Multi-Channel (MCH) PLBv46 Slave Burst (v1.00a)

Introduction

The Xilinx Multi-Channel (MCH) and PLBv46 Slave Burst (MCH_PLBv46_Slave_Burst) provides a bi-directional interface between a parameterizable number of channel interfaces and an PLBv46 Slave Burst interface to an IP core.

The channel interfaces can be configured with a Xilinx Cachelink (XCL) protocol to provide a direct cacheline interface from processors to external memories.

Features

The MCH_PLBv46_Slave_Burst is a soft IP core designed for Xilinx FPGAs and contains the following features:

- Parameterizable inclusion of PLB slave interface compatible with IBM CoreConnect PLBV4.6 buses of 32, 64, 128-bits.
- Parameterizable number of channel interfaces.
  - Each channel can be configured with a Xilinx Cachelink (XCL) protocol. Each XCL channel provides parameterizable:
    - Cacheline size (1, 4, 8, or 16 words)
    - Single, cacheline, or no write transactions
- Parameterizable depth of channel buffers.
- Fixed arbitration mode between channel interfaces and PLB interface.
- Supports target-word first XCL cache-line transactions of 1, 4, 8 and 16 words.
- Supports target-word first PLB Cacheline read and line-word first PLB Cacheline write transactions of 4, 8 and 16 words.
- Supports low latency PLB Point-to-Point topology.
- Design optimized to minimize latency.

LogiCORE™ IP Facts		
Core Specifics		
Supported Device Family	See EDK Supported Device Families.	
Version of Core	mch_plbv46_slave_burst	v1.00a
Resources Used		
	Min	Max
Slices	Refer to the Table 10	
LUTs		
FFs		
Block RAMs	N/A	
Special Features	N/A	
Provided with Core		
Documentation	Product Specification	
Design File Formats	VHDL	
Constraints File	N/A	
Verification	N/A	
Instantiation Template	N/A	
Reference Designs & Application notes	N/A	
Additional Items	N/A	
Design Tool Requirements		
Xilinx Implementation Tools	See Tools for requirements.	
Verification		
Simulation		
Synthesis		
Support		
Provided by Xilinx, Inc.		

Functional Description

The MCH_PLBv46_Slave_Burst is designed to provide multiple channel interfaces and an optional PLB interface to an IP core. The block diagram in Figure 1 shows an example use of the MCH_PLBv46_Slave_Burst with an SDRAM, EMC, or DDR memory controller in a MicroBlaze™ system.

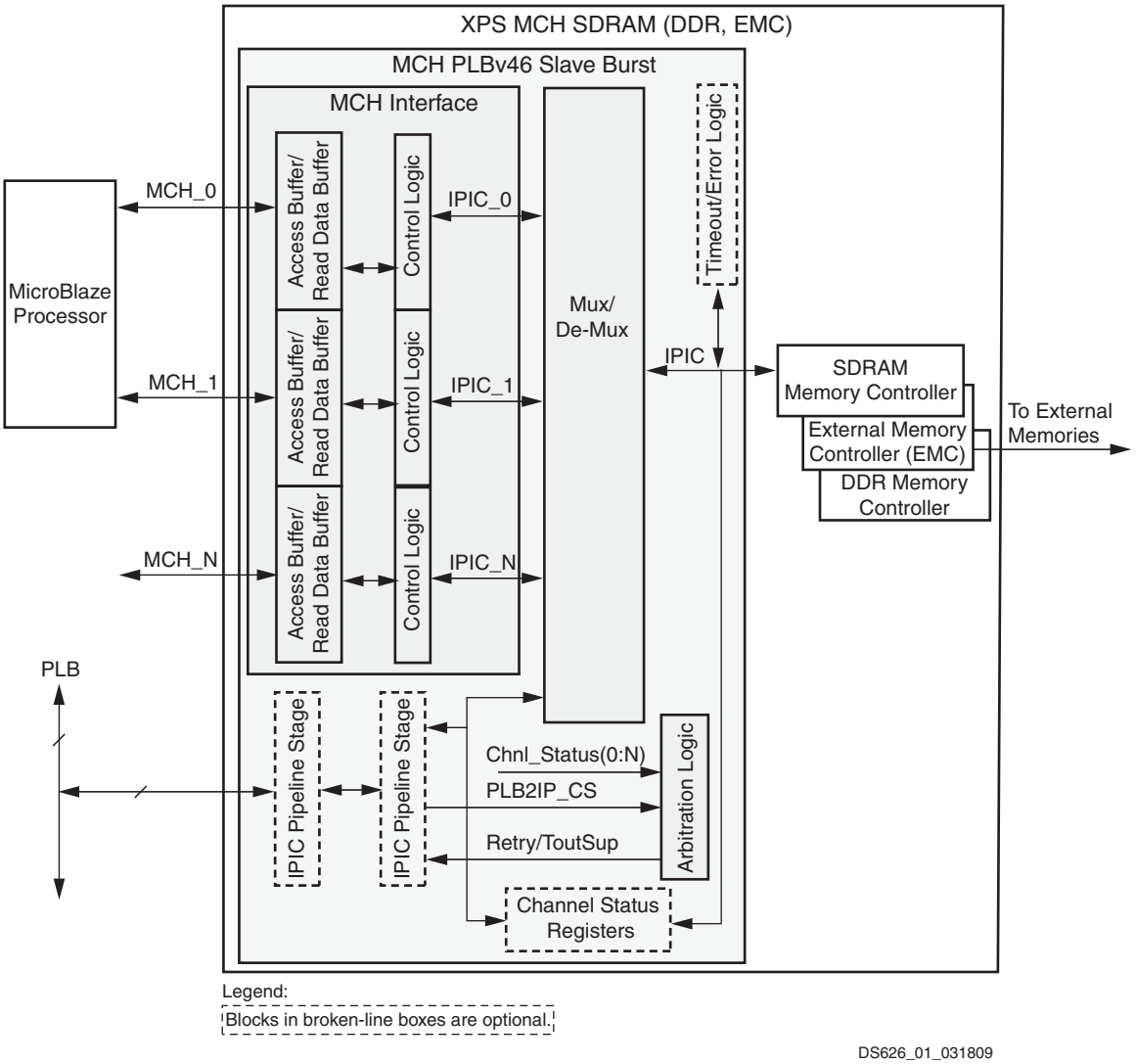

Figure 1: **MCH PLBV46 Slave Burst Block Diagram**

The MCH_PLBv46_Slave_Burst consists of an optional PLB slave interface, a parameterizable number of channel interfaces, arbitration logic to select between the channel interfaces and the PLB interface and the appropriate multiplexors/de-multiplexors to connect the selected channel or PLB transaction to the IP core. An optional timer can be included in the MCH_PLBv46_Slave_Burst to return an error if the IP core does not return an acknowledge to a channel read transaction within a parameterizable number of clocks.

Each channel consists of 2 buffers per channel, an Access buffer that contains the information about the requested transaction, and a ReadData buffer that contains the resulting data from a Read transaction.

A cacheline access is requested by writing the transaction information into the Access Buffer of the channel. The Channel Logic monitors this buffer, gets the transaction from the Access Buffer, then converts it to the proper IPIC signals. The Arbitration Logic monitors the channel interfaces and the PLB interface and arbitrates between the pending transactions. Based on the arbitration scheme chosen, the highest priority transaction is passed through the IPIC Mux/DeMux block to the attached IP. The IPIC responses are passed back to the appropriate channel or to the PLB Slave Burst.

The advantage of having Channel Logic for each channel is that arbitration is done on the IPIC signal set, thus there is no delay after arbitration for decoding of the channel signals into the appropriate IPIC signals. An additional advantage is that the logic to generate the addresses for the channel is parameterized to only provide the addressing scheme required by that channel and there is no delay in dynamically loading and switching the address generation mode.

In the example system shown in Figure 1, the Microblaze processor has two MCH interfaces – one for the instruction cache and one for the data cache. Both of these channels would have be set to have the XCL protocol. Because the instruction cache performs read transactions only and will consumes data as soon as it is available, set the entry in C_XCL_WRITEXFER_ARRAY for this channel to 0, no write transfers. Also, set the entry in C_MCH_RDDATABUF_DEPTH_ARRAY to zero so that there will be no additional latency in reading data from a buffer when it is available.

PLB Slave Interface

The PLB Slave Interface is provided by the PLBv46 Slave Burst IP Core described in DS562 PLBv46 Slave Burst V1.00.a Product Specification and will not be described in this document. Note that the IPIC signal protocol implemented by the MCH_PLBv46_Slave_Burst is also described in DS562 PLBv46 Slave Burst V1.00.a Product Specification. The PLBv46 Slave Burst IP Core will be configured in a Slave-only mode utilizing only those portions of the PLBv46 Slave Burst required for attachment to the current memory controller cores. If the optional Timeout/Error logic is included in the design, the output from this logic can generate an interrupt, utilizing the interrupt service provided by the PLBv46 Slave Burst.

Xilinx Multi-Channel (MCH) Interfaces

Each channel can be configured to support the Xilinx Cachelink (XCL) protocol.

XCL channels are designed to interface to processors for cacheline accesses to and from memory. An XCL channel only performs cacheline read transactions of length C_MCH_CACHESIZE_ARRAY(n). Write transactions are configured by the C_XCL_WRITE_XFERS_ARRAY. If the XCL channel is connected to an Instruction Cache which would not perform write transfers, then C_XCL_WRITE_XFERS_ARRAY(n) = 0. All write transactions are single writes when C_XCL_WRITEXFERS_ARRAY(n) = 1. All write transactions are cacheline write transactions of length C_XCL_LINESIZE_ARRAY(n) when C_XCL_WRITEXFERS_ARRAY(n) = 2. Cacheline read and cacheline write transactions are always word-width transfers. Single writes can be byte, half-word, and word-width transfers.

MCH_PLBv46_Slave_Burst I/O Signals

Table 1 provides a summary of all MCH_PLBv46_Slave_Burst input/output (I/O) signals, the interfaces under which they are grouped, and a brief description of the signal.

Table 1: **MCH_PLBv46_Slave_Burst IO Descriptions**

Port	Signal Name	Interface	I/O	Initial State	Description
System					
P1	SPLB_Clk	System	I	-	System clock.
P2	SPLB_Rst	System	I	-	System reset.
MCH Signals					
P3	MCH_Access_Control(0:C_NUM_CHANNELS-1)	MCH	I	-	Control signal to the Access buffers of the channel interfaces. This signal indicates the type of access to be performed (read or write) and the size of the access (byte, halfword, or word).
P4	MCH_Access_Data(0:C_NUM_CHANNELS*C_MCH_SIPIF_DWIDTH-1)	MCH	I	-	Write Data to the Access buffers of the channel interfaces.
P5	MCH_Access_Write(0:C_NUM_CHANNELS-1)	MCH	I	-	Write signals to Access buffers of the channel interfaces.
P6	MCH_Access_Full(0:C_NUM_CHANNELS-1)	MCH	O	0	Indicator that the Access buffers of the channel interfaces are full.
P7	MCH_ReadData_Control (0:C_NUM_CHANNELS-1)	MCH	O	1	Control signals for the ReadData buffers of the channel interfaces. These signals indicate if the data from the ReadData buffer is valid.
P8	MCH_ReadData_Data(0:C_MCH_SIPIF_DWIDTH*C_NUM_CHANNELS-1)	MCH	O	Zeros	Read data from the ReadData buffers of the channel interfaces.
P9	MCH_ReadData_Read(0:C_NUM_CHANNELS-1)	MCH	I	-	Read signals to the ReadData buffers of the channel interfaces.
P10	MCH_ReadData_Exists(0:C_NUM_CHANNELS-1)	MCH	O	0	Indicator that the ReadData buffers of the channel interfaces are non-empty.
PLB Slave Signals					
P11	PLB_ABus(0:31)	PLB Bus	I	-	See table note(1).
P12	PLB_PAValid	PLB Bus	I	-	See table note(1).
P13	PLB_masterID(0:C_SPLB_MID_WIDTH-1)	PLB Bus	I	-	See table note(1).
P14	PLB_RNW	PLB Bus	I	-	See table note(1).
P15	PLB_BE(0:[C_SPLB_DWIDTH/8]-1)	PLB Bus	I	-	See table note(1).
P16	PLB_MSize(0:1)	PLB Bus	I	-	See table note(1).
P17	PLB_size(0:3)	PLB Bus	I	-	See table note(1).
P18	PLB_type(0:2)	PLB Bus	I	-	See table note(1).
P19	PLB_wrDBus(0:C_SPLB_DWIDTH-1)	PLB Bus	I	-	See table note(1).
P20	PLB_wrBurst	PLB Bus	I	-	See table note(1).

Table 1: **MCH_PLBv46_Slave_Burst IO Descriptions *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P21	PLB_rdBurst	PLB Bus	I	-	See table note(1).
P22	Sl_addrAck	PLB Bus	O	0	See table note(1).
P23	Sl_SSize(0:1)	PLB Bus	O	0	See table note(1).
P24	Sl_wait	PLB Bus	O	0	See table note(1).
P25	Sl_rearbitrate	PLB Bus	O	0	See table note(1).
P26	Sl_wrDack	PLB Bus	O	0	See table note(1).
P27	Sl_wrComp	PLB Bus	O	0	See table note(1).
P28	Sl_wrBTerm	PLB Bus	O	0	See table note(1).
P29	Sl_rdBus(0:C_SPLB_DWIDTH-1)	PLB Bus	O	0	See table note(1).
P30	Sl_rdDAck	PLB Bus	O	0	See table note(1).
P31	Sl_rdComp	PLB Bus	O	0	See table note(1).
P32	Sl_rdBTerm	PLB Bus	O	0	See table note(1).
P33	Sl_rdWdAddr(0:3)	PLB Bus	O	0	See table note(1).
P34	Sl_MBusy(0:C_SPLB_NUM_MASTERS-1)	PLB Bus	O	0	See table note(1).
P35	Sl_MWrErr(0:C_SPLB_NUM_MASTERS-1)	PLB Bus	O	0	See table note(1).
P36	Sl_MRdErr(0:C_SPLB_NUM_MASTERS-1)	PLB Bus	O	0	See table note(1).
Unused PLB Signals					
P37	PLB_UABus(0:31))	PLB Bus	I	-	Unused.
P38	PLB_SAValid	PLB Bus	I	-	Unused.
P39	PLB_rdPrim	PLB Bus	I	-	Unused.
P40	PLB_wrPrim	PLB Bus	I	-	Unused.
P41	PLB_abort	PLB Bus	I	-	Unused.
P42	PLB_busLock	PLB Bus	I	-	Unused.
P43	PLB_TAttribute(0:15)	PLB Bus	I	-	Unused.
P44	PLB_lockerr	PLB Bus	I	-	Unused.
P45	PLB_wrPendReq	PLB Bus	I	-	Unused.
P46	PLB_rdPendReq	PLB Bus	I	-	Unused.
P47	PLB_rdPendPri(0:1)	PLB Bus	I	-	Unused.
P48	PLB_wrPendPri(0,1)	PLB Bus	I	-	Unused.
P49	PLB_reqPri(0:1)	PLB Bus	I	-	Unused.
P50	Sl_MIRQ(0:C_SPLB_NUM_MASTERS-1)	PLB Bus	O	0	Unused.
IPIC					
P51	Bus2IP_Clk	IP Core	O	0	Synchronization clock provided to User IP. This is the same as SPLB_Clk.

Table 1: **MCH_PLBv46_Slave_Burst IO Descriptions *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P52	Bus2IP_Reset	IP Core	O	0	Active high reset for use by the User IP. It is a pass through of the SPLB_Rst input.
P53	Bus2IP_Data(0:C_MCH_SIPIF_DWIDTH-1)	IP Core	O	-	Write data bus to the user IP. Write data is accepted by the IP by assertion of the IP2Bus_Ack signal at the rising edge of the Bus2IP_Clk (or unconditionally on the cycle presented for posted writes).
P54	Bus2IP_Addr(0:C_SPLB_AWIDTH-1)	IP Core	O	-	Address bus indicating the desired address of the requested read or write operation.
P55	Bus2IP_RNW	IP Core	O	-	This signal indicates the sense of a requested operation with the user IP. High is a read, low is a write.
P56	Bus2IP_BE(0:(C_MCH_SIPIF_DWIDTH/8)-1)	IP Core	O	-	Byte enable qualifiers for the requested read or write operation with the user IP. Bit 0 corresponds to Byte lane 0, Bit 1 to Byte lane 1, and so on.
P57	Bus2IP_Burst	IP Core	O	-	Active high signal indicating that the active read or write operation with the user IP is utilizing bursting protocol. This signal is asserted at the initiation of a burst transaction with the user IP and de-asserted at the completion of the second to last data beat of the burst data transfer.
P58	Bus2IP_BurstLength(0:$\log_2$(16*C_MCH_SIPIF_DWIDTH/8))(2)	User IP	O	0	This value is an indication of the number of bytes being requested for transfer aligned with data phase and is valid when the cycle is of burst type when Bus2IP_CS=1.
P59	Bus2IP_AddrBurstLength(0:$\log_2$(16*C_MCH_SIPIF_DWIDTH/8))	User IP	O	0	This value is an indication of the number of bytes being requested for transfer aligned with address phase and is valid when the cycle is of burst type when Bus2IP_CS=1.
P60	Bus2IP_AddrBurstCntLoad	User IP	O	0	
P61	Bus2IP_CS(0:((C_PLB_ARD_ADDR_RANGE_ARRAY'LENGTH)/2)-1)	IP Core	O	0	Active High chip select bus. Each bit of the bus corresponds to an entry in the C_ARD_ID_ARRAY. Assertion of a chip select indicates a active transaction request to the chip select's target address space.
P62	Bus2IP_WrReq	User IP	O	0	Active high signal indicating the initiation of a write operation with the IP. It is asserted for 1 Bus2IP_Clk during single data beat transactions and remains high to completion on burst write operations

Table 1: **MCH_PLBv46_Slave_Burst IO Descriptions *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P63	Bus2IP_RdReq	User IP	O	0	Active high signal indicating the initiation of a read operation with the IP. It is asserted for 1 Bus2IP_Clk during single data beat transactions and remains high to completion on burst read operations.
P64	Bus2IP_RdCE(0: see note(3))	IP Core	O	0	Active high chip enable bus. Chip enables are assigned per the user's entries in the C_PLB_ARD_NUM_CE_ARRAY. These chip enables are asserted only during active read transaction requests with the target address space and in conjunction with the corresponding sub-address within the space.
P65	Bus2IP_WrCE(0: see note(3))	IP Core	O	0	Active high chip enable bus. Chip enables are assigned per the user's entries in the C_PLB_ARD_NUM_CE_ARRAY. These chip enables are asserted only during active write transaction requests with the target address space and in conjunction with the corresponding sub-address within the space.
P66	IP2Bus_Data(0:C_MCH_SIPIF_DWIDTH-1)	IP Core	I	-	Input Read Data bus from the user IP. Data is qualified with the assertion of IP2Bus_Ack signal and the rising edge of the Bus2IP_Clk.
P67	IP2Bus_AddrAck	IP Core	I	-	Active high signal that advances the IPIF Address counter during multiple data beat transfers.
P68	IP2Bus_WrAck	User IP	I	-	Active high Write Data qualifier. Write data on the Bus2IP_Data Bus is deemed accepted by the User IP at the rising edge of the Bus2IP_Clk and IP2Bus_WrAck asserted high by the User IP.
P69	IP2Bus_RdAck	User IP	I	-	Active high read data qualifier. Read data on the IP2Bus_Data Bus is deemed valid at the rising edge of Bus2IP_Clk and the assertion of the IP2Bus_RdAck signal by the User IP.
P70	IP2Bus_Error	IP Core	I	-	Active high signal indicating the user IP has encountered an error with the requested operation. This signal is asserted in conjunction with IP2Bus_Ack.

1. This signal's function and timing is defined in the IBM® 128-Bit Processor Local Bus Architecture Specification Version 4.6.
2. Log_2 represents a logarithm function of base 2. For example, $log_2(1)=0$, $log_2(2)=1$, $log_2(4)=2$, $log_2(8)=3$, $log_2(16)=4$, etc.
3. The size of the Bus2IP_RdCE and the Bus2IP_WrCE buses is the sum of the integer values entered in the **C_PLB_ARD_NUM_CE_ARRAY**.

MCH_PLBv46_Slave_Burst Design Parameters

The MCH_PLBv46_Slave_Burst provides for tailoring of the core via VHDL generic parameters. These parameters are detailed in Table 2.

Table 2: **MCH_PLBv46_Slave_Burst Design Parameters**

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
General					
G1	Target FPGA family	C_FAMILY	spartan3, spartan3e, spartan3a, spartan3an, virtex4, qvirtex4, qrvirtex4, virtex5	virtex5	string
G2	Include PLB Interface	C_INCLUDE_PIB_IPIF	0 = don't include PLB IPIF 1 = include PLB IPIF	1	integer
Arbiter					
G3	Arbitration Mode	C_PRIORITY_MODE(1)	0 = fixed priority mode	0	integer
Multi-channel Interface					
G4	Number of channel interfaces	C_NUM_CHANNELS	0 - 4	2	integer
G5	Address width of plbv46/channel interfaces	C_MCH_SPLB_AWIDTH	32	32	integer
G6	Address width of slave interfaces	C_MCH_SIPIF_DWIDTH	32	32	integer
G7	Array indicating the supported protocol of each channel	C_MCH_PROTOCOL_ARRAY(2)	One entry for each channel. Allowed entries are: 0 = XCL protocol	{0}	integer array
G8	Array indicating the address ranges of User IP available to MCH interfaces. The address ranges in this array correspond the to Bus2IP_CS generated from the MCH interfaces.	C_MCH_USERIP_ADDRRANGE_ARRAY(3)(4)	Valid base address and high address pairs	User must set values	SLV64_ARRAY_TYPE
G9	Array indicating the depth of the Access buffer for each channel	C_MCH_ACCESSBUF_DEPTH_ARRAY	One entry for each channel. Allowed entries are: 4, 8, 16	{16}	integer array
G10	Array indicating the depth of the ReadData buffer for each channel	C_MCH_RDDATABUF_DEPTH_ARRAY(5)	One entry for each channel. Allowed entries are: 0,4, 8, 16	{16}	integer array

Table 2: **MCH_PLBv46_Slave_Burst Design Parameters *(Cont'd)***

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
XCL Channels					
G11	Array indicating the size of the cacheline in number of 32-bit words for each channel configured with the XCL protocol (1)	C_XCL_LINESIZE _ARRAY(6)	One entry for each XCL channel. Allowed entries: 1,4,8,16 for XCL channels	{4}	integer array
G12	Array indicating the type of write transactions for each channel configured with the XCL protocol	C_XCL_WRITEXFER _ARRAY(6)(7)	One entry for each XCL channel. Allowed entries: 0 = no write transfers 1 = single transfers only 2 = cacheline transfers only	{1}	integer array
PLB Interface					
G13	Data width of plbv46 interfaces	C_SPLB_DWIDTH	32, 64, 128	32	integer
G14	Selects point-to-point bus topology	C_SPLB_P2P	0,1	0	integer
G15	Cache Line Addressing Mode	C_CACHLINE_ADDR_MO DE	0 = Target word first on reads. 1 = Line word First on reads	0	integer
G16	Write Buffer Depth	C_WR_BUFFER_DEPTH	0, 16, 32, or 64 0 = no write buffer implemented	16	integer
G17	Data width of the smallest master	C_SPLB_SMALLEST_MA STER	32, 64, 128		integer
G18	Number of PLB Masters	C_SPLB_NUM_MASTERS	1 - 16	1	integer
G19	PLB Master ID Bus width	C_SPLB_MID_WIDTH	0 - 4	1	integer

Table 2: **MCH_PLBv46_Slave_Burst Design Parameters *(Cont'd)***

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
G20	Array of Base Address / High Address Pairs for each Address Range. The address ranges in this array correspond the to Bus2IP_CS generated from the PLBv46 Slave Burst.	C_PLB_ARD_ADDR_RANGE_ARRAY(8) (9)(10)	See the PLBv46 Slave Burst V1_00_a Product Specification for description.	User must set values.	SLV64_ARRAY_TYPE
G21	Array of the desired number of chip enables for each address space	C_PLB_ARD_NUM_CE_ARRAY(8) (9)	See the PLBv46 Slave Burst V1_00_a Product Specification for description.	User must set values.	integer array

1. Only fixed arbitration is supported at this time.
2. Each channel can be configured to support a transfer protocol. Only the Xilinx Cachelink (XCL) protocol is supported at this time.
3. This array should only contain user IP address ranges which are accessible through the MCH interfaces. The order of entries through this array should match the order of entries in C_PLB_ARD_ADDR_RANGE_ARRAY.
4. Best performance (higher F_{MAX}, lower latency) is achieved when there is only one user IP address range accessible through the MCH interfaces.
5. If the channel is connected to a master which can consume data as soon as it is available (i.e., instruction side interfaces), set the depth of the read data buffer to zero for that channel to save resources and latency.
6. If C_MCH_CHANNEL_PROTOCOL(x) is not set to XCL, then entry x in these arrays is unused.
7. If the master connecting to channel x will only perform read transfers, (i.e., instruction cache masters), set entry x in this array to 0.
8. The generics associated with the PLB Slave interface are unused when C_INCLUDE_PLB_IPIF=0.
9. The generics associated with the PLB Slave Interface are described in the PLBv46 Slave Burst Parameter Detailed Description section of the Xilinx LogiCORE DS424 PLBv46 Slave Burst v1.00.A Product Specification and will not be described in this document.
10. User IP address ranges which are also accessible through MCH interfaces should be listed first in these arrays.

Allowable Parameter Combinations

The PLB slave interface is only included in this design if C_INCLUDE_PLB_IPIF is set to 1. When C_INCLUDE_PLB_IPIF = 0, the generics associated with the PLB interface (C_SPLB_P2P, C_CACHLINE_ADDR_MODE, C_WR_BUFFER_DEPTH, C_PLB_ARD_ADDR_RANGE_ARRAY and C_PLB_ARD_NUM_CE_ARRAY) are unused.

The only channel transfer protocol supported at this time is the Xilinx Cachelink (XCL) interface. The parameters to configure a Dynamic Address Generator (DAG) channel are unused and are only documented here for future reference.

If an XCL channel is connected to a master that will only perform read transactions, then the entry in C_XCL_WRITEXFER_ARRAY should be set to 0 indicating that no write transfers will be performed. This will reduce the channel logic to only contain logic for read transactions. Also, if an XCL channel is connected to a master that can consume data as soon as its available, the entry in the C_MCH_RDDATABUF_DEPTH_ARRAY for that channel should be set to zero. This will eliminate the read data buffer and eliminate the latency that would normally exist in reading data from this buffer.

If the PLB slave interface is included in the design (C_INCLUDE_PLB_IPIF=1), care must be taken when assigning User IP address ranges in the PLB Slave Interface generics. Since C_PLB_ARD_ADDR_RANGE_ARRAY can contain address ranges for services internal to the PLB Slave Burst and there may be address ranges within the User IP that will only be accessible through the PLB, the C_MCH_USERIP_ADDRRANGE_ARRAY may be a subset of the

C_PLB_ARD_ADDR_RANGE_ARRAY. Therefore, the User IP address ranges accessible by both the MCH interfaces and the PLB should occur first in C_PLB_ARD_ADDR_RANGE_ARRAY. Also, the order in which the address ranges are specified in the C_PLB_ARD_ADDR_RANGE_ARRAY and the C_MCH_USERIP_ADDRRANGE_ARRAY must be the same so that the chip selects to the User IP address ranges will be consistent between the PLB interface and the MCH interfaces. Note that it is assumed that all User IP address ranges accessible through the MCH interfaces are also accessible through the PLB interface.

Parameter/Port Dependencies

The dependencies between the MCH_PLBv46_Slave_Burst design parameters and I/O signals are shown in Table 3. It gives information about how the ports and parameters get affected by changing certain parameters.

Table 3: **Parameter-Port Dependencies**

Generic or Port	Name	Affects	Depends	Relationship Description
Design Parameters				
G2	C_INCLUDE_PLB_IPIF	P11 - P50	-	PLB input signals are unused and output signals are tied to 0 when C_INCLUDE_PLB_IPIF = 0.
G4	C_NUM_CHANNELS	P3 - P10	-	Input signals are unused and the output signals are tied to the default value when C_NUM_CHANNELS = 0 .
G7	C_MCHx_PROTOCOL	-	G4	Unused when x is greater than C_NUM_CHANNELS - 1.
G9	C_MCHx_ACCESSBUF_DEPTH	-	G4	Unused when x is greater than C_NUM_CHANNELS - 1.
G10	C_MCHx_RDDATABUF_DEPTH	-	G4	Unused when x is greater than C_NUM_CHANNELS - 1.
G11	C_XCLx_LINESIZE	-	G4	Unused when x is greater than C_NUM_CHANNELS - 1.
G12	C_XCLx_WRITEXFER	-	G4	Unused when x is greater than C_NUM_CHANNELS - 1.
G13	C_SPLB_DWIDTH	P15, P19, P29	-	Data bus width of MCH/PLB interface.
G14	C_SPLB_P2P	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
G15	C_CACHLINE_ADDR_MODE	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
G16	C_WR_BUFFER_DEPTH	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
G17	C_SPLB_SMALLEST_MASTER	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
G18	C_SPLB_NUM_MASTERS	P34, P35, P36, P50	G2	Defines width of slave response signals. Unused when C_INCLUDE_PLB_IPIF = 0.

Table 3: **Parameter-Port Dependencies *(Cont'd)***

Generic or Port	Name	Affects	Depends	Relationship Description
G19	C_SPLB_MID_WIDTH	P13	G2, G18	Specifies width of master ID bus Unused when C_INCLUDE_PLB_IPIF = 0. Width is equal to $\log_2$(C_SPLB_NUM_MASTERS).
I/O Signals				
P3	MCHx_Access_Control	-	G4	Unused when C_NUM_CHANNELS = 0.
P4	MCHx_Access_Data	-	G4, G6	Unused when C_NUM_CHANNELS = 0.
P5	MCHx_Access_Write	-	G4	Unused when C_NUM_CHANNELS = 0.
P6	MCHx_Access_Full	-	G4	Unused when C_NUM_CHANNELS = 0.
P7	MCHx_ReadData_Control	-	G4	Unused when C_NUM_CHANNELS = 0.
P8	MCHx_ReadData_Data	-	G4, G6	Unused when C_NUM_CHANNELS = 0.
P9	MCHx_ReadData_Read	-	G4	Unused when C_NUM_CHANNELS = 0.
P10	MCHx_ReadData_Exists	-	G4	Unused when C_NUM_CHANNELS = 0.
P11	PLB_ABus	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P12	PLB_PAValid	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P13	PLB_masterID	-	G2, G19	Unused when C_INCLUDE_PLB_IPIF = 0.
P14	PLB_RNW	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P15	PLB_BE	-	G2, G13	Unused when C_INCLUDE_PLB_IPIF = 0.
P16	PLB_MSize	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P17	PLB_size	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P18	PLB_type	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P19	PLB_wrDBus	-	G2, G13	Unused when C_INCLUDE_PLB_IPIF = 0.
P20	PLB_wrBurst	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P21	PLB_rdBurst	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P22	Sl_addrAck	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P23	Sl_SSize	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P24	Sl_wait	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.

Table 3: **Parameter-Port Dependencies *(Cont'd)***

Generic or Port	Name	Affects	Depends	Relationship Description
P25	SI_rearbitrate	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P26	SI_wrDAck	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P27	SI_wrComp	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P28	SI_wrBTerm	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P29	SI_rdDBus	-	G2, G13	Unused when C_INCLUDE_PLB_IPIF = 0.
P30	SI_rdDAck	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P31	SI_rdComp	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P32	SI_rdBTerm	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P33	SI_rdWdAddr	-	G2	Unused when C_INCLUDE_PLB_IPIF = 0.
P34	SI_MBusy	-	G2, G18	Unused when C_INCLUDE_PLB_IPIF = 0.
P35	SI_MWrErr	-	G2, G18	Unused when C_INCLUDE_PLB_IPIF = 0.
P36	SI_MRdErr	-	G2, G18	Unused when C_INCLUDE_PLB_IPIF = 0.

XCL Read Transfer Protocol

The protocol for cacheline read transfers via XCL is as follows:

The processor cache controller performs a single write to the Access buffer with the format shown in Table 4 and waits for the data to be input into the ReadData buffer.

Table 4: **Access Buffer Signals for Cacheline Read Transfer**

MCH Access Signal	Value
MCH_Access_Control(n)	'0'
MCH_Access_Data(n*C_MCH_SIPIF_DWIDTH : (n+1)*C_MCH_SIPIF_DWIDTH-1)	Memory address causing cache miss

When the cacheline data is available, it is written into the ReadData buffer with the format shown in Table 5. The control bit in the ReadData buffer should be viewed as a "data valid" indicator. When '1', the read operation occurred without error and the data is valid. When '0', the read operation had an error or did not return a transfer acknowledge within the parameterizable time period and the data is not valid. The data is returned in a *target-word* manner. This means that the data for the first address

requested is returned first even if that address was not on a cacheline boundary. The subsequent data words are from sequential addresses wrapping around on the size of the cacheline.

Table 5: **ReadData Buffer Signals for Cacheline Read Transfer**

MCH ReadData Signal	Value	
MCH_ReadData_Control(n)	'0'	timeout or error occurred, data is invalid
	'1'	data is valid
MCH_ReadData_Data(n*C_MCH_SIPIF_DWIDTH: (n+1)*C_MCH_SIPIF_DWIDTH-1)	X	control = '0'
	cacheline data word	control = '1'

Figure 2 shows the timing for a XCL cacheline read. The diagram is provided to indicate relative timing and example responses from some arbitrary IP in order to demonstrate the XCL protocol. It is not meant to reflect exact response times of a particular memory controller and should not be used to determine transaction latency or performance.

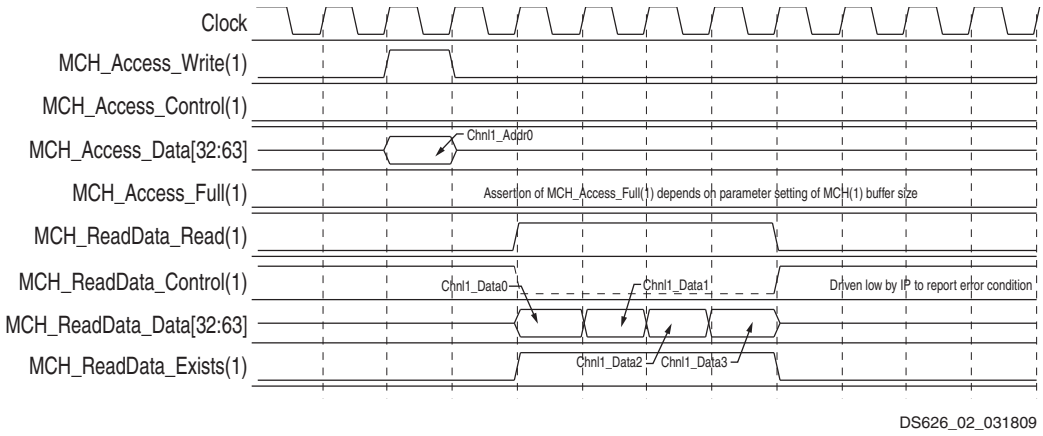

Figure 2: **XCL Cacheline Read**

XCL Single Write Transfer Protocol (C_XCL_WRITEXFER_ARRAY(n)=1)

The protocol for single write transfers via XCL is as follows:

The cache controller performs two writes into the Access buffer. There is no return status or value. The formats of these writes are shown in Table 6 and Table 7.

Table 6: **Access Buffer Signals for Single Write Transfer - First Write**

MCH Access Signal	Value	
MCH_Access_Control(n)	'1'	
MCH_Access_Data(n*C_MCH_SIPIF_DWIDTH: (n+1)*C_MCH_SIPIF_DWIDTH-3)	Word aligned address	

Table 6: **Access Buffer Signals for Single Write Transfer - First Write**

MCH Access Signal	Value	
MCH_Access_Data((n+1)*C_MCH_SIPIF_DWIDTH-2): (n+1)*C_MCH_SIPIF_DWIDTH-1)	If byte write	"00"
		"01"
		"10"
		"11"
	If half-word write	"01"
		"11"
	If word write	"00"

Table 7: **Access Buffer Signals for Single Write Transfer - Second Write**

MCH Access Signal	Value	
MCH_Access_Control(n)	If byte write	'1'
	If half-word or word write	'0'
MCH_Access_Data(n*C_MCH_SIPIF_DWIDTH: (n+1)*C_MCH_SIPIF_DWIDTH-1)	If byte write	valid data byte in correct byte lane
	If half-word write	valid data half-word in correct byte lanes
	If word write	word value

XCL Cacheline Write Transfer Protocol (C_XCL_WRITEXFER_ARRAY(n)=2)

The protocol for cacheline write transfers via MCH is as follows:

The cache controller performs a series of writes into the Access buffer. There is no return status or value. The first write provides the starting address of the cacheline and the remaining writes provide the data to be written. The writes into the Access buffer must occur on each clock. There can be no gaps in the writing of data into the Access buffer. The format of these writes is shown in Table 8 and Table 9.

Table 8: **Access Buffer Signals for Cacheline Write Transfer - First Write**

MCH Access Signal	Value
MCH_Access_Control(n)	'1'
MCH_Access_Data(n*C_MCH_SIPIF_DWIDTH: (n+1)*C_MCH_SIPIF_DWIDTH-1)	Word aligned address

Table 9: **Access Buffer Signals for Cacheline Write Transfer - Remaining writes**

MCH Access Signal	Value
MCH_Access_Control(n)	'1'
MCH_Access_Data(n*C_MCH_SIPIF_DWIDTH: (n+1)*C_MCH_SIPIF_DWIDTH-1)	cacheline word value

Figure 3 shows the timing diagram for a XCL Cacheline Write. Note that the writes into the Access buffer occur on every clock once the first write has occurred. There can be no gaps in writing data into the Access buffer. The diagram is provided to indicate relative timing and example responses from

some arbitrary IP to demonstrate the XCL protocol. It is not meant to reflect exact response times of a particular memory controller and should not be used to determine transaction latency or performance.

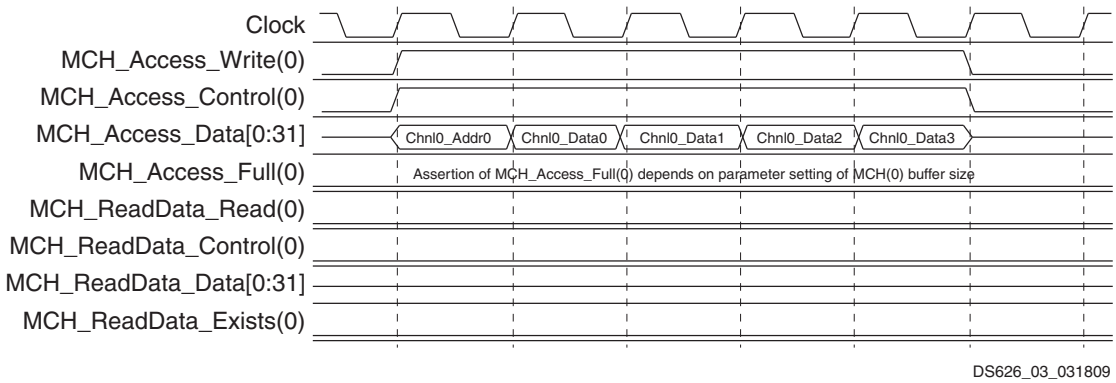

Figure 3: **XCL Cacheline Write**

Channel Block Diagram

The Channel Control block controls reading from the Access Buffer and writing to the ReadData Buffer. It implements the decoding of the channel transfer protocol as determined by the C_MCH_PROTOCOL_ARRAY. An address generator is provided for each channel which provides the addressing sequence required by that channel. For XCL channels, the Address/BE Generation block generates the intermediate cacheline addresses in a manner to support target-word first accesses. The IPIC Interface block implements the necessary IPIC signal protocol and timing to communicate with the IP Core.

As there is a request in the Access Buffer, the Channel Control logic reads the transaction request from the Access Buffer, decodes the request, loads the address generator, and starts the IPIC interface logic. At this point, the IPIC transaction is ready and waiting for the Arbitration Logic to grant this channel access to the IP core. The goal is to have be no wasted cycles after arbitration.

When the arbitration logic selects this channel to have access to the IP core, the IPIC signal protocol provides the necessary handshaking to complete the transaction. For a read transaction, the Channel Control block will write the cacheline data into the ReadData buffer. If a timeout occurs, the Channel Control block writes the appropriate values into the ReadData buffer and terminates the transaction.

A block diagram of a channel is shown in Figure 4.

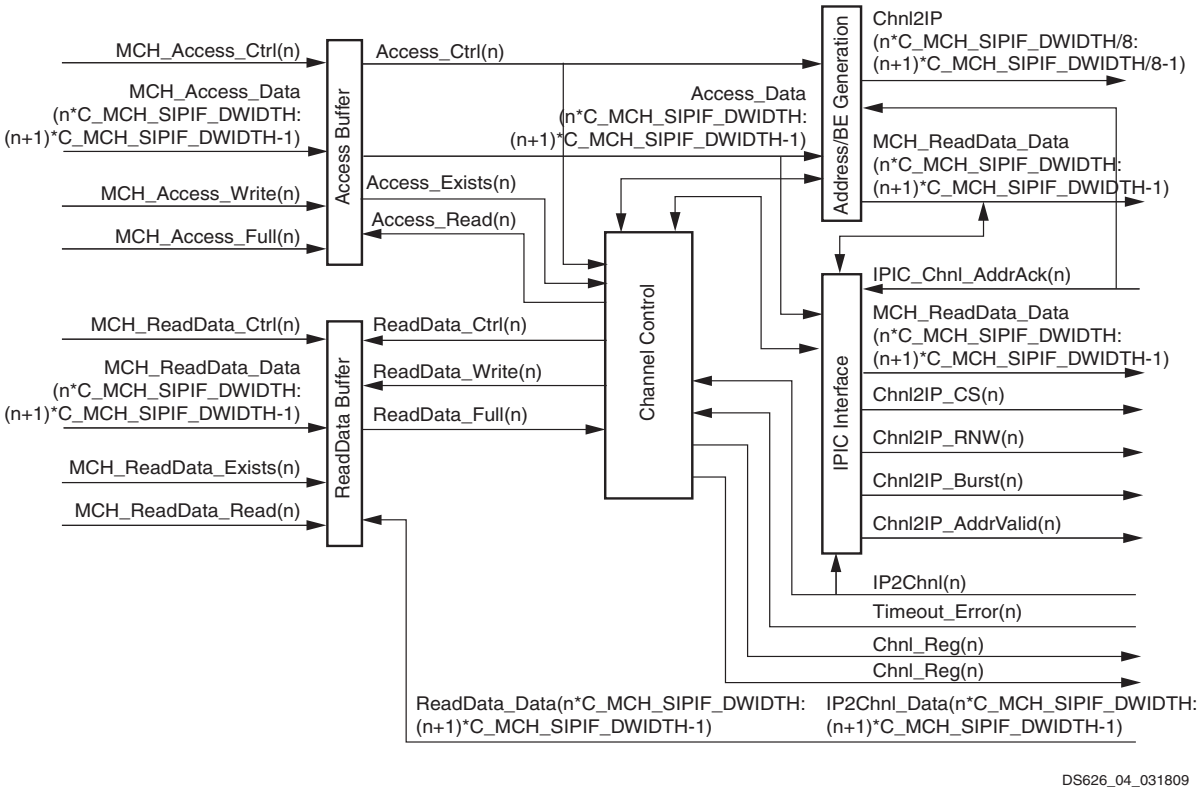

Figure 4: **Channel Interface Block Diagram**

Arbitration and Mux/De-Mux Logic

For the first phase of the MCH_PLBv46_Slave_Burst, only a fixed-priority arbitration scheme is supported. Requests from the MCH interfaces are indicated by internal signals output from the Channel Logic. Requests from the PLB interface are indicated by the internal PLB2IP_CS signals output from the PLBv46 Slave Burst.

To support memory controllers and other IP that support separate address and data phases, the MCH_PLBv46_Slave_Burst provides a multiplexor/de-multiplexor for the IPIC address phase signals and a separate multiplexor/de-multiplexor for data phase signals which are controlled by the arbitration logic.

Arbitration to determine the next transaction master occurs when the MCH_PLBv46_Slave_Burst is idle. There are no channel or PLB requests or during the final transfers of the address phase of the active transaction. Once a transaction is in progress, it will not be preempted to service a higher priority request. However, when possible, arbitration will be overlapped with the final address transfers of the active transaction to allow continuous access to the IP without any *dead* arbitration cycles.

The arbitration logic is shown in Figure 5.

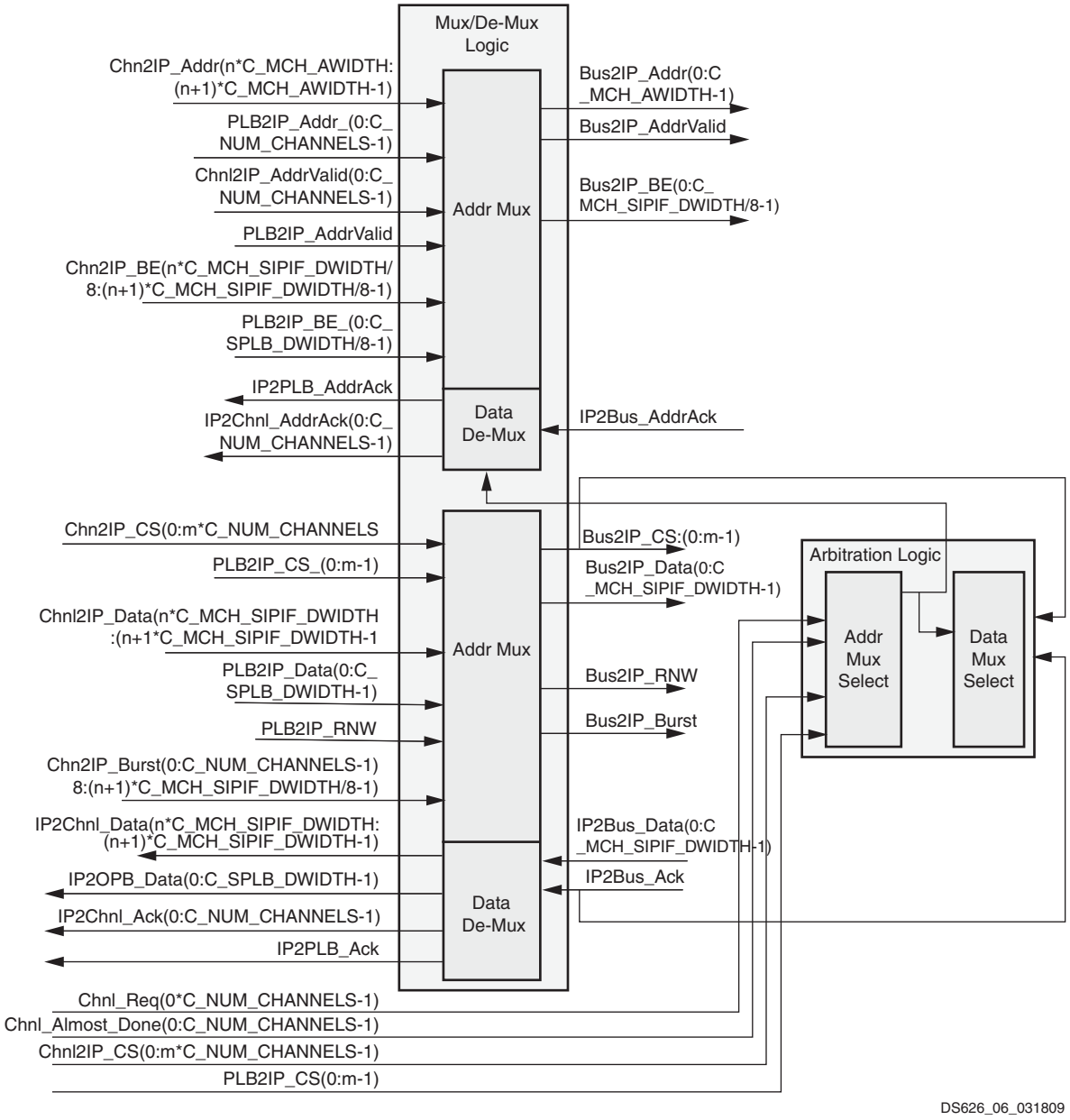

Figure 5: **Arbitration and Mux/De-Mux Logic**

The timing of the arbitration cycles and multiplexor control accounts for any registers on the output of the MUX for the IPIC interface signals that may be necessary to increase F_{MAX}. The timing relationships of these signals are shown in Figure 6. The timing diagram is provided only to indicate relative timing and example responses from an arbitrary IP to demonstrate the overlapped arbitration and control of the multiplexor/de-multiplexor logic. The diagram is not intended to reflect exact response times of a particular memory controller and should not be used to determine transaction latency or performance.

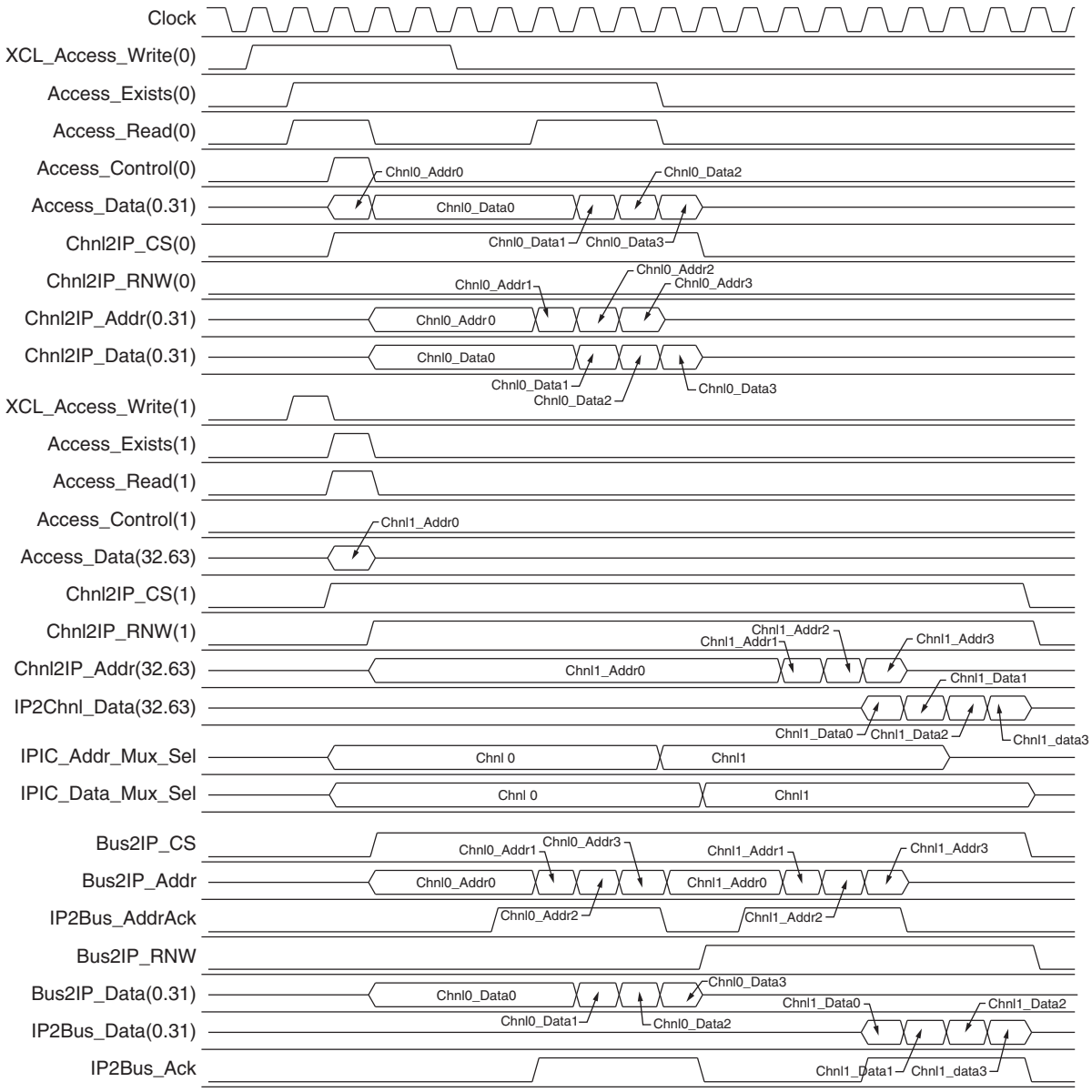

Figure 6: **Overlapped Arbitration Timing**

If the request winning arbitration is accessing the same external memory bank as the active transaction, such as Chnl2IP_CS = Bus2IP_CS, the IPIC address mux is switched to this channel during the final address transfers of the active transaction. This allows the address signals of the next transaction to be output to the IP and provide continuous address cycles. The IPIC data mux is switched to the new channel during the final data transfer of the active transaction. However, if the request winning arbitration is accessing a different external memory bank than the active transaction, such as, Chnl2IP_CS /= Bus2IP_CS, the both the IPIC address mux and the IPIC data mux are switched to the new channel during the final data transfer of the active transaction.

There are no *grant* signals from the Arbitration Logic to the channel interfaces or the PLB interface. The IPIC signals of the selected transaction will be multiplexed to the IPIC interface of the IP core and the responses from the core and will be routed back to the selected channel. The channel interfaces do not

require a response on write transactions, but will wait for a entry into the ReadData buffer for a read transaction.

Fixed Priority Mode

The fixed priority mode will be such that Channel 0 is the highest priority request, followed by Channel 1 through Channel N. Requests from the PLB interface will be the lowest priority. Priority of a channel is set by its connection to the MCH_PLBv46_Slave_Burst.

Design Implementation

Target Technology

The intended target technology is the Virtex® and Spartan® FPGA families.

Device Utilization and Performance Benchmarks

Because the MCH_PLBv46_Slave_Burst is a module that will be used with other design modules in the FPGA, the utilization and timing numbers reported in this section are just estimates. As the MCH_PLBv46_Slave_Burst is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing will vary from the results reported here.

The MCH_PLBv46_Slave_Burst resource utilization for various parameter combinations measured with Virtex-5 as a target device are detailed in Table 10.

Table 10: **Performance and Resource Utilization Benchmarks on Virtex-5 (xc5vlx50-ff1153-1)**

Parameter Values								Device Resources			
C_NUM_CHANNELS	C_PLB_ARD_ADDR_RANGE_ARRAY Pairs	C_PLB_ARD_NUM_CE_ARRAY	C_INCLUDE_PLB_IPIF	C_SPLB_DWIDTH	C_SIPIF_DWIDTH	C_WR_BUFFER_DEPTH	C_SPLB_P2P	Slice	Slice Flip-Flops	Slice LUTs	F_{MAX}(1)
0	1	1	1	32	32	0	1	68	104	122	209
0	1	1	1	32	32	0	0	55	134	127	223
0	1	1	1	32	32	16	0	101	187	180	200
1	1	2	1	32	32	16	0	187	398	480	213
1	1	4	1	32	32	16	0	204	399	484	200
2	2	4,4	1	32	32	16	0	309	541	627	202
2	2	4,4	1	64	32	16	0	345	542	628	200

Table 10: **Performance and Resource Utilization Benchmarks on Virtex-5 (xc5vlx50-ff1153-1) *(Cont'd)***

4	2	4,4	1	128	32	16	0	553	820	1120	200
4	4	4,4,4,4	1	32	32	16	0	510	809	1127	203
4	4	4,4,4,4	0	32	32	16	0	423	561	882	200

1. F_{MAX} represents the maximum frequency of the MCH_PLBv46_Slave_Burst in a standalone configuration. The actual maximum frequency will depend on the entire system and may be greater or less than what is recorded in this table.

Reference Documents

The following documents contain reference information important to understanding the Xilinx MCH_PLBv46_Slave_Burst design.

1. DS562, *PLBv46_Slave_Burst V1.00.A Product Specification*
2. *IBM CoreConnect 128-Bit Local Peripheral Bus, Architectural Specification* (v4.6)

Support

Xilinx provides technical support for this LogiCORE IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
5/25/07	1.0	Initial Xilinx release.
7/28/08	1.1	Added QPro Virtex-4 Hi-Rel and QPro Virtex-4 Rad Tolerant FPGA support.
9/29/08	1.2	Incorporated CR473185; updated PDF properties; corrected link in Ref Docs section.
4/24/09	1.3	Replaced references to supported device families and tool names with hyperlink to PDF file.

Notice of Disclaimer