Clock Module Reference design available in the EDK Toolkit.

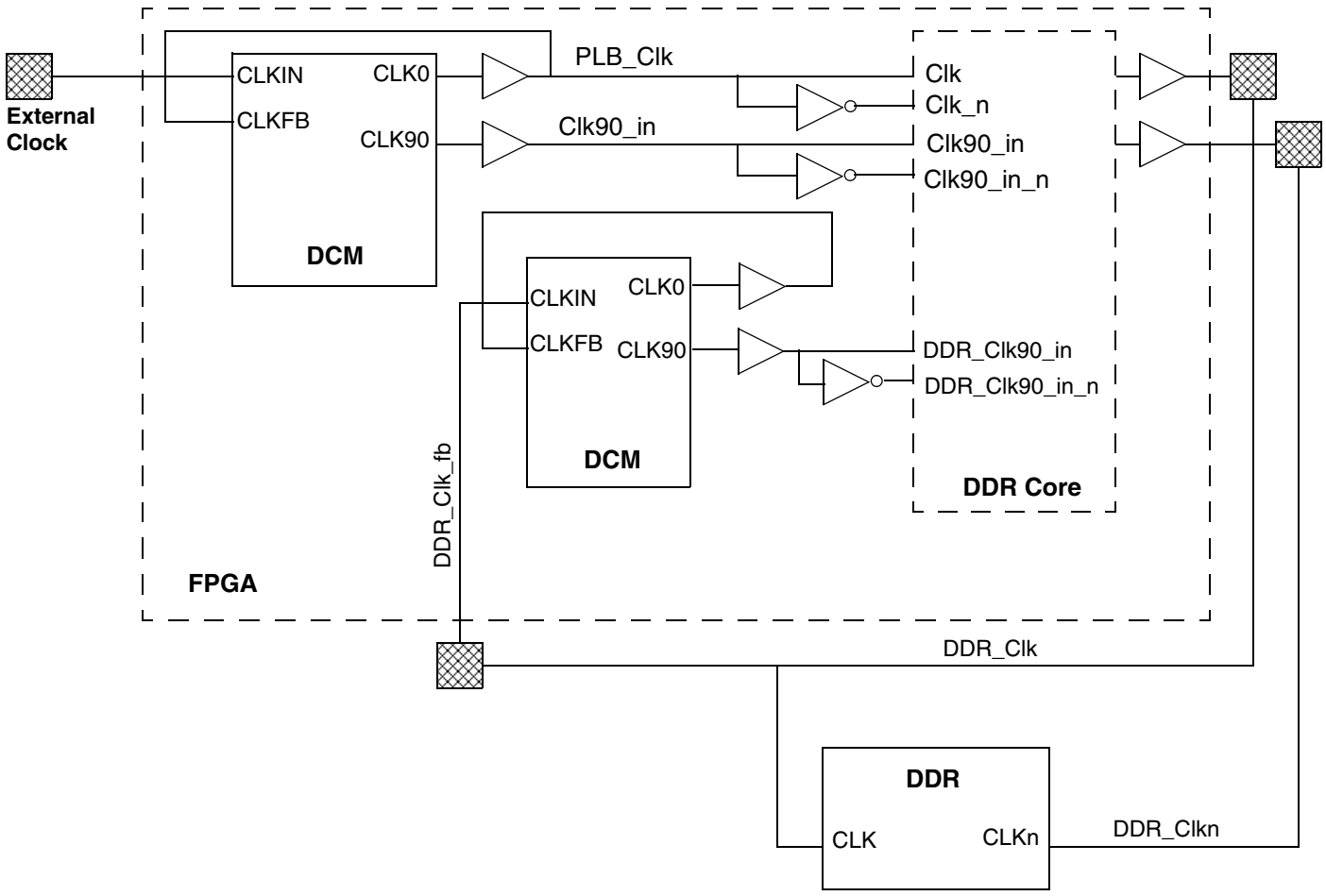


Figure 24: **DDR Clocking (Option 1)**

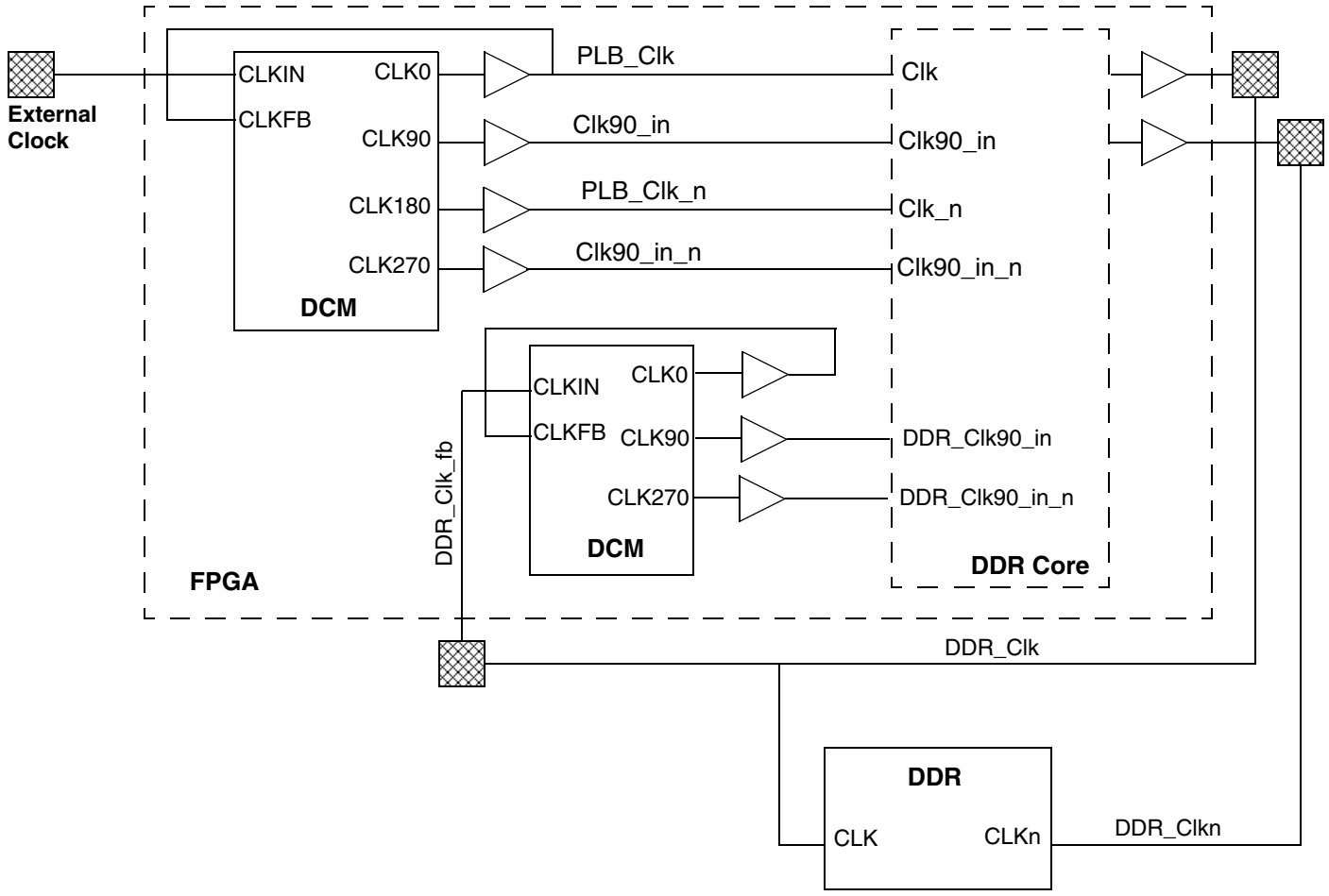


Figure 25: **DDR Clocking (Option 2)**

Clk and Clk90 Generation

A DCM is required to generate the clock used internal to the FPGA as shown in Figure 25 or Figure 26. A 90 degree phase output of the DCM is input to the DDR Controller core and is used to generate the DDR clock and DQS signals. PLB_Clk and Clk90_in are the outputs of a DCM and global buffers.

DDR Clock Generation

The clock output to the DDR SDRAMs is generated using the DDR I/O registers as shown in Figure 26. The 90 degree phase-shift clock is used to generate the DDR_Clk (and DDR_Clkn) so that the clock is centered in the data output to the DDR.

Note that the DDR clock frequency is the same as the PLB clock frequency.

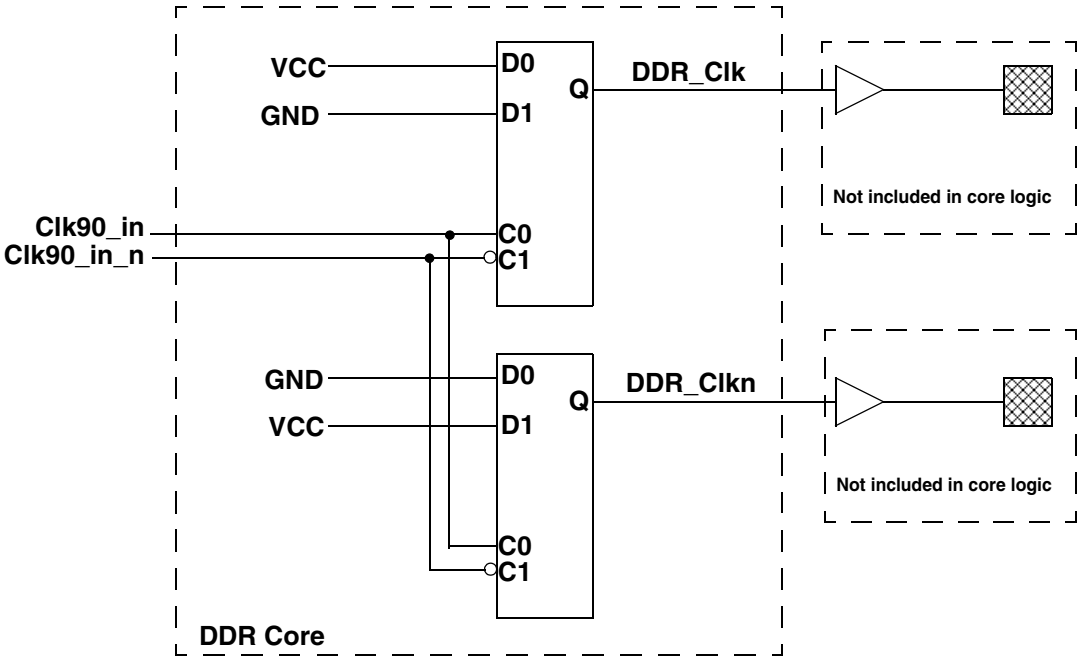


Figure 26: **DDR Clock Generation**

DDR Clock Input Synchronization

Another DCM will be required by this design to align the clock output to the DDR registers with the data from the DDR to accurately register this data. The DDR_Clk output shown in Figure 16 will need to be connected to the DDR_Clk_fb shown in Figure 25 as an external board connection. The Clk90 output of the DDR Clock DCM is input to the DDR Controller core and is used to clock in the DDR data.

Due to the variation in board layout, the DDR clock and the DDR data relationship can vary. Therefore, the designer should analyze the time delays of the system and set all of the attributes of the phase shift controls of the DCM as needed to insure stable clocking of the DDR data.

Timing Diagrams

The following diagrams illustrate the relationship between the PLB bus signals and the DDR memory signals for various transactions.

The timing diagrams shown in Figures 17-40 are listed below.

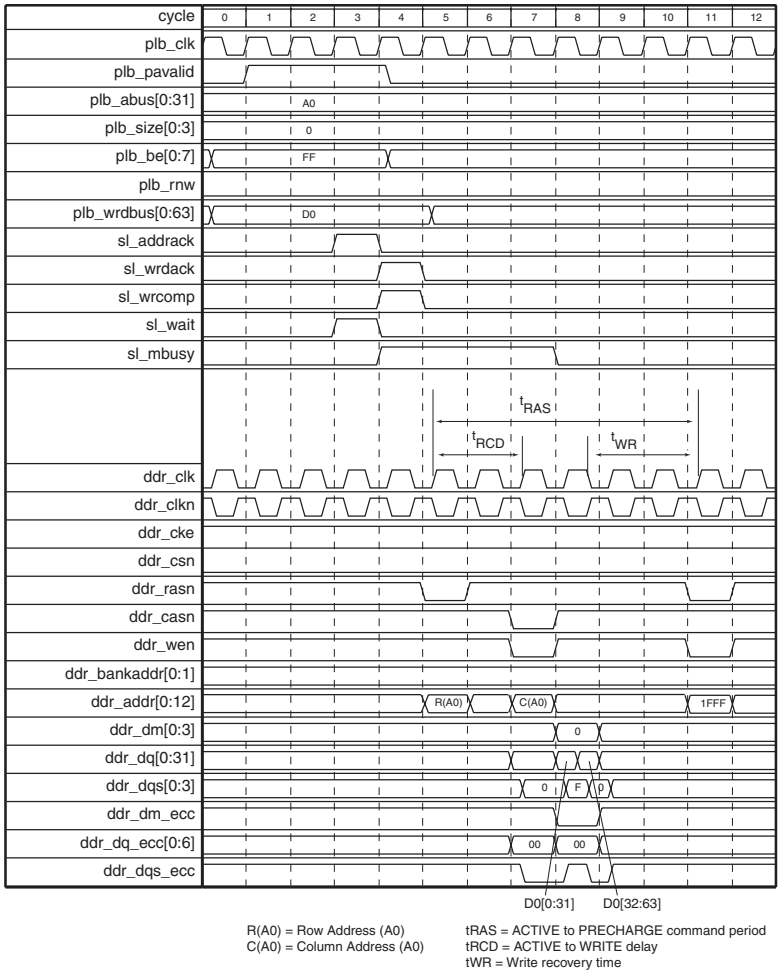
For 32-bit data width interface with DDR SDRAM:

1. Single doubleword write (Figure 27)
2. Single doubleword write with ECC (Figure 28)
3. Single doubleword read (Figure 29)
4. Single doubleword read with ECC (Figure 30)
5. Single word write (Figure 31)
6. Single word read (Figure 32)

7. 4 word cachline write (Figure 33)
8. 4 word cachline write with ECC (Figure 34)
9. 4 word cachline read (Figure 35)
10. 4 word cachline read with ECC (Figure 36)
11. Write burst of 4 doublewords (Figure 37)
12. Read burst of 4 doublewords (Figure 38)
13. Indeterminate write burst of 5 doublewords (Figure 39)
14. indeterminate read burst of 5 doublewords (Figure 40)

For 64-bit data width interface with DDR SDRAM:

1. Single doubleword write (Figure 41)
2. Single doubleword read (Figure 42)
3. Single word write (Figure 43)
4. Single word read (Figure 44)
5. Cacheline (4 word) write (Figure 45)
6. Cacheline (4 word) read (Figure 46)
7. Write burst (4 doublewords-fixed burst) (Figure 47)
8. Read burst (4 doublewords-fixed burst) (Figure 48)
9. Indeterminate write burst (5 doublewords) (Figure 49)
10. Indeterminate read burst (5 doublewords) (Figure 50)



Latency Calculations:

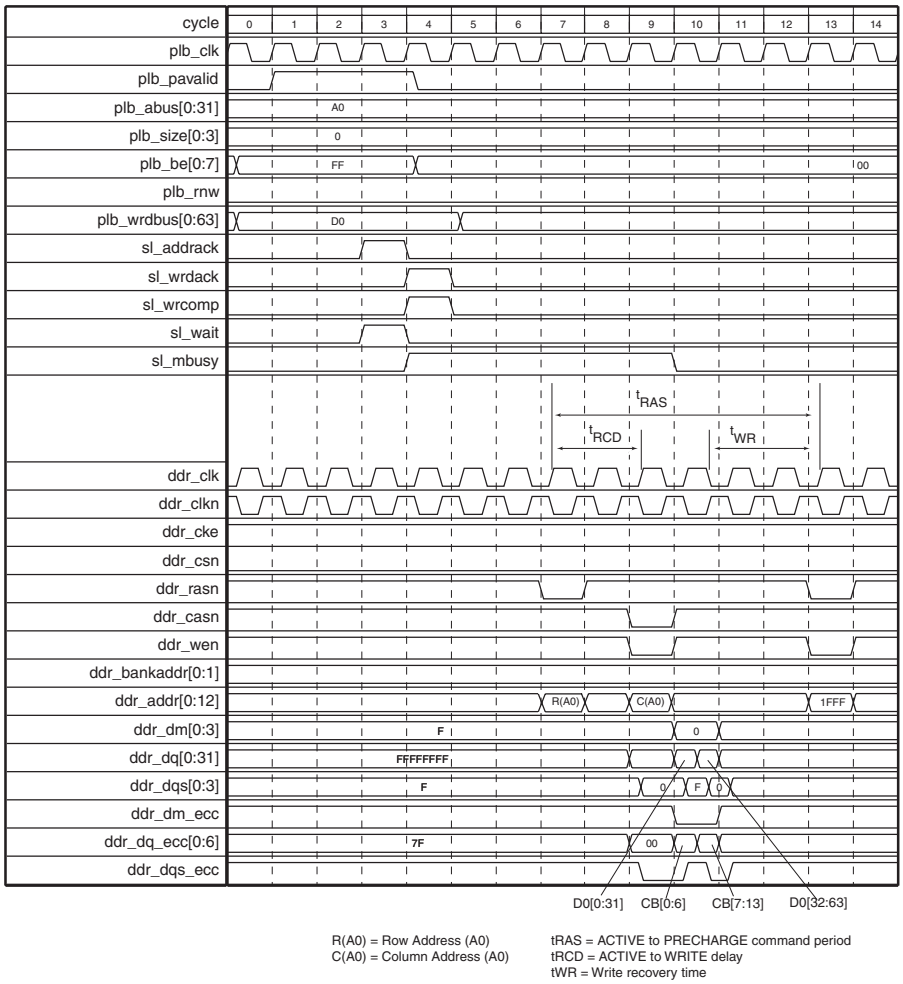
plb_PAValid to plb_AddrAck
= 3 plb_clk cycles

plb_AddrAck to plb_WrAck
= 1 plb_clk cycle

plb_AddrAck to next plb_AddrAck
= 5 plb_clk cycles

Note: when ECC is enabled, C_INCLUDE_ECC_SUPPORT=1, but ECC write is turned off in ECCCR, then zeros will be written as the ECC check bits into memory.

Figure 27: **Single Doubleword Write Transaction for (C_DDR_DWIDTH=32)**



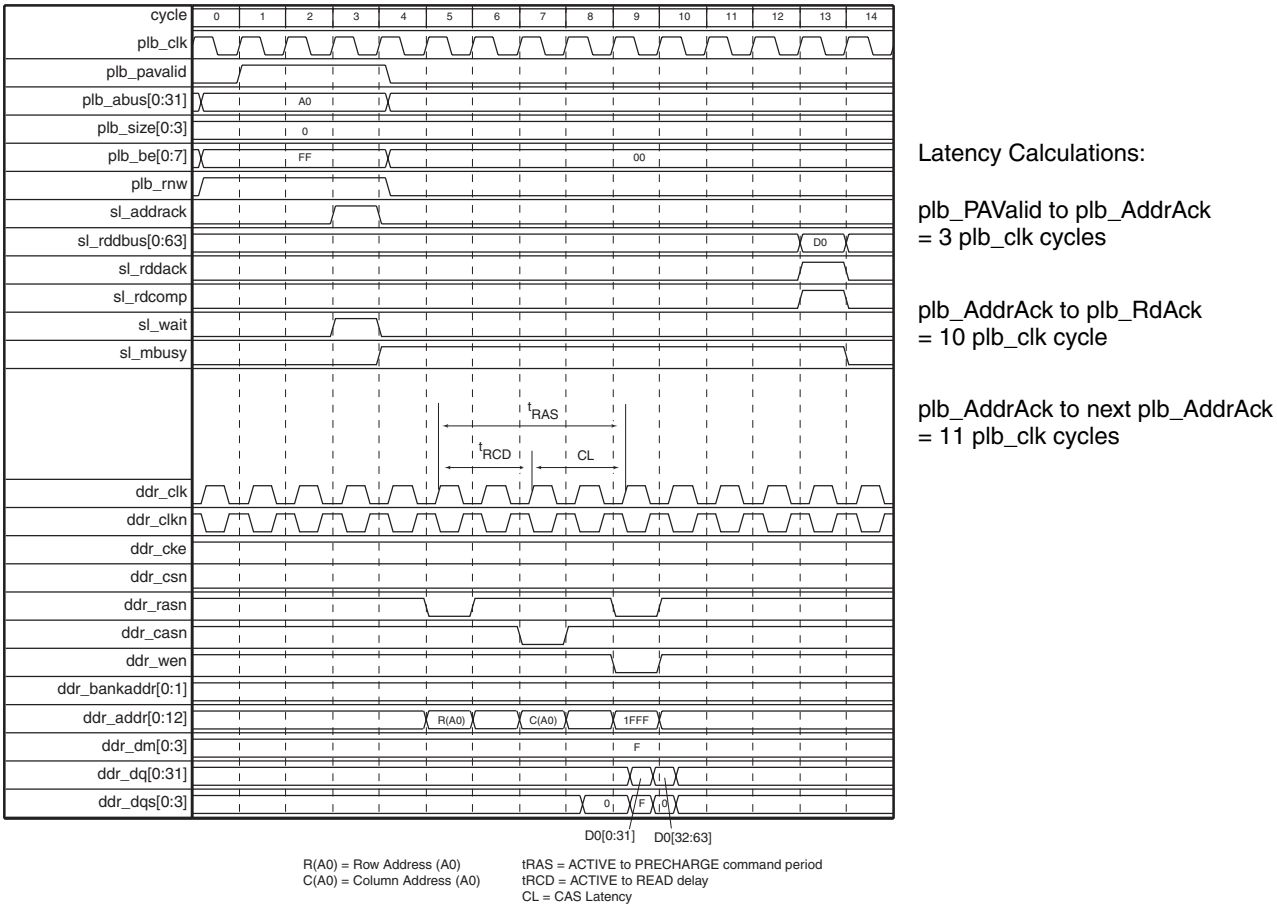
atency Calculations:

b_PAValid to plb_AddrAck
3 plb_clk cycles

b_AddrAck to plb_WrAck
1 plb_clk cycle

b_AddrAck to next plb_AddrAck
7 plb_clk cycles

Figure 28: **Single Doubleword Write with ECC Transaction (C_DDR_DWIDTH=32)**



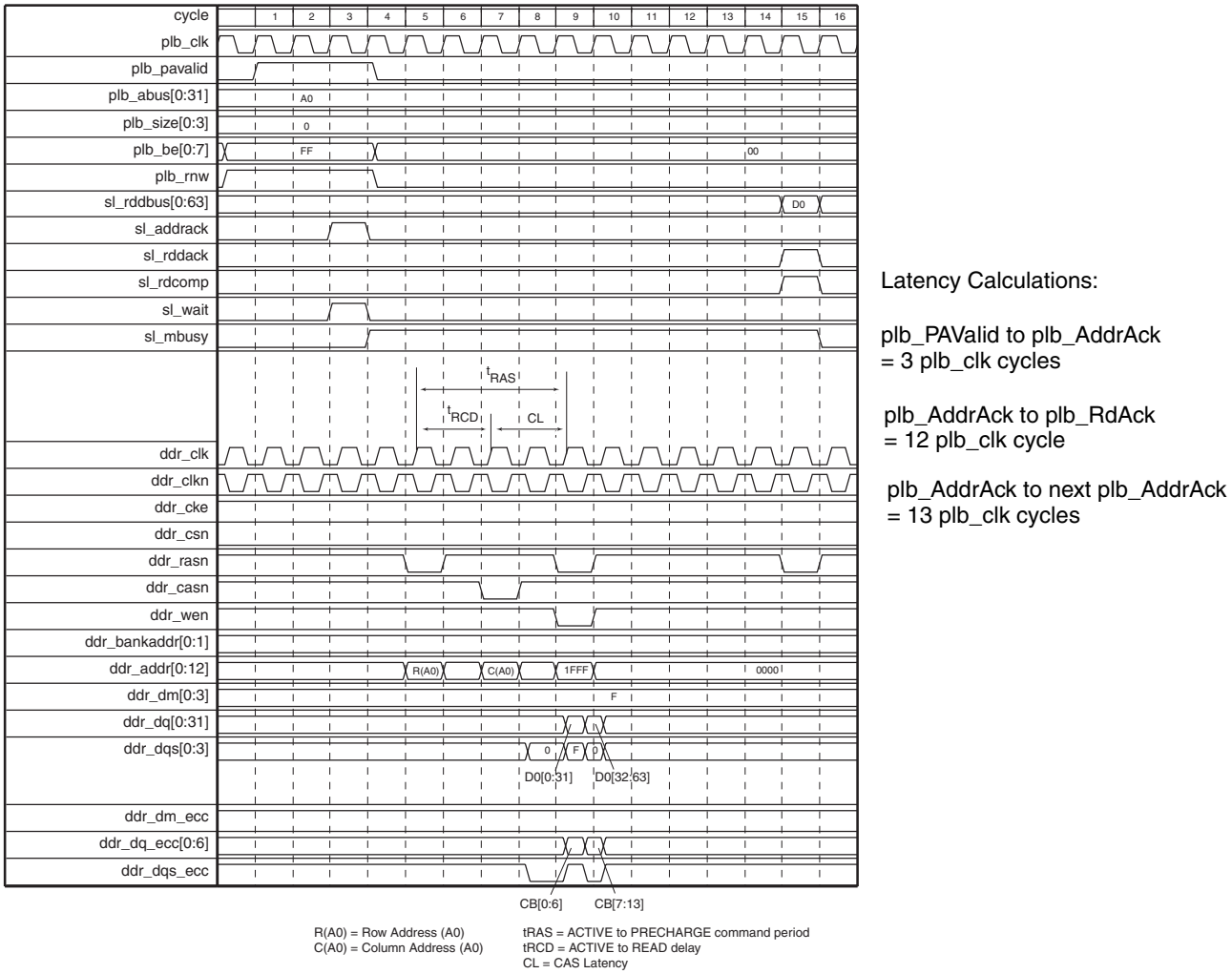
Latency Calculations:

plb_PAValid to plb_AddrAck
= 3 plb_clk cycles

plb_AddrAck to plb_RdAck
= 10 plb_clk cycle

plb_AddrAck to next plb_AddrAck
= 11 plb_clk cycles

Figure 29: **Single Doubleword Read Transaction (C_DDR_DWIDTH=32)**



Latency Calculations:

plb_PAValid to plb_AddrAck
= 3 plb_clk cycles

plb_AddrAck to plb_RdAck
= 12 plb_clk cycle

plb_AddrAck to next plb_AddrAck
= 13 plb_clk cycles

Figure 30: **Single Doubleword Read with ECC Transaction (C_DDR_DWIDTH=32)**

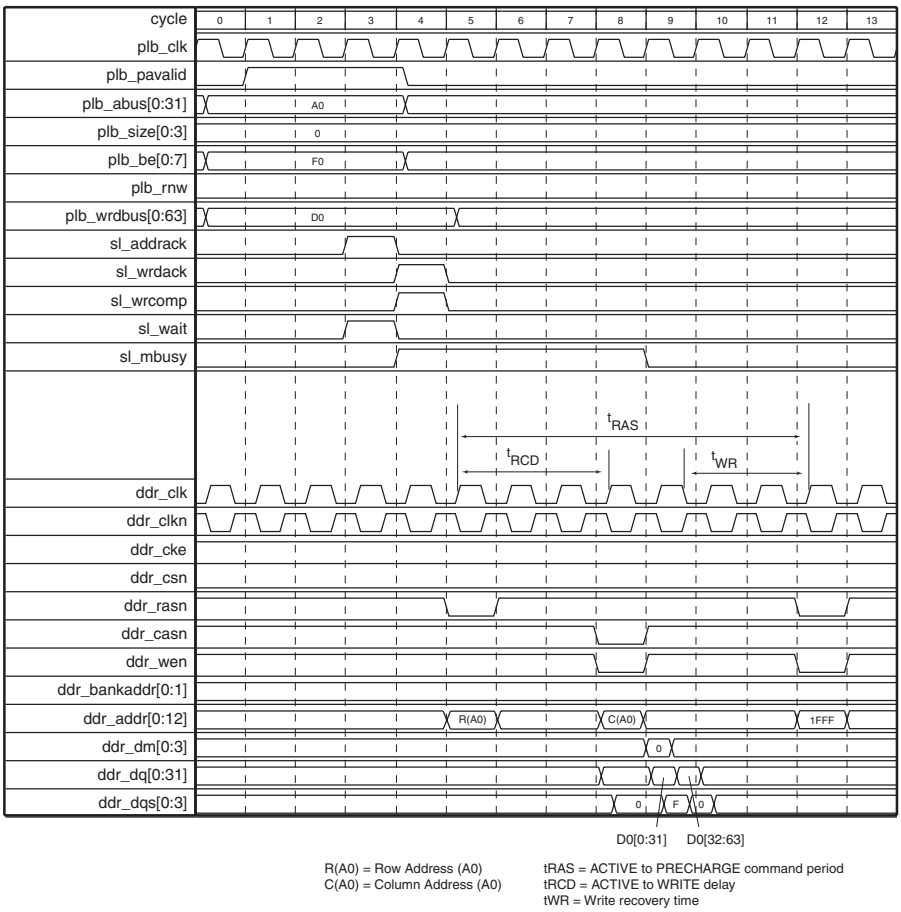


Figure 31: **Single Word Write Transaction (C_DDR_DWIDTH=32)**

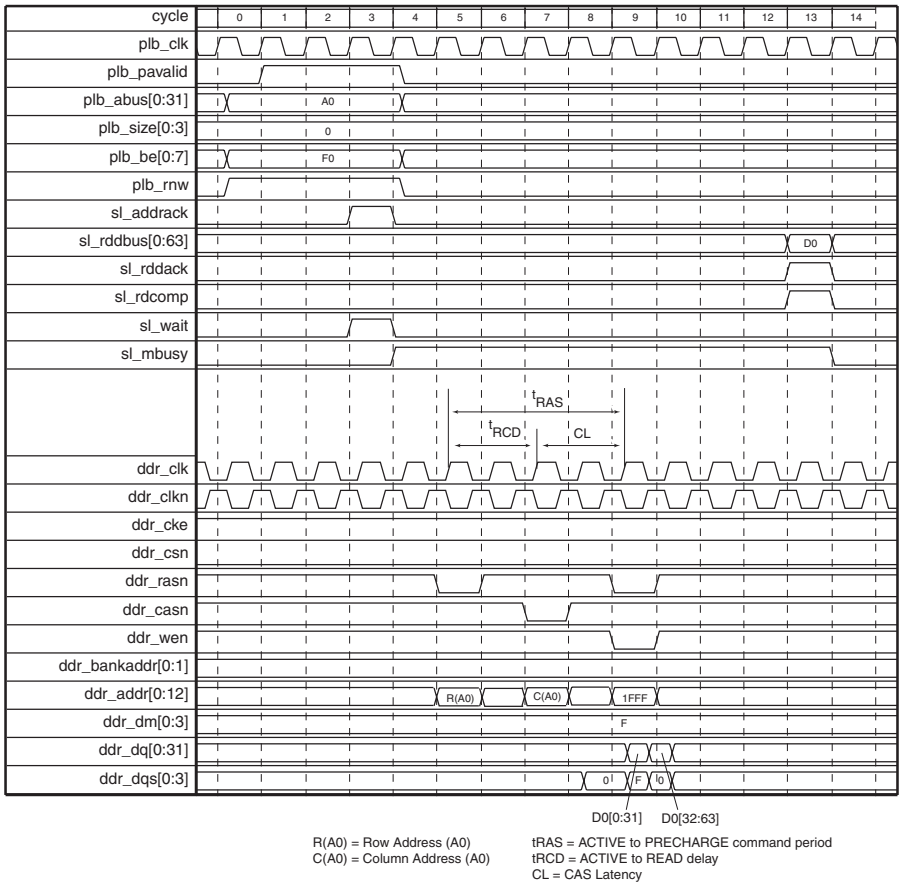
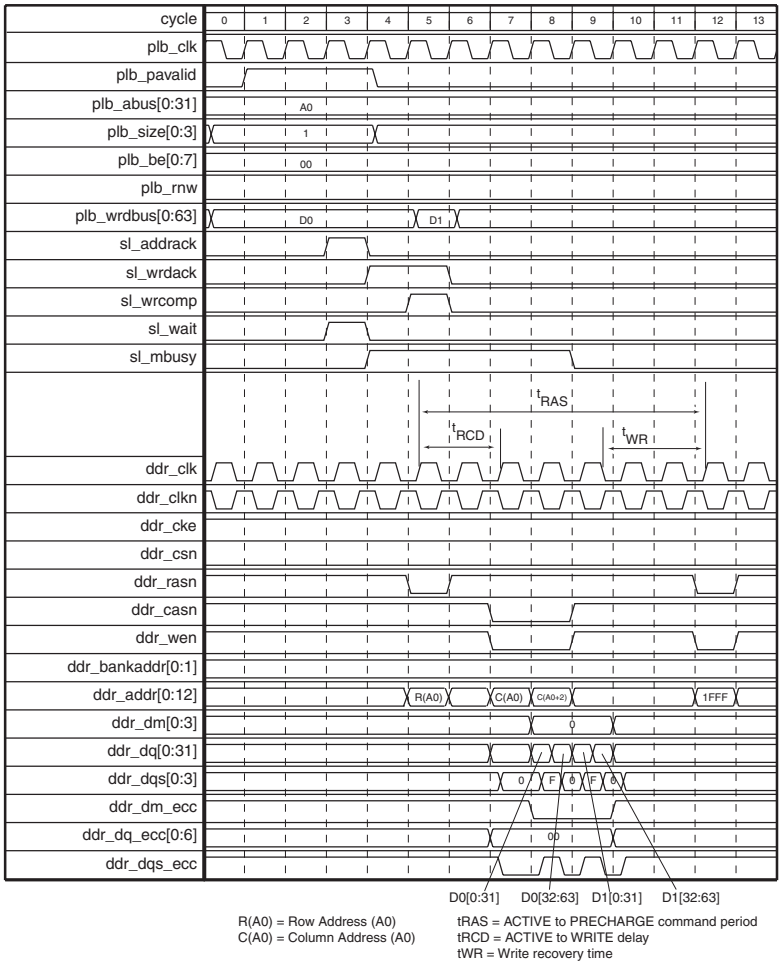


Figure 32: **Single Word Read Transaction (C_DDR_DWIDTH=32)**



Latency Calculations:

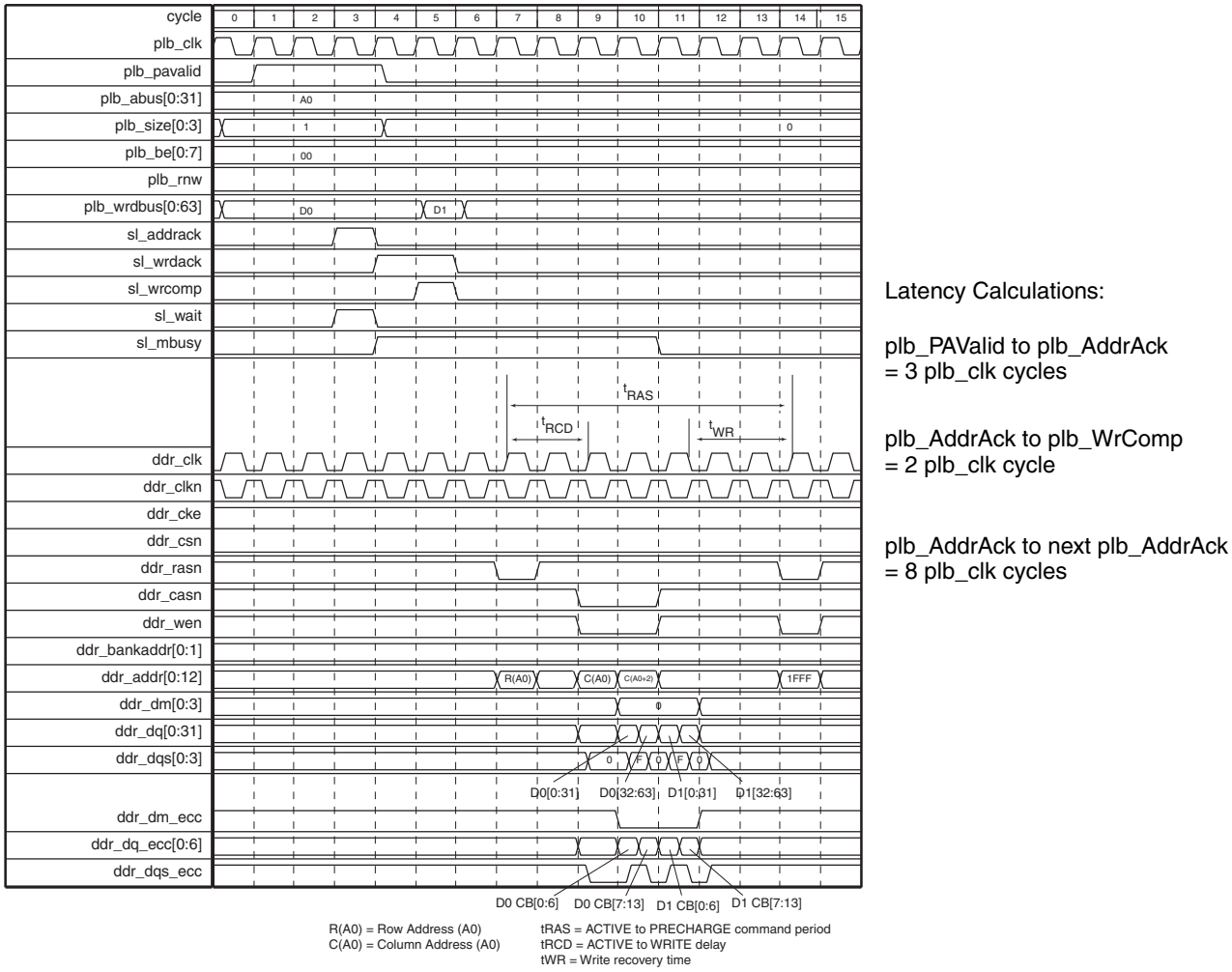
plb_PAValid to plb_AddrAck
= 3 plb_clk cycles

plb_AddrAck to plb_WrComp
= 2 plb_clk cycle

plb_AddrAck to next plb_AddrAck
= 6 plb_clk cycles

Note: when ECC is enabled, C_INCLUDE_ECC_SUPPORT=1, but ECC write is turned off in ECCCR, then zeros will be written as the ECC check bits into memory.

Figure 33: **Cacheline (4 word) Write Transaction (C_DDR_DWIDTH=32)**



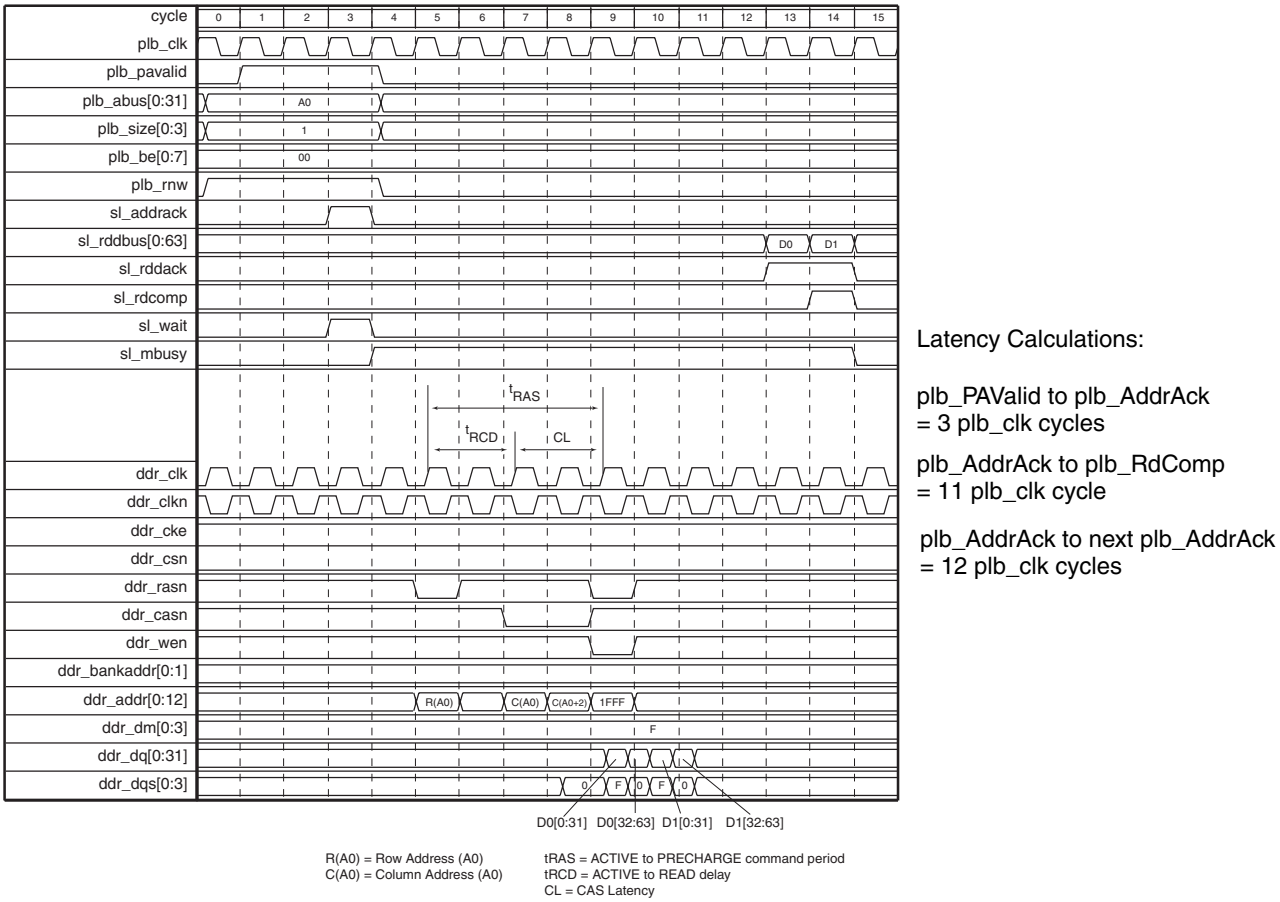
Latency Calculations:

plb_PAValid to plb_AddrAck
= 3 plb_clk cycles

plb_AddrAck to plb_WrComp
= 2 plb_clk cycle

plb_AddrAck to next plb_AddrAck
= 8 plb_clk cycles

Figure 34: **Cacheline (4 word) Write with ECC Transaction (C_DDR_DWIDTH=32)**



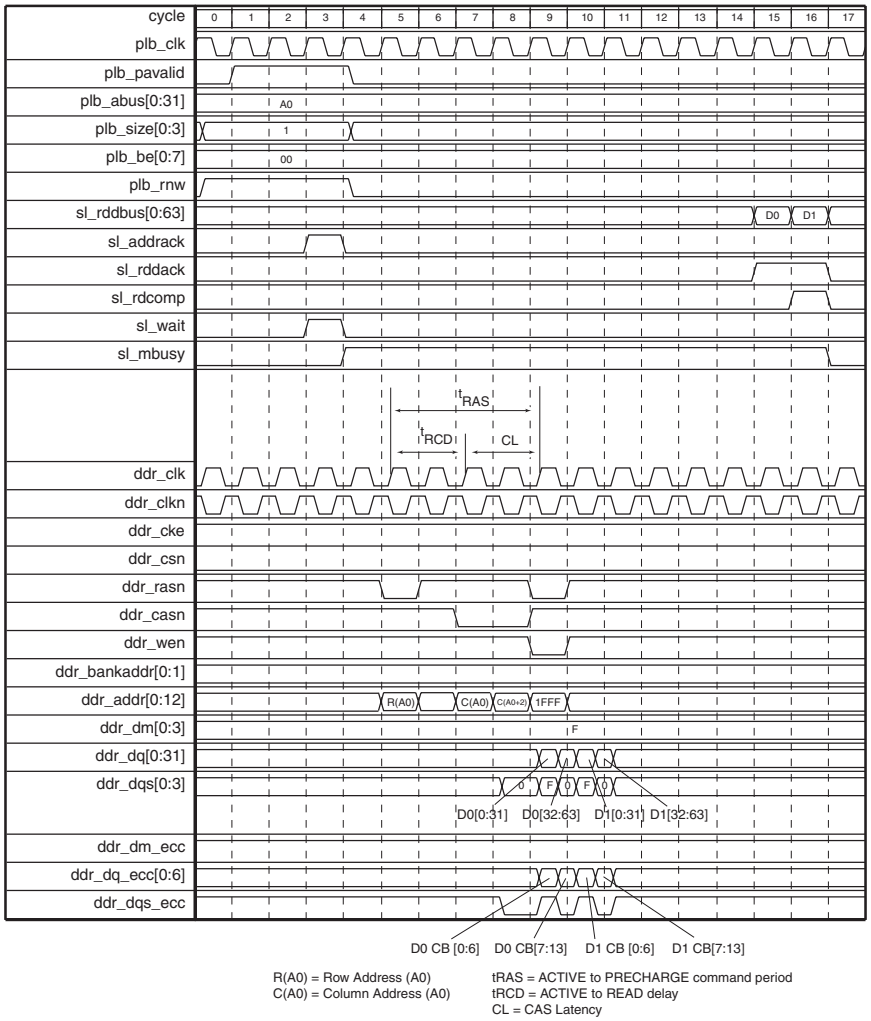
Latency Calculations:

plb_PAValid to plb_AddrAck
= 3 plb_clk cycles

plb_AddrAck to plb_RdComp
= 11 plb_clk cycle

plb_AddrAck to next plb_AddrAck
= 12 plb_clk cycles

Figure 35: **Cacheline (4 word) Read Transaction (C_DDR_DWIDTH=32)**



Latency Calculations:

plb_PAValid to plb_AddrAck
= 3 plb_clk cycles

plb_AddrAck to plb_RdComp
= 13 plb_clk cycle

plb_AddrAck to next plb_AddrAck
= 14 plb_clk cycles

Figure 36: **Cacheline (4 word) Read with ECC Transaction (C_DDR_DWIDTH=32)**

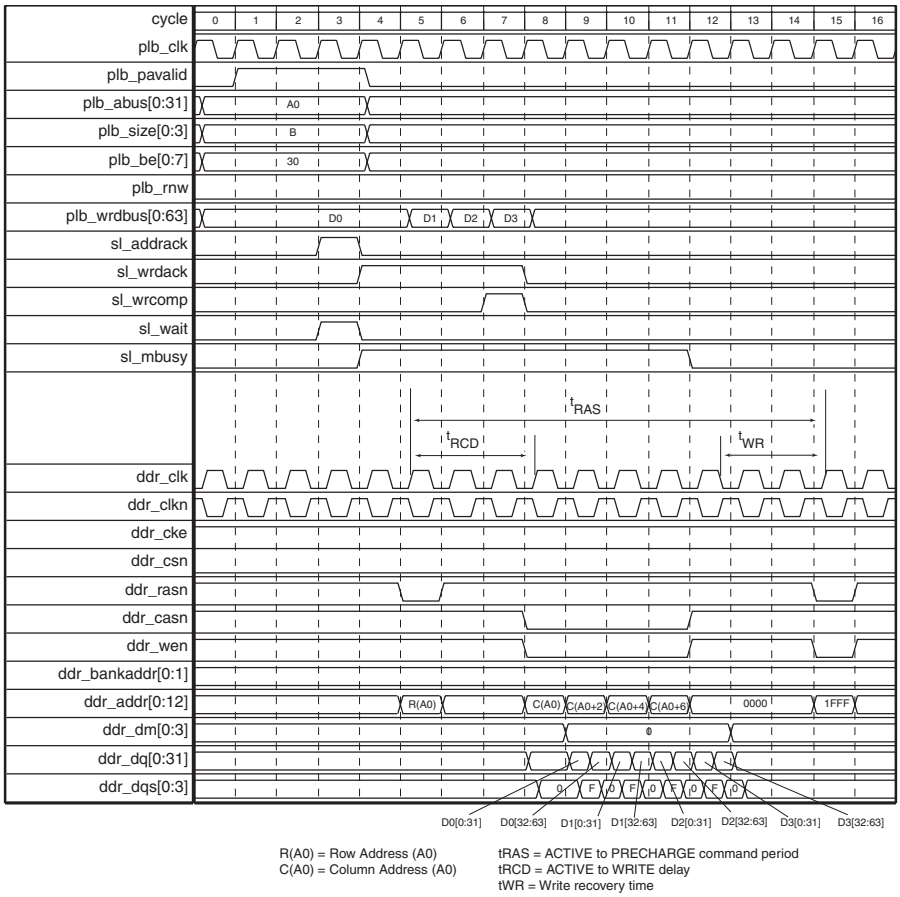


Figure 37: **Write Burst (4 doublewords) Transaction (C_DDR_DWIDTH=32)**

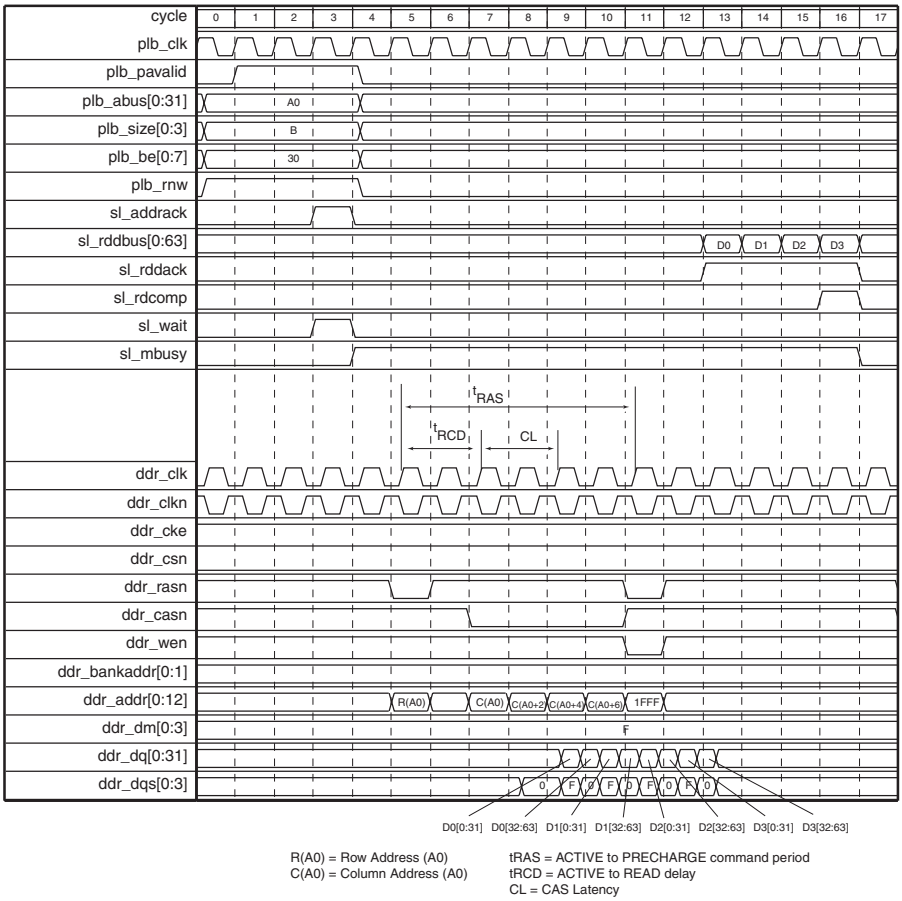


Figure 38: **Read Burst (4 doublewords) Transaction (C_DDR_DWIDTH=32)**

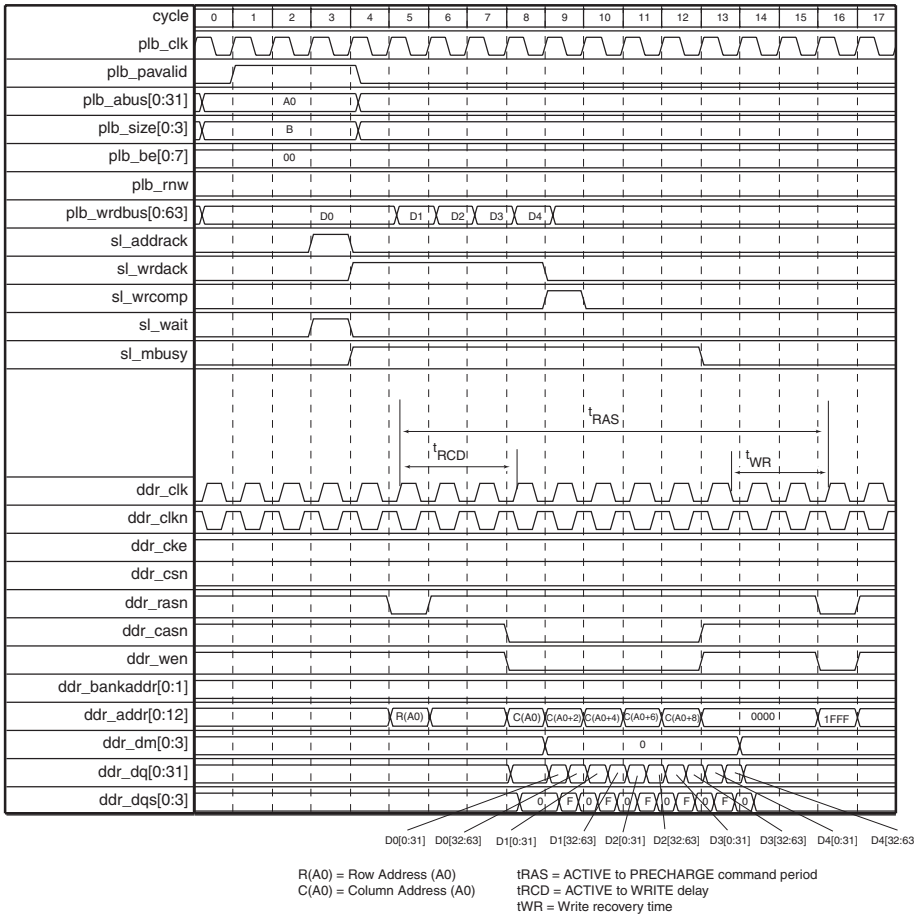


Figure 39: **Indeterminate Write Burst (5 doublewords) Transaction (C_DDR_DWIDTH=32)**

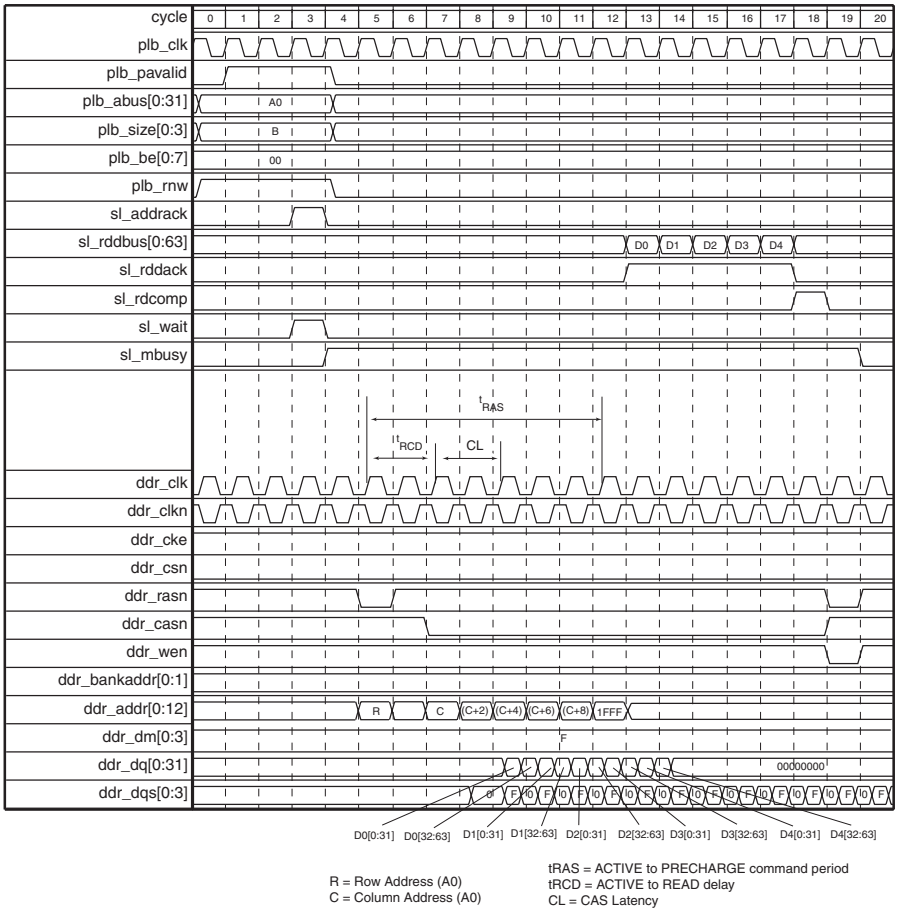


Figure 40: **Interdeterminate Read Burst (5 doublewords) Transaction (C_DDR_DWIDTH=32)**

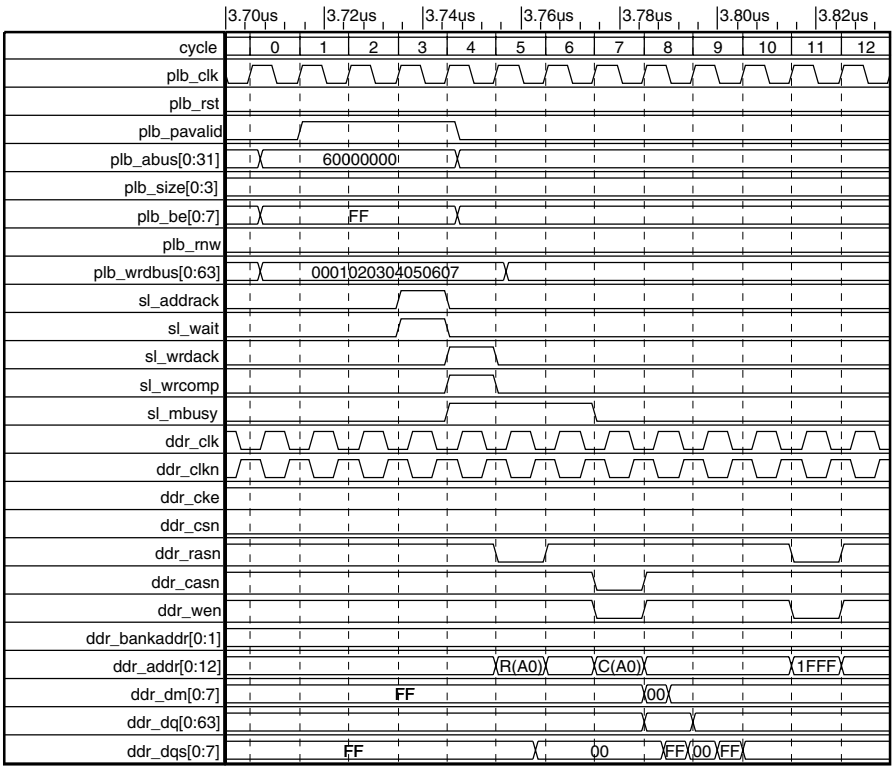


Figure 41: **Single Doubleword Write Transaction (C_DDR_DWIDTH=64)**

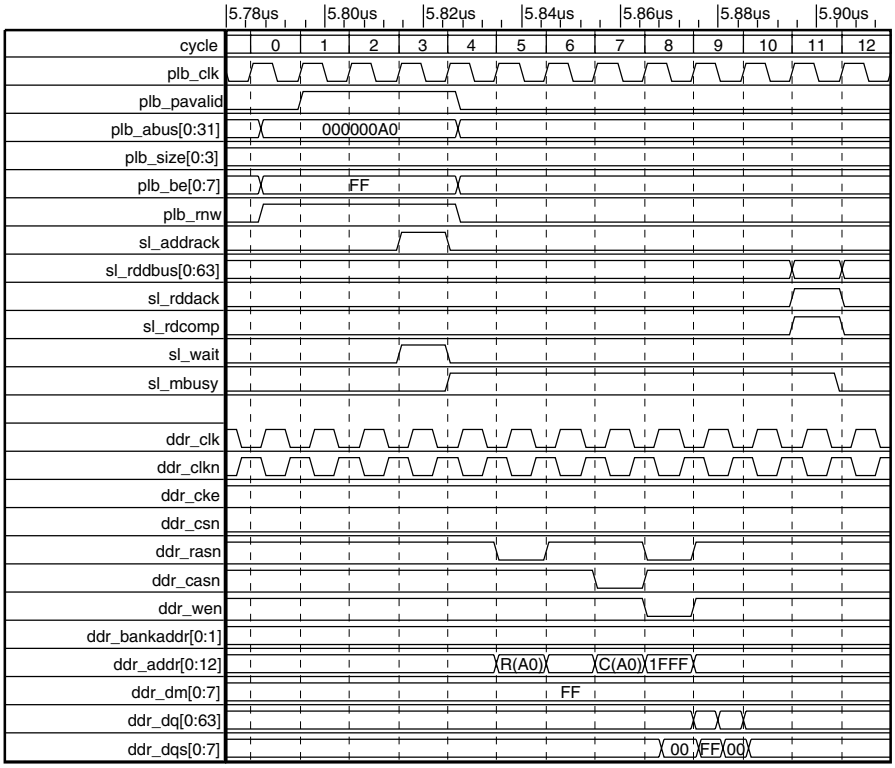


Figure 42: **Single Doubleword Read Transaction (C_DDR_DWIDTH=64)**

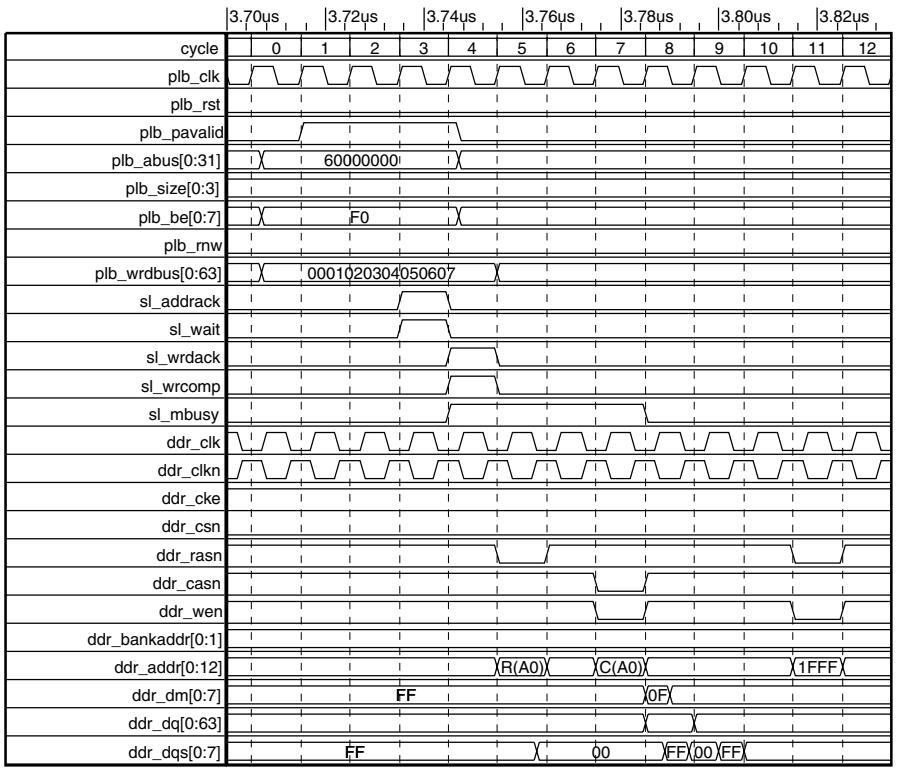


Figure 43: **Single Word Write Transaction (C_DDR_DWIDTH=64)**

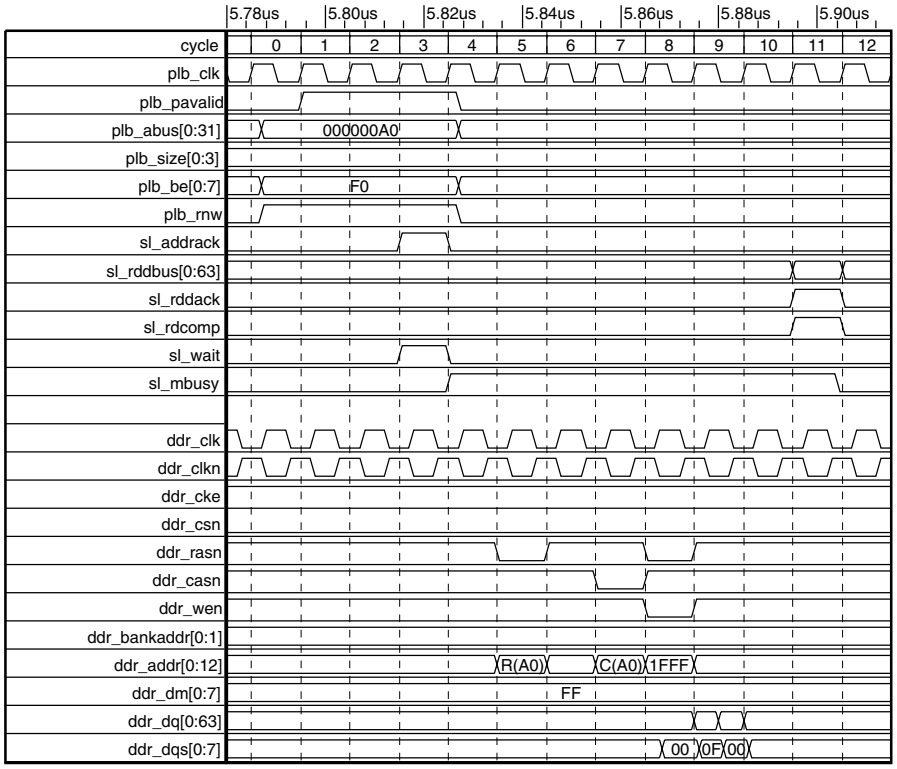


Figure 44: **Single Word Read transaction (C_DDR_DWIDTH=64)**

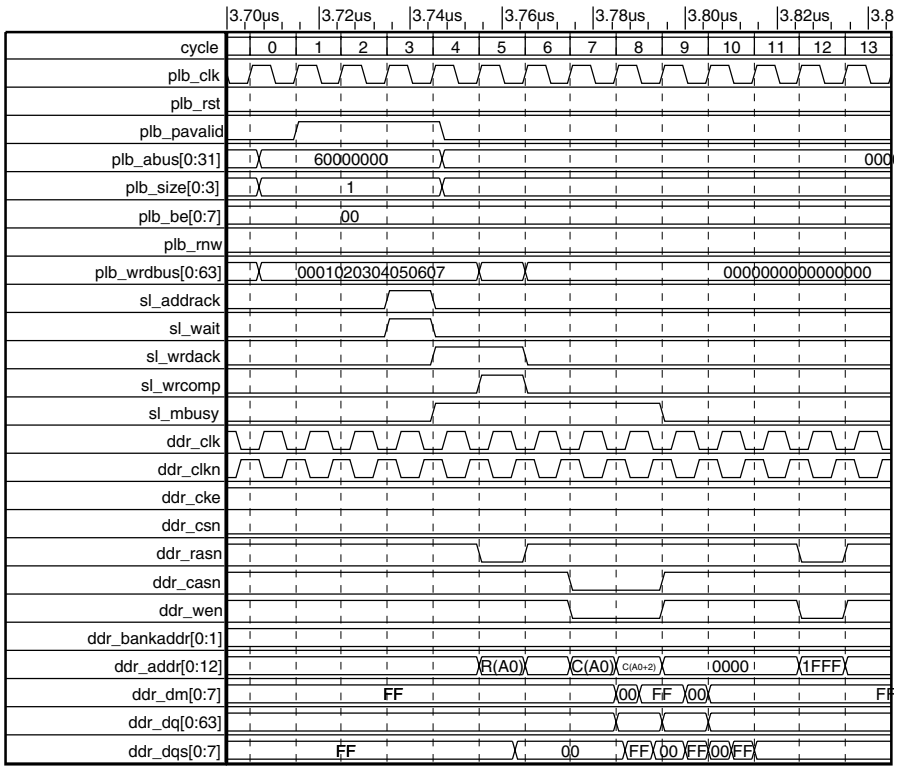


Figure 45: **Cacheline (4 word) Write Transaction (C_DDR_DWIDTH=64)**

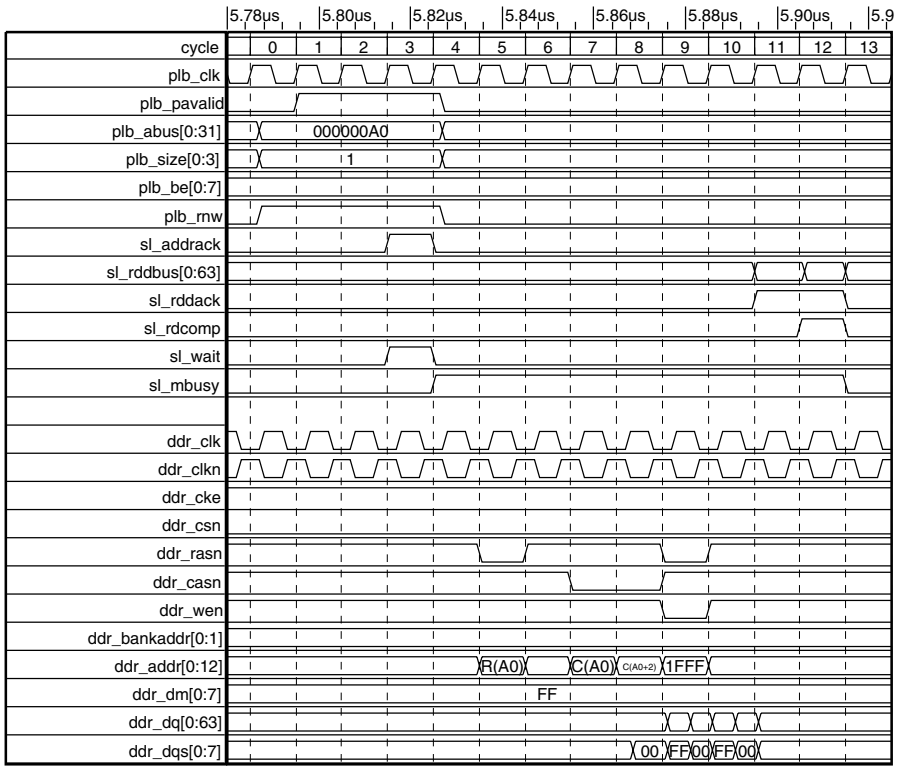


Figure 46: **Cacheline (4 Word) Read Transaction (C_DDR_DWIDTH=64)**

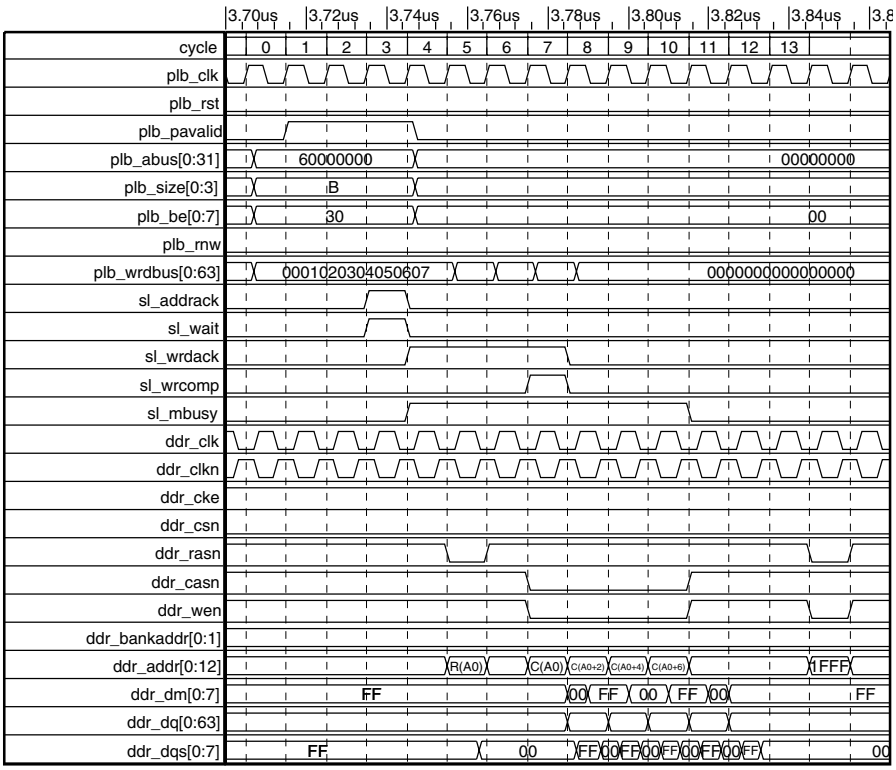


Figure 47: **Write Burst (4 Doublewords) Transaction (C_DDR_DWIDTH=64)**

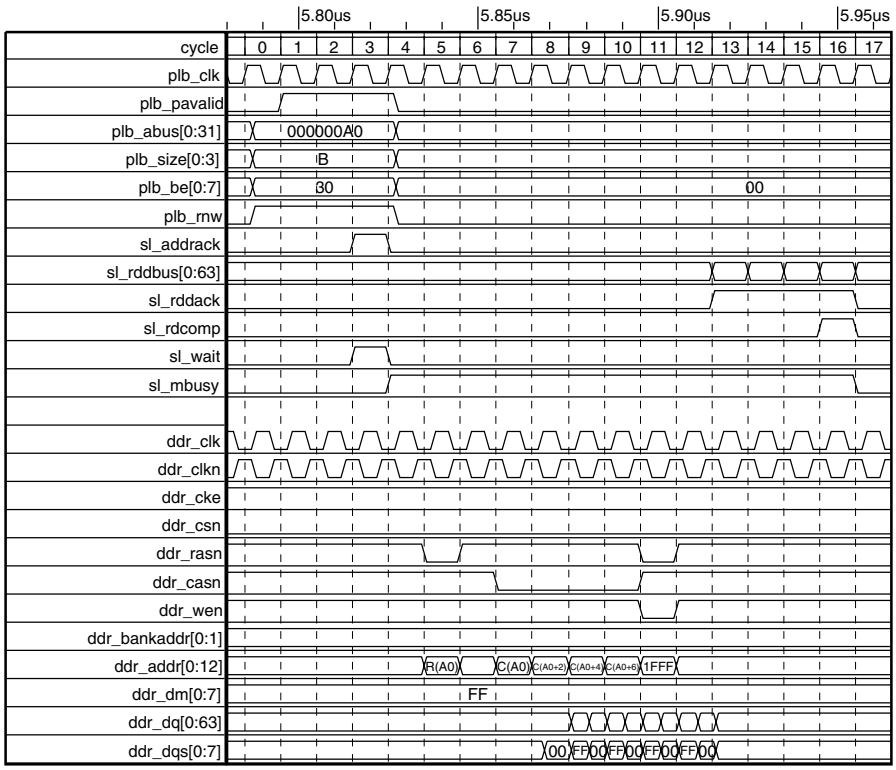


Figure 48: **Read Burst (4 Doublewords-fixed burst) Transaction (C_DDR_DWIDTH=64)**

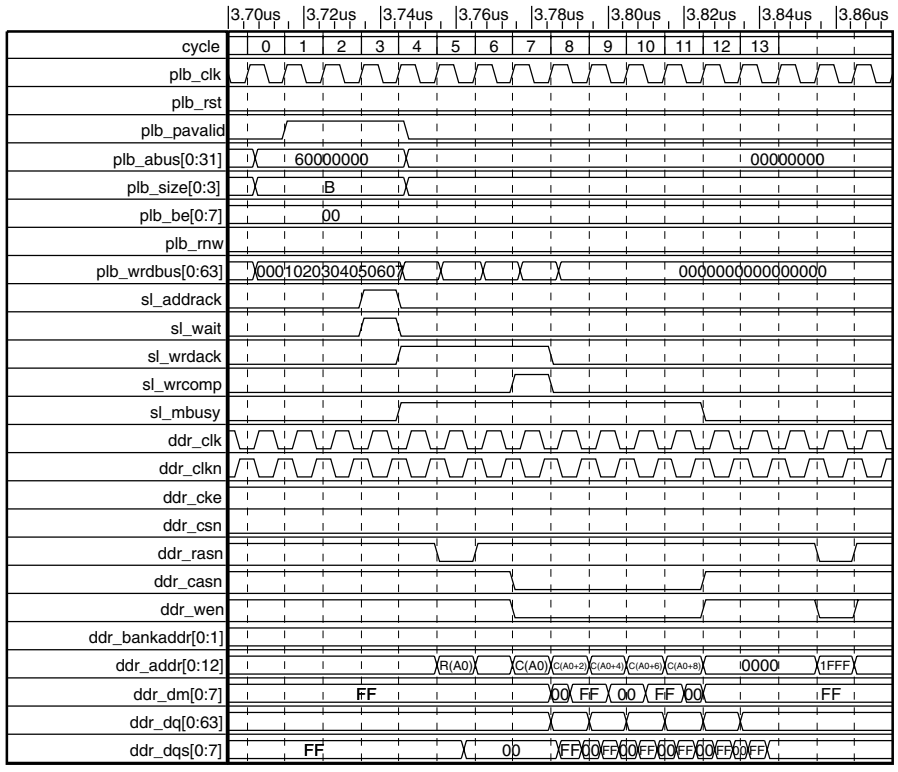


Figure 49: **Indeterminate Write burst (5 double words) Transaction (C_DDR_DWIDTH=64)**

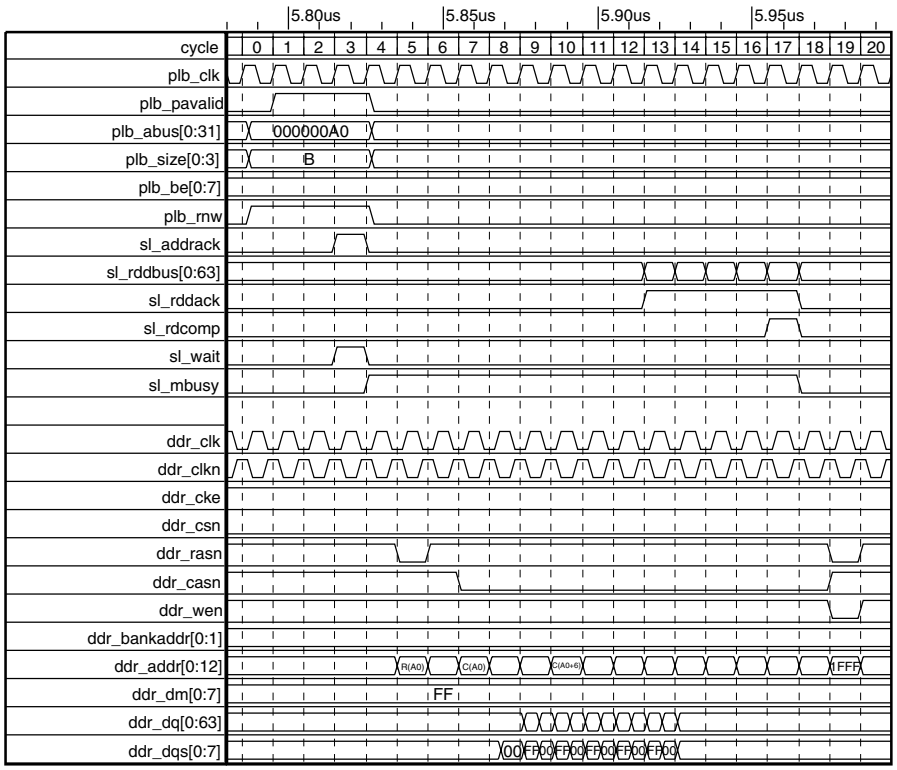


Figure 50: **Indeterminate Read Burst (5 Doublewords) Transaction (C_DDR_DWIDTH=64)**

Design Constraints

Note: An example UCF for this core is available and must be modified for use in the system. Please refer to the *EDK Getting Started Guide* for the location of this file.

Timing Constraints

A timing constraint should be placed on the system clock, setting the frequency to meet the bus timing requirements. A timing constraint should also be placed on the DDR feedback clock to set the frequency of this clock. An example is shown in Figure 51.

```
NET "PLB_Clk" TNM_NET = "PLB_Clk";
TIMESPEC "TS_PLB_Clk" = PERIOD "PLB_Clk" 7 ns HIGH 50 %;

NET "Clk90_in" TNM_NET = "Clk90_in";
TIMESPEC "TS_CLK90" = PERIOD "Clk90_in"  7 ns HIGH 50% ;

TIMESPEC "TSCLK2CLK90" = FROM "PLB_Clk" TO "Clk90_in" 2 ns;

NET "DDR_Clk90_in" TNM_NET = "DDR_Clk90_in";
TIMESPEC "TS_DDR_Clk90_in" = PERIOD "DDR_Clk90_in" 7 ns HIGH 50 %;
```

Figure 51: **DDR Timing Constraints**

Pin Constraints

The DDR I/O should be set to the SSTL2 I/O standard. If external pullups/pulldowns are not available on the DDR DQ and DQS signals, then these pins should be specified to use pullup or pulldown resistors. Pulldown resistors are preferred. An example is shown in Figure 52.

```
NET "DDR_DQS<0>"    IOSTANDARD=SSTL2_I;
NET "DDR_DQS<0>"    PULLDOWN;
NET "DDR_DQS<1>"    IOSTANDARD=SSTL2_I;
NET "DDR_DQS<1>"    PULLDOWN;
```

Figure 52: **DDR Pin Constraints**

Design Implementation

Target Technology

The intended target technology is a Virtex family FPGAs.

Device Utilization and Performance Benchmarks

The PLB DDR SDRAM Controller is a module that will be used with other design pieces in the FPGA, the utilization and timing numbers reported in this section are estimates. As the PLB DDR SDRAM Controller is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing of the PLB DDR SDRAM Controller design will vary from the results reported here.

The DDR SDRAM Controller benchmarks are shown in Table 24 for a Virtex-II Pro FPGA.

Table 24: **DDR FPGA Performance and Resource Utilization Benchmarks (Virtex-II Pro)**

Parameter Values (other parameters at default values)								Device Resources			f_{MAX} (MHz)
C_INCLUDE_BURST_CACHELN_SUPPORT	C_REG_DIMM	C_NUM_BANKS_MEM	C_INCLUDE_ECC_SUPPORT	C_ENABLE_ECC_REG	C_INCLUDE_ECC_INTR	C_INCLUDE_ECC_TEST	C_DDR_DWIDTH	Slices	Slice Flip-Flops	4-input LUTs	f_{MAX}
1	0	1	0	0	0	0	32	872	848	910	148
1	0	2	0	0	0	0	32	862	847	893	147
1	0	4	0	0	0	0	32	872	855	902	149
0	0	1	0	0	0	0	32	733	799	677	146
1	1	1	0	0	0	0	32	901	938	890	146
1	1	2	0	0	0	0	32	906	940	888	146
0	1	1	0	0	0	0	32	769	889	672	146
1	0	1	1	0	0	0	32	1402	1343	1676	144
0	0	1	1	0	0	0	32	1258	1293	1446	143
1	0	1	1	1	0	0	32	1619	1594	2083	142
1	1	1	1	1	0	0	32	1649	1705	2091	142
0	1	1	1	1	0	0	32	1521	1649	1842	142
1	0	1	1	1	1	0	32	1676	1652	2165	137
1	0	2	1	1	1	0	32	1693	1660	2119	143
1	0	1	1	1	0	1	32	1835	1877	2412	144
1	0	1	1	1	1	1	32	1894	1936	2497	117
1	1	1	1	1	1	1	32	1943	2049	2497	142
1	0	2	1	1	1	1	32	1906	1943	2531	136
0	0	1	0	0	0	0	64	857	684	604	110
0	1	1	0	0	0	0	64	898	684	604	116
1	0	1	0	0	0	0	64	1011	731	763	118

Table 24: **DDR FPGA Performance and Resource Utilization Benchmarks (Virtex-II Pro) *(Continued)***

1	1	1	0	0	0	0	64	1034	849	767	112
0	0	2	0	0	0	0	64	849	691	617	110
0	1	2	0	0	0	0	64	938	807	628	116
1	0	2	0	0	0	0	64	1001	736	781	119
1	1	2	0	0	0	0	64	1049	852	776	120
0	0	4	0	0	0	0	64	845	696	628	121
0	1	4	0	0	0	0	64	938	813	634	120
1	0	4	0	0	0	0	64	1038	741	792	110

Notes:
1. These benchmark designs contain only the DDR SDRAM Controller without any additional logic. Benchmark numbers approach the performance ceiling rather than representing performance under typical user conditions.

Reference Documents

The following documents contain reference information important to understanding the PLB DDR SDRAM Controller design:

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
06/04/04	1.0	Initial release
8/12/04	1.1	Updated for Gmm; updated trademarks and supported device family listing
12/16/05	1.2	Added latency information on Read Modify Write logic of ECC and modified timing diagram for 32-bit DDR