Blue Cheetah Analog Design
AIB Gen2 PHY Integration Guide

Version 9.5

March 6, 2024

Table of Contents

Table of Figures

Table of Tables

1.Introduction

This document is the integration guide for Blue Cheetah Analog Design's (BCA's) Advanced Interface Bus 2.0 (AIB2.0) PHY. It describes the PHY's important aspects to be considered during integration of the hard block on SoC.

1.1. PHY Overview

The PHY is delivered as a single, hard macro implemented in Intel's 22FFL process technology. It is compliant to the new, AIB 2.0 specification supporting 24 channels, each with 40 transmit lanes and independent 40 receive lanes (also referred to as 80 IOs balanced), each lane operating at data rates of up to 4 Gbps. The PHY is designed as a dual-mode interface, able to be configured as either a leader or a follower.

BCA PHY implements an Avalon interface for host access purposes. Through the interface, the host can initialize control registers and read status registers. As Avalon interface is used to access several test registers, SoC integration shall guarantee the interface is functional at chip top level for DFT purpose (Manufacturing Test).

The figure below shows the AIB PHY block diagram.

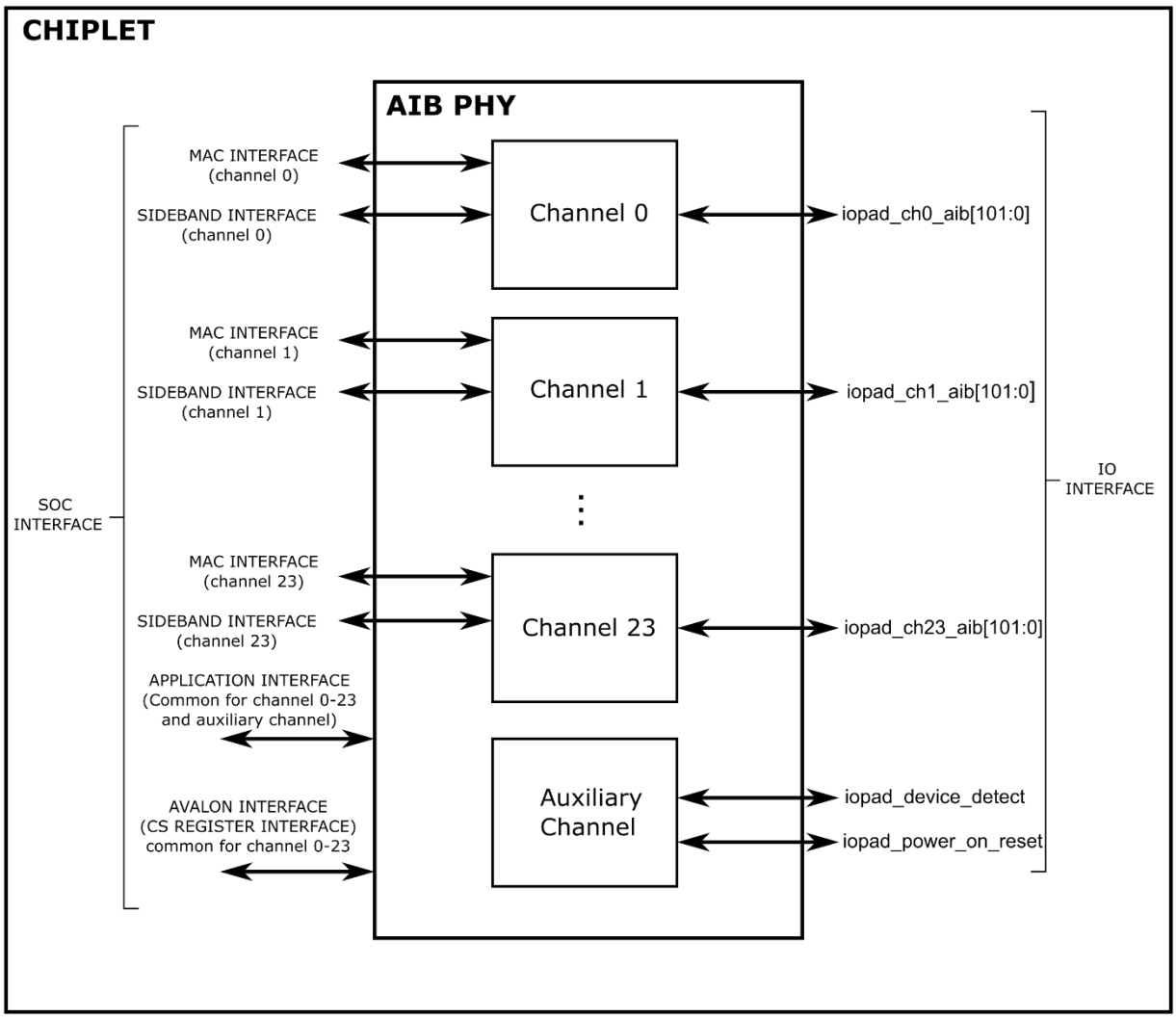


Figure 1: AIB PHY block diagram.

2.Interface Signals

The table below describes all the interface signals implemented on BCA AIB 2.0 PHY.

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
POWER SUPPLY PORTS						
vddc1	inout	1	-	-	-	This is low noise power rail used to power the clock circuits. This is shorted to vddc2 at the board level.
vddc2	inout	1	-	-	-	The digital core voltage. Supplies power to all non-IO drive circuits in the PHY. In any given AIB link, there are two VDDC rails, one for the Leader and one for the Follower, which come up independently.
vddtx	inout	1	-	-	-	Supplies power to the high-speed data channel buffers when operating in Gen1/Gen2 mode: 0.4V for GEN2. 0.85V for GEN1.
vss	inout	1	-	-	-	Power supply ground pin
IO PADS[1]						
iopad_ch0_aib	Inout	102	-	-	-	Channel 0 IO pads
iopad_ch1_aib	Inout	102	-	-	-	Channel 1 IO pads

[1] For AIB Gen 1.0 only implementations, unused IO pads, which corresponds to the 20 transmit data MSBs and 20 receive data MSBs of each channel, can be left as unconnected. For more details about IO pads, see AIB PHY functional spec.

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
iopad_ch2_aib	Inout	102	-	-	-	Channel 2 IO pads
iopad_ch3_aib	Inout	102	-	-	-	Channel 3 IO pads
iopad_ch4_aib	Inout	102	-	-	-	Channel 4 IO pads
iopad_ch5_aib	Inout	102	-	-	-	Channel 5 IO pads
iopad_ch6_aib	Inout	102	-	-	-	Channel 6 IO pads
iopad_ch7_aib	Inout	102	-	-	-	Channel 7 IO pads
iopad_ch8_aib	Inout	102	-	-	-	Channel 8 IO pads
iopad_ch9_aib	Inout	102	-	-	-	Channel 9 IO pads
iopad_ch10_aib	Inout	102	-	-	-	Channel 10 IO pads
iopad_ch11_aib	Inout	102	-	-	-	Channel 11 IO pads
iopad_ch12_aib	Inout	102	-	-	-	Channel 12 IO pads
iopad_ch13_aib	Inout	102	-	-	-	Channel 13 IO pads
iopad_ch14_aib	Inout	102	-	-	-	Channel 14 IO pads
iopad_ch15_aib	Inout	102	-	-	-	Channel 15 IO pads
iopad_ch16_aib	Inout	102	-	-	-	Channel 16 IO pads
iopad_ch17_aib	Inout	102	-	-	-	Channel 17 IO pads
iopad_ch18_aib	Inout	102	-	-	-	Channel 18 IO pads
iopad_ch19_aib	Inout	102	-	-	-	Channel 19 IO pads
iopad_ch20_aib	Inout	102	-	-	-	Channel 20 IO pads
iopad_ch21_aib	Inout	102	-	-	-	Channel 21 IO pads
iopad_ch22_aib	Inout	102	-	-	-	Channel 22 IO pads
iopad_ch23_aib	Inout	102	-	-	-	Channel 23 IO pads
AUXILIARY CHANNEL IOs						
iopad_device_detect	Inout	1	-	-	1	Asserted by the leader to Indicate the presence of a valid leader
iopad_power_on_reset	Inout	1	-	-	1	Performs a power-on-reset in the adapter
MAC INTERFACE						
data_out_f	output	7680(320x24)	0x0	m_rd_clk domain	-	Rx FIFO data output from adapter to MAC. 320 bits per channel.

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
data_out	output	1920 (80x24)	0x0	fs_fwd_clk domain	-	Data output for register mode. 80 bits per channel.
m_fs_rcv_clk	output	24	0 (during power-on reset and configuration phase)	fs_rcv_clk domain	-	Clock received from the far side and converted from quasi-differential to single-ended. One per channel. **It shall not be used in GEN2 mode.**
m_fs_fwd_clk	output	24	0 (during power-on reset and configuration phase)	fs_fwd_clk domain	-	Clock received from the far side and converted from quasi-differential to single-ended. One bit per channel.
ns_fwd_clk	output	24	0 (during power-on reset and configuration phase)	ns_fwd_clk	-	Near side forwarded clock to SOC/MAC. One bit per channel.
ns_fwd_clk_div	output	24	0	ns_fwd_clk	-	Divided near side forwarded clock to SOC/MAC. One bit per channel. It shall be used for FIFO mode only.
fs_fwd_clk	output	24	0 (during power-on reset and configuration phase)	fs_fwd_clk domain	-	Far side forwarded clock to SoC/MAC. One bit per channel.
fs_fwd_clk_div	output	24	0 (during power-on reset and configuration phase)	fs_fwd_clk domain	-	Divided Far side forwarded clock to SoC/MAC. One bit per channel. It shall be used for FIFO mode only.
fs_mac_rdy	output	24	0 (during power-on reset and configuration phase)	ASYNC	1	Indicates that the far-side MAC/SoC is ready to transmit data.
m_rx_align_done	output	24	0	m_rd_clk domain	1	Indicates that the receiving AIB Adapter is aligned to incoming word marked data

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
data_in_f	input	7680(320x24)	-	m_wr_clk domain	-	Data input for FIFO mode.
data_in	input	1920 (80x24)	-	m_ns_fwd_clk	-	Data input for register mode.
m_ns_rcv_clk	input	24	-	m_ns_rcv_clk	-	Near side received clock from MAC to adapter used for transmitting data. **It is ignored for GEN2 mode**. One bit per channel.
m_ns_fwd_clk	input	24	-	m_ns_fwd_clk	-	Near side forwarded clock from MAC to adapter used for transmitting data. One bit per channel.
m_wr_clk	input	24	-	m_wr_clk	-	Write clock to write data from MAC to TX FIFO. One bit per channel.
m_rd_clk	input	24	-	m_rd_clk	-	Read clock to read data in Rx FIFO. One bit per channel.
ns_adapter_rstn	input	24	-	ASYNC	0	Active low asynchronous reset to adapter. One bit per channel. **Only for AIB plus.**
ns_mac_rdy	input	24	-	ASYNC	1	used to communicate that the MAC/SoC is ready for data transmission. One bit per channel.
APPLICATION INTERFACE						
m_device_detect	output	1	Does not come from register. It depends on auxiliary channel IO pad value.	ASYNC	1	A copy of the AIB interface signal device_detect from the Leader, qualified by m_device_detect_ovrd.
o_m_power_on_reset	output	1	Does not come from register. It	ASYNC	1	A copy of the power_on_reset

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
			depends on auxiliary channel IO pad value.			signal from the Follower, qualified by override from m_por_ovrd.
i_m_power_on_reset	input	1	-	ASYNC	1	Controls the power_on_reset signal sent to the Leader. Must be stable at power-up.
m_por_ovrd	input	1	-	ASYNC	1	Intended for standalone test without an AIB partner. If LO, the Leader is not in reset. If HI and the AIB interface signal power_on_reset input is 1, the Leader is held in reset. The Leader's power_on_reset input is required to have a weak pullup so that a no connect on that input will result in power_on_reset=HI.
m_device_detect_ovrd	input	1	-	ASYNC	1	Intended for standalone test without an AIB partner. If LO, the Follower uses the AIB interface device_detect input. If HI, the Follower outputs m_device_detect=1.
dual_mode_select	input	1	-	STATIC	1	Indicates the mode selected: Low: AIB interface is a Follower. High: AIB interface is a Leader.
m_gen2_mode	input	1	-	ASYNC	1	If LO when the AIB interface is released from reset, the AIB interface is in Gen1 mode. If HI when the AIB interface is

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
						released from reset, the AIB interface is in Gen2 mode. The signal shall be static after reset is released.
i_conf_done	input	1	-	ASYNC	1	Single control to reset adapter of all AIB channels in the interface. LO=reset, HI=out of reset. Most of CSR registers are configured when i_conf_done = 0
JTAG BOUNDARY SCAN INTERFACE						
o_jtag_tdo	output	1	0	JTAG domain	-	last boundary scan chain output, TDO
i_jtag_clkdr	input	1	-	JTAG domain	-	Boundary scan test clock
i_jtag_clksel	input	1	-	JTAG domain	1	Select the JTAG clock or the operational clock for the register in the I/O block
i_jtag_intest	input	1	-	JTAG domain	1	Enable testing of data path
i_jtag_mode	input	1	-	JTAG domain	1	Teste mode select
i_jtag_rstb	input	1	-	JTAG domain	0	JTAG reset
i_jtag_rstb_en	input	1	-	JTAG domain	0	JTAG reset enable
i_jtag_tdi	input	1	-	JTAG domain	-	Test data input, TDI
i_jtag_tx_scanen	input	1	-	JTAG domain	1	JTAG shift DR, active high
i_jtag_weakpdn	input	1	-	JTAG domain	1	Enable weak pull-down on all AIB IO blocks
i_jtag_weakpu	input	1	-	JTAG domain	1	Enable weak pull-up on all AIB IO blocks
SCAN INTERFACE						

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
i_scan_dout	output	24x200	0	Scan Clock[2]	-	Scan data output
i_scan_clk	input	1	-	NA	-	**Unused port.**
i_scan_clk_500m	input	1	-	NA	-	**Unused port.**
i_scan_clk_1000m	input	1	-	NA	-	**Unused port.**
i_scan_en	input	1	-	Scan Clock[2]	1	Scan enable
i_scan_mode	input	1	-	Scan Clock[2]	1	Scan mode
i_scan_din	input	24x200	-	Scan Clock[2]	-	
AVALON INTERFACE						
o_cfg_avmm_rdatavld	output	1	0	Avalon domain	-	Asserted to indicate read data is valid.
o_cfg_avmm_rdata	output	32	0x0	Avalon domain	-	Avalon read data bus.
o_cfg_avmm_waitreq	output	1	1	Avalon domain	1	Asserted to indicate that Avalon interface is not ready for access.
i_cfg_avmm_clk	input	1	-	Avalon domain	-	Avalon interface clock.
i_cfg_avmm_rst_n	input	1	-	Avalon domain	0	Avalon interface reset (active low). **IMPORTANT: Reset should be released synchronously with respect to positive edge of i_cfg_avmm_clk . A double-flop reset synchronizer can be used for this purpose.**
i_cfg_avmm_addr	input	17	-	Avalon domain	-	Avalon interface address.

[2] See **Design for Test (DFT) for AIB2.0** to find more information about scan clock.

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
i_cfg_avmm_byte_en	input	4	-	Avalon domain	1	Avalon byte enable.
i_cfg_avmm_read	input	1	-	Avalon domain	1	Avalon read control.
i_cfg_avmm_write	input	1	-	Avalon domain	1	Avalon write control.
i_cfg_avmm_wdata	input	32	-	Avalon domain	-	Avalon write data bus.
SIDEBAND INTERFACE						
ms_tx_transfer_en	output	24	0x0	i_osc_clk (leader)/fs_sr_clk (follower)	1	Indicates that leader has completed its TX path calibration and is ready to receive data.
ms_rx_transfer_en	output	24	0x0	i_osc_clk (leader)/fs_sr_clk (follower)	1	Indicates that leader has completed its RX path calibration and is ready to receive data.
sl_tx_transfer_en	output	24	0x0	i_osc_clk (leader)/fs_sr_clk (follower)	1	Indicates that follower has completed its TX path calibration and is ready to receive data
sl_rx_transfer_en	output	24	0x0	i_osc_clk (leader)/fs_sr_clk (follower)	1	Indicates that follower has completed its RX path calibration and is ready to receive data
sr_ms_tomac	output	1944 (81x24)	0x0	i_osc_clk (leader)/fs_sr_clk (follower)	-	Leader sideband data bus.
sr_sl_tomac	output	1752 (73x24)	0x0	i_osc_clk (leader)/fs_sr_clk (follower)	-	Follower sideband data bus.
i_osc_clk	input	1	-	Oscillator clock domain	-	Free running oscillator clock for leader interface
ms_rx_dcc_dll_lock_req	input	24	-	Oscillator clock domain	1	Initiates calibration of receive path for a leader interface
ms_tx_dcc_dll_lock_req	input	24	-	Oscillator clock domain	1	Initiates calibration of transmit path for a leader interface

Signal Name	Port type	Size	Reset Value	Clock Domain	Active Value	Description
sl_tx_dcc_dll_lock_req	input	24	-	Oscillator clock domain	1	Initiates calibration of receive path for a follower interface
sl_rx_dcc_dll_lock_req	input	24	-	Oscillator clock domain	1	Initiates calibration of transmit path for a follower interface.
sl_external_cntl_26_0	input	648 (24x27)	-	Oscillator clock domain	-	User defined bits 26-0 for slave shift register
sl_external_cntl_30_28	input	72 (24x5)	-	Oscillator clock domain	-	User defined bits 30-28 for slave shift register
sl_external_cntl_57_32	input	624 (24x26)	-	Oscillator clock domain	-	User defined bits 57-32 for slave shift register
ms_external_cntl_4_0	input	120 (24x5)	-	Oscillator clock domain	-	User defined bits 4-0 for master shift register
ms_external_cntl_65_8	input	1392 (24x58)	-	Oscillator clock domain	-	User defined bits 65-8 for master shift register

Table 1: Interface Signals.

3. Clocking

Basically, the AIB PHY has a group of dedicated clock pins per channel which are used for transmitting data and receiving data. The only exceptions are calibration clock, control status register clock (Avalon), scan clock and boundary scan clock (JTAG) which has single ports shared with all the channels. The table below shows the clocks on PHY interface with SoC (for details about microbumps clocks, refers to AIB.20 protocol specification).

Clock	Input/Output	Maximum Frequency [MHz]	Description
m_ns_fwd_clk	Input	2000	Clock for transmitting data from near side to far side.
m_ns_rcv_clk	Input	2000	Receive clock forwarded from the near side to the far side for transmitting data from the far side. This clock is not used internally on PHY to sample data. In Gen2 mode this input is ignored by the PHY.
m_wr_clk	Input	2000	Clock used by AIB PHY for writing transmit data (*data_in_f*) into TX adapter for FIFO mode. The difference between m_wr_clk and m_fs_fwd_clk divided by 1, 2 or 4 according to the used TX FIFO mode shall be 0PPM.
m_rd_clk	Input	2000	Clock used by AIB PHY for reading receive data in RX adapter for FIFO mode. The difference between m_rd_clk and fs_fwd_clk divided by 1, 2 or 4 according to the used RX FIFO mode shall be 0PPM.
m_fs_rcv_clk	Output	2000	Clock received from the far side and converted from quasi-differential. In Gen2 mode this output should be used. This clock is not used internally on PHY to sample data.
m_fs_fwd_clk	Output	2000	Clock received from the far side and converted from quasi-differential.
ns_fwd_clk	Output	2000	Near side forwarded clock derived from m_ns_fwd_clk after DCC/DLL adjustment. It can be used as an SoC implementation option to launch *data_in* in register mode.

Clock	Input/Output	Maximum Frequency [MHz]	Description
ns_fwd_clk_div	Output	2000	Divided near side forwarded clock derived from m_ns_fwd_clk after DCC/DLL adjustment. It can be used as an SoC implementation option to launch *data_in_f* in FIFO mode.
fs_fwd_clk	Output	2000	Clock received from far side after Rx DLL phase adjustment. It can be used as an SoC implementation option to capture *data_out* in register mode.
fs_fwd_clk_div	Output	2000	Divided version of the clock received from far side after Rx DLL phase adjustment. It can be used as an SoC implementation option to capture *data_out_f* in FIFO mode.
i_cfg_avmm_clk	Input	500	Avalon interface clock used for control status register configuration. The same clock port is used on all the channels.
i_osc_clk	Input	1000	Free running oscillator clock for a leader interface used on sideband interface and calibration process.
i_jtag_clkdr	Input	125	Boundary scan clock (JTAG).
i_scan_clk	Input	125	Test clock used on scan process.
i_scan_clk_500m	Input	-	Unimplemented.
i_scan_clk_1000m	Input	-	Unimplemented.

Table 2: Clock pins for SoC interface.

3.1. Tx Clocking

BCA AIB PHY implementation provides a clock input, usually generated by a PLL, used on data transmission (*m_ns_fwd_clk)* and two clocks outputs that can be used on transmit data launch at SoC side (*ns_fwd_clk* and *ns_fwd_clk_div*). n*s_fwd_clk* clock comes from DCC/DLL block in custom implementation where its phase can be adjusted by control register configuration. *fs_fwd_clk* clock *is* recommended to launch data in register mode at SoC level because there is no extra delay added by Tx clock divider logic in its path. Therefore, the time closure in the data path between *SoC data flops* and the flops which sample transmit data in PHY can be easily met.

For FIFO mode, the PHY implementation provides *ns_fwd_clk_div* which is a divided clock derived from DCC/DLL block. The required frequency to launch data from SoC to PHY in FIFO mode can be selected in PHY Tx clock divider through control register according to the used mode (FIFO 1x, FIFO 2x or FIFO 4x). Besides, in FIFO mode, the CDC FIFO implemented on PHY allows that SoC can launch data in a clock frequency up to four times slower than the maximum frequency of PHY RTL implementation (for instance, 500MHz instead of 2GHz) when operating in FIFO 4x.

So that PHY/SoC implementation operates properly, *m_wr_clk* clock used on FIFO mode and SoC clock used for data launch in register mode shall have 0PPM of frequency tolerance with respect to ns_fwd_clk and ns_fwd_clk_div. The figure below shows transmit data path involved in data transfer from SoC to PHY for register mode and FIFO mode.

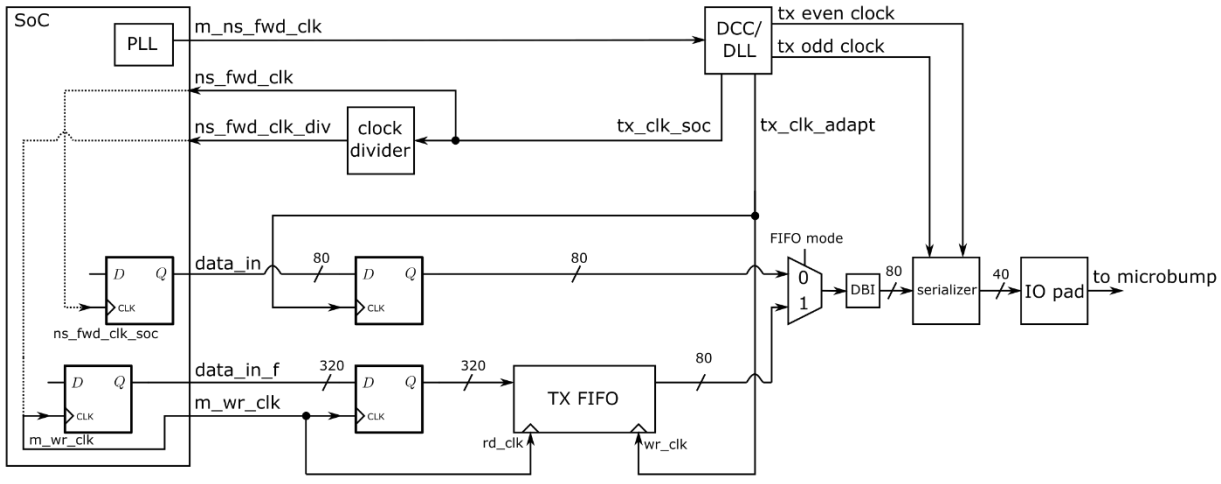


Figure 2: Transmit Channel Clock Diagram

3.2. Rx Clocking

BCA AIB PHY implementation provides two clocks outputs that can be used on receive data capture at SoC side. The first is *fs_fwd_clk* output that comes from Rx DLL block in custom implementation where its phase can be adjusted. The second is *fs_fwd_clk which is* recommended to sample data in register mode in SoC because there is no extra delay added by Rx clock divider logic in its path. Therefore, the time closure in the data path between *data_out* and the flops which sample the data at SoC can be easily met.

For FIFO mode, the PHY implementation provides *fs_fwd_clk_div* which is a divided clock derived from Rx DLL block. The required frequency to sample data in FIFO mode at SoC can be selected in PHY clock divider through control register according to the used mode (FIFO 1x, FIFO 2x or FIFO 4x). Besides, in FIFO mode, the CDC FIFO implemented on PHY allows that SoC can capture the data in a clock frequency up to four times slower than the maximum frequency of PHY RTL implementation (for instance, 500MHz instead of 2GHz) when operating in FIFO 4x.

So that PHY/SoC implementation operates properly, *m_rd_clk* clock used on FIFO mode and SoC clock used for data capture in register mode shall have 0PPM of frequency tolerance with respect to fs_fwd_clk and fs_fwd_clk_div. The figure below shows receive data path involved in data transfer from PHY to SoC for register mode and FIFO mode.

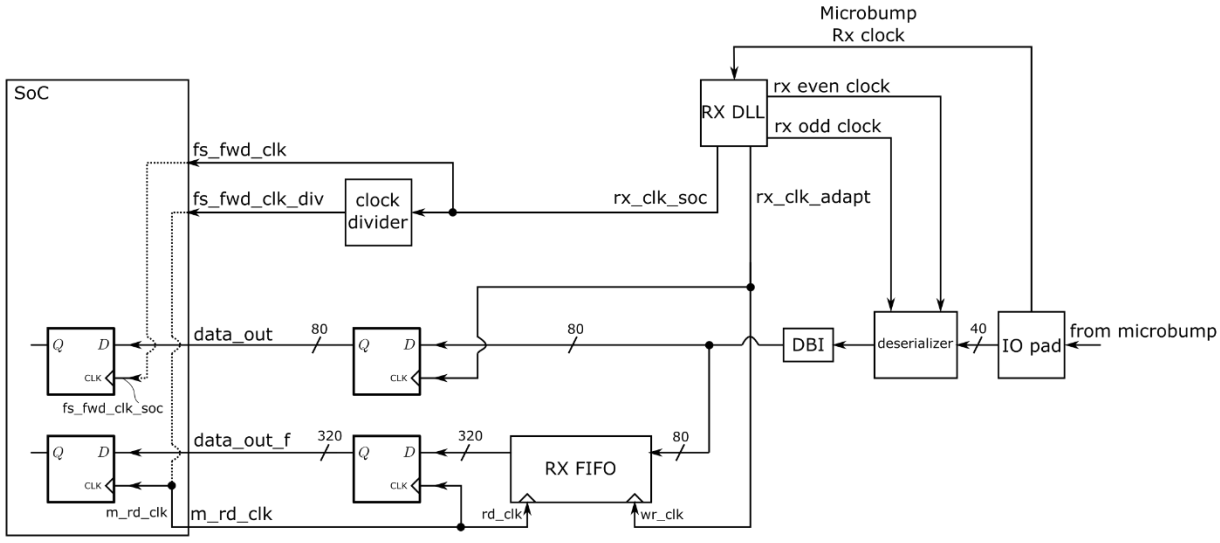


Figure 3: Receive Channel Clock Diagram

3.3. Bonded Channels versus Independent Channels

On transmit side, basically there are two ways of connecting the clocks at SoC level: bonded clock connection and independent clock connection which are shown in the figure below. For independent channels, the SoC has the flexibility to support entirely separate frequency sources for each channel. For bonded channels, the SoC must distribute m_ns_fwd_clk to all PHY channels with sufficiently low jitter. Note that the PHY-level implementation is independent of the scheme implemented by the SoC.

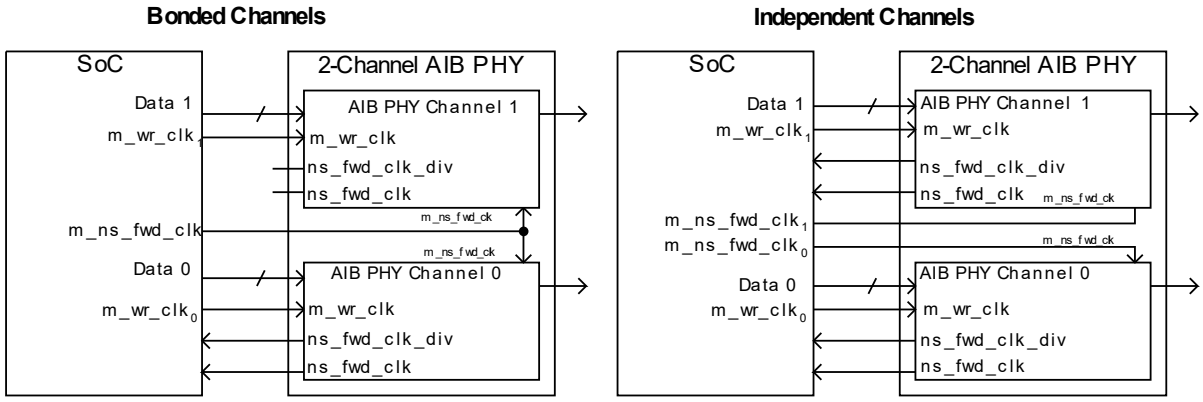


Figure 4: Example of bonded and independent clock connections for 2-channel implementation.

4. Reset

Before starting PHY data transfer through the microbumps, SoC/MAC needs to perform the initialization process which consists of some steps: *Power-on reset synchronization*, *configuration* and *calibration*. See AIB 2.0 protocol specification or BCA AIB PHY functional specification to find more details. The figure below shows the ports in PHY interface and the logic related to the asynchronous reset paths. CSRs implemented on top of AIB PHY are reset by i_cfg_avmm_rst_n. So that metastability problems do not occur, i_cfg_avmm_rst_n input shall be deasserted synchronously with the positive edge of i_cfg_avmm_clk. A double-flop reset synchronizer can be used for this purpose.

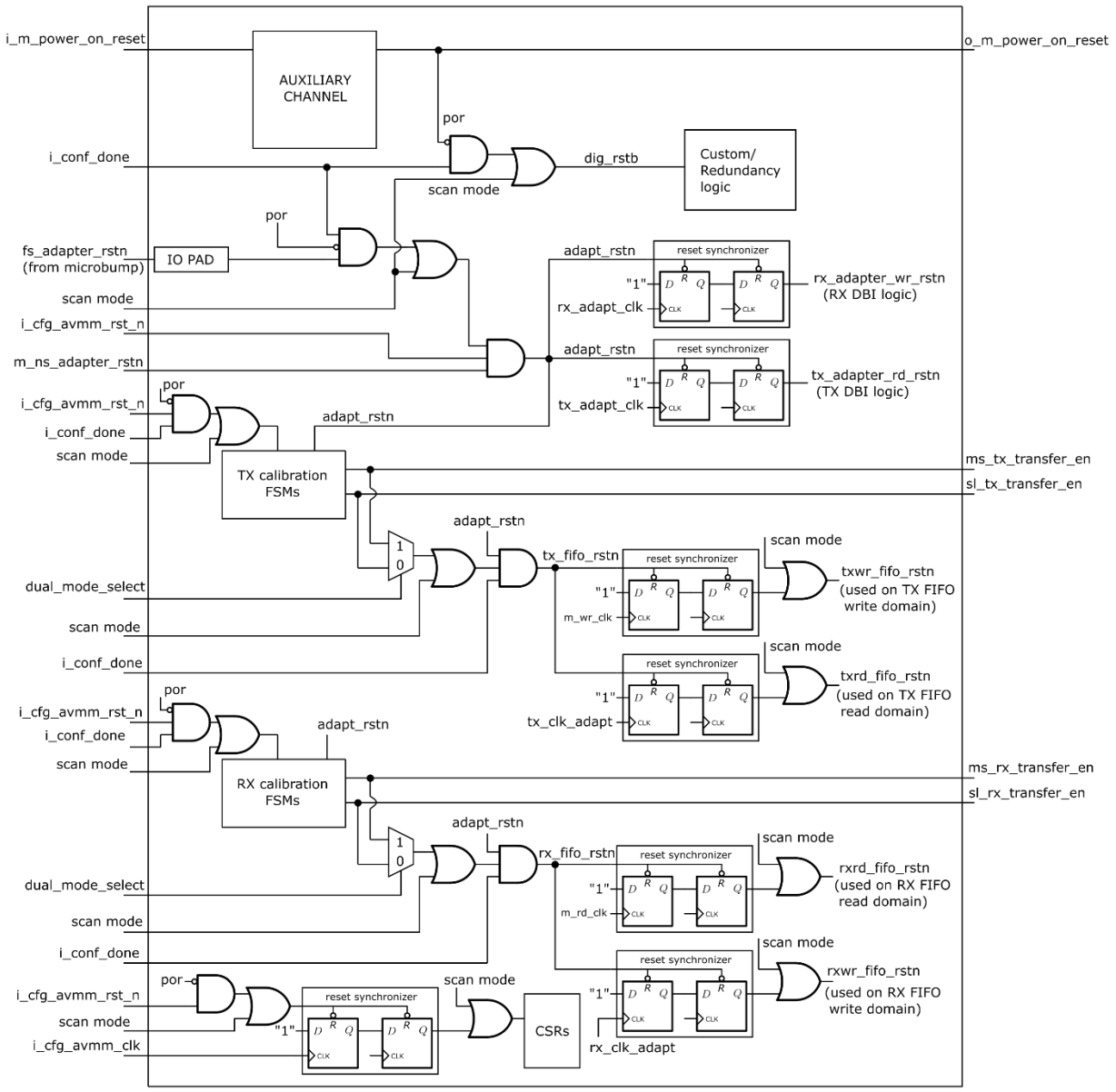


Figure 5: Reset ports and internal logic.

5. RCOM setup

The following registers shall be programed during configuration phase to set up properly pull-up/down drivers in IO pads as part of RCOMP functionality.

Mode	GEN1		
Register name	io_ctrl1		
Field name	txnpu_code[7:0]	txppu_code[7:0]	txnpd_code[7:0]
Config. value	0	213	26
MODE	GEN2		
Register name	io_ctrl1		
Field name	txnpu_code[7:0]	txppu_code[7:0]	txnpd_code[7:0]
	72	0	38

Table 3: RCOMP setup.

6. Input CAP

	Input Cap (fF)	Vs. AIB 2.0 Standard
Gen1	438	Meets spec
Gen2	406	106fF above spec

Table 4 Input Cap Values

7. Unconnected Instances

An unconnected instance should adhere to following guideline to avoid any unexpected behavior of IP:

- All the pads must be unconnected.
- The VDDTX pin must be supplied in order to keep PHY outputs connected to SoC stable and with default value.
- The VDDC1 pin must be supplied in order to keep PHY outputs connected to SoC stable and with default value.
- After the VDDC2 is stable the input to IP must be defined state and IP must be held in reset state to avoid any unwarranted leakage.
- No activity is expected on any input signal. The behavior of output pins is not defined for unconnected instance during any activity on input signals.

8. Connection with AIB 1.0 implementation

When BCA PHY is connected with an AIB 1.0 implementation, PHY unused pins can be left unconnected since the clocks that go to IO cell are gated if m_gen2_mode is tied to logic level zero. This avoids that spurious signals reach the digital logic and saves power. The table below shows the unused IO ports for AIB 1.0.

Unused IO pad	Function Description
iopad_chx_aib[0-19]	Transmit data bit 20-39
iopad_chx_aib[82-101]	Receive data bit 20-39

Table 5: Unused IOs for connection with AIB 1.0 implementation.

9. Design for Test (DFT) for AIB2.0

This section explains DFT signals related hook-up details.

9.1. Digital Logic Scan

- 10 scan chains per channel (Chain length 1000+ flops)
- Hence, a total of 240 scan chains at AIB Top (There are 24 channels).
- 241th chain at AIB-TOP level.
- All 241 chains shall be hooked to EDT present at Chiplet.

Below Table indicates AIB top scan signals to Chiplet level mapping. This can be used as Integration guideline.

Sr. No.	Scan Signal Type	AIB_TOP Level Signal Name	Hookup Requirement at Chiplet level	Pin Activity
1	Scan Input	i_scan_din[240:0]	Connect to EDT decompressor out	Dynamic Input Toggling at <= 100 MHz
2	Scan Output	i_scan_dout[240:0]	Connect to EDT compressor in chiplet level	Dynamic Output Toggling at <= 100 MHz
3	Scan Enable	i_scan_en	Chiplet level Pin	Dynamic Input

				1 – Shift 0 - Capture
4	Scan Reset	i_cfg_avmm_rst_n ns_adapter_rstn[23:0]	Chiplet level Functional Reset	Dynamic Input (will toggle for Reset Capture patterns)
5	Scan Clock	m_rd_clk[23:0] m_wr_clk[23:0] i_cfg_avmm_clk i_osc_clk m_ns_fwd_clk[23:0]	Chiplet level Functional/Scan Clocks (PLL should be Bypassed)	Dynamic Input (Shift clock at 100 or 125 MHz) Only stuck-at planned hence capture is also slow clock from top.
6	Additional Pin Constraints	i_conf_done	Functional conn	Static Input. This goes 0 to 1 during initial power on reset sequence. Then held 1. We do not toggle this during capture.
7		i_scan_mode	This can be glue or top connection.	Static Input constrained to 1.
8		i_m_power_on_reset	This can be glue or top connection.	This is toggled from 0 to 1.
9		m_por_ovrd	This can be glue or top connection.	Static Input constrained to 1.
10		dual_mode_select	This can be glue or top connection.	Static Input constrained to 0.

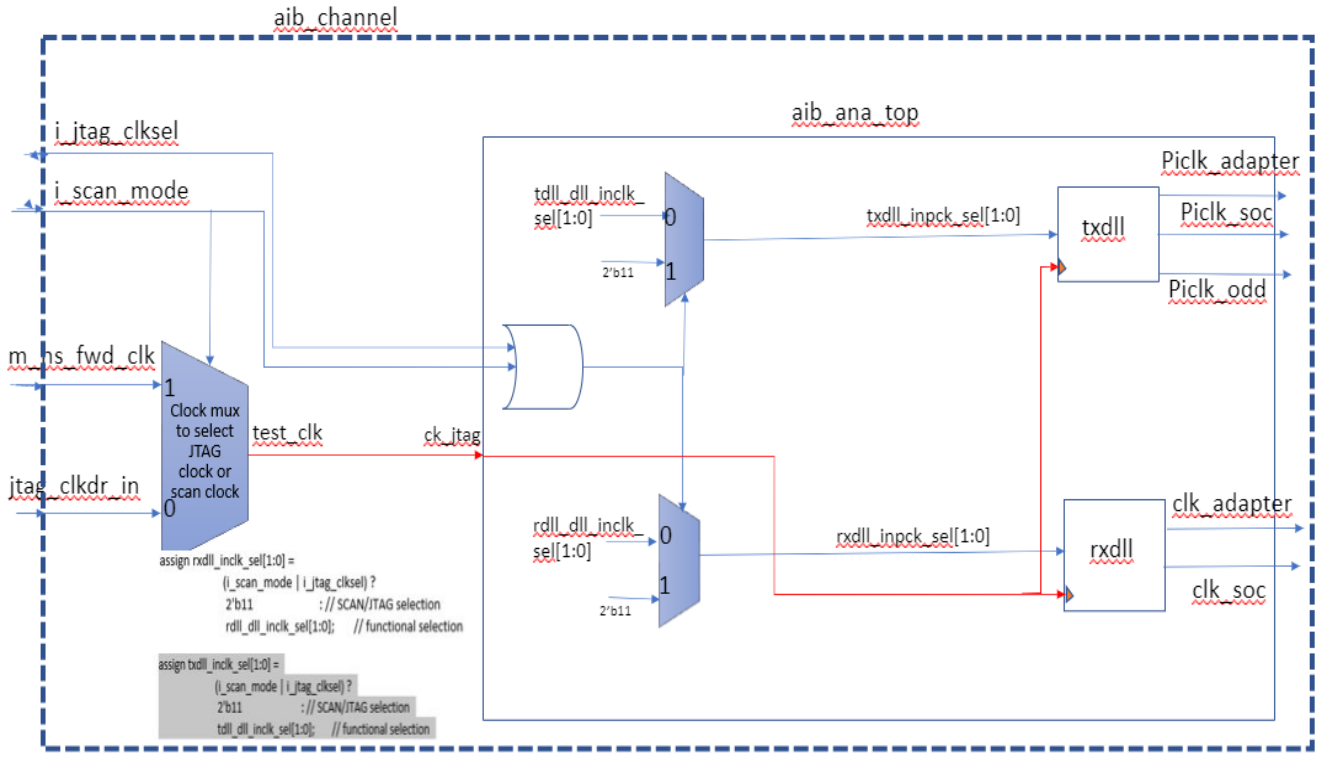
Key Notes:

- AIB HIP (Hard IP) level chain length is ~1056 flops.
- AIB would be sharing EDT controller from Chiplet. Hence overall chain length at Chiplet level would be ~1056.
- Only Stuck-at is planned for AIB HIP.

- Any clocking requirement for Chiplet logic (shift or capture for stuck-at / at-speed) should be met by Chiplet implementation team. No dependency shall be assumed from AIB HIP.
- Please note that there is not a singular scan clock but instead the functional clock inputs as listed in the table above are used for scan when in scan mode. It is up to the SOC integrator to drive either functional or scan clock into these pins. For example a 2:1 mux selecting between functional clock or SOC scan clock, or using the driving PLL in bypass mode.

9.2. DLL clocking in Scan Mode

Below is schematic which indicates how DLL would be working in scan mode. Inpclk_sel[1:0] is set to 2'b11 for both DLLs during scan mode. Also jtag_clkdr_in is muxed with one of the functional clock, which is controlled during scan mode. This muxing ensures that jtag clock is not needed during scan mode. Certain critical input ports need to be maintained stable during scan mode so that DLL output clocks are in phase w.r.to. ck_jtag.



9.3. Boundary Scan (JTAG)

BSCAN Test is initiated through JTAG.

Sr.No.	Signal Type	AIB_TOP Level Signal Name	Hook-up Requirement at Chiplet level	Pin Activity
1	TCK	i_jtag_clkdr	TCK passing through Glue logic based clock gating.	Clock Toggling at <= 25 MHz
2	Private Instr	i_jtag_clksel	This can be Glue logic with Top connection or connected to one of the JTAG DR register bit	Static signal
3	Private Instr	i_jtag_intest	Connected to JTAG DR register bit	Static signal
4	Bscan mode	i_jtag_mode	Connected to IR decoder output	Static signal
5	Reset	i_jtag_rstb	Unused inside AIB TOP	Slow signal
6	Reset Control	i_jtag_rstb_en	Unused inside AIB TOP	Slow signal
7	TDI	i_jtag_tdi	Connect to TDI pin at Chiplet level	Dynamic Input @25MHz
8	ShiftDR	i_jtag_tx_scanen	Connected to ShiftDR of Jtag	Slow signal; FSM output
9	Pull control	i_jtag_weakpdn	Driven by Chiplet TAP	Static signal
10	Pull control	i_jtag_weakpu	Driven by Chiplet TAP	Static signal
11	TDO	o_jtag_tdo	Connect to TDO pin at Chiplet level.	Dynamic Output @25MHz
12	Additional Constraint	i_scan_mode	Driven by Chiplet top pin or glue logic.	Static signal held 0
13	Power on reset	i_m_power_on_reset	Driven by Chiplet top pin or glue logic.	This is toggled from 1 to 0 (active high) for Lead Mode.

		io_pad_power_on_reset	Driven by Chiplet top pin or glue logic.	This is toggled from 1 to 0 (active high) for Follow Mode.
14	Mode select	dual_mode_select	Driven by glue logic or it would Tie logic.	Tie1– Lead Tie0 - Follow
15	Pin Compliance	m_gen2_mode	Drive by glue logic w.r.to j_jtag_mode.	Drive LOW when i_jtag_mode is 1.

Key Notes:

- JTAG is located outside AIB HIP. It is part of Chiplet.
- BSCAN chain is part of each Channel.
- Each Channel has 102 AIB PADs.
- Custom BSCAN is used for each pad.
- Each BSCAN cell used for pad has 3 Shift cells (Only single rank)
- Hence each channel BSCAN shift length is 306.
- Currently we have cascaded BSCAN chain of each channel serially. Hence, AIB_TOP level BSCAN chain length is 7,344.

8.3.1 Custom behavior of AIB2.0 JTAG:

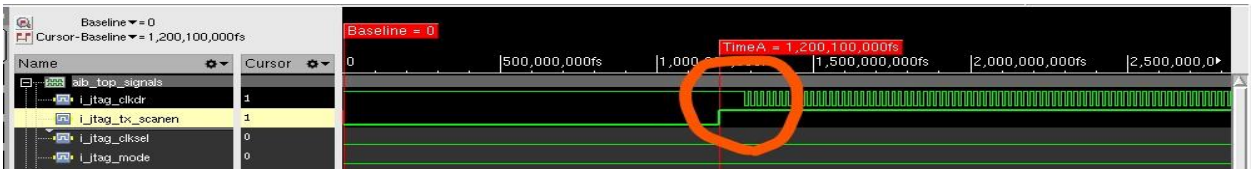
The BSCAN implementation is not as per standard IEEE1149.1 standards. Below are some points which deviate from standard implementation.

Sr. No.	**IEEE1149.1 BSCAN and JTAG**	**AIB2.0 BSCAN and JTAG**
1	Dual Rank BSCAN flops (Shift and Update)	Single rank BSCAN flops (Shift only)
2	Pads are held stable because of update flops (during shift operation).	This is achieved by keeping final mux select (Mode mux) line 0 (mission mode). Functionally logic driving pad is stable and hence pads are stable.
3	TCK free running (It can be gated with update_dr and/or capture_dr)	Here i_jtag_clockdr is gated version of free running TCK (which is driven by Custom TAP).

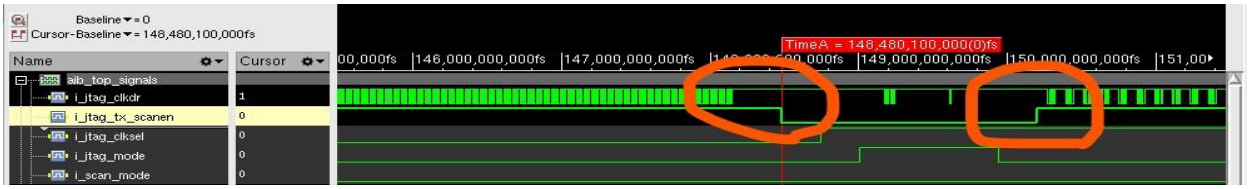
4	Follows IEEE standard FSM.	There is some deviation from standard FSM. Please check the waveforms below to understand custom behavior.
5	TCK is 010 polarity clock (return to 0).	I_jtag_clockdr is 101 polarity clock (return to 1).

i_jtag_tx_scanen corresponds to Shiftdr signal. i_jtag_clockdr is inverse polarity. Also it is not pulsing for some time when i_jtag_tx_scanen is asserted / de-asserted. Please refer waveform snippet 1 and 2.

- **Waveform snippet 1**: Relation between i_jtag_clockdr and i_jtag_tx_scanen

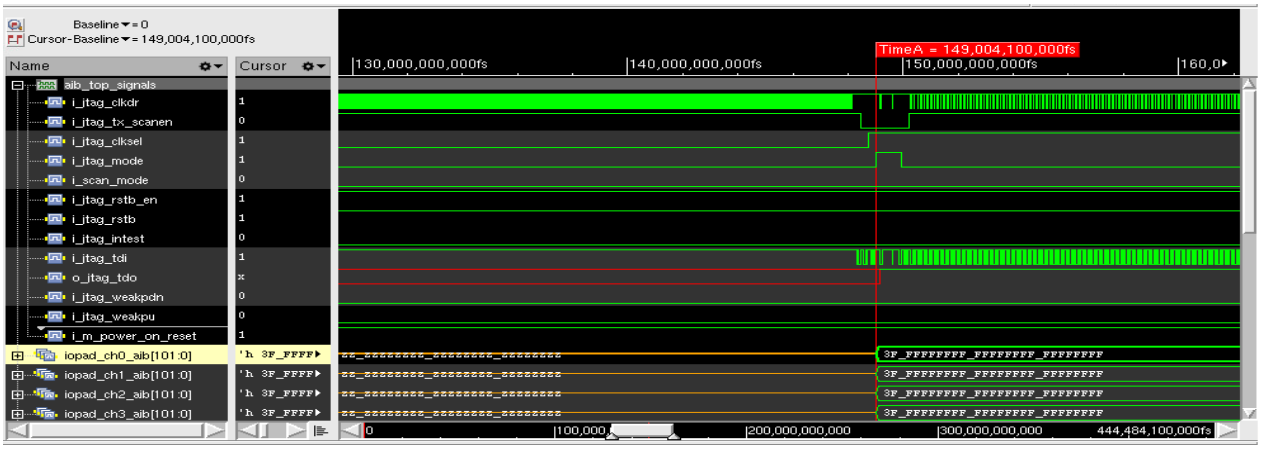


- **Waveform snippet 2**: Relation between i_jtag_clockdr and i_jtag_tx_scanen.



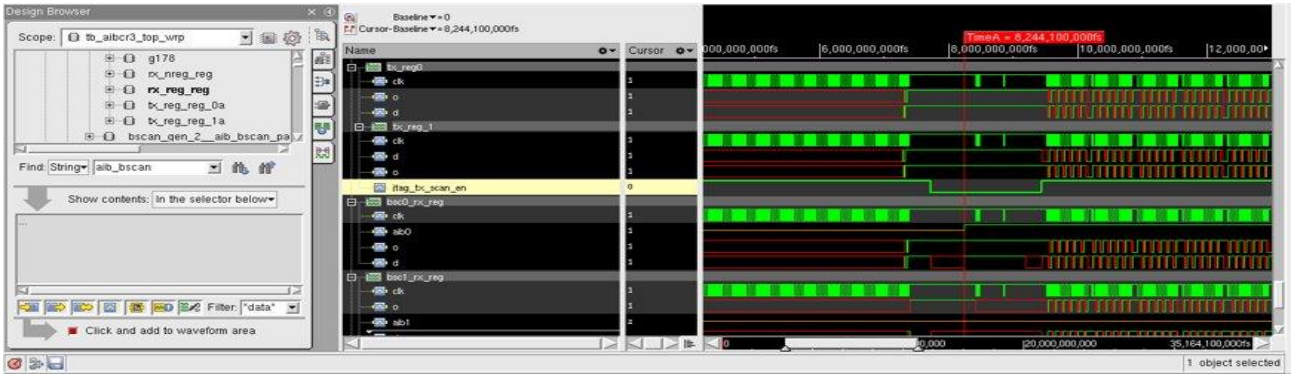
- **Waveform snippet 3**: aibio pads being stable HighZ and getting updated to shift value.

In below snap i_jtag_tx_scanen is 1 during shift and 0 during pad update. i_jtag_mode is mode mux select line which drives frozen shift value to pads.



- **Waveform snippet 4**: rx flop capture related

Here in below waveform aib0 pad is transitioning from Z to 1. while aib1 pad is Z i.e. disabled. The corresponding RX flops capture 1 and X respectively. Check rx_reg0 and rx_reg1.



- **Log snippet 5**: Alternate aibio pads enabled and disabled.

Here alternate pads enabled and disabled (51 enabled and disabled). Sequence “101” corresponds to tx-disable and sequence “111” corresponds to tx enabled and tx pad being driven high. Alternate RX flops receive 1 and X respectively.

```
[         7420100000] Generating segment [         0] = 101
[         7420100000] Generating segment [         1] = 111
[         7420100000] Generating segment [         2] = 101
[         7420100000] Generating segment [         3] = 111
[         7420100000] Generating segment [         4] = 101
[         7420100000] Generating segment [         5] = 111
```

```
enable mode select so that the tx_reg selected instead of idat_adapt.
[         9584100000]DATA COMPARE segment [         0]: received = 10x | expected = 101
[         9704100000]DATA COMPARE segment [         1]: received = 111 | expected = 111
[         9824100000]DATA COMPARE segment [         2]: received = 10x | expected = 101
[         9944100000]DATA COMPARE segment [         3]: received = 111 | expected = 111
[        10064100000]DATA COMPARE segment [         4]: received = 10x | expected = 101
[        10184100000]DATA COMPARE segment [         5]: received = 111 | expected = 111
[        10304100000]DATA COMPARE segment [         6]: received = 10x | expected = 101
```

8.3.2 SoC integration Guideline and Schematic

Few additional points which would help SoC team to integrate AIB2.0 from DFT perspective.

- TAP instructions are one hot in nature. Enable instruction state should be preserved until reset by Disable instruction.
- JTAG_CLKSEL Instruction – It is very similar to other Enable/Disable instruction, nothing special in it. But it does not have corresponding disable instruction. Since we are testing only Async portion of pad during Bscan, this particular instruction has no effect as such.
- Jtag_rstb – is active low signal.
- During specified 16 instructions specified in the document, Tap controller TDO mux should connect jtag_si and jtag_so to TDI and TDO of the chip.
- Shift Clock: The top TCK is gated with shift_dr state so that clock is provided to BSR flops during shift. The shift operation is enabled during Private Instruction "**AIB_SHIFT_EN"** which in turn asserts i_jtag_tx_scanen signal.
- Capture Clock: There is no separate capture-dr signal. "**AIB_TRANSMIT_EN**" shall decide whether clock goes to BSR flops during RX capture. i_jtag_mode signal gets asserted when user programs **AIB_TRANSMIT_EN** private instr which in turn asserts i_jtag_mode signal. Please note that i_jtag_mode indicates Mode mux select line. It is equivalent to Industry Standard Bscan Extest mode.
- Please refer to the schematic below. It indicates the equation for jtag_clkdr w.r.to TCK.
- Update needed: m_gen2_mode port needs to be driven through glue logic.
  assign m_gen2_mode = Functional signal & (~i_jtag_mode);

Below is schematic representation of Key DFT connections between Chiplet and Macro. This diagram may need to be updated for extra signal m_gen2_mode connection.

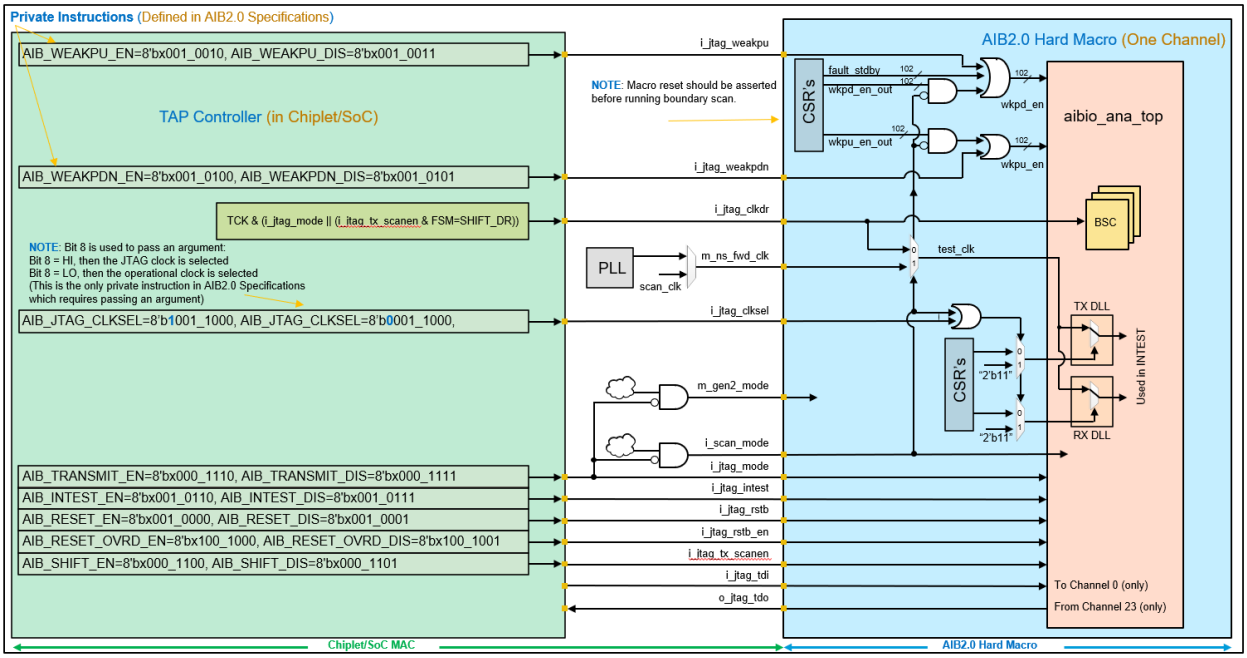


Figure 6 Custom JTAG Instruction

8.3.3 Additional Information about m_gen2_mode

m_gen2_mode pin governs the operation of AIBIO pads.
In GEN1 mode, unused pads are disabled but weal pull-down is not set.
In GEN2 mode, unused pads are disabled and weal pull-down is set.

Please note that Pull has higher precedence than that of pad values. Hence Pull overrides the pad values. Hence, GEN1 mode is preferred for boundary scan Test, where pulls are NOT active.

Gen1 Mode:

PAD	signal name	Status in GEN1 mode
IOPAD0	*tx38*	Pad disabled (weak pull-down is NOT active)
IOPAD1	*tx39*	Pad disabled (weak pull-down is NOT active)
IOPAD2	*tx36*	Pad disabled (weak pull-down is NOT active)
IOPAD3	*tx37*	Pad disabled (weak pull-down is NOT active)
IOPAD4	*tx34*	Pad disabled (weak pull-down is NOT active)
IOPAD5	*tx35*	Pad disabled (weak pull-down is NOT active)

IOPAD6	*tx32*	Pad disabled (weak pull-down is NOT active)
IOPAD7	*tx33*	Pad disabled (weak pull-down is NOT active)
IOPAD8	*tx30*	Pad disabled (weak pull-down is NOT active)
IOPAD9	*tx31*	Pad disabled (weak pull-down is NOT active)
IOPAD10	*tx28*	Pad disabled (weak pull-down is NOT active)
IOPAD11	*tx29*	Pad disabled (weak pull-down is NOT active)
IOPAD12	*tx26*	Pad disabled (weak pull-down is NOT active)
IOPAD13	*tx27*	Pad disabled (weak pull-down is NOT active)
IOPAD14	*tx24*	Pad disabled (weak pull-down is NOT active)
IOPAD15	*tx25*	Pad disabled (weak pull-down is NOT active)
IOPAD16	*tx22*	Pad disabled (weak pull-down is NOT active)
IOPAD17	*tx23*	Pad disabled (weak pull-down is NOT active)
IOPAD18	*tx20*	Pad disabled (weak pull-down is NOT active)
IOPAD19	*tx21*	Pad disabled (weak pull-down is NOT active)
IOPAD82	*rx21*	Pad disabled (weak pull-down is NOT active)
IOPAD83	*rx20*	Pad disabled (weak pull-down is NOT active)
IOPAD84	*rx23*	Pad disabled (weak pull-down is NOT active)
IOPAD85	*rx22*	Pad disabled (weak pull-down is NOT active)
IOPAD86	*rx25*	Pad disabled (weak pull-down is NOT active)
IOPAD87	*rx24*	Pad disabled (weak pull-down is NOT active)
IOPAD88	*rx27*	Pad disabled (weak pull-down is NOT active)
IOPAD89	*rx26*	Pad disabled (weak pull-down is NOT active)
IOPAD90	*rx29*	Pad disabled (weak pull-down is NOT active)
IOPAD91	*rx28*	Pad disabled (weak pull-down is NOT active)
IOPAD92	*rx31*	Pad disabled (weak pull-down is NOT active)
IOPAD93	*rx30*	Pad disabled (weak pull-down is NOT active)
IOPAD94	*rx33*	Pad disabled (weak pull-down is NOT active)
IOPAD95	*rx32*	Pad disabled (weak pull-down is NOT active)

IOPAD96	*rx35*	Pad disabled (weak pull-down is NOT active)
IOPAD97	*rx34*	Pad disabled (weak pull-down is NOT active)
IOPAD98	*rx37*	Pad disabled (weak pull-down is NOT active)
IOPAD99	*rx36*	Pad disabled (weak pull-down is NOT active)
IOPAD100	*rx39*	Pad disabled (weak pull-down is NOT active)
IOPAD101	*rx38*	Pad disabled (weak pull-down is NOT active)

Gen2 Mode:

PAD	signal name	Status in GEN2 mode
IOPAD42	*ns_rcv_clk*	pad disabled and weak pull-down active
IOPAD43	*ns_rcv_clk_b*	pad disabled and weak pull-down active
IOPAD45	*ns_sr_clk_b*	pad disabled and weak pull-down active
IOPAD56	*fs_sr_clk_b*	pad disabled and weak pull-down active
IOPAD58	*fs_rcv_clk_b*	pad disabled and weak pull-down active
IOPAD59	*fs_rcv_clk*	pad disabled and weak pull-down active

One experimental run given to check failures in Gen2 mode. Stuck-at 1 failed for certain set of pads which has pull down enabled.

```
55493 445254100000: tb_aibcr3_top_wrp.Finish: finishing simulation..
55494
55495 ////////////////////////////////////////////////////////////////////////////////
55496 445264100000: Simulation ended, ERROR count: 192
55497 ////////////////////////////////////////////////////////////////////////////////
55498
55499 Simulation complete via $finish(1) at time 445264100 PS + 0
55500 ./vectors/weakpu_TOP_Edited.sv:341        $finish;
55501 xcelium> exit
55502 TOOL:   xrun(64)       21.03-s007: Exiting on Jul 27, 2022 at 16:28:12 IST  (total: 00:04:23)
```

- 6 pads are disabled as per Table shown above for Gen2 mode.
- 2 spare pads (iopad 50 and iopad 51)
- Total 8 pads Fail only for stuck-at 1
- 8x24=192 errors shown in log file

9.4. NTL Test

This is functional mode Test.

Key Points:

- Configure the hardware in nearside loopback mode (Custom portion of RX Datapath)
- Enabling the ntl_en signal through the Avalon interface by configuring the aib_ntl1[15] register.
    - 1 – enable NTL FSM, 0 – disable NTL FSM
- Configure the pad_num (aib_ntl1[6:0]) to select the PAD number, on which to check the NTL test.
- The NTL FSM will check for the condition on the selected PAD's receive data (rx_data_even/ rx_data_odd) to get the data transition from High Impedance(z) to 1 to 0.
- When the data is sampling from 1 to 0, the counter will start incrementing w.r.to the sys_clk (250MHz). Once the data is reached to 0, then the counter is latched to a value.
- Hence, the ntl_done signal is asserted. The counter value and ntl_done signal will be monitored through the AVMM interface.

Note:

- Please refer the **AIB_Gen2_PHY_Functional_Spec** to know information about the registers.

Signal Type	**AIB_TOP Level Signal Name**	**Hook-up Requirement at Chiplet level**	**Pin Activity**
Clock	m_ns_fwd_clk	Derived from m_ns_fwd_clk, Operating at 250MHz	2GHz clock. Internally. Interface running @250MHz
Reset	i_cfg_avmm_rst_n, o_m_power_on_reset	Derived from i_cfg_avmm_rst_n & o_m_power_on_reset	Active-low reset, slow signal

Data_in	i_cfg_avmm_wdata	Connect to Chiplet level interface pins	Data path closed at 500MHz, but driven @100 MHz for Production Test.
Address	i_cfg_avmm_addr	Connect to Chiplet level interface pins	Address path is closed @ 500MHz, but driven @100 MHz for Production Test.
Read Access	i_cfg_avmm_read	Connect to Chiplet level interface pins	Active high, slow signal
Write Access	i_cfg_avmm_write	Connect to Chiplet level interface pins	Active high, slow signal
Data_out	o_cfg_avmm_rdata	Connect to Chiplet level interface pins	Data path closed at @500MHz but For Production Test being shifted @100MHz.

9.5. Manufacturing Test

Manufacturing Test list is being tracked in separate sheet AIB2.0-MFG-Test-List.xlsx

10. Timing

The deliverables include liberty files for the phy top as well as an example timing sdc (soc_integration.sdc).

Please use this sdc as a reference point for creating the needed constraints. The exact settings will depend on the configuration you are using. Refer to the **AIB_Gen2_PHY_Functional_Spec** for information about the different modes and clock relations.

10.1. Use of ns_fwd_clk_div and fs_fwd_clk_div

In register mode these clock pins should not be used. Only the non-divided version of these clocks should be used in register mode.

In FIFO mode these clocks must be created at the pin of the phy as in the example sdc file. They should not be created as a generated clock as this will introduce an incorrect clock latency.

10.2. Liberty warning

In Genus we see two classes of warnings when reading in the liberty file. These are LBR-126 (Warning : Found a combinational arc in a sequential cell.) Similar warning may be seen in other tools. This warning can be ignored, for further details see below.

9.2.1 LBR-126

These are from output pins driven by clock divider circuits. These circuits are combinational for 1x division but appear sequential for 2x and 4x division. This causes what appears to be a conflict. Following section 9.1 these pins are only used with created clocks and the liberty file pin definition is ignored.

11. Power

Internal to the phy there are microbumps for VDDTX, VDDC1 and VSS. In addition to these there are power pins along the SoC side of the phy. All of these power pins should be connected to SoC side power nets.

VDDC2 is only connected by the power pins on the SoC side. VDDC2 should be connected to the core power rail of the SoC.

It is expected that there will be C4 bumps for each power rail outside of the phy itself.

The liberty files are not annotated with power values. Please see the following table for power and current draw values.

12. Layout

The AIB PHY includes an ESD power clamps for each power rail. It is recommended that clamps are added outside the AIB PHY connected to any C4 bumps that are within the SoC.

DIC cells are not included on AIB PHY hard macro, therefore SoC implementation should use DIC cells in the design to meet Intel 22 design rules.

Two signals (iopad_power_on_reset and iopad_device_detect) should be connected directly from a C4 bump to the GMB pin of the phy.

FULL MACRO SIZE	
Width [µm]	914.112

Height [µm]	8026.2

Table 6: PHY macro size.

DISTANCE FROM MICROBUMP CENTER TO DIE EDGE	
Center of microbump to left edge [µm]	77.122
Center of microbump to right edge [µm]	29.48
Center of microbump to top edge [µm]	15.96
Center of microbump to bottom edge [µm]	15.96

Table 7: Distance from microbump center to die edge.

METAL LAYER	
Power Pins	Metal8/GM0/GMB
Signal Pins	Metal6/Metal8
iopad_power_on_reset / iopad_device_detect	GMB

Table 8: Metal layer.

13. Revision History

Version	Dates	Comments
1	Jan 10th, 2022	Initial version.
5	Jul 14th, 2022	Updated Table 1 “Interface Signals” to include active value for interface signals.
	Jul 18th, 2022	Added section 7.3.2 “SoC integration Guideline Schematic”
	Jul 19th, 2022	Added section 7.2 “DLL clocking in Scan Mode”
	Jul 26th, 2022	Edited section 7.3 Table i_jtag_clkdr hook up requirement. Also added point 13,14,15
	Jul 26th, 2022	Updated schematic diagram in section 7.3.2.
	Jul 28th, 2022	Added new section 7.3.3: “Additional Information about m_gen2_mode”.

Version	Dates	Comments
6	Sep 16th, 2022	Added information about DIC cells and supply clamps Added information about AVMM asynchronous reset requirements.
7	Sept 20th, 2022	Added timing information. Added power information. Updated layout section with pin layers and notes about connection. Removed links to other documents.
8	Sept 26th, 2022	Added Rcomp values
9	Sept 27th, 2022	Updated clock domain and reset value columns of Table 9: Interface Signals.
9p1	Oct 12th , 2022	Corrected Dual mode select pin values in BSCAN related table in section 8.3. i_jtag_rstb and i_jtag_rstb_en signals are unused inside AIB TOP IP. No hook-up requirement from Chiplet side.
9p2	Oct 19th, 2022	Added section for liberty warnings. Added information about the use of scan clocks.
9p3	Oct 20th, 2022	Modified liberty warning section to remove reference to LBR-166
9p4	November 4th, 2022	Added input cap table. Updated custom JTAG picture. Fixed dual mode select typo
9p5	March 6th, 2024	Updated for Open Source release