ACCELERATING INNOVATION THROUGH A STANDARD CHIPLET INTERFACE: THE ADVANCED INTERFACE BUS (AIB)

5/2019

David C. Kehlet, Research Scientist

Julie Zhang, Senior Design Engineer

Legal Information

This presentation contains the general insights and opinions of Intel Corporation ("Intel"). The information in this presentation is provided for information only and is not to be relied upon for any other purpose than educational. Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at www.intel.com.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

† Tests measure performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.

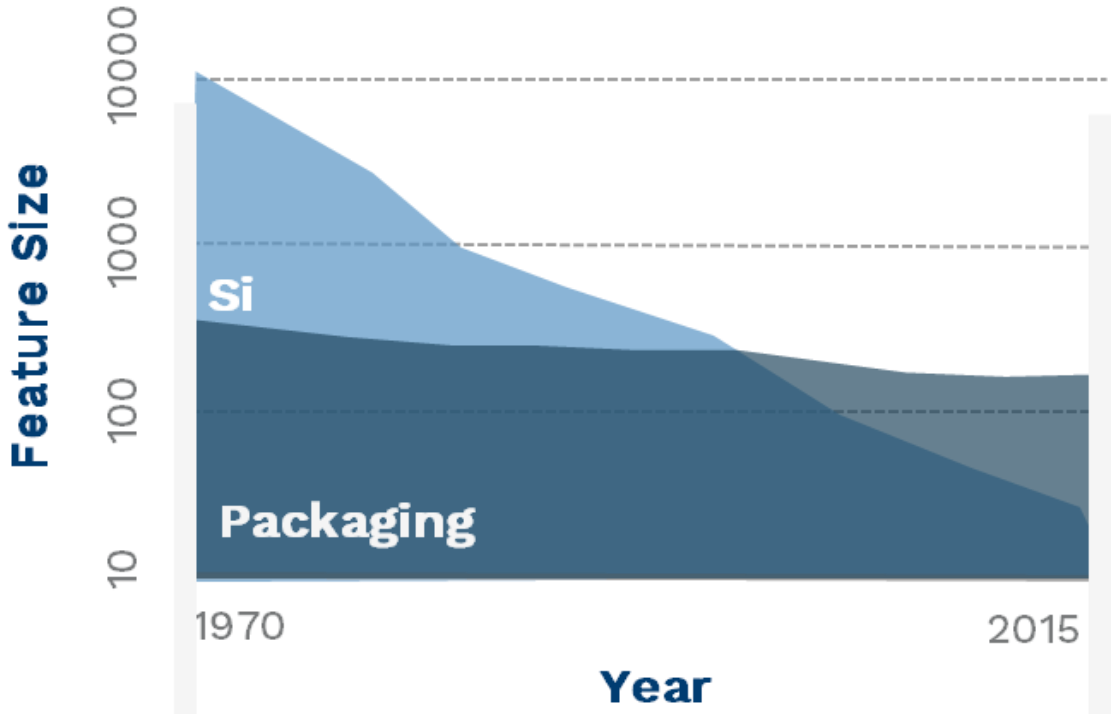
The Case for Chiplets

"It may prove to be more economical to build large systems out of smaller functions, which are separately packaged and interconnected." [1]

Gordon E. Moore

[1] 3rd Page of Moore's 1965 paper, "Cramming more components onto integrated circuits"

Cost and Performance: Silicon vs. Packaging



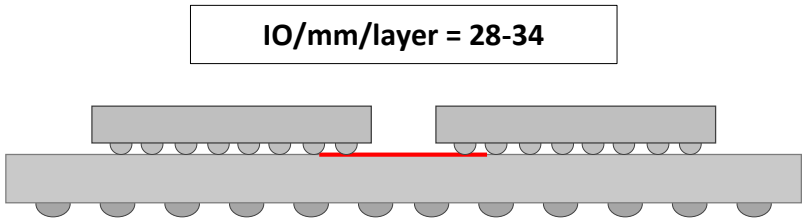
Silicon vs. packaging scaling disparity drove serializer/deserializer (**SERDES)** to connect the dies

→ Increased **complexity**, system **latency** and **power**

→ Long coupled development cycles for analog + digital on the same monolithic die

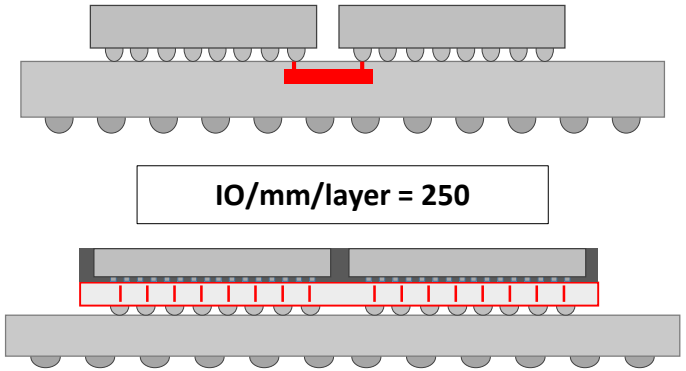
High-Density Packaging

Standard Flip-Chip Packaging Technology



FCXGA, FCCSP

High-Density Packaging Technology



EMIB

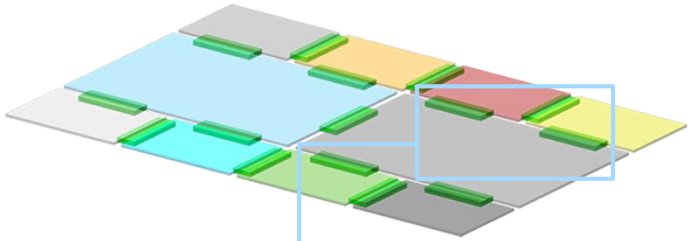
Silicon Interposer

High-density packaging technology provides 7-8x IO density increase

High-Density Packaging

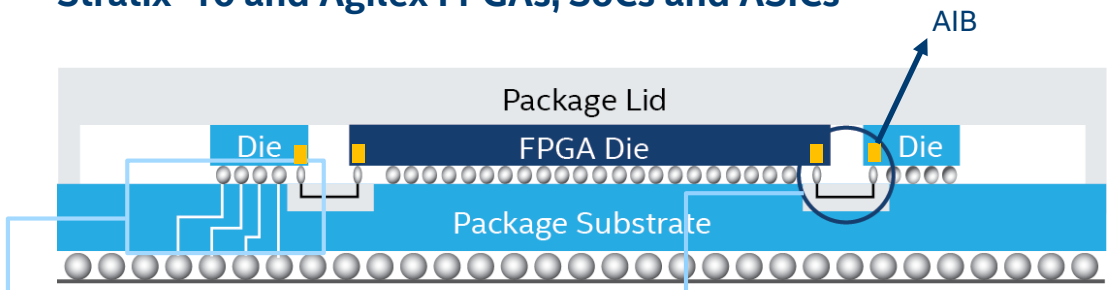
EMIB or Silicon Interposer: Enabling technology for AIB

AIB over EMIB



Multiple bridges possible

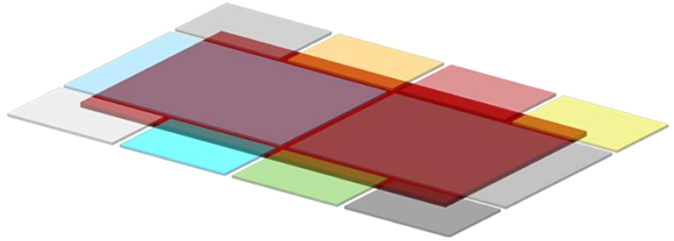
Stratix® 10 and Agilex FPGAs, SoCs and ASICs

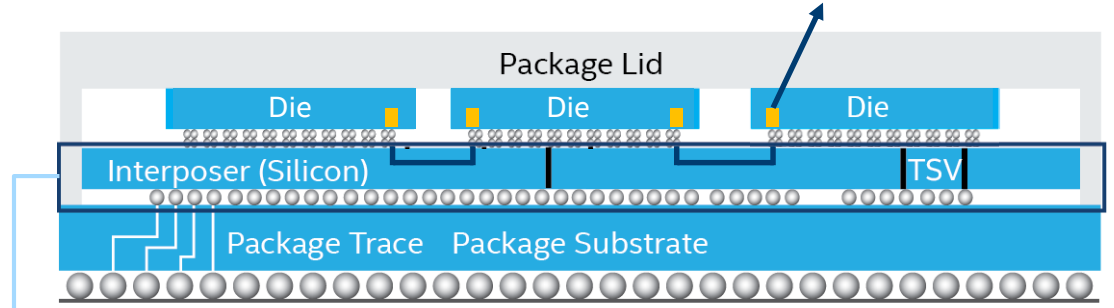


The rest of the package: RF, other I/Os, and power are unaffected

Localized high-density die-to-die interconnect

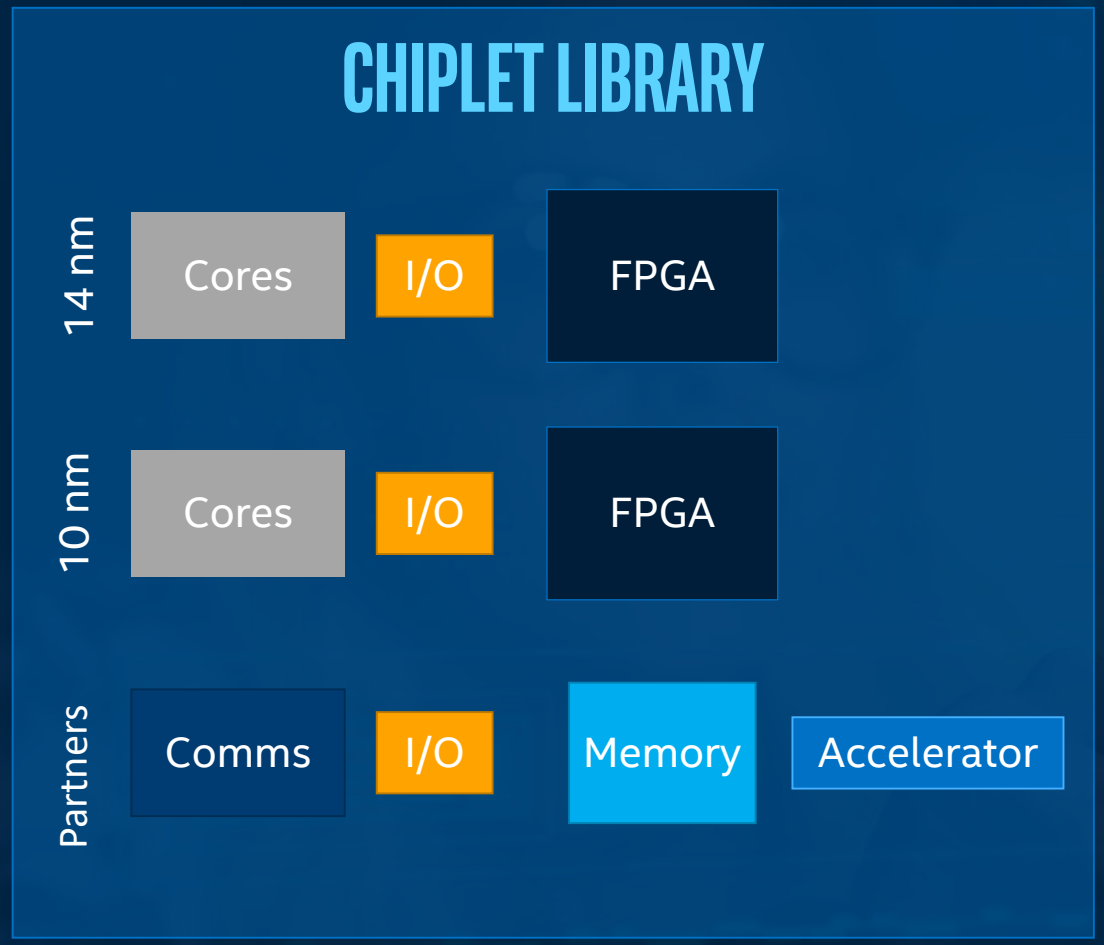
AIB over Silicon Interposer

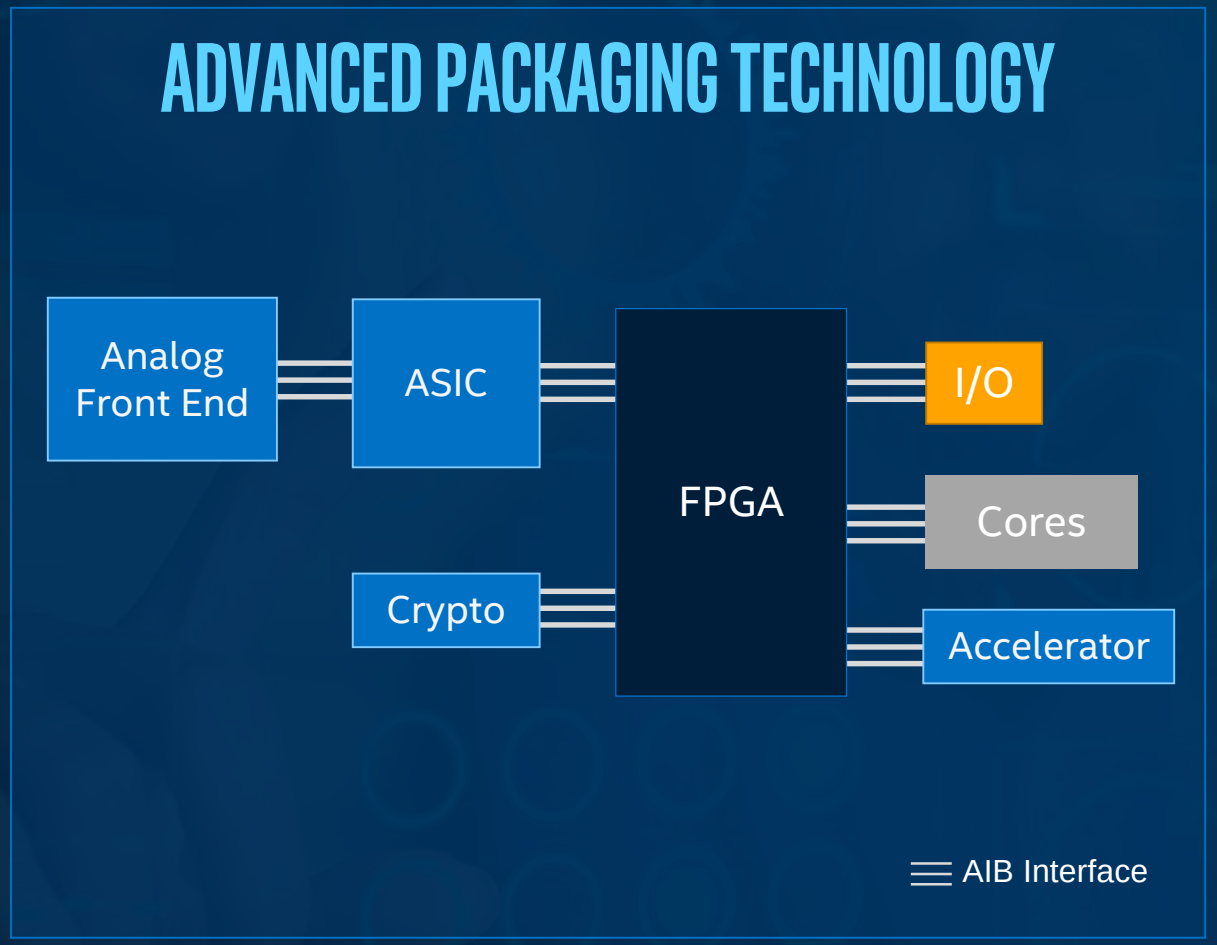




All signals will travel through the Interposer, including RF signals

Example of How This Works





Intel and Chiplets

New Chiplets with Intel® 10 nm FPGA

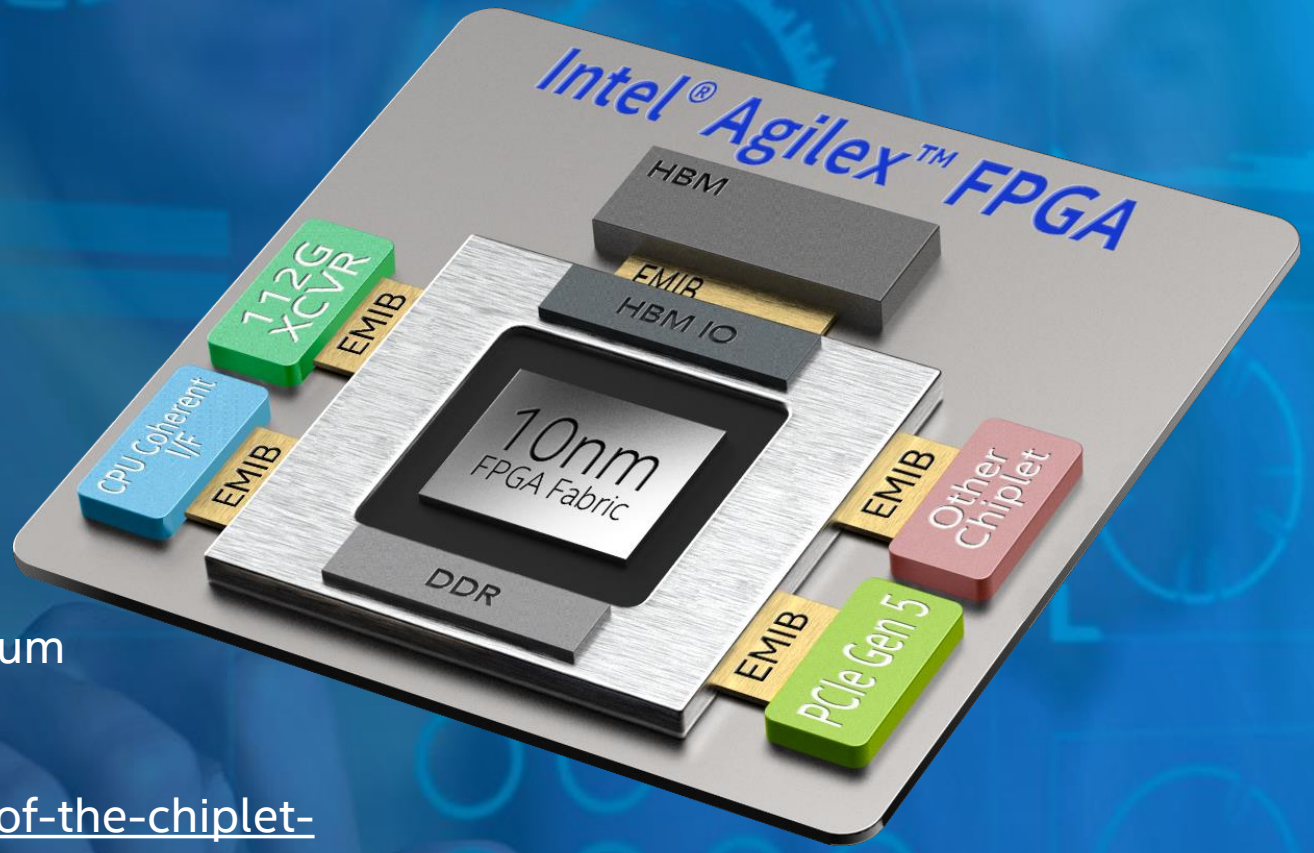
- 112 Gbps serializer /deserializer (SERDES)
- PCI Express* (PCIe) Gen5
- Compute Express Link (CXL)
- Intel® 3D Xpoint™
- Custom intellectual property (IP)

IP Reuse from Intel Stratix® 10 FPGA

- 58 Gbps SERDES E-Tile

Chiplet Articles

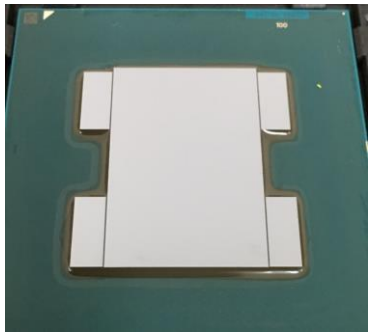
- Ramune Nagisetty from Intel talks to IEEE Spectrum about chiplets:
  - https://spectrum.ieee.org/tech-talk/semiconductors/processors/intels-view-of-the-chiplet-revolution
- Microprocessor Report: Intel 10 nm FPGAs Add Chiplets
  - https://www.linleygroup.com/mpr/article.php?id=12125

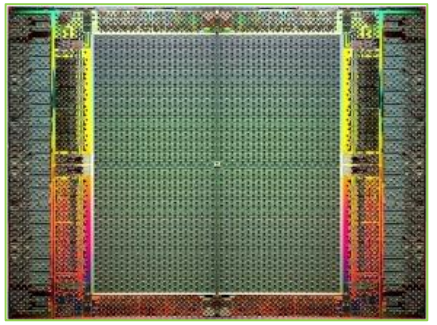


High-Bandwidth Interfaces

Near Monolithic Metrics

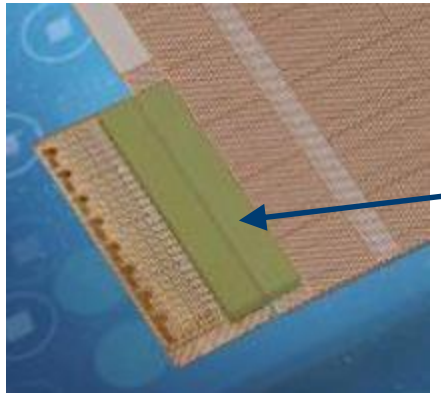
AIB





	On-Board	On-Package	On-Die
Distance	Meter	3 mm	<1 mm
Wire Density (lines/mm/layer)	2	250 (EMIB)	1000
Power (pJ/b)	20	<1	0.1
Standard	PCIe, DDR SDRAM, …	HBM, AIB	AXI*, APB*, …

When Wires Are Free



Silicon Interposer

- 500 wires/mm → 4,000 wires in Intel® Stratix® 10 FPGA EMIB
- Typical interposer → USD 0.0005 per wire
- Conclusion: spend very little at each end of the wire
  - *Don't spend silicon on serialization, complex SERDES, encoding schemes, I/O training, pre-emphasis or equalization*
  - *More Gbps/wire is not better when you can go wider*

Google's FASTER Trans-Pacific Cable (2016)

- 6 fiber pairs: 12 "wires"
- Cable cost USD 300M → USD 25M per wire
- Conclusion?
  - Spend everything you can at each end of the wire!

5 trillion times more!†

Image Source: US Navy

Bandwidth Density and CHIPS

Monolithic-like bandwidth density is needed to make chiplets practical

- What does it take to approach on-die interconnect's bandwidth density?

DARPA set a target of 1 Tbps/mm (1,000 Gbps)

On die interconnect bandwidth density

- 1 Gbps/wire, 1u wire spacing → 1 Tbps/mm

Typical PCI Express* bandwidth density

- 16 Gbps/wire, 0.3 mm SERDES spacing → 0.05 Tbps/mm
- 1/20th the bandwidth density of on-die routing!

DARPA What is CHIPS?

CHIPS will develop design tools, integration standards, and IP blocks required to demonstrate modular electronic systems that can leverage the best of DoD and commercial designs and technology.

CHIPS Program Interface Standard Metrics	
Bandwidth density	> 1000 Gbps/mm

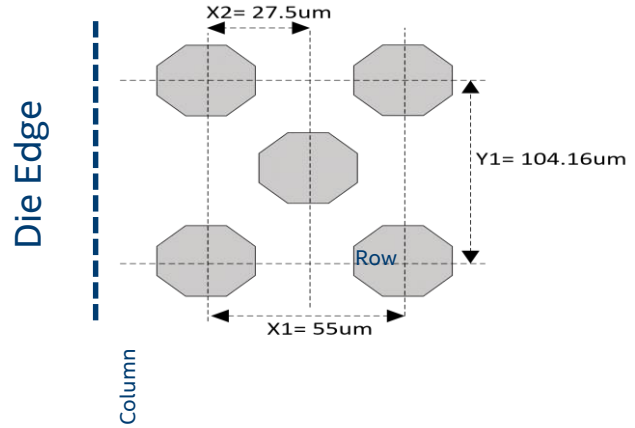
Source: DARPA

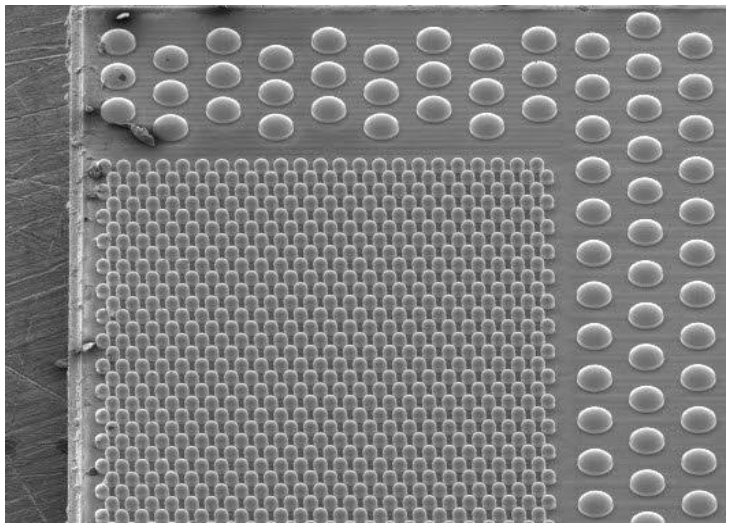
Bandwidth and Area

Achieving DARPA Target of 1 Tbps/mm

- SERDES: silicon area limited, not C4 bump limited
  - Typical 56 Gbps SERDES at 0.31 mm high, 2 mm deep (1 Tbps = 18 of these)
    - 1Tbps/mm SERDES array: 3 high, 6 deep (=12 mm deep!!!)
  - Suppose a future 112 Gbps SERDES in half the depth (4X improvement)†
    - 1Tbps/mm SERDES array: 3 high, 3 deep (=3 mm deep!!)
  - ***Future 1 Tbps/mm SERDES chiplet would be 3 mm wide just for die-to-die connection***
- Wide parallel on high-density packaging
  - Microbump limited (simple I/O cell is much smaller than a microbump space
  - How deep is 1 Tbps?
    - Nine microbump rows /mm (~=1mm/104u) with column spacing (X1)/2 = 27.5u
    - 1 Tbps = 2 Gbps/microbump * 9 microbump rows/mm * N columns
    - N = 67 columns
      - Assume 20% overhead for power and ground microbumps
  - Microbump array depth = 1.85 mm (2016 technology)
  - ***At 4 Gbps/wire future technology, microbump array depth = 0.92 mm***

Intel® Stratix® 10 FPGA Microbump Spacing



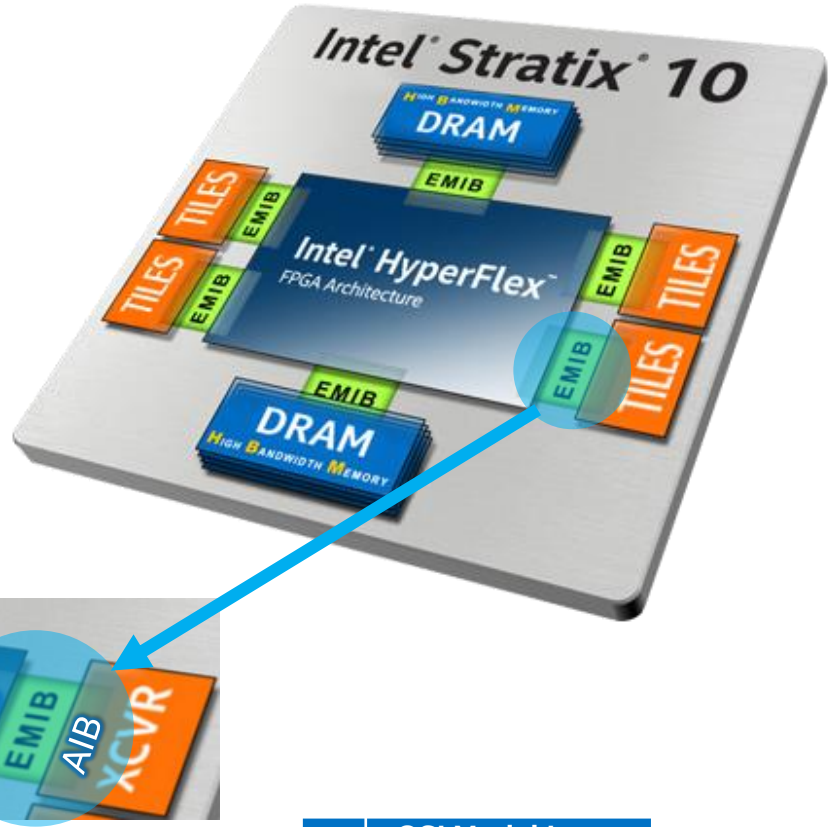


Intel® Stratix® 10 FPGA 55u microbump array and neighbouring 130u C4 flip chip bumps

AIB Die-to-Die Physical Interface

AIB: Common chiplet physical interface

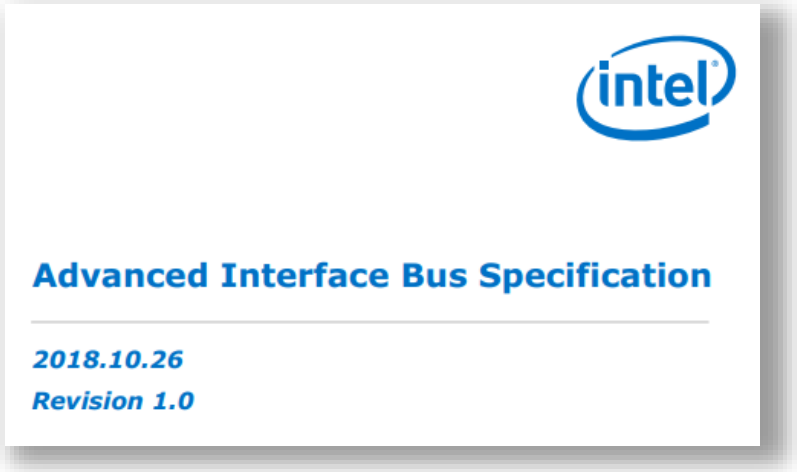
- **A**dvanced **I**nterface **B**us (AIB)
- AIB is a clock-forwarded parallel data transfer like DDR DRAM
- High density with a 2.5D interposer like CoWoS or EMIB for multi-chip packaging
- AIB is PHY level: OSI Layer 1
- Build protocols like AXI*-4 or PCI Express* on top of AIB



	OSI Model Layer
7	Application
6	Presentation
5	Session
4	Transport
3	Network
2	Data Link
1	Physical

AIB

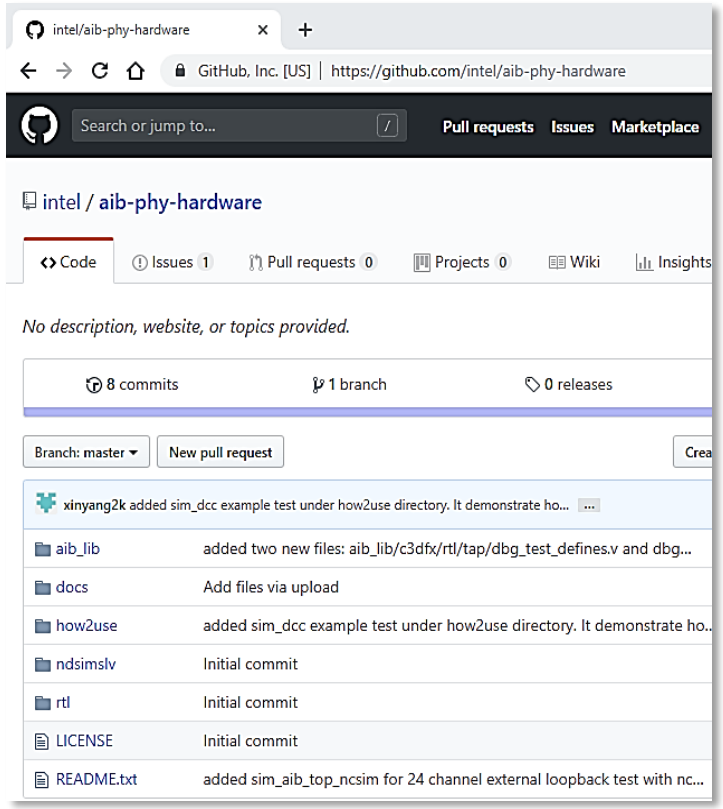
AIB Public Releases



AIB Specification Released October 2018

AIB Public Specification includes:

- Electrical specs, bump array mechanicals, data/clock/ control signal definitions, reset handshaking, JTAG reqts
- Available at https://github.com/intel/aib-phy-hardware/tree/master/docs

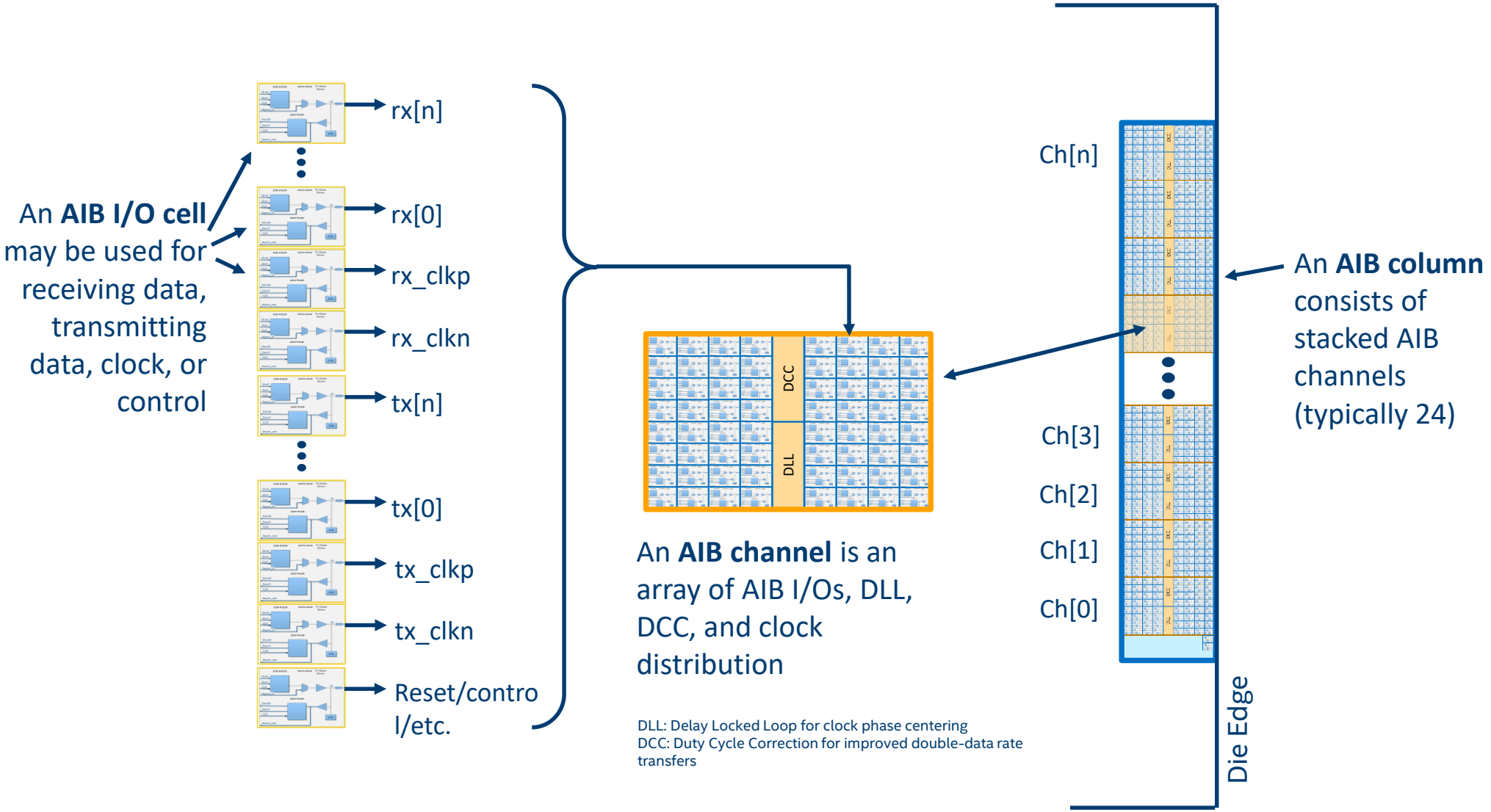


AIB Open Source Released January 2019

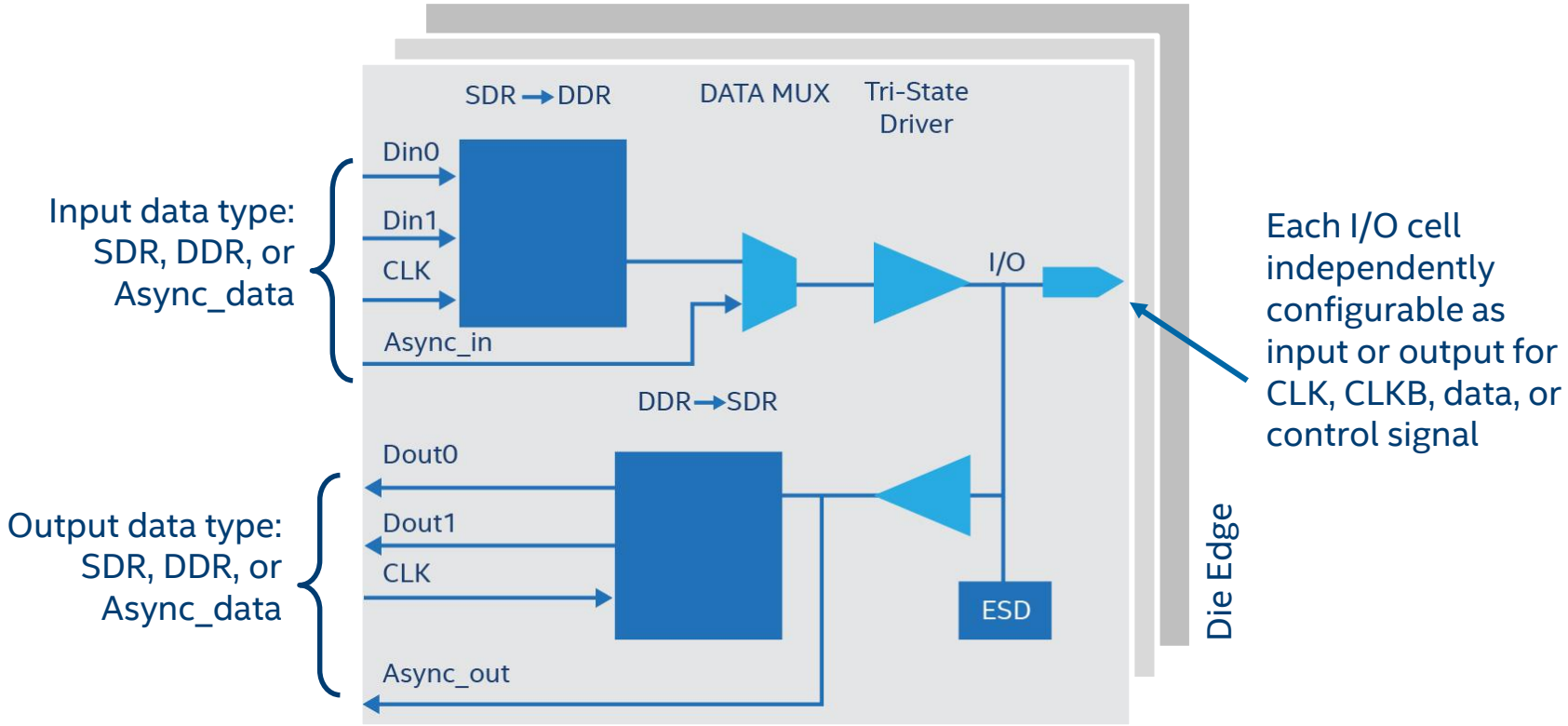
AIB Open Source on GitHub

- Register transfer level (RTL), netlists, generic cell library
- Available at https://github.com/intel/aib-phy-hardware
- Purpose: **reduce development cost**

AIB I/O Cell, Channel, and Column



AIB I/O Basic Structure (Example)



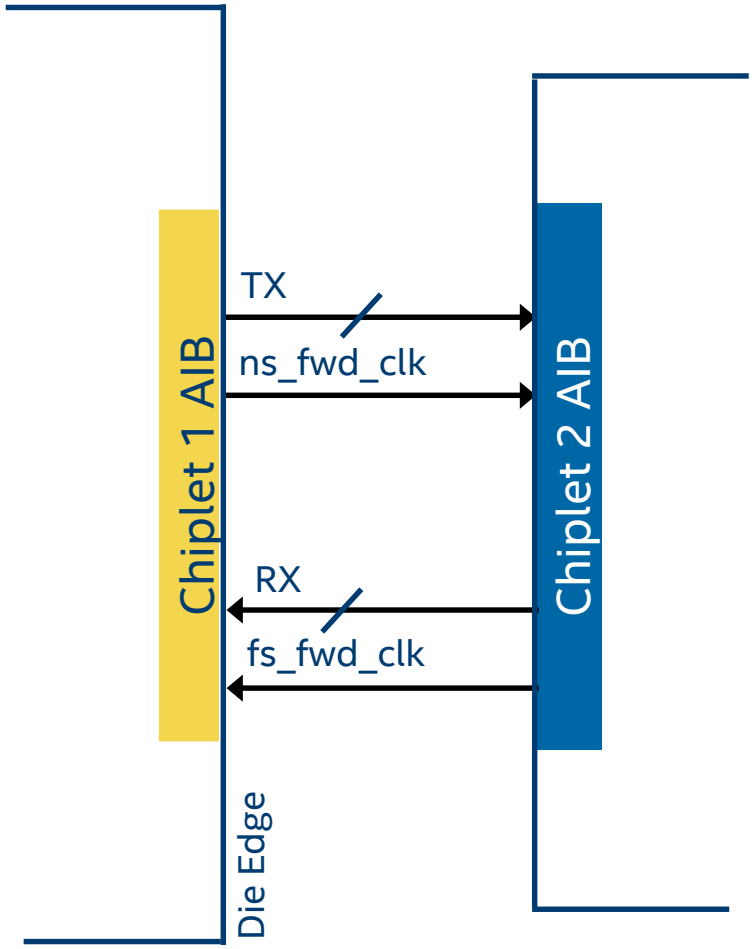
Simple!

I/Os are statically assigned input or output so a fixed input or output design that meets AIB signaling specification can be compliant

Why might you make the cell have both input and output when it will only be used in one mode?

- Testability! Provides a loopback path at the pin

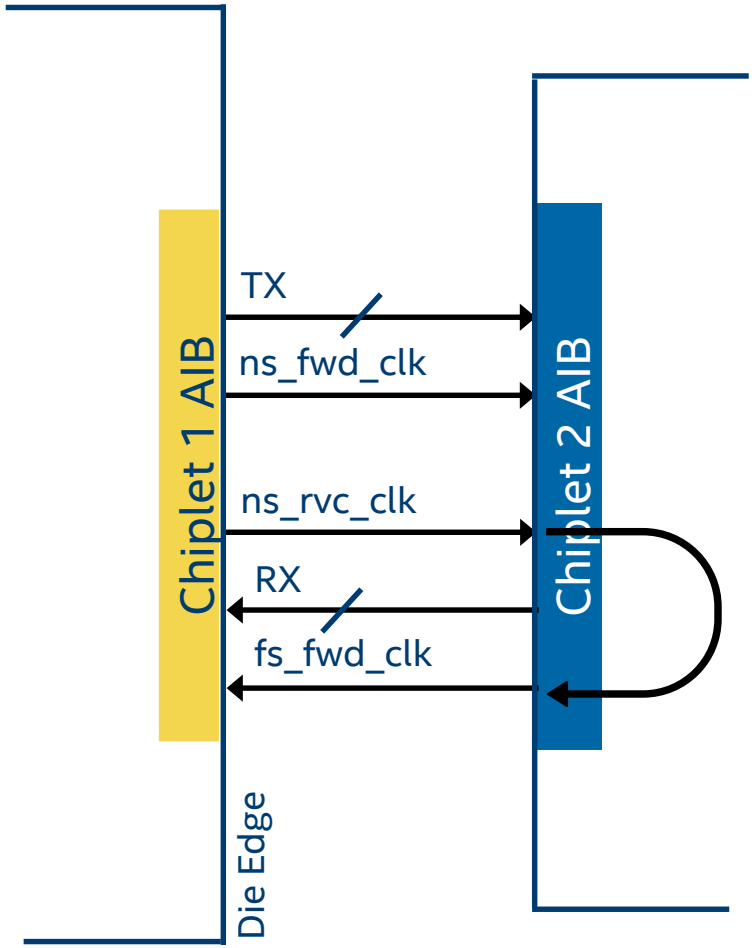
Simplest AIB Clocking Configuration (One Channel)



Each chiplet originates its own clock for transmitting to the other side

- Each channel can forward an independent clock or the same clock

Typical AIB Clocking Configuration (One Channel)



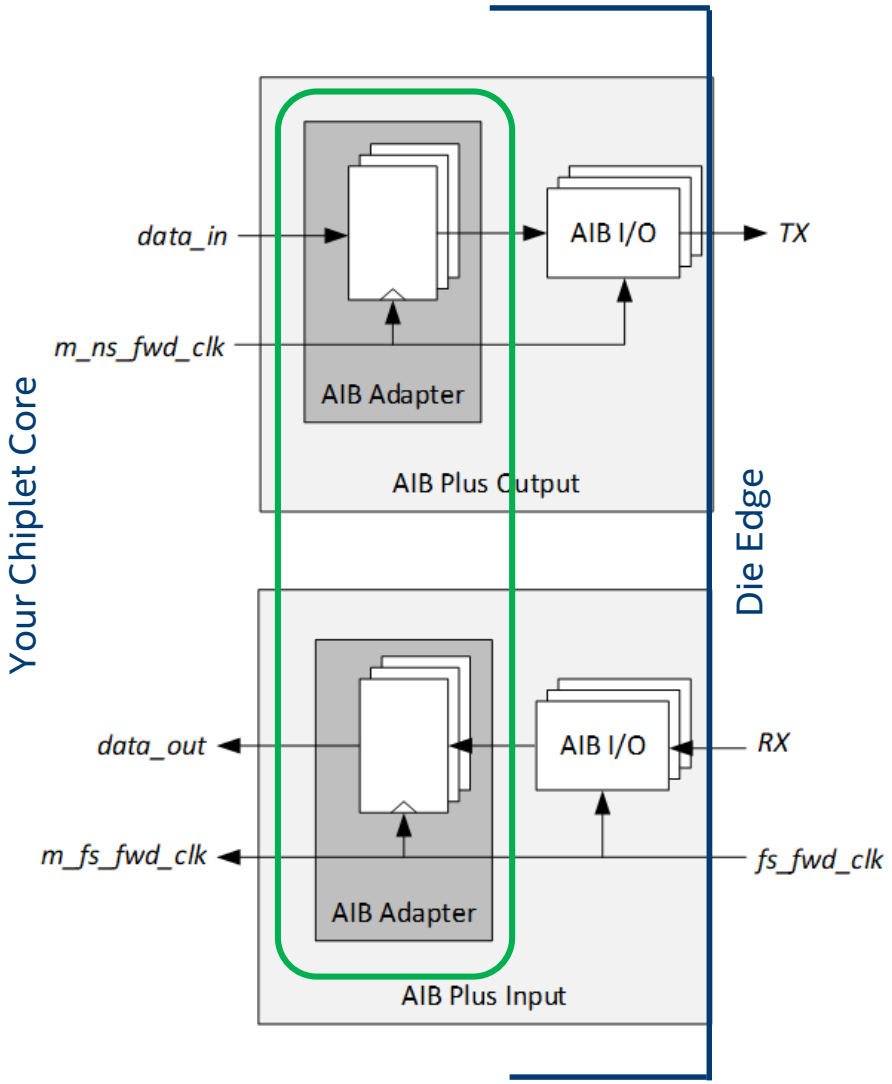
Chiplet 1 here sends the “ns_rvc_clk” for Chiplet 2 to use internally and as the source for Chiplet 2’s fs_fwd_clk

Why might you do this?

Often convenient if the clock generation is all on one chiplet

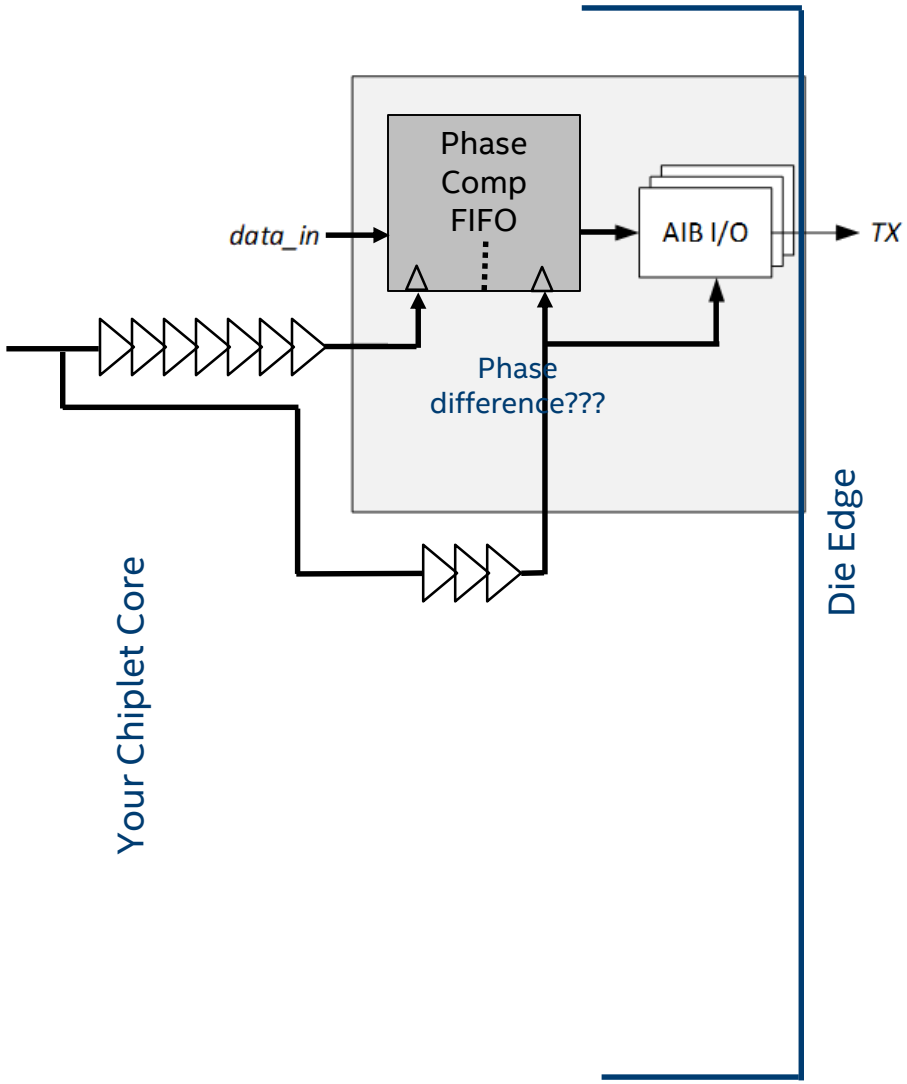
- SERDES chiplet, for example, may contain all the phase-locked loop (PLL) clock generation

Clocking Inside Your Chiplet



Likely you will need (at least) a retiming register in the AIB adapter to interface between the AIB I/O and your core registers

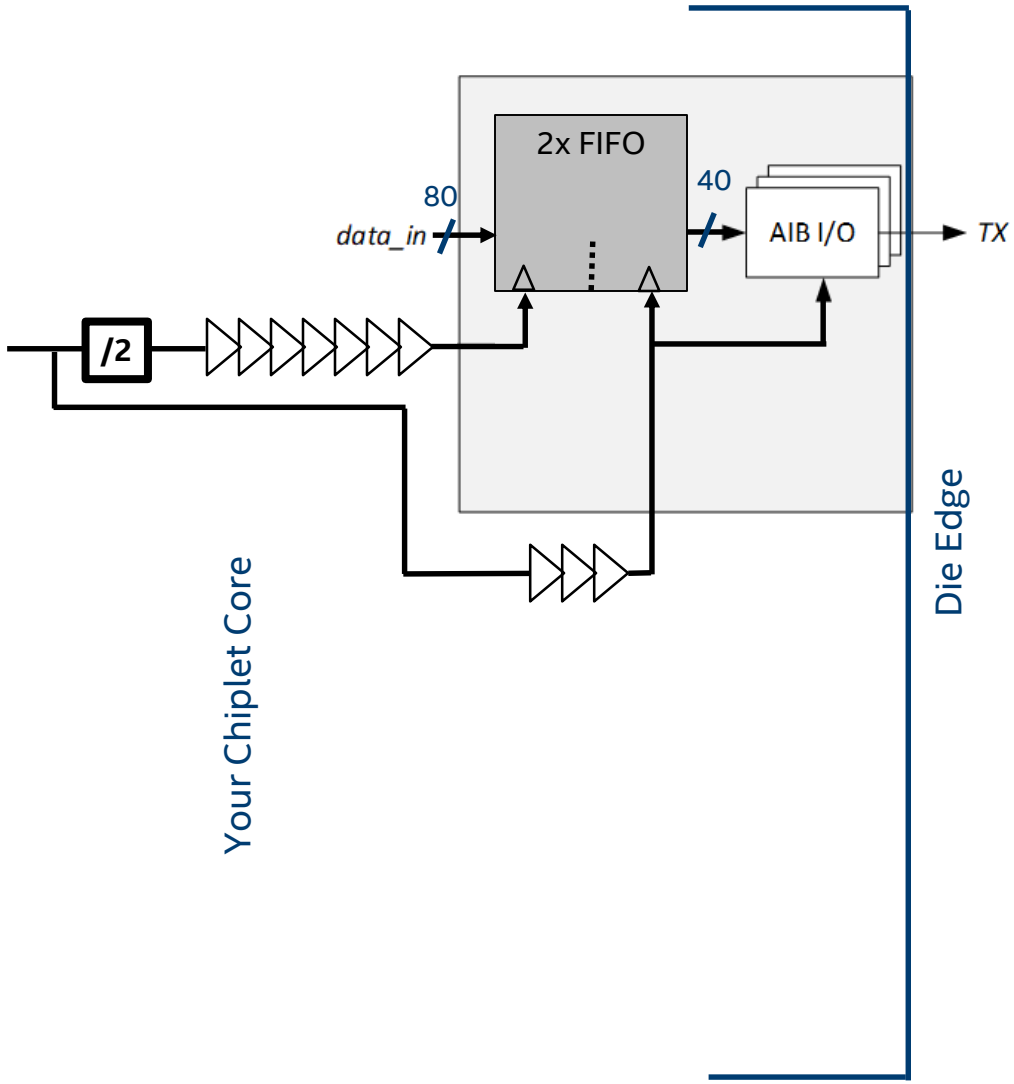
Clocking Inside Your Chiplet



The clock distribution between your chiplet core and the AIB I/O may be a completely different tree, even when the clocks are the same source (0 PPM difference)

- *Mesochronous: same frequency, unknown phase*
- You need a phase compensation FIFO in this case

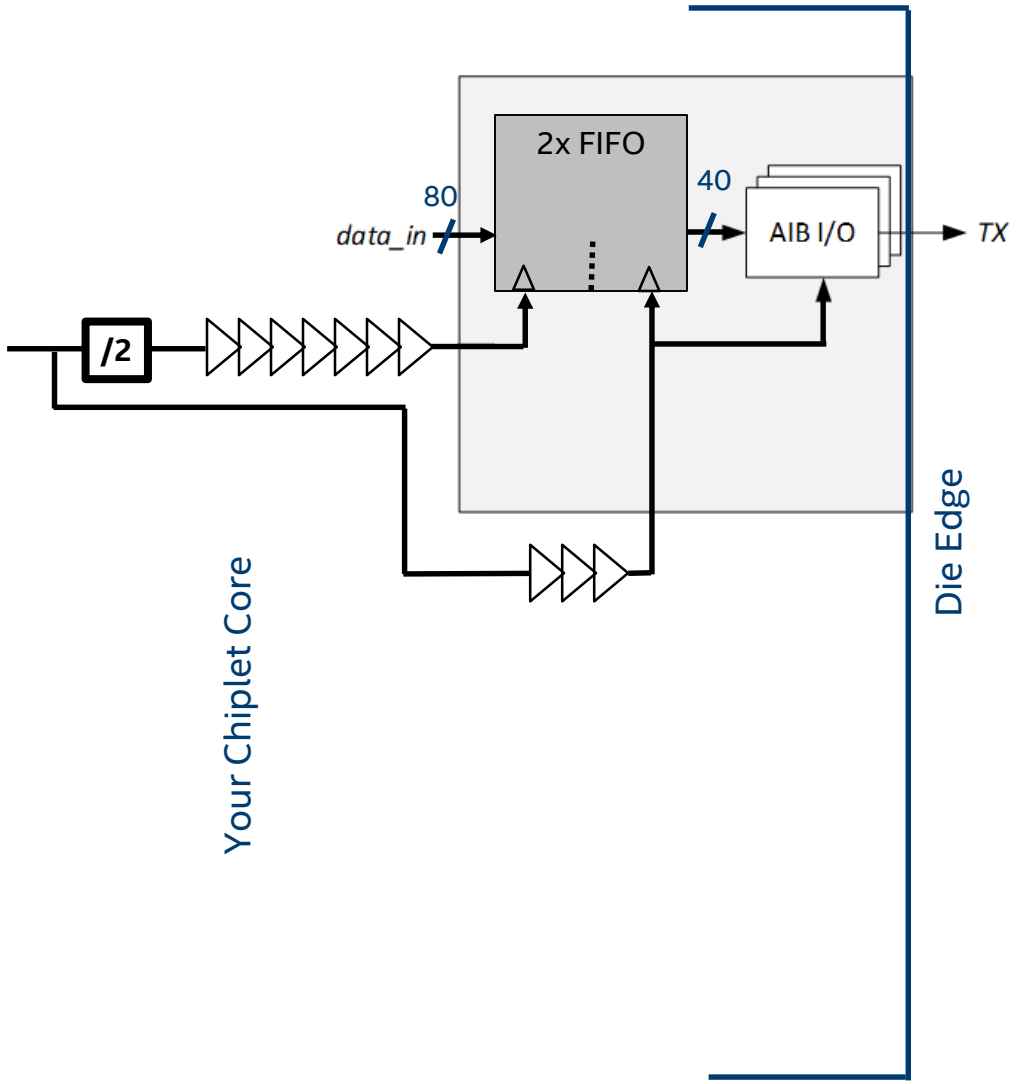
Clocking Inside Your Chiplet (Application)



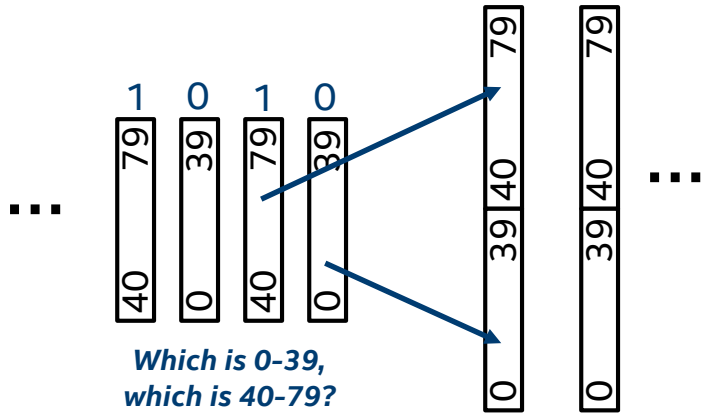
What if you can't run your core frequency at 1 GHz to match the 2 Gbps DDR I/O speed?

- You need to run your core at half rate (e.g. 500 MHz) and add a level of serialization
- Phase compensation FIFO is a convenient place to do this

Clocking Inside Your Chiplet (Application)



With 2X mode, how does the far side correctly assemble an 80 bit word out of two 40 bit words?



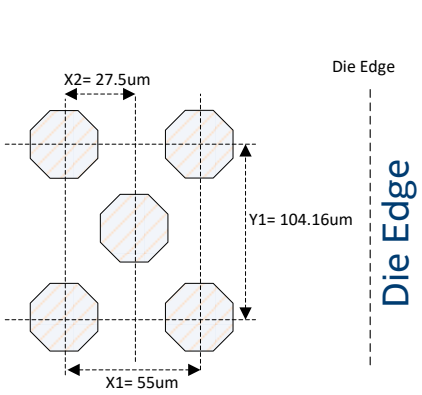
Intel® Stratix® 10 FPGA scheme:

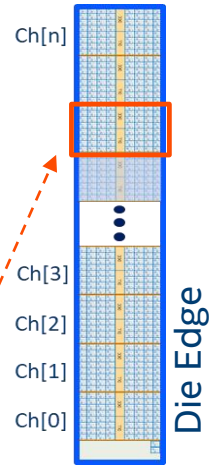
- Set bit 39 = 0, bit 79 = 1

Could devise a training mode: once the receiving side knows which is lower and upper, the sending side could use 39 and 79 for data

- Requires a training mode entry/exit and handshake… not worth it yet

AIB Microbump Array (one 55u channel)





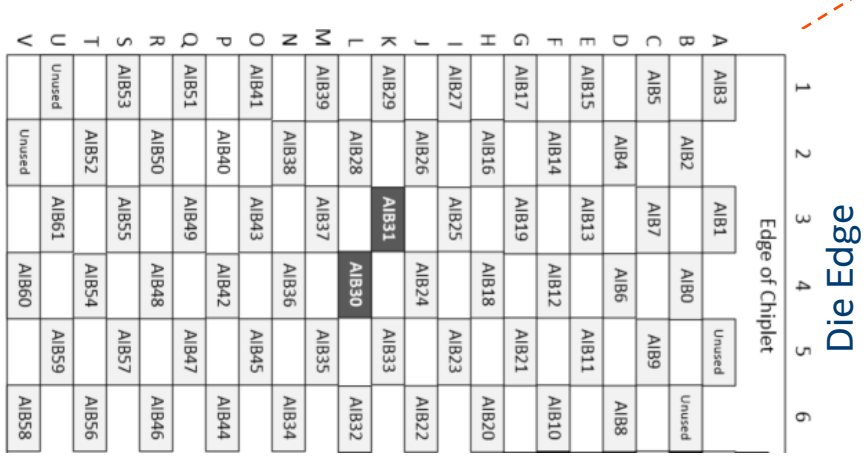


Figure 58. Low-Density Bump Map (AIB Plus, 40 I/Os, Balanced)

Bump ID	Bump Name	IO
AIB60	RX[19]	in
AIB58	RX[17]	in
AIB56	RX[15]	in
AIB54	RX[13]	in
AIB52	RX[11]	in
AIB50	fs_fwd_clkb	in
AIB48	RX[9]	in
AIB46	RX[7]	in
AIB44	RX[5]	in
AIB42	RX[3]	in
AIB40	RX[1]	in
AIB38	fs_rcv_clkb	in
AIB36	fs_sr_clkb	in
AIB34	fs_sr_load	in
AIB32	fs_adapter_rstn	in
AIB30	spare[0]	I/O
AIB28	ns_mac_rdy	out
AIB26	ns_sr_data	out
AIB24	ns_sr_clk	out
AIB22	ns_rcv_clk	out
AIB20	TX[0]	out
AIB18	TX[2]	out
AIB16	TX[4]	out
AIB14	TX[6]	out
AIB12	TX[8]	out
AIB10	ns_fwd_clk	out
AIB8	TX[10]	out
AIB6	TX[12]	out
AIB4	TX[14]	out
AIB2	TX[16]	out
AIB0	TX[18]	out

Bump ID	Bump Name	IO
AIB61	RX[18]	in
AIB59	RX[16]	in
AIB57	RX[14]	in
AIB55	RX[12]	in
AIB53	RX[10]	in
AIB51	fs_fwd_clk	in
AIB49	RX[8]	in
AIB47	RX[6]	in
AIB45	RX[4]	in
AIB43	RX[2]	in
AIB41	RX[0]	in
AIB39	fs_rcv_clk	in
AIB37	fs_sr_clk	in
AIB35	fs_sr_data	in
AIB33	fs_mac_rdy	in
AIB31	spare[1]	I/O
AIB29	ns_adapter_rstn	out
AIB27	ns_sr_load	out
AIB25	ns_sr_clkb	out
AIB23	ns_rcv_clkb	out
AIB21	TX[1]	out
AIB19	TX[3]	out
AIB17	TX[5]	out
AIB15	TX[7]	out
AIB13	TX[9]	out
AIB11	ns_fwd_clkb	out
AIB9	TX[11]	out
AIB7	TX[13]	out
AIB5	TX[15]	out
AIB3	TX[17]	out
AIB1	TX[19]	out

Table 46. Example Bump Table (AIB Plus, 40 I/Os, Balanced)

AIB Electrical Specification, Eye Mask Definition

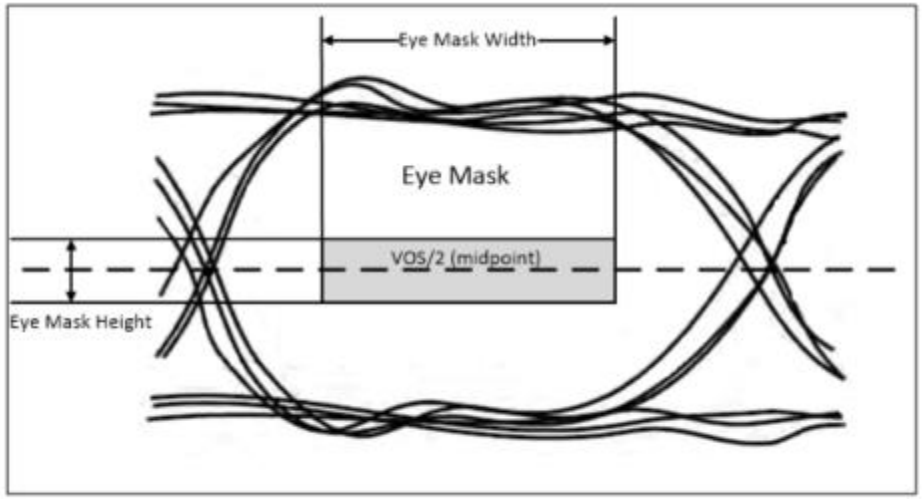


Figure 45. Compliance Eye Mask

Symbol	Parameter		Near end	Trace	Far end
V_{EH}	Minimum difference between LO and HI (eye diagram height)		±90 mV		±90 mV
V_{HI}	Minimum output voltage for HI state		0.7 V		0.7 V
V_{LO}	Maximum output voltage for LO state		0 V		0 V
V_{OSmin}	Minimum output swing		0.7 V		0.7 V
V_{OSmax}	Maximum output swing		0.9 V		0.9 V
t_{EW}	Minimum duration of valid output (eye diagram width)	Data	0.56 UI	0.1 UI	0.4 UI
		Forwarded Clock	0.56 UI	0.1 UI	0.4 UI
t_{BEW}	Minimum duration of valid output (eye diagram width) for receive-domain clock		0.66 UI	0.1 UI	0.56 UI

Table 17. Electrical signal specifications

Also DCD specs, data-to-data and data-to-clock skew specs

AIB Metrics

Metric	AIB Gen1 (Intel® Stratix® 10 FPGA)
Bandwidth/wire	2 Gbps
Wires/channel As used by Intel FPGA Specification and technology capability	 40 160
Bump density As used by Intel FPGA (defined by interposer/bridge technology)	55 micron
Bandwidth/mm of die edge shoreline As used by Intel FPGA Specification and technology capability	 256 1024
I/O voltage	0.9 V
Energy/bit	0.85 picojoule†

Data Rate

Why not push for higher Gbps/wire?

Clearly possible:

- 112 Gbps XSR SERDES, GDDR6-like 16 Gbps/wire, others with unique signaling technology

Questions to ask

- Cost
  - How much does it cost to build or license IP in your silicon process?
  - What is the technology company's business model? Monetizing the interface?
  - How much silicon area does the interface really use? (PLL, voltage regulators, independent supplies, serializers, encoders)
- Complexity
  - How do you interface your 500 MHz – 1 GHz core chiplet logic to the intellectual property (IP)?
  - How much latency from your core to the other chiplet's core?
  - How do you control training and link states?
  - How do you debug and test the technology?
- Why does multi-die heterogenous integration make sense now, when it didn't make sense before?
  - *Answer:* **High-density packaging as enabling technology!**

Channels

Use case 1: I have a 512 bit data bus each way. Why do I need to use channels?

Use case 2: I have many independent data streams, like Gigabit Ethernet links. I'd like more smaller channels.

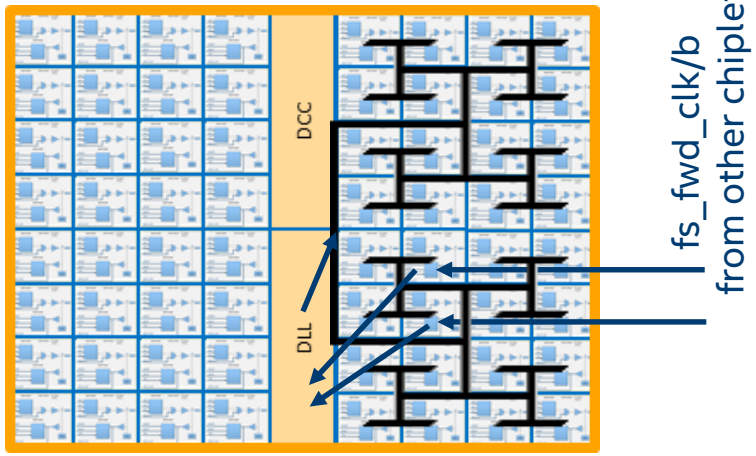
A single transmit (TX) clock must be balanced to each TX I/O cell in the channel, same with receive (RX) clock

- Clock routing and controlling skew limit the number of I/Os per clock, currently envisioned as practical up to 80 I/Os per clock for 2 Gbps
- A wide bus will need to be spread across channels

AIB, like other interfaces, has same-sized channels in an interface

- Pick too many I/Os per channel, then:
  - You either waste I/Os in low-bandwidth applications like Gigabit Ethernet
  - Or you have to multiplex links at a higher level (adds complexity, and may not work if each link needs its own clock)
- Pick too few I/Os per channel then:
  - You have to spread most higher bandwidth links across more than one channel (why is this less than optimal?)

For FPGA, 20 TX/ 20 RX per channel is practical now, 40 TX / 40 RX possible in the future



Example of AIB channel balanced clock distribution

AIB Technology Drivers

Microbump spacing

- 55 u in production
- 35 u would provide a nice doubling of density!

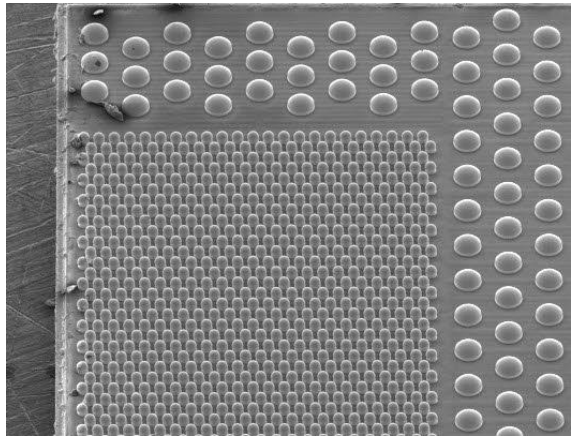
I/O voltage

- Lower voltage would drive lower energy per bit
- Compatibility with current AIB 0.9 V chiplets is very important

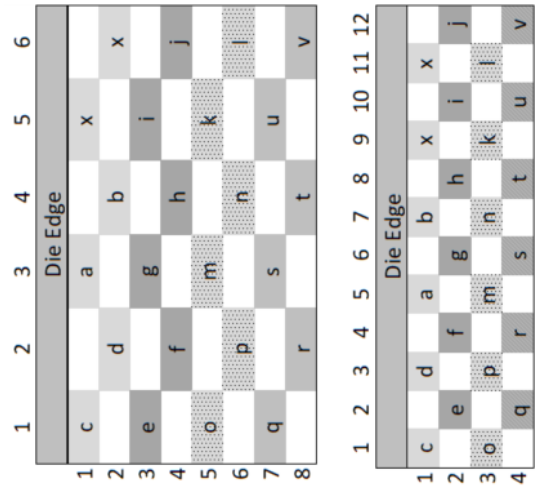
Line rate

- Very high-bandwidth applications such as direct RF sampling analog-to-digital converters (ADCs) or digital-to-analog converters (DACs) will continue to push higher bandwidth
- Doubling the data rate is feasible with careful I/O design

Microbump spacing + I/O voltage + line rate → **AIB 2.0**



Intel® Stratix® 10 FPGA 55 u microbump array



55 u to 35 u microbump assignment for straight-line routing compatibility

How to Get Started Building with AIB

Check out the AIB Open Source http://github.com/intel/aib-phy-hardware examples

- Open Source AIB adapter has helpful features, such as configuration registers and clock-crossing support

Read through the AIB specification. Think AIB Plus (2 Gbps) while you are reading.

AIB WORKSHOP: AIB OPEN SOURCE

FCCM 2019

Julie Zhang, Senior Design Engineer

AIB Open Source Agenda

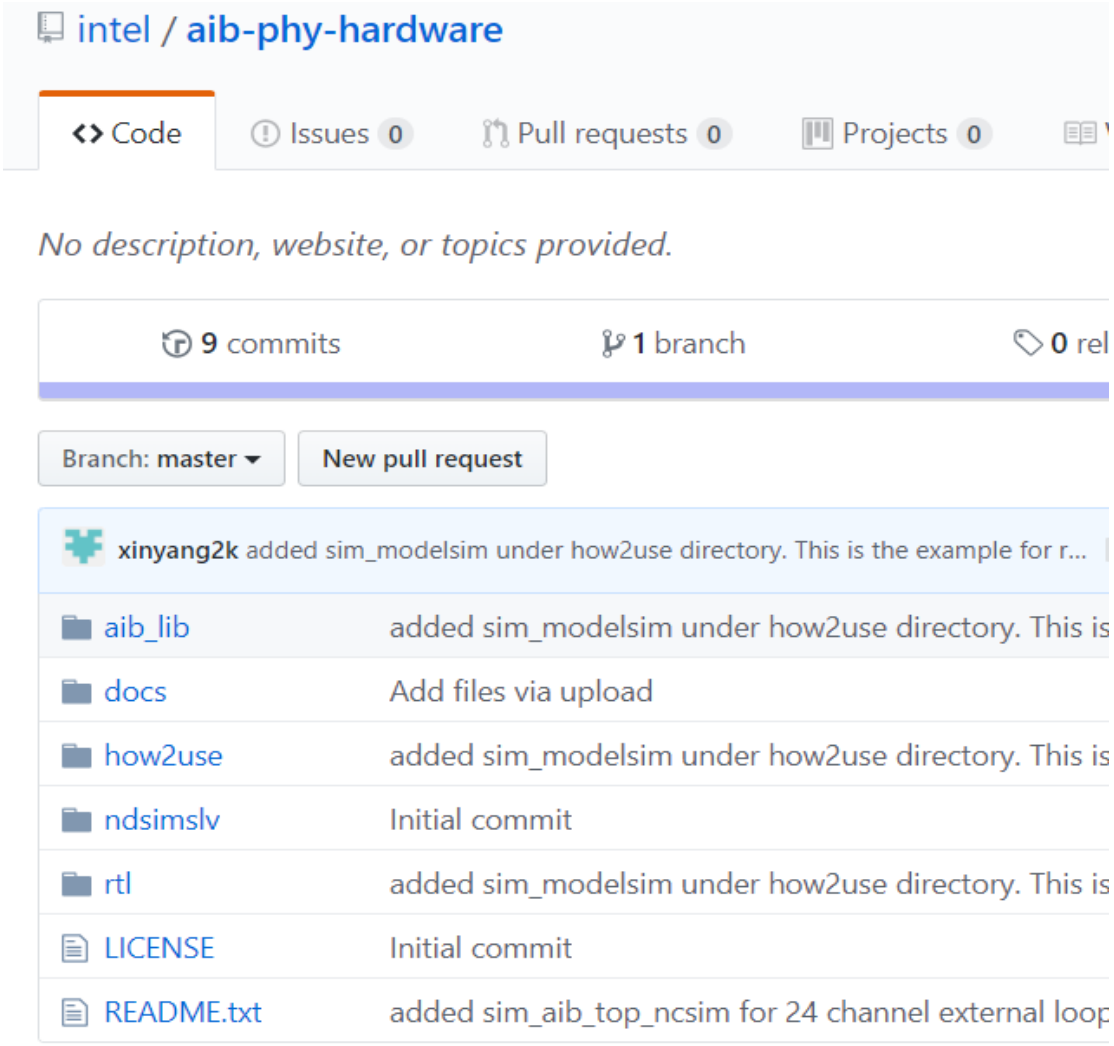
- What is in the open source release?
- AIB functional block diagram
- Dive into AIB design file structures
- AIB Adaptor
- DLL/DCC illustration in AIB I/O module
- Generic cell library
- How to use AIB with examples from single channel to 24 channels
- Call for AIB open source contributions

What is in the AIB Open Source?

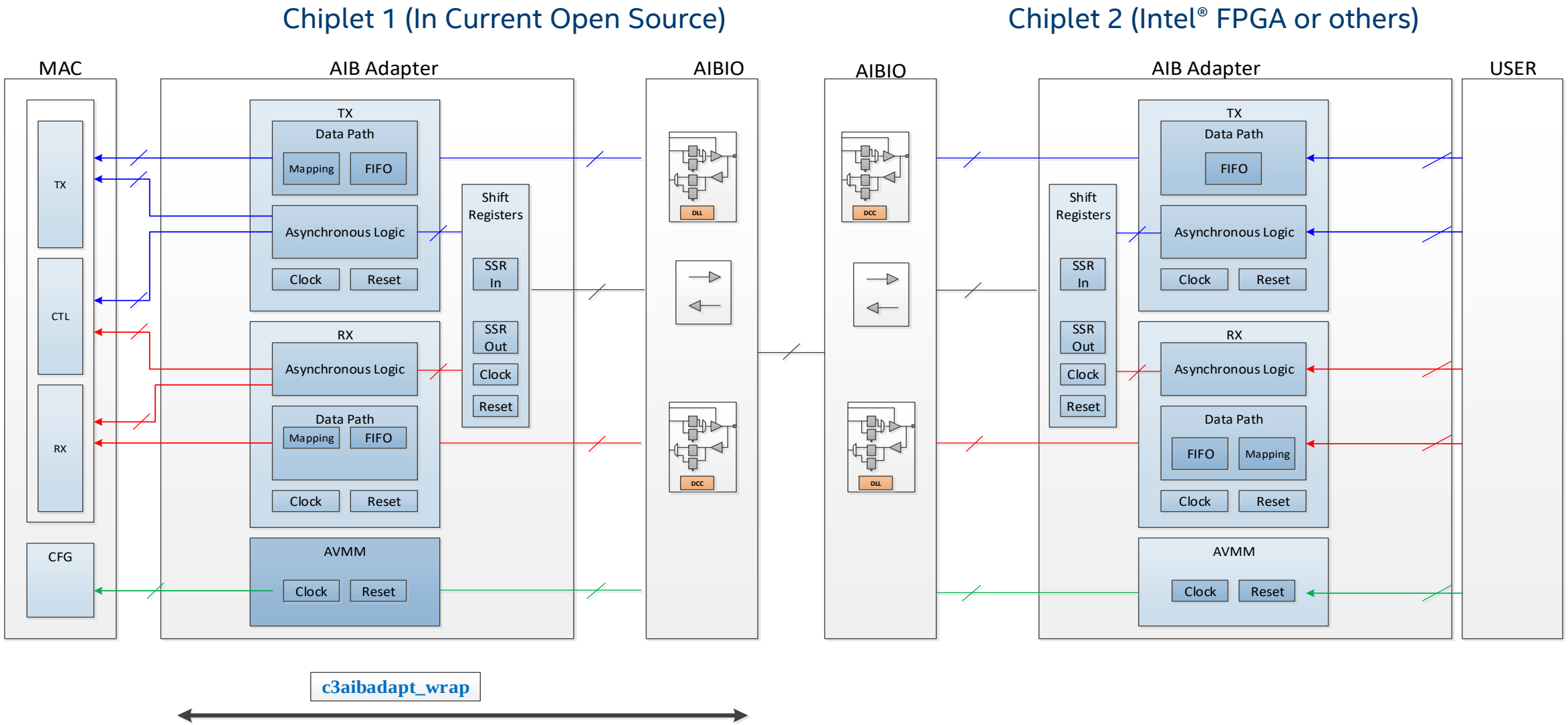
AIB open source location:

https://github.com/intel/aib-phy-hardware

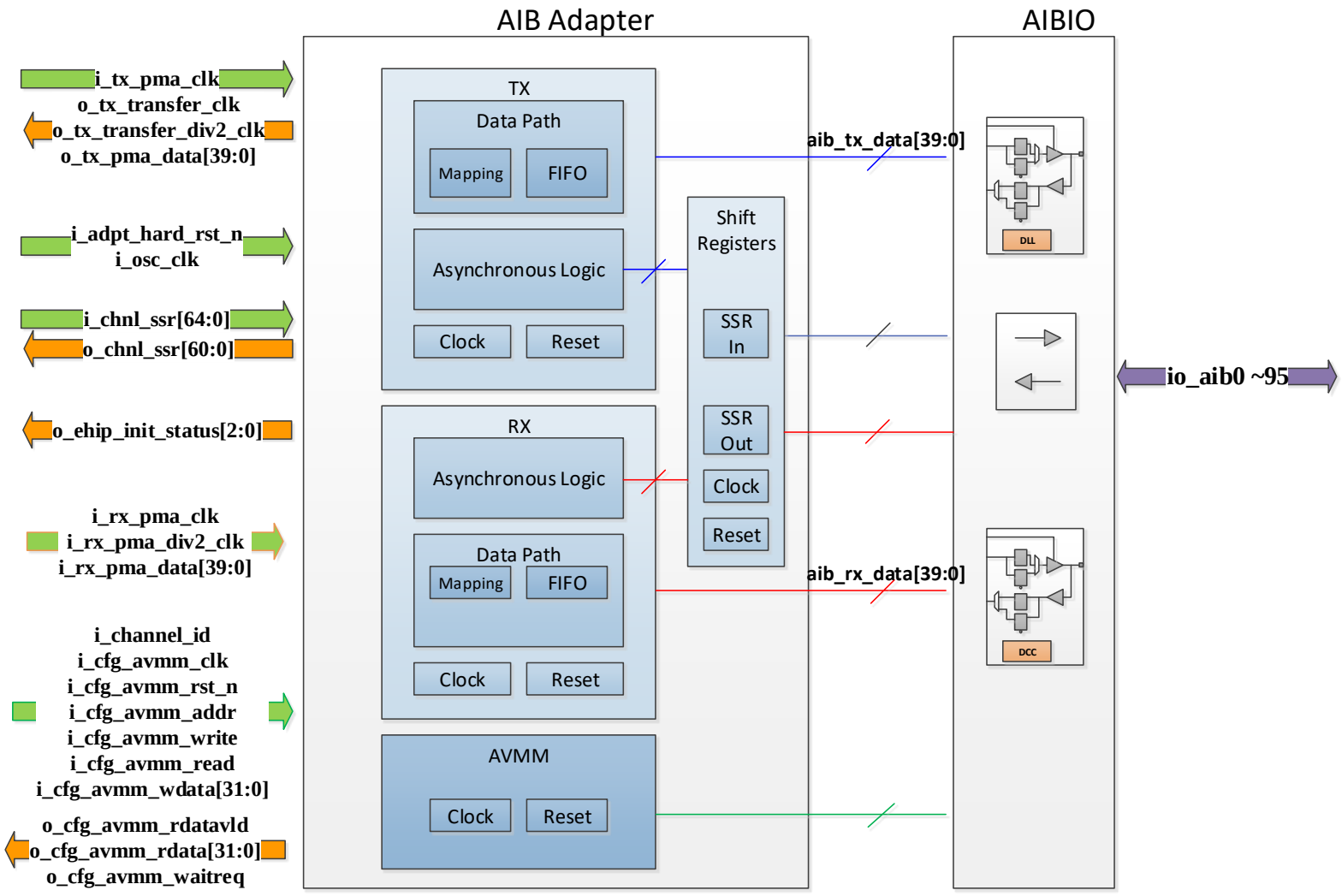
First Page



AIB Channel Interactive Block Diagram



Channel AIB Ports and Reset Sequence

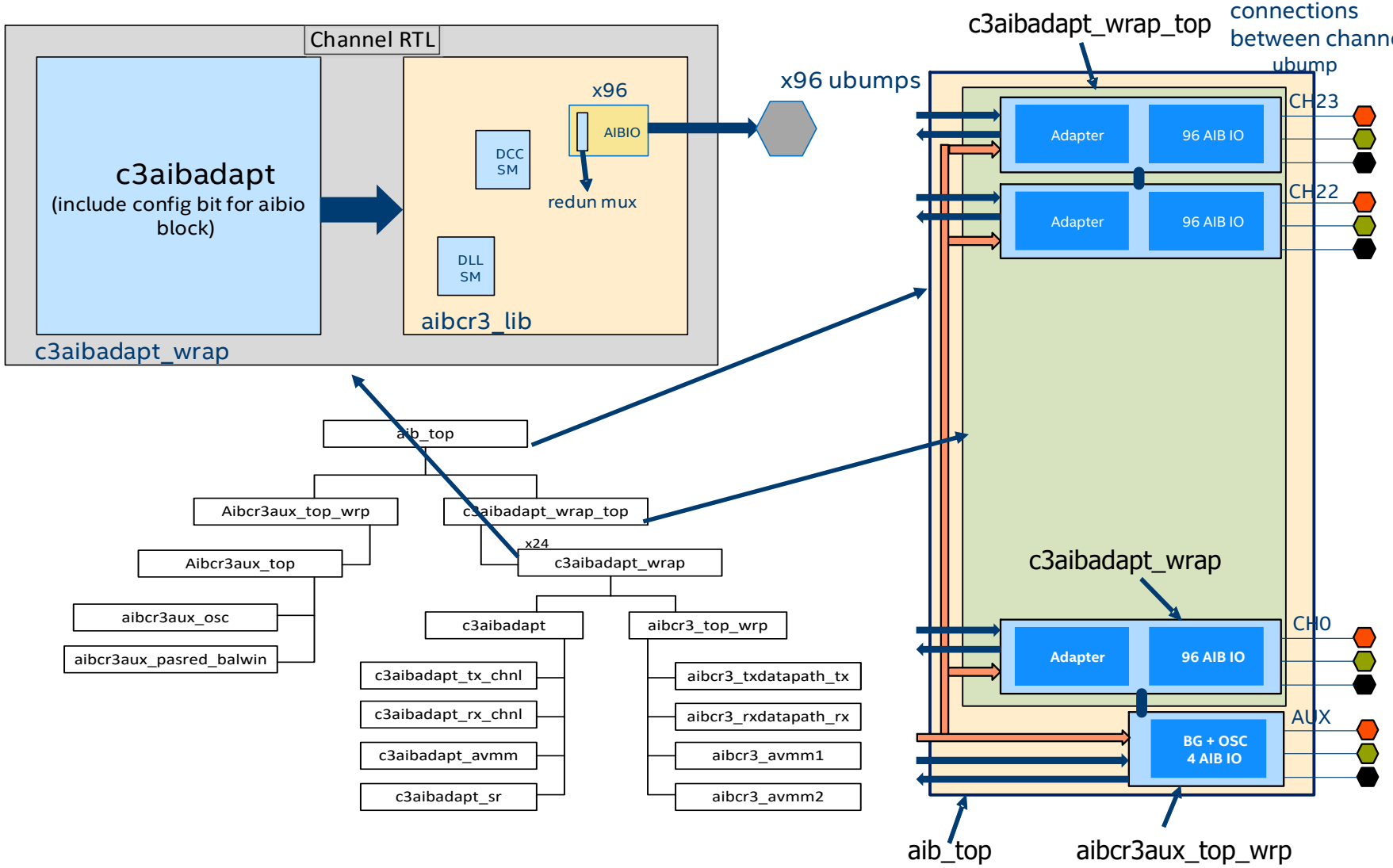


Bring up sequence:

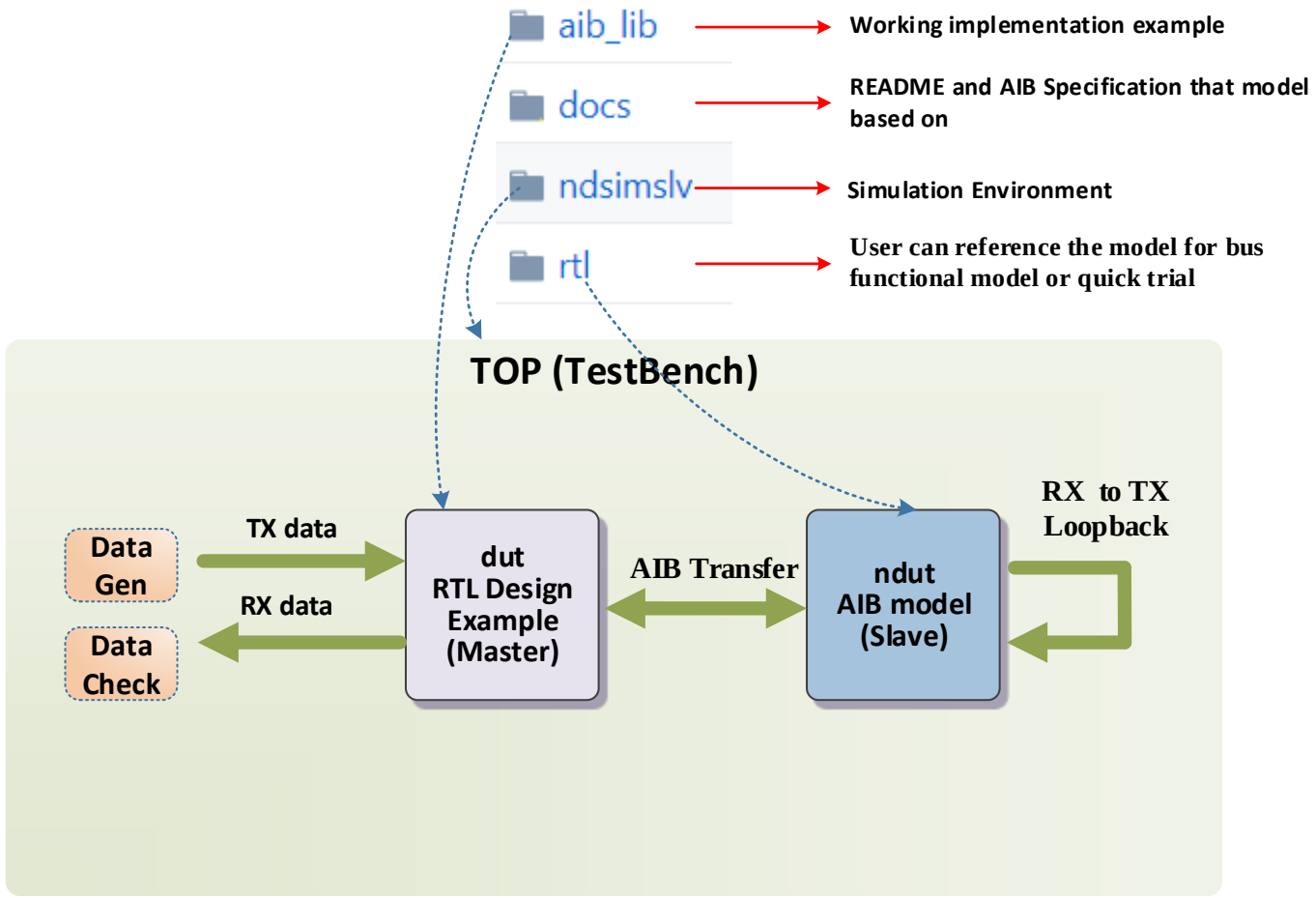
1) Avalon® Memory Mapped (Avalon-MM) interface config adapter and AIBIO after power-up.
2) i_adpt_hard_rst_n de-asserted after config.
3) Reset sequence involves the coordination oscillator/TX/RX state machine of both near-side and far-side chiplet.
4) o_ehip_init_status[2:0] = 3'b111 indicates DLL/DCC locked and the successful link-up of the parallel I/O.
5) AIB interface is ready to transfer data.

AIB Design Files Hierarchical Structure

AIB adapter and AIB I/O vertical connections between channels

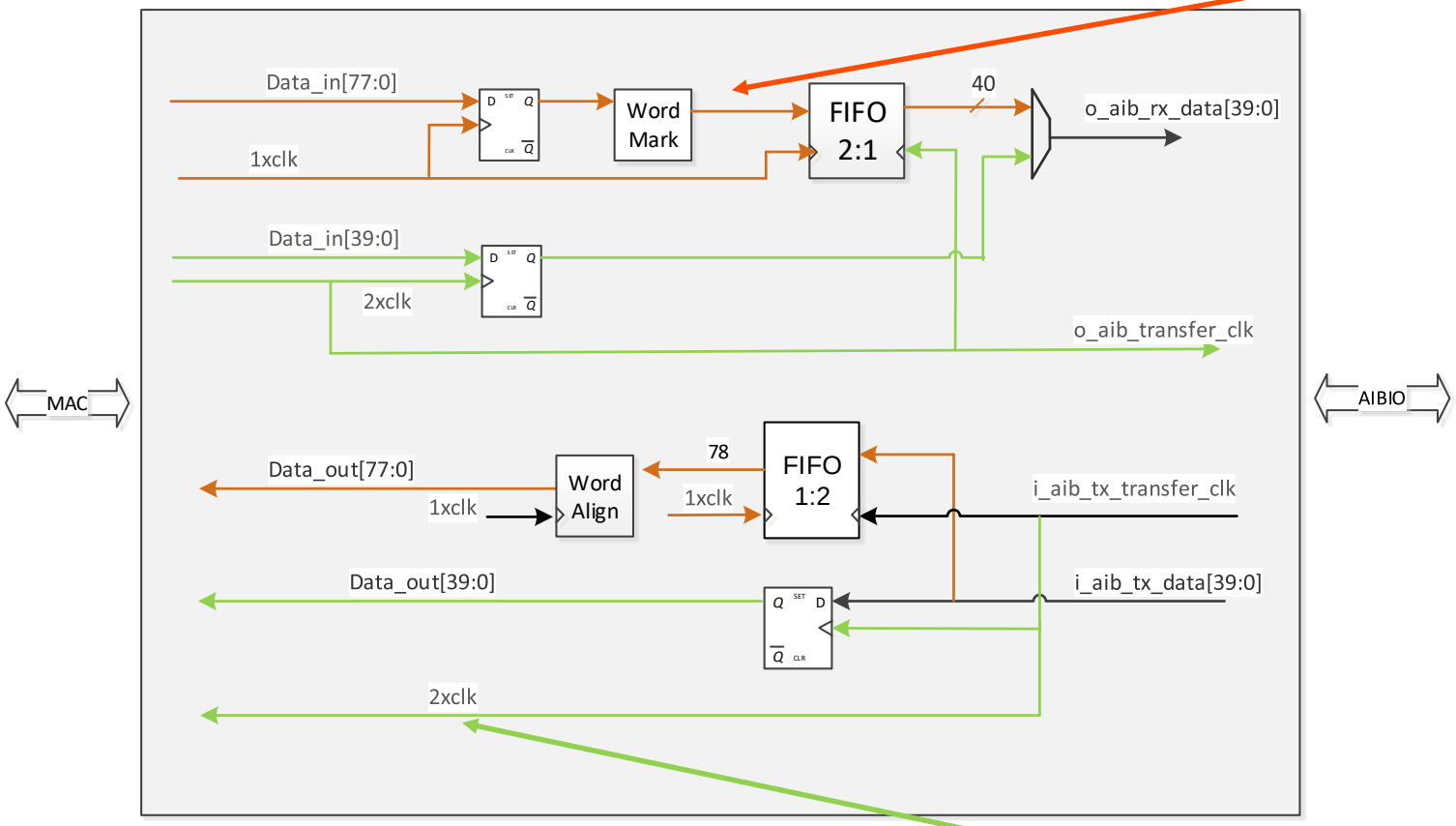


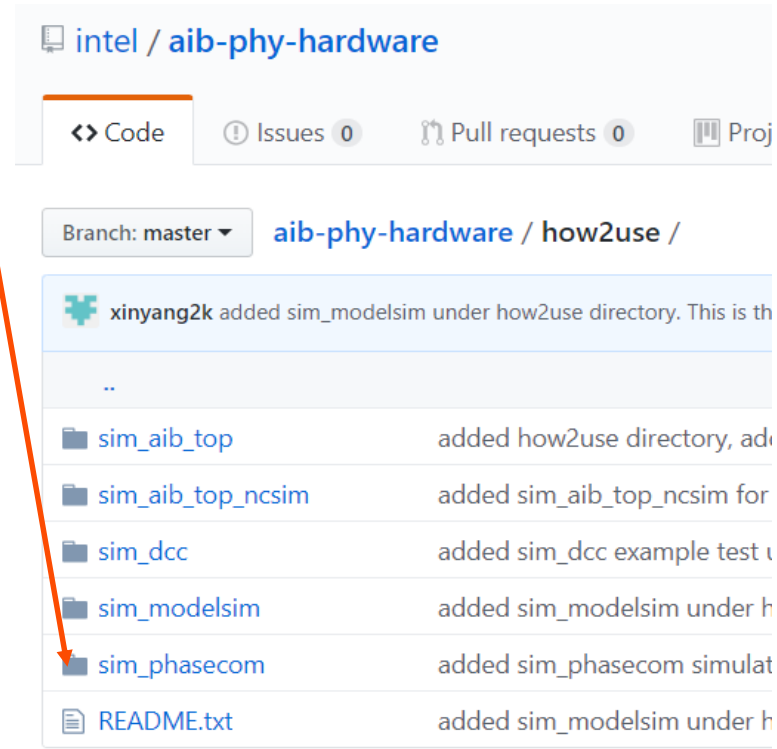
AIB Simulation with AIB Model



Adaptor Channel Datapaths

Phase comp mode
--Upper Word {1'b1, data[78:40]}
-- Lower Word {1'b0, data[38:0]}

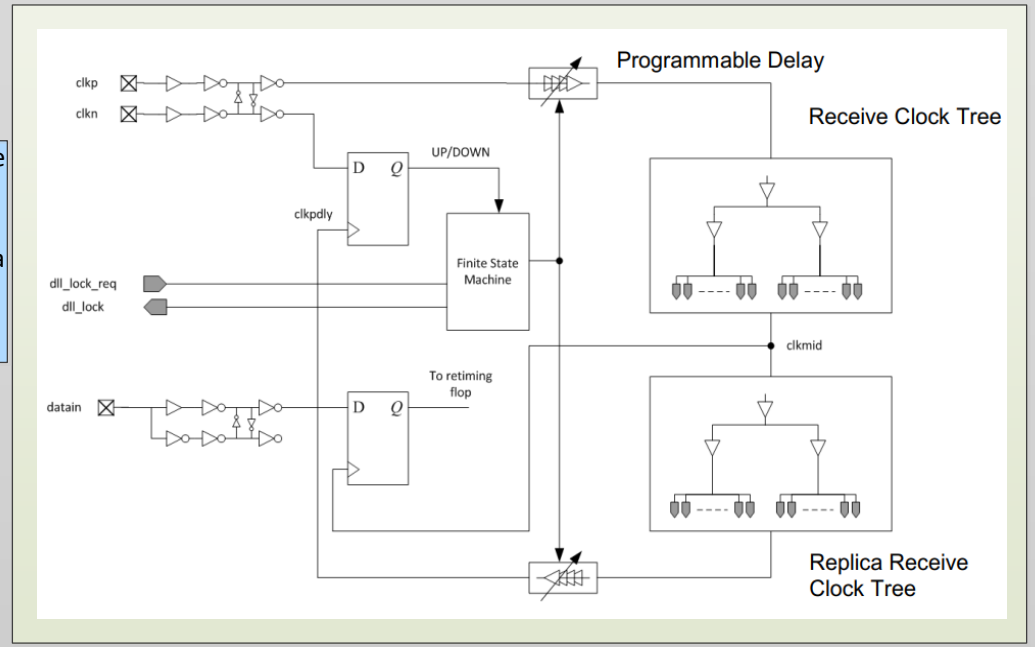


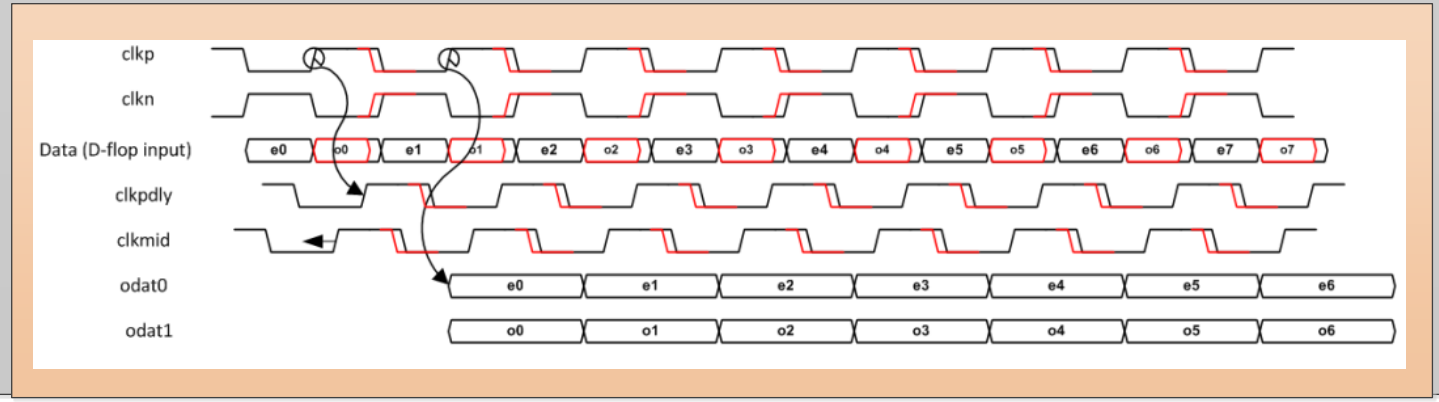


Direct register mode
(default) (40 bit)

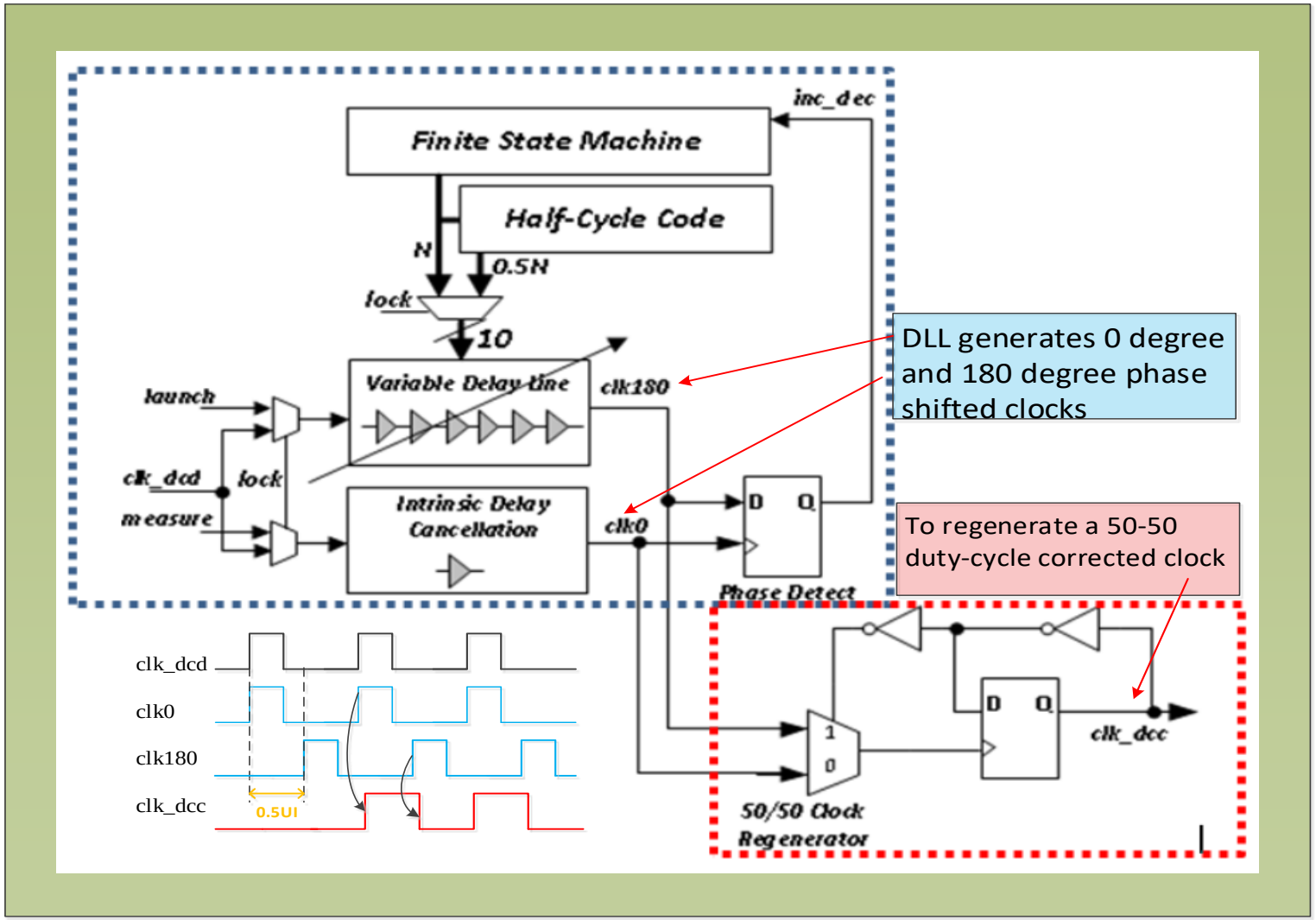
DLL in AIBIO module (Strobe Alignment)

The scheme is to place the clkmid edge at the center of rising/falling edge of clkp:
1. clkpdly 180 degree delay of clkp.
2. clkmid is used to sample the incoming data
3. clkmid is capable of tracking the distorted clock edge



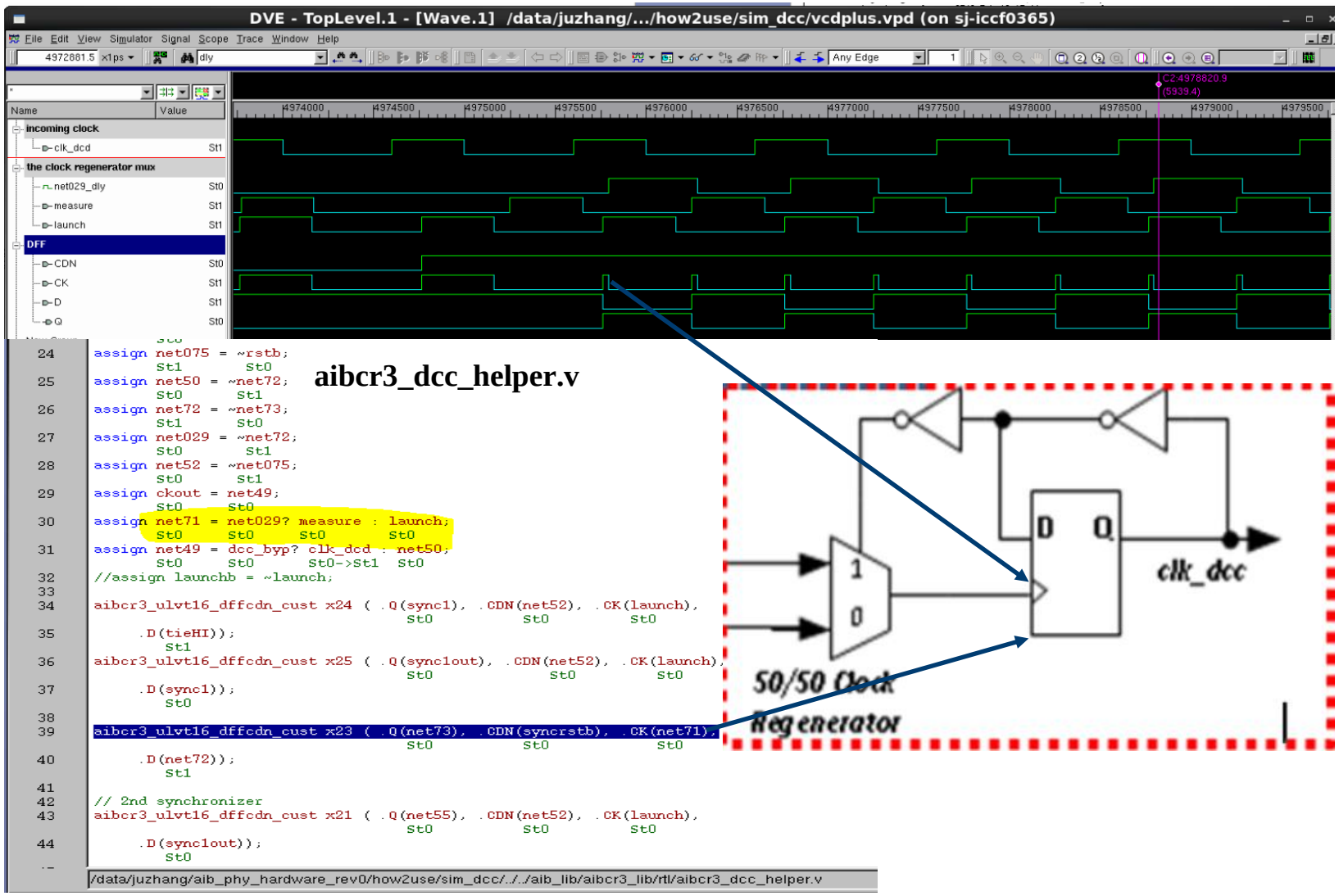


Duty Cycle Correction in AIBIO



- The data-rates will dictate duty cycle correction (DCC) enabled/disabled
- DCC is enabled typically > 400 MHz
- State machine is engaged to calibrate out the duty cycle distortion of the incoming clock

sim_dcc Example Under How2use



40/60 duty cycle clocks generated

50/50 clock generated

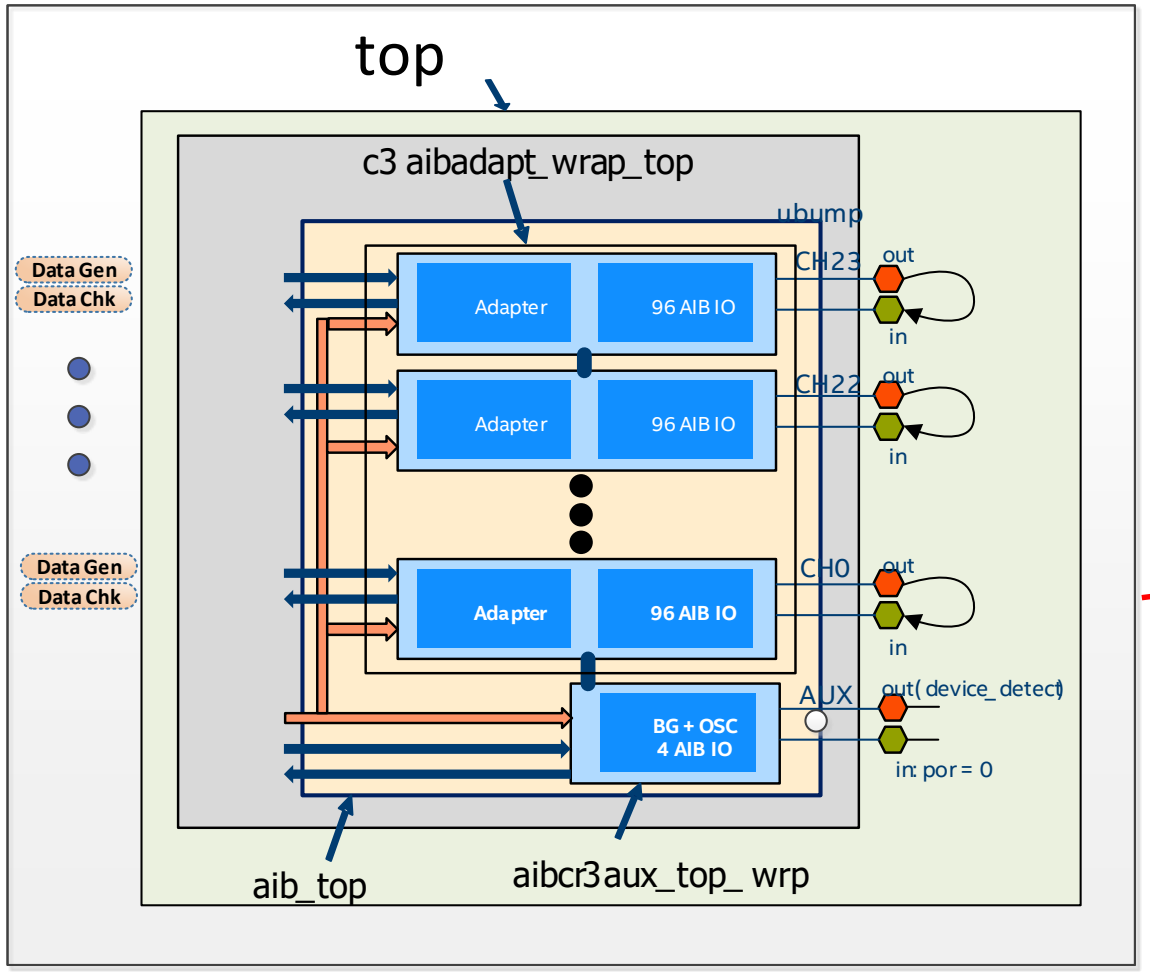
Generic Cell Library

```
// SPDX-License-Identifier: Apache-2.0
// Copyright (C) 2019 Intel Corporation. All rights reserved
// ******************************************************************************
// ******************************************************************************
// Copyright © 2016 Altera Corporation. All rights reserved.  Altera products are
// protected under numerous U.S. and foreign patents, maskwork rights, copyrights and
// other intellectual property laws.
// ******************************************************************************
//  Module Name :  c3lib_and2_svt_4x
//  Date        :  Thu Sep 15 13:44:58 2016
//  Description :  2-input AND gate (SVT, 4x drive strength)
// ******************************************************************************

module c3lib_and2_svt_4x(

  in0,
  in1,
  out

  );

input           in0;
input           in1;
output          out;

  assign out = in0 & in1;

endmodule
```

For primitive cell, in this example, the behavioral model is provided.

User needs to replace it with vendor-specific cell library.
./aib_lib/c3lib/rtl/primitives/c3lib_and2_svt_4x.sv

Example of Testing 24 Channel AIB



intel / aib-phy-hardware

Code | Issues 0 | Pull requests 0 | Proj

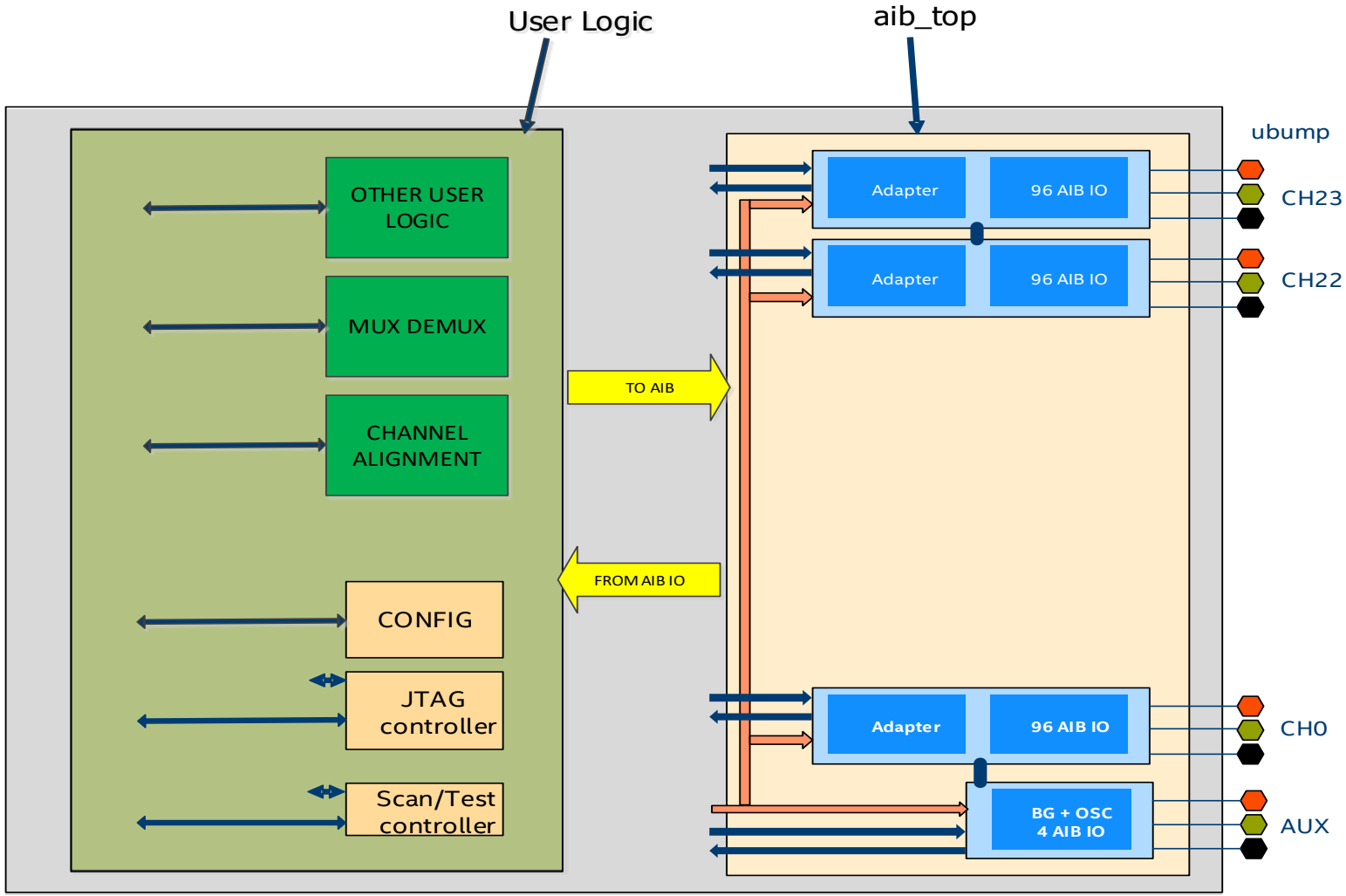
Branch: master ▾ aib-phy-hardware / how2use /

xinyang2k added sim_modelsim under how2use directory. This is th

..

sim_aib_top	added how2use directory, ad
sim_aib_top_ncsim	added sim_aib_top_ncsim for
sim_dcc	added sim_dcc example test u
sim_modelsim	added sim_modelsim under h
sim_phasecom	added sim_phasecom simulat
README.txt	added sim_modelsim under h

Example of AIB Integration By User

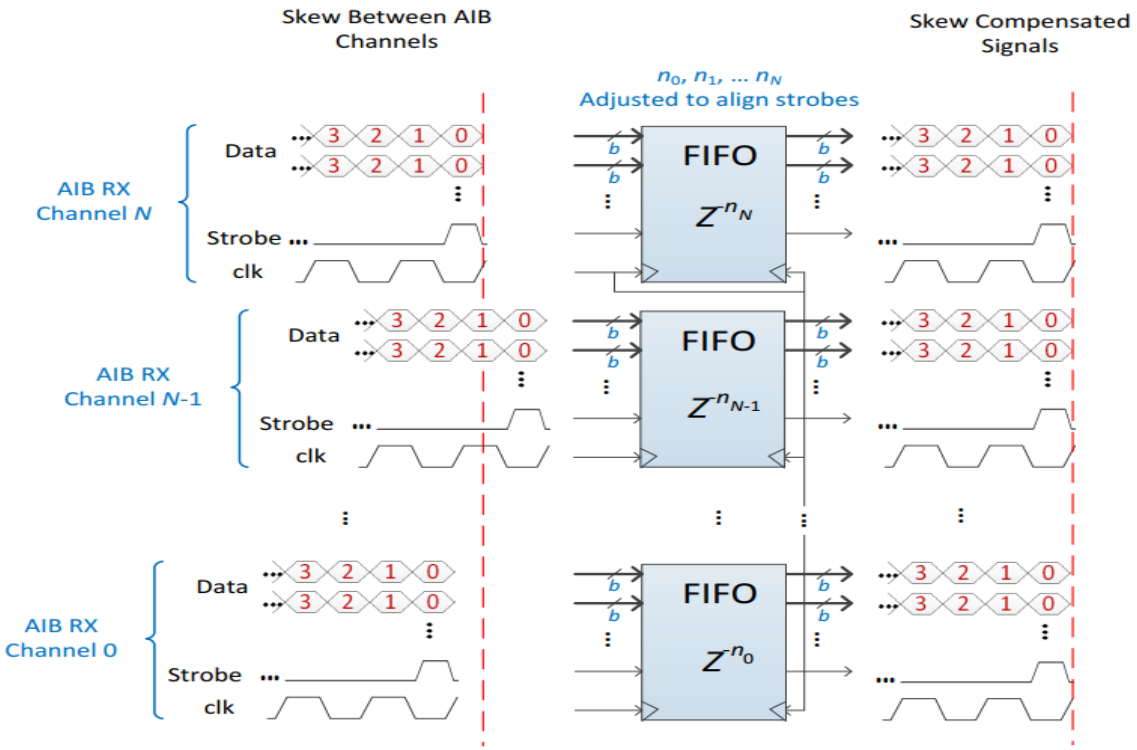


Channel Alignment (1 of 2)

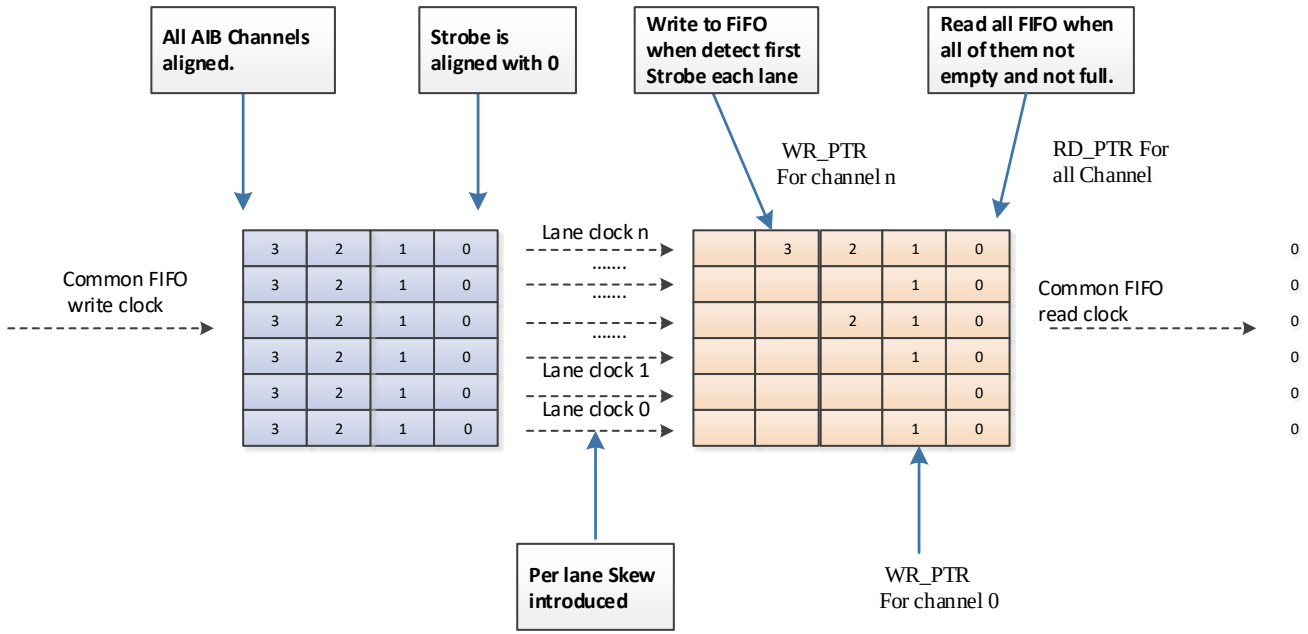
No AIB channel alignment support within the AIB channel. It is up to application to support deskewing.

A FIFO-based skew compensation scheme is addressed here:

- A strobe signal (alignment pattern) present in all AIB channels.
- The receiver uses variable delay FIFOs to align the strobe signals and the associated data bits.
- The strobe is normally zero and pulses high for one clock cycle



Channel Alignment (2 of 2)

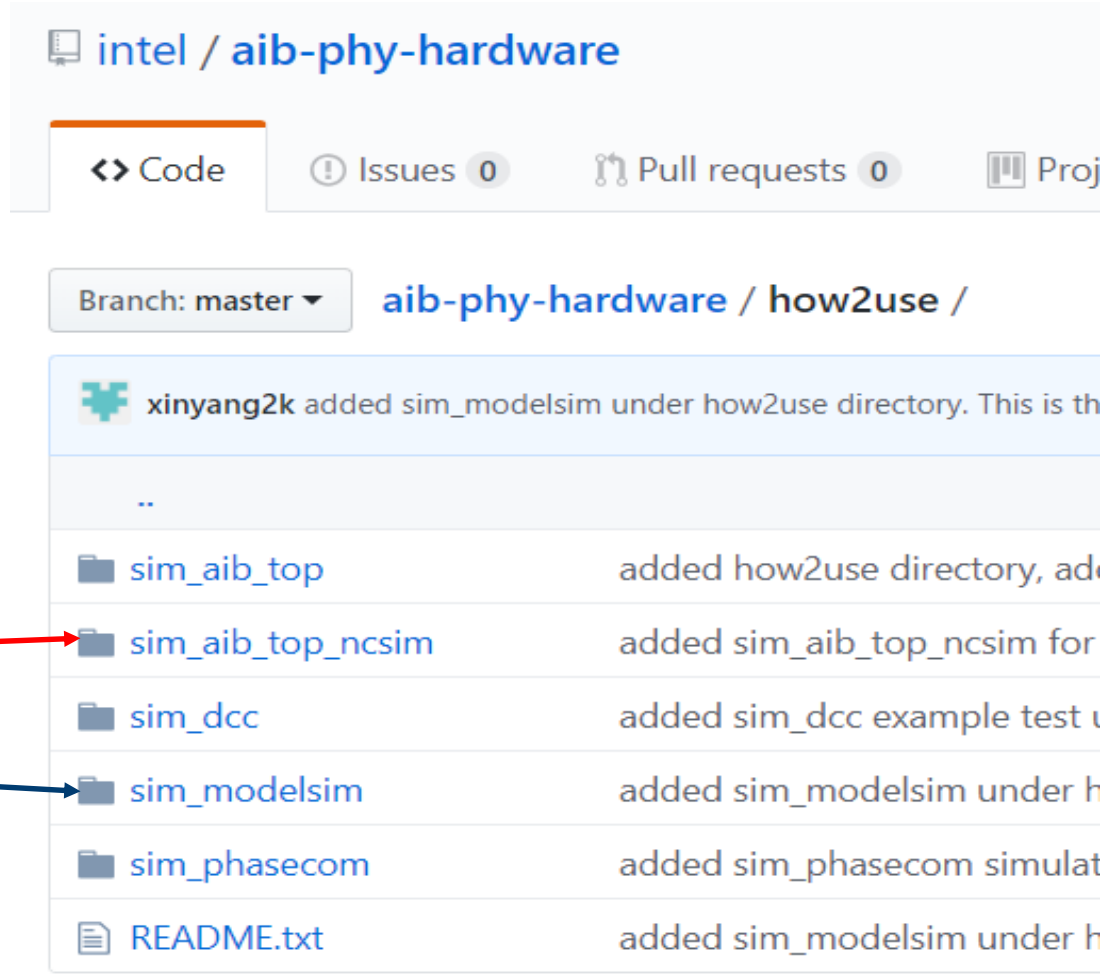


1. Strobe bit with correct interval programmed to all channels simultaneously.
2. When first “strobe” bit was detected, each channel pushes the whole word into its FIFO. Subsequent words are pushed into the FIFO.
3. When all FIFOs are not empty, i.e. there is at least one word in all the FIFOs, the read side controller will start reading. This guarantees the words read from all channels are aligned.
4. Status bits can be provided to indicate the alignment has been achieved or not.

Simulating Tools Supported and Examples

Three simulators are supported:

- Cadence* NCSIM example
- Mentor* QUESTASIM example
- Synopsys* VCS is the default simulator



AIB WRAP UP

Call for AIB Open Source Contributions

User Guide

AIB Implementation

- Wrapper to convert Open Source signal names into AIB specification signal names
- SDC timing constraints
- Different implementations of DLL, DCC

AIB System Integration

- Configuration controller with serial peripheral interface (SPI)
- Channel alignment implementation
- Simple streaming and memory-mapped protocols on AIB reference designs
- Optimized adapter for increased usability and lowest latency

Compatibility and Verification

- Simulation test suite