WHITE PAPER

Heterogeneous Integration

Accelerating Innovation Through A Standard Chiplet Interface: The Advanced Interface Bus (AIB)

Authors

David Kehlet
Research Scientist
Intel® Programmable Solutions Group

Introduction

The semiconductor industry has been on a decades-long quest to place as much functionality as possible on a single die. For most of that time, a monolithic implementation has provided the best combination of performance, power, and capability, as compared to connecting two chips together using the packaging and interconnect technologies available at the time.

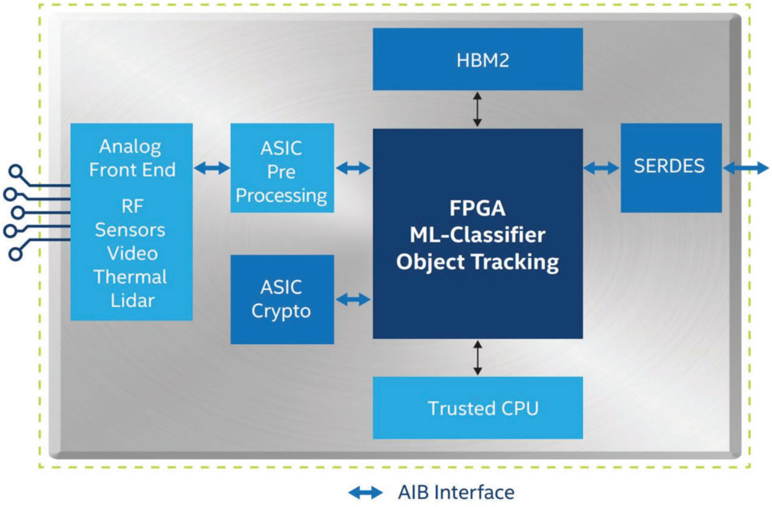


Figure 1. An example of AIB application, where the analog front-end, signal pre-processing, and SERDES are connected, all by AIB, to an FPGA implementing classification and object tracking.

But new integration technologies involving silicon bridges, interposers, aggressive geometries, and micron-scale microbump connections have changed the calculus. Back in 1965, Gordon Moore noted that, "...It may prove to be more economical to build large systems out of smaller functions, which are separately packaged and interconnected." More than 50 years later, we're achieving Moore's heterogenous integration vision.

Many of today's SoCs resemble each other in core processing while differing in specific peripheral functions. One application may need vision processing; another application requires taking signals directly from an antenna; yet another application needs more memory than was possible on an SoC. Part of the value of separating these functions out is in mixing and matching capabilities, but another important part is that each function – processing, analog, memory, digital signal processing (DSP) – may be better optimized on a different process from the one used for the core computing.

Table of Contents

Given these separate pieces– which are implemented on so-called chiplets or tiles – the challenge is to interconnect them all in a single package while keeping performance and power close to what would be possible if they were all monolithic. That challenge is being met with the combination of the Advanced Interface Bus (AIB) with packaging technologies that allow heterogeneous integration of multiple die into a single package. The AIB interconnect scheme, recently announced, reflects an effort to provide a high-speed, layout-friendly, flexible way of interconnecting chips and chiplets.

This white paper describes the high-level characteristics and usage of AIB. The AIB specification is public and can be accessed at http://github/intel/aib-phy-hardware. We will show how the various AIB features support the design and manufacturing of reliable, high-speed connections with high yields.

The AIB Objective

Device-to-device interfaces over the last 25 years have advanced by using complex circuits to push high speed through a few wires; PCI Express* is one such example. AIB reverses this trend, using a very wide parallel interface supported by new high-density packaging technology. By running each wire of the interface at a relatively low speed, the circuitry for each transmitter and receiver is greatly simplified and uses very little silicon area.

AIB moves data from microbumps on one chiplet to microbumps on another adjacent device. The very fine pitch of new high-density packaging microbumps keeps the real estate required for the interface modest. High-density packaging technologies typically support microbumps at 55-micron spacing, compared to standard flip-chip packaging that uses bumps spaced 130 or 150 microns apart.

This very fine pitch permits a single AIB interface to use thousands of signal wires, compared to a traditional interface like DDR memory on standard technology that could reasonably use only a few hundred wires. Individual AIB data wires are clocked at GHz clock speeds, with numerous configuration and speed options to ensure that AIB can support a wide variety of applications.

While AIB does not specify a maximum clock rate, and the minimum is very low (50 MHz), AIB shines at high bandwidth and the typical data rate per wire is 2G bits per second. Each chiplet documents its intended range of clock rate so that a designer selecting different devices can ensure that they operate at compatible speeds. In general, it is intended that clocks operate at or below 1 GHz, but higher speeds are allowed as long as both sides of the interface support those speeds.

AIB is a physical-layer (PHY) specification; it occupies the lowest level in the OSI Reference Model. On one side it connects to a corresponding AIB interface on a separate chip or chiplet; on the other side it connects to the Media Access Controller (MAC). It is solely intended to take data from the MAC and send it out to the connected chip or receive signals from the connected chip and hand them to the MAC.

The footprint is designed to be as small as possible within the limits of microbump pitch. Signals are clustered together for more efficient use of the edge of the die – referred to as the shoreline – and to provide fast, short, low-skew signal wires. As the data rate of each wire in an AIB interface is 2 Gbps for AIB Gen1, training and signal conditioning such as equalization and pre-emphasis are avoided to keep the circuit size small. Common microbump pitch in the industry is 55 µm with future AIB support planned for as low as 10 µm as bumping technology evolves.

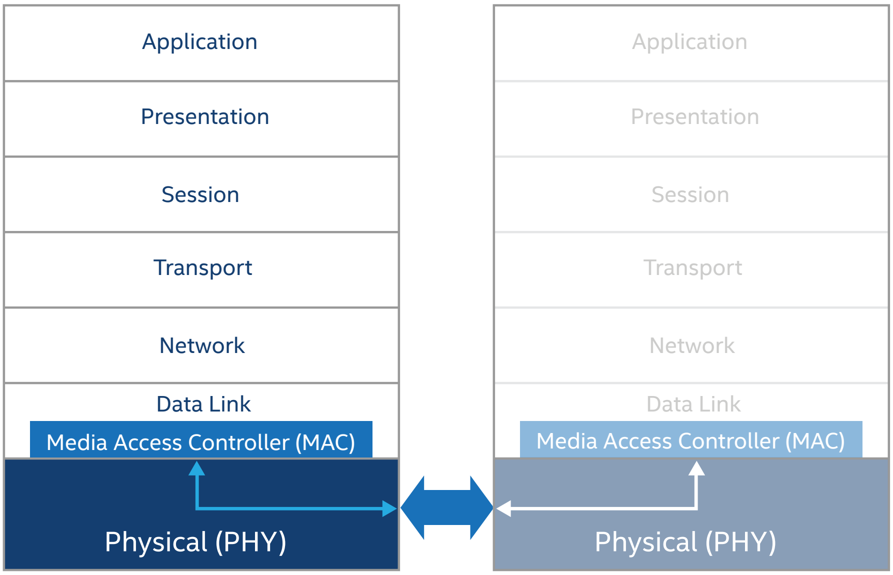


Figure 2. AIB is a physical-layer specification.

AIB Configurations

There are two fundamental configurations of AIB. AIB Base is intended for lighter-weight implementations requiring a minimum of circuitry. AIB Plus handles higher speeds and has features for reliable operation at those high speeds.

The biggest difference between the two configurations relates to data speed. AIB Base signals using a single-data-rate (SDR) scheme, meaning that new data is transferred on one edge of the clock. AIB Plus supports double-data-rate (DDR) signaling in addition to SDR. With DDR, data is transferred on both edges of the clock, doubling the data rate as compared to SDR.

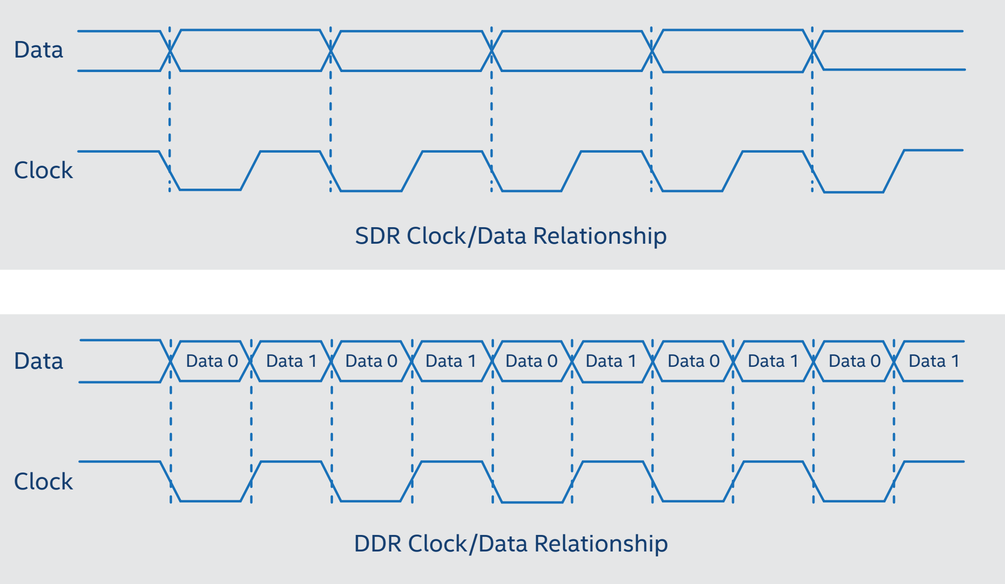


Figure 3. SDR signals clock on every other clock edge - in this example, the falling edge. DDR, by contrast, clocks on both falling and rising edges for twice the data rate.

Because of the DDR capability, AIB Plus interfaces can move data in the range of 2 Gbps. But maintaining signal skews and tight timing becomes more difficult in this range. Delay-locked loops (DLLs) help adjust phase relationships, and duty-cycle correction (DCC) circuits help ensure as close to a 50% clock duty cycle as possible. Initialization and calibration of these circuits are provided to ensure smooth bring-up and operation. The following table summarizes the differences between AIB Base and AIB Plus and the features are described below.

CAPABILITY	AIB BASE	AIB PLUS
SDR (nominally 1 Gbps)	X	X
DDR (nominally 2 Gbps)		X
Phase and duty-cycle adjustment (DLL, DCC)		X
Signal retiming option		X
Clock forwarding	X	X
Transmit clock provided by receiving chiplet		X

Table 1. A Comparison of AIB Base and AIB Plus Capabilities.

The AIB Architecture

An AIB interface comprises I/Os that are grouped into channels, which themselves may be stacked into a column. A column consists of 1, 2, 4, 8, 12, 16, or 24 identical channels. A channel can have up to 160 I/Os for 55-µm microbumps; that number will go up with decreasing bump pitch.

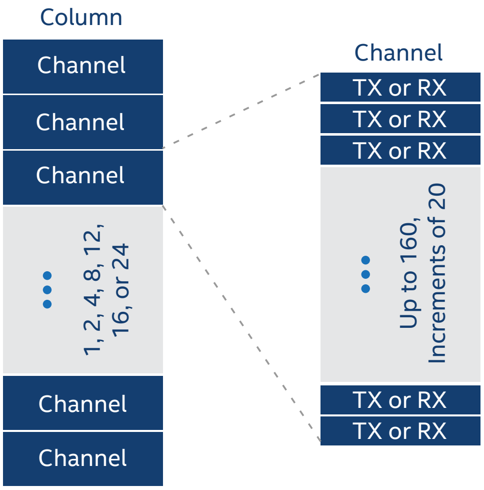


Figure 4. A column is made up of up to 24 channels; each channel consists of up to 160 I/Os.

The I/O blocks are illustrated below, depicting transmit (TX) and receive (RX) blocks in both SDR and DDR (AIB Plus only) versions.

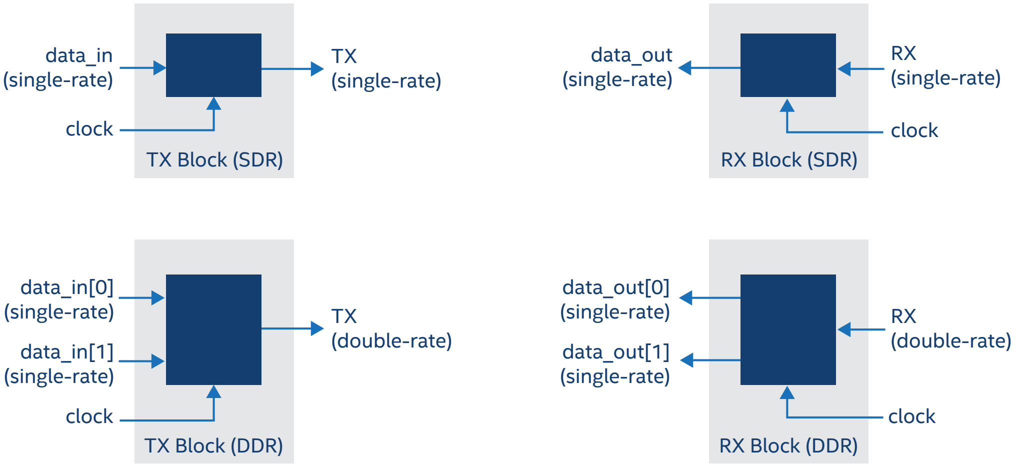


Figure 5. A comparison of SDR and DDR (AIB Plus only) TX and RX blocks.

The I/O scheme is intended to be simple while allowing for two primary arrangements. A given I/O will be either a TX or an RX signal; there are no bidirectional signals (with the exception of loopback options for testing). When configuring a channel, one can have all TX, all RX, or half-and-half TX/RX. This scheme provides flexibility for dealing, on the one hand, with chiplets that may be taking inputs and passing them on to another chip (meaning, on one side, it will be an interface from an all-TX version to an all-RX version). On the other hand, where a chiplet returns a result over the same interface over which it received its input – like a memory – then a balanced TX/RX interface can be used.

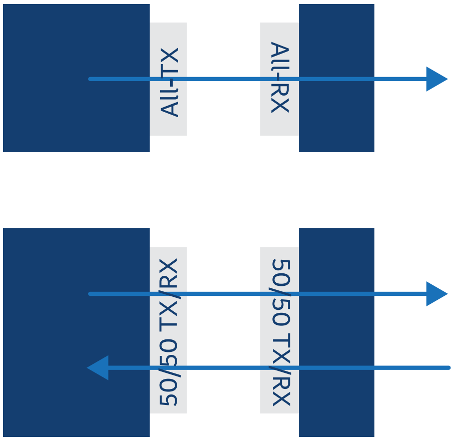


Figure 6. A channel can have all TX, all RX, or half-and-half TX and RX signals.

AIB has two interfaces: one through the microbumps to the corresponding AIB interface in a nearby chiplet, and one to the MAC code inside its own chiplet. The first interface consists of the I/Os, forwarded clocks, and control signals used during initialization and calibration.

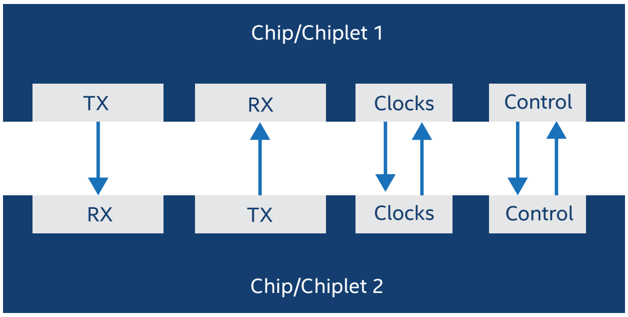


Figure 7. The AIB interface to another AIB interface consists of TX, RX, clock, and control signals.

The MAC interface consists of signals performing the same functions as the external interface, although the details and specific electrical format of those signals will be different. For example, a clock signal will be received from the MAC as a single-ended internal signal, while that corresponding clock signal will be sent across the external interface to the adjacent chiplet as two SDR signals (that is, a double-ended clock).

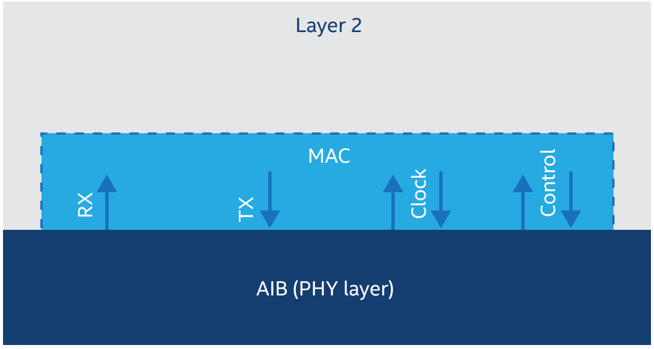


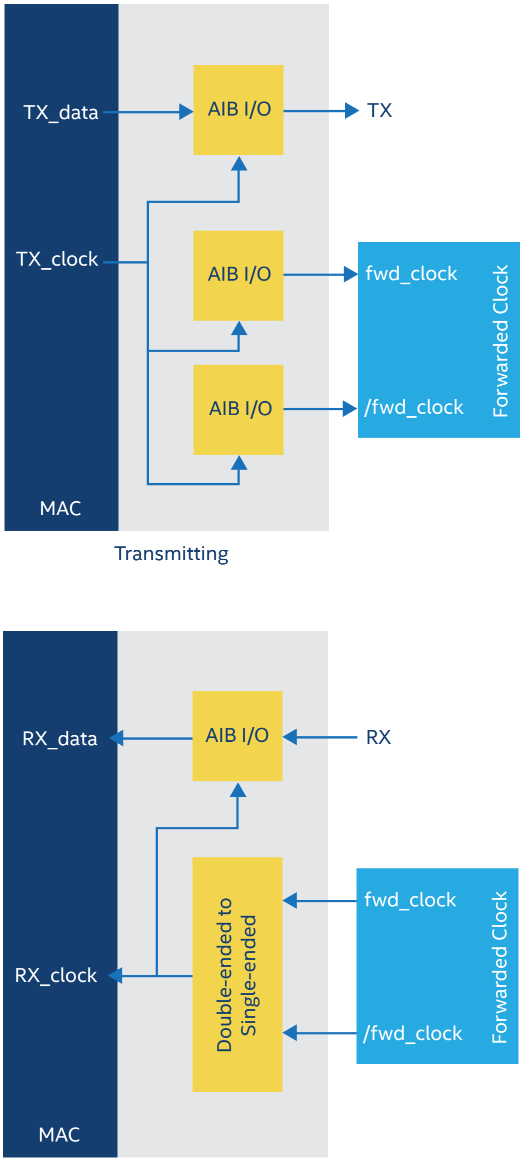
Figure 8. The AIB-to-MAC interface contains similar signals as the AIB-to-AIB interface, although formatted differently.

Features for High Data Rates

At GHz speeds, timing is tight. Signal-to-signal skew matters, and clock-to-signal skew and jitter also matter. Where both edges of the clock are used for clocking data (DDR), the duty cycle is also critical. For this reason, several features have been included in the AIB block – particularly for the AIB Plus configuration, which supports double data rates.

Forwarded Clocks

To ensure the successful receipt of data in a receiving AIB block, the clock used for transmitting the data is forwarded to the receiving side where it can be used for data capture. This clock will run all the way into the MAC, so there isn't necessarily any clock domain change within AIB – although there may be one within the MAC. This feature is available in both AIB Base and AIB Plus configurations.



Transmitting

Receiving

Figure 9. Both AIB configurations provide clock forwarding, where the TX clock is sent as double-ended to maintain low skew with respect to the data being transmitted.

The clock signal is forwarded as a double-ended clock, with both true and inverted versions sent for reconversion back to a single-ended clock on the receiving side. This maintains the quality of the clock signal, since common-mode noise on the clocks will wash out when the double-ended clocks are recombined. Edge alignment is ensured because I/O cells are used to send both versions of the clock as well as the data.

Receive-Domain Clock

Some chiplets may not have their own independent clock source, preferring instead to leverage the clock of the chip or chiplet to which it's connected. For example, a memory may simply run on the clock of the CPU chip that's accessing the memory.

Continuing with the memory/CPU example, when the memory receives data – say, an address for data to be fetched – the forwarded clock keeps the memory in synch with the CPU chip. But when the memory sends back the fetched data, it needs a clock, and the forwarded clock affects only receive capture, not transmission.

For this reason, the CPU chip clock – which, in this example, will be receiving data from the memory – can be brought over for use as the memory's transmit clock. This clock is referred to as the receive-domain clock. It's available only on AIB Plus interfaces.

In the following figure, you can see a TX cell using the receive domain clock. In this example, that clock is sent to the MAC, where it is turned back and used for the transmit clock – which is then forwarded back to the receiving side. Because the forwarded clock in this case is actually the same as the original receive clock, this might seem inefficient. But the diagram is slightly misleading in that all of the various clock signals aren't quite identical: they'll vary in phase. By taking the receive-domain clock and forwarding it back to the receive side, you are assured edge alignment between that clock and the data being transmitted. Note that, while the MAC receives the receive-domain clock, it need not use it for transmitting data; it can have a separate clock domain for that. What's illustrated below is but one possible configuration.

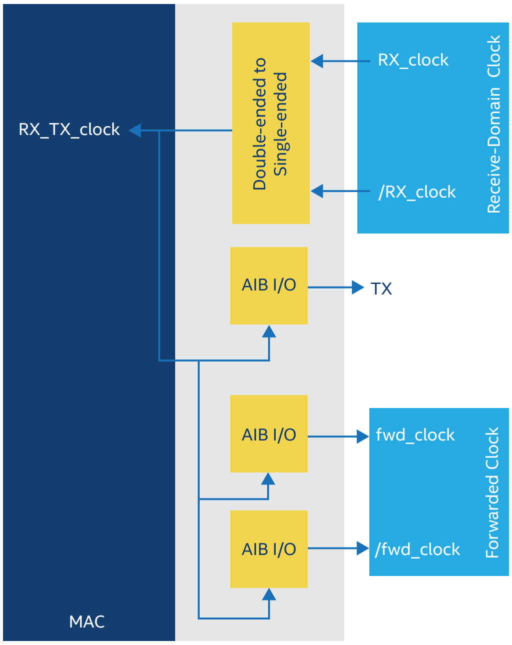


Figure 10. AIB Plus interfaces can transmit using the receive-domain clock (which is then forwarded back to the receive domain).

Duty-Cycle Correction

For DDR data exchange, the duty cycle for the clock, by specification, cannot vary by more than 3%, since both edges are used to clock data. At 1 GHz – meaning 2 Gbps data – that's an extremely tough specification to meet without some help. As a result, AIB Plus specifies a duty-cycle correction (DCC) block. Technically, that block is optional, but, from a practical standpoint, it's most likely going to be needed.

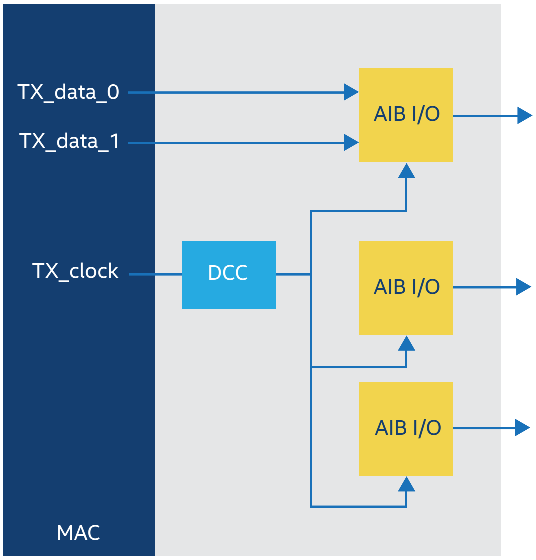


Figure 11. A duty-cycle correction circuit helps to meet a tight duty-cycle spec for DDR data exchange on AIB Plus interfaces.

Forwarded Clock Phase Adjustment

In a similar vein, on the receiving side of a DDR connection, the forwarded clock may have acquired some additional skew between the sending and receiving chiplets. Subtle phase shifts can cause problems at this speed, so a delay-locked loop (DLL) is specified for AIB Plus. Again, it's an option, but it's most likely that it will be needed to ensure robust operation over all conditions.

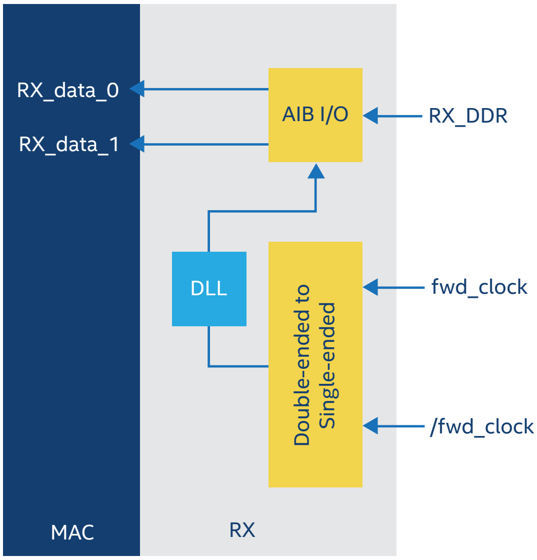


Figure 12. A delay-locked loop helps correct for any clock-phase distortions that would otherwise limit operating speed on AIB Plus interfaces.

Retiming

At double-data-rate speeds, data path timing can be tough to meet in an ASIC or FPGA driving data out through the AIB interface. AIB Plus implementations have the option of a retiming block ahead of the I/O block. There is flexibility as to how this retiming may work. It could be as simple as one or two registers simply to break up the data path and make it easier to achieve timing closure on the complete circuit. Or one might want to go further and include clock-phase compensation by implementing a FIFO. Intel® FPGA applications of AIB generally use a phase-compensation FIFO. Even if a chip uses the same reference clock for the core and the I/O, the phase difference between those clocks may vary substantially or be unknown. A phase-compensation FIFO assures correct clock-phase domain crossing. In general, the details of how this retiming happen aren't specified by AIB, but it's very likely that at least one retiming flip-flop will be needed.

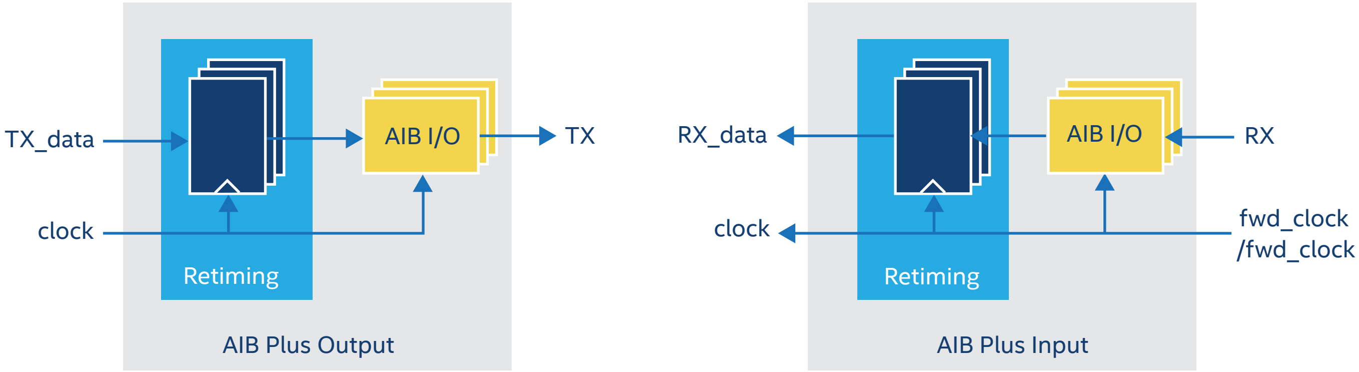


Figure 13. An optional AIB Plus retiming block can help ease ASIC or FPGA timing closure.

AIB's Physical Arrangement

AIB has been designed in a way that makes it straightforward to connect chiplets. Signals within a channel are striped in bump rows. The number of bump rows is kept to a minimum, except that wire length and routing must be accommodated. That balance has been struck in the assignment of AIB signals to microbumps. Bump assignments have also been made in a way that should make the wires between bumps more or less equal in length, minimizing interposer-induced signal skew.

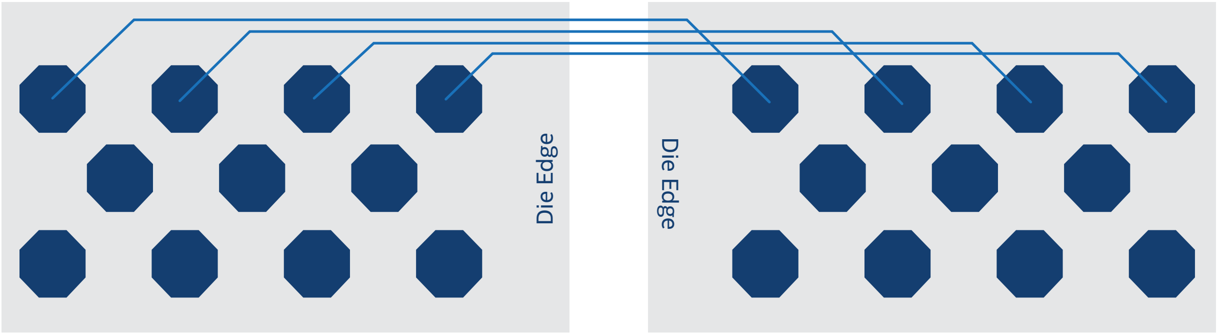


Figure 14. 55-um microbumps are arranged in staggered rows. Bumps are assigned to keep wire lengths short but equal.

The signals are all collocated, minimizing the impact of the interface on bump placement. For example, no other non-AIB signals will have bumps placed anywhere within the region utilized by AIB. This also provides for minimal shoreline usage by the AIB interface along the side of a chip or chiplet. The resulting total shoreline will, of course, depend on the number of signals per channel and the number of channels in a column.

Intel® Agilex™ and Intel Stratix® 10 FPGAs and SoCs with intel EMIB

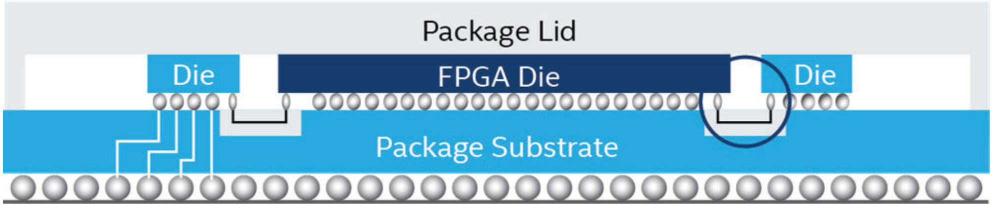


Figure 15. AIB connections can be made by wires on an interposer or, as shown here, by using a bridge technology like Intel's EMIB bridge.

During normal AIB operation, there is no difference between the two sides of the interface. But during bring-up, one of the two sides has to control and track the initialization and calibration sequences. During that bring-up phase only, one side of the interface acts as a master, and the other acts as a slave. Whether an interface is a master or slave will be documented on the chip or chiplet datasheet. Masters must connect to slaves, and slaves must connect to masters.

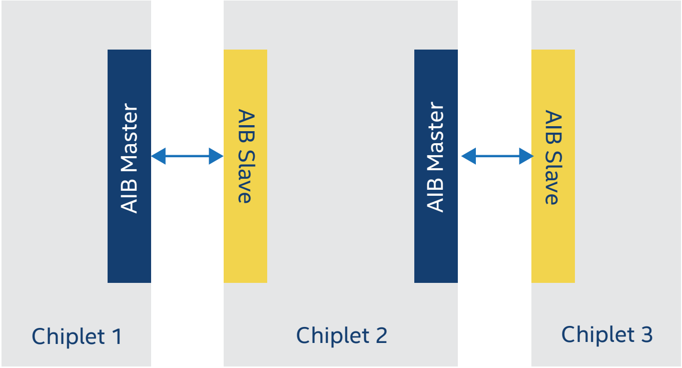


Figure 16. AIB masters must connect to AIB slaves; AIB Slaves must connect to AIB masters.

There are instances where you might want flexibility so that a given side of an interface can be a master or a slave. Such an interface is called a dual-mode interface, and it can be configured to act as a slave if connected to a master or as a master if connected to a slave.

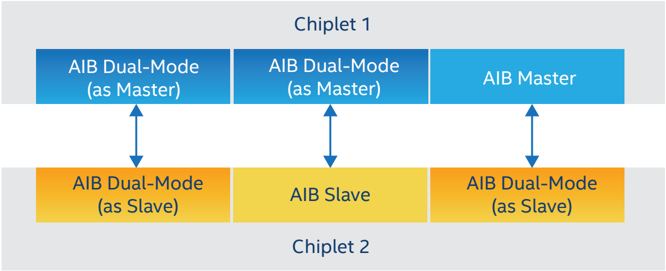


Figure 17. AIB Dual-Mode interfaces can act as master or slave, but must still be configured such that masters connect to slaves and vice versa.

While all channels in a column are identical, they are numbered for convenience. With respect to AIB, there is no meaning attached to those numbers, and any channel can be used for anything. There may be instances, however, where channels are bonded together at a higher level in the OSI stack such that there is an order and significance to different channels for a specific application. Depending on how a chiplet is oriented, the channels may come together with order reversed. In other words, if there are 12 channels, then, in some cases, channel 0 on one side will connect to channel 0 on the other side, while, in other cases, channel 0 may connect to channel 11. In most cases, channel 0 is identical to channel 11, and this reversal won't matter. If it does matter, then channel reversal may be needed within the MAC for those channels where order is important.

Redundancy

An AIB connection may involve up to 3840 I/Os, implemented as traces on an interposer. Interposer manufacturing yields may be high with respect to how many lines have defects, but, with so many wires, there is a risk that a single faulty wire can ruin an entire assembly. 99.9% wire yield may still be 0% module yield.

To improve module yields, AIB allows for redundancy of two types. I/Os participate in active redundancy. If a connection is found to be faulty, then all signals towards the center of the interface will shift to neighboring microbumps, taking advantage of two spare signals in the middle. One such wire fault can be corrected per channel – enough to make yields economical.

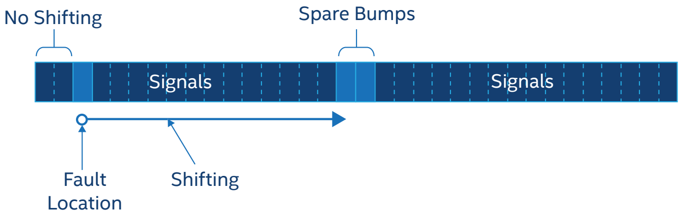


Figure 18. If a wire connecting an AIB signal microbump is faulty, then I/Os can shift over to use neighboring wires, making use of spare bumps to ensure good manufacturing yields.

This redundancy can be activated at test, where the shift position is saved in the module so that, on power up in a system, the correct connections will come up in their shifted positions.

There are two signals used during the earliest phase of power-up – before active redundancy can be used. For those signals, passive redundancy is used. This amounts to two bumps for each signal, such that, if one connection fails, the other can remain intact.

AIB Metrics

METRIC	AIB GEN1 (INTEL® STRATIX® 10 DEVICES)
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 1,024
I/O voltage	0.9-0.7 V
Energy/bit	0.85 picojoule

Table 2. AIB Metrics

The performance figures shown are optimized on Intel technology. Performance varies based on system configuration.

AIB Latency Comparison with SERDES

Compared to a typical serializer/deserializer (SERDES), AIB has much lower latency. A JESD204C implementation requires transport-layer mapping, 64B/66B encoding/decoding, and serialization/deserialization resulting in a longer digital delay. The SERDES analog delay is longer due to converting single-ended to differential and back, and for clock-data recovery.

	JESD204C	AIB	
MAXIMUM LINE RATE	32	2	Gbps
Total Digital Delay TX	17.53	0.75	ns
Analog Delay TX	2	1	ns
PCB/Interposer Delay	0.06	0.06	ns
Analog Delay RX	2	1	ns
Total Digital Delay RX	20.62	0.75	ns
Total Delay	**42.21**	**3.56**	**ns**

Table 3. JESD204C SERDES vs. AIB Latency Comparison. Source: Internal Intel analysis.

AIB Future Directions

Very high bandwidth applications such as direct RF sampling analog-to-digital converters (ADCs)/ digital-to-analog converters (DACs) will continue to push higher bandwidth, lower area, and lower power requirements on AIB. Doubling the data rate to 4 Gbps is feasible with careful I/O design. As high-density packaging technology improves, the industry standard microbump pitch will decrease, possibly to 35-micron pitch vs. today's 55 microns. To achieve energy per bit below 0.5 picojoule, the I/O voltage swing may decrease to 0.4 V. Compatibility of future AIB generations with first-generation AIB devices is very important and will be a strong factor in future AIB improvements.

Summary

The AIB interface is a new short-range, high-speed connection between chips and chiplets.

- It provides data rates as high as 2 Gbps (or even higher)
- It can support over 3,000 transmit and receive signals
- It utilizes a compact die layout to minimize the chip-edge shoreline, and it uses microbump technology to minimize bump area
- A carefully-designed signal layout and timing circuits help maintain tight skew for high-speed signals
- Redundancy helps ensure high manufacturing yields.

Implementations of AIB are in production today, and the specification provides a way forward for future tighter microbump technologies. You can download the AIB specification, along with other supporting documentations, at the AIB Open Source location at http://github/intel/aib-phy-hardware.

This paper contains the general insights and opinions of Intel Corporation ("Intel"). The information in this paper 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.