PCI Express Bus Functional Model 3

Reference Manual

Version 1.6.4 December 2014

PCIe BFM 3

Reference Manual

Documentation Change History

Date	Version Number	Changes
December 2014	1.6.4	• Added XBFM_LOG_NORAW. • Updated PIPE configurations table.
September 2014	1.6.2	• Updated xbfm_set_requesterid, xbfm_custom_tlp, and xbfm_custom_tlp_id tasks.
April 2014	1.6.0	• No change to BFM Reference Manual.
January 2014	1.5.9	• Updated PIPE interface configuration information. • Added xbfm_set_l2, xbfm_set_pme, xbfm_register_write, and xbfm_register_read tasks. • Removed support for VHDL.
November 2013	1.5.7	• Added xbfm_memory_compare_count task.
April 2013	1.5.2	• Updated descriptions of PIPE_MIXED and xbfm_set_aspm.
August 2012	1.4.0	• Added xbfm_set_maxcredit task.
April 2012	1.4.0	• Updated description for xbfm_set_l2 and xbfm_set_pme.
January 2012	1.3.0	• Added support for PIE-8 interface. • Added task 'xbfm_set_cplsize'.
April 2011	1.0	First Release

Proprietary Notice

Table of Contents

List of Tables

List of Figures

Preface

About this Document

This document has been written for design managers, system engineers, and designers of ASICs and FPGAs who are evaluating or using PLDA PCI Express products.

Additional Reading

This section lists additional resources from PLDA and third-parties.

PLDA periodically updates its documentation. Please contact PLDA Technical Support or check the Web site at http://www.plda.com for current versions.

PLDA Publications

This reference manual describes the PLDA BFM that is compatible with PCI Express-based IP core. Please refer to the Reference Manual and Getting Started guides of your particular IP core for more details.

Other Publications

Please refer to the following documents for information on specification standards:

- *PCI Express® Base Specification Revision 4.0 Version 0.3*
- *PCI Express Base Specification Revision 3.0*
- *PHY Interface for the PCI Express, SATA and USB 3.1 Architectures, Version 4.2*
- *PHY Interface Extensions Supporting 8GT/s PCIe PIE-8 Revision 2.02*

Feedback and Contact Information

Feedback about this Document

PLDA welcomes comments and suggestions about this documentation. Please contact PLDA Technical Support and provide the following information:

- the title of the document
- the page number to which your comments refer
- a description of your comments

Contact information

Corporate Headquarters

PLDA
805, avenue J.R.G.G. de la Lauzière
13290 Aix-en-Provence
France

Tel: USA +1 408 273 4528 - International +33 442 393 600
Fax: +33 442 394 902

Sales

For sales questions, please contact sales@plda.com.

Technical Support

For technical support questions, please contact PLDA Support at http://www.plda.com/plda_login.php using the Support Center if you have a PLDA online account.

If you don’t have a PLDA account, contact http://www.plda.com/support_enquiry.php.

Chapter 1 Introduction

PLDA's PCI Express Bus Functional Model (BFM) allows you to test your PCI Express 3.0 design. The BFM, which emulates a PCIe environment, is connected to a Design Under Test (DUT) and supports:

- sending packets to the DUT
- receiving packets from the DUT
- monitoring bus activity
- reporting errors

1.1 The BFM as Rootport

In order to test an Endpoint or Switch upstream port design, the BMF must act as a Rootport and support:

- link training
- initialization of the DUT
- send/receive transactions to/from the DUT

The figure below illustrates a system architecture where the DUT is an Endpoint or Switch upstream port and the BFM is treated as a Rootport:

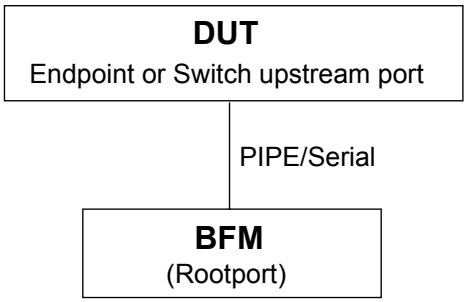

Figure 1: Configuring the BFM as a Rootport

1.2 The BFM as Endpoint

In order to test a Rootport or Switch downstream port design, the BFM must act as an Endpoint and be able to send and receive transactions to/from the DUT.

The figure below illustrates a system architecture where the DUT is a Rootport or Switch downstream port and the BFM is treated as an Endpoint:

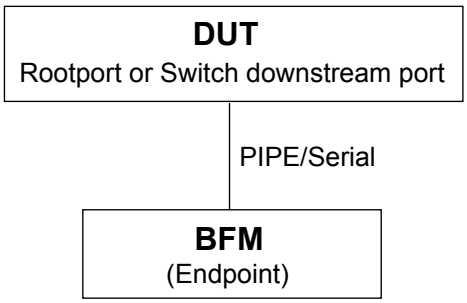

Figure 2: Configuring the BFM as an Endpoint

Chapter 2 Exploring the BFM Architecture

2.1 Introduction

PLDA's BFM includes two functional modules, as illustrated below:

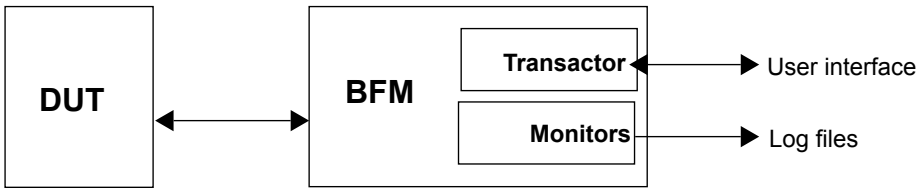

Figure 3: BFM system architecture

2.2 Transactor

The Transactor is the heart of the BFM, responsible for sending and receiving transactions to and from the DUT. It is also responsible for:

- Implementing a memory space with a 64-bit address, a memory space with a 32-bit address, and an I/O space (each space is configurable from 4KB to 16MB)
- performing link training
- detecting events
- sending packets and performing customer-designed actions

2.3 Monitors

Monitor modules survey PCIe bus activity in both directions and log all activity in a readable format. Information is logged at the PIPE, raw data and packet level.

2.3.1 PIPE Monitor

10-bit codes are logged and special codes are replaced with their mnemonics. This log also reports LTSSM state transitions in order to help troubleshoot link training and link instability issues.

The following figure illustrates an extract from the PIPE log:

```
-------------+-----------------------+-----------------------+
             |          TX           |          RX           |
     Time    |  0  |  1  |  2  |  3  |  0  |  1  |  2  |  3  |
-------------+-----------------------+-----------------------+

     6622 ns : LTSSM transition: Detect.Active -> Polling.Active
     6650 ns | COM-| COM-| COM-| COM-| --- | --- | --- | --- |
     6654 ns | PAD+| PAD+| PAD+| PAD+| --- | --- | --- | --- |
     6658 ns | PAD+| PAD+| PAD+| PAD+| --- | --- | --- | --- |
     6662 ns | 06+ | 06+ | 06+ | 06+ | --- | --- | --- | --- |
     6666 ns | 02- | 02- | 02- | 02- | --- | --- | --- | --- |
```

Figure 4: Sample of the PIPE log file

2.3.2 Packet Monitor

At the packet level, data is deskewed, descrambled, and TLP/DLLP packets are decoded. This log also reports LTSSM state transitions in order to help troubleshoot link training and link instability issues.

The figure below illustrates an extract from the packet log:

```
------------+-+---+--------+-+---+--+----+----------------+----------------+-+-+----+------+---+
            | |   |        |T|DC |HC|    |    BYTECOUNT    |      FLAGS     | | |    | MSG  |MSG|
            |D|   |        |C| / | /|    | & LOWER ADDRESS |       /        |L|F|    | CODE |RTG|
    TIME    |I|SEQ| PACKET |V|LEN|TA| REQ|        /        |      DATA      |B|B|CMPL|  /   | / |
            |R|NUM|  TYPE  |C|GTH| G| ID |     ADDRESS     |   MSB      LSB |E|E| ID |STATUS|AT |
------------+-+---+--------+-+---+--+----+----------------+----------------+-+-+----+------+---+
    35848 ns T 00e ACK
    36056 ns T --- UFC_P    0 084 24
    36152 ns T --- UFC_NP   0 016 1b
    36456 ns T 00a CPLD     0 001 0a 0000      004       04 -------- ---RO--- - - 0100 SC     00
                                                            89abcdef
    36728 ns R 00a ACK
    36808 ns R 00f MWR      0 020 9f 0000 22222222 22220000 -------- ---RO--- f f ---- ------ 00
                                                          07060504 03020100
                                                          0f0e0d0c 0b0a0908
                                                          17161514 13121110
                                                          1f1e1d1c 1b1a1918
                                                          27262524 23222120
                                                          2f2e2d2c 2b2a2928
                                                          37363534 33323130
                                                          3f3e3d3c 3b3a3938
                                                          47464544 43424140
                                                          4f4e4d4c 4b4a4948
                                                          57565554 53525150
                                                          5f5e5d5c 5b5a5958
                                                          67666564 63626160
                                                          6f6e6d6c 6b6a6968
                                                          77767574 73727170
                                                          7f7e7d7c 7b7a7978
```

Figure 5: Sample of the Packet log file

2.3.3 RAW Monitor

The RAW Monitor reports raw symbols received or transmitted by the BFM. It is very similar to the PIPE log, but data is displayed unscrambled. Some ordered-sets do not appear in this output.

```
------------+----------------------+----------------------+
            |          TX          |          RX          |
     Time   |  0  |  1  |  2  |  3  |  0  |  1  |  2  |  3  |
------------+----------------------+----------------------+
    26056 ns | 45  | 45  | 45  | 45  | COM | COM | COM | COM |
             | 45  | 45  | 45  | 45  | 00  | 00  | 00  | 00  |
             | 45  | 45  | 45  | 45  | 00  | 01  | 02  | 03  |
             | 45  | 45  | 45  | 45  | ff  | ff  | ff  | ff  |
    26072 ns | COM | COM | COM | COM | 02  | 02  | 02  | 02  |
             | SKP | SKP | SKP | SKP | 00  | 00  | 00  | 00  |
             | SKP | SKP | SKP | SKP | 45  | 45  | 45  | 45  |
             | SKP | SKP | SKP | SKP | 45  | 45  | 45  | 45  |
    26088 ns | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
             | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
             | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
             | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
    26104 ns | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
             | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
             | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
             | 00  | 00  | 00  | 00  | 45  | 45  | 45  | 45  |
```

Figure 6: Sample of the RAW log file

Chapter 3 Wrapper Implementations

PLDA provides two wrappers for the BFM:

- Serial wrapper
- PIPE wrapper

3.1 Serial Wrapper

The serial wrapper binds the BFM with logic in order to provide clock, reset, and a serial 1-bit interface.

The following table describes the serial wrapper interface:

Table 1: Serial wrapper interface

Name	Type	Description
BFM_TYPE	Integer parameter	Indicates BFM type ('0'=RP,'1'=EP)
NUM_LANES	Integer parameter	Indicates number of connected lanes (1,2, 4, or 8)
IO_SIZE	Integer parameter	Specifies the size of internal IO space. Possible values range from 12 (2^{12}=4KB) to 24 (2^{24}=16MB)
MEM32_SIZE	Integer parameter	Specifies the size of internal 32-bit addressing memory space. Possible values range from 12 (2^{12}=4KB) to 24 (2^{24}=16MB)
MEM64_SIZE	Integer parameter	Specifies the size of internal 64-bit addressing memory space. Possible values range from 12 (2^{12}=4KB) to 24 (2^{24}=16MB)
RAW_LOG_NAME	String parameter	Specifies the name of the RAW log file.
PACKET_LOG_NAME	String parameter	Specifies the name of the packet log file.
pclk	Input	62.5 MHz clock for 2.5 Gbps mode. This clock must be switched to 125 MHz when DUT enters 5.0 Gbps mode, and 250 MHz in 8.0 Gbps mode.
resetn	Input	Active-low reset
tx_rate	Output[1:0]	Indicates transmitted data rate: • 00 = 2.5 Gbps • 01 = 5.0 Gbps • 10 = 8.0 Gbps • 11 = 16.0 Gbps This signal must be used to control the frequency of pclk.
tx_1b	Output [NUM_LANES-1:0]	TX lanes
rx_1b	Input [NUM_LANES-1:0]	RX lanes

3.2 PIPE Wrapper

The PIPE wrapper binds the BFM with logic in order to provide clock, reset, and a PIPE 32-bit interface.

The table below describes the PIPE wrapper interface:

Table 2: PIPE wrapper interface

Name	Type	Description
BFM_TYPE	Integer parameter	Indicates BFM type ('0'=RP,'1'=EP)
NUM_LANES	Integer parameter	Indicates number of connected lanes (1, 2, 4, 8, or 16)
IO_SIZE	Integer parameter	Specifies the size of internal IO space. Possible values range from 12 (2^{12}=4KB) to 24 (2^{24}=16MB)
MEM32_SIZE	Integer parameter	Specifies the size of internal 32-bit addressing memory space. Possible values range from 12 (2^{12}=4KB) to 24 (2^{24}=16MB)
MEM64_SIZE	Integer parameter	Specifies the size of internal 64-bit addressing memory space. Possible values range from 12 (2^{12}=4KB) to 24 (2^{24}=16MB)
PIPE_LOG_NAME	String parameter	Specifies the name of the PIPE log file.
PIPE_IF_W	Integer parameter	Maximum PIPE interface width: 2 or 4. See Table 3 for a list of possible PIPE configurations.
PIPE_IF_GEN3	Integer parameter	PIPE interface width at 8.0 Gbps speed: 2 or 4.
PIPE_IF_GEN2	Integer parameter	PIPE interface width at 5.0 Gbps speed: 1, 2 or 4.
PIPE_IF_GEN1	Integer parameter	PIPE interface width at 2.5 Gbps speed: 1, 2 or 4.
RAW_LOG_NAME	String parameter	Specifies the name of the RAW log file.
PACKET_LOG_NAME	String parameter	Specifies the name of the packet log file.
pclk	Input	PIPE clock. Its frequency depends on the signaling rate and PIPE interface width configuration. See Table 3 for details.
resetn	Input	Active-low reset
tx_rate	Output [1:0]	Indicates the data rate at which the BFM is transmitting (same encoding as for rate, below). This signal is used for diagnostic purposes only.
rate	Input[1:0]	PIPE interface signals common to all lanes. The rate signal indicates DUT data rate: • 00 = 2.5 Gbps • 01 = 5.0 Gbps • 10 = 8.0 Gbps • 11 = 16.0 Gbps
txdetectrx	Input	
phystatus	Output	
powerdown	Input	

Table 2: PIPE wrapper interface

Name	Type	Description
txelecidle	Input [NUM_LANES-1:0]	PIPE interface signals for each lane
txcompliance	Input [NUM_LANES-1:0]	
rxpolarity	Input [NUM_LANES-1:0]	
rxelecidle	Output [NUM_LANES-1:0]	
rxvalid	Output [NUM_LANES-1:0]	
txstartblock	Output [NUM_LANES-1:0]	
rxstartblock	Input [NUM_LANES-1:0]	
txdatavalid	Output [NUM_LANES-1:0]	
rxdatavalid	Input [NUM_LANES-1:0]	
txsyncheader	Output [2*NUM_LANES-1:0]	
rxsyncheader	Input [2*NUM_LANES-1:0]	
txdata	Input [8*PIPE_IF_W* NUM_LANES-1:0]	
txdatak	Input [PIPE_IF_W* NUM_LANES-1:0]	
rxdata	Output [8*PIPE_IF_W* NUM_LANES-1:0]	
rxdatak	Input [PIPE_IF_W* NUM_LANES-1:0]	
macdataen	Input [NUM_LANES-1:0]	MAC to PHY Transfer Progress Indicator
macdata	Input [6*NUM_LANES-1:0]	MAC to PHY Transfer Control/Data
phydataen	Output [NUM_LANES-1:0]	PHY to MAC Transfer Progress Indicator
phydata	Output [6*NUM_LANES-1:0]	PHY to MAC Transfer Control/Data
blockaligncontrol	Input	Block Alignment Control: Must be tied to 1 if not available on the controller.
rxstandby	Input [NUM_LANES-1:0]	PHY RX Control: Must be tied to 0's if not available on the controller.

The behavior of the `MACDATAEN`, `MACDATA`, `PHYDATAEN`, and `PHYDATA` signals is defined in the PIE-8 specification. If the DUT does not support 8.0 Gbps link speed, these signals must be tied to 0. If the DUT does support 8.0 Gbps, however, these signals must be properly controlled, otherwise link equalization will fail.

The following table shows the various PIPE configurations that are possible depending on the PIPE speed and link width:

Possible Link Widths	G_PIPE_IF_W (Max. or Gen4 PIPE width)	G_PIPE_IF_GEN3 (Gen3 PIPE config)	G_PIPE_IF_GEN2 (Gen2 PIPE config)	G_PIPE_IF_GEN1 (Gen1 PIPE config)
x1-x16	2 (16 bits 1 GHz)	2 (16 bits 500 MHz)	2 (16 bits 250 MHz)	2 (16 bits 125 MHz)
x1-x16	2 (16 bits 1 GHz)	2 (16 bits 500 MHz)	2 (16 bits 250 MHz)	1° (8 bits 250 MHz)
x1-x16	2 (16 bits 1 GHz)	2 (16 bits 500 MHz)	1° (8 bits 500 MHz)	1° (8 bits 250 MHz)
x1-x16	2 (16 bits 1 GHz)	2 (16 bits 500 MHz)	1° (8 bits 500 MHz)	0° (8 bits gapped[1] 500 MHz)
x1-x8	4 (32 bits 500 MHz)	4 (32 bits 250 MHz)	4 (32 bits 125 MHz)	4 (32 bits 62.5 MHz)
x1-x8	4 (32 bits 500 MHz)	4 (32 bits 250 MHz)	4 (32 bits 125 MHz)	2* (16 bits 125 MHz)
x1-x8	4 (32 bits 500 MHz)	4 (32 bits 250 MHz)	2* (16 bits 250 MHz)	2* (16 bits 125 MHz)
x1-x8	4 (32 bits 500 MHz)	4 (32 bits 250 MHz)	2* (16 bits 250 MHz)	1*° (8 bits 250 MHz)

Table 3: PIPE Configurations

*In these modes, TXDATA[31:16], RXDATA[31:16], TXDATAK[3:2], and RXDATAK[3:2]are not used.

°In these modes, TXDATA[15:8], RXDATA[15:8], TXDATAK[1], and RXDATAK[1]are not used.

[1]In this mode, `TXDATAVALID/RXDATAVALID` are used to enable one data byte every 2 clock cycles.

Chapter 4 XBFM Package Constants and Tasks

4.1 Constants

The following constants, all integers, are defined in the xbfm package:

Transaction types

- XBFM_CFGRD0: configuration read type 0
- XBFM_CFGRD1: configuration read type 1
- XBFM_IORD: IO read
- XBFM_MRD: memory read
- XBFM_MRDLK: memory read locked
- XBFM_CFGWR0: configuration write type 0
- XBFM_CFGWR1: configuration write type 1
- XBFM_IOWR: IO write
- XBFM_MWR: memory write
- XBFM_FADD: Fetch and Add atomic operation
- XBFM_SWAP: unconditional Swap atomic operation
- XBFM_CAS: Compare and Swap atomic operation

Completion status

- XBFM_SC: successful
- XBFM_CA: completer abort
- XBFM_UR: unsupported request
- XBFM_CR: configuration retry status
- XBFM_TIMEOUT: timeout occurred, completion was not received within 50 us

Events

- XBFM_REQ_RCVD: request received
- XBFM_CPLD_RCVD: completion with data is received
- XBFM_CPL_RCVD: completion without data received (any completion status)
- XBFM_CPLSC_RCVD: completion without data received with successful status
- XBFM_CPLUR_RCVD: completion without data received with unsupported request status
- XBFM_CPLCA_RCVD: completion without data received with completer abort status
- XBFM_INTAA_RCVD: "interrupt pin A asserted" message received
- XBFM_INTAD_RCVD: "interrupt pin A de-asserted" message received
- XBFM_INTBA_RCVD: "interrupt pin B asserted" message received
- XBFM_INTBD_RCVD: "interrupt pin B de-asserted" message received
- XBFM_INTCA_RCVD: "interrupt pin C asserted" message received
- XBFM_INTCD_RCVD: "interrupt pin C de-asserted" message received
- XBFM_INTDA_RCVD: "interrupt pin D asserted" message received
- XBFM_INTDD_RCVD: "interrupt pin D de-asserted" message received
- XBFM_PMPME_RCVD: "PME pin event" message received
- XBFM_IO_HIT: IO space received a request

- XBFM_MEM32_HIT: Request received for 32-bit address memory space
- XBFM_MEM64_HIT: Request received for 64-bit address memory space
- XBFM_L0_STATE: BFM LTSSM in L0 state
- XBFM_L0S_STATE: BFM LTSSM in L0s state
- XBFM_L1_STATE: BFM LTSSM in L1 state
- XBFM_L2_STATE: BFM LTSSM in L2 state

Spaces

- XBFM_IO: IO space
- XBFM_MEM32: Memory space with 32-bit address
- XBFM_MEM64: Memory space with 64-bit address

Log

- XBFM_LOG_CMD: log command issued
- XBFM_LOG_DATA: log transmitted / expected data
- XBFM_LOG_NOPIPE: disable PIPE logging
- XBFM_LOG_NORAW: disable RAW logging

ASPM

- XBFM_ASPM_L0S: allow active state power management L0s state
- XBFM_ASPM_L1: allow active state power management L1 state

Transaction Types

- XBFM_RW: read or write request
- XBFM_READ: read request only
- XBFM_WRITE: write request only

4.2 Tasks

The following section describes the Tasks in the XBFM Package.

4.2.1 Printing Tasks

4.2.1.1 xbfm_print

Prints a line to a log file.

Name	Type	Description
message	input string	Message string, up to 100 characters

Table 4: xbfm_print parameters

4.2.1.2 xbfm_print_comment

This Task prints a message to a log file. This is useful for commenting a simulation and making log files more readable.

Name	Type	Description
message	input string	Message string, up to 100 characters

Table 5: xbfm_print_comment parameters

4.2.1.3 xbfm_print_error

This Task prints an error message, with a time stamp, to a log file. This is useful for reporting errors that are not fatal and that do not require stopping a simulation.

Name	Type	Description
message	input string	Message string, up to 100 characters

Table 6: xbfm_print_error parameters

4.2.2 Initialization Tasks

4.2.2.1 xbfm_set_requesterid

This Task changes the BFM requester ID. The BFM will use this value as its requester ID when issuing transactions. The default requester ID for the BFM is 0000h; it is normally not necessary to change this ID.

Name	Type	Description
id	input [15:0]	• id[15:8]: bus number • id[7:3]: device number • id[2:0]: reserved

Table 7: xbfm_set_requesterid parameters

4.2.2.2 xbfm_init

This Task, which must be called at the beginning of a simulation, initializes a BFM and sets the base addresses for I/O and memory address ranges.

Name	Type	Description
io_baseaddr	input [31:0]	Specify base address for IO space (0=not used)
mem32_baseaddr	input [31:0]	Specify base address for MEM32 space (0=not used)
mem64_baseaddr	input [63:0]	Specify base address for MEM64 space (0=not used)

Table 8: xbfm_init parameters

4.2.2.3 xbfm_configure_log

This Task is used to configure log files. A value of 0, the default value, disables all options. You may implement several options at once by adding the desired values.

Name	Type	Description
value	integer	Indicates log file options • XBFM_LOG_CMD: log command issued • XBFM_LOG_DATA: log transmitted / expected data • XBFM_LOG_NOPIPE: disable PIPE logging • XBFM_LOG_NORAW: disable RAW logging

Table 9: xbfm_configure_log parameters

4.2.2.4 xbfm_set_maxpayload

This Task is used to configure the maximum payload size of the testbench.

Name	Type	Description
value	integer	Indicates maximum payload in unit of bytes. Allowed values are: 128,256,512,1024,2048,4096

Table 10: xbfm_set_maxpayload parameters

4.2.2.5 xbfm_set_aspm

This Task is used to configure active state power management. A support value of 0 disables ASPM. You may implement several options by adding the desired values.

Name	Type	Description
support	integer	Indicates log file options • XBFM_ASPM_L0S: allow ASPM L0s • XBFM_ASPM_L1: allow ASPM L1
delay	integer	Indicates the idle time delay to automatically enter low power state (4...31) in steps of 256 ns.

Table 11: xbfm_set_aspm parameters

4.2.2.6 xbfm_set_maxcredit

This Task changes the amount of credits advertised by the BFM. It must called at the beginning of simulation and before link training starts.

Name	Type	Description
nph	integer	Indicates the amount of non-posted header credits (1..127).
npd	integer	Indicates the amount of non-posted data credits (1..127).
ph	integer	Indicates the amount of posted header credits (1..127).
pd	integer	Indicates the amount of posted data credits (8..1024).

Table 12: xbfm_set_maxcredit parameters

4.2.2.7 xbfm_set_l2

This Task is used to make the BFM enter or exit L2 state when it is a Rootport.

Name	Type	Description
value	integer	• 1: request L2 and/or remain in L2 state. • 0: do not request L2 and/or exit L2 state.

Table 13: xbfm_set_l2 parameters

Note: L2 entry request is automatically cleared if the device is directed to Detect.Quiet; this happens for example when the link partner requests L2 exit.

4.2.2.8 xbfm_set_pme

This Task is used to make the BFM request a power management event (PME#). This can be used when the BFM is an Endpoint in order to request an exit from a low-power state.

Name	Type	Description
value	integer	• 1: request PME#. • 0: no action.

Table 14: xbfm_set_pme parameters

4.2.2.9 xbfm_register_write

This Task is used to write to the BFM internal configuration registers. See the *XpressRich3 Reference Manual* for information about configuration space registers.

Name	Type	Description
address	input[11:0]	Address of configuration register to write to (must be a multiple of 4).
data	input[31:0]	Data to be written.

Table 15: xbfm_register_write parameters

4.2.2.10 xbfm_register_read

This Task is used to read the BFM internal configuration registers. See the *XpressRich3 Reference Manual* for information about configuration space registers.

Name	Type	Description
address	input[11:0]	Address of configuration register to read (must be a multiple of 4).
data	input[31:0]	Returns read value.

Table 16: xbfm_register_read parameters

4.2.3 Transaction Tasks

4.2.3.1 xbfm_dword

This Task directs the BFM to send a 1 DW request. The transfer byte count is always four bytes and the address is 32-bit only.

Name	Type	Description
command	integer	Indicates the type of request, supported values are: • XBFM_CFGRD0: configuration read type 0 • XBFM _CFGRD1: configuration read type 1 • XBFM _IORD: IO read • XBFM _MRD: memory read • XBFM _CFGWR0: configuration write type 0 • XBFM _CFGWR1: configuration write type 1 • XBFM _IOWR: IO write • XBFM _MWR: memory write
address	input [31:0]	Specifies target address: this address must be DW-aligned (bits [1:0]=00)
be	input [3:0]	Byte enables: any byte-enable value can be supplied. Default is 1111b which indicates that all bytes must be transferred.
data	input [31:0]	• Write requests: indicates the value to write. • Read requests: if a valid value is supplied, then the XBFM will automatically check that it matches read value. Use XXXXXXXXh in order to disable data checking.

Table 17: xbfm_dword parameters

4.2.3.2 xbfm_dword_id

This Task is the same as xbfm_dword except for an additional parameter that indicates the ID of the issued transaction. This ID can be use to track transaction with xbfm_wait_id..

Name	Type	Description
command	integer	Indicates the type of request, supported values are: • XBFM_CFGRD0: configuration read type 0 • XBFM _CFGRD1: configuration read type 1 • XBFM _IORD: IO read • XBFM _MRD: memory read • XBFM _CFGWR0: configuration write type 0 • XBFM _CFGWR1: configuration write type 1 • XBFM _IOWR: IO write • XBFM _MWR: memory write
address	input [31:0]	Specifies target address: this address must be DW-aligned (bits [1:0]=00)
be	input [3:0]	Byte enables: any byte-enable value can be supplied. Default is 1111b which indicates that all bytes must be transferred.
data	input [31:0]	• Write requests: indicates the value to write. • Read requests: if a valid value is supplied, then the XBFM will automatically check that it matches read value. Use XXXXXXXXh in order to disable data checking.
id	output	ID of issued transaction.

Table 18: xbfm_dword parameters

4.2.3.3 xbfm_burst

This Task directs the BFM to send a burst request. See the memory_write Task description for instructions about how to prepare a data buffer.

Name	Type	Description
command	integer	Indicates the type of request. Supported values are: • XBFM _MRD: memory read • XBFM _MRDLK: memory read locked • XBFM _MWR: memory write • XBFM_FADD: Fetch and Add atomic operation • XBFM_SWAP: unconditional Swap atomic operation • XBFM_CAS: Compare and Swap atomic operation
address	input [63:0]	Specifies the target address, which can start on any byte
bytecount	integer	Total transfer byte count: any value from 1 to 4096 bytes can be specified.
data	input [*][31:0]	• Write requests: indicates the values to write. • Read requests: if a valid data are supplied then the XBFM will automatically check that these match read values. Use XXXXXXXXh for first data word in order to disable data checking.

Table 19: xbfm_burst parameters

Name	Type	Description
tc	input [2:0]	Traffic class: any value from 000b to 111b is allowed; default is 000b.
attr	input [1:0]	Allows you to specify values for relaxed-ordering (RO) and no-snoop (NS) bits. Any value is allowed. Default is 00b.

Table 19: xbfm_burst parameters

4.2.3.4 xbfm_burst_id

This Task is the same as xbfm_burst except that it outputs an additional parameter that indicates the ID of the issued transaction. This ID can be use to track the transaction using xbfm_wait_id.

Name	Type	Description
command	integer	Indicates the type of request. Supported values are: • XBFM _MRD: memory read • XBFM _MRDLK: memory read locked • XBFM _MWR: memory write • XBFM_FADD: Fetch and Add atomic operation • XBFM_SWAP: unconditional Swap atomic operation • XBFM_CAS: Compare and Swap atomic operation
address	input [63:0]	Specifies the target address, which can start on any byte
bytecount	integer	Total transfer byte count: any value from 1 to 4096 bytes can be specified.
data	input [*][31:0]	• Write requests: indicates the values to write. • Read requests: if a valid data are supplied then the XBFM will automatically check that these match read values. Use XXXXXXXXh for first data word in order to disable data checking.
tc	input [2:0]	Traffic class: any value from 000b to 111b is allowed; default is 000b.
attr	input [1:0]	Allows you to specify values for relaxed-ordering (RO) and no-snoop (NS) bits. Any value is allowed. Default is 00b.
id	output	ID of issued transaction.

Table 20: xbfm_burst parameters

4.2.3.5 xbfm_custom_tlp

This Task directs the BFM to send a TLP with a user-defined header.

Name	Type	Description
header	input (127:0)	Header of TLP: byte 0 of header are bits [127:120],.. byte 15 of header are bits [7:0]) Note that: • you must tie bits [31:0] to 0s if a 3 DW header is specified • non-posted TLPs bits [79:72] are ignored and a tag is automatically assigned by the BFM. • bits [95:80] must also be 0s and are automatically replaced with the BFM requester ID
dwordcount	integer	Number of DW of data
data	input [*][31:0]	Data buffer to hold up to 4096 bytes

Table 21: xbfm_custom_tlp parameters

Note: There is only one outstanding request with xbfm_custom_tlp: this is means that if this task is used to transmit a non-posted TLP, then no other TLP can be transmitted until the completion for the preceding request has been received.

4.2.3.6 xbfm_custom_tlp_id

This Task is the same as xbfm_custom_tlp, except that it outputs an additional parameter that indicates the ID of the issued transaction. This ID is used to track a transaction with xbfm_wait_id.

Name	Type	Description
header	input (127:0)	Header of TLP: byte 0 of header are bits [127:120],.. byte 15 of header are bits [7:0]) Note that: • you must tie bits [31:0] to 0s if a 3 DW header is specified • non-posted TLPs bits [79:72] are ignored and a tag is automatically assigned by the BFM. • bits [95:80] must be 0s as well and are automatically replaced with the BFM requester ID
dwordcount	integer	Number of DW of data
data	input [*][31:0]	Data buffer to hold up to 4096 bytes.
id	output	ID of issued transaction.

Table 22: xbfm_custom_tlp_id parameters

Note: There is only one outstanding request with xbfm_custom_tlp_id: this is means that if this task is used to transmit a non-posted TLP, then no other TLP can be transmitted until the completion for the preceding request has been received.

4.2.3.7 xfbm_set_cplstatus

This Task changes the status of subsequent completions sent by BFM.

Name	Type	Description
status	integer	Indicates the completion status to return, supported values are: • XBFM _SC: successful • XBFM _CA: completer abort • XBFM _UR: unsupported request
count	integer	Number of completions to return with specified status code.

Table 23: xbfm_set_cplstatus parameters

4.2.3.8 xfbm_set_cplsize

This Task changes the size of subsequent completions sent by BFM.

Name	Type	Description
value	integer	• 0: completions can be any size up to maximum payload • 64: completions are restricted to 64-byte packets • 128: completions are restricted to 128-byte packets

Table 24: xbfm_set_cplsize parameters

4.2.3.9 xbfm_wait

This Task waits until the BFM has finished processing outstanding requests. If outstanding requests require completions, then this Task will wait until all completions have been received

4.2.3.10 xbfm_wait_id

This Task returns the transaction status and received data (if any) at the end of a transaction transmitted with xbfm_dword_id, xbfm_burst_id, or sbfm_custom_tlp_id.

Name	Type	Description
trnid	integer	Transaction ID
status	output integer	Indicates the completion status of a transaction. Possible values are: • XBFM_SC: transaction successful • XBFM_CA: completer abort received • XBFM_UR: unsupported request received • XBFM_TIMEOUT: timeout occurred, completion for transaction was not received
data	output [*][31:0]	Data buffer to hold up to 4096 bytes of received data (if any)

Table 25: xbfm_wait_id parameters

4.2.3.11 xbfm_wait_event

This Task waits until the BFM signals a specific event. Note that execution of the calling process will be blocked if the specified event is not received.

Name	Type	Description
event_id	integer	Indicates which event is expected. The value of this parameter can be any one of the events listed in Section 4.1.

Table 26: xbfm_wait_event parameters

4.2.3.12 xbfm_wait_linkup

This Task waits until link training is completed. It does not contain any parameters.

4.2.3.13 xbfm_wait_space_hit

This Task waits until a request corresponding to a specific address range in BFM memory spaces is received.

Name	Type	Description
space	integer	Indicates which XBFM memory space is targeted. Valid values are: • XBFM_IO: IO space • XBFM_MEM32: 32-bit addressing memory space • XBFM_MEM64: 64-bit addressing memory space
offsetl	input[31:0]	Specifies the lowest offset of detection range within memory space.
offseth	input[31:0]	Specifies the highest offset of detection range within memory space.
rw	integer	Indicates which type of request will be detected. Valid values are: • XBFM_RW • XBFM_READ • XBFM_WRITE

Table 27: xbfm_wait_space_hit parameters

4.2.4 Memory Access Tasks

4.2.4.1 xbfm_memory_write

This Task writes the content of a data buffer to a BFM memory space starting from a specified address.

Name	Type	Description
space	integer	Specifies which XBFM memory space is targeted. Valid values are: • XBFM_IO: IO space • XBFM_MEM32: 32-bit addressing memory space • XBFM_MEM64: 64-bit addressing memory space
offset	input[31:0]	Specifies at which offset within the memory space data write must start. This value must be DW aligned (bits [1:0]=00)

Table 28: xbfm_memory_write parameters

Name	Type	Description
num_dword	integer	Number of DW to write
data	input [*][31:0]	Data buffer to hold up to 4096 bytes

Table 28: xbfm_memory_write parameters

4.2.4.2 xbfm_memory_read

This Task reads the content of a BFM memory space starting from a specified address.

Name	Type	Description
space	integer	Specifies which XBFM memory space is targeted. Valid values are: • XBFM_IO: IO space • XBFM_MEM32: 32-bit addressing memory space • XBFM_MEM64: 64-bit addressing memory space
offset	input [31:0]	Specifies at which offset within the memory space data read must start. This value must be DW aligned (bits [1:0]=00)
num_dword	integer	Number of DW to read
data	Output [*][31:0]	Data buffer to hold up to 4096 bytes

Table 29: xbfm_memory_read parameters

4.2.4.3 xbfm_memory_compare

This Task compares the content of a data buffer with data from a BFM memory space starting from a specified address. Mismatches are reported in a log file.

Name	Type	Description
space	integer	Specifies which XBFM memory space is targeted. Valid values are: • XBFM_IO: IO space • XBFM_MEM32: 32-bit addressing memory space • XBFM_MEM64: 64-bit addressing memory space
offset	input	Specifies at which offset within the memory space data compare must start. This value must be DW aligned (bits [1:0]=00)
num_dword	integer	Number of DW to compare
data	input [*][31:0]	Data buffer to hold up to 4096 bytes
fbe	input [3:0]	Specifies the byte enables for first DW of comparison. Default value is 1111b that indicates that all bytes must be compared.
lbe	input [3:0]	Specifies the byte enables for last DW of comparison. Default value is 1111b that indicates that all bytes must be compared

Table 30: xbfm_memory_compare parameters

4.2.4.4 xbfm_memory_compare_count

This Task is the same as xbfm_memory_compare, except that it also counts the number of erroneous DWORDs.

Name	Type	Description
space	integer	Specifies which XBFM memory space is targeted. Valid values are: • XBFM_IO: IO space • XBFM_MEM32: 32-bit addressing memory space • XBFM_MEM64: 64-bit addressing memory space
offset	input	Specifies at which offset within the memory space data compare must start. This value must be DW aligned (bits [1:0]=00)
num_dword	integer	Number of DW to compare
data	input [*][31:0]	Data buffer to hold up to 4096 bytes
fbe	input [3:0]	Specifies the byte enables for first DW of comparison. Default value is 1111b that indicates that all bytes must be compared.
lbe	input [3:0]	Specifies the byte enables for last DW of comparison. Default value is 1111b that indicates that all bytes must be compared.
error_count	output integer	Returns the number of erroneous DWORDs; 0 indicates that no error was detected.

Table 31: xbfm_memory_compare_count parameters

4.2.4.5 xfbm_memory_dump

This Task dumps the content of a memory space to a file.

Name	Type	Description
status	integer	Specifies which XBFM memory space is targeted. Valid values are: • XBFM_IO: IO space • XBFM_MEM32: 32-bit addressing memory space • XBFM_MEM64: 64-bit addressing memory space
file	String	File name, up to 100 characters

Table 32: xbfm_memory_dump parameters

4.2.5 Utilities

4.2.5.1 xbfm_buffer_fill

This Task fills a data buffer with a data ramp (00h, 01h, 02h.., FEh, FFh). This is useful for preparing data for a burst transfer.

Name	Type	Description
num_dword	integer	Number of DWs to fill
data	Output [*][31:0]	Data buffer to hold up to 4096 bytes

Table 33: xbfm_buffer_fill parameters