PLB External Memory Controller (PLB EMC) (2.00a)

DS418 March 1, 2006 **Product Specification**

Introduction

This specification defines the architecture and interface requirements for the PLB EMC. This module supports data transfers between the Processor Local Bus (PLB) and external synchronous and asynchronous memory devices.

Example synchronous devices for use with this controller are the synchronous Integrated Device Technology, Inc. IDT71V546 SRAM with ZBT™ Feature. Example asynchronous devices include the IDT71V416S SRAM and Intel 28F128J3A StrataFlash Memory.

To allow the user to obtain a PLB EMC that is uniquely tailored for their system, certain features are parameterizable in the PLB EMC 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.

Features

- Parameterized for up to a total of four memory (Synchronous/Asynchronous) banks
- Separate base addresses and address range for each bank of memory
- Memory data width is independent of PLB data width (memory data width must be less than or equal to PLB data width)
  - Supports memory widths of 64bits, 32 bits, 16 bits, or 8 bits
- Memory width can vary by bank
- Supports the following PLB transactions:
  - single beat read/write transfers
  - In-line burst for 4,8,16 word cacheline read/write transfers
  - Determinate/Indeterminate burst

LogiCORE™ Facts		
Core Specifics		
Supported Device Family	QPro™-R Virtex™-II, QPro Virtex-II, Spartan™-II, Spartan-IIE, Spartan-3, Spartan-3E, Virtex, Virtex-II, Virtex-II Pro, Virtex-4, Virtex-E	
Version of Core	plb_emc	v2.00.a
Resources Used		
	Min	Max
Slices	270	874
LUTs	364	1177
FFs	151	858
Block RAMs	0	0
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 6.2i or later	
Verification	N/A	
Simulation	ModelSim SE/EE 5.8b or later	
Synthesis	XST	
Support		
Support provided by Xilinx, Inc.		

Features (contd)

- Parameterizable memory data-width/bus data-width matching
  - Multiple memory cycles will be performed when the memory width is less than the PLB bus width to provide full utilization of the PLB bus
  - Data-width matching can be enabled separately for each memory bank
- Cycle time configurable per memory datasheet parameter
- Supports single-beat and burst transactions
- Operating frequency >=100 MHz

Functional Description

The PLB EMC receives control signals from the PLB to read and write to external memory devices.

The PLB EMC provides an interface between the PLB and one to four external banks of memory components. The EMC supports PLB data bus widths of 8,16, 32 & 64 bits, and memory subsystem widths of 8,16, 32 & 64 bits. The PLB EMC supports the PLB V3.4 byte enable architecture. Any access size up to the width of the PLB data bus is permitted. When the width of the memory is less than the width of the PLB, multiple memory cycles are performed to transfer the data width of the bus if data-width matching has been enabled for that memory bank.

The PLB EMC provides basic read/write control signals and the ability to configure the access times for read, write, and recovery times when switching from read to write or write to read.

When the PLB EMC is set for flash memory control it is organized like an SRAM interface. The PLB EMC assumes that the Flash programming circuitry is built into the Flash components and that the command interface to the Flash is handled in software.Refer to the section Data-Width Matching For Flash Memories for more information on programming Flash memories with Data-Width matching parameter enabled

The ZBT control does not support sleep mode, burst mode, parity checking or parity generating.

Figure 1 illustrates the top Level block diagram of the PLB EMC.

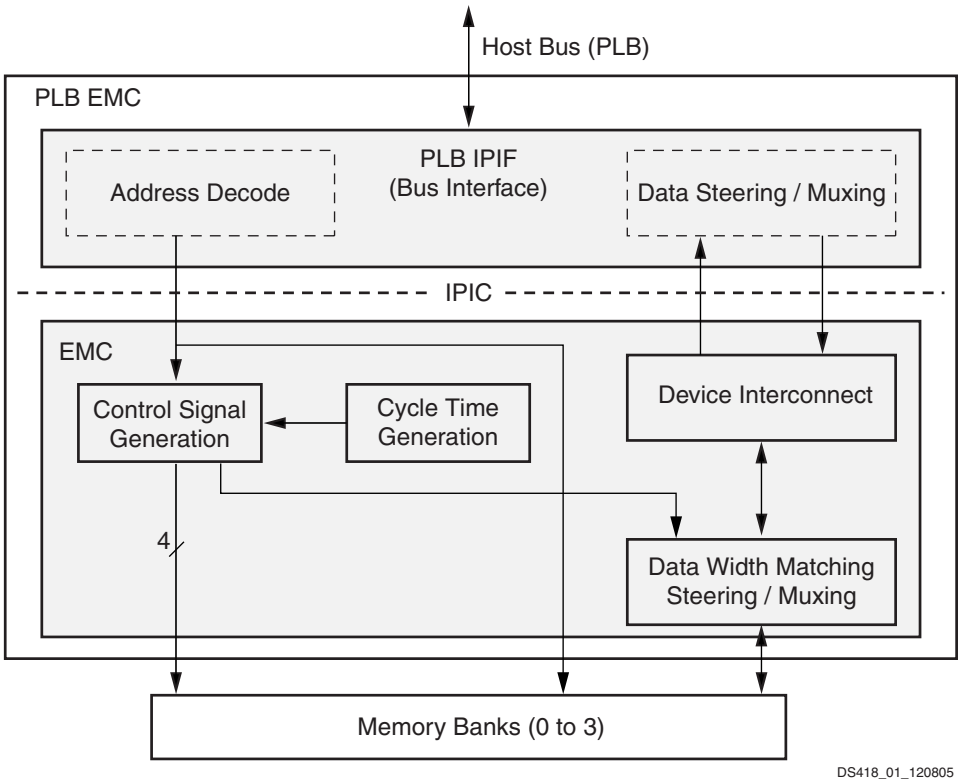


Figure 1: **PLB EMC Top-Level Block Diagram**

PLB IPIF Interconnect and (IPIC) Interface

Figure 2 illustrates all PLB slave interface input/output signals. See PLB Signals in the Processor Local Bus Architecture Specification (v3.4) for a detailed functional description of the signals. It should be noted that the values listed in the figure are the default values specified in the PLB Architecture Specification (v3.4).

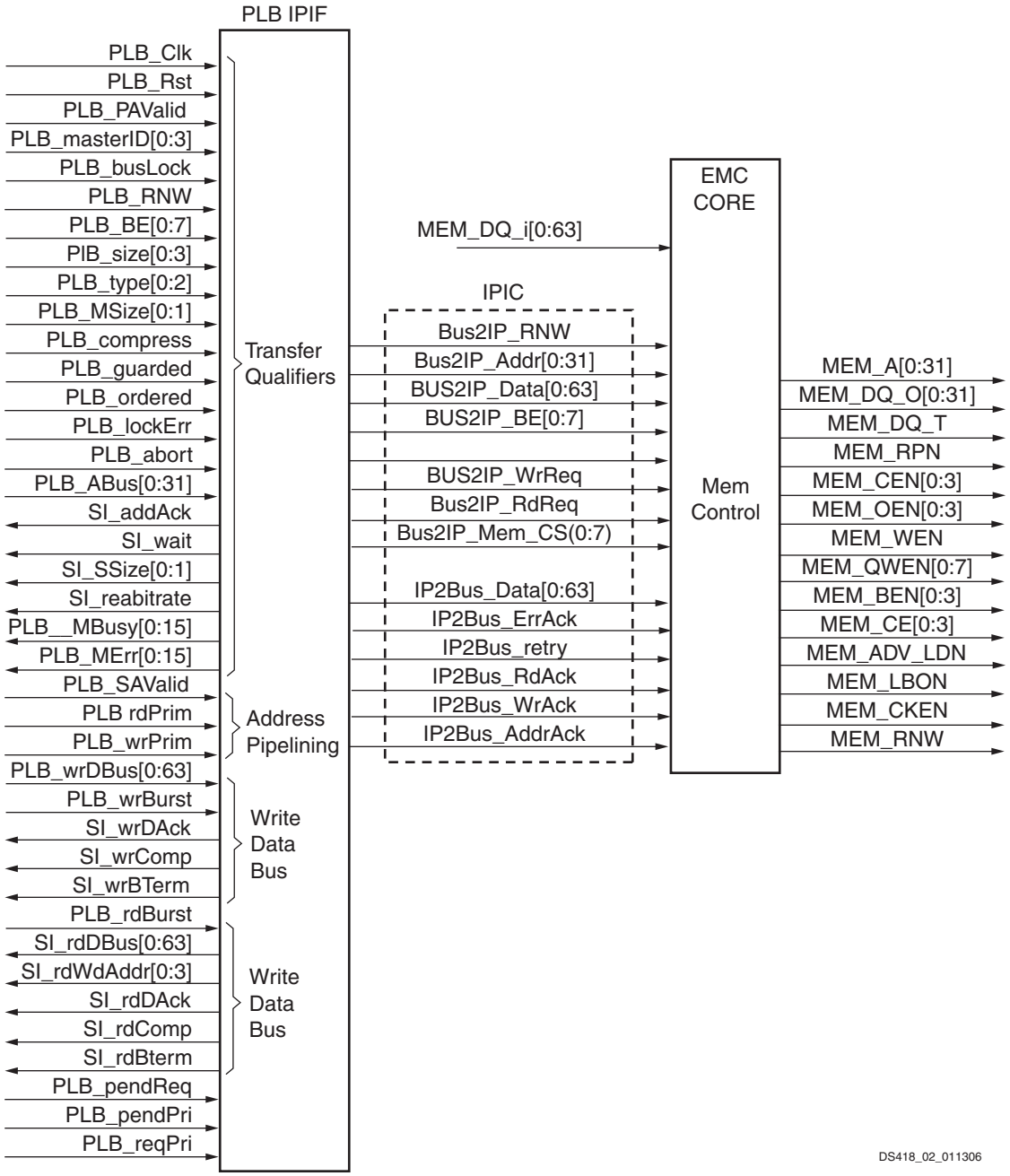


Figure 2: **PLB Slave Interface**

PLB EMC I/O Signals

The I/O signals for the PLB EMC are shown in Figure 2 and described in Table 1.

Table 1: **PLB EMC I/O Signals**

Signal Name	Interface	I/O	Description
PLB_abort	PLB	I	PLB abort bus request indicator
PLB_ABus(0:C_PLB_AWIDTH-1)	PLB	I	PLB address bus
PLB_BE(0:(C_PLD_DWIDTH / 8) -1)	PLB	I	PLB byte enables
PLB_busLock	PLB	I	PLB bus lock
PLB_compress	PLB	I	PLB compressed data transfer indicator
PLB_guarded	PLB	I	PLB guarded transfer indicator
PLB_lockErr	PLB	I	PLB lock error indicator
PLB_masterID(0:C_PLB_MID_WIDTH-1)	PLB	I	PLB current master indicator
PLB_ordered	PLB	I	PLB synchronize transfer indicator
PLB_PAValid	PLB	I	PLB primary address valid indicator
PLB_rdBurst	PLB	I	PLB burst read transfer indicator
PLB_rdPrim	PLB	I	PLB secondary to primary read request indicator
PLB_RNW	PLB	I	PLB read not write
PLB_SAValid	PLB	I	PLB secondary address valid indicator
PLB_size(0:3)	PLB	I	PLB transfer size
PLB_type(0:2)	PLB	I	PLB transfer type
PLB_wrBurst	PLB	I	PLB burst write transfer indicator
PLB_wrDBus(0:C_PLB_DWIDTH -1)	PLB	I	PLB write data bus
PLB_wrPrim	PLB	I	PLB secondary to primary write request indicator
PLB_MSize(0:1)	PLB	I	PLB master data bus size
Sl_addrAck	PLB	O	Slave address acknowledge
Sl_MBusy(0:C_NUM_MASTERS-1)	PLB	O	Slave busy indicator
Sl_MErr(0:C_NUM_MASTERS-1)	PLB	O	Slave error indicator
Sl_rdBTerm	PLB	O	Slave terminate read burst transfer
Sl_rdComp	PLB	O	Slave read transfer complete indicator
Sl_rdDAck	PLB	O	Slave read data acknowledge
Sl_rdDBus(0:C_PLB_DWIDTH -1)	PLB	O	Slave read bus
Sl_rdWdAddr(0:3)	PLB	O	Slave read word address
Sl_rearbitrate	PLB	O	Slave rearbitrate bus indicator
Sl_wait	PLB	O	Slave wait indicator
Sl_wrBTerm	PLB	O	Slave terminate write burst transfer
Sl_wrComp	PLB	O	Slave write transfer complete indicator
Sl_wrDAck	PLB	O	Slave write data acknowledge

Table 1: **PLB EMC I/O Signals *(Contd)***

Signal Name	Interface	I/O	Description
Sl_SSize(0:1)	PLB	O	Slave data bus size
PLB_pendReq	PLB	I	PLB pending bus request indicator
PLB_pendPri(0:1)	PLB	I	PLB pending request priority
PLB_reqPri(0:1)	PLB	I	PLB current request priority
PLB_Clk	System	I	System C2 clock
PLB_Rst	System	I	System PLB Reset
Mem_DQ_I(0:C_MAX_MEM_WIDTH-1)	External Memory	I	Memory Input Data Bus
Mem_DQ_O(0:C_MAX_MEM_WIDTH-1)	External Memory	O	Memory Output Data Bus
Mem_DQ_T(0:C_MAX_MEM_WIDTH-1)	External Memory	O	Memory Output 3-state Signal
Mem_A(0:C_PLB_AWDITH-1)	External Memory	O	Memory Address Bus
Mem_RPN	External Memory	O	Memory Reset/Power Down
Mem_CEN(0:C_NUM_BANKS_MEM-1)	External Memory	O	Memory Chip Enables (Active Low)(1)
Mem_OEN(0:C_NUM_BANKS_MEM-1)	External Memory	O	Memory Output Enable
Mem_WEN	External Memory	O	Memory Write Enable
Mem_QWEN(0:(C_MAX_MEM_WIDTH/8) -1)	External Memory	O	Memory Qualified Write Enables
Mem_BEN(0:(C_MAX_MEM_WIDTH/8) -1)	External Memory	O	Memory Byte Enables
Mem_CE(0:C_NUM_BANKS_MEM-1)	External Memory	O	Memory Chip Enables (Active High)(1)
Mem_ADV_LDN	External Memory	O	Memory Advance Burst Address/Load New Address
Mem_LBON	External Memory	O	Memory Linear/Interleaved Burst Order
Mem_CKEN	External Memory	O	Memory Clock Enable
Mem_RNW	External Memory	O	Memory Read Not Write

Notes:

1. Most asynchronous memory devices will only use Mem_CEN. Most synchronous memory devices will use both Mem_CEN and Mem_CE. Refer to the device data sheet for correct connection of these signals.

PLB EMC Parameters

Some features of the PLB EMC can be parameterized to allow the user to configure a design tailored for the required application and to utilize the resources actually required.

The parameterizable characteristics of the PLB EMC are described in Table 2:

- Number of separate memory banks
- Memory type, synchronous or asynchronous, per memory bank
- Data width of each memory bank
- Pipeline delay of each memory bank (synchronous memories)
- Read and write access times for each memory bank (asynchronous memories)
- Enabling of data-width matching per memory bank
- Enabling burst support
- Enabling negative edge IO registersEnabling of PLB burst and cacheline transaction support

Table 2: **PLBEMC Design Parameters**

Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
Number of Masters	C_NUM_MASTERS	1 - 8	8	integer
Number of Memory Banks	C_NUM_BANKS_MEM	1 - 4	2	integer
Output/Input data and control signals using the falling edge of the clock(18)	C_INCLUDE_NEGEDGE_IO REGS	0 = don't include negative edge IO registers (data and control signals are input/output on the rising edge of the clock) 1 = include negative edge IO registers (data and control signals are input/output on the falling edge of the clock)	0	integer
PLB Clock Period	C_PLB_CLK_PERIOD_PS	Integer number of picoseconds	10000	integer
Memory Bank x Base Address	C_MEMx(1,2)_BASEADDR	Valid Address Range(3)	None(4)	std_logic _vector
Memory Bank x High Address	C_MEMx(1,2)_HIGHADDR	Address range must be a power of 2 and ≤ PLB Address Space(3)	None(4)	std_logic _vector
PLB Data Bus Width	C_PLB_DWIDTH	64	64	integer
PLB Address Bus Width	C_PLB_AWIDTH	32	32	integer
Width of Memory Bank x Data Bus	C_MEMx_WIDTH(1)	8,16, 32 & 64	64	integer

Table 2: **PLBEMC Design Parameters *(Contd)***

Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
Execute multiple memory access cycles to match width of Memory Bank x data bus to PLB data bus	C_INCLUDE_DATAWIDTH_MATCHING_x(1,17)	0 = don't include data-width matching 1 = include data-width matching	1	integer
Include support for PLB burst and cacheline transactions	C_INCLUDE_BURST_CACHELN_SUPPORT	0 = Design will respond to plb burst and cacheline transactions and initiate single memory operations 1 = include logic to translate plb burst and cacheline transactions as memory burst operations	0	integer
Memory type	C_SYNCH_MEM_x(1)	0 = memory type is asynchronous 1 = memory type is synchronous	0	integer
Pipeline delay (only used if C_SYNCH_MEM_x = 1)	C_SYNCH_PIPEDELAY_x(1)	1 to 2 clocks	2	integer
Read cycle Chip Enable low to Data Valid. (5,6)	C_TCEDV_PS_MEM_x(1)	Integer number of picoseconds	0(15)	integer
Read cycle Address Valid to Data Valid. (5,7)	C_TAVDV_PS_MEM_x(1)	Integer number of picoseconds	0(15)	integer
Read cycle Chip Enable high to Data Bus High Impedance. (8,9)	C_THZCE_PS_MEM_x(1)	Integer number of picoseconds	0(15)	integer
Read cycle Output Enable high to Data Bus High Impedance. (8,10)	C_THZOE_PS_MEM_x(1)	Integer number of picoseconds	0(15)	integer
Write cycle time (4,11)	C_TWC_PS_MEM_x(1)	Integer number of picoseconds	0(15)	integer
Write Enable minimum pulse width. (4,12)	C_TWP_PS_MEM_x(1)	Integer number of picoseconds	0(15)	integer
Write cycle Write Enable high to Data Bus Low Impedance. (13,14)	C_TLZWE_PS_MEM_x(1)	Integer number of picoseconds	0(15)	integer

Table 2: **PLBEMC Design Parameters *(Contd)***

Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
Maximum width of the memory devices in all of the banks - width of memory bus out to the devices AUTOCALCULATED (16)	C_MAX_MEM_WIDTH	8,16,32,64	64	integer
Width of Master ID Bus AUTOCALCULATED(16)	C_PLB_MID_WIDTH	$\log_2$(C_NUM_MASTERS)	3	integer

Notes:

1. x = values for memory banks 0 to 3
2. This design can accommodate up to 4 banks of memory. The address range generics are designated as C_MEM0_BASEADDR, C_MEM1_BASEADDR, C_MEM0_HIGHADDR, C_MEM1_HIGHADDR, etc.
3. No default value will be specified for C_MEMx_BASEADDR, C_MEMx_HIGHADDR to insure that the actual value is set, i.e. if the value is not set, a compiler error will be generated. These generics must be a power of 2 and encompass the memory size for C_MEMx_BASEADDR, C_MEMx_HIGHADDR.
4. Write enable low time is the maximum of C_TWC_PS_MEM and C_TWP_PS_MEM.
5. Read cycle time is the maximum of C_TCEDV_PS_MEM and C_TAVDV_PS_MEM.
6. Chip Enable low to Data Valid, C_TCEDV_PS_MEM, is equivalent to t_{ACE} for asynchronous SRAM and t_{ELQV} for FLASH in the respective memory device data sheets.
7. Address Valid to Data Valid, C_TAVDV_PS_MEM, is equivalent to t_{AA} for asynchronous SRAM and t_{AVQV} for FLASH in the respective memory device data sheets.
8. Read cycle recovery to write is the maximum of C_THZCE_PS_MEM and C_THZOE_PS_MEM.
9. Chip Enable high to data bus High Impedance, C_THZCE_PS_MEM, is equivalent to t_{HZCE} for asynchronous SRAM and t_{EHQZ} for FLASH in the respective memory device data sheets.
10. Output Enable high to data bus High Impedance, C_THZOE_PS_MEM, is equivalent to t_{HZOE} for asynchronous SRAM and t_{GHQZ} for FLASH in the respective memory device data sheets.
11. Write cycle time, C_TWC_PS_MEM, is equivalent to t_{WC} for asynchronous SRAM and t_{CW} for FLASH in the respective memory device data sheets.
12. Write cycle minimum pulse width, C_TWP_PS_MEM is equivalent to t_{WP} for asynchronous SRAM and t_{PWE} for FLASH in the respective memory device data sheets.
13. Write Enable high to data bus Low Impedance, C_TLZWE_PS_MEM, is equivalent to t_{LZWE} for asynchronous SRAM and t_{WHGL} for FLASH in the respective memory device data sheets.
14. C_TLZWE_PS_MEM is the parameter set to meet write recovery to read time requirements.
15. A value must be set for this parameter if the memory type in this bank is asynchronous - refer to the memory device data sheet for the correct value.
16. This parameter is automatically calculated when using CoreGen, otherwise the user must set this parameter to the maximum value of the C_MEMx_WIDTH genericsand set C_PLB_MID_WIDTH to $\log_2$(C_NUM_MASTERS).
17. Refer to section Data-Width Matching For Flash Memories for programming FLASH memories with the Data-width matching parameter enabled.
18. This parameter should only be set to 1 under the following conditions:
    •Tfpga_outdelay + Tsetup_memory + Tboard_route_delay < Clock_period/2
    • Tmemory_outdelay + Tfpga_setup + Tboard_route_delay < Clock_period/2
    See section "IO Registers" on page 16 for more information on using this parameter.

Allowable Parameter Combinations

The EMC supports up to 4 banks of Synchronous and/or Asynchronous memory. Each bank of memory has it's own independent base address and address range. The address range of a bank of memory is restricted to be a power of 2.

If the desired address range is represented by 2^n, then the n least significant bits of the base address must be 0. For example, a memory bank with an addressable range of 16M (2^{24}) bytes could have a base address of 0xFF000000 and a high address of 0xFFFFFFFF.A memory bank with an addressable range of 64K (2^{16}) bytes could have a base address of 0xABCD0000 and a high address of 0xABCDFFFF.

If C_SYNCH_MEMORY=1, then C_SYNCH_PIPEDELAY specifies the pipeline delay of that synchronous memory type. All other timing parameters for that memory bank can remain at the default value of 0. If C_SYNCH_MEMORY=0, C_SYNCH_PIPEDELAY is unused. All other timing parameters for that memory bank must be set to the value specified in the memory device data sheet.

Please refer to the section, Data-Width Matching For Flash Memories for FLASH memories if C_INCLUDE_DATAWIDTH_MATCHING_x=1 for FLASH memory banks.

C_INCLUDE_NEGEDGE_IOREGS provides no benefit when interfacing to asynchronous memories. Therefore, if there are no synchronous memories in the system, this parameter should be set to 0.

Parameter-Port Dependencies

The width of many of the PLB EMC signals depends on the number of memories in the system and the width of the various data and address buses. In addition, when certain features are parameterized away, the related input signals are unconnected and the related output signals are set to a constant values. The dependencies between the PLB EMC design parameters and I/O signals are shown in Table 3.

Table 3: **Parameter-Port Dependencies**

Name	Affects	Depends	Relationship Description
Design Parameters			
C_NUM_MASTERS	SI_MBusy SI_MErr C_PLB_MID_WIDTH		Number of bits required based on the number of PLB Masters in the system
C_PLB_DWIDTH	PLB_BE PLB_wrDBus SI_rdDBus		Number of Byte Enables, width of the PLB Write Data Bus, and the width of the Slave Read Data Bus all vary based on the PLB data width
Parameters			
C_PLB_AWDITH	PLB_ABUS Mem_A		Width of the PLB Address Bus and the width of the Memory Address Bus all vary based on the PLB address width
C_PLB_MID_WIDTH	PLB_masterID		Size of the masterID is decoded as the function log2x(C_NUM_MASTERS)
C_MAX_MEM_WIDTH	Mem_DQ_I Mem_DQ_O Mem_DQ_T Mem_BEN Mem_QWEN		Width of the memory interface and control buses vary based on the maximum data width of the memory banks
C_NUM_BANKS_MEM	Mem_CEN Mem_CE Mem_OEN		Status signal per Memory Bank Chip Select (low) per Memory Bank Chip Enable (high) per Memory Bank Output Enable(low) per Memory Bank
I/O Signals			
PLB_ABus		C_PLB_AWIDTH	Width varies with the width of the PLB Address Bus

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

Name	Affects	Depends	Relationship Description
PLB_DBus		C_PLB_DWIDTH	Width varies with the width of the PLB Data Bus
PLB_BE		C_PLB_DWIDTH	Width varies with the width of the PLB Data Bus
PLB_masterID		C_PLB_MID_WIDTH	Width varies with the number of bits required to encode the number of PLB Masters
PLB_wrDBus		C_PLB_DWIDTH	Width varies with the width of the PLB Data Bus
Sl_MBusy		C_NUM_MASTERS	Width varies with the number of PLB masters
Sl_MErr		C_NUM_MASTERS	Width varies with the number of PLB masters
Sl_rdDBus		C_PLB_DWIDTH	Width varies with the width of the PLB Data Bus
Mem_DQ_I		C_MAX_MEM_WIDTH	Width varies with the width of the memory specified
Mem_DQ_O		C_MAX_MEM_WIDTH	Width varies with the width of the memory specified
Mem_A		C_PLB_AWDITH	Width varies with the width of the PLB Address Bus
Mem_CEN		C_NUM_BANKS_MEM	Width varies with the width of the number of memory banks specified
Mem_OEN		C_NUM_BANKS_MEM	Width varies with the width of the number of memory banks specified
Mem_QWEN		C_MAX_MEM_WIDTH	Width varies with the width of the memory specified
Mem_BEN		C_MAX_MEM_WIDTH	Width varies with the width of the memory specified
Mem_CE		C_NUM_BANKS_MEM	Width varies with the width of the number of memory banks specified

PLB EMC Address Map Description

The PLB EMC supports up to 4 banks memory. Each bank of memory has it's own independent base address and address range. The address range of a bank of memory is restricted to be a power of 2. If the desired address range is represented by 2^n, then the n least significant bits of the base address must be 0. For example, a memory bank with an addressable range of 16M (2^{24}) bytes could have a base address of 0xFF000000 and a high address of 0xFFFFFFFF. A memory bank with an addressable range of 64K (2^{16}) bytes could have a base address of 0xABCD0000 and a high address of 0xABCDFFFF.

The addresses for each bank of memory are shown in Table 4.

Table 4: **PLB EMC Memory Banks**

Memory	Base Address	High Address	Access
Bank 0	C_MEM0_BASEADDR	C_MEM0_HIGHADDR	Read/Write
Bank 1	C_MEM1_BASEADDR	C_MEM1_HIGHADDR	Read/Write
Bank 2	C_MEM2_BASEADDR	C_MEM2_HIGHADDR	Read/Write
Bank 3	C_MEM3_BASEADDR	C_MEM3_HIGHADDR	Read/Write

Memory Data Types and Organization

Memory can be accessed through the EMC as one of four types: byte (8 bits), halfword (2 bytes), word (4 bytes), and doubleword (8 bytes). 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 3.

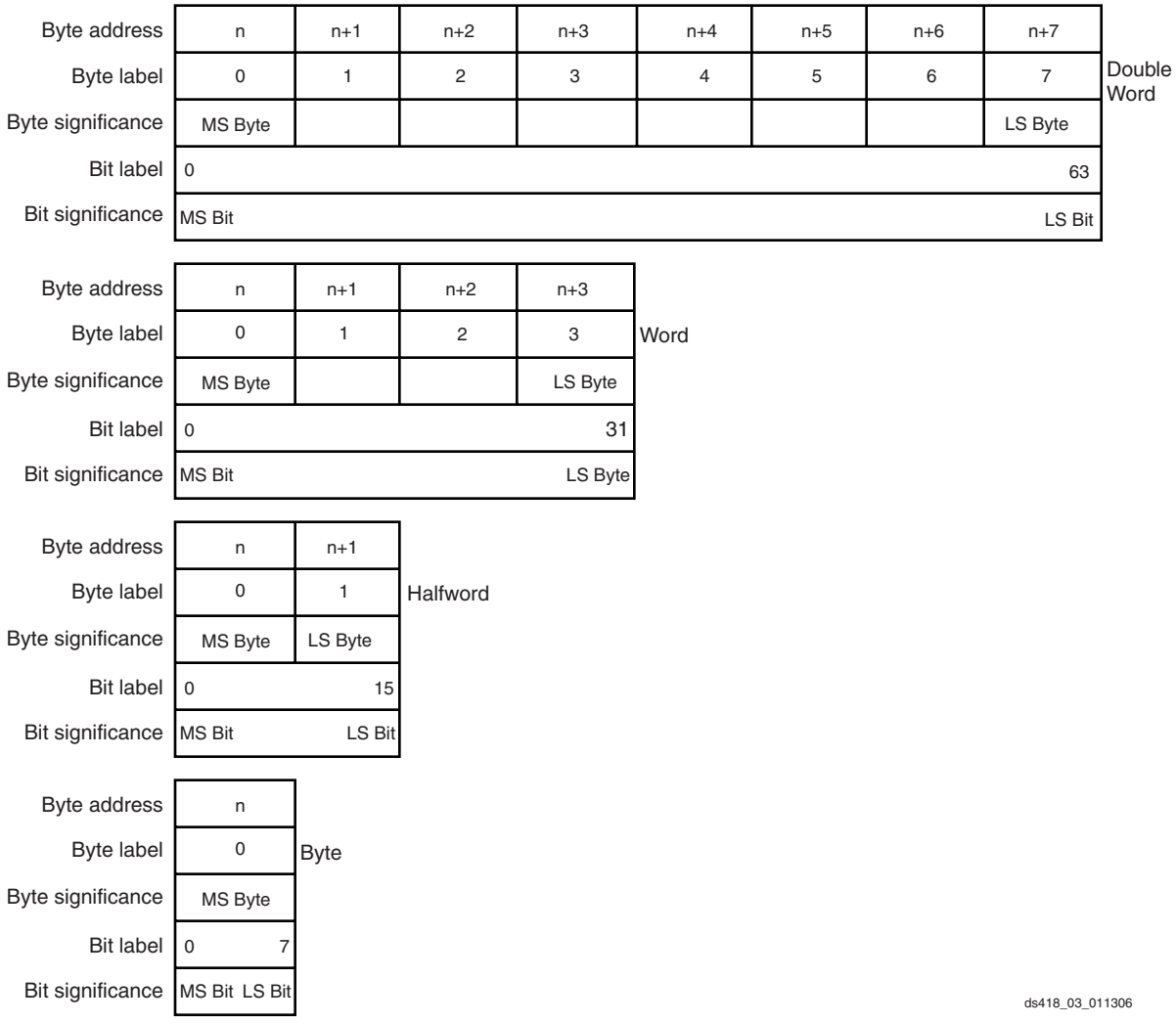
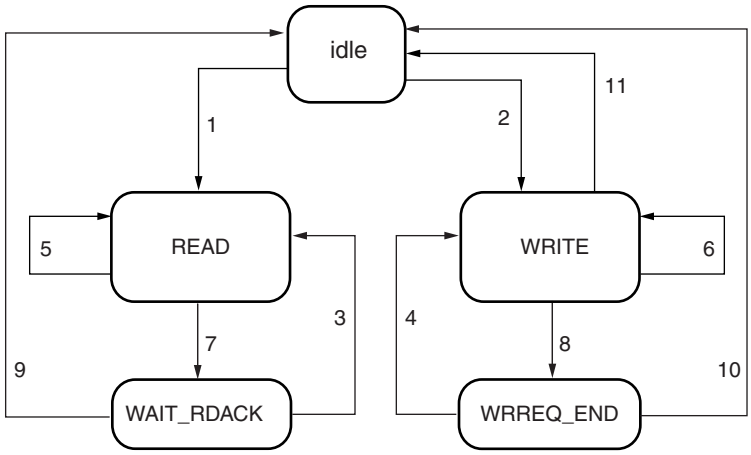


Figure 3: **PLB Big-Endian Data Types**

PLB EMC Operation

The PLB EMC state machine is shown in Figure 4.

EMC State Machine



Legend:

1. Bus2Mem_CS * Bus2IP_RdReq * Tlz_End
2. Bus2Mem_CS * Bus2IP_WrReq * Thz_End
3. Bus2Mem_Burst * Mem2Bus_RdAck
4. Bus2Mem_Burst * Mem2Bus_wrAck
5. (Datawidth_match * $\overline{\text{Cycle\_End}}$) + $\overline{\text{Trd\_End}}$)
6. (Datawidth_match * $\overline{\text{Cycle\_End}}$) + $\overline{\text{Twr\_End}}$)
7. ($\overline{\text{Datawidth\_match}}$ * Trd_End) + (Cycle_End * Trd_End)
8. ($\overline{\text{Datawidth\_match}}$ * Twr_End) + (Bus2IP_WrReq * Cycle_End * Twr_End)
9. $\overline{\text{Bus2Mem\_CS}}$
10. $\overline{\text{Bus2Mem\_CS}}$
11. $\overline{\text{Bus2Mem\_CS}}$ * (Cycle_End * Datawidth_match * $\overline{\text{Bus2IP\_WrReq}}$ * Twr_End)

DS418_04_011306

Figure 4: **PLB EMC State Machine**

Data-Width Matching

Data-width matching is the term used to describe the PLB EMC operation of performing multiple memory accesses to match the width of the memory bank's data bus to the width of the PLB data bus. When C_INCLUDE_DATAWIDTH_MATCHING=1 for a particular memory bank and the width of the memory bank is less than the width of the PLB data bus, multiple accesses are made to the memory bank so that the PLB data bus is fully utilized. An address counter within the PLB EMC is used to provide the correct memory addresses. The address counter is controlled by the state machine and is incremented as each transaction completes.

For example, if the memory bank being addressed is 8 bits wide, then 8 writes to or reads from that memory will be performed in order that the full 64 bits of the PLB data bus are used. For a write cycle, the first byte of the PLB data bus (bits 0 -7) is written to the memory on bits 0-7 of the memory data bus, then the second byte (bits 8-15) on bits 0-7 of the memory bus, and so on. The write data multiplexing is shown in Figure 5for a 32-bit processor data bus to illustrate the basic concept. This can easily be extended for a 64-bit processor data bus, but is not shown in this document.

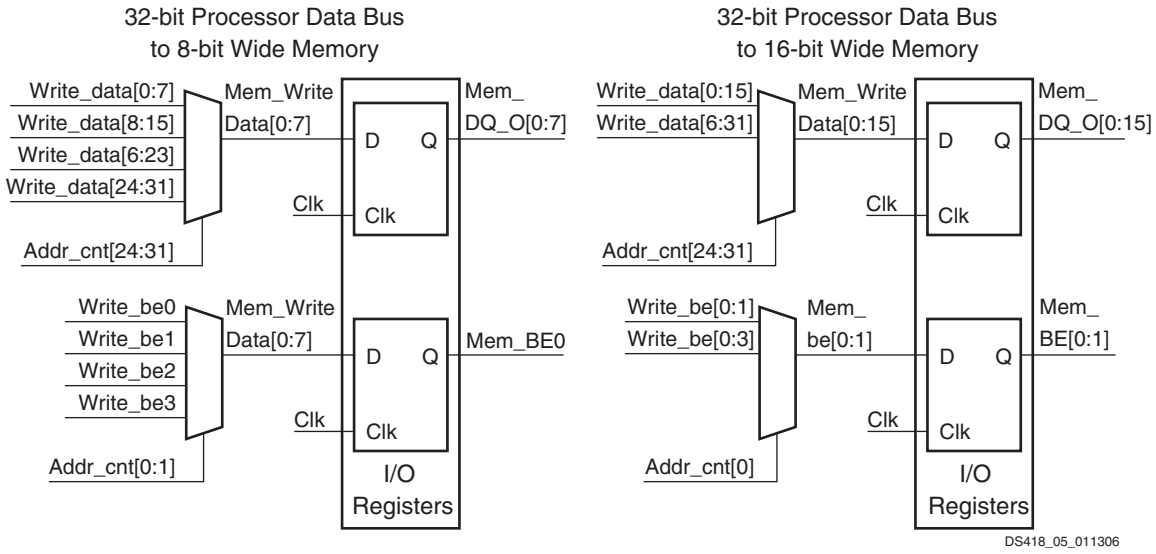


Figure 5: **Data-width matching for Write operations (32-bit Processor Data bus)**

For a read cycle, the data read on bits 0-7 of the memory bus is stored as bits 0-7 of the PLB data bus for the first byte, then the data read on bits 0-7 of the memory bus is stored as bits 8-15 PLB data bus for the second byte and so on until all bytes have been read. Then all bytes are presented to the PLB data bus and the read acknowledge is generated. This is shown in Figure 6 for a 32-bit wide processor data bus to illustrate the basic concept. This can easily be extended for a 64-bit processor data bus, but is not shown in this document. The signal Mem_width_bytes specifies the data-width of the memory bank currently being accessed.

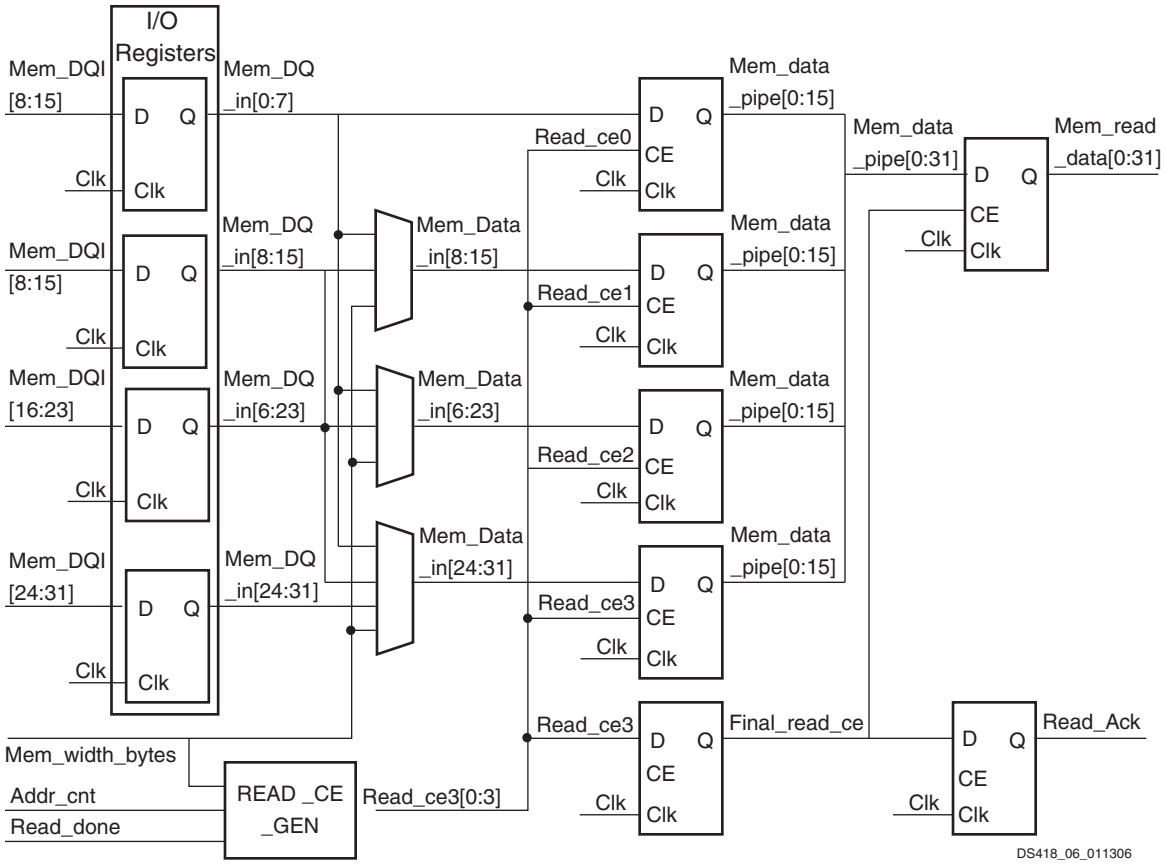


Figure 6: **Data-width matching for Read operations (32-bit data bus)**

The READ_CE signals are generated according to Table 5 for a 32-bit processor data bus to illustrate the basic logic. This can easily be extended for a 64-bit processor data bus, but is not shown in this document.

Table 5: **READ_CE Generation (32-bit processor data bus)**

Mem_width_ bytes	Addr_cnt	Read_ce(0)	Read_ce(1)	Read_ce(2)	Read_ce(3)
8	00	Read_done	0	0	0
	01	0	Read_done	0	0
	10	0	0	Read_done	0
	11	0	0	0	Read_done
16	0	Read_done	Read_done	0	0
	1	0	0	Read_done	Read_done
32	0	Read_done	Read_done	Read_done	Read_done

The data bus to/from the memory banks uses big-endian bit labeling (i.e. bit 0 is MSB) and is sized according to the C_MAX_MEM_WIDTH parameter. Memories that have smaller widths should connect to this bus starting at bit 0 (MSB - big endian bit labeling). For example, if 3 memory banks are present of sizes 8, 16, and 32 bits, the 8 bit wide memory bank should connect to bits 0-7 of the memory

data bus, the 16 bit wide memory bank should connect to bits 0 - 15 of the memory data bus, and the 32 bit wide memory should connect to bits 0 - 31 of the memory data bus.

Data-Width Matching For Flash Memories

Data-width matching is important for systems using smaller width Flash memories for program storage as cacheline accesses are expected to be transfers of the bus width. Data-width matching works well when doing reads from Flash memories as it makes a smaller width Flash memory appear to be the width of the bus. The software can access the Flash memory without knowledge of its actual width and cacheline accesses can be performed.

However, typical Flash memory programming algorithms require a command to be written before each data element is written. For example, writing 32-bit data to an 8-bit Flash memory won't work properly if data-width matching is turned on because the data write is converted into four 8-bit writes with the proper write enables asserted. This doesn't allow for the command to be written before the data. Therefore, care should be taken when software performs writes to Flash memory such that the write transfers are specified to the width of the Flash memory. Software is required to have knowledge of the actual width of the Flash memory. For example, if an 8-bit Flash memory is used, then writes to the Flash memory should always be byte-writes

IO Registers

Registers are used on all signals to and from the memory banks to provide consistent timing on the memory interface. The IO registers present in the design depend upon the setting of the C_INCLUDE_NEGEDGE_IOREGS. All signals output to the memory banks are registered on the rising edge of the system clock. If C_INCLUDE_NEGEDGE_IOREGS = 1, the signals are registered again on the falling edge of the system clock as shown in Figure 7 and can be used at lower clock frequencies to provide setup and hold to synchronous memories. This parameter can be used for asynchronous memories as well or if there is a mixture of synchronous and asynchronous memories in the system. However, if there are only asynchronous memories in the system, setting this parameter to 1 provides no value, and is therefore recommended to be set to 0.

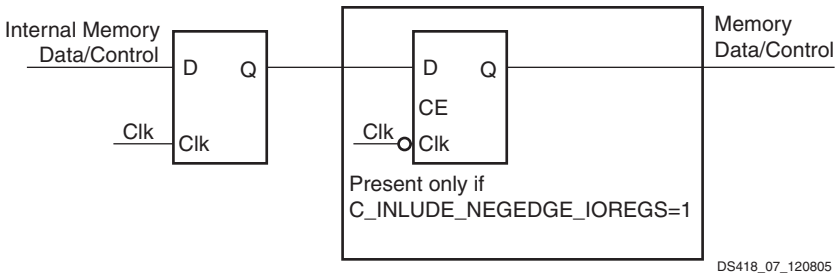


Figure 7: **PLB EMC Output Registers**

If C_INCLUDE_NEGEDGE_IOREGS = 1, the data input from the memories is clocked into the FPGA on the falling edge of the clock. This can be used at lower clock frequencies to obtain setup and hold from synchronous memories in the system. If data width matching is included for any of the memory banks, the data is again registered on the rising edge of the clock before going into the data width matching multiplexors. These registers are not included if data-width matching is not needed for any of the memory banks. This is shown inFigure 8 and can be used at lower clock frequencies to provide setup and hold from synchronous memories. This parameter can be used for asynchronous memories as well or if there is a mixture of synchronous and asynchronous memories in the system. However, if

there are only asynchronous memories in the system, setting this parameter to 1 provides no value, and is therefore recommended to be set to 0.

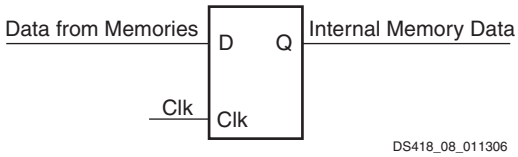


Figure 8: **PLB EMC Input Registers**

NOTE: C_INCLUDE_NEGEDGE_IOREGS should only be set to 1 if the output delay from the FPGA plus the setup requirement of the synchronous memories in the system plus the board route delay is less than half the clock period AND the output delay from the synchronous memories in the system plus the FPGA setup requirement plus the board route delay is less than half the clock period.

- Tfpga_outdelay + Tsetup_memory + Tboard_route_delay < Clock_period/2
- AND
- Tmemory_outdelay + Tfpga_setup + Tboard_route_delay < Clock_period/2

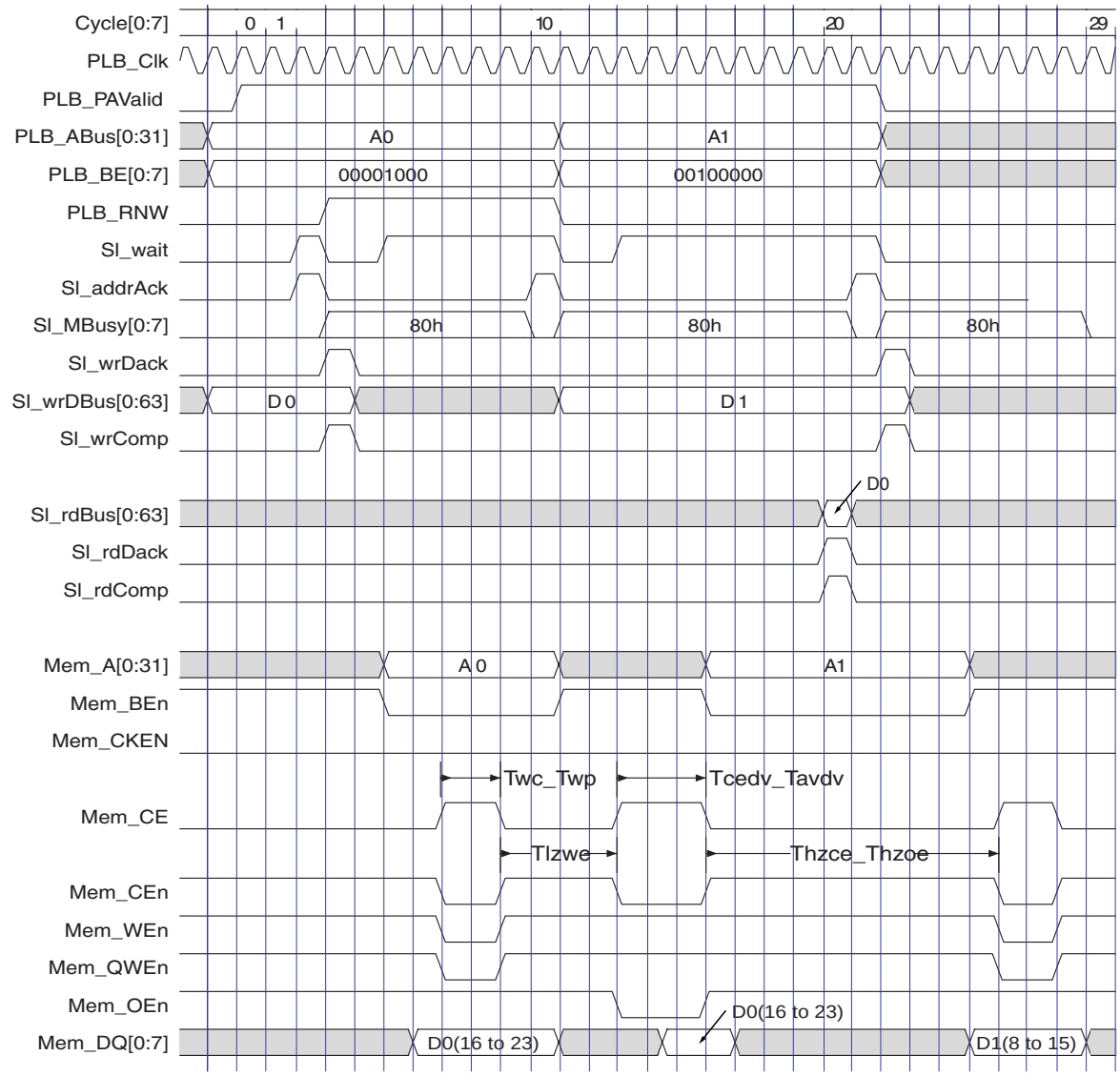
PLB Timing Diagrams

The following timing diagrams show various PLB bus transactions and the resulting memory accesses. Timing diagrams for all memory widths are not shown. A sampling is provide as an example of how various memory widths are supported.

PLB Accesses to 8-bit SRAM

This section provides timing diagrams for many different PLB transactions to an 8-bit SRAM.

Single Beat Byte Write-Read-Write to 8-bit Asynchronous SRAM



Twc_Twp = Maximum of C_TWC_PS_MEM_x and C_TWP_PS_MEM_x
Tcedv_Tavdv = Maximum of C_TCEDV_PS_MEM_x and C_TAVDV_PS_MEM_x
Thzce_Thzoe = The Maximum of C_THZCE_PS_MEMx and C_THZOE_PS_MEM_x
Tlzwe = C_TLZWE_PS_MEM_x
Note : All timing parameters are minimum.

DS418_09_011306

Figure 9: **Single Beat Byte Write-Read-Write to 8-bit Asynchronous SRAM**

Half-Word Burst Write to 8-bit Asynchronous SRAM

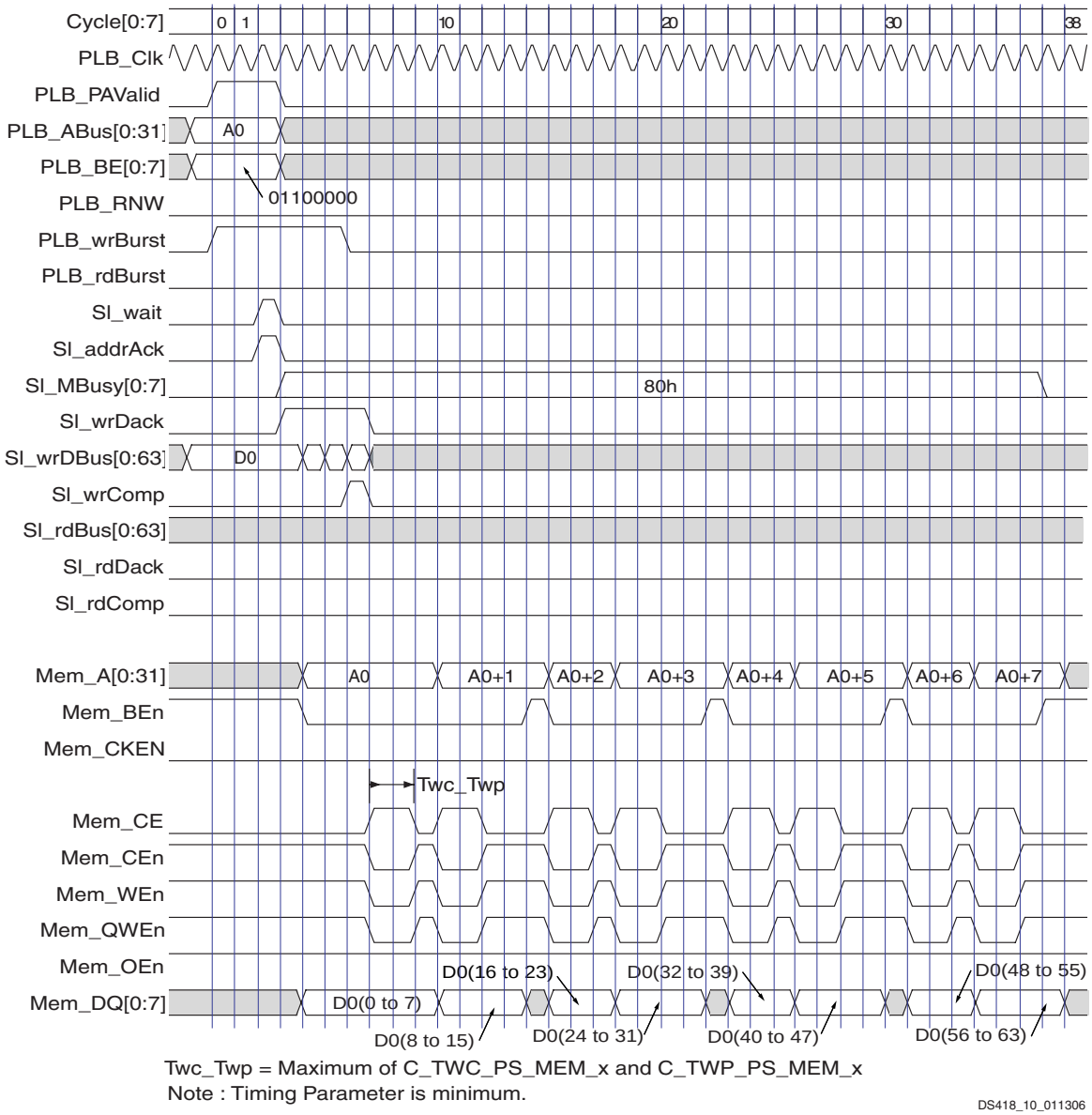


Figure 10: **Half-Word Burst Write to 8-Bit Asynchronous SRAM**

Half-Word Burst Read from 8-bit Asynchronous SRAM

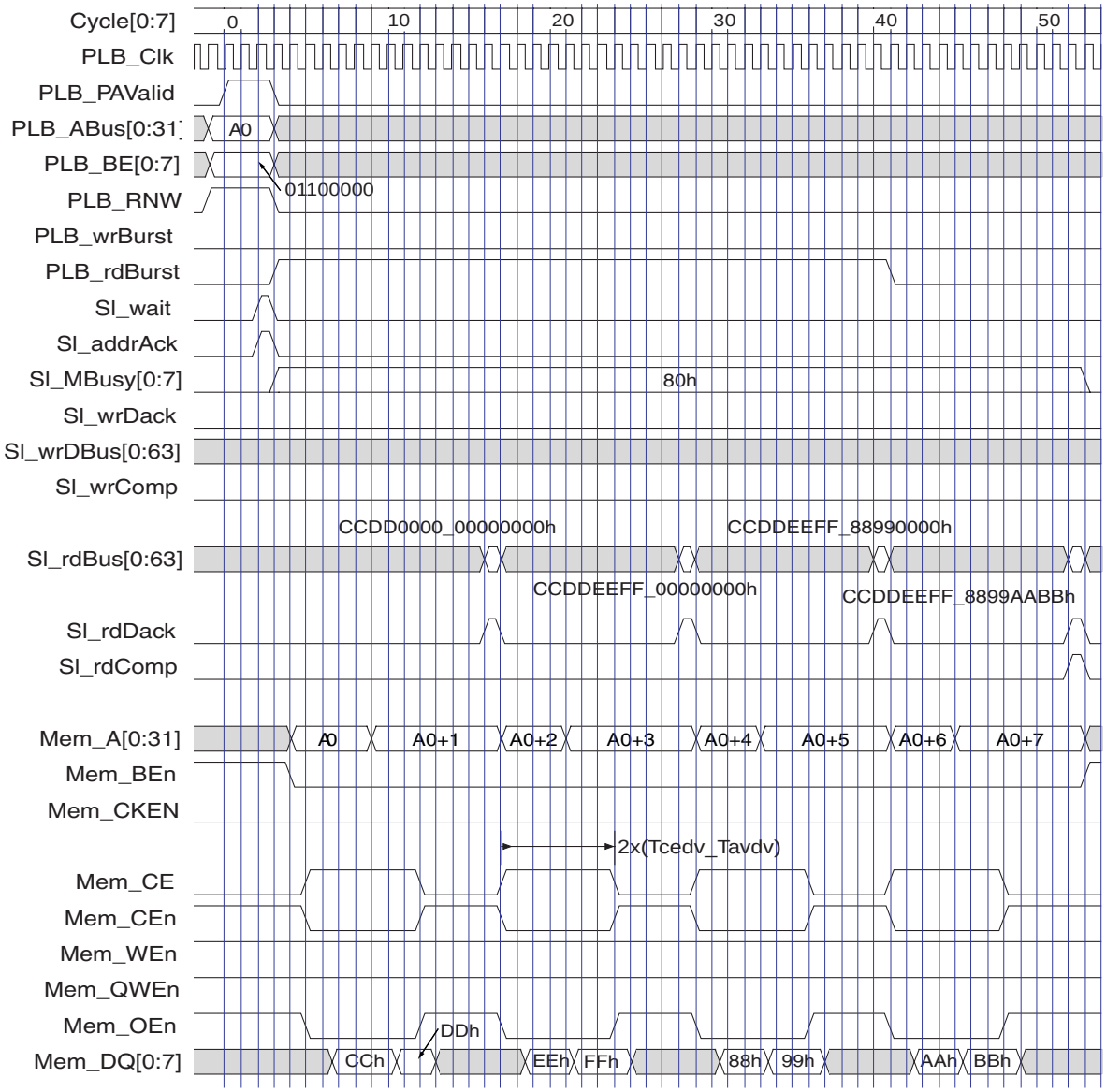
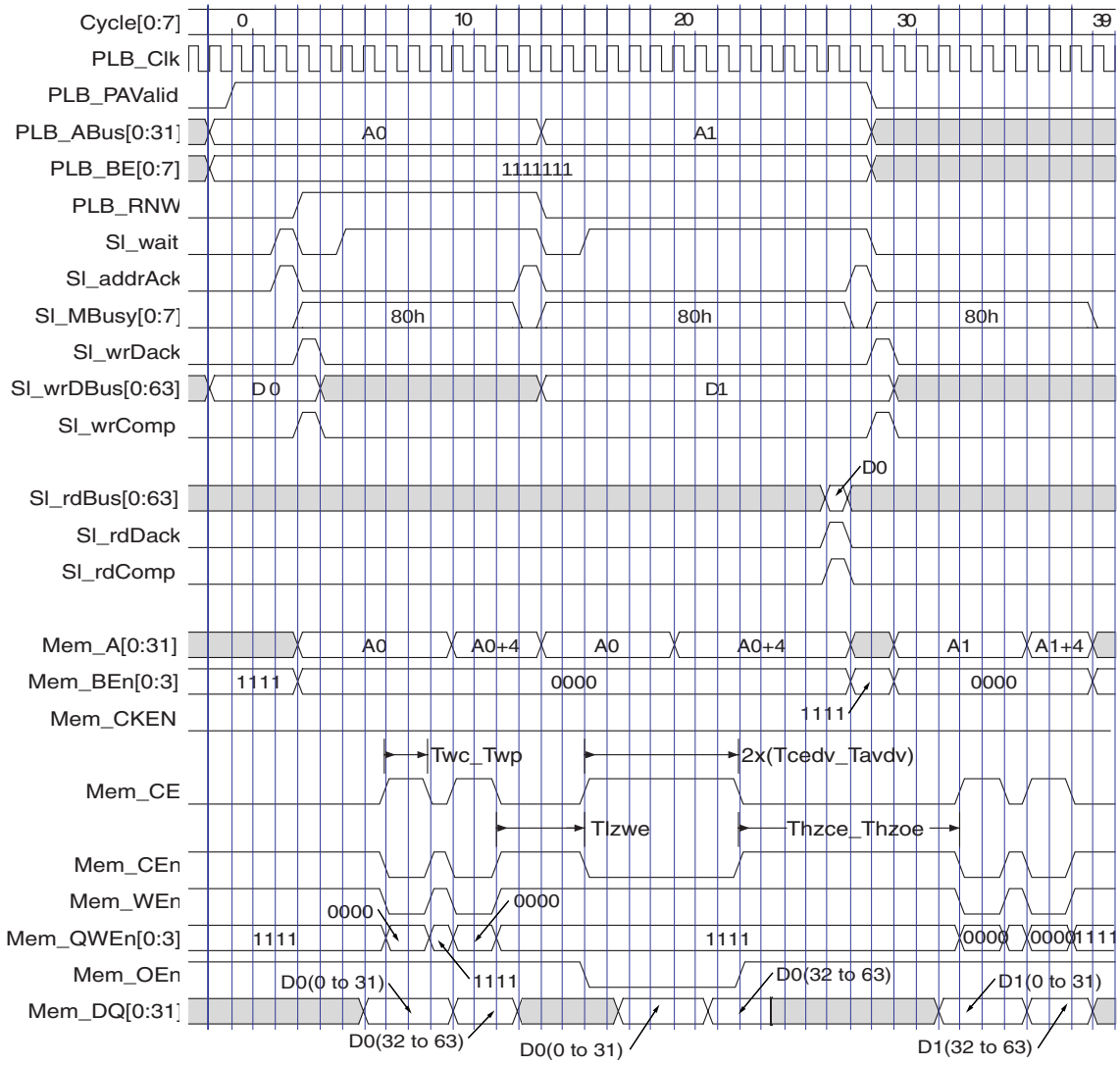


Figure 11: **Half-Word Burst Read from 8-Bit Asynchronous SRAM**

PLB Accesses to 32-bit SRAM

This section provides timing diagrams for PLB transactions to a 32-bit SRAM.

Single Beat Byte Write-Read-Write to 32-bit Asynchronous SRAM

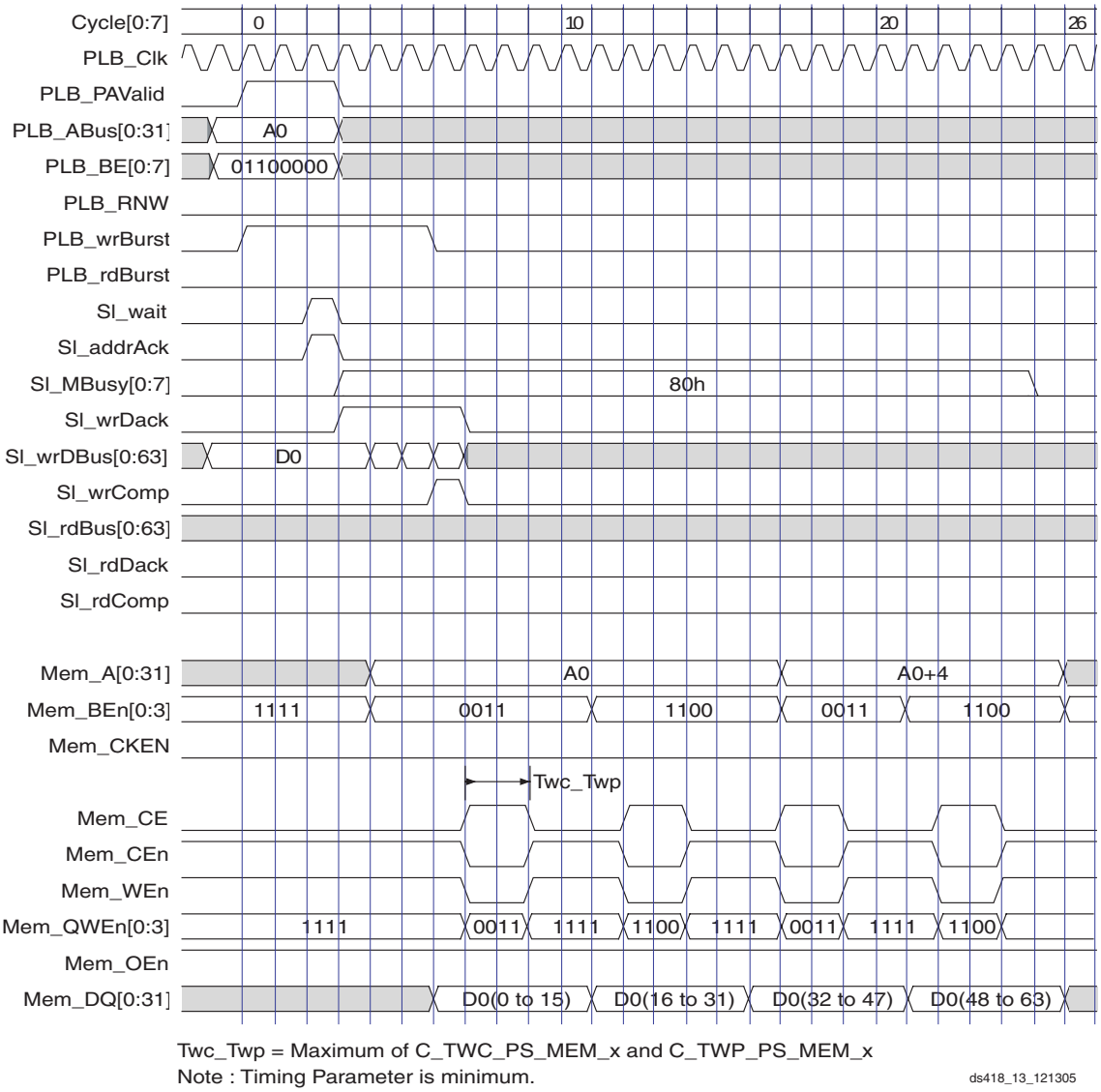


Twc_Twp = Maximum of C_TWC_PS_MEM_x and C_TWP_PS_MEM_x
Tcedv_Tavdv = Maximum of C_TCEDV_PS_MEM_x and C_TAVDV_PS_MEM_x
Thzce_Thzoe = The Maximum of C_THZCE_PS_MEMx and C_THZOE_PS_MEM_x
Tlzwe = C_TLZWE_PS_MEM_x
Note : All timing parameters are minimum.

DS418_12_011306

Figure 12: **Single Beat Byte Write-Read-Write to 32-bit Asynchronous SRAM**

Half-Word Burst Write to 32-bit Asynchronous SRAM

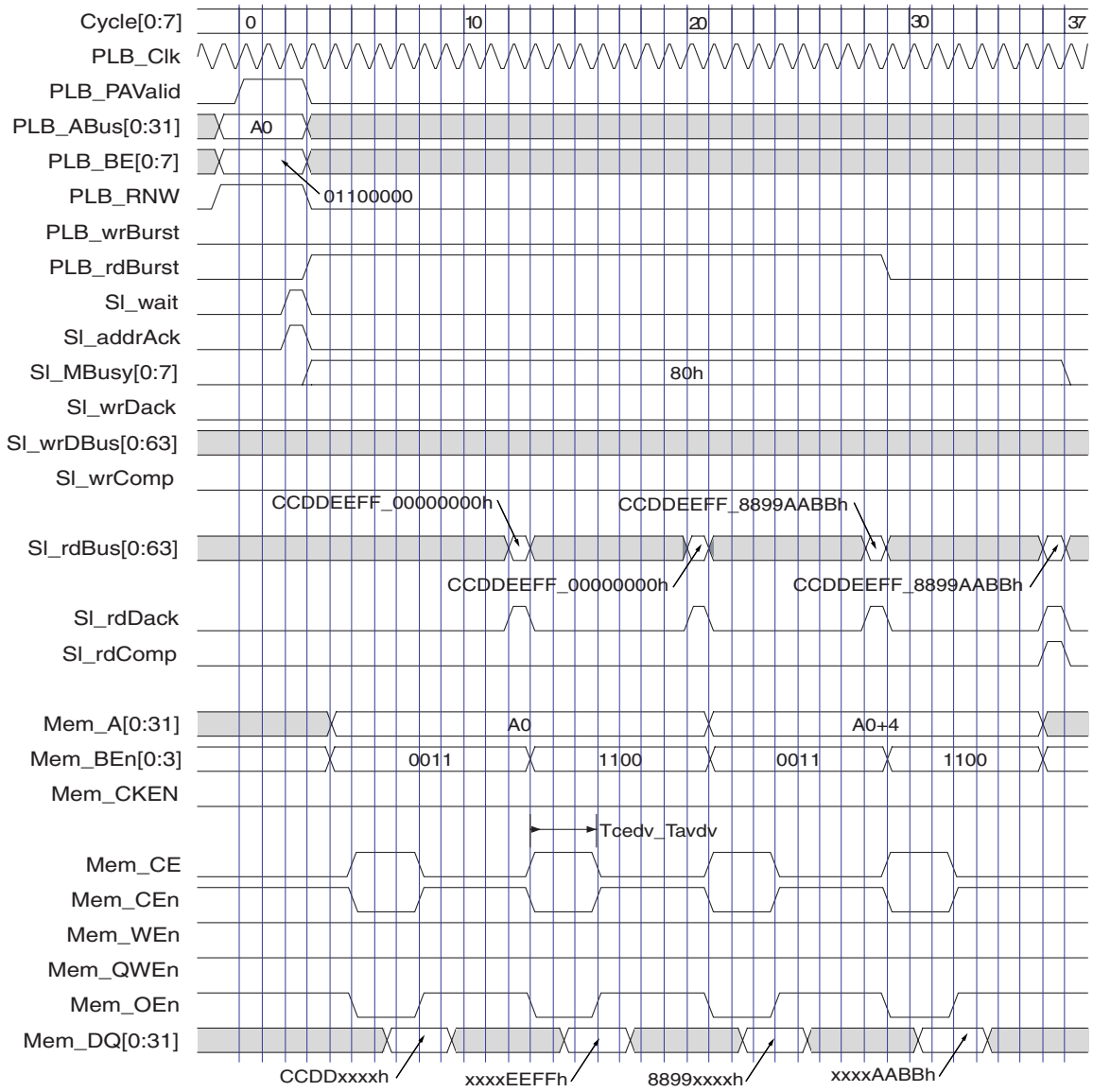


Twc_Twp = Maximum of C_TWC_PS_MEM_x and C_TWP_PS_MEM_x
Note : Timing Parameter is minimum.

ds418_13_121305

Figure 13: **Half-Word Burst Write to 32-bit Asynchronous SRAM**

Half-Word Burst Read from 32-bit Asynchronous SRAM



Tcedv_Tavdv = Maximum of C_TCEDV_PS_MEM_x and C_TAVDV_PS_MEM_x
Note : Timing parameter is minimum.

DS418_14_011306

Figure 14: **Half-Word Burst Read from 32-bit Asynchronous SRAM**

PLB Accesses to 64-bit SRAM

This section shows PLB transactions to a 64-bit SRAM. Note that all transaction types are supported, but timing diagrams are not provided for all.

Single Beat Double-Word Write-Read to 64-Bit Asynchronous SRAM

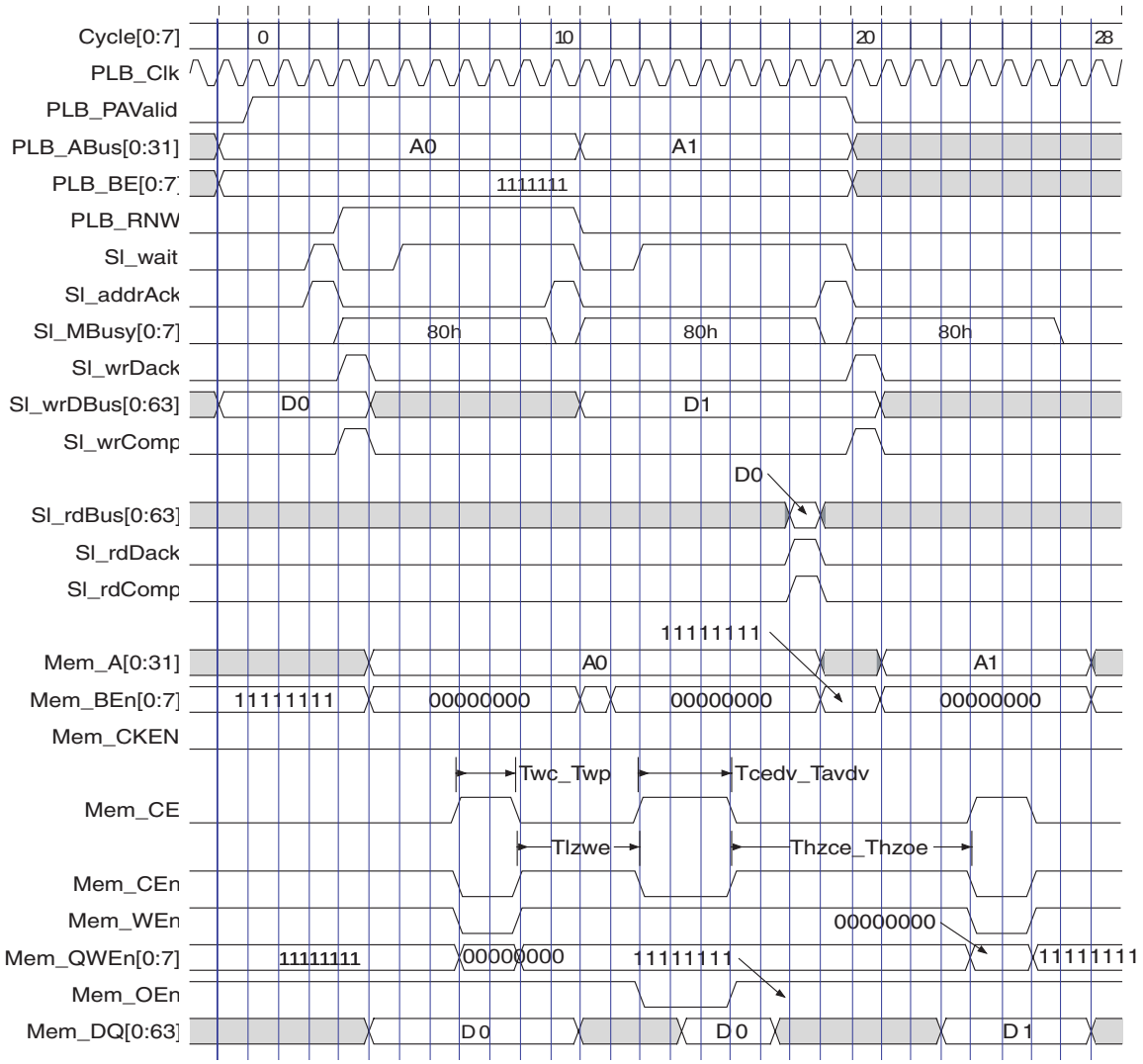
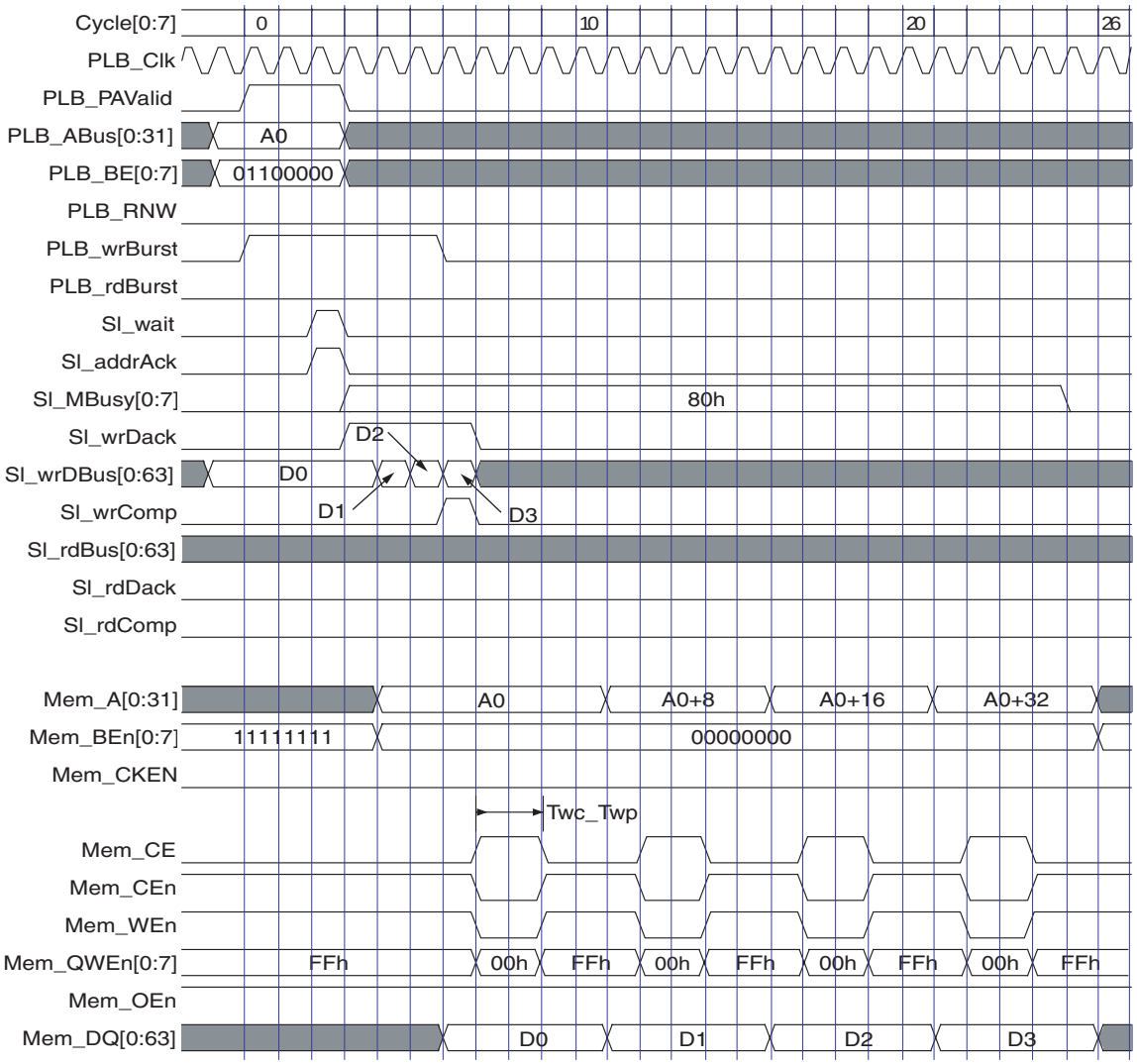


Figure 15: **Single Beat Double-Word Write-Read to 64-Bit Asynchronous SRAM**

Double-Word Burst Write to 64-Bit Asynchronous SRAM

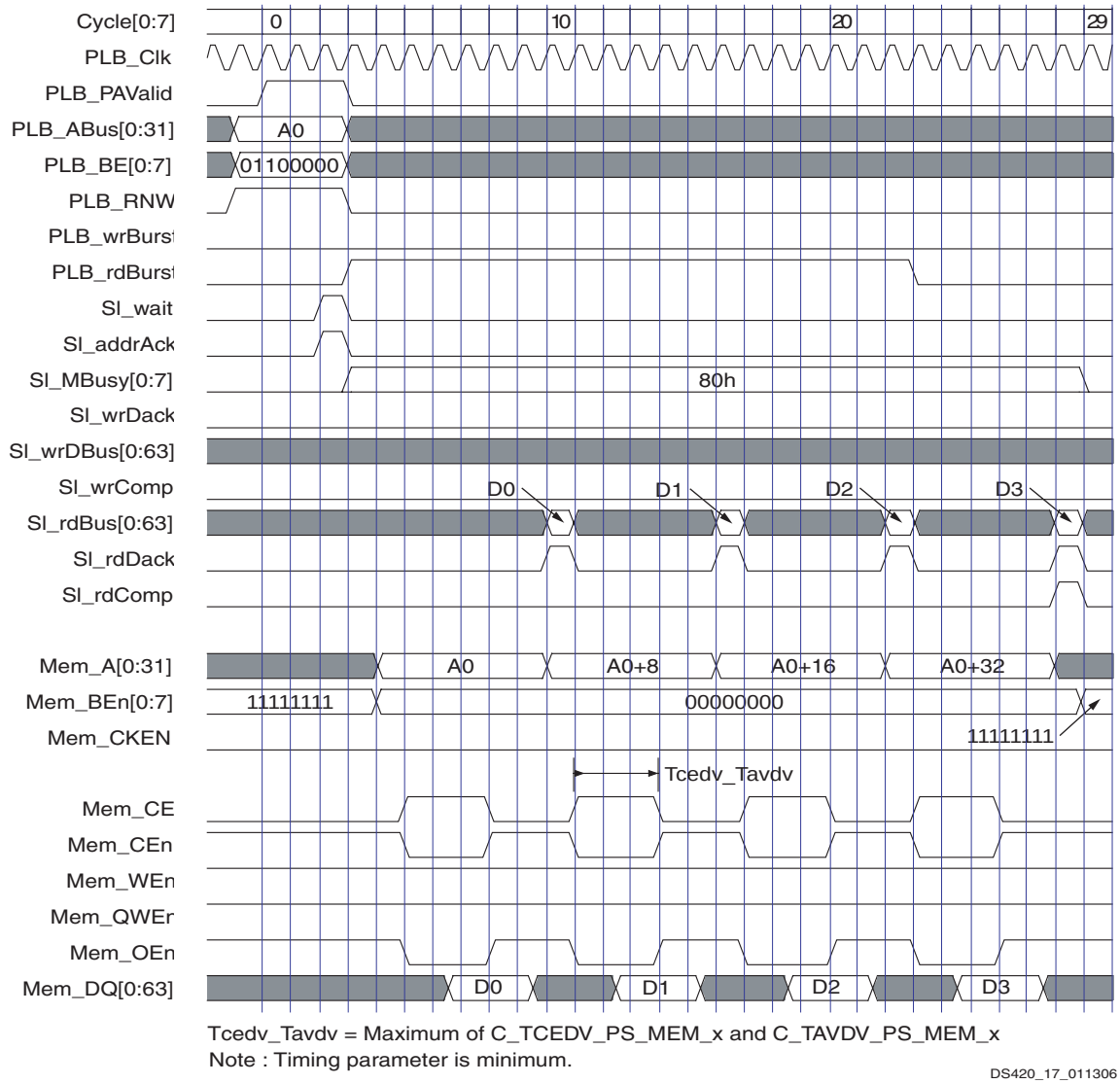


Twc_Twp = Maximum of C_TWC_PS_MEM_x and C_TWP_PS_MEM_x
Note : Timing Parameter is minimum.

DS418_16_011306

Figure 16: **Double-Word Burst Write to 64-Bit Asynchronous SRAM**

Double-Word Burst Read to 64-Bit Asynchronous SRAM

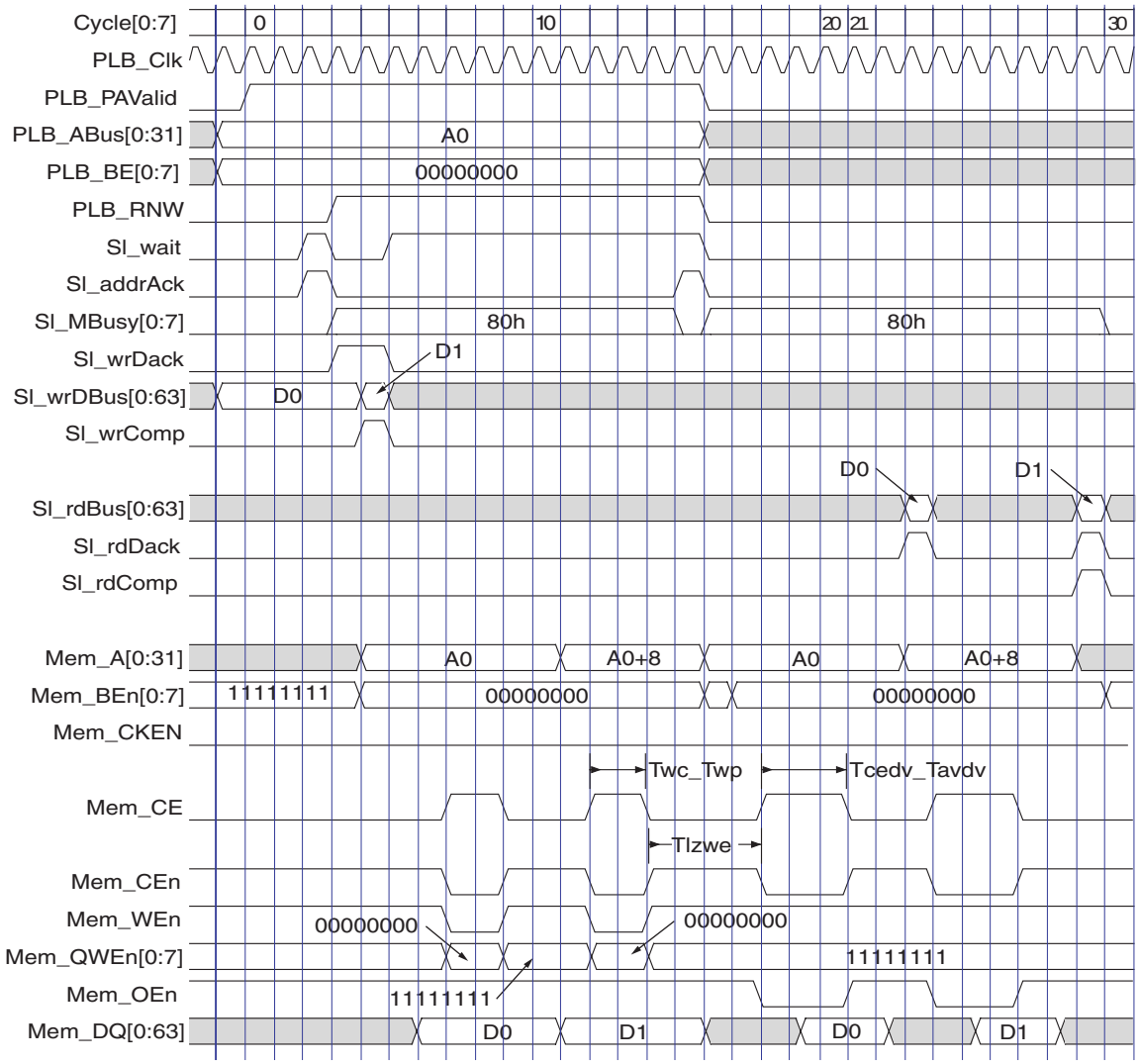


Tcedv_Tavdv = Maximum of C_TCEDV_PS_MEM_x and C_TAVDV_PS_MEM_x
Note : Timing parameter is minimum.

DS420_17_011306

Figure 17: **Double-Word Burst Read to 64-Bit Asnychronous SRAM**

Cacheline Write-Read to 64-Bit Asynchronous SRAM

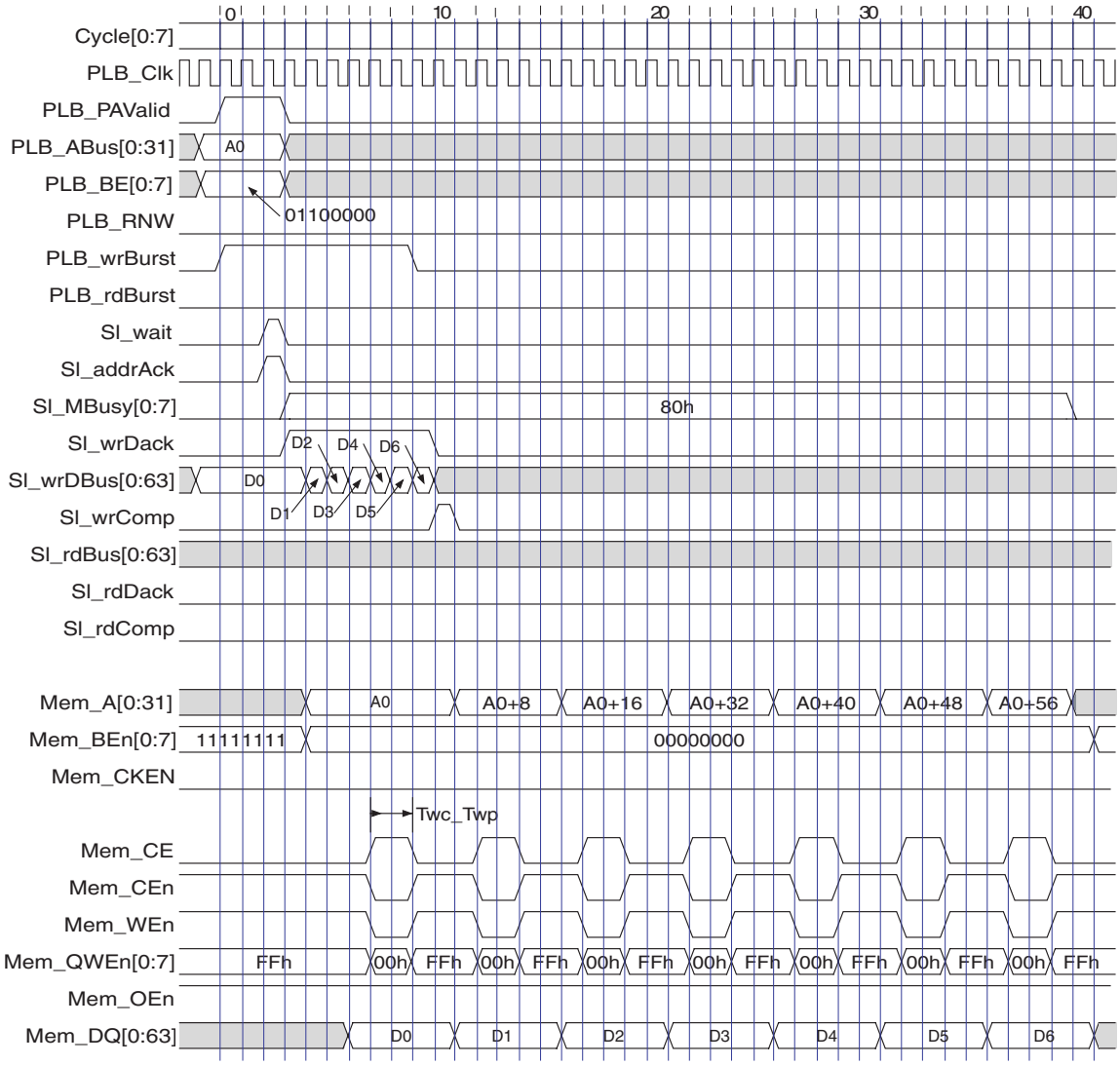


Twc_Twp = Maximum of C_TWC_PS_MEM_x and C_TWP_PS_MEM_x
Tcedv_Tavdv = Maximum of C_TCEDV_PS_MEM_x and C_TAVDV_PS_MEM_x
Thzce_Thzoe = The Maximum of C_THZCE_PS_MEMx and C_THZOE_PS_MEM_x
Tlzwe = C_TLZWE_PS_MEM_x
Note : All timing parameters are minimum.

DS418_18_011306

Figure 18: **Cacheline Write-Read to 64-Bit Asynchronous SRAM**

Indeterminate Burst Write to 64-Bit Asynchronous SRAM

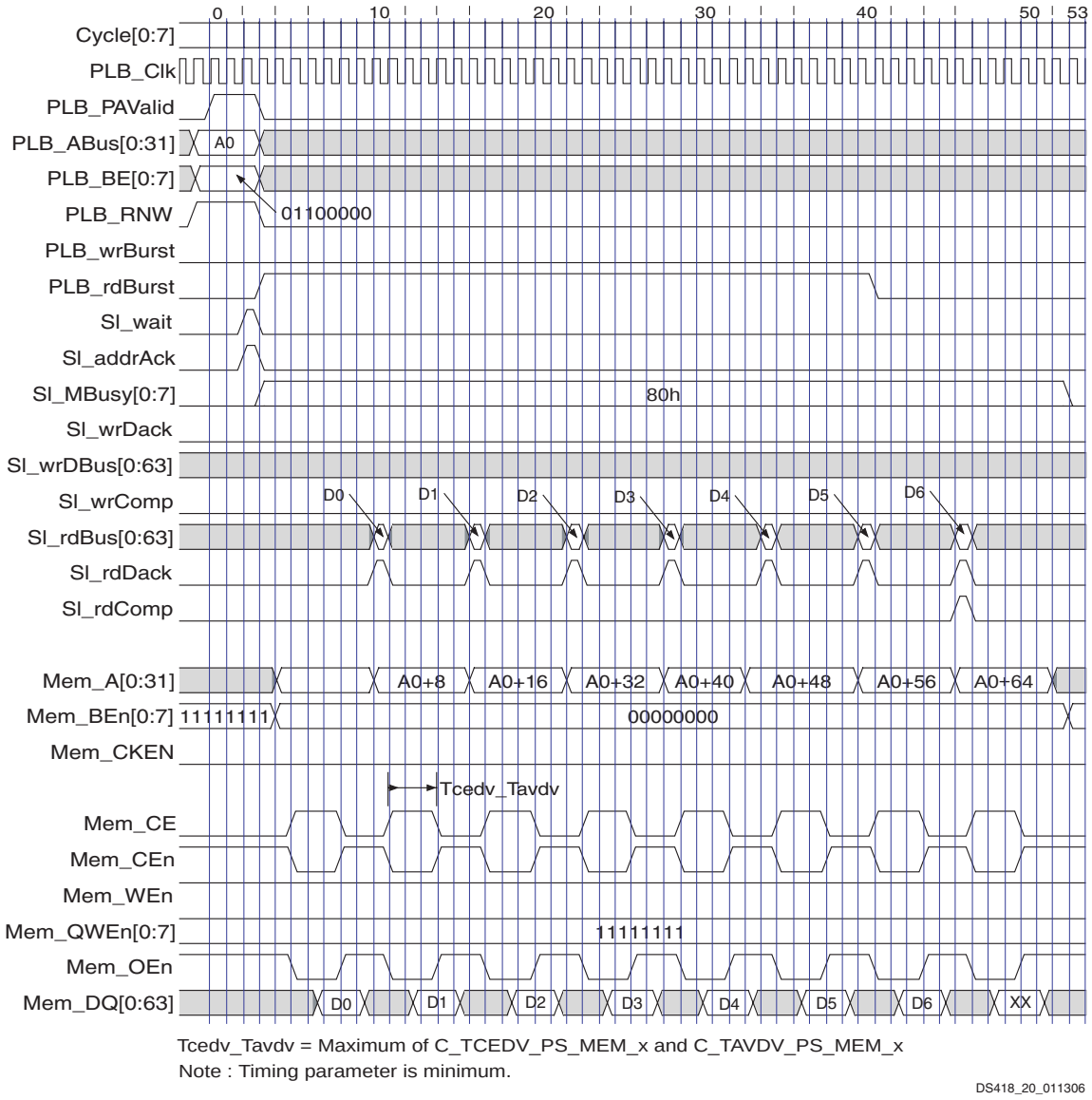


Twc_Twp = Maximum of C_TWC_PS_MEM_x and C_TWP_PS_MEM_x
Note : Timing Parameter is minimum.

DS418_19_011306

Figure 19: **Indeterminate Burst Write to 64-Bit Asynchronous SRAM**

Indeterminate Burst Read to 64-Bit Asynchronous SRAM



Tcedv_Tavdv = Maximum of C_TCEDV_PS_MEM_x and C_TAVDV_PS_MEM_x
Note : Timing parameter is minimum.

DS418_20_011306

Figure 20: **Indeterminate Burst Read to 64-Bit Asynchronous SRAM**

PLB Accesses to 32-bit Pipelined ZBT

This section shows a few PLB transactions to a 32-bit Pipelined ZBT. Note that all transaction types are supported, but timing diagrams are not provided for all.

Single Double-Word Write-Read to 32-Bit Pipelined ZBT

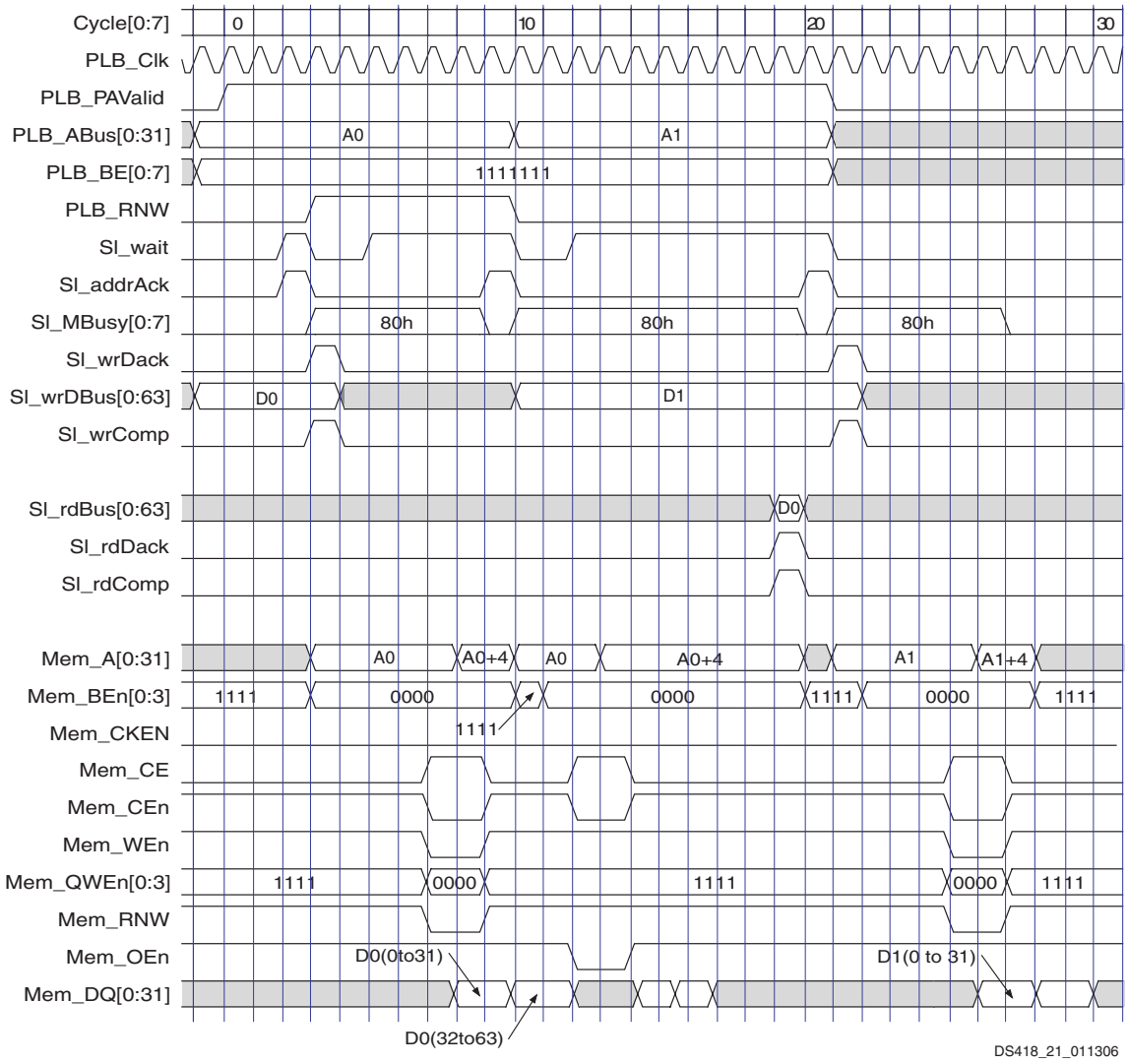
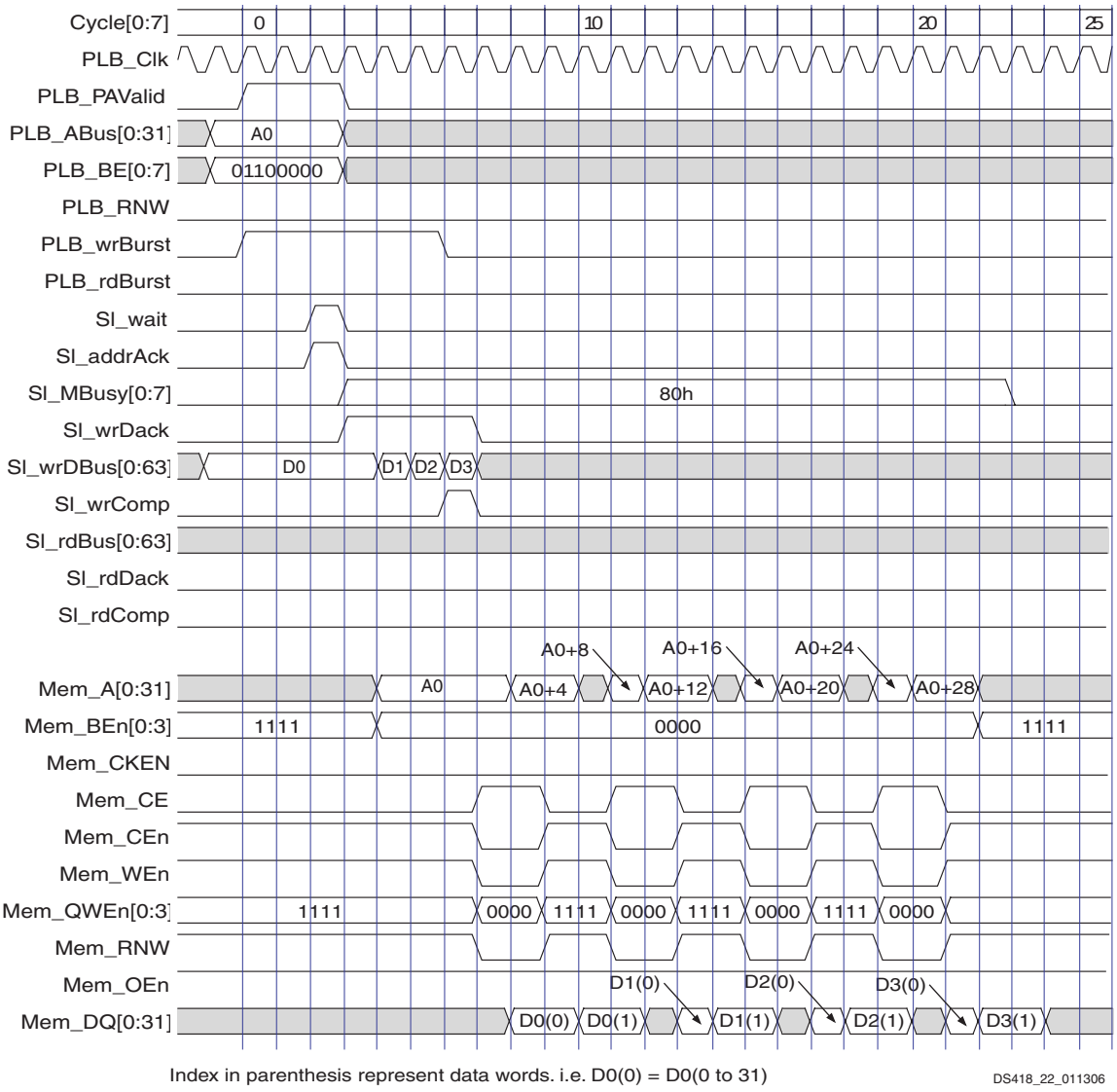


Figure 21: **Single Double-Word Write-Read to 32-Bit Pipelined ZBT**

Double-Word Burst Write to 32-Bit Pipelined ZBT



Index in parenthesis represent data words. i.e. D0(0) = D0(0 to 31)

DS418_22_011306

Figure 22: **Double-Word Burst Write to 32-Bit Pipelined ZBT**

Double-Word Burst Read From 32-Bit Pipelined ZBT

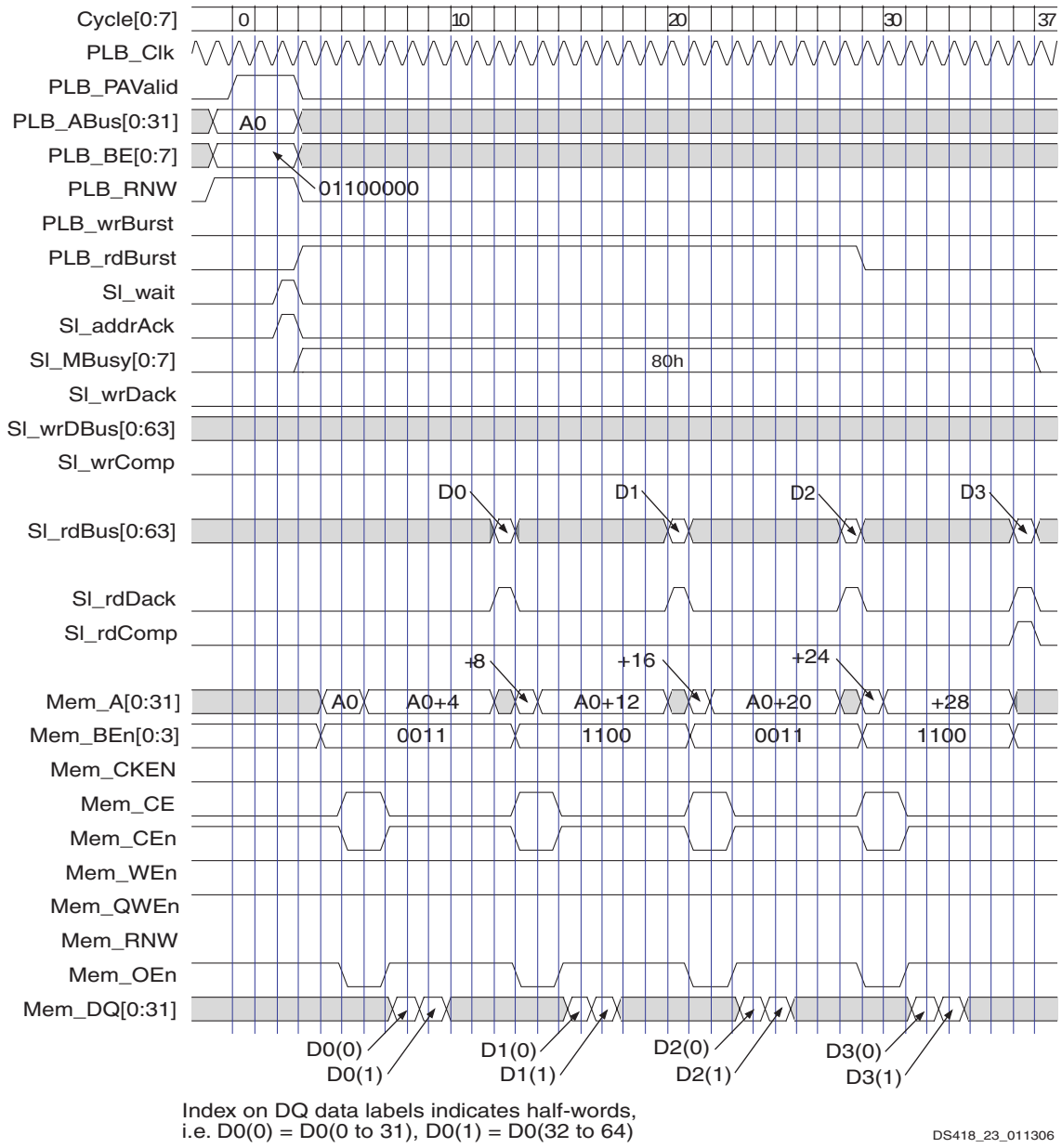


Figure 23: **Double-Word Burst Read From 32-Bit Pipelined ZBT**

Connecting to Memory

Clocking Synchronous Memories

The PLB EMC does not provide a clock output to any synchronous memories. The PLB clock should be routed through an output buffer to provide the clock to the synchronous memories.

To synchronize the synchronous memory clock to the internal FPGA clock, the FPGA system design should include a DCM external to the PLB EMC core that uses the synchronous memory clock input as

the feedback clock as shown in Figure 24. This means that the synchronous clock output from the FPGA must be routed back to the FPGA on a clock pin with a connection to a DCM clock feedback input.

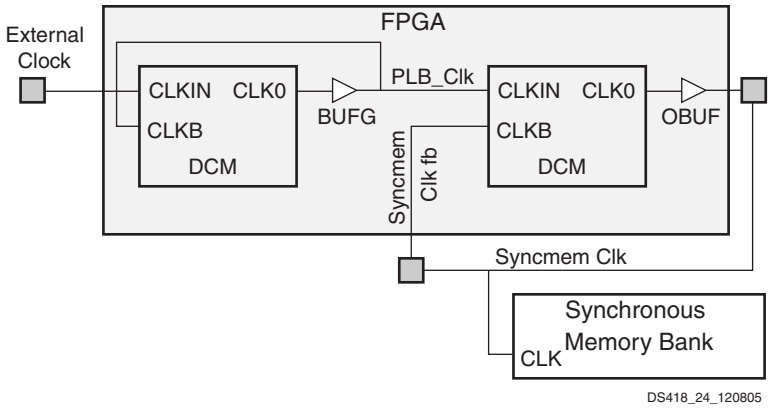


Figure 24: **Synchronous Memory Bank clocked by FPGA Output with feedback**

If the synchronous memory is clocked by the same external clock as the FPGA, or if the clock feedback is not available, the DCM shown in Figure 25 (or something similar) or Figure 26 should be included in the FPGA external to the PLB EMC core.

NOTE: If DLLs are used, the designer must reference XAPP132 v2.4, *Using the Virtex Delay-Locked Loop*, for the correct DLL implementation.

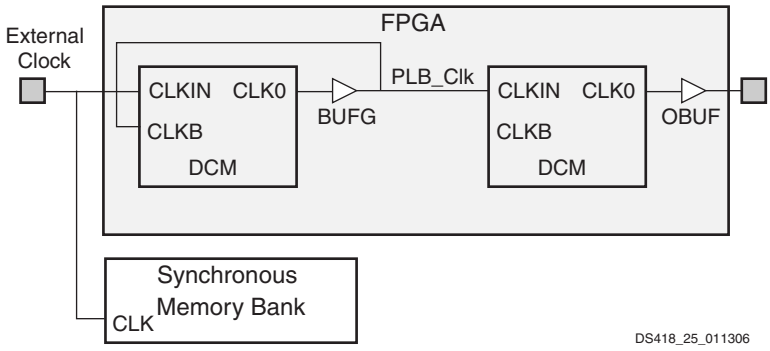


Figure 25: **Synchronous Memory clocked by external clock**

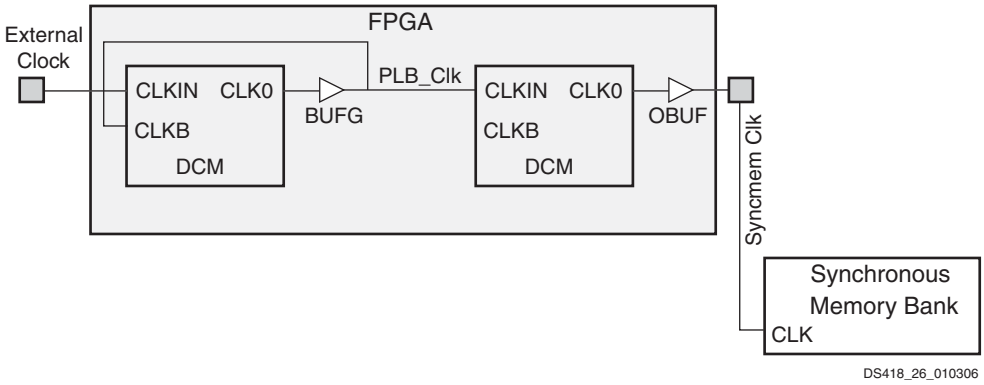


Figure 26: **Synchronous Memory clocked by FPGA Output - no feedback available**

Address Bus, Data Bus, and Control Signal Connections

The three primary considerations for connecting the controller to memory devices are the width of the PLB data bus, the width of the memory subsystem, and the number of memory devices used.

The width of the memory subsystem is simply the maximum width of data that can be read from or written to the memory subsystem. The memory width must be less than or equal to the PLB data bus width.

The data and address signals at the memory 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.

Care must be taken when connecting the chip enable signals. Most asynchronous memory devices will only use Mem_CEN, while most synchronous memory devices will use both Mem_CEN and Mem_CE. Mem_CEN is a function of the address decode while Mem_CE is a function of the state machine logic.

Caution must be exercised with the connections to the external memory devices to avoid incorrect data and address connections. The following tables show the correct mapping of memory controller pins to memory device pins.

Table 6 shows the variables used in defining the memory subsystem.

Table 6: **Variables Used in Defining Memory Subsystem**

Variable	Allowed Range	Definition
BN	0 to 3	Memory bank number
DN	0 to 63	Memory device number within a bank. The memory device attached to the **most significant bit** in the memory subsystem is **0**; device numbers increase toward the least significant bit.
MW	8 to 64	Width in bits of memory subsystem
DW	1 to 64	Width in bits of data bus for memory device
MAW	1 to 32	Width in bits of address bus for memory device

Table 6: **Variables Used in Defining Memory Subsystem *(Contd)***

Variable	Allowed Range	Definition
AU	1 to 64	Width in bits of smallest addressable data word on the memory device
AS	≥ 0	Address shift for address bus = $\log_2$(MW*AU/DW/8)
HAW	1 to 64	Width of PLB address bus in bits

Table 7 shows the EMC to memory interconnect.

Table 7: **EMC to Memory Interconnect**

Description	EMC Signal (MSB:LSB)	Memory Device Signal (MSB:LSB)
Data bus	Mem_DQ(DN*DW:(DN+1)*DW-1)	D(DW-1:0)
Address bus	Mem_A(HAW-MAW-AS:HAW-AS-1)	A(MAW-1:0)
Chip Enable, low-true	Mem_CEN(BN)	CEN
Output Enable, low-true	Mem_OEN	OEN
Write Enable, low-true	Mem_WEN	WEN (for devices that have byte enables or do not require byte enables)
Byte-Enable-Qualified Write Enable, low-true	Mem_QWEN(INT(DN*DW/8))	WEN (for devices that require byte enables and do not have them)
Byte Enable, low-true	Mem_BEN(INT(DN*DW/8): INT((DN+1)*DW/8-1))	BEN(DW/8-1:0)

Example Memory Connections

Example 1

Example 1: Connection to 32-bit memory using 2 IDT71V416S SRAM parts.

Table 8 shows the variables for a simple SRAM example.

Table 8: **Variables for Simple SRAM Example**

Variable	Value	Definition
BN	0	Memory bank number
DN	0 to 1	Memory device number within a bank. The memory device attached to the **most significant bit** in the memory subsystem is **0**; device numbers increase toward the least significant bit.
MW	32	Width in bits of memory subsystem
DW	16	Width in bits of data bus for memory device
MAW	18	Width in bits of address bus for memory device
AU	16	Width in bits of smallest addressable data word on the memory device
AS	2	Address shift for address bus = $\log_2$(MW*AU/DW/8)
HAW	32	Width of host address bus (e.g. OPB or PLB) in bits

Table 9 shows the connections to 32-bit memory using 2 IDT71V416S parts.

Table 9: **Connection to 32-bit Memory Using 2 IDT71V416S Parts**

DN	Description	EMC Signal (MSB:LSB)	Memory Device Signal (MSB:LSB)
0	Data bus	Mem_DQ(0:15)	I/O(15:0)
	Address bus	Mem_A(12:29)	A(17:0)
	Chip Enable, low-true	Mem_CEN(0)	CS
	Output Enable, low-true	Mem_OEN	OE
	Write Enable, low-true	Mem_WEN	WE
	Byte Enable, low-true	Mem_BEN(0:1)	$\overline{\text{BHE}}$:$\overline{\text{BLE}}$
1	Data bus	Mem_DQ(16:31)	I/O(15:0)
	Address bus	Mem_A(12:29)	A(17:0)
	Chip Enable, low-true	Mem_CEN(0)	CS
	Output Enable, low-true	Mem_OEN	OE
	Write Enable, low-true	Mem_WEN	WE
	Byte Enable, low-true	Mem_BEN(2:3)	$\overline{\text{BHE}}$:$\overline{\text{BLE}}$

Connecting to Intel StrataFlash

StrataFlash parts contain an identifier register, a status register, and a command interface, so the bit label ordering for these parts is critical to their proper functioning. The tables below show examples of how to connect the big-endian PLB EMC buses to the little-endian StrataFlash parts.

The proper connection ordering is also indicated in a more general form inTable 10. StrataFlash parts have a x8 mode and a x16 mode, selectable with the BYTE# input pin. To calculate the proper address shift, the minimum addressable word is 8 bits for both x8 and x16 mode, since A0 always selects a byte.

Example 2

Example 2: Connection to 32-bit memory using 2 StrataFlash parts in x16 mode (supports byte read, but no byte write; smallest data type that can be written is 16-bit data).

Table 10 shows variables for StrataFlash (x16mode) example

Table 10: **Variables for StrataFlash (x16 mode) Example**

Variable	Value	Definition
BN	0	Memory bank number
DN	0 to 1	Memory device number within a bank. The memory device attached to the **most significant bit** in the memory subsystem is **0**; device numbers increase toward the least significant bit.
MW	32	Width in bits of memory subsystem
DW	16	Width in bits of data bus for memory device
MAW	24	Width in bits of address bus for memory device

Table 10: **Variables for StrataFlash (x16 mode) Example *(Contd)***

Variable	Value	Definition
AU	8	Width in bits of smallest addressable data word on the memory device
AS	1	Address shift for address bus = $\log_2$(MW*AU/DW/8)
HAW	32	Width of host address bus (e.g. OPB or PLB) in bits

Table 11 shows the connection to 32-bit memory using 2 StrataFlash parts.

Table 11: **Connection to 32-bit Memory Using 2 StrataFlash Parts**

DN	Description	EMC Signal (MSB:LSB)	StrataFlash Signal (MSB:LSB)
0	Data bus	Mem_DQ(0:15)	DQ(15:0)
	Address bus	Mem_A(7:30)	A(23:0)
	Chip Enable, low-true	GND,GND,Mem_CEN(0)	CE(2:0)
	Output Enable, low-true	Mem_OEN	OE#
	Write Enable, low-true	Mem_QWEN(0)	WE#
	Reset/Power down, low-true	Mem_RPN	RP#
	Byte mode select, low-true	N/A - tie to VCC	BYTE#
	Program enable, high-true	N/A - tie to VCC	V_{PEN}
1	Data bus	Mem_DQ(16:31)	DQ(15:0)
	Address bus	Mem_A(7:30)	A(23:0)
	Chip Enable, low-true	GND,GND,Mem_CEN(0)	CE(2:0)
	Output Enable, low-true	Mem_OEN	OE#
	Write Enable, low-true	Mem_QWEN(2)	WE#
	Reset/Power down, low-true	Mem_RPN	RP#
	Byte mode select, low-true	N/A - tie to VCC	BYTE#
	Program enable, high-true	N/A - tie to VCC	V_{PEN}

Example 3

Example 3: Connection to 32-bit memory using 4 StrataFlash parts in x8 mode (supports byte reads and writes).

Table 12 shows the variable for StrataFlash (x8 mode) example.

Table 12: **Variables for StrataFlash (x8 mode) Example**

Variable	Value	Definition
BN	0	Memory bank number
DN	0 to 3	Memory device number within a bank. The memory device attached to the **most significant bit** in the memory subsystem is **0**; device numbers increase toward the least significant bit.
MW	32	Width in bits of memory subsystem

Table 12: **Variables for StrataFlash (x8 mode) Example *(Contd)***

Variable	Value	Definition
DW	8	Width in bits of data bus for memory device
MAW	24	Width in bits of address bus for memory device
AU	8	Width in bits of smallest addressable data word on the memory device
AS	2	Address shift for address bus = $\log_2$(MW*AU/DW/8)
HAW	32	Width of host address bus (e.g. OPB or PLB) in bits

Table 13 shows the connection to 32-bit memory using 4 StrataFlash parts.

Table 13: **Connection to 32-bit Memory Using 4 StrataFlash Parts**

DN	Description	EMC Signal (MSB:LSB)	StrataFlash Signal (MSB:LSB)
0	Data bus	Mem_DQ(0:7)	DQ(7:0)(1)
	Address bus	Mem_A(8:29)	A(23:0)
	Chip Enable, low-true	GND,GND,Mem_CEN(0)	CE(2:0)
	Output Enable, low-true	Mem_OEN	OE#
	Write Enable, low-true	Mem_QWEN(0)	WE#
	Reset/Power down, low-true	Mem_RPN	RP#
	Byte mode select, low-true	N/A - tie to GND	BYTE#
	Program enable, high-true	N/A - tie to VCC	V_{PEN}
1	Data bus	Mem_DQ(8:15)	DQ(7:0)(1)
	Address bus	Mem_A(8:29)	A(23:0)
	Chip Enable, low-true	GND,GND,Mem_CEN(0)	CE(2:0)
	Output Enable, low-true	Mem_OEN	OE#
	Write Enable, low-true	Mem_QWEN(1)	WE#
	Reset/Power down, low-true	Mem_RPN	RP#
	Byte mode select, low-true	N/A - tie to GND	BYTE#
	Program enable, high-true	N/A - tie to VCC	V_{PEN}
2	Data bus	Mem_DQ(16:23)	DQ(7:0)(1)
	Address bus	Mem_A(8:29)	A(23:0)
	Chip Enable, low-true	GND,GND,Mem_CEN(0)	CE(2:0)
	Output Enable, low-true	Mem_OEN	OE#
	Write Enable, low-true	Mem_QWEN(2)	WE#
	Reset/Power down, low-true	Mem_RPN	RP#
	Byte mode select, low-true	N/A - tie to GND	BYTE#
	Program enable, high-true	N/A - tie to VCC	V_{PEN}

Table 13: **Connection to 32-bit Memory Using 4 StrataFlash Parts *(Contd)***

DN	Description	EMC Signal (MSB:LSB)	StrataFlash Signal (MSB:LSB)
3	Data bus	Mem_DQ(24:31)	DQ(7:0)(1)
	Address bus	Mem_A(8:29)	A(23:0)
	Chip Enable, low-true	GND,GND,Mem_CEN(0)	CE(2:0)
	Output Enable, low-true	Mem_OEN	OE#
	Write Enable, low-true	Mem_QWEN(3)	WE#
	Reset/Power down, low-true	Mem_RPN	RP#
	Byte mode select, low-true	N/A - tie to GND	BYTE#
	Program enable, high-true	N/A - tie to VCC	V_{PEN}

Notes:
1. In x8 configuration, DQ(15:8) are not used and should be treated according to manufacturer's data sheet.

Design Constraints

Timing Constraints

A timing constraint should be placed on the system clock, setting the frequency to meet the bus timing requirements. An example is shown below.

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

Pin Constraints

If external pullups/pulldowns are not available on the MEM_DQ signals, then these pins should be specified to use pullup or pulldown resistors. An example is shown below.

```
NET "Mem_DQ<0>" PULLDOWN;
NET "Mem_DQ<1>" PULLDOWN;
. . . . .
NET "Mem_DQ<31>" PULLDOWN;
```

Design Implementation

Target Technology

The intended target technology is a Virtex-II Pro FPGA.

Device Utilization and Performance Benchmarks

This section will be updated when the design has been completed. It will contain the resources and timing for various values of the parameters.

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

The PLB EMC benchmarks are shown in Table 14 for a Virtex-II Pro -7 FPGA.

Table 14: **EMC FPGA Performance and Resource Utilization Benchmarks (Virtex-II Pro -7)**

Parameter Values						Device Resources			f_{MAX}
C_NUM_BANKS_MEM	C_INCLUDE_BURST_CACHELN_SUPPORT	C_MEM_WIDTH	C_SYNCH_MEM_x	C_INCLUDE_DATAWIDTH_MATCHING	C_INCLUDE_NEGEDGE_IOREGS	Slices	Slice Flip-Flops	4-input LUTs	
1	0	8	0	0	0	273	364	253	204.332
1	0	8	0	1	0	353	494	301	204.625
1	0	16	0	0	0	270	390	231	206.271
1	0	16	0	1	0	357	508	297	203.707
1	0	32	0	0	0	286	441	231	205.339
1	0	32	0	1	0	366	540	323	202.429
1	0	64	0	0	0	330	535	188	204.248
1	0	64	0	1	0	343	542	335	202.840
1	1	8	0	0	0	451	441	587	181.785
1	1	8	0	1	0	531	571	637	188.893
1	1	16	0	0	0	446	465	568	192.604
1	1	16	0	1	0	540	586	639	179.244
1	1	32	0	0	0	463	516	567	192.753
1	1	32	0	1	0	551	617	661	198.373
1	0	32	1	0	0	297	489	194	201.369
1	0	32	1	1	0	377	596	282	201.369
1	0	32	1	0	1	344	571	194	209.205
1	0	32	1	1	1	424	678	282	203.957
1	1	32	1	0	1	344	571	194	209.205
1	1	32	1	0	0	471	564	527	197.707
1	1	32	1	1	1	611	755	622	203.004

Table 14: **EMC FPGA Performance and Resource Utilization Benchmarks (Virtex-II Pro -7) *(Contd)***

1	0	64	1	0	0	373	647	151	206.484
1	0	64	1	1	0	389	654	302	201.816
1	1	64	1	1	0	568	732	641	200.602
1	1	64	1	1	1	634	854	641	169.549
2	1	8,64	1,0	1,0	1	751	974	861	188.466
2	1	8,64	0,1	0,0	0	657	755	775	184.366
4	1	8,16, 32,32	0,1,0,1	1,1,0,0	0	874	858	1,177	120.934

Notes:
1. These benchmark designs contain only the EMC 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 EMC design:

IBM PLB, Architectural Specification (v3.4)

Revision History

Date	Version	Revision
05/22/02	1.0	Initial Xilinx release.
06/04/02	1.1	Update for EDK 1.0
07/29/02	1.2	Add XCO parameters for System Generator
12/30/02	1.3	Updated document for PLB EMC core version 1.10a which added data-width matching and bursting
01/08/03	1.4	Update for EDK SP3
03/20/03	1.5	Updated document for PLB EMC core version 1.10b which provided a parameter to allow user to choose negative edge or positive edge IO registers.
07/14/03	1.6	Update to new template
07/28/03	1.6.1	Change DS number because of duplications
10/13/03	1.7	Updated Table 16benchmarks.
12/15/03	1.7.1	Remove unused signal.
03/11/04	1.8	Added timing diagrams
04/08/04	2.0	Updated document for EMC core version 2.00a which provided FPGA resource optimization and customer ease of use. General clean up.
08/26/04	2.1	Resource utilization update. General clean up.

Date	Version	Revision
7/14/05	2.3	Updated to incorporate CR204255 (removed C-FAMILY generic); put/published 071405.
1/13/06	2.4	Converted to new DS template; updated images to graphic standards.
3/1/06	2.5	In Table 2, Allowable Values for C_INLUDE_BURST_CACHELN_SUPPORT was: 0 = don't include negative edge IO registers (data and control signals are input/output on the rising edge of the clock) 1 = include negative edge IO registers (data and control signals are input/output on the falling edge of the clock)