LogiCORE IP AXI Slave Burst (v1.00a)

DS769 June 22, 2010 **Product Specification**

Introduction

The LogiCORE IP AXI Slave Burst is an interface between the AXI4 memory-mapped interface to the IPIC (IP Inter Connect). This core is designed to provide a smooth migration path to the burst-supported IP from PLBv46 to AXI4 with minor updates in the interface. The core provides a point to point bi-directional interface between a user IP core and the AXI4 interconnect. This core acts as master on IPIC while it behaves as a slave on AXI4.

Features

- Supports 1:1 (AXI4:IPIC) synchronous clock
- Supports 1:1 (AXI4:IPIC) data width
- AXI4 Interface
  - 32-bit address bus
  - Configurable 32/64-bit data bus
  - Supports AXI4 narrow transfers
  - Supports AXI4 unaligned transfers
  - Supports configurable interfaces
    - Read/Write Interface
    - Read-only Interface
    - Write-only Interface
  - Supports burst lengths in below format
    - 1-256 beats with INCR type transfers
    - 1-16 beats with FIXED type transfers
    - 2, 4, 8, 16 beats with WRAP type transfers
- IPIC Interface
  - 32-bit address bus
  - Configurable data bus width (32/64)
  - Supports single/burst transfers
  - Configurable read data buffer depth
  - Optional IPIC time-out generation
  - Optional byte enable generation for read transactions based upon transfer size

LogiCORE IP Facts Table					
Core Specifics					
Supported Device Family[1]	Artix™-7, Virtex®-7, Kintex™-7, Virtex-6[2], Spartan®-6[3]				
Supported User Interfaces	AXI4				
	Resources				**Frequency**
	LUTs	**FFs**	**DSP Slices**	**Block RAMs**	**Max. Freq.**
	See Table 11 through Table 13.				
Provided with Core					
Documentation	Product Specification				
Design Files	VHDL				
Example Design	Not Provided				
Test Bench	Not Provided				
Constraints File	Not Provided				
Simulation Model	Not Provided				
Tested Design Tools					
Design Entry Tools	XPS 13.2				
Simulation[4]	Mentor Graphics ModelSim				
Synthesis Tools	ISE 13.2				
Support					
Provided by Xilinx, Inc.					

Notes:

1. For a complete list of supported derivative devices, please see the IDS Embedded Edition Derivative Device Support.
2. For more information on the Virtex-6 devices, see the DS150, *Virtex-6 Family Overview*.
3. For more information on the Spartan-6 devices, see the DS160, *Spartan-6 Family Overview*.
4. For the supported versions of the tools, see the ISE Design Suite 13: Release Notes Guide.

Limitations and Unsupported Features

The following features are limited or unsupported:

- Write buffers are not implemented
- AXI4 Slave interface category
- Control Interface
- Stream Interface
- Atomic region is not supported
- Locked transfers are not supported
- Cache - type of transaction bufferable/cacheble treated as normal transaction
- Debug/Secure transactions are not supported
- User signals are not supported
- Quality of service signaling is not supported
- Out of order transactions
- Trust zone
- Protection unit
- Exclusive transactions based upon helper library core
- Low-power state
- Simultaneous read-write transactions from AXI4 are not supported
- Multiple chip enables per address range are not supported
- Region signals are not supported

Functional Description

Overview

The AXI Slave Burst is designed to provide the user with a quick way to implement light weight interface between the ARM AXI4 and a user slave IP core capable of supporting bursts. This slave interface allows for multiple user IPs to be interfaced to the AXI4 providing address decoding over various address ranges as configured by the user. The use of this core allows easy migration of user slave IP from earlier PLBv46, PLBv34, and OPB which used the respective IPIF cores. The AXI4 protocol is sufficiently simple to be designed to directly when unsupported features are needed or lowest latency/ highest throughput is required. Figure 1 shows a block diagram of the AXI Slave Burst. The port references and groupings are detailed in Table 2. The internal modules provide the basic functionality for connected slave IP operation based upon the AXI4 transaction. It implements the protocol and timing translation between the AXI4 and IPIC interface.

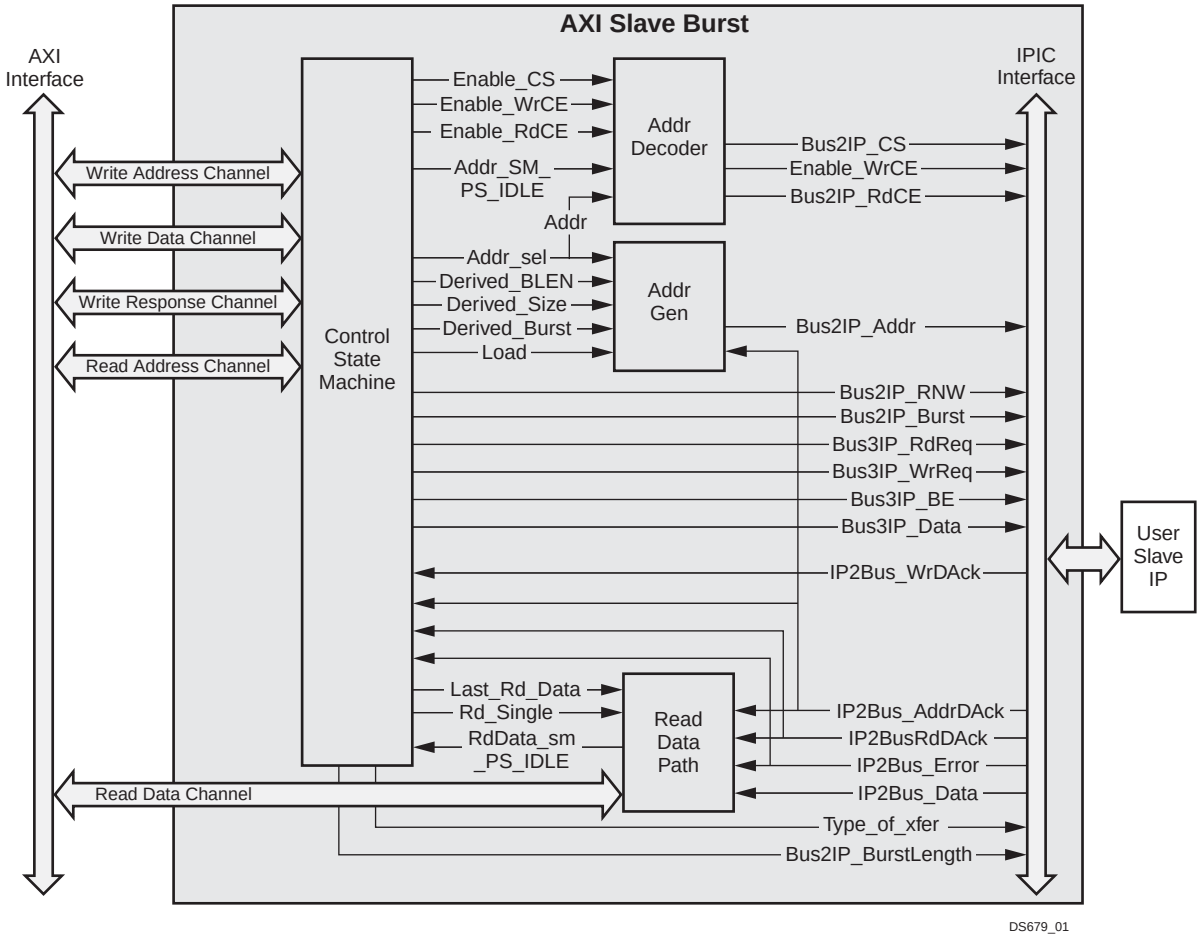


Figure 1: **AXI Slave Burst IP Core Block Diagram**

Internal Modules

Control State Machine

The Control State Machine monitors the AXI4 and IPIC activity and accordingly decides the state transitions and control signal generation. This module also includes optional time out conditional logic for address and data phase. If the C_INCLUDE_TIMEOUT_CNT = 1, then the user must define values for C_TIMEOUT_CNTR_VAL. Typical values for C_TIMEOUT_CNTR_VAL parameter are 8 or 16, default value is 8. The internal time-out counter logic is divided into address phase and data phase. During the address phase the address time-out counter will be loaded with the value mentioned for C_TIMEOUT_CNTR_VAL parameter and it will start down-counting as soon as valid control signals present on IPIC interface. It is expected that the user IP will acknowledge the address on-OR-before the final value of C_TIMEOUT_CNTR_VAL, else address phase time out will be generated. If the address phase time-out is generated, then the data phase corresponding to this address phase is skipped and new address will be placed on IPIC interface based upon byte/half-word/word/double-word size of AXI transfer. If the user IP acknowledges the address, the internal data phase time-out counter will reset to C_TIMEOUT_CNTR_VAL and now the same counter will monitor the data acknowledge from the user IP. If the user IP doesn't respond before the final count of data phase, the data time-out will be generated. Both the address and data phase time-out errors will be reported as slave error (SLVERR) on AXI. If C_INCLUDE_TIMEOUT_CNT = 0, then it is user's responsibility to make sure that the user IP will generate proper address and data acknowledge to complete the transactions else the core may not work properly. This is equally applicable to read and write transfers.

The control state machine also includes logic inclusion based on the C_RDATA_FIFO_DEPTH parameter option. If C_RDATA_FIFO_DEPTH = 0, then all the read transactions towards IPIC side will be considered as single transactions, with all bus2ip_burst and bus2ip_burstlength signals tied to '0'. However all the write transactions proceed with valid burst length (on bus2ip_burstlength signal) and burst information (on bus2ip_burst signal).

The control state machine also includes logic for generating the write response to AXI4. This block also generates the data and byte enable signals towards IPIC interface.

The control state machine has round-robin logic arbitration to avoid the starvation between the AXI4 read and write transactions. The power-on-reset preference is given to read transaction. If the read transaction is not present then the preference will shift to write transaction. Later on the preference will be switched between read and write.

If C_ALIGN_BE_RDADDR = 0 then during the complete read mode at IPIC, the Bus2IP_BE will be always driven high. If C_ALIGN_BE_RDADDR = 1 then during the complete read transaction byte enables will be aligned based upon the size of AXI transfer.

Address Generation

The Address Generation Module is responsible for generating addresses, which are data width aligned, on IPIC interface. During write and read transactions, the Bus2IP_Address will always be data width aligned (i.e either 32-bit aligned or 64-bit aligned). During the write mode, the Bus2IP_BE will be replicating the values on write strobes from AXI4.

Address Decoding

The Address Decoding Module generates the logic for decoding the incoming address and generating the Bus2IP_CS, Bus2IP_RdCE and Bus2IP_WrCE signals. For every address range, there is single instance of the above mentioned signals. At any given time, there will be only one read chip enable or write chip enable for any active transaction. Initially the Bus2IP_CS and read or write chip enable signals will be asserted simultaneously.

Read Data Path Module

This module will be included in the design if C_S_AXI4_SUPPORTS_READ parameter is set to '1' (read mode). This module includes read data state machine, optional FIFO related logic (if C_RDATA_FIFO_DEPTH = 32) and internal counter to trace the IPIC and AXI4 read activities. The read data path state machine also generates handshake signals towards control state machine.

Design Parameters

To allow the user to create the AXI4 Slave Burst IP core that is uniquely tailored for the user's system, certain features can be parameterized in the core design. This allows the user to have a design that utilizes only the minimum resources required by the system and runs at the best possible performance. The features of the AXI4 Slave Burst IP core can be customized for user interface tailoring via VHDL generic parameters. These parameters are detailed in Table 1.

Inferred Parameters

In addition to the parameters listed in Table 1, there are also parameters that are inferred for each AXI interface in the EDK tools. Through the design, these EDK-inferred parameters control the behavior of the AXI Interconnect. For a complete list of the interconnect settings related to the AXI interface, see the DS768, AXI Interconnect IP Data Sheet.

Table 1: **Design Parameters**

Generic	Feature/Description	Parameter Name	Allowable Values	Default Values	VHDL Type
System Parameters					
G1	Target FPGA family	C_FAMILY	artix7, virtex7, kintex7, virtex6, spartan6	virtex6	string
AXI4 Slave Burst IP Core Parameters					
G2	Use read data buffer for IPIC transaction	C_RDATA_ FIFO_DEPTH	0, 32(1)	0	integer
G3	Include time out counter for address as well as data phase	C_INCLUDE_ TIMEOUT_CNT	0,1(2)	0	integer
G4	Address/Data Phase Time-out counter value	C_TIMEOUT_ CNTR_VAL	8,16(2)	8	integer
G5	Align byte enables with read address based upon size of transfer	C_ALIGN_BE_RDADDR	0, 1(3)	0	integer
AXI4 Slave Burst IP Core's AXI4 Related Parameters					
G6	AXI4 Interface type: Write only	C_S_AXI_ SUPPORTS_ WRITE	0-1(4)	1	integer
G7	AXI4 Interface type: Read only	C_S_AXI_ SUPPORTS_ READ	0-1(4)	1	integer
G8	AXI4 address bus width	C_S_AXI_ ADDR_WIDTH	32	32	integer
G9	AXI4 data bus width	C_S_AXI_DATA_WIDTH	32, 64	32	integer
G10	AXI4 Identification tag width	C_S_AXI_ID_ WIDTH	1- 16	4	integer
AXI4 Slave Burst IP Core - Address Decoder Range Definition Parameters					
G11	Array of Base Address / High Address Pairs for each Address Range	C_ARD_ADDR_RANGE_ ARRAY(5)	See Descriptions of Parameters and Their Usage, page 12 for more details.	User must set values.	SLV64_ ARRAY_ TYPE(5)
G12	Array of the desired number of chip enables for each address range	C_ARD_ NUM_CE_ARRAY(5)		User must set values.	INTEGER_ ARRAY_ TYPE(5)

Table 1: **Design Parameters *(Cont'd)***

Generic	Feature/Description	Parameter Name	Allowable Values	Default Values	VHDL Type

Notes:

1. When C_RDATA_FIFO_DEPTH = 0, the read FIFO is not added in the system. The data from user IP will be sent to AXI4.
2. When C_INCLUDE_TIMEOUT_CNT = 1, then only C_TIMEOUT_CNTR_VAL is effective, else C_TIMEOUT_CNTR_VAL value will be ignored. The C_TIMEOUT_CNTR_VAL value is common for address as well data phase. The counter will start first for address phase and resets upon reception of address acknowledge from IP. Now it will be reloaded for data phase and will reset once the IP sends data acknowledge. If the slave doesn't respond either in address or in data phase within the given clock cycles, error response will be sent to the master and transaction will proceed to next level.
3. The C_ALIGN_BE_RDADDR = 0 will indicate that the byte enables during read transaction at IPIC are all '1'. The C_ALIGN_BE_RDADDR = 1 will indicate whether the byte enables during read transaction at IPIC are address aligned based upon the size of transfer. This parameter is applicable for read transactions only.
4. When C_S_AXI_SUPPORTS_WRITE = 1 and C_S_AXI_SUPPORTS_READ = 0, then only write operation related logic will be included in the design. When C_S_AXI_SUPPORTS_WRITE = 0 and C_S_AXI_SUPPORTS_READ = 1, then only read operation related logic will be included in the design. When C_S_AXI_SUPPORTS_WRITE = 1 and C_S_AXI_SUPPORTS_READ = 1, then both write and read operation related logic will be included in the design. The combination C_S_AXI_SUPPORTS_WRITE = 0 and C_S_AXI_SUPPORTS_READ = 0 is illegal and the slave core will not respond any AXI transaction.
5. This Parameter VHDL type is a custom type defined in the ipif_pkg.vhd. User can specify multiple ranges based upon the requirements.

I/O Signals

The AXI4 Slave Burst IP core signals are listed and described in Table 2.

Table 2: **I/O Signals Description**

Port	Signal Name	Interface	I/O	Initial State	Description
AXI4 Global System Signals					
P1	S_AXI_ACLK	AXI	I	-	AXI4 Clock
P2	S_AXI_ARESETN	AXI	I	-	AXI4 Reset, active LOW
AXI4 Write Address Channel Signals					
P3	S_AXI_AWID[(C_S_AXI_ID_WIDTH-1) : 0]	AXI	I	-	Write address ID: This signal is the identification tag for the write address group of signals.
P4	S_AXI_AWADDR[(C_S_AXI_ADDR_WIDTH-1) : 0]	AXI	I	-	AXI4 Write address: The write address bus gives the address of the first transfer in a write burst transaction.
P5	S_AXI_AWLEN[7 : 0]	AXI	I	-	Burst length: This signal gives the exact number of transfers in a burst. "00000000" - "11111111" indicates Burst Length 1 - 256.
P6	S_AXI_AWSIZE[2 : 0]	AXI	I	-	Burst size: This signal indicates the size of each transfer in the burst. "000" - 1 byte "001" - 2 byte (half word) "010" - 4 byte (word) "011" - 8 byte (double word) others - NA
P7	S_AXI_AWBURST[1 : 0]	AXI	I	-	Burst type: This signal coupled with the size information, details how the address for each transfer within the burst is calculated. "00" - FIXED "01" - INCR "10" - WRAP "11" - Reserved

Table 2: **I/O Signals Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P8	S_AXI_AWLOCK[1]	AXI	I	-	Lock type: This signal provides additional information about the atomic characteristics of the transfer.
P9	S_AXI_AWCACHE[3 : 0][1]	AXI	I	-	Cache type: This signal indicates the bufferable, cacheable, write-through, write-back and allocate attributes of the transaction Bit-0 : Bufferable (B) Bit-1 : Cacheable (C) Bit-2 : Read Allocate (RA) Bit-3 : Write Allocate (WA) The combination where C=0 and WA/RA=1 are reserved.
P10	S_AXI_AWPROT[2: 0][1]	AXI	I	-	Protection type: This signal indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access. Bit-0 : 0=Normal access, 1=Privileged access Bit-1 : 0=Secure access, 1=Non-secure access Bit- 2 : 0=Data access; 1=Instruction access
P11	S_AXI_AWVALID	AXI	I	-	Write address valid: This signal indicates that valid write address and control information are available.
P12	S_AXI_AWREADY	AXI	O	0	Write address ready: This signal indicates that the slave is ready to accept an address and associated control signals.
AXI4 Write Channel Signals					
P13	S_AXI_WDATA[(C_S_AXI_DATA_WIDTH-1) : 0]	AXI	I	-	Write data bus.
P14	S_AXI_WSTB[((C_S_AXI_DATA_WIDTH/8)-1) : 0]	AXI	I	-	Write strobes: This signal indicates which byte lanes in S_AXI_WDATA are/is valid.
P15	S_AXI_WLAST	AXI	I	-	Write last: This signal indicates the last transfer in a write burst.
P16	S_AXI_WVALID	AXI	I	-	Write valid: This signal indicates that valid write data and strobes are available.
P17	S_AXI_WREADY	AXI	O	0	Write ready: This signal indicates that the slave can accept the write data.
AXI4 Write Response Channel Signals					
P18	S_AXI_BID[(C_S_AXI_ID_WIDTH-1) : 0]	AXI	O	0	Write response ID: This signal is the identification tag of the write response. The BID value must match the AWID value of the write transaction to which the slave is responding.
P19	S_AXI_BRESP[1 : 0]	AXI	O	0	Write response: This signal indicates the status of the write transaction. “00“ - OKAY “01“ - EXOKAY - NA “10“ - SLVERR “11“ - DECERR - NA
P20	S_AXI_BVALID	AXI	O	0	Write response valid: This signal indicates that a valid write response is available.

Table 2: **I/O Signals Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P21	S_AXI_BREADY	AXI	I	-	Response ready: This signal indicates that the master can accept the response information.
AXI4 Read Address Channel Signals					
P22	S_AXI_ARID[(C_S_AXI_ID_WIDTH-1) : 0]	AXI	I	-	Read address ID: This signal is the identification tag for the read address group of signals.
P23	S_AXI_ARADDR[(C_S_AXI_ADDR_WIDTH -1) : 0]	AXI	I	-	Read address: The read address bus gives the initial address of a read burst transaction.
P24	S_AXI_ARLEN[7 : 0]	AXI	I	-	Burst length: This signal gives the exact number of transfers in a burst. “00000000“ - “11111111” indicates Burst Length 1 - 256.
P25	S_AXI_ARSIZE[2 : 0]	AXI	I	-	Burst size: This signal indicates the size of each transfer in the burst. “000“ - 1 byte “001“ - 2 byte (Half word) “010“ - 4 byte (word) "011" - 8 byte (double word) others - NA
P26	S_AXI_ARBURST[1 : 0]	AXI	I	-	Burst type: The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated. “00“ - FIXED “01“ - INCR “10“ - WRAP “11“ - Reserved
P27	S_AXI_ARLOCK[1]	AXI	I	-	Lock type: This signal provides additional information about the atomic characteristics of the transfer.
P28	S_AXI_ARCACHE[3 : 0][1]	AXI	I	-	Cache type: This signal provides additional information about the cacheable characteristics of the transfer. Bit-0 : Bufferable (B) Bit-1 : Cacheable (C) Bit-2 : Read Allocate (RA) Bit-3 : Write Allocate (WA) The combination where C=0 and WA/RA=1 are reserved.
P29	S_AXI_ARPROT[2 : 0][1]	AXI	I	-	Protection type: This signal provides protection unit information for the transaction.
P30	S_AXI_ARVALID	AXI	I	-	Read address valid: This signal indicates, when HIGH, that the read address and control information is valid and will remain stable until the address acknowledgement signal, S_AXI_ARREADY, is high.
P31	S_AXI_ARREADY	AXI	O	0	Read address ready: This signal indicates that the slave is ready to accept an address and associated control signals.
AXI4 Read Data Channel Signals					

Table 2: **I/O Signals Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P32	S_AXI_RID[(C_S_AXI_ ID_WIDTH - 1): 0]	AXI	O	0	Read ID tag: This signal is the ID tag of the read data group of signals. The S_AXI_RID value is generated by the slave and must match the ARID value of the read transaction to which it is responding.
P33	S_AXI_RDATA[(C_S_ AXI_DATA_WIDTH -1): 0]	AXI	O	0	Read data bus.
P34	S_AXI_RRESP[1 : 0]	AXI	O	0	Read response: This signal indicates the status of the read transfer. "00" - OKAY "01" - EXOKAY - NA "10" - SLVERR "11" - DECERR - NA
P35	S_AXI_RLAST	AXI	O	0	Read last: This signal indicates the last transfer in a read burst.
P36	S_AXI_RVALID	AXI	O	0	Read valid: This signal indicates that the required read data is available and the read transfer can complete.
P37	S_AXI_RREADY	AXI	I	-	Read ready: This signal indicates that the master can accept the read data and response information.
User IP Signals					
P38	Bus2IP_Clk	user IP	O	0	Synchronization clock provided to user IP: This is the same as S_AXI_Clk.
P39	Bus2IP_Resetn	user IP	O	0	Active low reset for use by the user IP: It is a pass through of the S_AXI_ARESETN input.
P40	IP2Bus_Data[(C_S_AXI_ DATA_WIDTH -1) : 0]	user IP	I	-	Input Read Data bus from the user IP: Data is qualified with the assertion of IP2Bus_RdAck signal.
P41	IP2Bus_WrAck	user IP	I	-	Active high write data acknowledge: Asserted by IP to indicate acceptance of data.
P42	IP2Bus_RdAck	user IP	I	-	Active high read data qualifier: Read data on the IP2Bus_Data Bus is valid when the IP2Bus_RdAck is high.
P43	IP2Bus_AddrAck	user IP	I	-	Active high signal: This signal from IPIC cause advances in the address counter and request state during multiple data beat transfers.
P44	IP2Bus_Error	user IP	I	-	Active high signal: This signal indicates the user IP has encountered an error with the requested operation. This signal is sampled only if IP2Bus_RdAck or the IP2Bus_WrAck is asserted.
P45	Bus2IP_Addr[(C_S_AXI_ ADDR_WIDTH - 1) : 0]	user IP	O	0	Address bus indicating the desired address of the requested read or write operation. Valid when Bus2IP_CS and Bus2IP_WrCE or Bus2IP_RdCE are driven high.
P46	Bus2IP_Data([C_S_AXI_ DATA_WIDTH -1) : 0]	user IP	O	0	Write data bus to the user IP: Valid when Bus2IP_CS and Bus2IP_WrCE and Bus2IP_WrReq are driven high.

Table 2: **I/O Signals Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P47	Bus2IP_RNW	user IP	O	0	Read not write signal: High is a read, low is a write. Valid when Bus2IP_CS and Bus2IP_WrCE or Bus2IP_RdCE are driven high.
P48	Bus2IP_BE[((C_S_AXI_ DATA_WIDTH/8) - 1]) : 0]	user IP	O	0	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. During write operation from AXI4, the S_AXI_WSTRB signals will be driven on this bus. During read operation from AXI4, based upon C_ALIGN_BE_RDADDR parameter settings either all '1' or size aligned values will be driven on this bus.
P49	Bus2IP_Burst	user IP	O	0	Bus2IP_Burst indicates that the current cycle is a burst cycle when asserted or a single beat transfer when not asserted. In case of burst transfer the Bus2IP_Burst will remain asserted until the second to last data is acknowledge by the user IP with a IP2Bus_RdAck or IP2Bus_WrAck as the case may be.
P50	Bus2IP_BurstLength[7 : 0]	user IP	O	0	This value is an indication of the number of beats being requested for transfer and is valid when the cycle is of burst type when Bus2IP_CS = 1.
P51	Bus2IP_WrReq	user IP	O	0	Bus2IP_WrReq indicates when write data is valid on Bus2IP_Data bus. For single beat write transfers, Bus2IP_WrReq will assert for one cycle regardless on whether the user IP acknowledges the cycle or not. For burst cycles Bus2IP_WrReq will remain asserted for the entire burst sequence.
P52	Bus2IP_RdReq	user IP	O	0	Bus2IP_RdReq indicates the requested cycle is a read type transfer. Bus2IP_RdReq will assert for one cycle for single beat read transfers coincident with Bus2IP_CS and Bus2IP_RdCE. For burst cycles, Bus2IP_RdReq will assert for each valid address of the burst cycle.
P53	Bus2IP_CS[((C_ARD_ ADDR_RANGE_ ARRAY'length/2) -1) : 0]	user IP	O	0	Active High chip select bus: Each bit of the bus corresponds to an address pair entry in the C_ARD_ADDR_RANGE_ARRAY. Assertion of a chip select indicates a active transaction request to the chip select's target address space.
P54	Bus2IP_RdCE: 0][(calc_num_ce[(C_ARD_NUM_CE_ARRAY)-1) : 0] (2)	user IP	O	0	Active high read chip enable bus: Chip enables are assigned per the user's entries in the C_ARD_NUM_CE_ARRAY. These chip enables are asserted only during active read transaction requests with the target address space and in conjunction with Bus2IP_CS for the corresponding sub-address within the space. One read chip enable is allowed per given address region

Table 2: **I/O Signals Description *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P55	Bus2IP_WrCE: 0[[(calc_num_ce[(C_ARD_NUM_CE_ARRAY)-1): 0] (2)	user IP	O	0	Active high write chip enable bus: Chip enables are assigned per the user's entries in the C_ARD_NUM_CE_ARRAY. These chip enables are asserted only during active write transaction requests with the target address space and in conjunction with Bus2IP_CS for the corresponding sub-address within the space. One write chip enable is allowed per given address region
P56	Type_of_xfer	user IP	O	0	Type of transfer 0 - FIXED transfer from AXI4 1 - INCR/WRAP transfer from AXI4 This signal should be used by the user IP to decide the response. If the user IP doesn't support any of FIXED or INCR/WRAP transfer, then the IP should respond with error if that particular transaction is received.

Notes:

1. These signals are present in the core at AXI4 interface level only. The design will not respond for any specific combination to these signals.
2. The size of Bus2IP_RdCE and Bus2IP_WrCE bus is the sum of the integer values entered in the C_ARD_NUM_CE_ARRAY parameter.

Parameter - Port Dependencies

The parameter port dependencies for the AXI Slave Burst IP Core are listed in below Table 3.

Table 3: **AXI4 Slave Burst IP Core Parameter - I/O Dependencies**

I/O Signal	Name	Affects	Depends	Relationship Description
Design Parameters				
G8	C_S_AXI_ADDR_WIDTH	P4, P23, P45	-	Defines the width of the address bus
G9	C_S_AXI_DATA_WIDTH	P13, P14, P33, P40, P46, P48	-	Defines the width of the data bus
G10	C_S_AXI_ID_WIDTH	P3, P18, P22, P32	-	Defines the ID width
G11	C_ARD_ADDR_RANGE_ARRAY	P53	-	The vector width of Bus2IP_CS is the number of elements in C_ARD_ADDR_RANGE_ARRAY/2
G12	C_ARD_NUM_CE_ARRAY	P54, P55	-	The vector width of Bus2IP_RdCE and Bus2IP_WrCE is the number of elements in C_ARD_NUM_CE_ARRAY.
I/O Signals				
P3	S_AXI_AWID[C_S_AXI_ID_WIDTH-1:0]	-	G10	Port width depends on the generic C_S_AXI_ID_WIDTH
P4	S_AXI_AWADDR[C_S_AXI_ADDR_WIDTH-1:0]	-	G8	Port width depends on the generic C_S_AXI_ADDR_WIDTH

Table 3: **AXI4 Slave Burst IP Core Parameter - I/O Dependencies *(Cont'd)***

I/O Signal	Name	Affects	Depends	Relationship Description
P13	S_AXI_WDATA[C_S_AXI _DATA_WIDTH-1:0]	-	G9	Port width depends on the generic C_S_AXI_DATA_WIDTH
P14	S_AXI_WSTB[C_S_AXI _DATA_WIDTH/8-1:0]	-	G9	Port width depends on the generic C_S_AXI_DATA_WIDTH
P18	S_AXI_BID[C_S_AXI_ID _WIDTH -1:0]	-	G10	Port width depends on the generic C_S_AXI_ID_WIDTH
P22	S_AXI_ARID[C_S_AXI_ID _WIDTH-1:0]	-	G10	Port width depends on the generic C_S_AXI_ID_WIDTH
P23	S_AXI_ARADDR[C_S_AXI _ADDR_WIDTH-1:0]	-	G8	Port width depends on the generic C_S_AXI_ADDR_WIDTH
P32	S_AXI_RID[C_S_AXI_ID _WIDTH-1:0]	-	G10	Port width depends on the generic C_S_AXI_ID_WIDTH
P33	S_AXI_RDATA[C_S_AXI _DATA_WIDTH-1:0]	-	G9	Port width depends on the generic C_S_AXI_DATA_WIDTH
P40	IP2Bus_Data[C_S_AXI _DATA_WIDTH-1:0]	-	G9	Port width depends on the generic C_S_AXI_DATA_WIDTH
P45	Bus2IP_Addr[C_S_AXI _ADDR_WIDTH - 1:0]	-	G8	Port width depends on the generic C_S_AXI_ADDR_WIDTH
P46	Bus2IP_Data[C_S_AXI _DATA_WIDTH-1:0]	-	G9	Port width depends on the generic C_S_AXI_DATA_WIDTH
P48	Bus2IP_BE[(C_S_AXI _DATA_WIDTH/8)-1:0]	-	G9	Port width depends on the generic C_S_AXI_DATA_WIDTH
P53	Bus2IP_CS[n:0]	-	G11	The vector width of Bus2IP_CS is the number of elements in C_ARD_ADDR_RANGE_ARRAY/2
P54	Bus2IP_RdCE[n:0]	-	G12	The vector width of Bus2IP_RdCE is the number of elements in C_ARD_NUM_CE_ARRAY
P55	Bus2IP_WrCE[n:0]	-	G12	The vector width of Bus2IP_WrCE is the number of elements in C_ARD_NUM_CE_ARRAY

Descriptions of Parameters and Their Usage

Address Range Definition Arrays

One of the primary functions of the AXI4 Slave Burst is to provide address decoding, chip enable/chip select control signal generation and optional read burst buffering.

The AXI4 Slave Burst employs VHDL generics that are defined as unconstrained arrays as the method for customizing address space decoding. These parameters are called the Address Range Definition (ARD) Arrays. There are two of these arrays used for address space definition in the axi_slave_burst. They can be recognized by the "C_ARD" prefix of the Generic name. The ARD Generics are:

- C_ARD_ADDR_RANGE_ARRAY
- C_ARD_NUM_CE_ARRAY

One of the big advantages of using unconstrained arrays for address decode space description is that it allows the user to specify as few or as many unique and non-contiguous AXI4 address spaces as the peripheral design needs.

The Slave Attachment decoding logic will be optimized to recognize and respond to only those defined address spaces during active AXI4 transaction requests.

Since the number of entries in the array can grow or shrink based on each user Application, the slave attachment is designed to analyze the user's entries in the arrays and then automatically add or remove resources, and interconnections based on the arrays' contents.

The ordering of a set of address space entries within the ARD arrays is not important. Each address space is processed independently from any of the other address space entries. **However, once an ordering is established in any one of the arrays, that ordering of the entries must be maintained in the other ARD array**. That is, the first two entries in C_ARD_ADRR_RANGE_ARRAY will be associated with the first CE Number entry in the C_ARD_NUM_CE_ARRAY.

C_ARD_ADDR_RANGE_ARRAY

The actual address range for an address space definition is entered in this array. Each address space is by definition a contiguous block of addresses as viewed from the host microprocessor's total addressable space. It's specification requires a pair of entries in this array. The first entry of the pair is the Base Address (starting address) of the block, the second entry is the High Address (ending address) of the block. These addresses are byte relative addresses. The array elements are defined as std_logic_vector(0 to 63) in the ipif_pkg.vhd file in Processor Common (proc_common) library. Currently, the largest address bus used on the AXI4 is 32 bits.

```
C_ARD_ADDR_RANGE_ARRAY : SLV64_ARRAY_TYPE : =
        --Base address and high address pairs.

          "X0000_0000_7000_0000",    --user control reg bank base address
          "X0000_0000_7000_FFFF",    --user control reg bank high address
          "X0000_0000_8000_0000",    --user status reg bank base address
          "X0000_0000_8000_FFFF",    --user status reg bank high address

          NOTE:
          In this example, there are two address pairs entered into the
          C_ARD_ADDR_RANGE_ARRAY VHDL Generic.

          This corresponds to the two address spaces being defined
          by the user. Each pair is comprised of a starting address and
          an ending address.

          Values are right justified and are byte address relative.
```

DS679_02

Figure 2: **Address Range Specification Example**

Note: The maximum burst length in the given address space must not exceed the 4k boundary.

For example, the minimum address space is needed that will include 4096 bytes (0x1000 hex) of the system memory space. Valid Base Address entries are 0x00000000, 0x00001000, 0xFFFFF000, 0x90001000, and so forth. A value of 0x00000120 is not valid because it is not a multiple of 0x1000 (4096). Also note that the High Address entry is equal to the assigned Base Address plus the block size minus 1. The address range can be of any length as shown in Figure 2.

C_ARD_NUM_CE_ARRAY

The slave decoding logic provides the user the ability to generate single chip enable within a single address space. Please note that each address space must have only 1 chip enable specified, for each address pair mentioned in C_ARD_ADDR_RANGE_ARRAY. Any other value entered may cause error-some results.

```
C_ARD_ADDR_NUM_CE : INTEGER_ARRAY_TYPE : =
        (
        1 - User Address Space 1
        1 - User Address Space 2
        )
```

NOTE:
In this example, the user defines the number of CE signals needed per address space.

Figure 3: **Chip Enable per Address Range Specification Example**

Available Support Functions for Automatic Ripping of CE and CS Buses.

The User may find it convenient to use some predefined functions developed by Xilinx to automatically rip signals from the Bus2IP_CS, Bus2IP_WrCE, and Bus2IP_RdCE buses. These functions facilitate bus ripping regardless of order or composition User functions in the ARD Arrays. This is extremely useful if User parameterization adds or removes User IP functions (which changes the size and ordering of the CS and CE buses).

These functions are declared and defined in the ipif_pkg.vhd source file that is located in the Xilinx EDK at:

```
\EDK\hw\XilinxProcessorIPLib\pcores\proc_common_v3_00_a\hdl\vhdl\ipif_pkg.vhd
```

The following library declaration must appear in the user's VHDL source:

```
library proc_common_v3_00_a;
use proc_common_v3_00_a.ipif_pkg.all;
```

Table 4 describes the above function and its usages.

Table 4: **Slave Attachment Support VHDL Functions**

VHDL Function Name	Input Parameter Name	Input Parameter Type	Return Type	Description
calc_num_ce	ce_num_array	INTEGER_ ARRAY_TYPE	Integer	This function is used to get the total number of signals that make up each of the Bus2IP_RdCE and Bus2IP_WrCE buses (they are all of same size and order) This information is derived from 'C_ARD_NUM_CE_ARRAY' parameter.

C_RDATA_FIFO_DEPTH

This parameter allows the inclusion of read FIFO in the AXI4 Slave Burst core design. The AXI4 Slave Burst core behavior during read from slave IP will be based upon this parameter. If C_RDATA_FIFO_DEPTH = 0, then FIFO will not be included in the design and every IPIC read towards slave IP will be considered as single reads. If C_RDATA_FIFO_DEPTH = 32, then FIFO will included in the design. In this case, every IPIC read to slave IP will be considered as burst read (except when AXI length is 1). Only 0 or 32 values are allowed.

C_INCLUDE_TIMEOUT_CNT

This parameter allows to include the time-out counter as a part of the design. If the user is not sure about responding mechanism of the IP, then this parameter should be set to '1'. Once it is set, then internal address and data phase

time-out counters will become active and will monitor the IPIC acknowledge behavior for address and data phase. The address and data phase time-out counters are mutually exclusive, meaning that initially only the address phase time-out counter will start and then data phase counter will start, if the address is properly acknowledged.

C_TIMEOUT_CNTR_VAL

This parameter is valid only when the C_INCLUDE_TIMEOUT_CNT = 1. The valid values to be set for this parameter are 8 or 16. The user should be aware of the IP requirements before setting this value. This value is internally provided to the address and data phase time-out counters. For both read and write transactions, this value will be loaded in the internal counters. The default value is 8.

C_ALIGN_BE_RDADDR

This parameter settings will be applicable only in all read transactions. When C_ALIGN_BE_RDADDR = 0, this parameter indicate that the byte enables generated on Bus2IP_BE will be all '1', irrespective of size (32 bit or 64 bit) of transfer. This means that, for every read at IPIC the slave IP should provide either 32 or 64 bit data based upon the data width size.

When assigned to 1, this parameter indicate that the byte enables generated on Bus2IP_BE will be always depend upon the size of transfer. For 32 bit data width and byte transfer as size, there would be only one byte enable active at a time. The address won't change on Bus2IP_Addr unless all the byte enables are activated term by term. The same implies for half word, word and double word transfer case.

C_S_AXI_SUPPORTS_READ and C_S_AXI_SUPPORTS_WRITE

This parameter decides the logic inclusion in AXI4 Slave Burst core.

Table 5: **Logic Inclusion for AXI4 Slave Burst IP Core**

C_S_AXI_SUPPORTS_READ	C_S_AXI_SUPPORTS_WRITE	Description
1	1	Both read and write paths will be the part of AXI4 Slave Burst design.
0	1	Only write paths will be the part of AXI4 Slave Burst design. The read transactions from AXI4 are not passed to the IPIC and will be returned with all '0's in response.
1	0	Only read paths will be the part of AXI4 Slave Burst design. The write transactions from AXI4 are not passed to the IPIC and will be returned with all '0's in response.
0	0	NA - The particular slot will be skipped. No control signals will be generated from the core either on AXI side or IPIC side interface.

Detailed Description - IPIC signals

Please note that the Bus2IP_CS, Bus2IP_RdCE and Bus2IP_WrCE are considered to be the IPIC control signals. The values driven on signals like Bus2IP_BE, Bus2IP_Addr, Bus2IP_Data are valid only when respective control signals are active.

Bus2IP_CS

The Bus2IP_CS signal width depends on the number of base-high address pairs (divided by two) declared in the C_ARD_ADDR_RANGE_ARRAY parameter. For each base-high address range, one Bus2IP_CS signal will be generated from the core. All other IPIC signals are qualified only when this signal is active. This signal used in conjunction with Bus2IP_RNW and corresponding read and write chip enable signal and is especially suited for reading and writing to memory type devices.

Bus2IP_RNW

Bus2IP_RNW is a signal indicating the type of transfer in progress and is valid when Bus2IP_CS and Bus2IP_WrCE or Bus2IP_RdCE is asserted. This signal will be driven logic high in case of read transaction and driven low in case of write transactions. The Bus2IP_RNW signal will active till the IPIC completes the transaction with slave IP.

Bus2IP_RdCE

The Bus2IP_RdCE signal width is depending upon the number of base-high address pairs declared in the C_ARD_ADDR_RANGE_ARRAY parameter. It is must that C_ARD_NUM_CE_ARRAY parameter must be set to '1' for each base-high address pair mentioned in the C_ARD_ADDR_RANGE_ARRAY. In case of read transactions from AXI4, at any given time, there will be only one active Bus2IP_RdCE signal which coincides with the Bus2IP_CS signal. Based upon either burst or single read transaction, this signal will be either active through out the transaction or, will be active only till the transaction is over. If C_RDATA_FIFO_DEPTH = 32 is mentioned in configuration, then based upon the AXI4 reading and slave core data sending speed, this signal may toggle in between the transactions. The de-assertion of this signal in between means that there is no space left in the read buffer in the core. If AXI4 does at least sixteen reads, then again this signal will be asserted on IPIC.

Bus2IP_RdReq

Bus2IP_RdReq indicates the requested cycle is a read type transfer. This signal behaves differently for read burst transaction as well as for read single transaction based upon C_RDATA_FIFO_DEPTH parameter settings. In case of single transaction (C_RDATA_FIFO_DEPTH = 0), this will be active only for one clock cycle at the start of each transfer in given transaction. In burst mode (C_RDATA_FIFO_DEPTH = 1), this signal is active till the last transaction's address acknowledge. For burst cycles, Bus2IP_RdReq will assert for each valid address of the burst cycle. For example if a 4 data beat burst read is being requested and the user slave IP drives IP2Bus_AddrAck ahead of IP2Bus_RdAck then Bus2IP_RdReq will de-assert after the 4 address is acknowledged which will be before the 4th data beat is acknowledged.

Bus2IP_WrCE

The Bus2IP_WrCE signal width is depend upon the number of base-high address pairs declared in the C_ARD_ADDR_RANGE_ARRAY parameter. It is must that C_ARD_NUM_CE_ARRAY parameter must be set to '1' for each base-high address pair mentioned in the C_ARD_ADDR_RANGE_ARRAY. In case of write transactions from AXI4, at any given time, there will be only one active Bus2IP_WrCE signal which coincides with the Bus2IP_CS signal.

Bus2IP_WrReq

Bus2IP_WrReq indicates when write data is valid on Bus2IP_Data bus. This signal behaves differently for write burst transaction as well as for write single transaction. In burst mode, this signal is active till the last transaction. In case of single transaction, this will be active only for one clock cycle at the start of transaction.

Bus2IP_BE

Bus2IP_BE is a vector of width C_S_AXI_DATA_WIDTH/8. Bus2IP_BE becomes valid coincident with Bus2IP_CS. During AXI4 write transactions, the S_AXI_WSTRB signal will drive this signal for each transaction beat in complete transfer. During AXI4 read transactions, based upon C_ALIGN_BE_RDADDR parameter setting this signal will be driven with all '1' or based upon the size it will be aligned to address for each transaction beat in complete transfer.

Bus2IP_Addr

Bus2IP_Addr is a 32 Bit vector that drives valid when Bus2IP_CS and Bus2IP_RdCE or Bus2IP_WrCE drives high. During AXI4 read or write transactions, this signal will be initially driven by the AXI4 read or write channel address values. Later on during write, with each address acknowledge from slave IP, based upon the type of transaction, burst size and burst length the address will be either fixed or incremented to next word or double word aligned boundary. During read, with each address acknowledge from slave IP, based upon burst length the value will be either fixed or incremented to next word or double word aligned boundary. For FIXED transactions from AXI4, the address on this signal will be constant throughout the transaction. For INCR or WRAP transactions from AXI4, the address will increment based upon IPIC data width.

The new address will be available on Bus2IP_Addr bus and it will change only in valid address acknowledge conditions. The valid address acknowledge condition combines the availability of read or write chip enable and the address acknowledge generated by the slave IP. If the valid read or write chip enable is not available during the transaction, the address acknowledge generated by the slave IP wont be considered in the AXI4 Slave Burst IP.

Bus2IP_Data

This signal will be driven during write operation from AXI4. The data will remain same till the slave IP accepts the data or generates the address/data phase time out signal. The value of this bus is valid only when the Bus2IP_CS and Bus2IP_WrCE signals are active.

Bus2IP_BurstLength

For all AXI4 transactions, this signal indicates the number of beats on IPIC. So the slave IP core need to monitor this signal to get the exact number of intended transfers from IPIC. The AXI4 read transaction length or write transaction length will be registered and made available on this signal till the completion of IPIC transaction. This signal is valid only when the Bus2IP_CS is active. The Bus2IP_BurstLength will be carrying the similar information about the transaction length which is available in AXI4 reference document.

Bus2IP_Burst

Bus2IP_Burst indicates that the current cycle is a burst cycle when asserted or a single beat transfer when not asserted. This signal will be active till second to last burst transaction acknowledged from IPIC. During single transactions, this signal will remain de-asserted. The activeness of this signal indicates that the IPIC is doing burst transaction to slave IP.

Type_of_transfer

Type of transfer signal is indication to slave IP about the exact type of transaction from AXI4.

If Type_of_transfer signal is '0', then there is FIXED burst transaction from AXI4. If Type_of_transfer signal is '1', then there is either INCR or WRAP burst transaction from AXI4. This indication will help slave IP to decide either to support the transaction or to generate the error response. For example, if any slave IP is not supporting the FIXED type of AXI4 transaction (where Bus2IP_Address will remain constant throughout the transaction), then the slave IP should respond with error till the transaction from IPIC ends and vice versa for INCR/WRAP transaction. This signal is valid only when the Bus2IP_CS is active.

IP2Bus_Data

This data bus is slave IP core's response for the read transaction from the AXI4 Slave Burst core. The data on this bus will be valid only when the Bus2IP_CS, Bus2IP_RNW and bus2IP_RdCE are active. It is expected that the this bus is registered bus in slave IP before reaching to the IPIC interface.

IP2Bus_RdAck

This is data qualifier signal from slave IP for read transaction initiated by the AXI4 Slave Burst core. A active high on this signal will indicate that the data on IP2Bus_Data is valid and should be passed to AXI4 interface. It is expected that this signal is registered in the slave IP before reaching to IPIC interface.

IP2Bus_AddrAck

This is address qualifier signal on IPIC from the slave IP. The active transaction on this signal means that the slave IP have accepted the address presented on Bus2IP_Addr bus. With valid address acknowledge signal from slave IP, based upon the size of transfer, the address counter will be incremented in the AXI4 Slave Burst core. The IPIC will consider this signal as valid only when it is asserted when the Bus2IP_CS and corresponding read or write chip enables are active on IPIC. It is expected that this signal is registered in the slave IP before reaching to IPIC interface.

IP2Bus_WrAck

This is data qualifier signal from slave IP for write transaction initiated by the AXI4 Slave Burst core. A active high on this signal will indicate that the data on Bus2IP_Data has been accepted by the slave IP. It is expected that this signal is registered in the slave IP before reaching to IPIC interface.

IP2Bus_Error

The active high on this signal indicates that there is error in either address or data access in the slave IP. This signal must be generated along with IP2Bus_WrAck (in case of write) or IP2Bus_RdAck (in case of read) then only it will be treated as valid error in the AXI4 Slave Burst core. It is expected that this signal is registered in the slave IP before reaching to IPIC interface.

Detailed Description - IPIC signal Behavior For C_RDATA_FIFO_DEPTH

It is recommended that all the IPIC signals from the slave IP are registered in the IP before reaching to IPIC interface and to the AXI4 Slave Burst Core. This will help in improving timing for the complete IP.

Below is the IPIC write transaction behavior.

In this condition, the write request from AXI4 will be transferred on IPIC as below. This behavior is common for C_RDATA_FIFO_DEPTH 0 or 32 condition.

- Bus2IP_BurstLength - This signal will indicate the number of beats in the transaction.
- Bus2IP_Burst - Applicable only when the AXI4 transfer is of burst type. This signal will be active until the last but one write transaction on IPIC. In case of single transfer this signal will be active low only.
- Bus2IP_CS - This signal will be active until the end of transaction for corresponding address range.
- Bus2IP_WrCE - This signal will be either active or toggle in between based upon AXI4 write speed.
- Bus2IP_Address - This signal carries the word (32 bit data width)/double word (64 bit data width) boundary addresses and changes based upon address acknowledge or address time out condition which is based upon size of transfer. If the proper address acknowledge is received from slave IP, the logic waits for data acknowledge. If from slave IP address acknowledge does not appear on, or before, the address time-out count, the address will be advanced to the next (data sized) location. The address increment to the next boundary depends on the size, burst type, and burst length of AXI4 signals. For example, for 64 bit IPIC data width, if the transaction from AXI4 is byte burst, the user IP should generate the eight address acknowledge to advance the IPIC address to the next double word boundary. For 32 bit IPIC, with the same transaction from AXI4, the IPIC will wait for 4 address acknowledge signals to advance to the next word boundary.
- Bus2IP_BE - The byte strobe from AXI4 will be transferred on this signal as it is per transaction.

- Bus2IP_WrReq - This signal will be active through out the transaction and will be de-asserted with last transactions address acknowledge.
- Bus2IP_RNW - This signal will be active low through out the transaction.

The read behavior is different between the C_RDATA_FIFO_DEPTH = 0 and the C_RDATA_FIFO_DEPTH = 32 conditions.

- C_RDATA_FIFO_DEPTH = 0

In this condition, the read request from AXI4 will be transferred on IPIC as below.

- The IPIC considers read transactions as single transactions.
- Bus2IP_BurstLength - This signal will be '0' for the complete transaction.
- Bus2IP_Burst - This signal will be '0' for the complete transaction.
- Bus2IP_CS - This signal will be active until the end of each single transaction for corresponding address range.
- Bus2IP_RdCE - This signal asserts and de-assert in each of the read transaction for corresponding address range along with Bus2IP_CS.
- Bus2IP_Address - This signal carries the word (32 bit dwidth)/double word (64 bit dwidth) boundary addresses and changes based on the address acknowledge or address time out condition. If the proper address acknowledge from slave IP is received, the logic waits for data acknowledge. The address updates to the next boundary based on either the data acknowledge or data time out condition. If address acknowledge from slave IP does not appear on, or before, the address time out count, the address advances to the next location.
- Bus2IP_BE - This signal carries valid byte lane enable signals based on C_ALIGN_BE_RDADDR parameter setting for each read transaction.
- Bus2IP_RdReq - This signal will be active only for one clock cycle in each IPIC read transaction.
- Bus2IP_RNW - This signal will be active in each IPIC read transaction.

- C_RDATA_FIFO_DEPTH = 32

In this condition, the read request from AXI4 will be transferred on IPIC as below.

- The IPIC will consider read transactions as single transactions.
- Bus2IP_BurstLength - This signal represents the number of beats in a given transaction.
- Bus2IP_Burst - This signal will be active until last, except one, for the active transaction.
- Bus2IP_CS - This signal will be active until the end of the transaction for the corresponding address range.
- Bus2IP_RdCE - This signal asserts along with the Bus2IP_CS. With the assertion of this signal, the IPIC will start the data collection from slave IP. Based on the AXI reading speed, the slave IP data will be stored temporarily in the local buffer which can store 32 beats of data. If the AXI reading speed is very slow, this buffer will be completely filled. In this scenario, this signal de-asserts and again re-asserts when the AXI reads data from the core buffer. At power on reset condition, this signal will be active till last-but-two spaces are left in the FIFO. After de-assertion, this signal will be asserted when read from AXI interface generates at least 16 empty spaces in local buffer. There after, the core supports each burst of 16 after the initial power on reset condition is over.
- Bus2IP_Address - This signal carries the word (32 bit dwidth)/double word (64 bit dwidth) boundary addresses and changes based on the address acknowledge or address time out condition. If the proper address acknowledge from the slave IP is received, the logic waits for data acknowledge. If address acknowledge from slave IP does not appear on, or before, the address time out count, the address advances to the next location. The core supports read address pipeline of 16 on the IPIC interface. There may be a difference in AXI read and IPIC read speeds, depending on which of the read FIFO in the core is

completely filled. In this scenario, the Bus2IP_RdCE will de-assert and last address will be stored in Bus2IP_Address bus.

- Bus2IP_BE - This signal will carry valid byte lane enable information based upon C_ALIGN_BE_RDADDR parameter setting for each read transaction. If C_ALIGN_BE_RDADDR = 1, then byte enables will be based upon the size of transfer. If C_ALIGN_BE_RDADDR = 1 and if Bus2IP_RdCE de-asserts and last transactions byte enables will be stored on Bus2IP_BE.
- Bus2IP_RdReq - This signal will be active throughout the read transction. It will be de-asserted once last transaction's address acknowledge is received from slave IP.

Detailed Description - IPIC signal Behavior For C_ALIGN_BE_RDADDR

The parameter C_ALIGN_BE_RDADDR is applicable only for read transactions.

When C_ALIGN_BE_RDADDR = 0, ir-respective of the size of transfer, the Bus2IP_BE signal will be tied to all '1'.

When C_ALIGN_BE_RDADDR = 1, the Bus2IP_BE vector signal will be representing the active byte lane based upon the size of transfer and will be aligned to Bus2IP_Addr.

Detailed Description - Core Behavior For C_INCLUDE_TIMEOUT_CNT

The parameter C_INCLUDE_TIMEOUT_CNT is applicable for address phase and data phase in given transaction. For every AXI4 transaction, there is address phase and corresponding data phase on IPIC towards user IP.

When C_INCLUDE_TIMEOUT_CNT = 1 (C_TIMEOUT_CNTR_VAL **should be assigned time-out count value)**

- The time -out counters will be included in the design, which are applicable for address and data phase for corresponding transactions.
- The address phase counter will start during the address phase and if the user IP generates the address acknowledge, then corresponding data phase counter will start.
- If user IP doesn't respond the address phase, then SLVERR will be generated for the corresponding address and there will not be data phase. The new address phase will start.
- If the user IP responds the present address phase then the data phase timeout counter will start for corresponding address phase. If the user IP doesn't respond the data phase in given specific time, then SLVERR will be generated.
- Please refer Time-out Condition Details section for more information on Time Out conditions.

When C_INCLUDE_TIMEOUT_CNT = 0 (C_TIMEOUT_CNTR_VAL **will be ignored**)

- It is expected that user is aware that there is no mechanism in the AXI Slave Burst core to generate SLVERR conditions in case of address phase and data phase time out conditions.
- It is user IP's responsibility to generate proper address acknowledges and corresponding data acknowledges for proper behavior of the core. In any other cases, if the address acknowledges or data acknowledges are missed, then the AXI Slave Burst core may not behave properly.

Design Details

The AXI4 transactions received by the AXI slave burst interface will be translated to IPIC interface based upon type, size and length of AXI transfer.

Round-robin arbitration logic avoids the starvation between read and write transactions.

At system reset the initial priority is given to the read write transfer. Priority and then toggles between each AXI transaction. At any given point it is possible to have read followed by another read if the write is not requested by AXI before the completion of first read and the next read request is pending on AXI4. Similarly, write followed by write is also possible if a read is not requested by AXI4 while the write request is pending.

The write data from AXI is directly sent to the IPIC - no buffering is done. This design expects the slave IPs to check the Bus2IP_WrCE before asserting the write address request on the target. As the S_AXI_WVALID can be de-asserted by AXI master during burst and write are not buffered. The condition of having Bus2IP_WrBurst going high but Bus2IP_WrCE going low may occur. To reduce the write latency, the S_AXI_WREADY signal is generated using combinational logic, all other AXI4 interface signals are registered in the core.

During the read, there is flexibility of data buffering from IPIC. The buffering is possible only when the parameter C_RDATA_FIFO_DEPTH = 32. The internal FIFO's data width is same as C_S_AXI_DATA_WIDTH parameter. Flexibility of temporary data storage allows IPIC to run faster than AXI reading speed under certain conditions. If C_RDATA_FIFO_DEPTH = 0, then every read transfer will be considered as single transaction. So IPIC and AXI need to correlate in terms of reading the data from slave IP in this particular case.

Address Generation Logic

The address generation logic is crucial part of design, which is responsible for generating the correct addresses on the IPIC signals. The address generation logic will accept the incoming address from AXI. (If the address falls in the BASE-HIGH address range, then IPIC control signals will be generated). As the AXI burst is restricted to 4k burst boundary, the lower 12 bits of AXI address (either read/write) will be used for address variation on IPIC from the base or the first address from the AXI transaction. The address on the IPIC will advance to the next value when a valid address acknowledge is generated by the slave IP (based upon the size of transfer), probably the Read Chip Enable or Write Chip Enable signal is active. The AXI transaction size determines the actual address acknowledge to be considered for address change. For example, if the AXI address is word aligned and transaction from AXI is byte burst, then slave IP need to generate the four address acknowledge's to make new address available on IPIC.

Address Decode Logic

The address decode logic generates the valid chip select and chip enable signals for the selected address range. During a read transaction, if C_RDATA_FIFO_DEPTH = 32, and the FIFO is full due to a difference in the operating speed of AXI reads and IPIC reads, the read chip enable signals will be de-asserted. If AXI reads data at the same rate as the speed of IPIC, then the read chip enable signal will be active throughout the transaction. In short, the chip select signal will be active for the complete transaction, while the read and write chip enable signals may be de-asserted until some activities happen from the AXI.

If C_RDATA_FIFO_DEPTH = 0, then during the entire write transaction, the write chip enable will be active. During read transaction, every transfer is considered as a single transaction and chip select, with read chip enable signals will toggling for each transfer.

Read chip enable signals are active only when C_S_AXI_SUPPORTS_READ = 1.

Write chip enable signals are active only when C_S_AXI_SUPPORTS_WRITE = 1.

Read Data Path Logic

This logic is included in the design when C_S_AXI_SUPPORTS_READ = 1. If C_S_AXI_SUPPORTS_READ = 0, then all the IPIC and AXI read transfer related signals will be assigned to '0'.

The read data path has an optional FIFO for improving AXI burst throughput. If C_RDATA_FIFO_DEPTH = 32, the IPIC will generate a burst equal to FIFO depth towards slave IP. The slave IP can provide data until the read chip enable is active. When a read chip enable signal is de-asserted, then any data and data acknowledgements will be ignored by the AXI Slave Burst core. The slave IP can re-send the previously ignored data when the slave burst core reloads the address. The FIFO logic is implemented so that, the read chip signal will be enabled again when there is empty space of at least a single location in the FIFO due to an AXI read.

The read data path also includes the logic for read transfer qualifier signals for AXI like S_AXI_RLAST, S_AXI_RVALID and S_AXI_RRESP.

When C_RDATA_FIFO_DEPTH = 32 and the AXI reading speed rate is slower than the IPIC reading rate of data from the slave IP, then the IPIC will continue reading the data from slave IP until the FIFO down-counts to the last count except two. Once this count is reached, then the IPIC will suspend the read operation on the IPIC side. During this period only the Bus2IP_RdCE signal will be de-asserted while the Bus2IP_CS is still in the asserted state. Once the AXI reads at least 16 locations from FIFO, then IPIC will re-initiate the Bus2IP_RdCE signal.

Control State Machine Logic

The control state machine is the basis of the AXI slave burst design which is included in design when any of the parameters C_S_AXI_SUPPORTS_READ and/or C_S_AXI_SUPPORTS_WRITE are set to 1.

This block includes the following functionality -

- address, data counter to track the number of address and data acknowledge generated by the slave IP
- address and data time out counter if C_INCLUDE_TIMEOUT_CNT = 1
- separate state machines for C_RDATA_FIFO_DEPTH = 0 and C_RDATA_FIFO_DEPTH = 32 conditions
- address aligned byte enable generation logic for read transactions only
- AXI write response generation logic
- round robin logic to prevent starvation of read/write AXI transfers

Transaction Translation

The translation of transactions from AXI to IPIC are shown in Table 6.

Table 6: **AXI Slave Burst IPIF Parameter - Port Dependencies**

AXI Transaction	IPIC Transaction	Notes
Read/Write of type FIXED of burst length 1-16	Burst read or write of length 1-16	Address remains same through out the transfer. During writes, the S_AXI_WSTB is transferred as Bus2IP_BE. During reads, based upon the setting of parameter C_ALIGN_BE_RDADDR, the Bus2IP_BE will be all '1' or address aligned based upon the size of transfer.
Read/Write of type INCR of burst length 1-256	Burst read or write of length 1-256	Address will increment depending on the size of transaction. During write S_AXI_WSTB is transferred as Bus2IP_BE. During read based upon setting of parameter C_ALIGN_BE_RDADDR, the Bus2IP_BE will be either all '1' or address-aligned based upon the size of transfer.
Read/Write of type WRAP of burst length 2/4/8/16	Burst read or write of length 2/4/8/16	Correct address generation is handled by the core. The start address will be the same as the address received from AXI, and the remaining address will be incrementally generated until the cache wraps over. After this, the next address will be at the top of the cacheline and the remaining addresses are incremented until the last data transfer is done. During read based upon setting of parameter C_ALIGN_BE_RDADDR, the Bus2IP_BE will be all '1' or address-aligned based upon the size of transfer.

Latency Data for AXI Slave Burst Core

Table 7: **Latency Type**

Latency Type	To and From	Clock Cycles
Read Latency	ARVALID: RVALID of first data (with and without FIFO inclusion)	3 clocks
Write Latency	WVALID: WREADY of first data.	0 Clocks

Bus Throughput

Table 8: **Write Only**

System Number	Burst Type and Details	Bus Utilization
1	INCR (100 Transactions / BL 256)	99.24
2	FIXED AND WRAP (100 Transactions / BL 16)	94.41

Table 9: **Read Only**

System Number	Burst Type and Details	Bus Utilization (C_RDATA_FIFO_DEPTH=O)(1)	Bus Utilization (C_DATA_FIFO_DEPTH=32)
1	INCR (100 Transactions / BL 256)	49.9	99.2
2	FIXED AND WRAP (100 Transactions / BL 16)	49.48	88.8

Notes:

1. When C_RDATA_FIFO_DEPTH=0, the Burst transfers are converted into single transactions on the IPIC; it introduces a clock delay for each beat transferred on the IPIC which causes the bus utilization to drop by half.

Table 10: **Write and Read**

System Number	Burst Type and Details	Bus Utilization (C_RDATA_FIFO_DEPTH=O)	Bus Utilization (C_DATA_FIFO_DEPTH=32)
1	INCR (100 Transactions / BL 256)	33.2 (33.2 and 33.3)	49.8 (49.8 and 49.7)
2	FIXED AND WRAP (100 Transactions / BL 16)	32.4 (33.3 and 31.47)	45.925 (47.2 and 44.65)

Byte Invariance

AXI4 is little endian and the same will be seen on IPIC interface. User IPs must manage byte invariance, if any.

AXI Write Transaction - Throttling Support

AXI master/interconnect can throttle the write data by deasserting the S_AXI_WVALID during a burst transfer. The Bus2IP_WrCE is de-asserts in response to this while the Bus2IP_Burst remains asserted.

This may be a change to existing slaves on the IPIC if they are using Bus2IP_Burst without checking the Bus2IP_WrCE.

AXI Read Transaction - Throttling Support

AXI allows the slave to throttle the data (S_AXI_RREADY = ’0’ when S_AXI_ARVALID =’ 1’), but on the IPIC the read data has to be accepted. To resolve this condition, optional read data buffer of depth determined by parameter C_RDATA_FIFO_DEPTH is implemented in the design.

Response Signaling

For a write transaction, there is only one response given for the entire burst and not for each data transfer within the burst.

In a read transaction, the slave can give different responses for different transfers within the burst. If any error occurs during the burst, the read response for that beat will be different than for other successful read transactions.

The AXI Slave Burst IP core will perform the required number of data transfers, even in the error conditions.

Slave Error condition

There are error scenarios supported in the core. While propagating these scenarios, only the Slave Error will be generated from the core.

- The IP2Bus_Error assertion on IPIC interface along with IP2Bus_WrAck causes the error response for write.
- The IP2Bus_Error assertion on IPIC interface along with IP2Bus_RdAck causes the error response for read.
- After completion of the address phase, if the IP fails to generate the data acknowledge, then data time-out condition will happen and causes the error response for that transaction.

Time-out Condition Details

The AXI Slave Burst IP includes address and data phase time-out counters. Both the counters become active in the design if C_INCLUDE_TIMEOUT_CNT = 1. The intention of including these time-out conditions is to complete the AXI request even though the slave IP is not responding in the address or data phase.

Address Phase Time-out Conditions

- Slave IP generates Bus2IP_Addr acknowledge before the final value of C_TIMEOUT_CNTR_VAL.

This address acknowledge will be considered as the valid address acknowledge which completes the current address phase, and then the next address will be available on Bus2IP_Addr bus.

- Slave IP generates Bus2IP_Addr acknowledge along with the final value of C_TIMEOUT_CNTR_VAL.

Even though the Bus2IP_Addr acknowledge from the slave IP and address time-out from the internal counter coincide, the priority is given to the Bus2IP_Addr acknowledge. This address acknowledge will be considered as valid address acknowledge, then next address will be available on Bus2IP_Address bus.

- Slave IP generates Bus2IP_Addr acknowledge after the final value of C_TIMEOUT_CNTR_VAL.

If the slave IP does not respond within the C_TIMEOUT_CNTR_VAL value, the AXI Slave Burst generates the address time-out condition. There will not be a data phase. The core will generate a SLVERR response on the AXI4 interface. The next address phase will start, where the internal address time-out counter starts monitoring the new address phase. In this case, the address acknowledge generated by the slave IP will be considered as valid for the new available address.

Data Phase Time-out Conditions

The AXI Slave Burst IP includes address and data phase time-out counters. Both the counters become active in the design if C_INCLUDE_TIMEOUT_CNT = 1. The data phase time out counter becomes active only when the slave IP generates the address acknowledge for the present address on the Bus2IP_Addr.

- Slave IP generates Bus2IP_Rdack/Bus2IP_Wrack before the final value of C_TIMEOUT_CNTR_VAL.

This data acknowledge is considered as valid data acknowledge.

- Slave IP generates Bus2IP_Rdack/Bus2IP_Wrack along with, or later than, the final value of C_TIMEOUT_CNTR_VAL.

This data acknowledge will not be considered as the valid data acknowledge, and error message (SLVERR) will be generated for this transaction.

Timing Diagrams

- C_RDATA_FIFO_DEPTH=0

Figure 4 represents AXI Slave Burst IP cores typical response for INCR read-write transactions from AXI4.

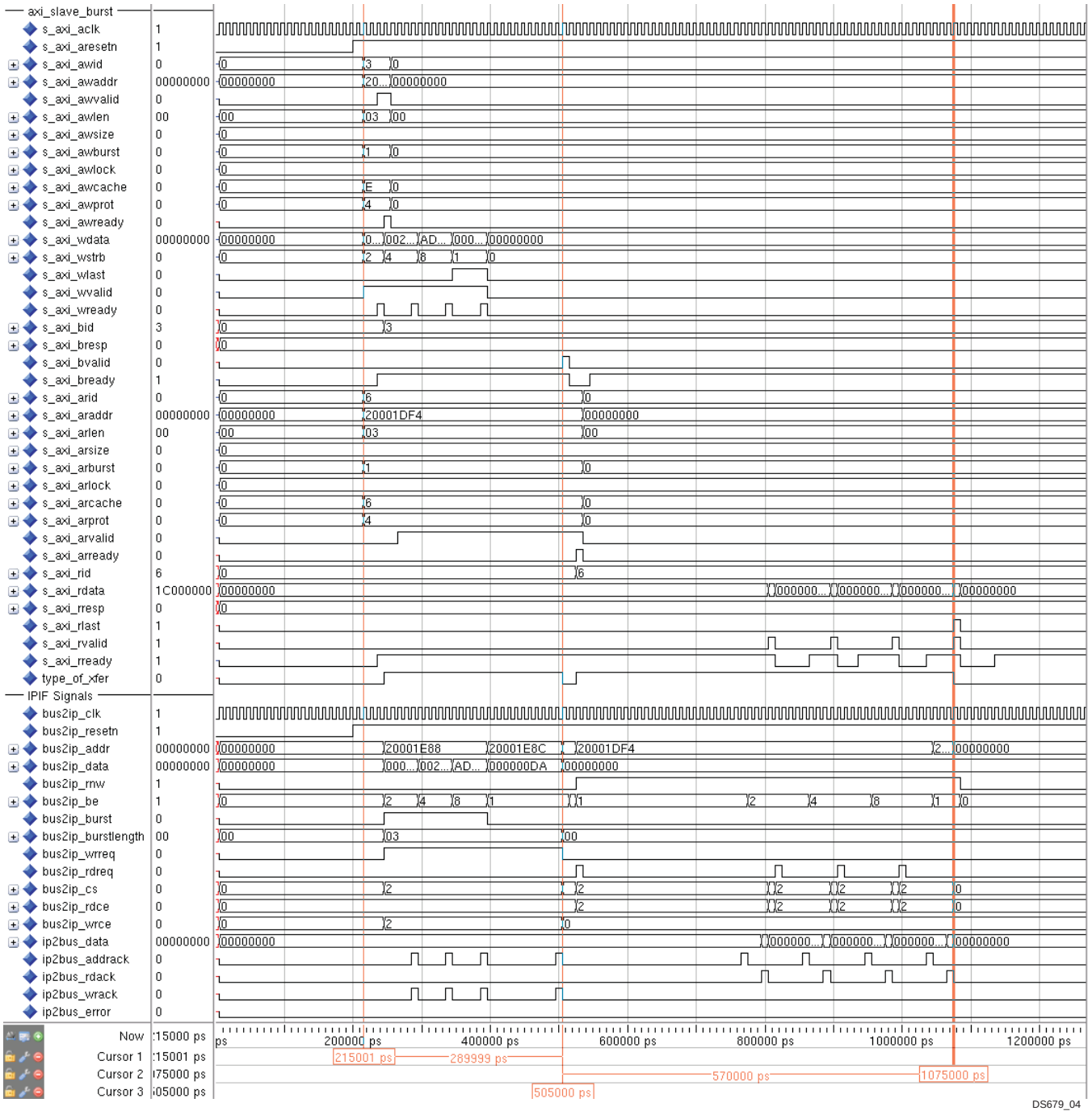


Figure 4: **AXI Slave Burst Response For AXI4 INCR Mode Transactions (FIFO Not Included)**

Figure 5 represents AXI Slave Burst IP core response for FIXED read-write transactions from AXI4.

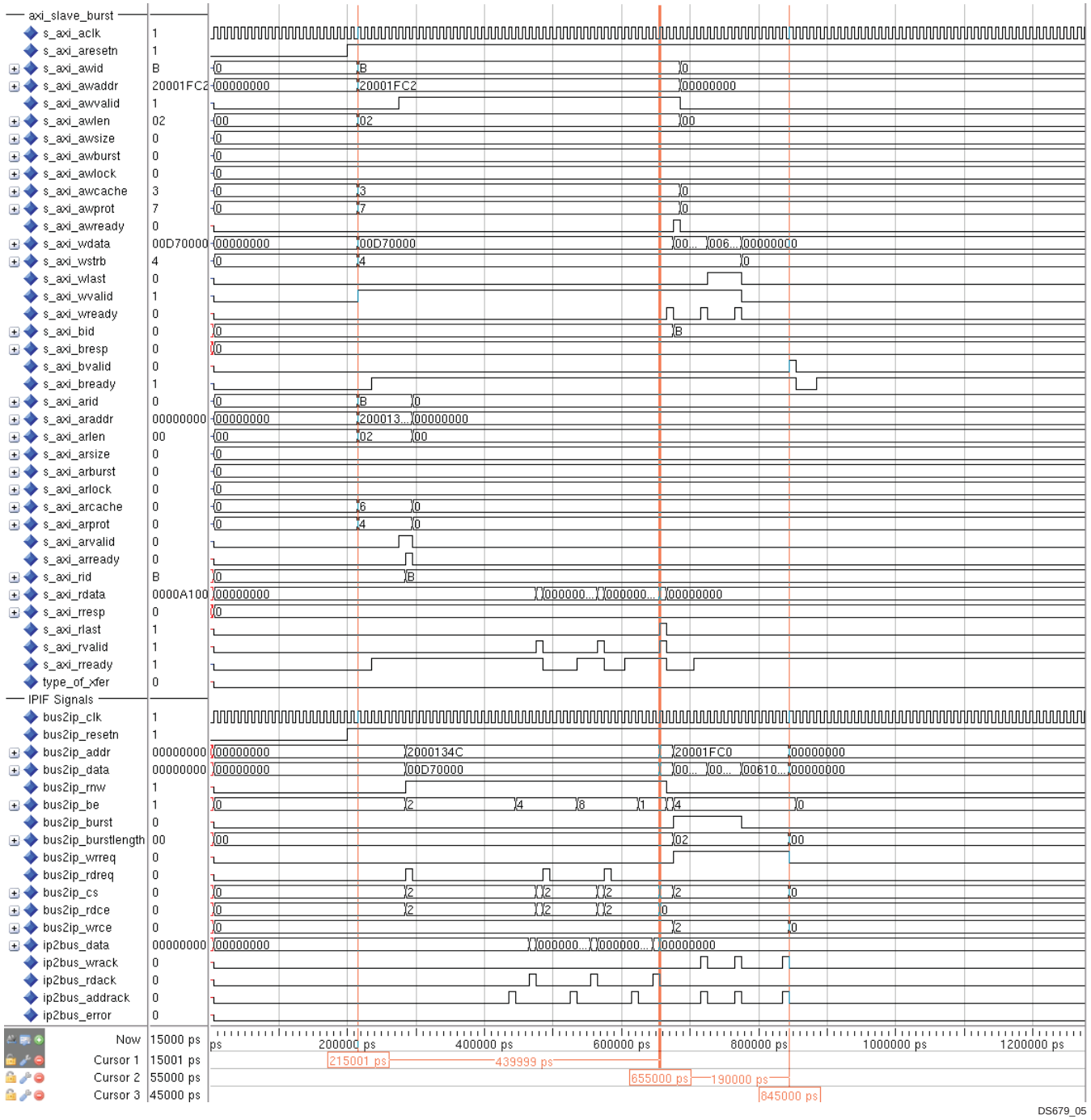


Figure 5: **AXI Slave Burst Response For AXI4 FIXED Read-Write Transactions (FIFO Not Included)**

Figure 6 shows AXI Slave Burst IP core response for WRAP read-write transactions from AXI4.

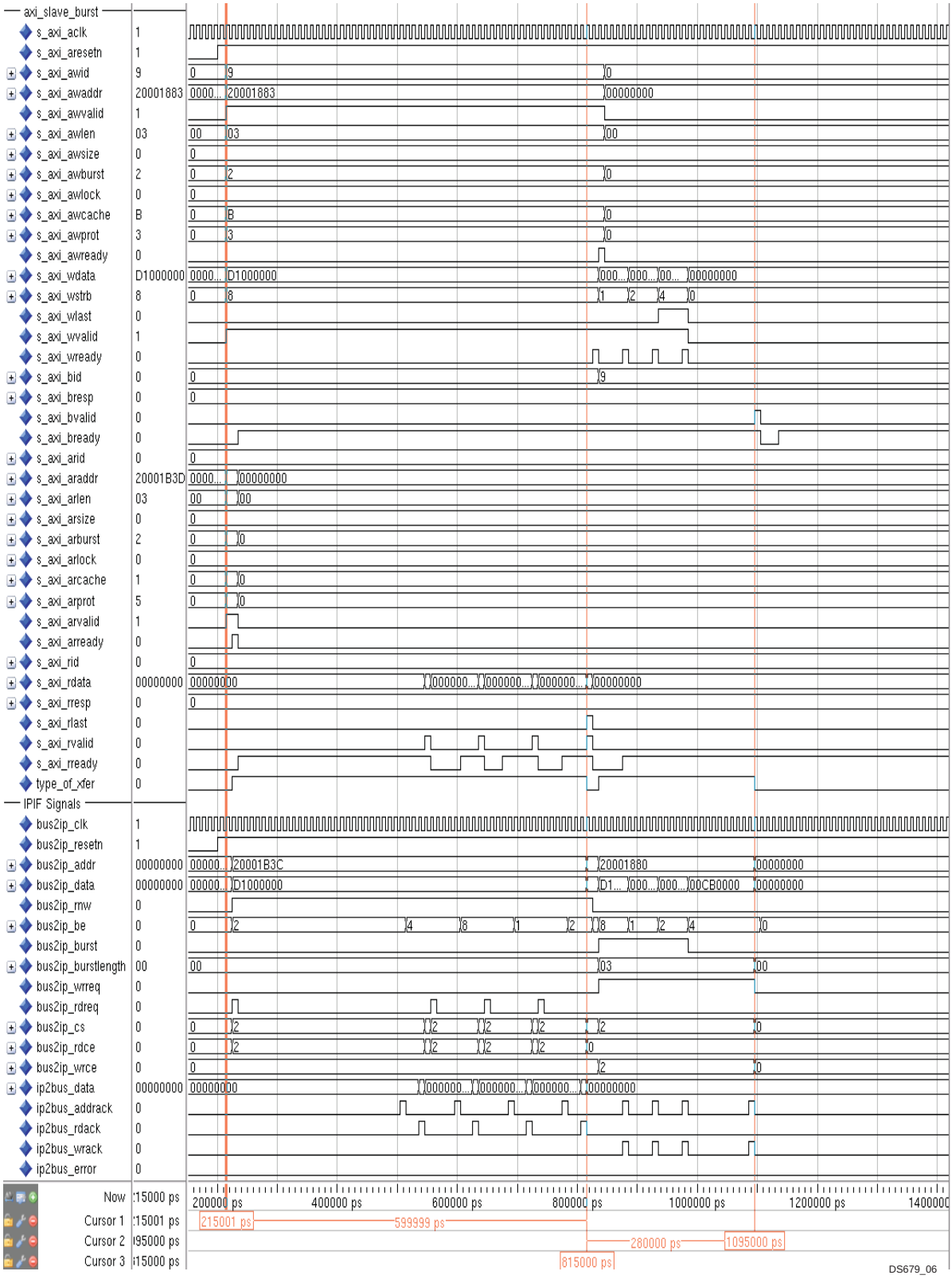


Figure 6: **AXI Slave Burst Response For AXI4 WRAP Read-Write Transactions (FIFO Not Included)**

Figure 7 shows core response for slave generating address time out conditions for INCR read-write transactions from AXI4. The transactions are completed with SLVERR response.

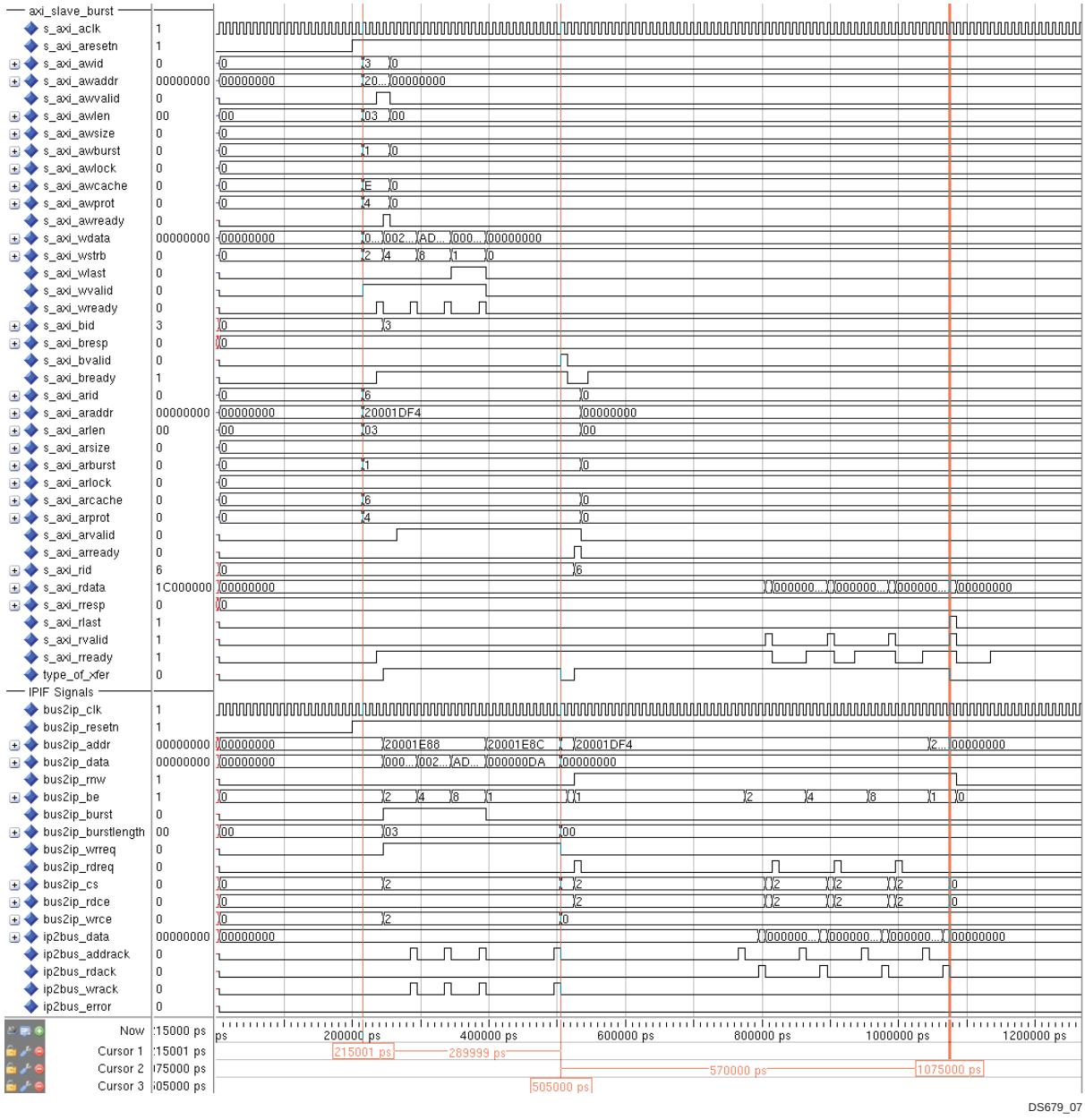


Figure 7: **AXI Slave Burst Response For AXI4 INCR Mode Transactions When Slave IP Does Not Respond the Address Available on Bus2IP_Addr (FIFO Not Included)**

Figure 8 shows core response for slave generating address time out conditions for FIXED read-write transactions from AXI4. The transactions are completed with SLVERR response.

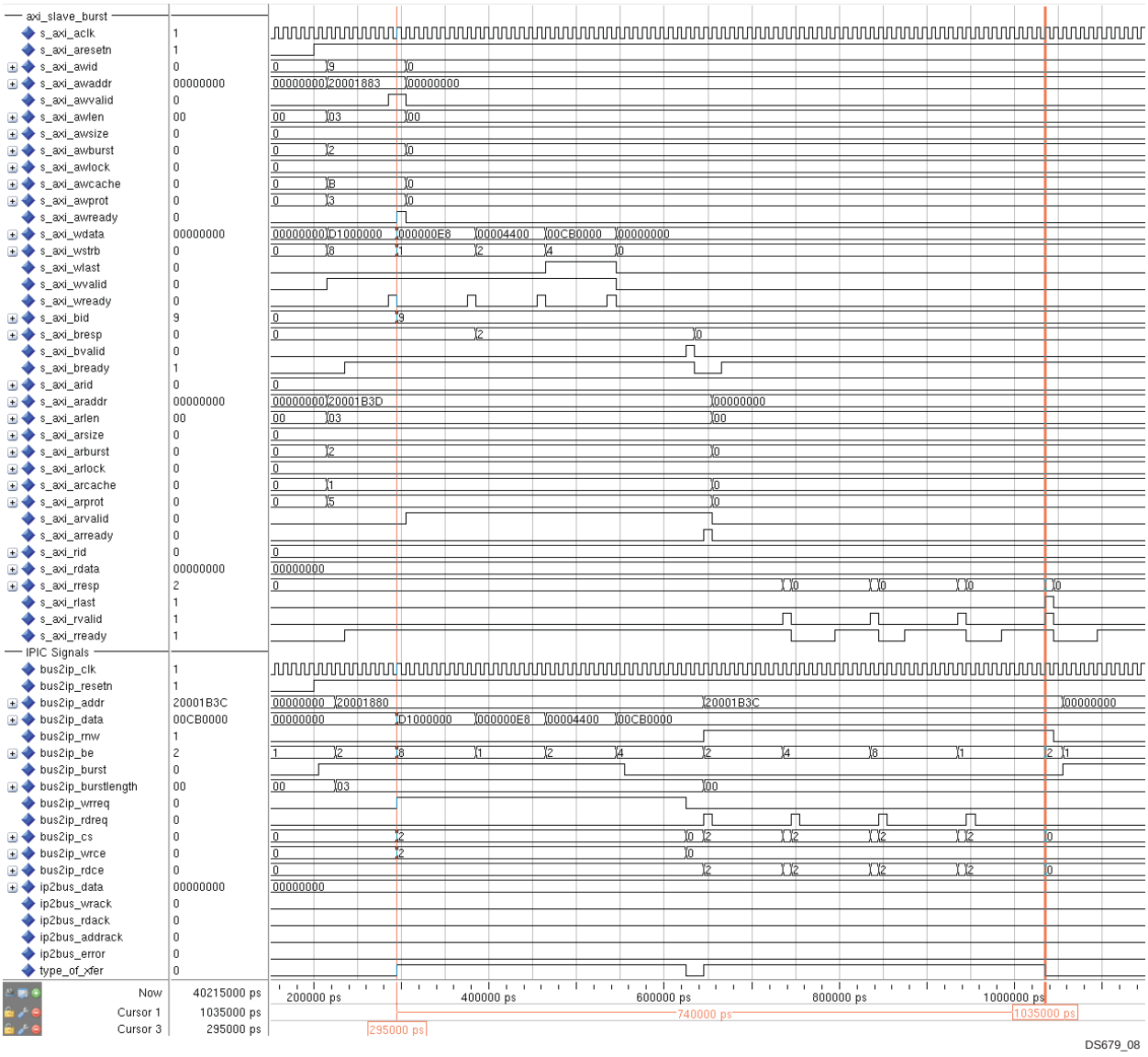


Figure 8: **AXI Slave Burst Response For AXI4 FIXED Mode Transactions When Slave IP Does Not Respond the Address Available on Bus2IP_Addr (FIFO Not Included)**

Figure 9 shows core response for slave generating address time out conditions for WRAP read-write transactions from AXI4. The transactions are completed with SLVERR response.

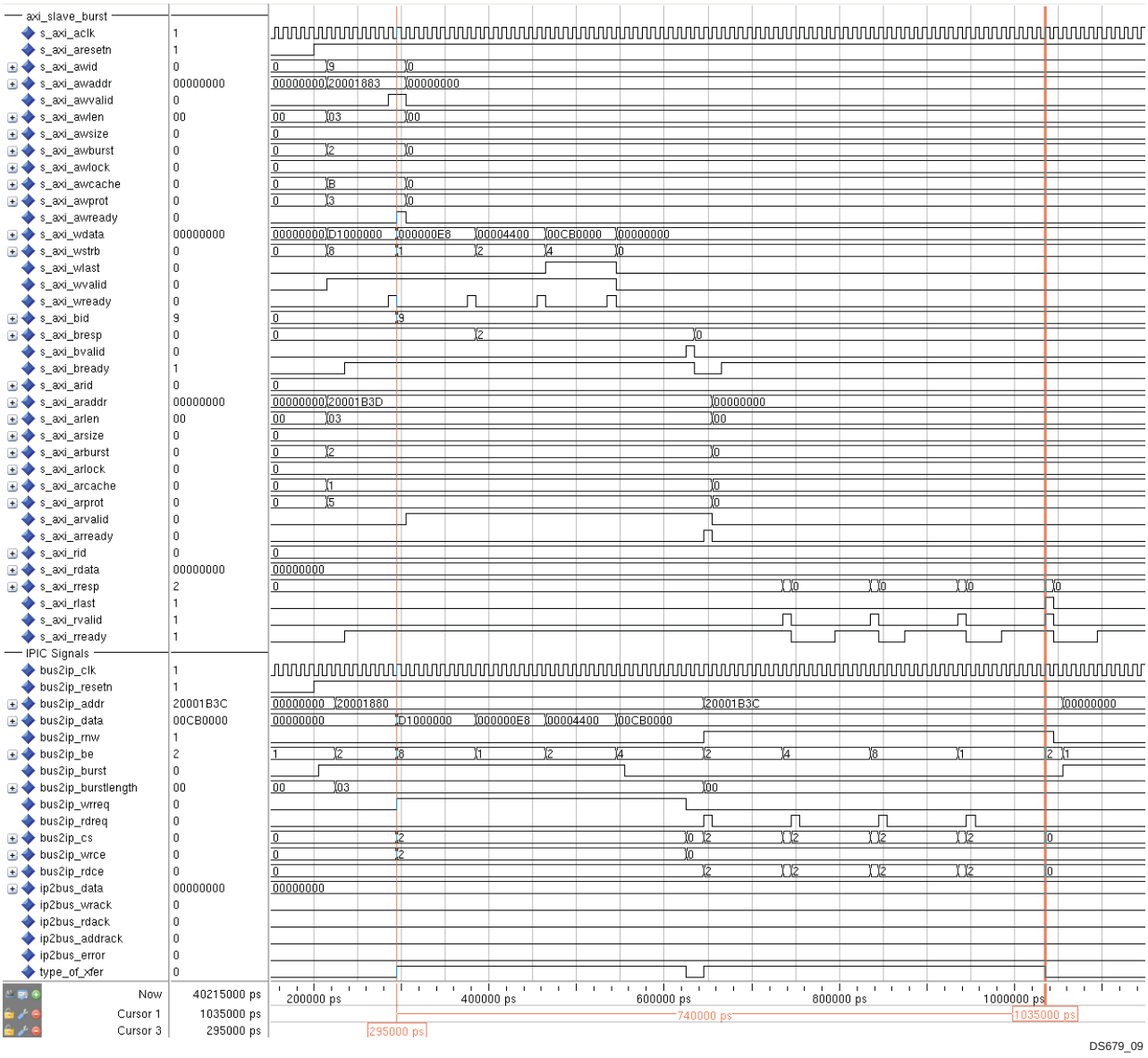


Figure 9: **AXI Slave Burst Response For AXI4 WRAP Mode Transactions When Slave IP Does Not Respond the Address Available on Bus2IP_Addr (FIFO Not Included)**

Figure 10 shows core response for slave generating data time out conditions for INCR read-write transactions from AXI4.The transactions are completed with SLVERR response.

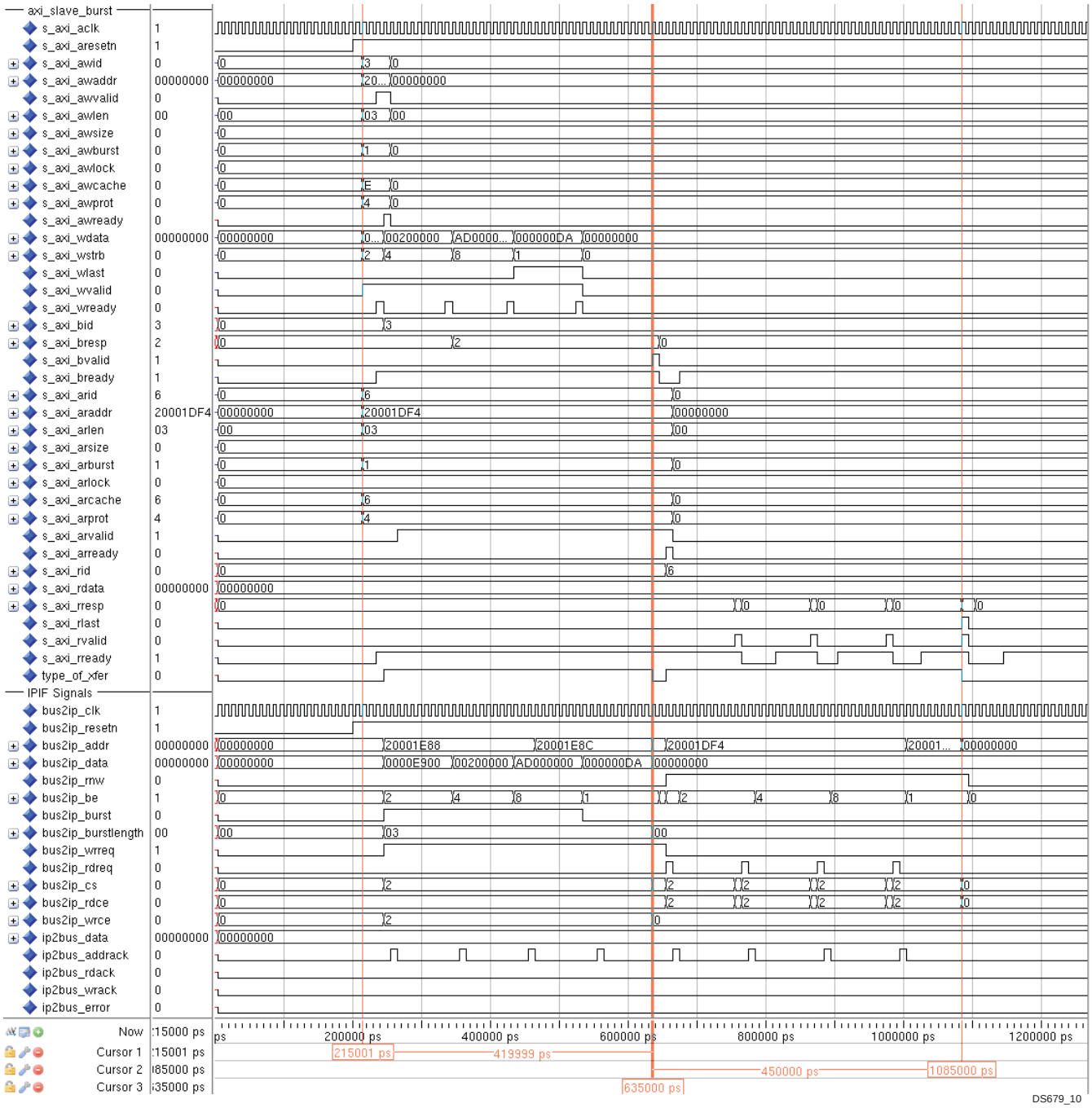


Figure 10: **AXI Slave Burst Response For AXI4 INCR Mode Transactions When Slave IP Does Not Respond the Read and Write Data Phase (FIFO Not Included)**

Figure 11 shows core response for slave generating data time out conditions for FIXED read-write transactions from AXI4.The transactions are completed with SLVERR response.

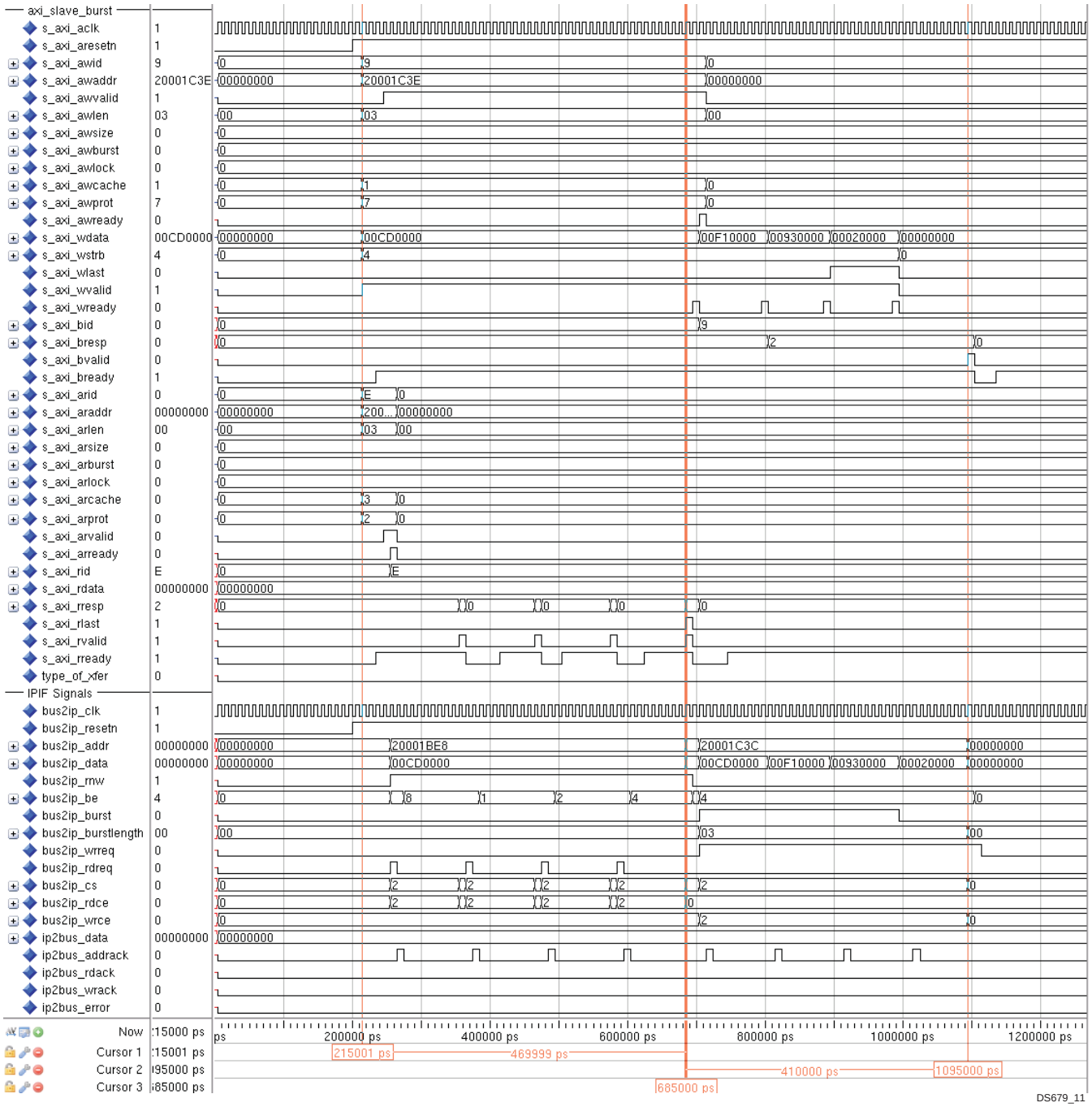


Figure 11: **AXI Slave Burst Response For AXI4 FIXED Mode Transactions When Slave IP Does Not Respond the Read and Write Data Phase (FIFO Not Included)**

Figure 12shows core response for slave generating data time out conditions for WRAP read-write transactions from AXI4.The transactions are completed with SLVERR response.

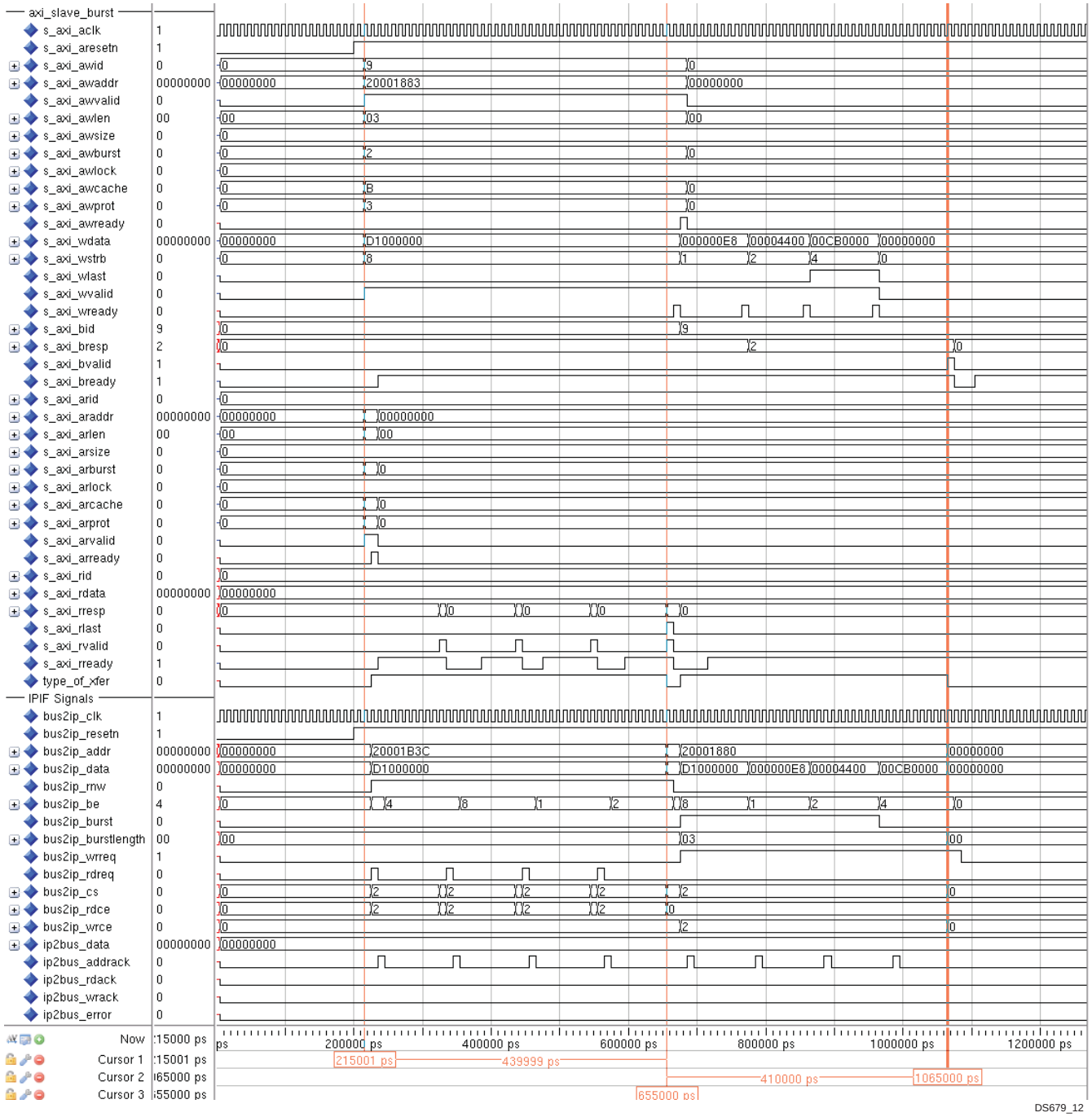


Figure 12: **AXI Slave Burst Response For AXI4 WRAP Mode Transactions When Slave IP Does Not Respond the Read and Write Data Phase (FIFO Not Included)**

- C_RDATA_FIFO_DEPTH= 32

Figure 13 shows AXI Slave Burst IP core response for INCR read-write transactions from AXI4.

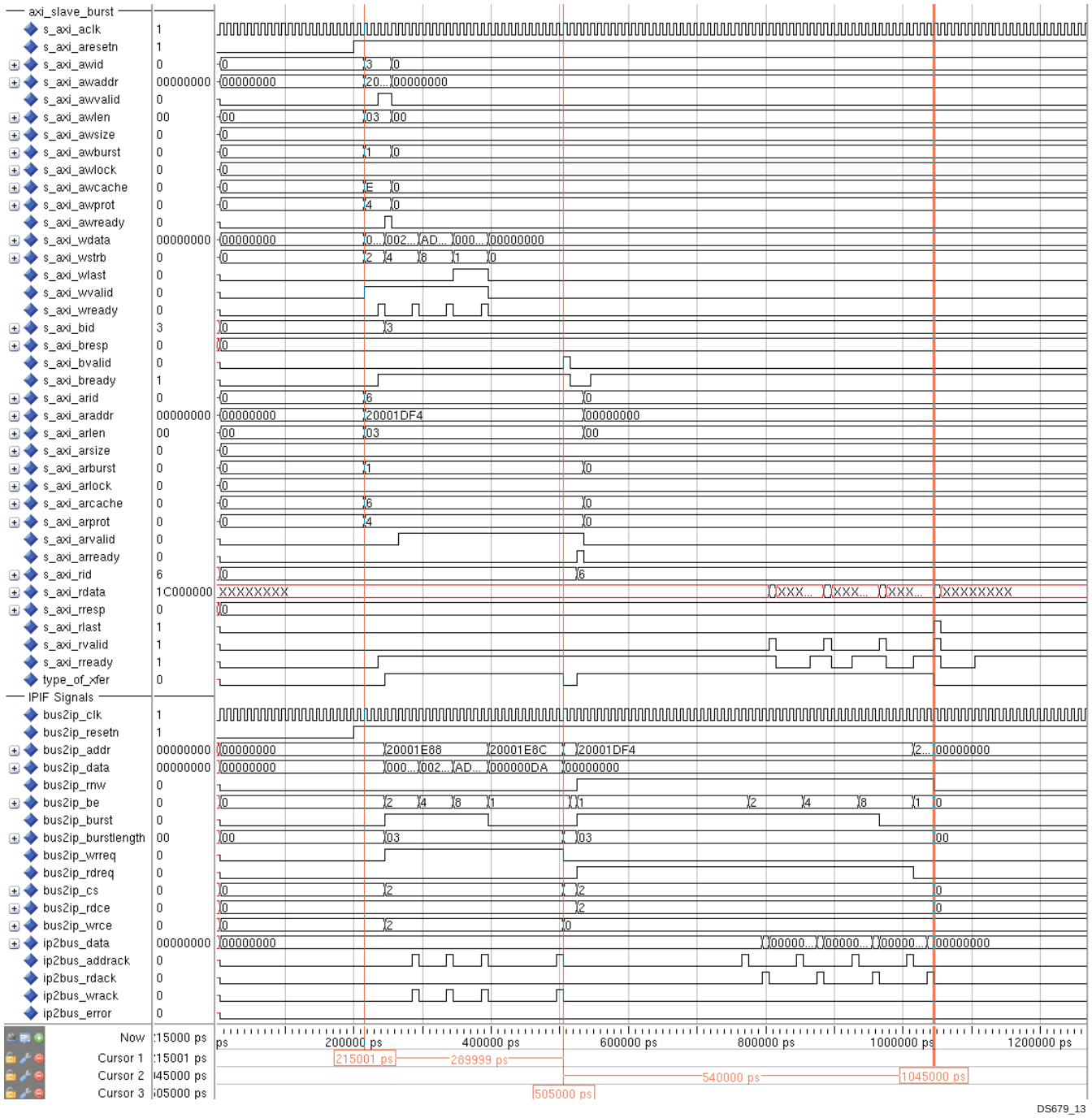


Figure 13: **AXI Slave Burst Response For AXI4 INCR Read-Write Transactions (FIFO is Included)**

Figure 14 shows AXI Slave Burst IP core response for FIXED read-write transactions from AXI4.

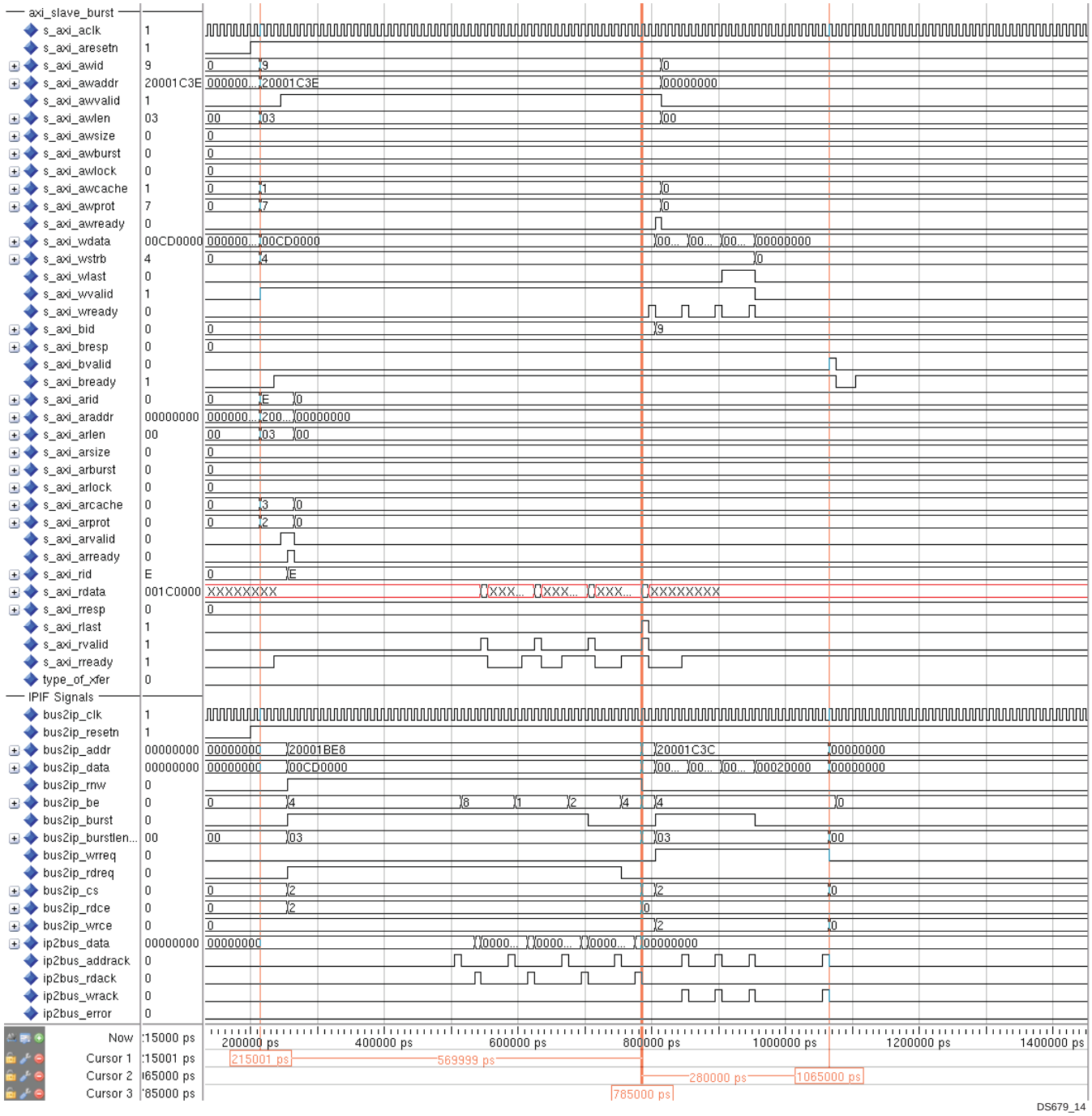


Figure 14: **AXI Slave Burst Response For AXI4 FIXED Read-Write Transactions (FIFO is Included)**

Figure 15 shows AXI Slave Burst IP core response for WRAP read-write transactions from AXI4.

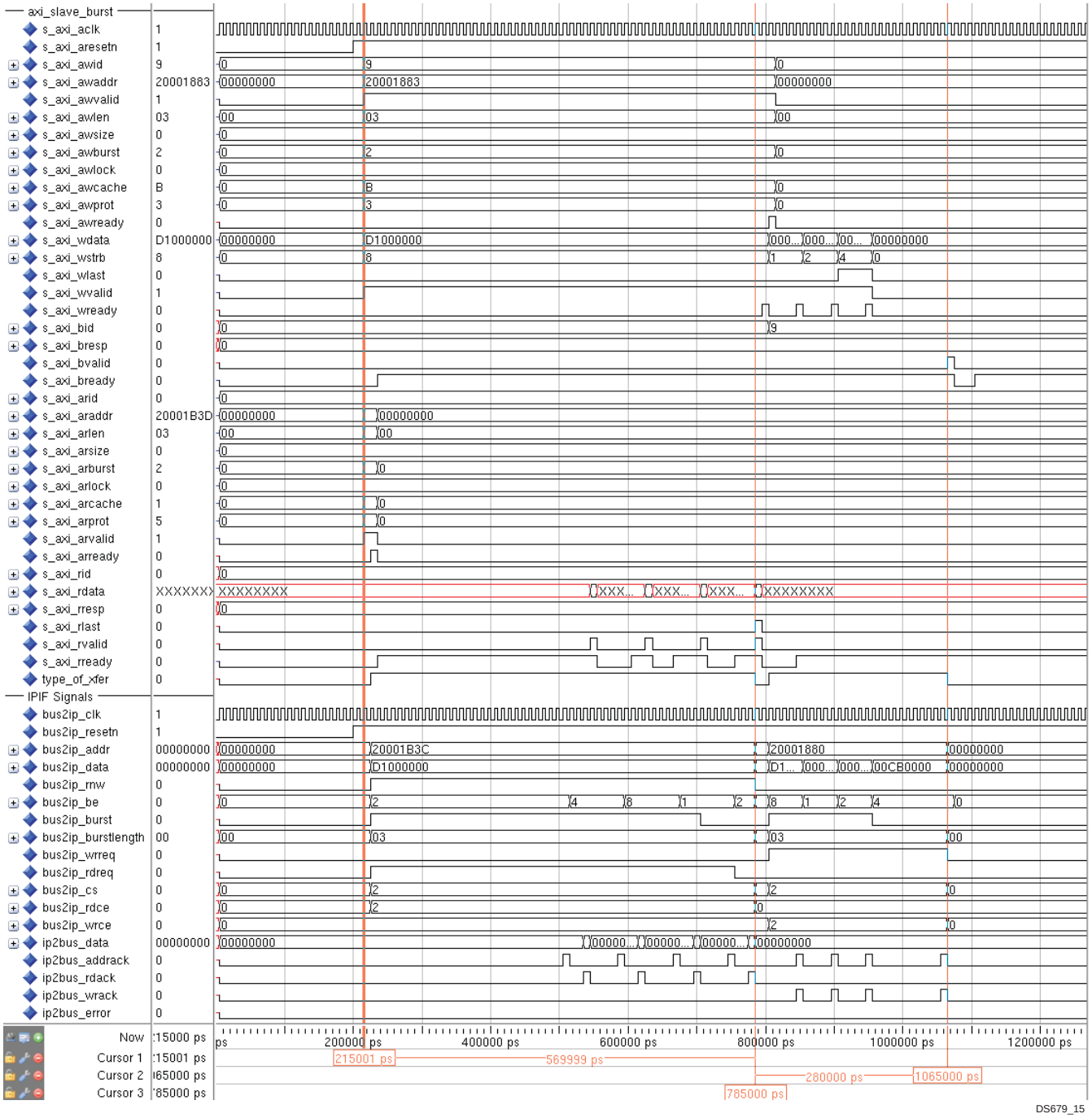


Figure 15: **AXI Slave Burst Response For AXI4 FIXED Read-Write Transactions (FIFO is Included)**

Figure 16 shows core response for slave generating write response when AWVALID is generated ahead of WVALID from AXI4.

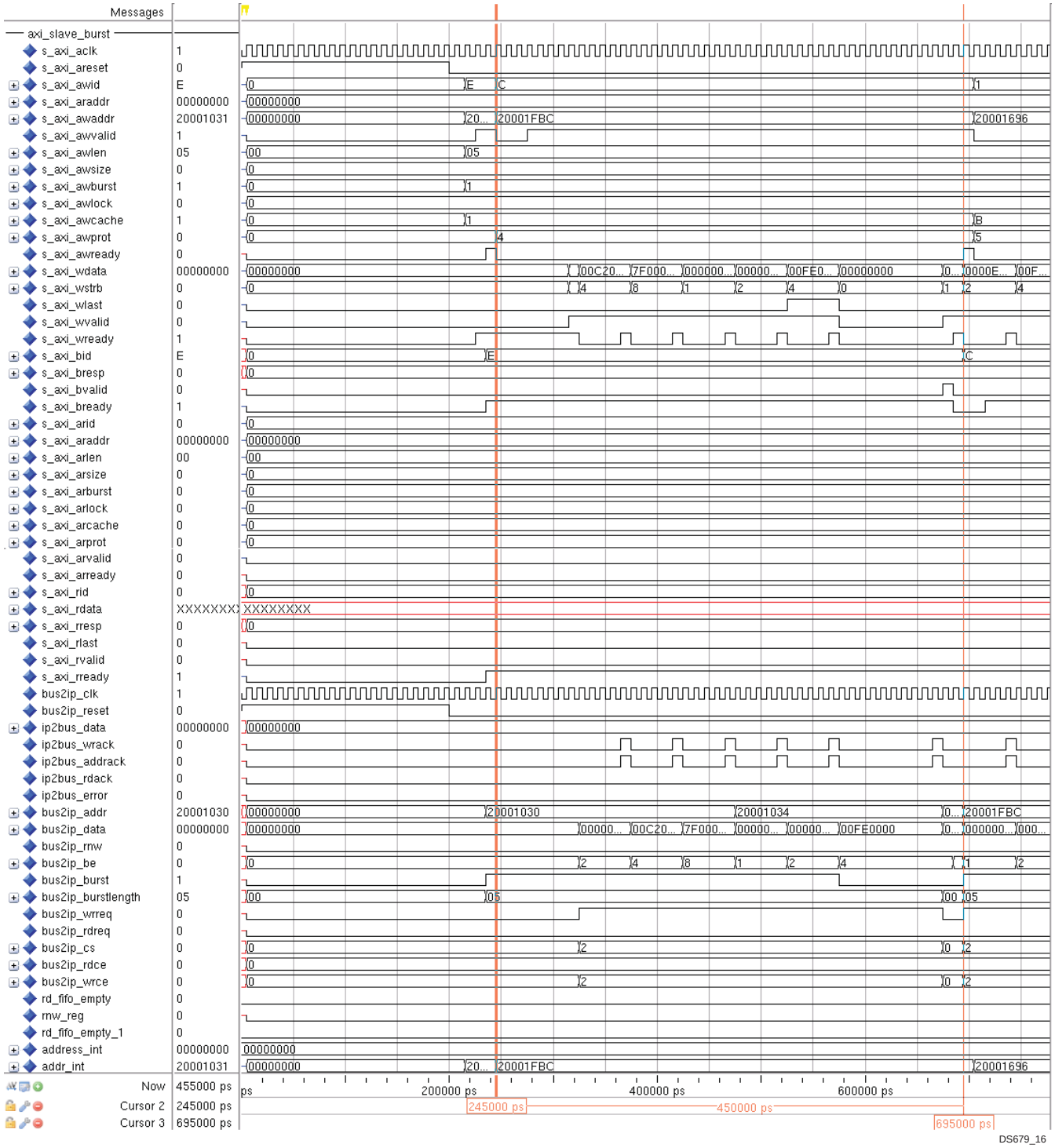
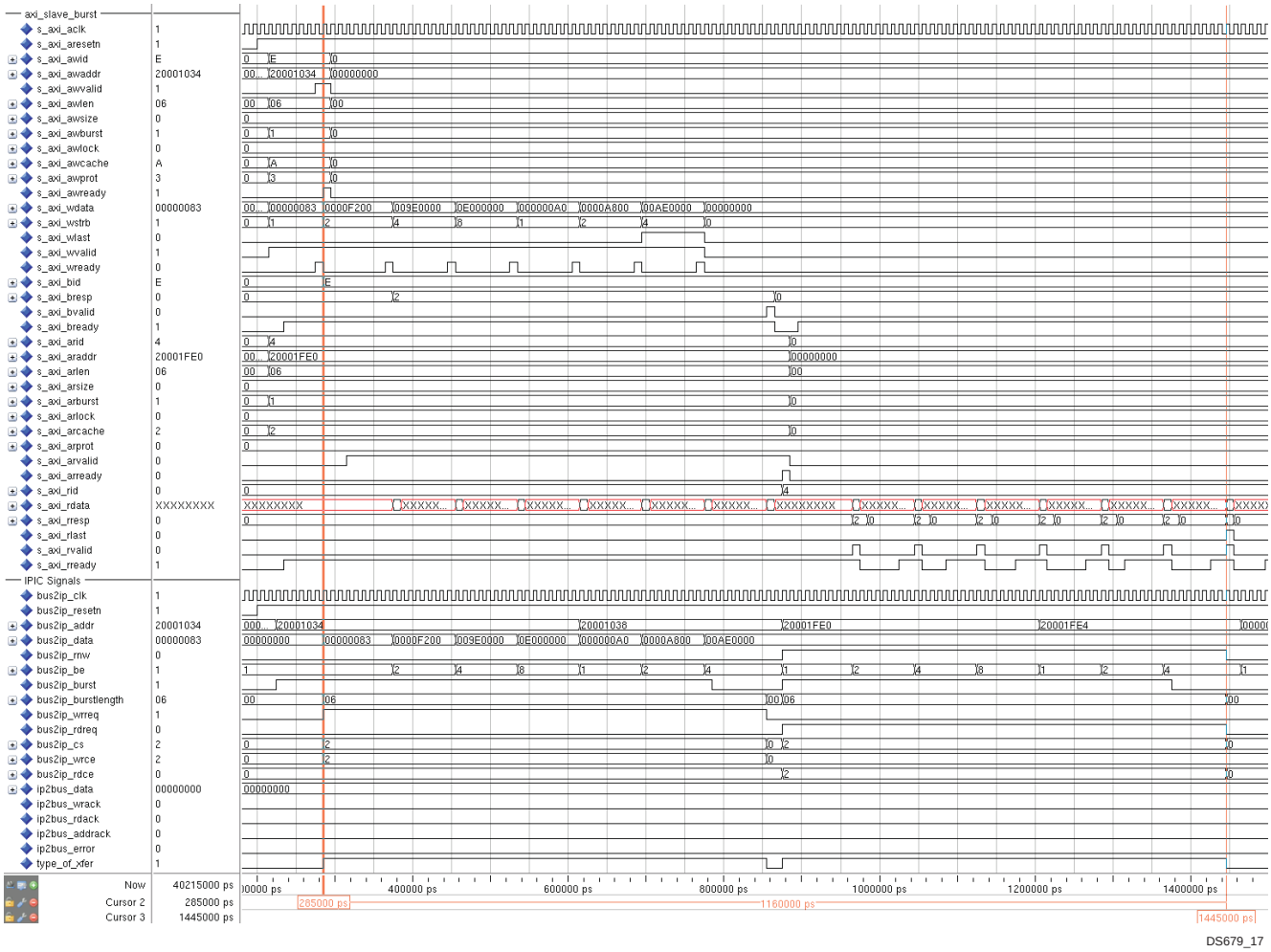


Figure 16: **AXI Slave Burst Response For AXI4 WRAP Read-Write Transactions (FIFO is Included) AWVALID and WVALID at a Different Time.**

Figure 17 shows core response for slave generating address time out conditions for INCR read-write transactions from AXI4. The transactions are completed with SLVERR response.



DS679_17

Figure 17: **AXI Slave Burst Response For AXI4 INCR Mode Transactions When Slave IP Does Not Respond the Address Available on Bus2IP_Addr (FIFO is Included)**

Figure 18 shows core response for slave generating address time out conditions for FIXED read-write transactions from AXI4. The transactions are completed with SLVERR response.

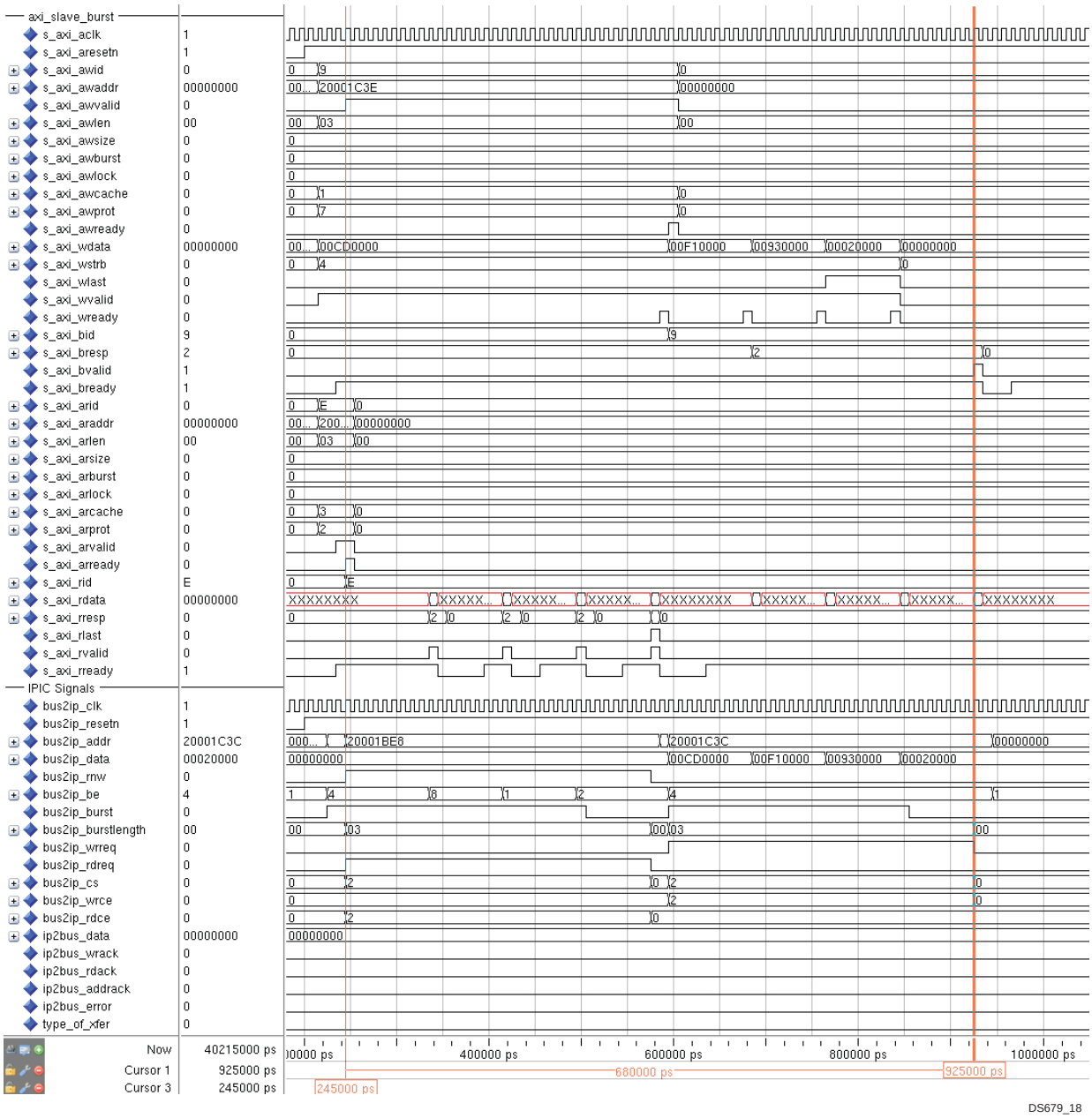


Figure 18: **AXI Slave Burst Response For AXI4 FIXED Mode Transactions When Slave IP Does Not Respond the Address Available on Bus2IP_Addr (FIFO is Included)**

Figure 19 shows core response for slave generating address time out conditions for WRAP read-write transactions from AXI4. The transactions are completed with SLVERR response.

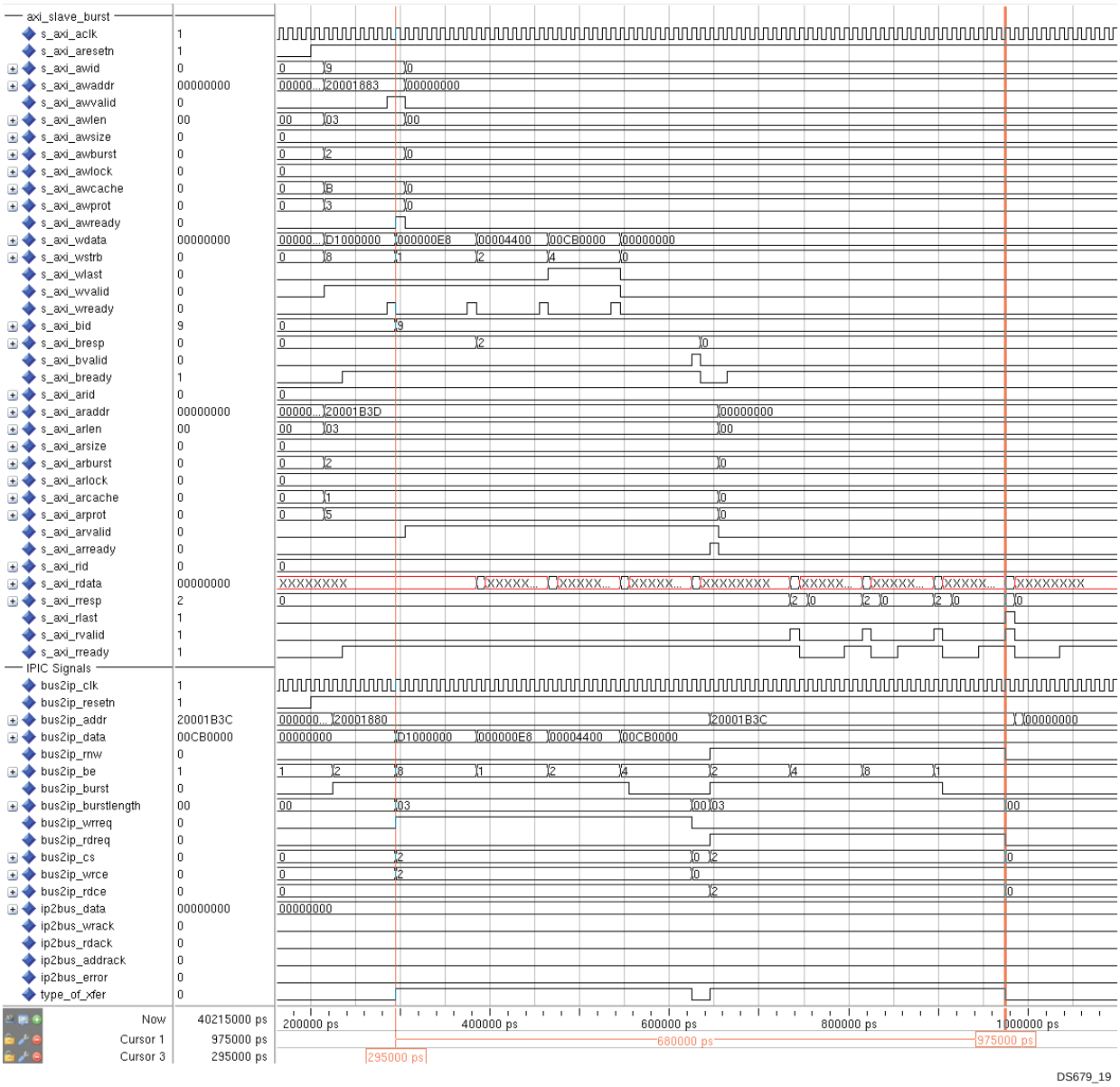


Figure 19: **AXI Slave Burst Response For AXI4 WRAP Mode Transactions When Slave IP Does Not Respond the Address Available on Bus2IP_Addr (FIFO is Included)**

Figure 20 shows core response for slave generating data time out conditions for INCR read-write transactions from AXI4. The transactions are completed with SLVERR response.

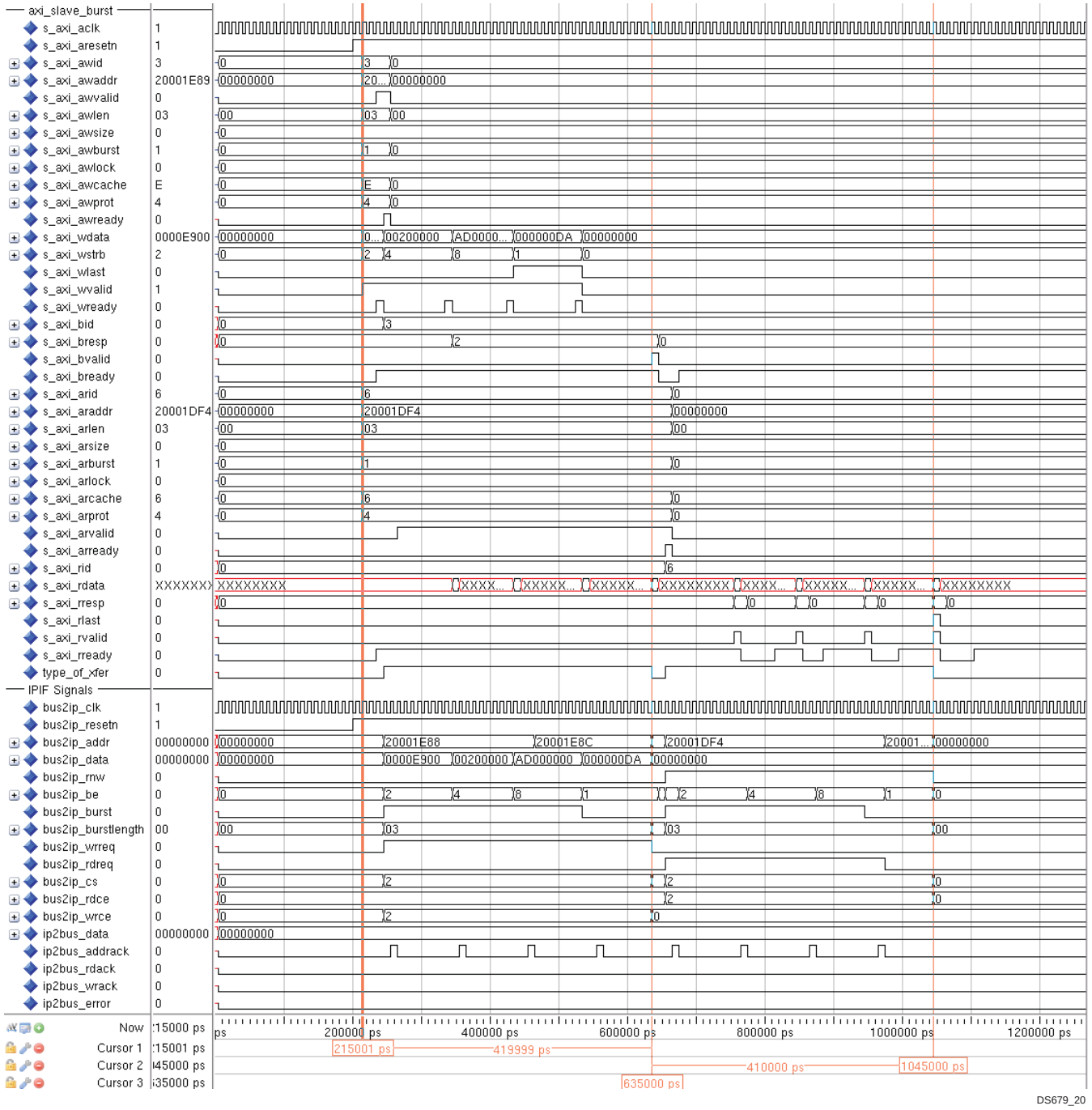


Figure 20: **AXI Slave Burst Response For AXI4 INCR Mode Transactions When Slave IP Does Not Respond the Read and Write Data Phase (FIFO is Included)**

Figure 21 shows core response for slave generating data time out conditions for FIXED read-write transactions from AXI4.The transactions are completed with SLVERR response.

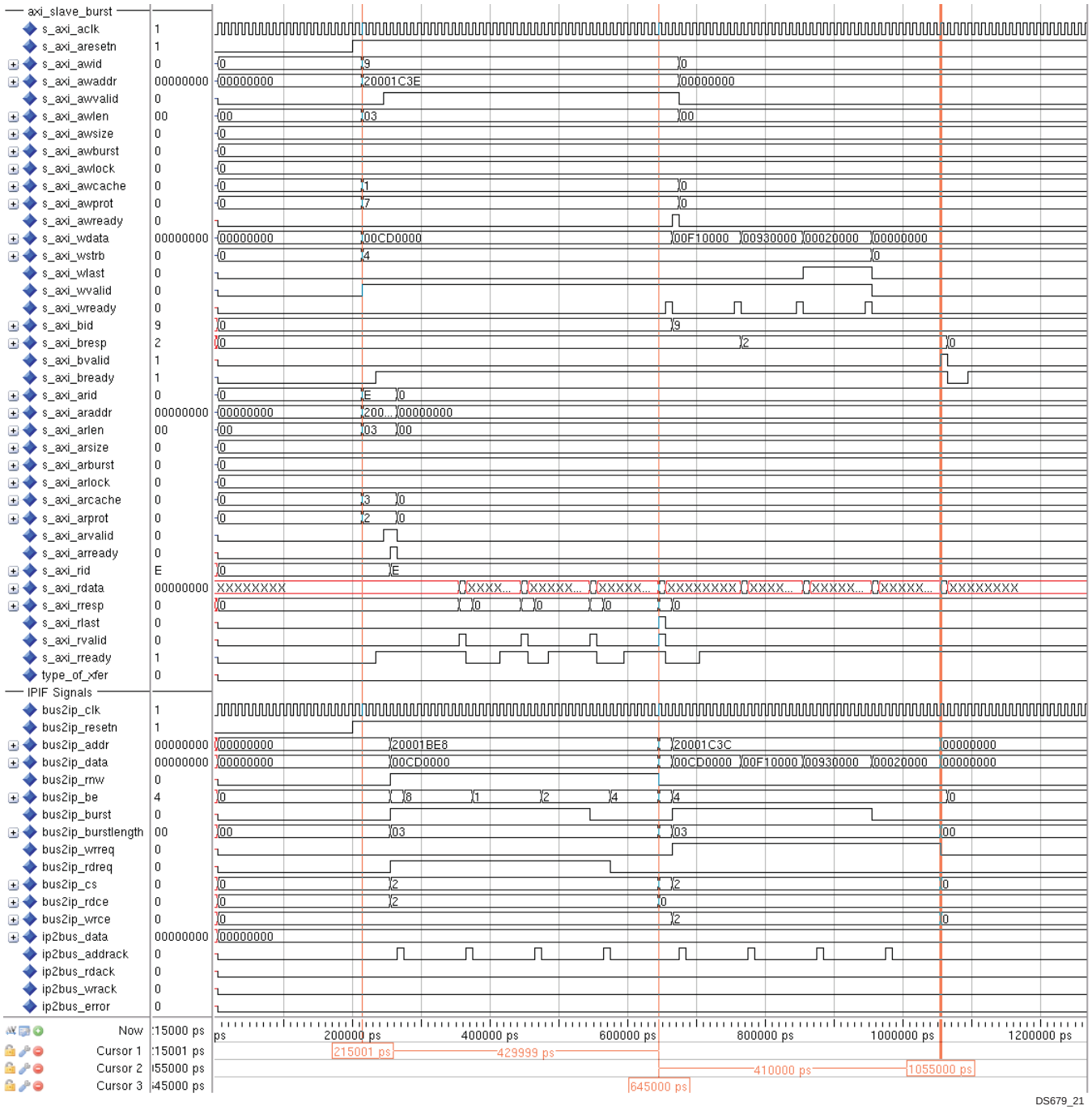
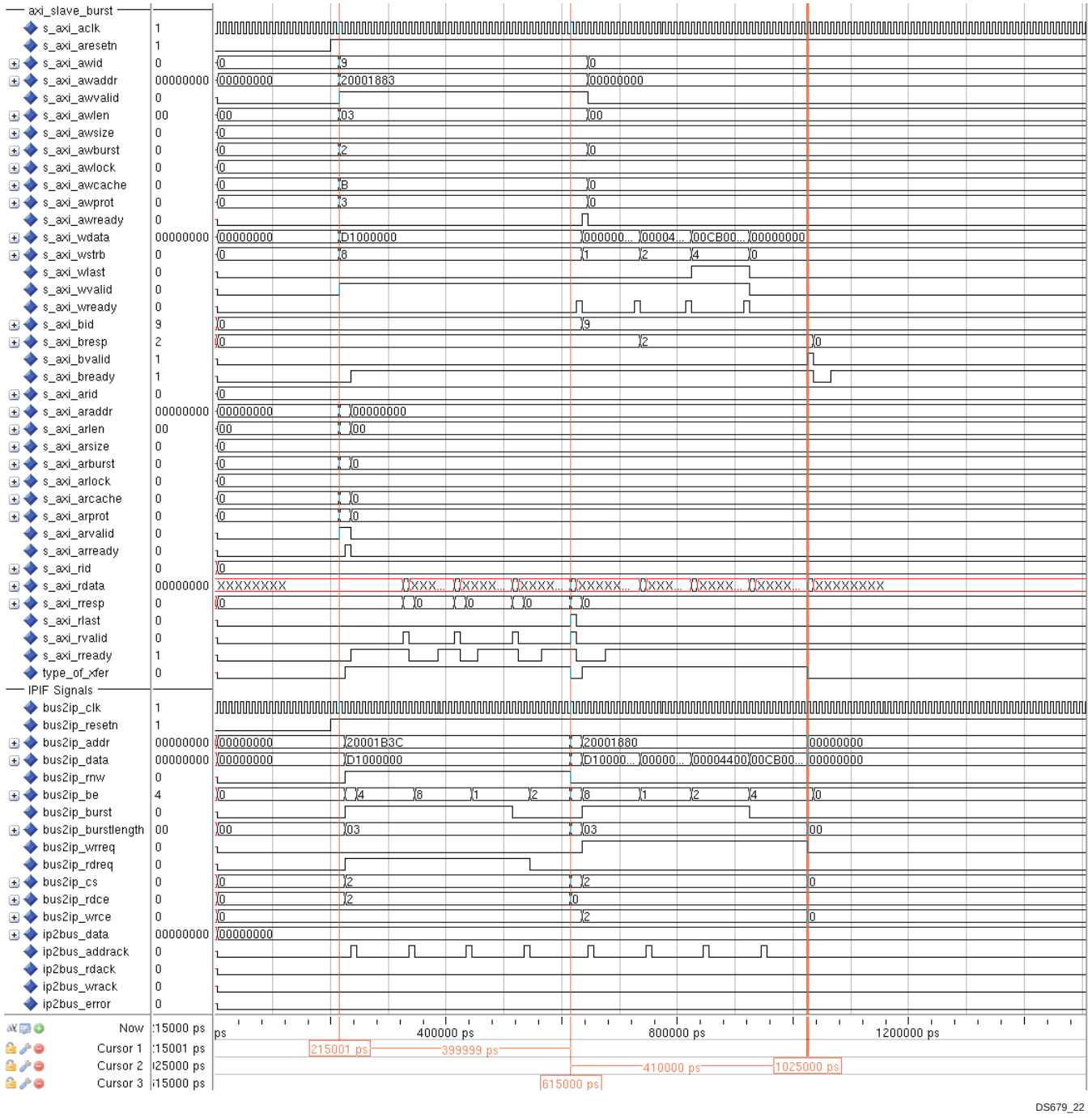


Figure 21: **AXI Slave Burst Response For AXI4 FIXED Mode Transactions When Slave IP Does Not Respond the Read and Write Data Phase (FIFO is Included)**

Figure 22 shows core response for slave generating data time out conditions for WRAP read-write transactions from AXI4. The transactions are completed with SLVERR response.



DS679_22

Figure 22: **AXI Slave Burst Response For AXI4 WRAP Mode Transactions When Slave IP Does Not Respond the Read and Write Data Phase (FIFO is Included)**

Device Utilization and Performance Benchmarks

Core Performance

Because the AXI Slave Burst is a module that will be used with other design pieces in the FPGA, the resource utilization and timing numbers reported in this section are estimates only. When the core is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing of the design will vary from the results reported here.

The axi_slave_burst resource utilization benchmarks for the Virtex®-7 FPGA on randomly generated cases is shown in Table 13.

Table 11: **FPGA Resource Utilization Benchmarks on the Virtex-7 FPGA (XC7V285T-FFG784-3)**

Parameter Value							Device Resources			Frequency
C_S_AXI_DATA_WIDTH	C_RDATA_FIFO_DEPTH	C_INCLUDE_TIMEOUT_CNT	C_TIMEOUT_CNTR_VAL	C_ALIGN_BE_RDADDR	C_S0_AXI_SUPPORTS_WRITE	C_S0_AXI_SUPPORTS_READ	Occupied Slices	Slice Registers	Slice LUTs	Fmax (MHz)
32	0	0	8	0	1	1	118	84	226	200
32	0	0	8	1	1	1	141	82	246	200
32	32	0	8	0	1	1	174	104	320	200
32	32	1	8	0	1	1	173	104	335	200
32	32	0	8	1	0	1	65	65	138	200
32	32	1	8	0	0	1	72	66	135	200
32	32	0	8	0	1	0	123	87	238	200
32	32	1	8	0	1	0	122	87	238	200
64	32	1	8	0	1	1	73	60	146	200
64	32	1	8	1	0	1	75	59	148	200

The axi_slave_burst resource utilization benchmarks for the Spartan®-6 FPGA on randomly generated cases is shown in Table 12.s

Table 12: **FPGA Resource Utilization Benchmarks on the Spartan-6 FPGA (XC6SLX150T-FGG900-3)**

Parameter Value							Device Resources			Frequency
C_S_AXI_DATA_WIDTH	C_RDATA_FIFO_DEPTH	C_INCLUDE_TIMEOUT_CNT	C_TIMEOUT_CNTR_VAL	C_ALIGN_BE_RDADDR	C_S0_AXI_SUPPORTS_WRITE	C_S0_AXI_SUPPORTS_READ	Occupied Slices	Slice Registers	Slice LUTs	Fmax (MHz)
32	0	0	8	1	1	1	115	82	229	130
32	0	1	8	0	1	1	122	105	251	128
32	32	0	8	1	1	1	170	106	330	117
32	32	1	8	0	1	1	167	120	348	128
32	32	0	8	1	0	1	110	88	224	138
32	32	1	8	0	0	1	111	88	224	137
32	32	0	8	1	1	0	67	61	145	139
32	32	1	8	0	1	0	83	61	150	129
64	32	0	8	0	1	1	123	173	291	132
64	32	1	8	1	0	1	127	86	252	140

The axi_slave_burst resource utilization benchmarks for the Virtex®-6 FPGA on randomly generated cases is shown in Table 13.

Table 13: **FPGA Resource Utilization Benchmarks on the Virtex-6 FPGA (XC6VLX195T-FF1156-1)**

Parameter Value							Device Resources			Frequency
C_S_AXI_DATA_WIDTH	C_RDATA_FIFO_DEPTH	C_INCLUDE_TIMEOUT_CNT	C_TIMEOUT_CNTR_VAL	C_ALIGN_BE_RDADDR	C_S0_AXI_SUPPORTS_WRITE	C_S0_AXI_SUPPORTS_READ	Occupied Slices	Slice Registers	Slice LUTs	Fmax (MHz)
32	0	0	8	0	1	1	101	79	232	202
32	0	0	8	1	1	1	109	103	252	226
32	32	0	8	0	1	1	140	96	326	201
32	32	1	8	0	1	1	161	127	379	202
32	32	0	8	1	0	1	122	87	224	200
32	32	1	8	0	0	1	127	109	278	226
32	32	0	8	0	1	0	85	93	173	225
32	32	1	8	0	1	0	92	83	222	200
64	32	1	8	0	1	1	66	66	140	243
64	32	1	8	1	0	1	156	108	310	226

Reference Documents

The following documents contain reference information important to understanding the AXI Slave Burst design:

1. *AMBA® AXI Protocol Version: 2.0 Specification* (ARM IHI 0022C)
2. DS768, *AXI Interconnect IP Data Sheet*
3. DS160, *Spartan-6 Family Overview*
4. DS150, *Virtex-6 Family Overview*
5. DS180, *7 Series FPGAs Overview*

Support

Xilinx provides technical support for this LogiCORE 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*.

Ordering Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx ISE Design Suite Embedded Edition software under the terms of the Xilinx End User License. The core is generated using the Xilinx ISE Embedded Edition software (EDK).

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE modules and software, please contact your local Xilinx sales representative.

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
9/21/10	1.0	Xilinx Initial Release.
6/22/11	2.0	Updated for Xilinx tools v13.2. Added support for Artix-7, Virtex-7, and Kintex-7 devices.

Notice of Disclaimer