Stratix 10 Chiplet Advanced Interface Bus (AIB) Profile and Usage Note

Authors

David Kehlet
Research Scientist

Tim Hoang
Principal Engineer

Intel® Programmable Solutions Group

1 Introduction

To interface with Stratix 10 a chiplet must implement a profile of AIB capabilities, conform to specific signal assignments and mechanical requirements, and meet additional requirements beyond the AIB specification. This document identifies the requirements for a chiplet to interoperate with Stratix 10; a chiplet that interoperates is referred to here as a Stratix 10 Chiplet or S10 Chiplet.

The **Advanced Interface Bus (AIB) Specification** [1] document, also known as the AIB Spec, is the source of requirement references.

2 S10 Chiplet AIB Standard, Mode and Channels

A S10 Chiplet shall implement the AIB Plus requirements of the AIB Spec. Additionally, a S10 Chiplet shall use:

- S10 Chiplet AIB signal to bump assignments in the tables below.
- S10 Chiplet AIB bump locations in the spreadsheet of section 5.1.
- IO voltage of 0.75V to 0.97V, supplied by Stratix 10 as described below.
- AIB Master mode for reset and shift register function.

3 S10 Chiplet AIB Features

3.1 S10 Chiplet AIB IOs

S10 Chiplet AIB IOs shall support Asynchronous mode, SDR from 0.1Gbps to 1Gbps, and DDR from 0.2Gbps to 2Gbps.

DLL and DCC per channel are strongly recommended for DDR above 800Mbps.

Redundancy is not required or recommended for a S10 Chiplet. Stratix 10 supports redundancy in a legacy (non-standard) mode and Intel manufacturing yield per AIB interface is over 99% without redundancy.

3.2 AIB Interface Signals

A S10 Chiplet shall implement the signals in Table 1 in each AIB channel. Near side refers to the S10 Chiplet; far side refers to the Stratix 10 FPGA main die.

Table 1. S10 Chiplet Signals in Each AIB Channel

Signal	Description
tx[19:0]	Synchronous data transmitted from the near side.
ns_fwd_clk/ns_fwd_clkb	Near-side transfer clock, forwarded from the near side to the far side for capturing received data. Used by the far side to capture the near side's TX signals.
ns_fwd_div2_clk/ns_fwd_div2_clkb[1]	ns_fwd_clk divided by 2
ns_rcv_clk/ns_rcv_clkb	Receive-domain clock forwarded from the near side to the far side for transmitting data from the far side. Far side uses this to produce fs_fwd_clk.
ns_rcv_div2_clk/ns_rcv_div2_clkb[1]	ns_rcv_clk divided by 2
rx[19:0]	Synchronous data received from the far side.
fs_fwd_clk/fs_fwd_clkb	Far-side transfer clock, forwarded from the far side to the near side for capturing received data. Used by the near side to capture RX signals.
fs_rcv_clk/fs_rcv_clkb[3]	Receive-domain clock forwarded from the far side to the near side for transmitting data to the far side. Near side uses this to produce ns_fwd_clk.
ns_sr_data	Time-multiplexed sideband-control data from near side to far side.
ns_sr_clk/ns_sr_clkb	Forwarded serial shift register clock from near side to far side chiplet, driven by free running clock.
ns_sr_load	Sideband control load signal from near side to far side.
fs_sr_data	Time-multiplexed sideband-control data from far side to near side.
fs_sr_clk/fs_sr_clkb	Forwarded serial shift register clock from far side to near side chiplet, driven by free running clock.
fs_sr_load	Sideband control load control signal from far side to near side.
fs_mac_rdy	Data-transfer-ready signal from far side to near side.
ns_mac_rdy	Data-transfer-ready signal from near side to far side.
fs_adapter_rstn	Asynchronous adapter reset signal from far side to near side.

1. These clocks are not in the AIB Spec and are not required, however an alternate scheme of providing half rate clocks may be required if these clocks are not used. See section 3.2.1. The DCD specifications of these clocks are the same as their source clocks.
2. The AIB Spec's ns_adapter_rstn is not used by an S10 Chiplet and that bump should be asserted HI on the S10 chiplet.
3. The AIB spec allows for fs_rcv_clk/fs_rcv_clkb to be sent by Stratix 10 to the S10 chiplet, with Stratix 10's limit on this clock expected to be 500MHz. However, this use is not recommended.
4. See Table 3 for the configuration of all other S10 Chiplet IO bumps not listed in Table 1.

3.2.1 Half Rate (Divided by 2) Clocks

The Stratix 10 core needs clocks to be 500MHz or less, and the AIB spec's transfer and receive clocks can easily exceed 500MHz. Stratix 10 has the capability to use a half rate clock from the same reference (0 PPM difference) as the related transfer or receive clock.

Stratix 10 can be used to generate the half rate clock from the common reference using a Stratix 10 internal PLL. That common reference may enter Stratix 10 through standard clock inputs. Alternatively, the method used by Intel is to have a S10 chiplet supply the half rate clocks through specific AIB bumps. Table 1 contains divided-by-2 clocks (see note 1) that come from the S10 chiplet to the Stratix 10 for use by the Stratix 10 core.

3.2.2 Typical Clock and Data Configuration

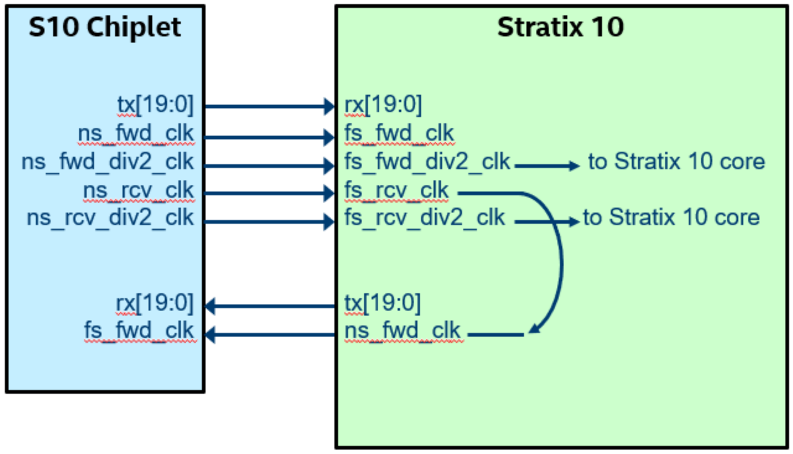

Figure 1. Typical Clock and Data Configuration

3.3 S10 Chiplet AIB Adapter

3.3.1 Word Marking and Word Alignment

Stratix 10 requires and outputs word marking to operate on 80bit quantities at half rate of the AIB clock (example: 2Gbps = double data rate, 1GHz = full rate, 500MHz = half rate). Word marking identifies the upper 40bits uniquely from the lower 40bits, facilitating demultiplexing into an 80bit half rate word. In a 80bit half rate data word, bits 79 and 39 are used for word marking. Figure 2 shows how bits 79 and 39 are marked and how the 78bit user data quantity from the MAC to AIB is sent over the AIB's 20bit wide *tx[19:0]* signals. Figure 2 and Figure 3 use the AIB Open Source's convention for bit numbering as d[77:0] at the MAC.

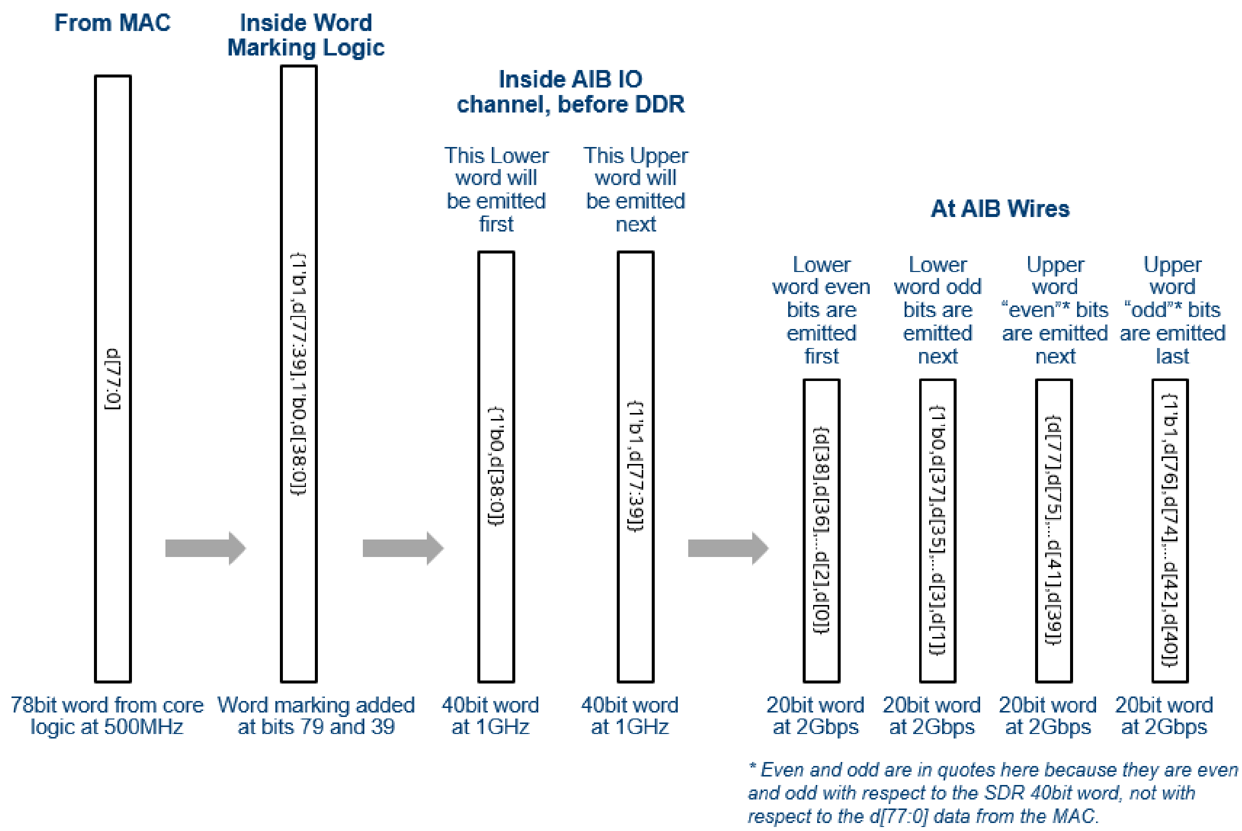

Figure 2. 2x Mode and Upper Word Marking

DDR to SDR conversion knows which 20bit word is even or odd by the edge of the clock (even is valid before the rising edge, odd valid before the falling edge). By examining the word marking bits, the receiving chiplet can reconstruct the 80bit word from the '0' marked lower 40bit word and the '1' marked upper 40bit word. This is shown in Figure 3.

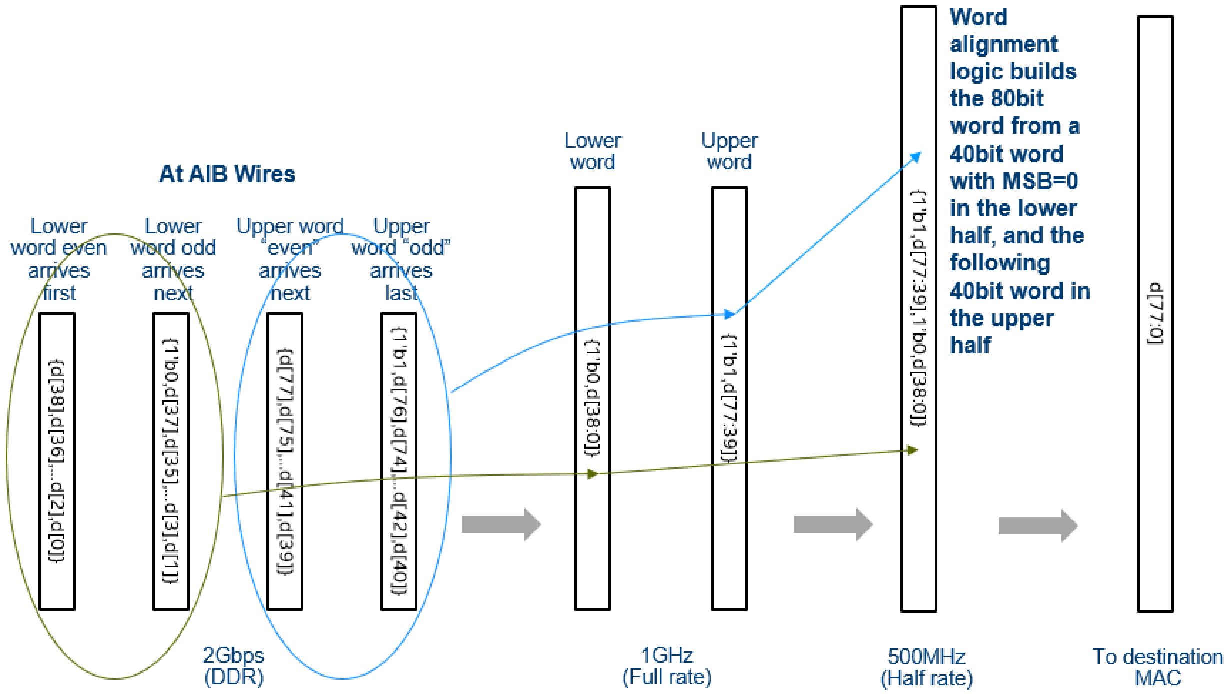

Figure 3. Reassembling Data Word in 2x Mode

3.3.2 Clocking

A S10 chiplet may implement the simple adapter retiming register as defined in the AIB Spec. A significant chiplet design is likely to need a phase compensation FIFO to pass data from the AIB into the chiplet's core, and another phase comp FIFO from the chiplet's core to AIB IO for data going in the chiplet to Stratix 10 direction.

3.3.3 Channel Alignment across Multiple AIB Channels

If the S10 chiplet sends or receives a data word that is spread over more than one AIB channel, then the chiplet and Stratix 10 need to engage in a channel alignment procedure. Once each channel's data is resynchronized to a common clock domain, the skew between channels may cause data to arrive on different cycles of the common clock. Channel alignment in Stratix 10 is performed by soft IP, therefore any channel alignment scheme between the chiplet and Stratix 10 may be implemented.

A useful channel alignment scheme with Stratix 10 dedicates a bit out of an 80bit word for each channel to be aligned; this bit is called the strobe bit. At the transmitter the strobe bit is set to 1 in each 80bit word across all channels to be aligned, then for the next several 80bit words the strobe bit is set to 0. The cycle repeats every N words. Logic at the receiving side can determine alignment by watching the arrival of the words with the strobe bit set to 1 bit set in the receiver's clock domain. Skew between channels is factored out by recording the relative cycle difference between channels and selecting the correct channel word to assemble into a larger bus. Figure 4 is an example of channel alignment using a strobe bit and using FIFOs on the receiving chiplet.

The value of N in the previous paragraph must be greater than the worst skew in 80bit cycles across all channels, and greater than the depth of the alignment FIFO. Typically N is guard banded heavily for N=16 (repeat every 16 80bit cycles) or N=32 (repeat every 32 80bit cycles).

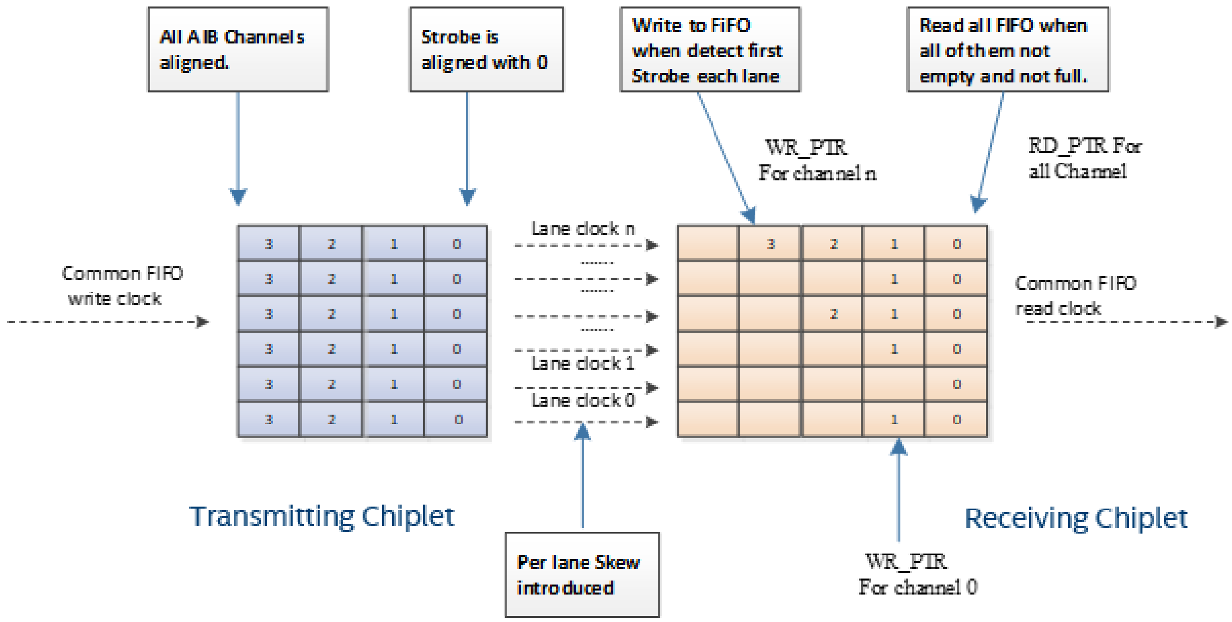

Figure 4. Channel Alignment

3.3.4 Serial Shift Chains and Reset

The S10 chiplet shall implement the Master serial shift chains and reset functions.

4 AIB Configuration and Control

You should implement the AIB configuration as a register file per channel to control input, output, DDR/SDR and other features. A fixed function may instead hardwire the feature selection, but register control provides more debugging capability.

AIB Physical Design

4.1 Stratix 10 EMIB

A S10 chiplet shall use the Stratix 10 EMIB that connects to the West side of Stratix 10. This defines a S10 chiplet as having AIB on its East side.

4.2 Data Channels

4.2.1 Divide by 2 Clocks

A S10 chiplet uses the Alternate (Master) Bump Map of the AIB specification. The additional divided by 2 clocks are located as shown in Table 2, and with all the bumps in Table 3.

Table 2. Divide by 2 Clock Bump Assignments

Bump ID	S10 Chiplet Bump Name	S10 Chiplet IO	Stratix 10 Bump Name	Stratix 10 IO
AIB53	ns_fwd_div2_clk	out	fs_fwd_div2_clk	in
AIB54	ns_fwd_div2_clkb	out	fs_fwd_div2_clkb	in
AIB48	ns_rcv_div2_clk	out	fs_rcv_div2_clk	in
AIB55	ns_rcv_div2_clkb	out	fs_rcv_div2_clkb	in

4.2.2 S10 Chiplet IO Input/Output Configuration

You should use the existing S10 Chiplet IO configuration of the AIB open source located at https://github.com/intel/aib-phy-hardware in your design. Since Stratix 10 has pins unused by the S10 Chiplet that are configured as input or output, the S10 chiplet needs to make sure it does not cause driver conflict or allow a Stratix 10 input to float. Table 3 summarizes the existing S10 Chiplet IO configuration. It is the same as the AIB Spec except for divide by 2 clocks as described previously and for the in/out assignment of S10 Chiplet unused pins. Changes from the AIB Spec are in bold.

Table 3. S10 Chiplet Bump Table with IO Configuration as Used by AIB Open Source

Bump ID	Bump Name	IO	Bump ID	Bump Name	IO
AIB61	unused_AIB61	**in**	AIB50	unused_AIB50[1]	**out**
AIB72	unused_AIB72	**in**	AIB73	unused_AIB73	**in**
AIB75	unused_AIB75	**out**	AIB74	unused_AIB74	**in**
AIB91	unused_AIB91	**out**	AIB90	unused_AIB90	**out**
AIB95	ns_sr_data	out	AIB94	ns_sr_load	out
AIB85	ns_sr_clk	out	AIB84	ns_sr_clkb	out
AIB76	unused_AIB76	**out**	AIB77	unused_AIB77	**out**
AIB58	unused_AIB58	**in**	AIB63	unused_AIB63	**in**
AIB48	**ns_rcv_div2_clk**	**out**	AIB55	**ns_rcv_div2_clkb**	**out**
AIB62	unused_AIB62	**out**	AIB60	unused_AIB60	**out**
AIB53	**ns_fwd_div2_clk**	**out**	AIB54	**ns_fwd_div2_clkb**	**out**
AIB49	ns_mac_rdy	out	AIB56	ns_adapter_rstn	out
AIB51	unused_AIB51	**out**	AIB52	unused_AIB52	**out**
AIB57	fs_rcv_clk	in	AIB59	fs_rcv_clkb	in
AIB64	unused_AIB64	**in**	AIB65	fs_adapter_rstn	in
AIB80	unused_AIB80	**in**	AIB81	unused_AIB81	**in**
AIB78	unused_AIB78	**in**	AIB79	unused_AIB79	**in**
AIB87	ns_rcv_clk	out	AIB86	ns_rcv_clkb	out
AIB83	fs_sr_clk	in	AIB82	fs_sr_clkb	in
AIB89	unused_AIB89	**in**	AIB88	unused_AIB88	**in**
AIB93	fs_sr_data	in	AIB92	fs_sr_load	in
AIB71	unused_AIB71	**out**	AIB70	unused_AIB70	**out**
AIB68	unused_AIB68	**out**	AIB69	unused_AIB69	**out**
AIB66	unused_AIB66	**out**	AIB67	unused_AIB67	**in**
AIB20	RX[0]	in	AIB21	RX[1]	in
AIB22	RX[2]	in	AIB23	RX[3]	in
AIB24	RX[4]	in	AIB25	RX[5]	in
AIB26	RX[6]	in	AIB27	RX[7]	in
AIB28	RX[8]	in	AIB29	RX[9]	in
AIB43	fs_fwd_clk	in	AIB42	fs_fwd_clkb	in
AIB30	RX[10]	in	AIB31	RX[11]	in
AIB32	RX[12]	in	AIB33	RX[13]	in
AIB34	RX[14]	in	AIB35	RX[15]	in
AIB36	RX[16]	in	AIB37	RX[17]	in
AIB38	RX[18]	in	AIB39	RX[19]	in
AIB44	fs_mac_rdy	in	AIB45	unused_AIB45	**in**
AIB18	TX[18]	out	AIB19	TX[19]	out
AIB16	TX[16]	out	AIB17	TX[17]	out
AIB14	TX[14]	out	AIB15	TX[15]	out
AIB12	TX[12]	out	AIB13	TX[13]	out
AIB10	TX[10]	out	AIB11	TX[11]	out
AIB41	ns_fwd_clk	out	AIB40	ns_fwd_clkb	out

Bump ID	Bump Name	IO	Bump ID	Bump Name	IO
AIB8	TX[8]	out	AIB9	TX[9]	out
AIB6	TX[6]	out	AIB7	TX[7]	out
AIB4	TX[4]	out	AIB5	TX[5]	out
AIB2	TX[2]	out	AIB3	TX[3]	out
AIB0	TX[0]	out	AIB1	TX[1]	out
AIB46	unused_AIB46	**out**	AIB47	unused_AIB47	**out**

1. Unused outputs should be set by the S10 Chiplet to 0.

4.3 AUX Channel

For the AUX channel, a S10 chiplet uses the bump assignment in Table 4.

Table 4. S10 Chiplet AUX Channel Bump Table with IO Configuration as Used by AIB Open Source

Bump ID	Bump Name	S10 Chiplet Full AUX IO	Alternate No AUX Package Connection to Stratix 10
AIB0	unused_AIB0	out	VSSP[1]
AIB1	unused_AIB1	out	VSSP
AIB10	unused_AIB10	out	VSSP
AIB11	unused_AIB11	out	VSSP
AIB12	unused_AIB12	out	VSSP
AIB13	unused_AIB13	out	VSSP
AIB14	unused_AIB14	out	VSSP
AIB15	unused_AIB15	out	VSSP
AIB16	unused_AIB16	out	VSSP
AIB17	unused_AIB17	out	VSSP
AIB18	unused_AIB18	out	VSSP
AIB19	unused_AIB19	out	VSSP
AIB2	unused_AIB2	out	VSSP
AIB20	unused_AIB20	out	VSSP
AIB21	unused_AIB21	out	VSSP
AIB22	unused_AIB22	out	VSSP
AIB23	unused_AIB23	out	VSSP
AIB24	unused_AIB24	no connect	no connect
AIB25	unused_AIB25	in	no connect
AIB26	unused_AIB26	out	VSSP
AIB27	unused_AIB27	in	no connect
AIB28	unused_AIB28	in	no connect
AIB29	unused_AIB29	in	no connect
AIB3	unused_AIB3	out	VSSP
AIB30	unused_AIB30	in	no connect
AIB31	unused_AIB31	in	no connect
AIB32	unused_AIB32	in	no connect
AIB33	unused_AIB33	in	no connect
AIB34	unused_AIB34	in	no connect
AIB35	unused_AIB35	in	no connect
AIB36	unused_AIB36	in	no connect
AIB37	unused_AIB37	in	no connect
AIB38	unused_AIB38	in	no connect

AIB39	unused_AIB39	in	no connect
AIB4	unused_AIB4	out	VSSP
AIB40	unused_AIB40	in	no connect
AIB41	unused_AIB41	out	VSSP
AIB42	unused_AIB42	out	VSSP
AIB43	unused_AIB43	out	VSSP
AIB44	unused_AIB44	out	VSSP
AIB45	unused_AIB45	in	no connect
AIB46	unused_AIB46	in	no connect
AIB47	unused_AIB47	in	no connect
AIB48	unused_AIB48	in	no connect
AIB49	unused_AIB49	in	no connect
AIB5	unused_AIB5	out	VSSP
AIB50	unused_AIB50	in	no connect
AIB51	unused_AIB51	in	no connect
AIB52	unused_AIB52	in	no connect
AIB53	unused_AIB53	in	no connect
AIB54	unused_AIB54	in	no connect
AIB55	unused_AIB55	in	no connect
AIB56	unused_AIB56	out	VSSP
AIB57	unused_AIB57	in	no connect
AIB58	unused_AIB58	out	VSSP
AIB59	unused_AIB59	in	no connect
AIB6	unused_AIB6	out	VSSP
AIB60	unused_AIB60	out	VSSP
AIB61	unused_AIB61	out	VSSP
AIB62	unused_AIB62	out	VSSP
AIB63	unused_AIB63	out	VSSP
AIB64	unused_AIB64	out	VSSP
AIB65	unused_AIB65	out	VSSP
AIB66	unused_AIB66	out	VSSP
AIB67	unused_AIB67	out	VSSP
AIB68	unused_AIB68	out	VSSP
AIB69	unused_AIB69	out	VSSP
AIB7	unused_AIB7	out	VSSP
AIB70	unused_AIB70	out	VSSP
AIB71	unused_AIB71	out	VSSP
AIB72	unused_AIB72	out	VSSP
AIB73	unused_AIB73	out	VSSP
AIB74	device_detect[3]	out	S10 chiplet C4 bump
AIB75	device_detect[3]	out	no connect
AIB76	unused_AIB76	out	VSSP
AIB77	unused_AIB77	out	VSSP
AIB78	unused_AIB78	out	VSSP
AIB79	unused_AIB79	out	VSSP
AIB8	unused_AIB8	out	VSSP
AIB80	unused_AIB80	in	no connect
AIB81	unused_AIB81	in	no connect
AIB82	unused_AIB82	in	no connect
AIB83	unused_AIB83	in	no connect
AIB84	unused_AIB84	in	no connect
AIB85	por[4]	in	S10 chiplet C4 bump
AIB86	unused_AIB86	in	no connect
AIB87	por[4]	in	no connect
AIB88	unused_AIB88	in	no connect
AIB89	unused_AIB89	no connect	no connect
AIB9	unused_AIB9	out	VSSP

AIB90	unused_AIB90	in	no connect
AIB91	unused_AIB91	no connect	no connect
AIB92	unused_AIB92	out	VSSP
AIB93	unused_AIB93	out	VSSP
AIB94	unused_AIB94	out	VSSP
AIB95	unused_AIB95	out	VSSP

1: All VSSP may be connected and pulled down by a single 1K resistor to VSS or GND.
2. Unused outputs should be set by the S10 Chiplet to 0.
3. The two device_detect bumps should be connected by S10 Chiplet wiring after its output buffer. This provides redundancy in the event of an open on one or the other bump.
4. The two por bumps should be connected by S10 Chiplet wiring before its AUX input. This provides redundancy in the event of an open.

4.3.1 Alternate No AUX Channel

With only 4 bumps utilized in the AUX channel, it is anticipated that a S10 chiplet may be built without an AIB AUX channel. The S10 chiplet in this case may drive *device_detect* from a C4 bump and receive *por* with a C4 bump, each connected to respective Stratix 10 EMIB microbumps through package surface traces. Unused S10 Chiplet outputs in this No AUX case do not actually exist. To avoid floating inputs to Stratix 10, at the EMIB those pins may be tied to VSSP (1K ohm pulldown to VSS or GND). Unused inputs to the S10 chiplet, which also do not actually exist in the No AUX case, may be "no connect" at the EMIB from Stratix 10.

This is a means for a S10 chiplet to avoid implementing the AIB AUX channel, which may be useful in fitting a 24 channel AIB chiplet into an 8mm maximum die height.

The S10 chiplet with the alternate no AUX Channel shall drive *device_detect* and observe *por* with the same behavior as described in the AIB Spec.

4.4 Channel Stacking

A S10 chiplet uses the East channel stacking as shown in the AIB Spec. A S10 chiplet shall add two rows of microbumps in gaps between AUX and channel0, channels 5 and 6, 11 and 12, 17 and 18, and above 23 as shown in Figure 5:

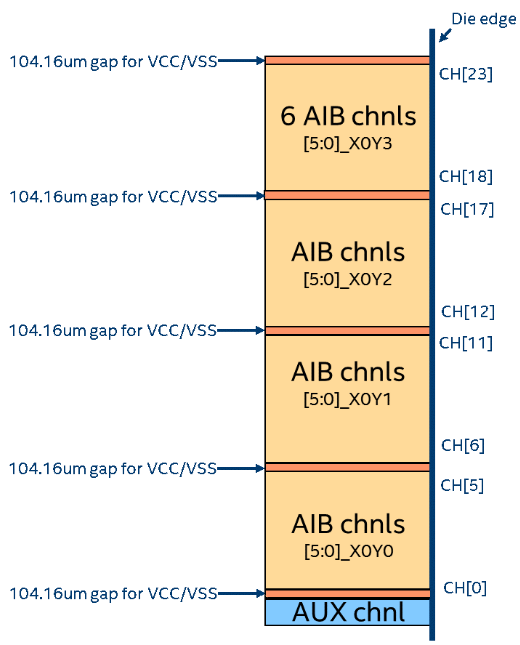

Figure 5. AIB Channel Stacking

4.5 AIB Power and Ground

The Alternate Bump Map in the AIB Spec identifies bumps as VCCIO and VDD.

4.5.1 VCCIO

VCCIO is supplied by Stratix 10. A Stratix 10 chiplet shall use VCCIO to power the input and output stages of its AIBIO cells, and shall draw no more current than the maximum of Table 5, as stated in the AIB Spec.

Table 5. S10 Chiplet Maximum Current Draw from VCCIO

VCCIO Maximum Current
40mA per channel

For testability of the S10 chiplet, the S10 chiplet shall connect VCCIO to C4 bumps, typically 3 per 24 channel AIB interface. Microbumps are typically not probed, and the C4 bumps are a means for test equipment to power the S10 chiplet's VCCIO before assembly.

A Stratix 10 chiplet shall provide voltage translators between the input and output stages of the AIBIO cells and the rest of the S10 chiplet.

4.5.2 VDD

VDD is optionally supplied by a S10 chiplet into the VDD ubumps on rows AQ through AT, and drawn from the VDD ubumps on rows Y and Z. The purpose is to supply power to S10 chiplet circuitry in the area under the AIB bump array that is not powered by VCCIO. See Figure 6.

4.5.3 Power and Ground Bumps

The Stratix 10 EMIB connects all the VDD microbumps together, all the VSS microbumps together and all the VCCIO microbumps together. The multi-chip package of S10 chiplets + Stratix 10 should connect VDD C4 bumps and microbumps together, also VSS C4 bumps and microbumps together using surface traces. VCCIO microbumps should be connected in the package using surface traces.

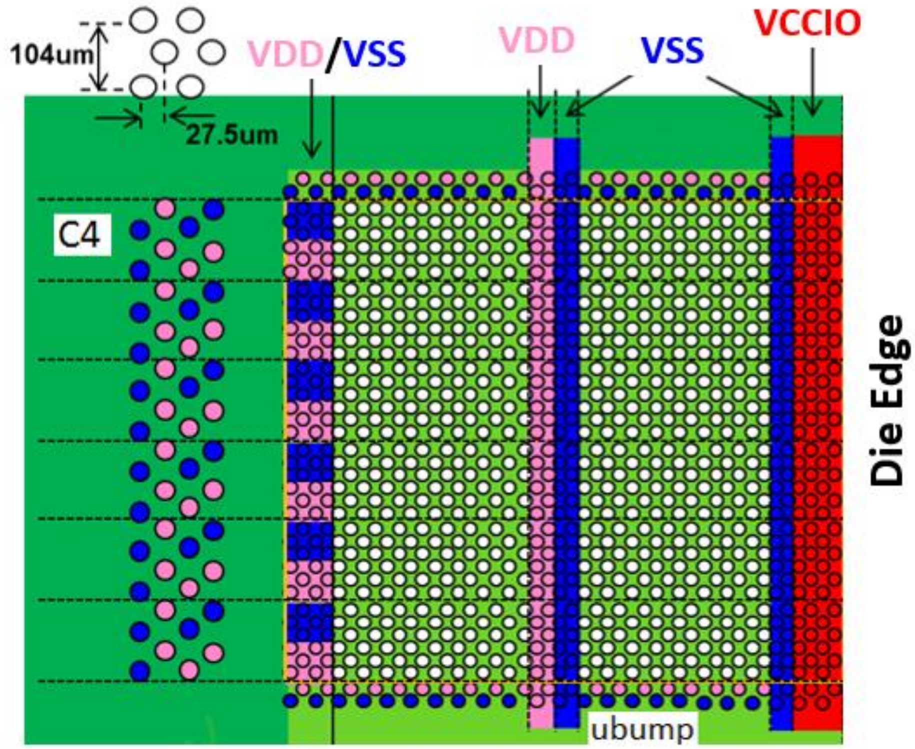

Figure 6. S10 Chiplet AIB Power Microbumps

5 AIB Bump Locations and Assignments

The locations of microbumps and surrounding C4 bumps are given in the spreadsheet referenced in 5.1. Table 6 gives the spreadsheet's bump names and description, and the translation to the Verilog names of module aib_top.

Table 6. S10 Chiplet AIB Bumps

Location Spreadsheet Bump/Pin name	Verilog aib_top signal name	Pin direction	Bump type	Pin Type/Function
AIB{0-95}_CH{0-5}_X0Y0	io_aib_ch{0-5}[95:0]	inout	µbump	AIB signals for channels 0-5
AIB{0-95}_CH{0-5}_X0Y1	io_aib_ch{6-11}[95:0]	inout	µbump	AIB signals for channels 6-11
AIB{0-95}_CH{0-5}_X0Y2	io_aib_ch{12-17}[95:0]	inout	µbump	AIB signals for channels 12-17
AIB{0-95}_CH{0-5}_X0Y3	io_aib_ch{18-23}[95:0]	inout	µbump	AIB signals for channels 18-23
AIB_AUX{0-95}_X0Y0	io_aib_aux[95:0]	inout	µbump	AIB signals for aux channel
VCC_HSSI (AIB Spec uses VDD)	n/a	power	C4/µbump	Digital supply for AIB and MAC circuits on chiplet core side of AIBIO (away from ubumps)
VCCL_HSSI (AIB Spec uses VCCIO)	n/a	power	C4/µbump	IO supply from Stratix 10 regulated to 0.75V-0.97V for S10 chiplet AIBIO circuits on ubump side of AIB IO cell
VSSGND (AIB Spec uses VSS)	n/a	ground	C4/µbump	Digital VSS

5.1 AIB Bump Locations Spreadsheet

See companion file "Stratix 10 Chiplet AIB Profile_v1_0_aib_bump_locations.xlsx".

5.2 S10 Chiplet Mechanical

The figure and table below are for S10 chiplets using the existing Stratix 10 EMIB.

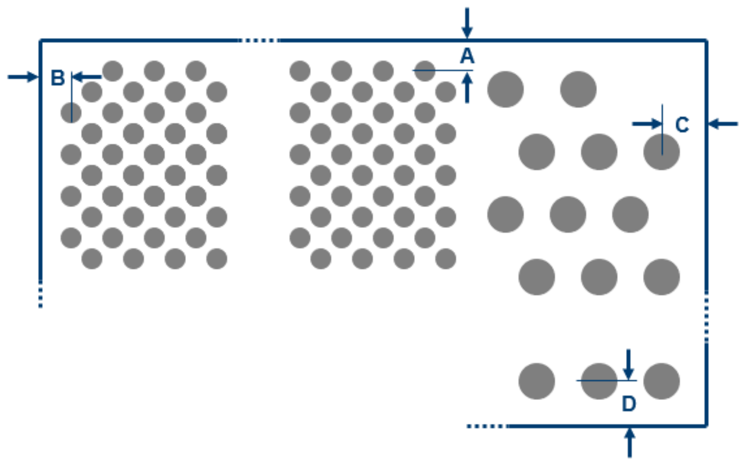

Figure 7. S10 Chiplet Bump Centers to Die Edges

Table 7. S10 Chiplet Feature Dimensions

Feature	Dimension Max	Dimension Min
A	127.6 um	50 um
B	117.9 um	50 um
C	139.9 um	63.5 um
D	162.2 um	63.5 um

The dimensions in Figure 7Table 7 are to the finished die edges. We recommend that you use the Dimension Max as your bump center to scribe line center dimension. Process-dependent dicing will produce smaller dimensions to finished die edges. Use the bump locations spreadsheet for the microbump to C4 bump spacing and positions.

6 AIB Open Source

6.1 AIB Open Source Signal Naming

The AIB Open Source code at https://github.com/intel/aib-phy-hardware uses an earlier naming scheme for AIB signals. Table 8 is a mapping of the AIB spec names to AIB Open Source names.

Table 8. AIB Spec Signal Names to AIB Open Source Signal Names Mapping

AIB Spec Signal Name	AIB Open Source Signal Name
tx[19:0]	u_rx_data_out[19:0]
ns_fwd_clk/ns_fwd_clkb	u_rx_transfer_clk/u_rx_transfer_clk_n
ns_fwd_div2_clk/ns_fwd_div2_clkb	u_pld_pcs_rx_clk_out/u_pld_pcs_rx_clk_out_n
ns_rcv_clk/ns_rcv_clkb	u_pma_aib_tx_clk/u_pma_aib_tx_clk_n
ns_rcv_div2_clk/ns_rcv_div2_clkb	u_pld_pcs_tx_clk_out/u_pld_pcs_tx_clk_out_n
rx[19:0]	u_tx_data_in[19:0]
fs_fwd_clk/fs_fwd_clkb	u_tx_transfer_clk/u_tx_transfer_clk_n
ns_sr_data	u_ssr_data_out
ns_sr_clk/ns_sr_clkb	u_sr_clk_out/u_sr_clk_n_out
ns_sr_load	u_ssr_load_out
fs_sr_data	u_ssr_data_in

AIB Spec Signal Name	AIB Open Source Signal Name
fs_sr_clk/fs_sr_clkb	u_sr_clk_in/u_sr_clk_n_in
fs_sr_load	u_ssr_load_in
fs_mac_rdy	u_pld_pma_rxpma_rstb
ns_mac_rdy	u_pld_pma_clkdiv_rx_user
fs_adapter_rstn	u_adapter_rx_pld_rst_n

7 AIB Initialization

The following steps are a combination of the AIB Spec, Stratix 10 requirements and application of the AIB Open Source.

1. The S10 Chiplet at power on shall drive device_detect high as required by the AIB Spec. The AIB AUX por signal from Stratix 10 (in the Open Source the por signal is represented as the PHY to MAC signal aibaux_por_vccl) should be used by the S10 Chiplet to hold its AIB in reset and keep its AIB outputs in standby mode (see the AIB spec for standby mode). If you are using the AIB Open Source then this step is handled for you.

2. At power on the S10 Chiplet MAC should hold the AIB adapters in reset using the Open Source MAC to PHY signal i_adpt_hard_rst_n asserted LO. The S10 Chiplet MAC should pull down its open-drain CONF_DONE pin. The i_adpt_hard_rst_n also keeps the AIB outputs in standby mode. The S10 Chiplet should deassert ns_mac_rdy as LO, exposed at the Open Source MAC/PHY interface as ns_mac_rdy.

3. At power on the S10 chiplet should execute its own internal power-on reset including achieving stable internal clocks and free-running clock for AIB.

4. After the S10 Chiplet exits its own power-on reset and once the por signal from Stratix 10 is received as LO then the S10 Chiplet should begin configuration. The S10 chiplet MAC should continue to hold its AIB outputs in standby mode using i_adpt_hard_rst_n asserted LO.

5. After the S10 Chiplet has completed configuration, the S10 Chiplet MAC should release CONF_DONE. The S10 Chiplet MAC should monitor CONF DONE for other chiplets that may still be pulling CONF_DONE LO. Once the S10 Chiplet MAC reads CONF_DONE as HI, the S10 Chiplet MAC should deassert i_adpt_hard_rst_n. Data at this point is don’t care. The S10 Chiplet MAC should assert ns_mac_rdy HI for each channel to signal it is ready to start calibration.

6. After the S10 Chiplet has deasserted i_adpt_hard_rst_n, the S10 Chiplet should start sending and receiving using the sideband control shift registers. If you are using the AIB Open Source adapter then this step is handled for you.

7. Each AIB channel continuously monitors its fs_adapter_rstn input from Stratix 10. The channel’s calibration state machines are held in reset while fs_adapter_rstn is LO as required by the AIB Spec. Each AIB channel of the Stratix 10 chiplet should start calibration when fs_adapter_rstn is deasserted. If you are using the AIB Open Source adapter then this step is handled for you.

8. For each channel, the S10 Chiplet can determine that calibration on each side has completed and the link is up by monitoring the sl_tx_transfer_en shift register bit from Stratix 10, and the ms_tx_transfer_en from the S10 Chiplet’s own AIB Adapter. As the AIB spec indicates, when both sl_tx_transfer_en and ms_tx_transfer_en are true, then the link shall be ready to transmit data. At the Open Source PHY/MAC interface, sl_tx_transfer_en is visible as the signal sl_tx_transfer_en and ms_tx_transfer_en is visible as the signal ms_tx_transfer_en.

At any point, if AIB AUX por is asserted, go back to step 1.

7.1 Stratix 10 device_detect workaround

Stratix 10 has an issue with reading device_detect where it expects unsupported JTAG behavior from the S10 Chiplet if Stratix 10 reads device_detect as HI. To work around this issue, in the package disconnect AUX AIB74 and AUX AIB75 between the S10 Chiplet and the Stratix 10 EMIB. Ground those AUX AIB74 and AUX AIB75 Stratix 10 EMIB wires on the package.

8 Additional Information

[1] "Advanced Interface Bus Specification," Revision 1.1, Intel Corporation, April 2019, https://github.com/intel/aib-phy-hardware/tree/master/docs

Revision History

6/2019: Updates to initialization, mechanical, added the ns_mac_rdy signal from the AIB Spec into the list supported by a S10 Chiplet. Changed the MAC data bit numbering in the Word Marking section to match the AIB Open Source.