2023

AXI4 Protocol over AIB Latency Reference Guide Revision 1.0

27 JUNE 2023

Contents

1. Background 2
2. Simulation Parameters 2
3. Channel Aligner Modes 3
4. AXI4-Stream Latency and FIFO Buffer (Channel Aligner in Synchronous Mode, SYNC_FIFO=1) 4
5. AXI4 Latency and FIFO Buffer (Channel Aligner in Synchronous Mode, SYNC_FIFO=1) 5
6. AXI4-Stream Latency and FIFO Buffer (Channel Aligner in Asynchronous Mode, SYNC_FIFO=0) 6
7. AXI4 Latency and FIFO Buffer (Channel Aligner in Asynchronous Mode, SYNC_FIFO=0) 7
8. How to Specify RX FIFO Buffer Size 8

8.1. AXI4-Stream 8

8.2. AXI4-MM 8

9. Latency Optimization 11

10. Addendum: Example waveform capture for Parameters A and B (Config 2 for AXI4-ST) 12

10.1. Latency A 12

10.2. Latency B 13

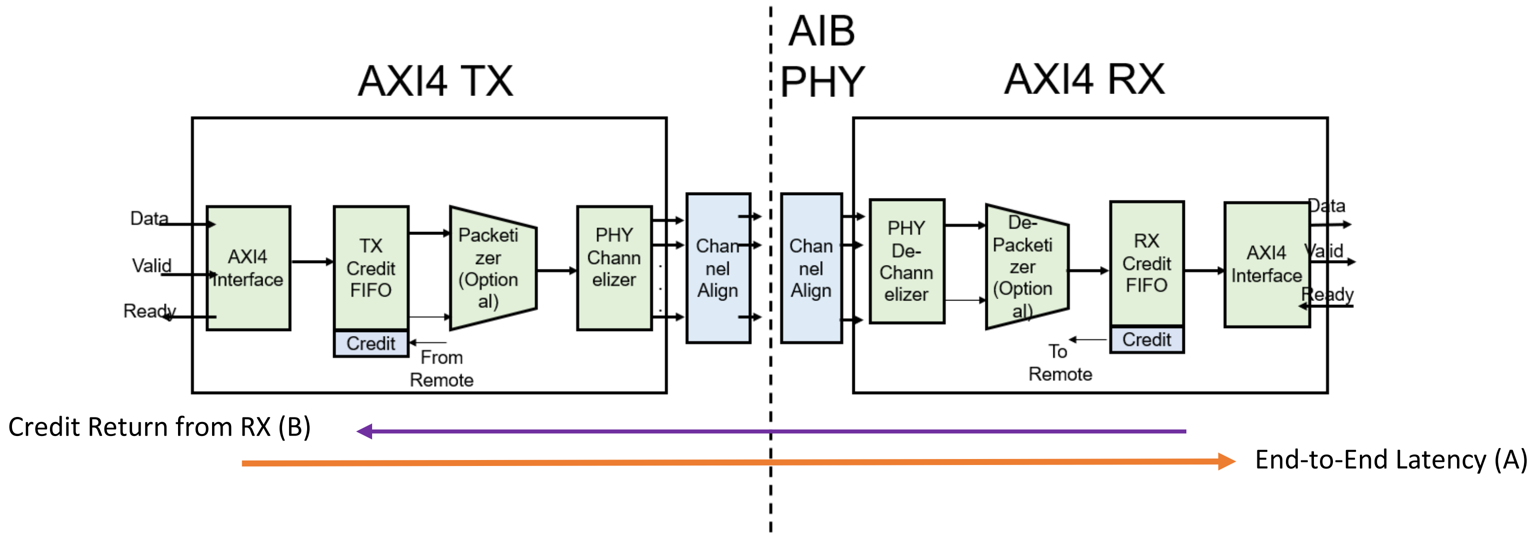
1. Background

There are 2 critical latency paths in AXI4 protocol:

A) End-to-End latency – from TX to RX
B) Credit return latency – from RX to TX

For AXI4 without flow control (AXI4-Stream without ready) B is not relevant for other AXI4 protocols with flow control both A and B play an important role as discussed below.

AXI4 protocol over AIB uses credit mechanism to manage flow control between follower and leader chiplet. This credit mechanism also allows to manage the multiple clock cycle handshake between chiplets.



To avoid stalling in the traffic (bubbles) the recommendation is to size the RX credit FIFO to match A+B+k where k is an additional buffer. RX FIFO size is parameter for IP generation.

2. Simulation Parameters

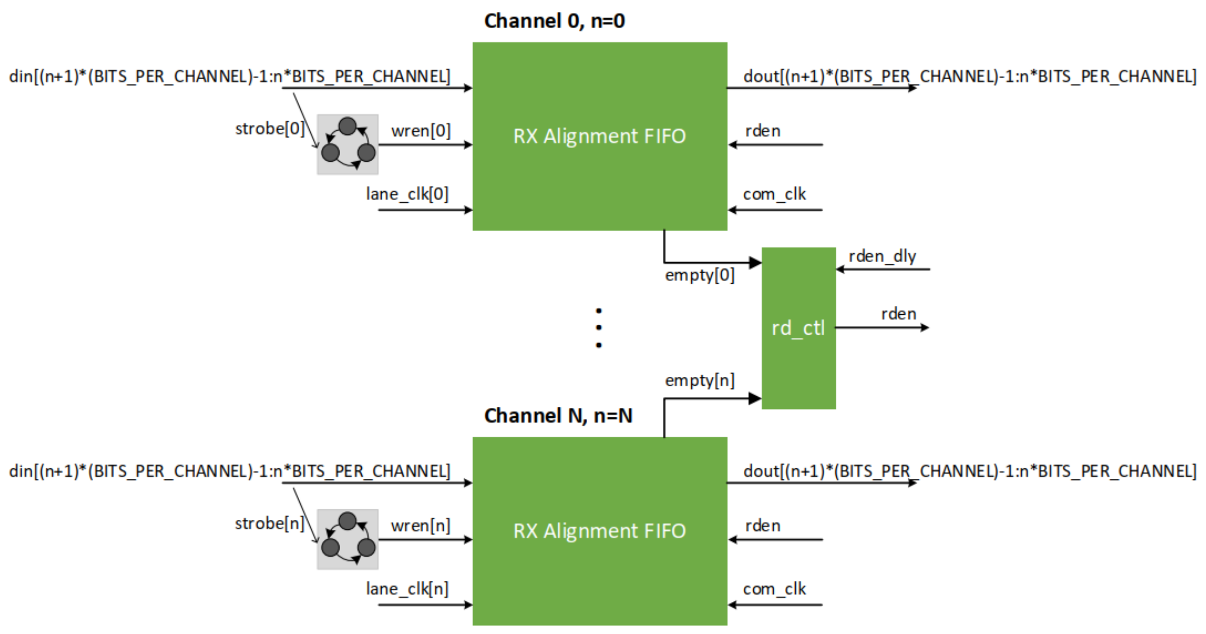
Below is a table of measured latency in simulation and RX FIFO buffer size recommendation for AXI4-Stream and AXI4 based on the following parameters.

AXI4 Mode	**Freq (Ghz)**
Half (Gen1)	0.5
Full (Gen1)	1
Quarter (Gen2)	0.5
Half (Gen2)	1
Full (Gen2)	2

3. Channel Aligner Modes

Supports 2 modes

1) SYNC_FIFO=1 channel align FIFO operates in synchronous mode, lane_clk and com_clk from same source which is usually m_rd_clk/m_wr_clk
2) SYNC_FIFO=0 channel align FIFO operates in asynchronous mode, lane_clk and com_clk from different source, FIFO write clock is lane_clk from the channel and the com_clk is m_rd_clk/m_wr_clk


Channel 0, n=0
din[(n+1)*(BITS_PER_CHANNEL)-1:n*BITS_PER_CHANNEL]
strobe[0]
wren[0]
lane_clk[0]
RX Alignment FIFO
dout[(n+1)*(BITS_PER_CHANNEL)-1:n*BITS_PER_CHANNEL]
rden
com_clk
empty[0]
rden_dly
rd_ctl
rden
empty[n]
Channel N, n=N
din[(n+1)*(BITS_PER_CHANNEL)-1:n*BITS_PER_CHANNEL]
strobe[n]
wren[n]
lane_clk[n]
RX Alignment FIFO
dout[(n+1)*(BITS_PER_CHANNEL)-1:n*BITS_PER_CHANNEL]
rden
com_clk

Based on which mode you choose latency could be different, async mode results in slightly longer latency and therefore FIFO depth could be slightly more for async mode.

4. AXI4-Stream Latency and FIFO Buffer (Channel Aligner in Synchronous Mode, SYNC_FIFO=1)

Config No	Leader AIB Mode	Follower AIB Mode	FIFO Mode (TX→RX)	A (normalized to AIB Fwd clk)	B (normalized to AIB Fwd clk)	k	Freq Factor between AXI and AIB (F)	RX FIFO Size = Ceiling ((A+B+k)/F)
1	AIB2.0 (Gen1)	AIB 1.0	Half→Half	44	40	2	2	43
2	AIB2.0 (Gen1)	AIB1.0	Full→Half	40	26	2	1	68
3	AIB2.0 (Gen1)	AIB1.0	Half→Full	34	40	2	1	76
4	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Half→Half	50	38	2	2	45
5	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Full→Full	31	28	2	1	61
6	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Quarter→Quarter	88	66	2	4	39
7	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Full→Half	48	30	2	1	80
8	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Full→Quarter	84	31	2	1	117
9	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Half→Full	34	38	2	1	74
10	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Half→Quarter	88	44	2	1	134
11	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Quarter→Full	39	65	2	1	106
12	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Quarter→Half	58	66	2	1	126

5. AXI4 Latency and FIFO Buffer (Channel Aligner in Synchronous Mode, SYNC_FIFO=1)

Config No	Leader AIB Mode	Follower AIB Mode	FIFO Mode (TX→RX)	A (normalized to AIB Fwd clk)	B (normalized to AIB Fwd clk)	k	Freq Factor between AXI and AIB (F)	RX FIFO Size = Ceiling ((A+B+k)/F)
1	AIB2.0 (Gen 1)	AIB 1.0	Half→Half	44	40	2	2	43
2	AIB2.0 (Gen 2)	AIB2.0 (Gen2)	Half→Half	52	42	2	2	48
3	AIB2.0 (Gen 2)	AIB2.0 (Gen2)	Full→Full	33	26	2	1	61
4	AIB2.0 (Gen 2)	AIB2.0 (Gen2)	Quarter→Quarter	92	72	2	4	42

6. AXI4-Stream Latency and FIFO Buffer (Channel Aligner in Asynchronous Mode, SYNC_FIFO=0)

Config No	Leader AIB Mode	Follower AIB Mode	FIFO Mode (TX→RX)	A (normalized to AIB Fwd clk)	B (normalized to AIB Fwd clk)	k	Freq Factor between AXI and AIB (F)	RX FIFO Size = Ceiling ((A+B+k)/F)
1	AIB2.0 (Gen1)	AIB 1.0	Half→Half	54	50	2	2	53
2	AIB2.0 (Gen1)	AIB1.0	Full→Half	50	31	2	1	83
3	AIB2.0 (Gen1)	AIB1.0	Half→Full	39	49	2	1	90
4	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Half→Half	50	38	2	2	45
5	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Full→Full	36	33	2	1	71
6	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Quarter→Quarter	88	66	2	4	39
7	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Full→Half	58	35	2	1	95
8	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Full→Quarter	104	36	2	1	142
9	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Half→Full	39	49	2	1	90
10	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Half→Quarter	108	54	2	1	164
11	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Quarter→Full	46	84	2	1	132
12	AIB2.0 (Gen2)	AIB2.0 (Gen2)	Quarter→Half	68	88	2	1	158

7. AXI4 Latency and FIFO Buffer (Channel Aligner in Asynchronous Mode, SYNC_FIFO=0)

Config No	Leader AIB Mode	Follower AIB Mode	FIFO Mode (TX→RX)	A (normalized to AIB Fwd clk)	B (normalized to AIB Fwd clk)	k	Freq Factor between AXI and AIB (F)	RX FIFO Size = Ceiling ((A+B+k)/F)
1	AIB2.0 (Gen 1)	AIB 1.0	Half→Half	54	50	2	2	53
2	AIB2.0 (Gen 2)	AIB2.0 (Gen2)	Half→Half	62	52	2	2	58
3	AIB2.0 (Gen 2)	AIB2.0 (Gen2)	Full→Full	38	31	2	1	71
4	AIB2.0 (Gen 2)	AIB2.0 (Gen2)	Quarter→Quarter	112	92	2	4	52

8. How to Specify RX FIFO Buffer Size

Can be specified as a configuration parameter in IP configuration file. Here are examples for AXI4-Stream and AXI4-MM

8.1. AXI4-Stream

```
MODULE axi_st
// PHY and AIB Configuration
NUM_CHAN                                7
CHAN_TYPE                               Gen1Only  //Gen1Only, Gen2Only, Gen2, AIBO
// Channel Alignment Strobe Configuration
TX_ENABLE_STROBE                        True        // If False, all strobe functionality is removed.
RX_ENABLE_STROBE                        True        // If False, all strobe functionality is removed.
TX_PERSISTENT_STROBE                    True        // If True strobes are persistent (always there). If false, they are recoverable and can be
reused for data
RX_PERSISTENT_STROBE                    True        // If True strobes are persistent (always there). If false, they are recoverable and can be
reused for data
TX_USER_STROBE                  True    // If True, then we input user generated signal

RX_USER_STROBE                  True    // If True, then we input user generated signal
TX_STROBE_GEN1_LOC                      8           // Location of Strobe when in Gen1 Mode
RX_STROBE_GEN1_LOC                      8           // Location of Strobe when in Gen1 Mode

// Word Marker Configuration
TX_ENABLE_MARKER                        True        // If False, all Marker functionality is removed.
RX_ENABLE_MARKER                        True        // If False, all Marker functionality is removed.
TX_PERSISTENT_MARKER                    True        // If True Markers are persistent (always there). If false, they are recoverable and can be
reused for data
RX_PERSISTENT_MARKER                    True        // If True Markers are persistent (always there). If false, they are recoverable and can be
reused for data//
TX_USER_MARKER                          True
RX_USER_MARKER                          True
TX_MARKER_GEN1_LOC                      39          // Location of Marker when in Gen1 Mode
RX_MARKER_GEN1_LOC                      39          // Location of Marker when in Gen1 Mode
TX_REG_PHY              False
RX_REG_PHY              False

SUPPORT_ASYMMETRIC                      True        // Support Asymmetric Gearboxing (e.g. Full to Half)
llink ST
{
  TX_FIFO_DEPTH       1
  RX_FIFO_DEPTH       68 # for Full→Half

  output user_tdata   256
  input user_tready   ready
  output user_tvalid  valid
}
```

8.2. AXI4-MM

```
MODULE axi_mm


// PHY and AIB Configuration
NUM_CHAN                                2
CHAN_TYPE                               Gen1Only              //Gen1Only, Gen2Only, Gen2, AIBO
TX_RATE                                 Half                  // Full, Half, Quarter
RX_RATE                                 Half                  // Full, Half, Quarter
```

```
// Channel Alignment Strobe Configuration
TX_ENABLE_STROBE                True        // If False, all strobe functionality is removed.
RX_ENABLE_STROBE                True        // If False, all strobe functionality is removed.
TX_PERSISTENT_STROBE                        True        // If True strobes are persistent (always there). If false, they are recoverable and can
be reused for data
RX_PERSISTENT_STROBE                        True        // If True strobes are persistent (always there). If false, they are recoverable and can
be reused for data
TX_USER_STROBE                              False       // If True, then we input user generated signal
RX_USER_STROBE                              False       // If True, then we input user generated signal
TX_STROBE_GEN1_LOC                          7           // Location of Strobe when in Gen1 Mode
RX_STROBE_GEN1_LOC                          7           // Location of Strobe when in Gen1 Mode

// Word Marker Configuration
TX_ENABLE_MARKER                True        // If False, all Marker functionality is removed.
RX_ENABLE_MARKER                True        // If False, all Marker functionality is removed.
TX_PERSISTENT_MARKER                        True        // If True Markers are persistent (always there). If false, they are recoverable and can
be reused for data
RX_PERSISTENT_MARKER                        True        // If True Markers are persistent (always there). If false, they are recoverable and can
be reused for data//
TX_USER_MARKER                              False
RX_USER_MARKER                              False
TX_MARKER_GEN1_LOC                          39          // Location of Marker when in Gen1 Mode
RX_MARKER_GEN1_LOC                          39          // Location of Marker when in Gen1 Mode
TX_REG_PHY          False
RX_REG_PHY          False

// Packetization
TX_ENABLE_PACKETIZATION                                 True
RX_ENABLE_PACKETIZATION                                 True
TX_PACKET_MAX_SIZE                                      0           // Number of bits to packetize to. 0 means all available data.
RX_PACKET_MAX_SIZE                                      0           // Number of bits to packetize to. 0 means all available data.
PACKETIZATION_PACKING_EN                    False       // If True, enable packing which makes better use of the BW

llink AR
{
 TX_FIFO_DEPTH       1
 RX_FIFO_DEPTH       43

 output user_arid     4
 output user_arsize   3
 output user_arlen    8
 output user_arburst  2
 output user_araddr   32
 output user_arvalid  valid
 input  user_arready  ready
}

llink AW
{
 TX_FIFO_DEPTH       1
 RX_FIFO_DEPTH       43

 output user_awid     4
 output user_awsize   3
 output user_awlen    8
 output user_awburst  2
 output user_awaddr   32
 output user_awvalid  valid
 input  user_awready  ready
}

llink W
{
 TX_FIFO_DEPTH       1
 RX_FIFO_DEPTH       43
```

```
  output user_wid     4
  output user_wdata   128
  output user_wstrb   16
  output user_wlast
  output user_wvalid  valid
  input  user_wready  ready
}
llink R
{
  TX_FIFO_DEPTH      1
  RX_FIFO_DEPTH      43

  input  user_rid     4
  input  user_rdata   128
  input  user_rlast
  input  user_rresp   2
  input  user_rvalid  valid
  output user_rready  ready
}
llink B
{
  TX_FIFO_DEPTH      1
  RX_FIFO_DEPTH      43

  input  user_bid     4
  input  user_bresp   2
  input  user_bvalid  valid
  output user_bready  ready
}
```

9. Latency Optimization

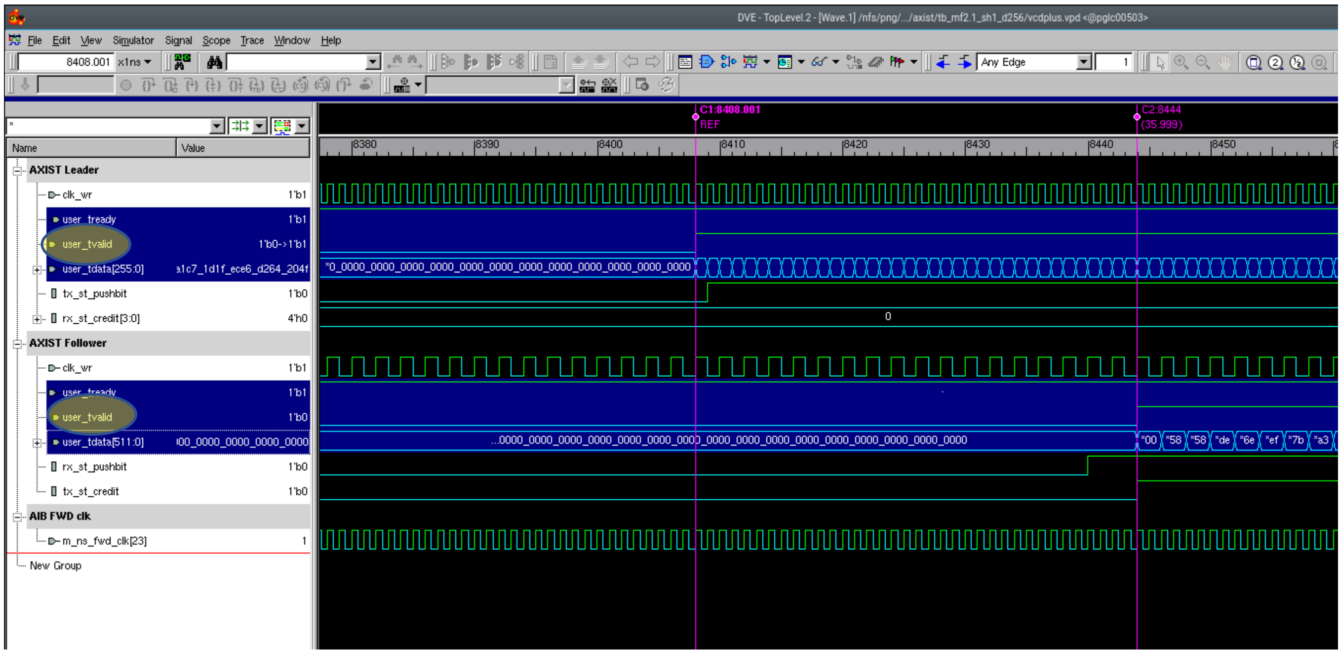
These latencies reported here are out of the box. Here are some tips to improve latency with system knowledge:

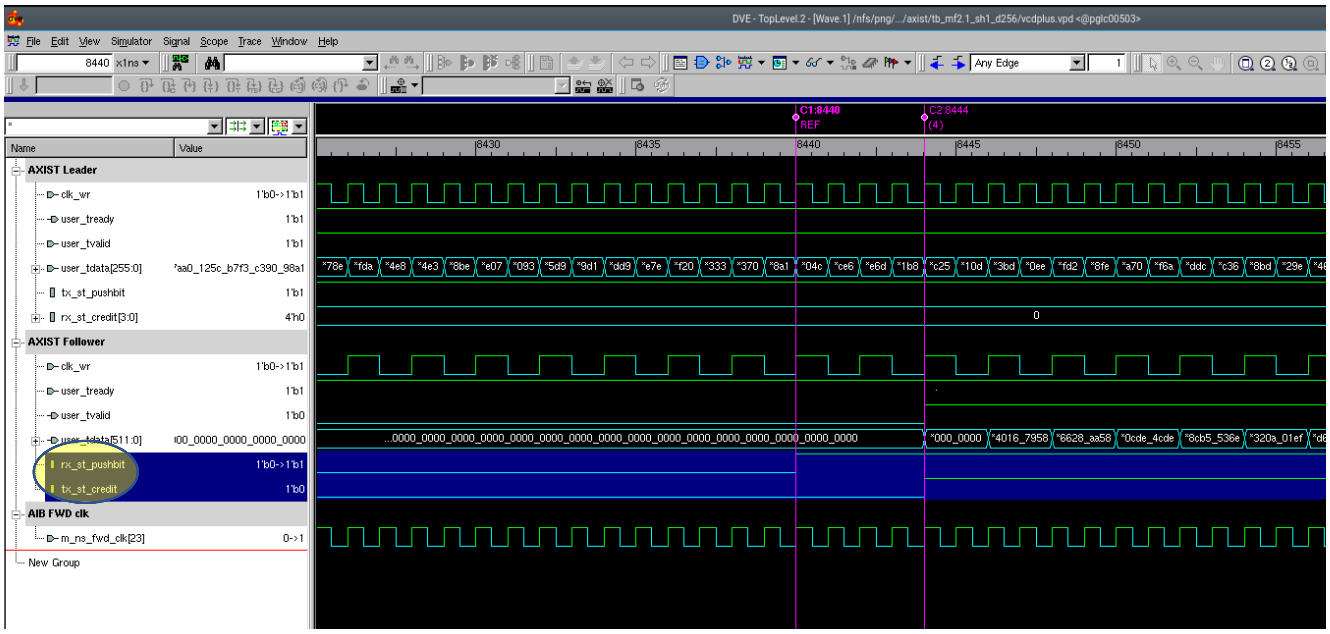
a) Lane to lane to skew modeled in testbench as delay between AIB and CA. This delay is worst cased at 5 clock cycle this can be optimized for your interface and system once you have the system parameters extracted.
b) Improve RX/TX phasecomp RD delay with the knowledge of AIB write vs read clock wander margin. These values can be programmed to AIB CSR 0x208/0x218.

10. Addendum: Example waveform capture for Parameters A and B (Config 2 for AXI4-ST)

10.1. Latency A

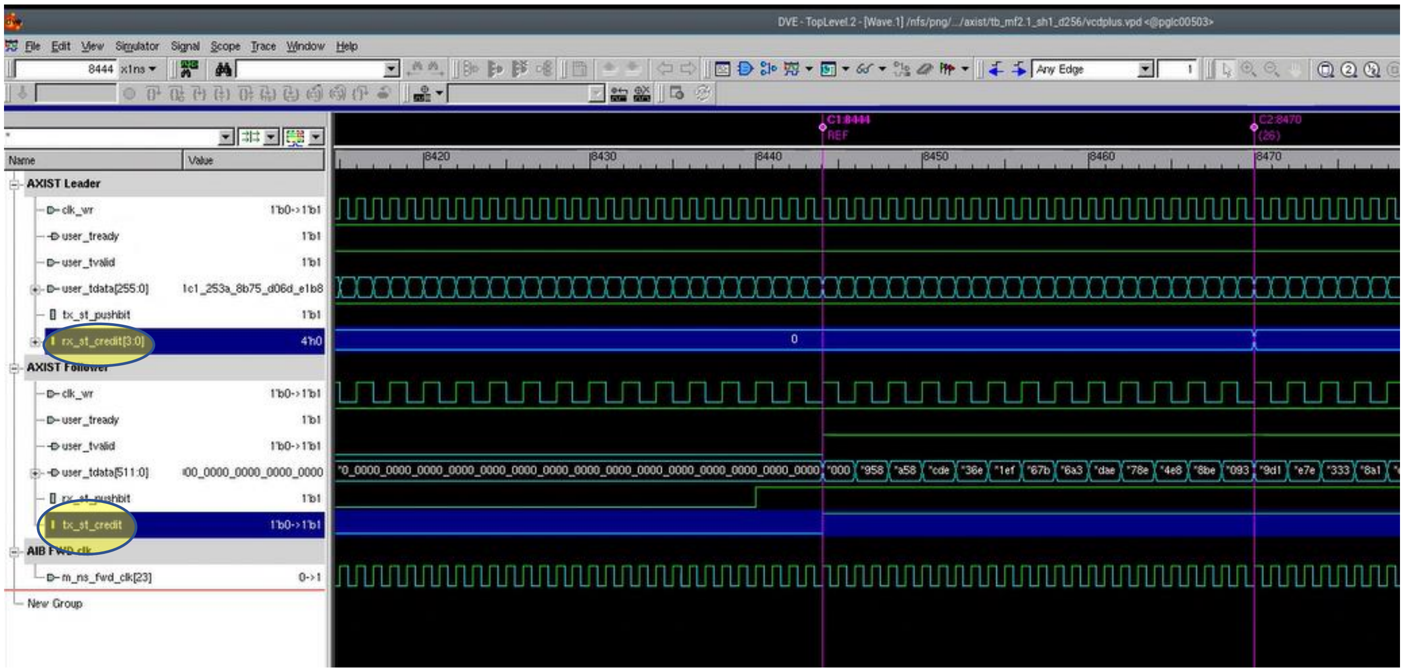
Sum of valid to valid + push to pop on RX FIFO Buffer





10.2. Latency B

Credit return from follower to leader


DVE - TopLevel.2 - [Wave.1] /nfs/png/.../axist/tb_mf2.1_sh1_d256/vcdplus.vpd <@pglc00503>
File Edit View Simulator Signal Scope Trace Window Help
8444 x1ns
Any Edge
C1:8444 REF
C2:8470 (26)
Name
Value
AXIST Leader
clk_wr 1'b0->1'b1
user_tready 1'b1
user_tvalid 1'b1
user_tdata[255:0] 1c1_253a_8b75_d06d_e1b8
tx_st_pushbit 1'b1
rx_st_credit[3:0] 4'h0
AXIST Follower
clk_wr 1'b0->1'b1
user_tready 1'b1
user_tvalid 1'b0->1'b1
user_tdata[511:0] 00_0000_0000_0000_0000
rx_st_pushbit 1'b1
tx_st_credit 1'b0->1'b1
AIB FWD clk
m_ns_fwd_clk[23] 0->1
New Group
8420
8430
8440
8450
8460
8470
0