
# The Open Domain-Specific Architecture:

## A Chiplet-Based Open Architecture

- DRAFT - v0.9

# 1 Introduction

It is now commonly acknowledged that device scaling at the predicted rate of Moore's Law and correspondingly, the era of application power-performance scaling entirely through improvements in general-purpose CPUs will also end [7, 38]. The end of Moore's Law will increase the need for, and the use of, domain-specific accelerators (DSAs) to meet power and performance requirements in cloud infrastructure, network infrastructure and IoT/wireless edge applications [7]. As an important data point, recent HotChips conferences showcased domain-specific SoCs for many infrastructure applications including several neural network accelerators [11, 17], cloud processing [30], security processing and switching fabrics.

## 1.1 Heterogeneous Integration - Chiplets

DSAs have typically been developed and implemented as monolithic ICs. In a monolithic ASIC, all the components in the accelerator are designed and fabricated on one die in one technology. As process geometries shrink, the cost of developing ASICs has become prohibitively high, for example more than $250M at the 7nm process node [14]. With today's processes, only very large markets can justify the development of custom ASICs.

DSAs typically serve smaller markets than do general-purpose CPUs. ASIC designers try to reduce the cost of design by incorporating a large amount of 3rd party intellectual property (IP) to reduce the cost of design. A second approach to contain costs is to choose a more economical process node, say 16nm instead of 7nm or even 22nm. At older nodes, including all the functionality required by the application may make a die too large to manufacture economically.

Heterogeneous integrated systems offer a new design option. In these systems, different components in the product are designed and implemented on separate die, referred to as chiplets [18]. Components may be built with different process nodes and indeed may be supplied by different commercial vendors. Third party chiplets can reduce both design time and cost. The viability of this approach has often been limited by the performance and availability of inter-die interconnect. Until recently, the power-performance of inter-die interconnect was 3-4 orders of magnitude lower than intra-die interconnect. This required resources that mandated high-bandwidth access, such as external memory interfaces and host interfaces from being moved off-chip. Several new technologies ranging from simple through highly parallel [ to high-speed serial interfaces have been developed to improve the power-performance of inter-die connectivity [28, 29]. New packaging techniques have been developed to support multi-chip packaging with these various interfaces [1].

## 1.2 Open Chiplet Architecture

Chiplet technology developments have attracted the attention of large commercial companies and government research agencies. Intel [31], AMD [19] and Xilinx [62] address a complete stack - connectivity, logical data transfer and application execution on a multi-chiplet system. Their efforts largely use proprietary protocols and are closed systems with an entire heterogeneous system controlled by a single vendor. Cloud and network operators' power, performance and cost requirements will vary across where the accelerator is deployed in the network. Operators would also prefer to assemble custom accelerators by combining best-in-class solutions across multiple vendors.

Standardization efforts have largely been restricted to the PHY layer protocol for inter-die communication. The best known standard is the high performance 3D-stacked memory based on an open High-Bandwidth Memory (HBM) interface [29]. DARPA's program [18] focuses on creating and standardizing an open connectivity protocol between chiplets. One limitation is that the program has a focus on supporting process nodes important to the defense industry, but which may not be relevant to commercial development. This limits the protocol to technologies with limited analog performance demands on the interface.

Two other attributes of DSAs must be addressed in a multi-chiplet architecture. The first is memory management. DSAs are typically attached to a host processor and complete application flow is a combination of data processing on the DSA itself and the host. Coordination is achieved by coordinating memory state between the host and the DSA. This is typically achieved either with a memory coherence protocol or by programmer-managed data transfer between the host and the DSA. The second is control and management of the accelerator from the operating system on the host.

This document is a call to action to develop standards for the open architecture and to develop prototype products that can serve as a template and launch pad for the open architecture. The Open Domain-Specific Architecture (ODSA) Workgroup proposes a low-cost, high-performance open-accelerator architecture to address the full stack of requirements to develop a DSA, with the following components:

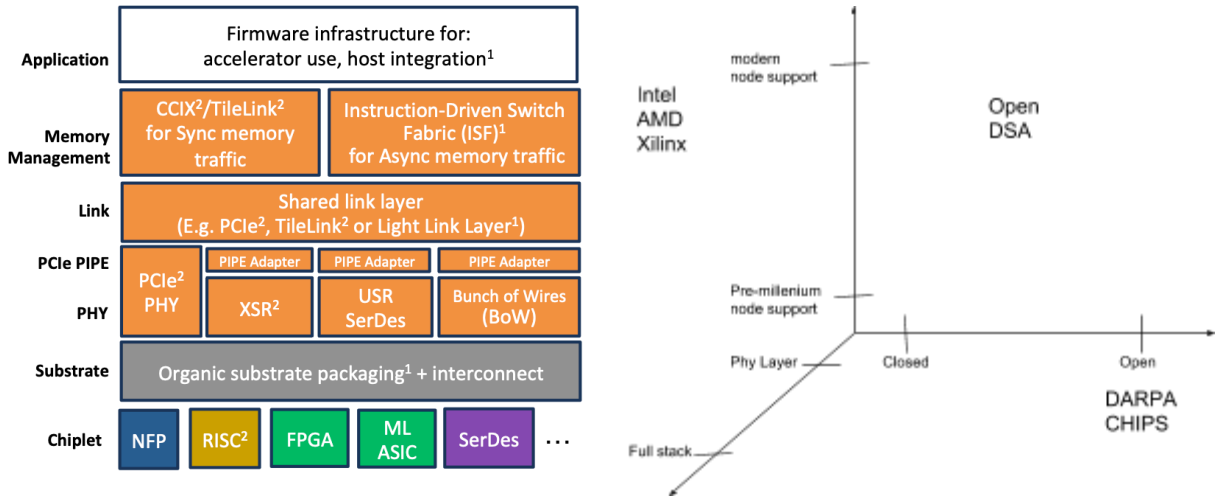
- Support for multiple forms of physical communication between chiplets
- Message-based protocols for coherent and bulk data movement between chiplets
- Host-integration software to integrate the accelerator with the host
- An industry consortium to enable a supply of chiplets for this approach

The ODSA Workgroup also proposes to make a platform prototype available with the following components:

- A low cost multi-chiplet package on an organic substrate
- A networking chiplet to implement message-based communication to which all other components are connected

- A long-range SerDes chiplet
- A RISC CPU chiplet
- A multi-chip package
- Host integration software for networking acceleration

Figure 1 contrasts the ODSA Workgroup with recent chiplet efforts from industry and government. Specifically, relative to the DARPA initiative, the ODSA focuses on the protocols for memory management above the stack. Relative to products from large companies, the ODSA promotes an open architecture where products from multiple vendors can interoperate. Taken together, these attributes will lower the cost and time needed to develop and deploy power-efficient, high-performance accelerators in a wide range of applications.




[1] New Open IP/Specification
[2] Existing Open Standard

Figure 1: The ODSA protocol stack and focus area

The open architecture will enable vendors to develop best-in-class, DSAs:

- Developers can combine heterogeneous best-of-breed components when developing product. Developers can combine die from multiple vendors. Also, for lower-performance parts, developers can even reuse die currently sold as packaged ICs.
- The architecture will provide high-performance, multi-terabit interconnect at low per-unit cost, by adapting to multiple low power methods of multi chip interconnect on an inexpensive substrate.
- The architecture will lower development costs by enabling a large degree of hardware and software reuse.

Current business models need to evolve to support the manufacture and distribution of die as chiplets. This is discussed in greater detail in Section 8.

## 1.3 Document Outline

The rest of this document is organized as follows. Section 2 reviews technologies relevant to the ODSA Architecture. Section 3 discusses technology proof points related to realizing cost-efficient implementations for the ODSA. Recent work in DSAs is reviewed in Section 4. Sections 5 discusses the various components in the ODSA. In Section 6, a proposal for a prototype implementation of the ODSA is developed. Section 7 discusses potential applications and deployment models for the ODSA. Section 8 discusses new business models.

# 2 Technologies Review

We review, the motivation for chiplet technology in greater detail. Recent work in two areas critical to successfully implementing chiplets, inter-die communication protocols and multi-die packaging technology are surveyed.

## 2.1 Motivation for Chiplets

Traditionally accelerators have been implemented as monolithic ASICs which include all of the functionality on one die which are typically connected to other chips in the system with medium or long reach SerDes interfaces. This allows for the most power and area efficient communication between sub-blocks.

### 2.1.1 ASIC Implementation Trends

Traditionally, IC designers have had two options when developing their next generation chips. The main approach to developing a new chip has been to incorporate increased bandwidth, increased processing capability (frequency, processing cores) and other feature updates in the next available process node. The second approach is to build additional features into the same node in order to reduce the investment in masks and tooling for a smaller geometry.

With the benefit of Moore's Law a designer could often combine two ASICs in their system into a single monolithic design in the next process node, with a potential increase in frequency as well. As multiple parts are combined into a single device interface power is removed, an additional benefit beyond the active power improvement of moving to a smaller geometry. Unfortunately, as advanced technology has moved to finer and finer features to enable area and power scaling, the costs of implementing these devices has skyrocketed. Figure 2 shows the rapid increase in development costs as designs migrate to advanced process nodes. For many accelerator devices with limited market and application space, this additional cost is just not feasible.
To justify development costs, ASIC designs being larger than they actually need to be as they are designed with the superset of needed function for a variety of applications. Unfortunately,

the superset of functions reduces the benefit of the technology shrink and results in more complex chips that require more effort to design, layout and especially develop software for, matching or exceeding the increased mask and process cost for newer, smaller technology. Even as these chips are becoming more and more complex to reduce investment costs, cost per transistor improvements are slowing due to more complex lithography and process (double, triple patterning, EUV, etc).

With these trends, it seems an obvious choice would be to build very large monolithic die in older technology nodes. Updating devices in the same process node also poses a challenge. Often the process geometry and yield benefit of smaller die shown in the previous section make it unaffordable or impossible to combine two different designs into a single chip and meet cost or reticle limits. Although eliminating an interface could provide a real improvement in interface power, the overall increased cost of the device makes this benefit difficult to achieve.

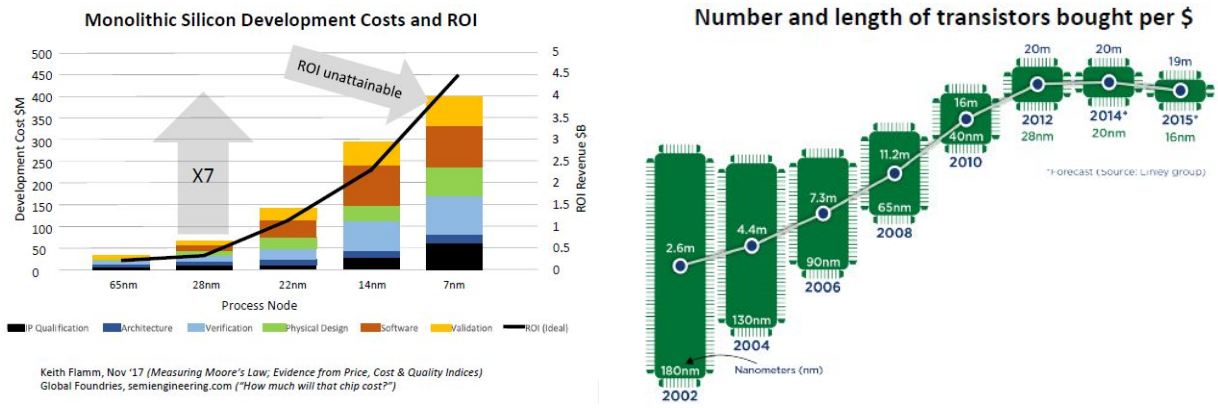



Figure 2: ASIC development costs and transistor cost efficiency

There are also cost implications driven by defectivity of large die, technical limitations due to the limit of the reticle used in lithography tools and limits of reliable large die attach to laminates. Figure 3 compares the yield of two die, one 10x10 and the second 20x20. Even with very good d0 (0.1) four 10x10 die yield 29% more good die per 300 mm wafer than 20x20 die. (source: https://caly-technologies.com/die-yield-calculator/)

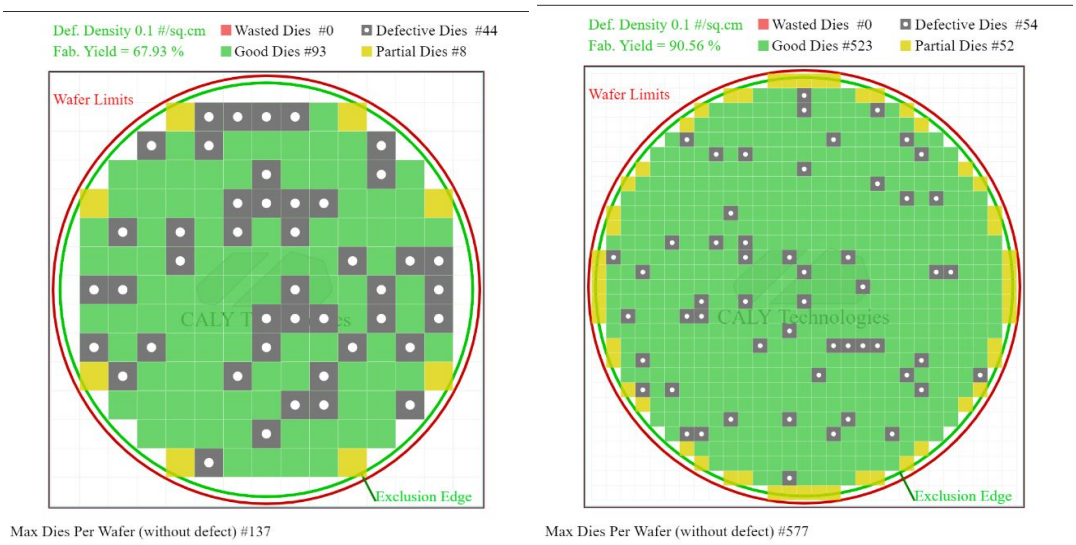



Figure 3: Die size impact on yield

An FPGA-based design is another implementation option for accelerators that has very low development costs. Many accelerator developers, with applications that serve a limited market, are simply unable to justify the expense even though they can provide a significant cost and power advantage versus using processors and FPGAs to implement the needed function. Figure 4 shows the significant area and power advantages possible with ASIC implementations relative to FPGAs.

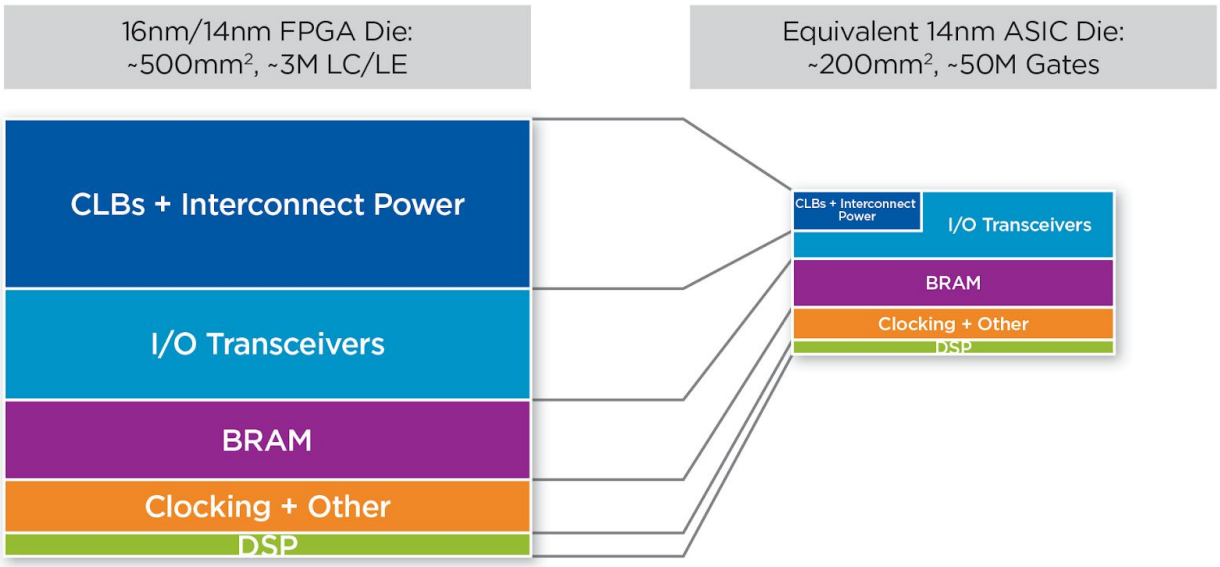



Figure 4: Comparative study of same content in ASIC vs. FPGA showing significant area advantage

### 2.1.2 Chiplets Overview

In the open architecture discussed in this document, we explore a third path, heterogeneous integration of multiple component die using lower power interfaces such as USR, Bunch of Wires (BoW) or emerging 112G SiP standards. By layering a common protocol over different interfaces a common ‘building block’ based approach can be leveraged to create systems on a substrate by simply varying the Bill of Materials (BoM) for an Multi-Chip Module (MCM).

Some of the component blocks (such as a Long Reach SerDes tile or Electrical to Optical Interface) may move to more advanced nodes where needed, however others are likely to remain in cost-effective nodes to reduce overall investment. As shown in the Figure 5 below, while not attaining the same area and power advantage of a technology shrink, this third path provides a considerable area and power savings over monolithic integration in more cost effective nodes by reducing interface area and power significantly. While multi-chip systems typically have a higher cost than individual die, the incremental investment can be somewhat offset by these area and power savings.

In this diagram, the base investment is shown as a reference point for an original design that needs to be updated by combining components and adding new features by either integrating or pushing a larger design to a newer technology. Integration onto a single substrate provides many of the benefits of a process shrink at significantly lower investment costs.

The integrated system on a substrate also provides a major savings in board real estate and routing laters, significant system costs that can often dwarf any increased cost to design and integrate the MCM.

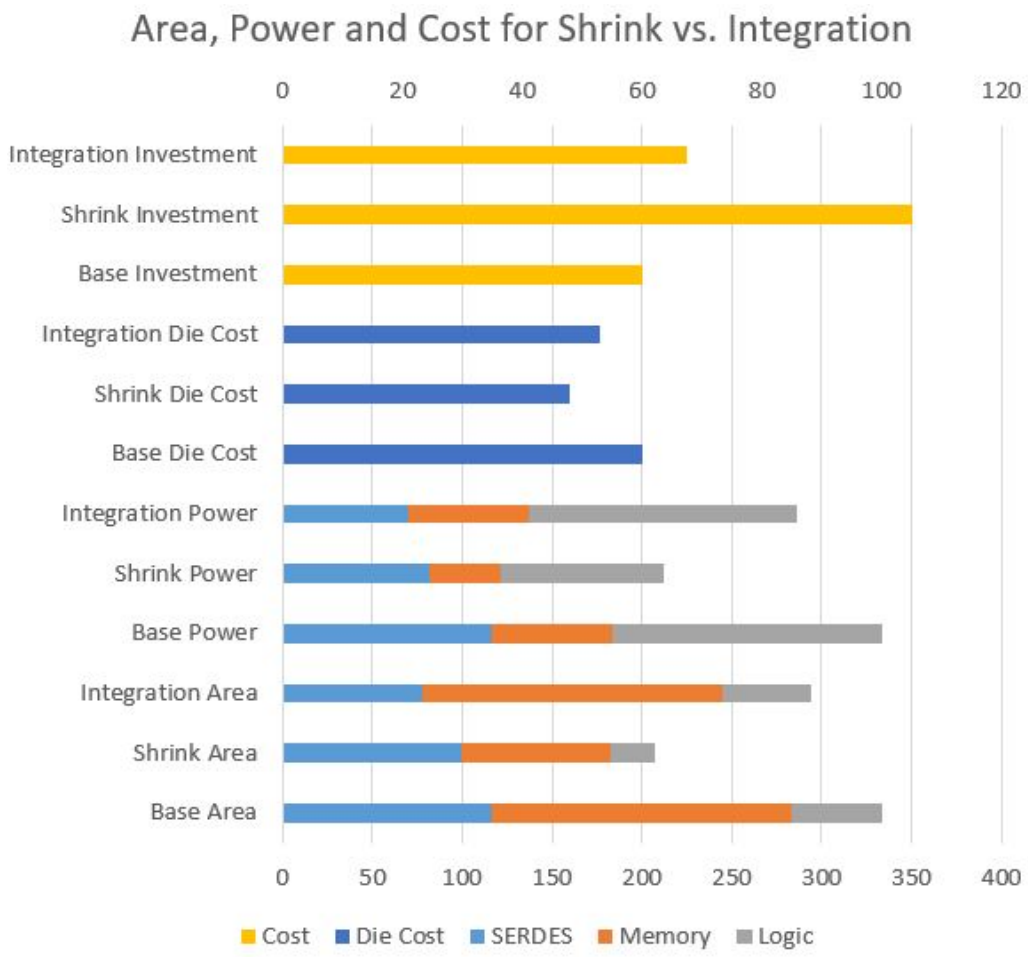



Figure 5: Comparing(multi-chip) integration with a process node shrink

## 2.2 Die-to-Die Interconnect and External Interfaces

One of the primary challenges associated with development of physically disaggregated (heterogeneous) MCM solutions is moving data between die while still maintaining competitive cost and manufacturability. While enormous strides continue to be made in high-density, low-cost packaging solutions, there are still significant advantages in choosing interconnect technologies that are compatible with packaging solutions that are available for high-volume manufacturing today. Additionally, the use of proven packaging solutions improves compatibility with external interconnects such as PCI Express and Ethernet interfaces that are likely to be required for these SiP solutions.

The search for the “one true interconnect” for die-to-die communications within a single MCM has been ongoing for many years. The challenge lies in the desire to optimize six often-competing but interrelated factors:

1. Cost of packaging solutions

2. Die area per unit bandwidth (square mm per Gigabits per second)
3. Power per bit
4. Scalability of bandwidth
5. Complexity of integration and use at the system level
6. Realizable in any semiconductor process node

The ideal solution is an interconnect technology that is infinitely-scalable (at fine-grained resolution), low power, area-efficient, totally transparent to the programming model, and buildable in a low-cost silicon and packaging technology. Generally speaking, there are three classes of technologies that service this space:

- Traditional Medium and Long-Reach SerDes
- Shorter reach SerDes of different ranges - XSR or SiP SerDes, USR Femto SerDes
- Parallel interfaces - High-Bandwidth Memory (HBM), Advanced Interface Bus (AIB), “Bunch of Wires” (BoW) interfaces

We briefly review each of these die-to-die communication technologies.

### 2.2.1 Traditional SerDes

Traditional Medium and Long-reach (MR and LR) SerDes (e.g. PCI Express, Ethernet, etc) present a few key advantages as die-to-die interconnect solutions. These SerDes interfaces tend to be available in a broad selection of silicon process nodes at a reasonable variety of speed/power optimization points. Most standards incorporate the concept of bandwidth scaling by design in order to support legacy modes of operation. The integration on the die and into the system programming model tends to be very robust and well-understood as these interfaces are widely used. The required packaging technology to integrate and use these interfaces is broadly available and inexpensive.

However, they also present a number of significant drawbacks. As these implementations tend to be generalized and focused on communication within a physically large system (e.g. a server or enterprise router), they are not power- or area-efficient relative to a dedicated die-to-die solution. The programing model, while well-understood, is intended to communicate between devices that are physically and logically distinct and therefore tends to incorporate robust flow control and significant system overhead that adds latency and complexity to interfaces that are ideally transparent in a SiP.

### 2.2.2 XSR or SiP SerDes

XSR/SiP is a relatively new class of SerDes interface that began to be introduced at the 50Gb/s speed node[1] and is receiving additional focus and attention as the industry looks for future

[1] Optical Internetworking Forum, *Implementation Agreement OIF-CEI-04.0,* http://www.oiforum.com/wp-content/uploads/OIF-CEI-04.0.pdf, 2017

solutions at 100Gb/s. XSR/SiP SerDes represents a heavily-optimized and typically very high-speed serial interface that is purpose-built for die-to-die communication. Based on a traditional SerDes architecture (incorporating a clock-data recovery circuit) but with a severely constrained insertion loss budget, these XSR/SiP links are power- and area-efficient, allowing for extremely high-bandwidth links within a SiP. As these tend to be treated as simple “bit pipes,” the system programming model can be very flexible, with the constraint that significant latency can be introduced due to the necessity for Forward Error Correction (FEC) to be applied to the interface at the 100Gb/s speeds needed to optimize power and area efficiency.

As with other solutions, there are some negative aspects to XSR/SiP SerDes-based interconnects. To support good signal integrity at very high speeds, higher-performance (and therefore more expensive) package substrate materials are required to support large-scale integrations. Total interface bandwidth between die can be scaled up very effectively, but the circuit overhead required to run at 50 or 100Gb/s requires that the minimum interface bandwidth and unit of adding bandwidth be several hundred Gb/s in order to realize the power and area efficiency targets. And finally, advanced silicon process nodes are typically required to support the design of these SerDes, limiting the flexibility for system designers to put “the right die in the right process.”

### 2.2.3 USR Femto SerDes

Further optimized for specific die-to-die communications, USR Femto SerDes employs enhanced signaling schemes (clock-forwarding, advanced encoding, multi-bit/multi-wire transmission, etc) to provide extremely power-efficient solutions. Capable of high data rates per wire, these interfaces can provide a good balance of bandwidth and cost to implement by using existing packaging technology. While not yet capable of hitting the same absolute interface bandwidths that a 100Gb/s XSR/SiP solution can, USR designs can provide appreciably better power efficiency. Like XSR/SiP SerDes, the system integration model can be flexible and very lightweight, but FEC may need to be applied at higher data rates to support acceptable data transmission integrity. Additionally, USR serial interfaces are typically customized “hardened” macros that often involve proprietary encoding schemes, meaning that a custom design effort into a specific process technology is likely to be required to implement them. Interoperability and compatibility with legacy technology can also present challenges.

### 2.2.4 “BoW”

The simplest solution to implement for a die-to-die interface is a wide, clock-forwarded parallel bus similar to that used for DDR-style memory interfaces. Flexible, scalable, and simple to implement and use from a system and software perspective, these designs can be implemented in nearly any silicon process, achieving extremely low power in more advanced nodes that support lower voltages. In general, BoW solutions will represent the lowest power, most dense solution but with one significant drawback: packaging costs increase significantly once the interface exceeds a certain bandwidth. Due to limitations on current organic packaging substrate technology, a move to either a silicon-based interconnect medium or a high-density

organic solution is required once the bandwidth between die exceeds roughly 400Gb/s per millimeter.

### 2.2.5 Parallel Interface AIB/HBM

High Bandwidth Memory (HBM) has been a major player in driving multi-chip integration in the industry using a relatively low speed parallel interface on Silicon interposer, using a very wide interface with a fine wiring pitch. AIB is an interface being driven by the DARPA CHIPS initiative along with industry partners which largely matches HBM in terms of data-rates and the use of fine pitch wiring on silicon bridge or interposer technologies. Both of these technologies achieve relatively high bandwidth densities, but also require relatively complex silicon based interconnect technologies.

### 2.2.6 Comparing Protocols

Each of the solutions discussed above tends to optimize some or most of these factors, but in many cases the best solution is highly dependent on the application. Parallel interfaces such as BoW, AIB, HBM offer low power, low latency and high bandwidth, but at the cost of requiring many wires between die. The wiring requirement can only be met using expensive interposer or bridging technology. Relative to parallel interfaces, SerDes offer similar bandwidth, but incur a little additional power and transaction latency. SerDes have been used to provide high-bandwidth off-die communication with a limited number of physical wires in a variety of standards. However, most SerDes such as those used for Ethernet communication or PCI Express though area efficient, consume too much energy. USR SerDes offer off-die communication at a Figure of Merit closer to that of on-chip interconnect. However, SerDes based communication typically incurs a greater latency relative to on-chip networks due to serialization overhead.

System designers should consider all of the relevant requirements for their application prior to choosing a die-to-die interconnect. The graph below in Figure 6 summarizes the relative advantages and disadvantages of each interface along various parameters of interest, such as the Figure of Merit - The ratio of bandwidth density to power and bandwidth density across both silicon and laminate substrates.

The ODSA discussed in this paper abstracts the PHY protocol by using a common transaction protocol for data transfers. The ODSA transaction layer allows system designers to choose the best interface for their function without mandating a specific solution. As shown in Figure 7, multi-chip systems leverage multiple interface technologies based on budget constraints, availability, bandwidth and power requirements.

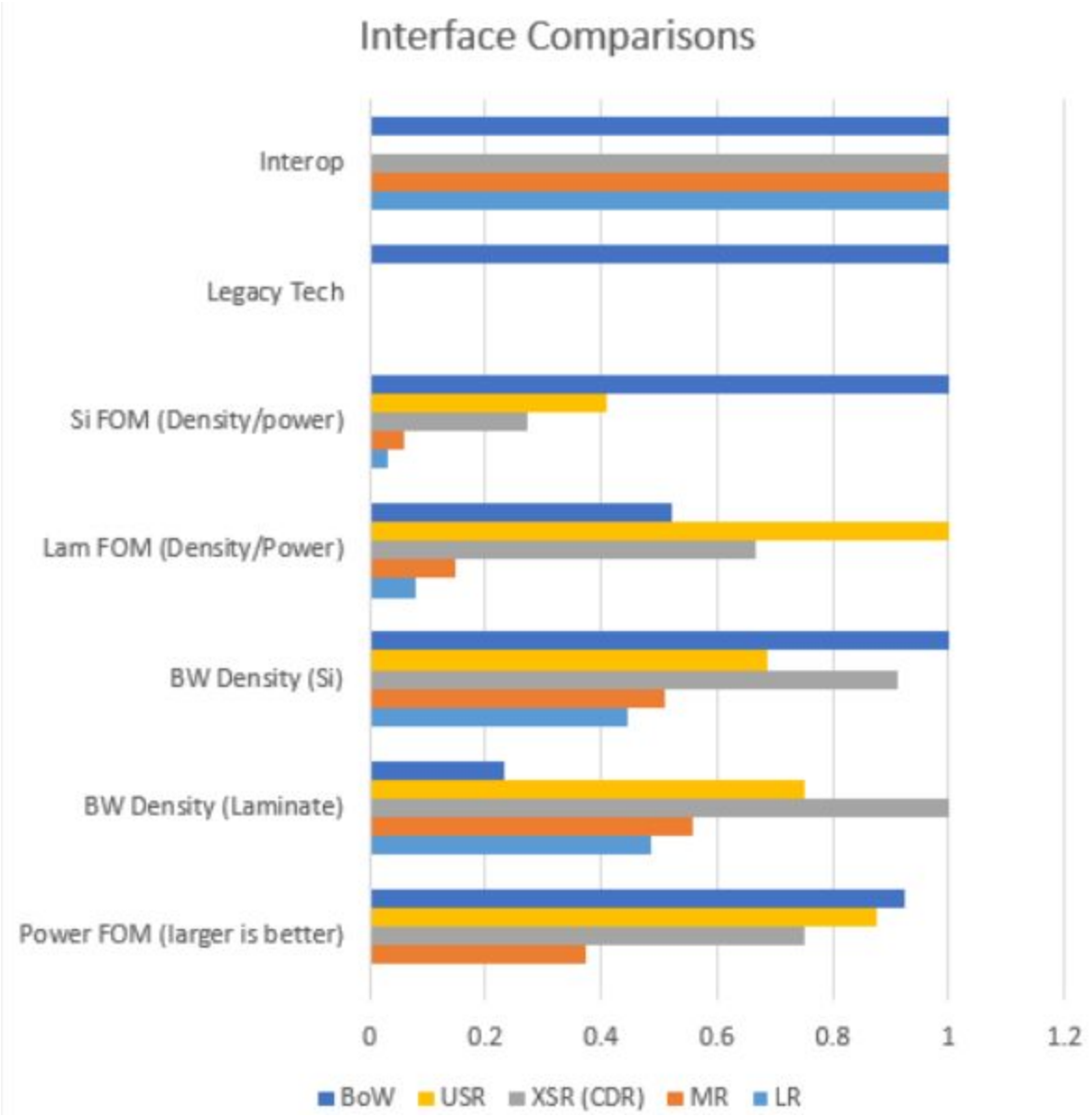



Figure 6: Comparing die-to-die protocols

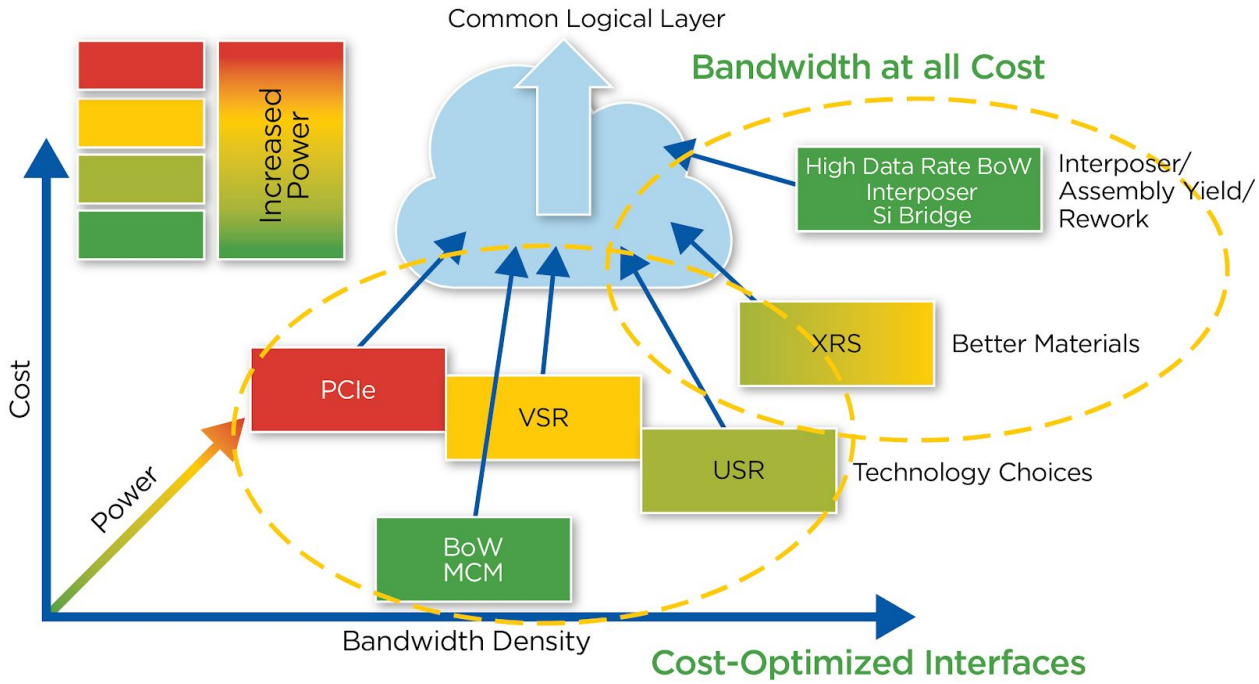



Figure 7: Comparing new inter-die interconnection technologies

## 2.3 Substrate and Packaging

Driven by product and market requirements to enable tighter integration, packaging technologies have gone through revolutionary transitions over the last few years. The requirements on the package has grown from purely enabling the electrical and mechanical connection to the outside world, to now supporting the multitude of interface technologies between different chips on an multi-chiplet package as well.

### 2.3.1 Multi-chiplet Packaging Requirements

The integration of multiple chips and/or packages into a single MCM has driven to significantly larger package body sizes while requiring much smaller signal lines and spaces. The definition of the best chip-to-chip interface for a particular purpose cannot be done without considering the corresponding packaging technology. The chip-to-chip interface directs and influences the choice of packaging technology, specifically the substrate needed to integrate multiple chiplets. Packaging is turning into one of the key areas to enable multi-chiplet integration. Three related issues drive the selection of the right package technology for a multi-chiplet design:

- The chip-to-chip interface
- Cost constraints and performance requirements
- The total size of the multi-chiplet package

For example, high wire density in inter-die interconnect may mandate the use of a substrate or bridging technology that supports high wire density. The enablement of the High Bandwidth

Memory (HBM) is likely the best demonstrator of this trend – as the HBM can only be integrated with the ASIC in the same package, and at this point only in a 2.5D Silicon Interposer configuration. The HBM chip's external interconnection is DRAM like which requires a large amount of I/Os. As a result, its package uses micro-bump (u-bump) with a pitch of 55um and a diameter of 25um. In HBM2 the total number of u-bumps is 4,942. Since the ASIC chip has to use the same amount of I/Os when communicating with the HBM chip, the ASIC chip has to use u-bump with similar u-bump pitch and diameter. Because today the minimum allowable bump pitch for C4 and for Cu-pillar is about 130um and 90um respectively in an organic substrate, silicon interposer becomes the only viable solution.

While silicon-based packaging technologies have evolved into volume manufacturing solutions, cost and complexity may prevent them from being the right solution for most lower-end applications. Standard FC-BGA packaging is a lower cost option, and supports several interconnect technologies (SerDes and BoW), but will not be able to achieve the same bandwidth as a silicon-based solution.

This difference is significant because silicon interposers bring high cost, high manufacturing complexity and yield loss. The NRE cost of designing and prototyping a 2.5D silicon interposer package can easily go to US$1M or more, and the production assembly cost per chip can approach US$100. As a comparison, the NRE cost of designing and prototyping a MCM package is only a fraction of the US$1M, and the production assembly cost per chip is much lower than US$100. Economically, there is a large benefit to get rid of the silicon interposer and to go back to the traditional MCM package.

Using an organic substrate is much like just using a traditional PCB. Both PCBs and organic substrates are fabricated through the use of traditional etching processes that do not rely on the use of semiconductor fabrication equipment, which silicon interposers require in order to deliver their fine pitch.

Package size, determined largely by the total die sizes of the components is a second issue to consider. Ball Grid Array (BGA) packages can reach 75mm x 75mm in size reliably before warpage and thermal expansion issues reach their limits. BGA packages can either be soldered down or mounted in sockets. Land Grid Array (LGA) sockets that can reach 110mm x 110mm in size. LGA sockets include a tiny leaf spring that allows some warpage and expansion to occur. TE Connectivity has a LGA product line called the XLA sockets that delivers just such capability along with still good SerDes signal integrity, easily handling 56G SerDes and perhaps handling 112G SerDes. 110mm x 110mm is the size of a large piece of toast and can hold a great deal of functionality.

Standard Silicon Interposers have traditionally been limited to the reticle size. For most Silicon fabrication equipment used, this reticle limit is in the range of 32mm x 26mm. More advanced solutions include the stitching of multiple reticle fields to form larger interposers, or the use small interposers ("Silicon Bridges") only in the areas where they are needed.

### 2.3.2 Packaging for Parallel Interface Integration

Parallel interfaces such the AIB or HBM, or the proposed more generalized BoW interface approach significantly increases the demands on the package technologies that can be used. BoW generally will feature slower signal speed than serial solutions, but significantly more interconnects between chips. Depending on the amount of bandwidth that needs to be supported between chips, different package technologies can be selected:

- For low-to-medium bandwidth requirements between chips, the same standard FC-BGA substrate technology can be used, with the only addition that smaller wires may be needed vs. the rather large wiring used for SerDes signals (20um Lines/Spaces).
- For high-bandwidth applications, the number of signal wires needs to get maximized, and therefore the lines and spaces need to shrink significantly further. The smallest geometries that can be printed will always be on Silicon, and therefore 2.5D (or alternate silicon-based technologies, such as EMIB) can provide very high bandwidth density. While enabling significant benefits, the use of Silicon as a package interconnect media also leads to a complex and expensive package solution.
- Several new technologies are currently being developed to target the “space in-between” - a solution that is lower cost, but still enables a very high density of interconnects. Those solutions include organic laminates with additional fine-pitch layers processed on top of the conventional layers (“2.1D”), and a number of novel Wafer-Level Fan-Out technologies that are being targeted at achieving similar wiring densities as the HBM requires.

### 2.3.3 Packaging for SerDes Integration

The development of the USR SerDes interconnection technology significantly reduces the total number of I/Os required to communicate between semiconductor chips. It allows the organic substrate to provide the interconnection between dice and enables the mature MCM technology to serve us again.

Traditional SerDes, as well as the evolving XSR and USR Fempto SerDes all share the advantage that there is a lower number of signal lines (running at higher speeds). This enables a fairly standard package solution, such as FC-BGA. The main new elements for FC-BGA package technology to support MCM integration now allow:

- Enabling larger package size:
    - up to the 32/28nm node, there was a fairly conservative belief that packages larger than ~55mm would lead to problems at card assembly. This view has significantly evolved in the last several years, with 70mm packages in production, and 80-100mm packages in many companies roadmaps.
- Supporting the electrical requirements for very high-speed signals (such as the 100G XSR):

- Lower loss dielectrics in the package substrate are needed to enable longer trace length while keeping the insertion loss at an acceptable level.

The high speed signals, such as those from USR or LR SerDes, can be tunneled through the silicon interposer. The typical technique is to employ several neighboring die micro-bumps to form an output that maintains the impedance and thus signal integrity of a traditional die bump for the SerDes. There are also more micro-bumps to form a tighter barrel of ground bumps surrounding the high-speed transmission line.

However, since using SerDes reduces the wirecount requirement, it may be possible to use cheaper glass or even organic substrates to build large multi-chiplet packages. Using these substrates lowers total package development costs.

## 2.4 Inter-Chiplet Data Transfer Protocols

In most acceleration applications, in a multi-chiplet product the datapath is executed on more than one chiplet. The multiple chiplets share data through data transfer protocols. Coherent protocols use hardware support to provide a software developer with a consistent memory state at both the host and the DSA. The cost of providing this consistency is proportional to the physical area over which it needs to be achieved. On a very large area, the latency cost of achieving consistency may be significant and a programmer has limited direct control over this latency. There are several open protocols for state consistency for accelerators, including CCIX, TileLink and OpenCAPI.

A non-coherent data fabric is an alternative to coherency protocols. In systems with non-coherent fabrics, the developer explicitly controls data transfers. TensorFlow is an example of an accelerator that uses non-coherent data transfers. There are two options to develop a non-coherent data fabric for chiplets. One option is to extend an on-chip fabric for off-chip transfers. Most on-chip fabrics use a synchronous global bus. These buses are not easily extended off-chip. The second option is to use chip-to-chip non-coherent data transfer protocols for chiplet-to-chiplet data transfer. PCI Express the most common example of an inter-chip non-coherent data transfer protocol. However, the overhead to use it within a package may be significant. Netronome has developed a light scalable fabric and protocol for non-coherent data transfer that can be extended to inter-chiplet data transfer. This will be discussed in detail in Section 3.

# 3 Technology Implementation Proof Points

This section reviews proof points for the advanced technologies to be used in the prototype. Specifically, we discuss implementations for the USR SerDes, substrates and non-coherent data transfer protocols. Section 6 will discuss a prototype implementation for the ODSA.

## 3.1 Ultra-Short Reach SerDes

In the past two years, a significant alternative to the use of silicon interposers or silicon substrates has arisen. That alternative is the combination of low-cost substrates and highly power-efficient USR SerDes. The emergence of these USR SerDes, particularly the Glasswing SerDes from Kandou that uses the CNRZ-5 modulation technique, has allowed complex systems to be constructed on large MCMs. The CNRZ-5 modulation technique delivers NRZ-shaped receive eyes and as a result enables minimal equalization to be used, even at high rates, thus lowering the power.

### 3.1.1 Modulation over Organic Substrates - CNRZ-5 vs PAM-4 vs NRZ

The selection of the modulation technique to be used on the SerDes that run over the organic packages is important. The three relevant choices are CNRZ-5, PAM-4 and NRZ.

PAM-4 is a poor choice of modulation technique for use on substrates because it has bad native error performance that has to be protected by significant and sometimes high latency Forward Error Control (FEC) blocks and/or large high-power equalizers. The reason for this bad native performance is that PAM-4 has the combination of large eyes followed by small eyes on the same link, a consequence of the PAM-4's three stacked eyes. The energy from the large eyes reflects off of any impairments and the receiver. This reflected energy tends to close the small eyes. Reflections are the key impairment in USR/XSR links. A similar vulnerability exists for non-reflected Inter-Symbol Interference (ISI).

PAM-4 is used by the coming generation of XSR SerDes that are being developed for optical module applications. In that application, a large system FEC is present to protect the optical link, so it comes for free on the electrical links that directly carry data to the optical links. In almost every other application, significant FEC blocks need to specifically accompany the link. This means that they typically need both enhanced equalization and FEC. The native error performance of the OIF CEI-56G-XSR-PAM4 Interoperability Agreement is 1E-9.

CNRZ-5 has excellent native error performance because it has NRZ-shaped eyes at the receiver decision point. It has none of the eye stacking that PAM-4 has at its decision point. This means that CNRZ-5 can handle large reflections in much the same way that NRZ can. The native error performance of the GW16-500 SerDes without FEC is 1E-15.

NRZ USR Phys are robust and can typically live without a FEC. At the same baud rate NRZ PHYs are less pin-efficient than CNRZ-5 PHYs. Pin efficiency matters in USR applications because of the constrained number of die balls available. The reduced pin efficiency directly lowers the beachfront bandwidth of a die using NRZ. That said, 25Gb/s NRZ USR PHYs are on the market and 50Gb/s NRZ USR PHYs are currently under development by several vendors.

### 3.1.2 Kandou's Chiplet USR Proof Point

The GW16-500 Quad Glasswing Phy is the first PHYs to utilize Chord™ signaling, an innovative new PHY technology. The Glasswing uses CNRZ-5 Chord signaling, a form of signaling that fits in the space between single-ended and differential signaling. Chord signaling can make almost every interface better by getting more bits through with lower power and fewer pins.

The Glasswing FemtoSerDes PHY carries 5 bits over six wires using the CNRZ-5 (Chord signaling-based Non-Return-to-Zero, 5 channel) modulation technique. It has excellent signal integrity (SI) properties because it uses a specific multi-wire code in combination with specific chord receiver that has been designed with SI in mind. This good SI allows the Glasswing to run at low signal swings with minimal equalization, saving power without the 2X wire penalty of differential signaling.

The GW16-500 Glasswing PHY delivers four sets of five 25Gb/s channels for a total of 500Gb/s over 24 data wires plus two clock wires in each direction. To the system, it looks like 20 25Gb/s SerDes that share a clock. The PHY also save power through the use of clock-data alignment (CDA) together with a forwarded clock. Figure 8 below shows the Glasswing bump map, which occupies just 2.4mm of chip beachfront.

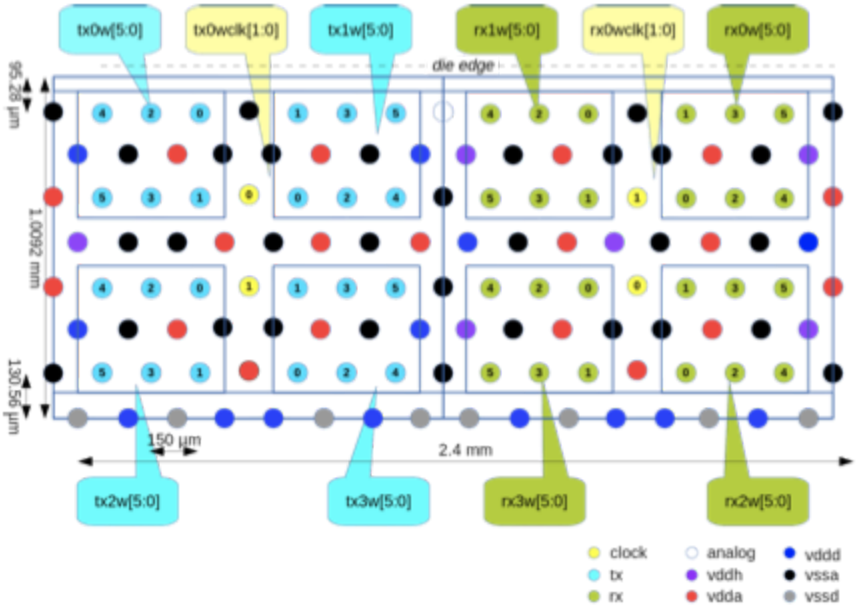



Figure 8: GW16-500-USR bump map

### 3.1.3 AQlink Chiplet USR Proof Point

AQlink™ is a USR PHY technology developed by Aquantia that uses differential NRZ signaling. The smallest AQlink™ building block is a transceiver that includes two differential pairs for transmit and receive interface and one differential pair for clock interface. AQlink™ takes advantage of clock forwarding to simplify data recovery circuitry in the receiver and therefore helps minimize the power and area in the receiver. In the clock forwarding scheme, the clock that is used to clock the transmitter is forwarded with the transmit data signal to the link partner transceiver. The link partner receives the forwarded clock and regenerates a new clock signal that is phase-aligned to the center of the data signal. To limit the number of clock signal bumps and traces over the package substrate, each differential clock signal can be used by multiple transceivers. An example of such topology is the AQlink-Quad1 module as shown in Figure 9. The AQlink-Quad1 module includes four differential data pairs and one differential clock. To improve bump and trace efficiency, several transceivers can share a single differential clock signal. However, for robust performance at data rates >50Gb/s, a clock signal is recommended to be shared with up to three transceivers (six differential pairs).

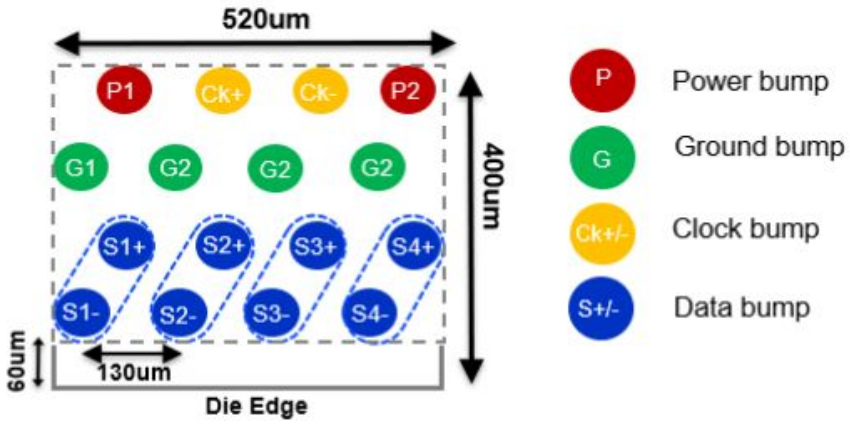



Figure 9:: AQlink-Quad1 bump map with four differential data and one differential clock pair

AQlink-Quad1 has been implemented for the first time in 14nm GF process node. It supports a throughput range of 20Gb/s-56Gb/s per port, or a total of 112Gb/s transmit and 112Gb/s receive, over 25mm trace on a typical organic package substrate (e.g. GZ41substrate material). Lower data rates down to 10Gb/s per port can be achieved by operating the PHY at half rate, where every bit is replicated twice and transmitted in consecutive bit times. AQlink-Quad1 has a 64-bit wide receiver interface and a 64-bit wide transmit interface on the parallel side that is clocked at maximum 1.75GHz clock rate. An optional 2x Adapter block can be added to the parallel interface to increase the receive and/or transmit width to 128 bits parallel clocked at maximum 875MHz.

AQlink-Quad1 offers very competitive power efficiency, which itself is a function of signaling baud rate, trace length, supply voltage and temperature. The transceiver consumes maximum

energy per bit at 56Gb/s per port over 25mm trace length at 110C. AQlink-Quad1 silicon, which has already been integrated in different IC products, has been fully characterized across process, voltage and temperature (PVT), delivering BER<1E-15 without a FEC, and sustaining ESD of 400V HBM and 100V CDM.

Several AQlink modules can be grouped together to build a higher throughput data interface on a die edge. For example, AQlink-Tera is implemented by grouping 10 of AQlink-Quad1 modules abutted together to deliver a throughput of up to 1.12Tb/s. Figure 10 shows AQlink-500G that is created by grouping 5 of AQlink-Quad1 modules to deliver up to 560Gb/s. AQlink-500G IP core occupies 1.04mm$^2$ with 0.4mm of height over 2.6mm of chip edge. On its parallel side, it has 320-bit wide receive interface and 320-bit wide transmit interface. The parallel interface can be increased to 640-bit wide receive and 640-bit wide transmit using a 32:64 Adapter per port.

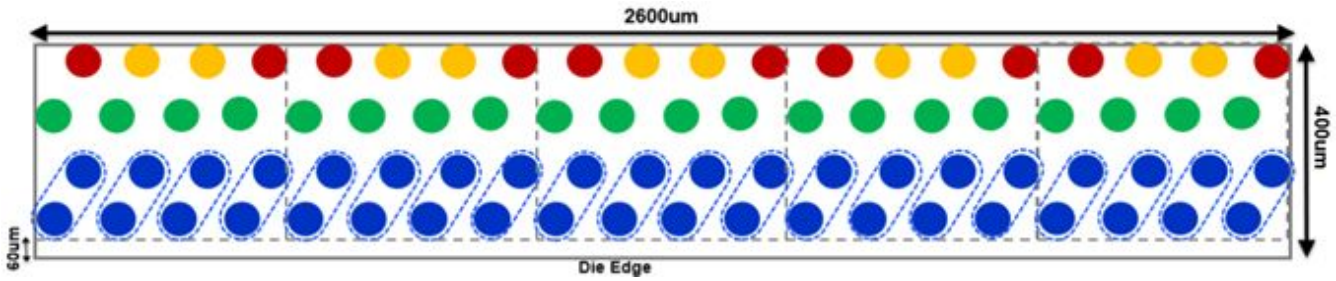



Figure 10: AQlink-500G bump map

### 3.1.4 PIPE PHY Interface Layer

As mentioned in Section 2, there are multiple options to choosing a PHY protocol for multi-chiplet systems. If different interfaces could present a common interface to higher data transport layers, system design can be simplified. The PIPE interface has been a key interface between PCIe controllers and PHYs. The initial draft of PIPE was defined by Intel in 2002. It has been updated many times since as PCIe evolved.

For example, when the CCIX protocol was defined, its architects reused portions of the PCIe architecture including the PIPE interface. Some or all of the CCIX controller implementations on the market use the PIPE interface as the link between the Link Layer and the PHY Layer. CCIX added an additional, optional PHY baud rate, but this modification was done fully within the context of PIPE. Sixteen lanes is the principal width of high-bandwidth implementations of PCIe and CCIX. An important CCIX controller on the market, only supports a sixteen lane PIPE interface.

> For example, a 16-lane PIPE interface can be carried over the Kandou Glasswing USR SerDes using a PIPE Adapter. This block connects the PIPE interface of the CCIX IP Controller to the Glasswing. At boot time, the five 32-bit sub-channels of each of the four Chords of a Glasswing are bonded together using the Autostart mechanism. This forms four 160-bit interfaces. The Autostart aligns the five sub-channels of each chord into a

single 160-bit link. Divide these further to form sixteen 40-bit slots as per the longer definition of the adapter.

The maximum capacity of each slot is 512/16 = 32Gb/s. The ten-bit data coming from the PIPE interface is put into the respective slot sequentially. For 32GT/s (PCIe Gen 5), run the Glasswing at 25.6 GBaud. For 25GT/s (CCIX ESM) run the Glasswing at 20 GBaud. For 16GT/s (PCIe Gen 4), run the Glasswing at 12.8 GBaud in half-rate mode.

Use the mode in section 4.2 (of version 5.1) of the PIPE interface on the CCIX or PCIe controller. The Glasswing and the PCIe/CCIX controller must be clocked from the same transmit clock. The Glasswing's and the adapter clock generator must be based on the same source. The jitter and wander of all components needs to be accounted for in the buffering.

PCIe and CCIX controllers both support retry buffers that have the capability of covering any errors that the Glasswing's $1E^{-15}$ error ratio supports. This error ratio is better than that specified for PCIe/CCIX.

## 3.2 Substrate and Packaging

We review recent results that demonstrate the potential for significant cost reductions in multi-chiplet packaging.

### 3.2.1 Organic Substrates

Typically, the high wire density required by parallel interfaces, such as High Bandwidth Memory, mandates the use of silicon interposer technology. As mentioned before, silicon interposers are significantly more expensive than organic interposers. Both commercial vendors and academic researchers have demonstrated significant density increases with organic interposers. A increase in bump density from 150um to 40-80um and increase in wire density from 5um spacing instead of a typical 30um spacing. That is, low-cost organic substrates may be able to achieve a density comparable to that achieved with high-cost silicon interposers.

### 3.2.2 Glass Core Technology

A high-performance, cost-effective alternative to silicon interposers and organic substrates is Glass Core Technology (GCT). GCT leverages the benefits of glass, over silicon, to allow smaller diameter vias and a smaller pitch from via to via. GCT uses Through-Glass Vias (TGV) to connect to a Redistribution Layer (RDL) to create a desired circuit on the glass substrate. The dielectric properties of glass make it ideal for low-loss, very high-speed applications. This allows the ICs to be directly placed on the glass substrate, and makes glass suitable for multi-chip packaging involving very high speeds.

Figure 11: Glass substrate for packaging

### 3.2.3 Optics and Micro-coax to the Package

In order to increase package I/O bandwidth without increasing power consumption, new miniature onboard optics modules have been developed that can be placed in close proximity to the IC package.  Samtec's Firefly flyover system, compatible with both optical and micro-coax module, is an example of such a solution. It is currently available at speed of 28Gb/s per lane.


Figure 12: Samtec Firefly

To take full advantage of the increase in speed and reduction in power enabled by the USR low-power SerDes, a further evolution is to place the flyover module connector directly at the edge of the package.

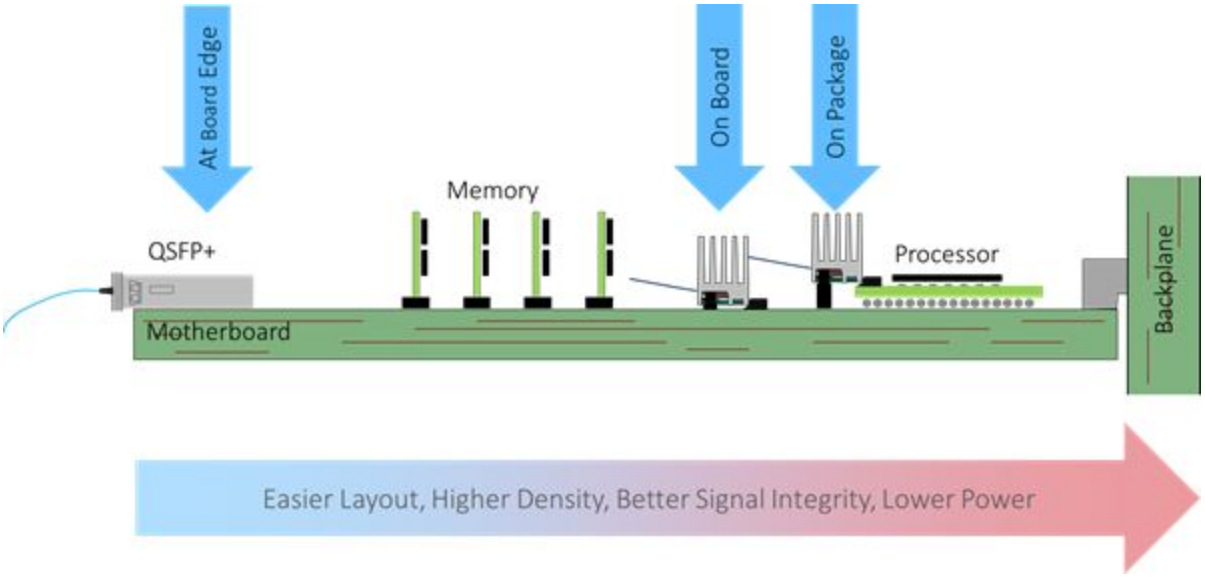



Figure 13: Connector positioning for fiber to the package

The Firefly connector can be small enough to fit on the edge of the package as shown in Figure 14.

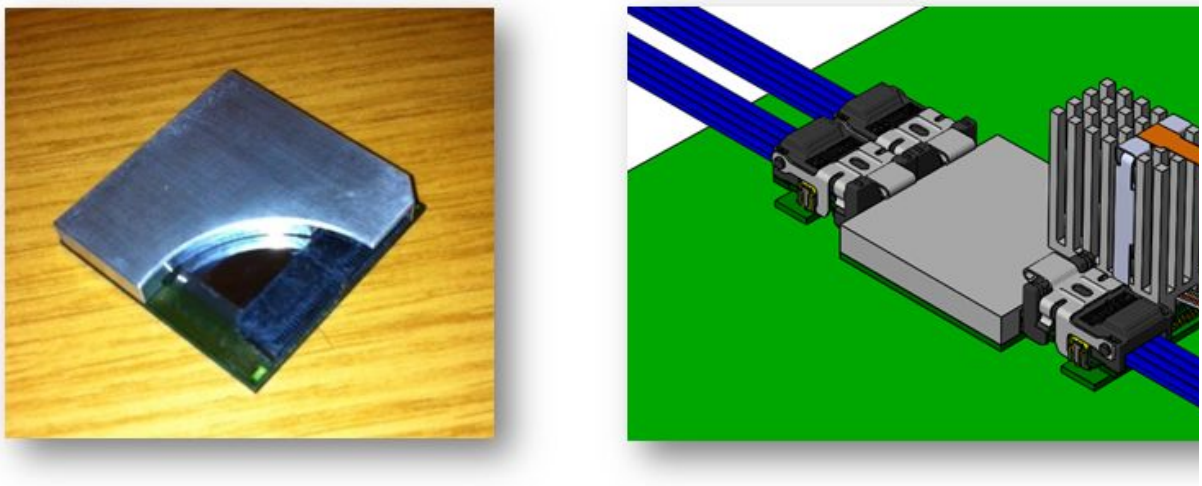


Figure 14:  Two Firefly connectors embedded in a 45mm square package

Furthermore, with the optical module is such close proximity, one can dispense with the usual CDR repeater in the optical module. Direct-connect versions of Firefly modules at 56Gb/s (shown below for both the optical and electrical versions) are currently in development.

## 3.3 Instruction-Driven Switch Fabric (ISF): Scalable Data Transfer

Typically, on-chip communication between peer elements on an ASIC SoC is synchronous. Full-chip communication networks incur a significant area and power penalty for deep pipelining which is necessary to enable this synchronous communication. The larger the area of an ASIC, the greater the penalty for synchronous on-chip communication. Netronome has developed and used a light-weight message-based protocol for on-chip data communication. This allows full-chip communication to be implemented using a simple scalable-distributed switch fabric.

The ISF interconnect is the main global bus in the Netronome Network Flow Processor (NFP). As shown in Figure 15 below, an NFP is physically implemented as a tiled array of Logic Blocks (or Islands). Each Logic Block is connected by a simple BoW interface with its immediate physical neighbors. Data transfers on the ISF are orchestrated programmatically. The ISF command syntax is extensible and supports commands for data transfer and even processing at remote bus agents in a different island to process data at a remote location where it resides minimizing data movement and processing time.

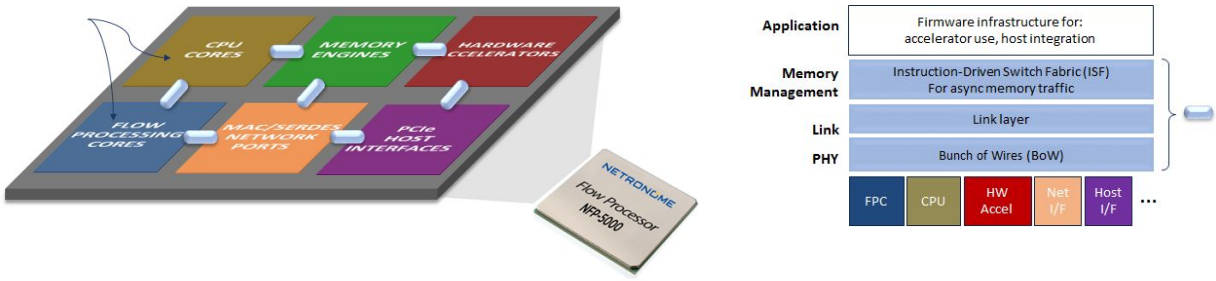



Figure 15: Scalable ISF and data transport protocol

In current generation silicon, each Logic Block can have up to six DSF interfaces (K=6). Each 64-bit wide DSF interface link at 1GHz delivers 128Gb/s of bidirectional bandwidth into each Logic Block at nodal point. So, a total of K*128Gb/s throughput is theoretically possible across each Logic Block. In the example in Figure 15 each Logic Block can support 768Gb/s of bidirectional bandwidth. Latency of access to data across the die (in a typical Netronome product) composed of the Logic Blocks varies between 5 and 20 nanoseconds. The distributed implementation offers the following advantages:

- High Bandwidth: It enables high-aggregate bandwidth, comparable to a crossbar, because many transactions can be executed in parallel. The ISF has separate data/command paths and masters for each island. With distributed arbitration, many near-neighbor connections can operate simultaneously. The bus can be clocked at a higher frequency because all wires are local.

- Power-efficient: With almost no global physical connections, the ISF dissipates less power and can be clocked faster than a global bus.
- Scalable: The ISF infrastructure is scalable across many different Logic Block instances, supporting rates peaking at billions of commands and terabits of data transfers per second giving this distributed bus throughput comparable to a crossbar. However, such a physically distributed bus consumes less silicon die area.

We propose to extend this scalable design to off-die data communication in a multi-chip package.

# 4 Domain-Specific Accelerators

Hardware accelerators of different types have been proposed and developed from almost the same time as general-purpose CPUs. The end of semiconductor growth along Moore's Law has triggered a significant rise interest in accelerators. Indeed, both the Amazon and Microsoft Azure clouds give users the ability to create custom accelerators by providing cloud servers with FPGAs and GPUs attached for general use.

Domain-specific architectures, as the name implies, are tailored to a class of workloads that share a common characteristic. The devices are programmable, not hardwired as are traditional ASICs. They feature integrated application and deployment-aware development of devices, firmware, systems and software and they support domain-specific languages for ease of use. Key attributes of a domain-specific architecture are parallelized data processing, function-specific logic, application-aware data management and control. As shown in Figure 18, the result is significantly better performance/watt relative to CPUs and FPGAs. Two well-known, domain-specific silicon-based examples are shown, namely Google’s TPU (Tensor Processing Unit) for machine learning [17, 35, 52], and AI, and Netronome’s NFP [61] for networking and security.

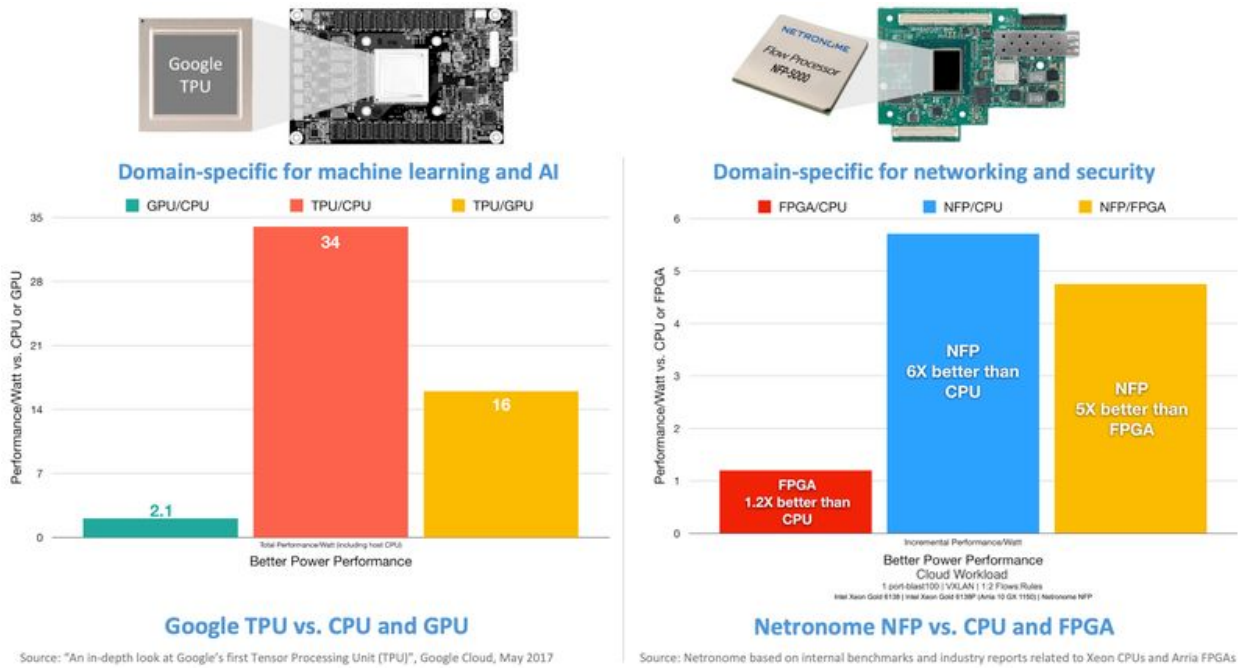



Figure 16: Domain-specific silicon delivers higher performance/watt

This section surveys recent commercial work and academic work in accelerators. We define accelerators broadly as any non-CPU hardware dedicated to running domain-specific workloads. Systems based on GPUs, FPGA and ASICs are all DSAs under this definition. We also include neural-net accelerators. Neural nets are not programmable with microcode, but are instead programmed by adjusting the weights on connections between neurons in the network.

## 4.1 Commercial Accelerators

Commercial accelerators in a wide range of applications are surveyed in this section.

### 4.1.1 Machine Learning

ML (Machine learning), especially based on neural networks, has been the most popular use case for commercial applications. Initially, most efforts at acceleration focused on scale-out acceleration, reducing compute time by distributing the workload across an ever-growing pool of servers. As energy costs became untenable at larger data rates, accelerators began receiving more commercial attention. Data center operators have adopted different approaches to accelerating learning (the act of building a model) and inferencing (the act of making predictions with the model) workloads in ML. He et al. refer to server systems with accelerator hardware and special networks for acceleration as AI-able servers and provide a comprehensive overview of AI-able servers at all large data center operators [8].

Many learning systems are built on NVIDIA GPUs.  NVIDIA has reference designs for learning (DGX-1) [21]. Facebook developed an Open Compute platform, the Big Basin, for learning tasks

in the data center [12]. Many companies are also developing specialized accelerators targeting learning. Tang maintains a list of silicon/IP solutions for ML 68]. Among large commercial vendors, NVIDIA Volta chips and Intel Nervana (in development ) and Movidius chips are examples of DSAs for learning. While NVIDIA's efforts focus all ML activities on GPU-based architectures, Intel advocates for a heterogeneous approach spread across multiple-types of computing elements. Intel also provides platforms that combine an x86 CPU with an FPGA for example. Xilinx, similarly, has a heterogeneous learning acceleration product, Everest, that combine FPGAs, general-purpose CPUs and a network optimized for neural network processing. As with any emerging area, ML startups have attracted a significant amount of venture funding. Wave Computing [67] and Graphcore [5] are examples of companies with domain-specific silicon targeting learning.

While learning is typically an offline activity, inferencing workloads are continually executed on real-time data in a production environment. Hence, it occurs in multiple physical locations from the network data center through the network edge and at end devices. Today, inferencing has more performance, throughput and latency constraints relative to learning. Inferencing can also be executed with reduced precision, for example with 8-bit, 4-bit or even 1-bit precision, instead of 32-bits and with integer precision instead of floating point. Consequently, many accelerators targeting inferencing have been developed in academia and industry. Sze et al [23] provide a comprehensive survey of academic and commercial efforts to accelerate inferencing with domain-specific hardware. NVDLA is an open-source framework to accelerate inferencing developed by NVIDIA that's capable of supporting multiple types of hardware accelerators. Tensorflow is the most well-known inferencing accelerator widely used by Google in production data centers. An accelerator from Cambricon is optimized for the reduced data flow in inferencing in sparse neural networks, built by pruning (or reducing) neural networks after learning without sacrificing performance [15].

### 4.1.2 Cryptocurrency

Bitcoin mining and cryptocurrency mining in general is a compute-intensive activity. Early in its development, Bitcoin mining was executed on CPUs. Early acceleration efforts focused on mining on GPUs and FPGAs. ASICs to mine cryptocurrency, primarily Bitcoin, received a significant amount of investment, primarily because they offered significantly better power-performance than FPGAs and GPUs. These ASICs accelerate the hash operations in cryptocurrency. Taylor [54] provides a good overview of recent developments in cryptocurrency mining. Bitcoin Wiki also provides a good overview of cryptocurrency ASICs [63]. One unique aspect of currency miners is that they are almost always deployed as arrays with multiple accelerators.

### 4.1.3 Other Applications

Companies have produced DSAs for several applications beyond ML and cryptocurrency mining. Netronome has produced several generations of a DSA for networking functions, the NFP used in SmartNICs and networking appliances. ThinCI recently introduced a Graph

Streaming Processor for IoT applications to process multiple parallel data streams [30]. Memcached is a distributed memory-caching system designed to reduce the cost of disk accesses. LegUp computing developed a memcached accelerator deployed on Amazon servers with FPGAs attached [9]. Authors from HPE proposed SoC accelerators for Memcached that would be implemented on SmartNICs [55]. One newer area that has received significant attention is accelerators for near-memory computing. These accelerators are designed to reduce memory traffic on system buses by moving processing elements close to the external/internal memory interface. IBM covered the use of near-memory processing for supercomputing applications, and HPE discusses the use of near-memory processing in the "The Machine" architecture.

## 4.2 Academic Accelerators

Academic efforts may be taken as a leading indicator of workloads that can be accelerated commercially or new approaches to accelerating known workloads.

### 4.2.1 Machine Learning

As it has with industry, ML has received substantial research in academia. FPGAs have been particularly fertile ground for research on neural network acceleration. Guo et al provide a survey of multiple, primarily academic, efforts at accelerating FPGAs [64]. Academic research has also focused on accelerating specific properties of neural networks, such as sparsity [48]; types of neural networks such as HMMs [41]; developing new approaches to store the matrix of weights in neural networks in accelerators [51]. The survey paper referred to above also has a good overview of academic hardware accelerators for inferencing. Neural network accelerators expend significant time (and silicon area) in transferring data from memory to the execution logic. Researchers have investigated reducing these costs by directly incorporating execution logic into the memory or at memory interfaces. In general, at-memory execution requires that neural network complexity be reduced [58].

### 4.2.2 Near-Memory Computing

The potential power-performance improvements from near-memory/in-memory acceleration have received significant attention from academia [25, 43]. Accelerators have focused on reducing the data transfer involved in memory-intensive applications. Some examples are reducing the time to traverse trees in memory, reducing the cost of hash, sort join operations required for databases, and placing reconfigurable logic directly at DRAM interfaces [22].

### 4.2.3 Other Applications

As expected, the range of DSA applications investigated in academia is broader than those in commercial applications. Lindsey et al. [6] investigate both a domain-specific language and an accelerator for Monte Carlo simulation. Ng et al [46] survey two decades of development of FPGA accelerators for genetic sequence alignment. Wu et al [45] proposed the Q100 database

accelerator. Researchers from Korea have recently proposed an accelerator for LZ Compression [20].

### 4.2.4 Accelerator Frameworks, Etc.

Researchers have also investigated a number of interesting topics relevant to building accelerators. These efforts are aimed at reducing the development cost of accelerators realized in FPGAs or as ASICs. Lindsey et al show that if the cost of developing accelerators can be reduced, say by developing accelerators at older silicon process nodes, application TCO can go down by deploying custom ASIC clouds in data centers. One interesting academic focus has been on developing frameworks for accelerators. Cong et al [10] discusses a framework to enable memory coherence between host and accelerator memory. Celerity [66] and Basejump [53] are open source accelerator frameworks based on a many-core architecture where the cores are lightweight cores based on the open source RISC-V ISA. Gray also demonstrated a many-core RISC-V based implementation on an FPGA. Typically, accelerators require the coordinated development of hardware and software. COSMIC is a framework that attempts to co-synthesize the hardware and software for neural network accelerators in a scale-out framework. LSSD is a framework for that composes architectures automatically from multiple cores. Moorthy and Gopalakrishnan [65] propose a service framework for FPGAs where custom logic can drop into a framework that provides the I/O and memory access required by any accelerator [65]. They estimate this reduces development effort significantly. ST-Accel is an accelerator synthesis tool focused on streaming applications [49].

### 4.2.5 Domain-Specific Languages

Researchers have also focused on creating domain-specific languages as a mechanism to express and abstract primitives for acceleration. DSLs also become a mechanism to port application workloads across multiple implementations without significant development effort. P4 is a domain-specific language, proposed in academia, to describe the network data plane that has received a substantial amount of both academic and commercial attention [69]. Initially, efforts in P4 focused on describing traffic in network switches. More recently, it has been shown that P4 also has significant value as an abstraction layer for accelerators and portability at network endpoints [71].

One noteworthy new effort is that of creating a VM environment, based on the Tensorflow language from Google, to execute neural net workloads in software [56]. This approach parallels the eBPF VM used to enable custom networking in the Linux kernel [70]. Netronome recently demonstrated the ability to transparently port eBPF execution to accelerators, without any developer effort specifically for acceleration [72]. The Tensor VM may offer similar benefits in transparently porting neural network workloads to accelerators.

# 5 Open Domain-Specific Architecture (ODSA)

In addition to being application-aware, effective DSA designs are vertically integrated. Silicon design needs to comprehend both the developer programming model for firmware on the DSA itself and how the DSA is integrated into the system-level application data flow and control management.

The ODSA is based on a reference multi-die architecture that recognizes the need to support integrated development. It is derived from a reference monolithic accelerator design that is in turn based on the common elements found in accelerators for a wide range of applications. The intent of the multi-die reference design is to preserve the software model from a monolithic accelerator. This is made possible by a new cross-die link layer that abstracts away the new inter-die connectivity in a multi-die accelerator. The reference multi-die architecture is based on a reference monolithic accelerator architecture that incorporates elements found in a wide range of commercial and academic accelerators.

## 5.1 Monolithic DSA Architecture

Researchers have aimed to specify the architectural characteristics of a DSA, to develop methods to automatically synthesize them. The following list of characteristics is based on the the lists in [44].

General-purpose CPU design is centered around handling diverse applications with varying degree of parallelism and unpredictable control flow. In contrast, almost all accelerators are aimed at exploiting the parallelism and deterministic control flow present in a specific algorithm or range of algorithms for the application. Specifically, algorithms based on light threads. For example, in a neural network, the computations associated with a single neuron are relatively simple, but each network consists of several thousand neurons.

### 5.1.1 Common DSA Components

The logic in accelerators can be split into three categories:

1. Algorithm-Specific Logic aimed at concurrency based on a specific control flow:
    a. Hardware concurrency: a large number of functional units, each typically with a simple datapath to exploit the the enormous parallelism typically present in accelerated algorithms. GPUs and Neural Network accelerators are common examples.
    b. Application-Specific Logic: The functional units themselves contain problem-specific functional units for computation. For example, NPUs may contain bit-manipulation logic.

    c. Hardwired-Control Flow: The predictable control flow is reflected in a network that implements explicit communication of data between units. For example, many neural network accelerators have a near-neighbor array structure.
2. Coordinated Execution Logic: The software in programmable accelerators typically do not function in isolation. Instead accelerator code is typically executed in coordination with software on a general-purpose CPU. Most accelerators contain additional hardware, to coordinate the application-specific logic with software and memory on the host:
    a. Host interfaces to general-purpose CPUs
    b. Significant amounts of internal memory. Interfaces to external memory. This is not specifically coordinating logic. We include it in this section as memory implementation is largely application independent.
    c. Hardware support for data structures and reuse. Engines for bulk data transfer and/or agents to support memory-coherence with a general-purpose CPU.
    d. General test and debug hardware to support manufacture and operations (in commercial accelerators)
3. An on-chip network in the form of a network-on-a-chip or one more buses that links the application-dependent and application-independent logic.

### 5.1.2 Reference Architecture

The following proposed reference design, made of four components, captures the primary components of an accelerator. The significant components are:

1. Logic elements
    a. Domain-specific logic
    b. A general-purpose CPU
2. I/O elements
    a. A network interface
    b. A host interface
3. Intra-die networking logic
    a. An interconnection network
    b. A communication agent at each logic and I/O element

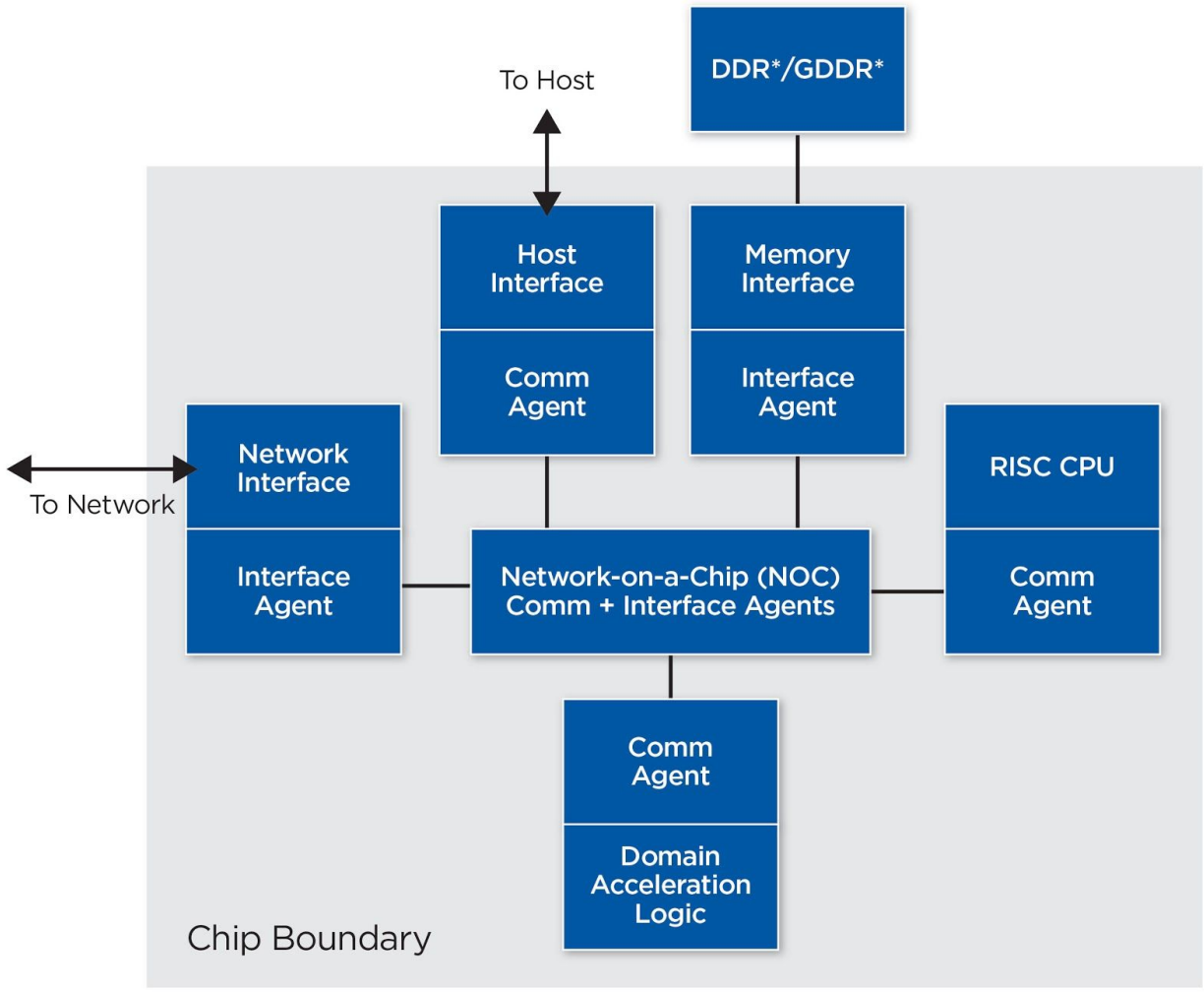



Figure 17: Reference monolithic domain-specific accelerator architecture

The interconnection network may be a network-on-a-chip or a set of one-or-more buses. The comm agents may range from simple engines to receive and transmit data to more complex logic to maintain memory coherence across the various logic elements on the chip. The application itself is usually decomposed into firmware and software running on the domain acceleration logic, the interconnection network (for data transfers), the RISC CPU and the host.

The table below maps a few of the accelerators discussed in the section above onto the components:

| Accelerator | Application-specific or Acceleration Logic | Logic for Coordinated Execution | Network |
|---|---|---|---|
| Tensorflow | MAC Array | PCI interface<br>Transfer Engine | Buses |

| | | | |
|---|---|---|---|
| Intel Xeon-FPGA Platform | FPGA | PCI Interface | N/A |
| Xilinx Everest | FPGA<br>Hardware accelerators | Host interface<br>ARM CPU | Proprietary<br>Network-on-a-chip |
| TVM | Accelerator on FPGA | Non-coherent memory | Buses |
| Stream Processor | Coarse-Grained Reconfigurable Array | Coherent Memory Interface | Buses |
| Bioinformatics | DALIGN on FPGA | Microcontroller<br>Network Interface | Near-neighbor connections |
| HP The Machine | SoCs, Flexible | Coherence fabric for attached NVM array | N/A |
| Memcached | Memcached core on FPGA | Virtual network over | PCIe |

## 5.2 Reference Multi-Chiplet DSA Architecture

We develop a reference multi-die architecture for DSAs derived from the reference monolithic architecture. The monolithic design is decomposed into individual chiplets as follows:

1. A network I/O chiplet
2. A RISC CPU chiplet
3. A DSA chiplet which may be implemented as:
    a. An FPGA
    b. A many-core RISC processor
    c. Domain-specific logic
4. A switching and interface chiplet to which all the other chiplets are connected

These chiplets are now packaged together on a common substrate. The chiplets will have to implement comm agents to support inter-die networking.

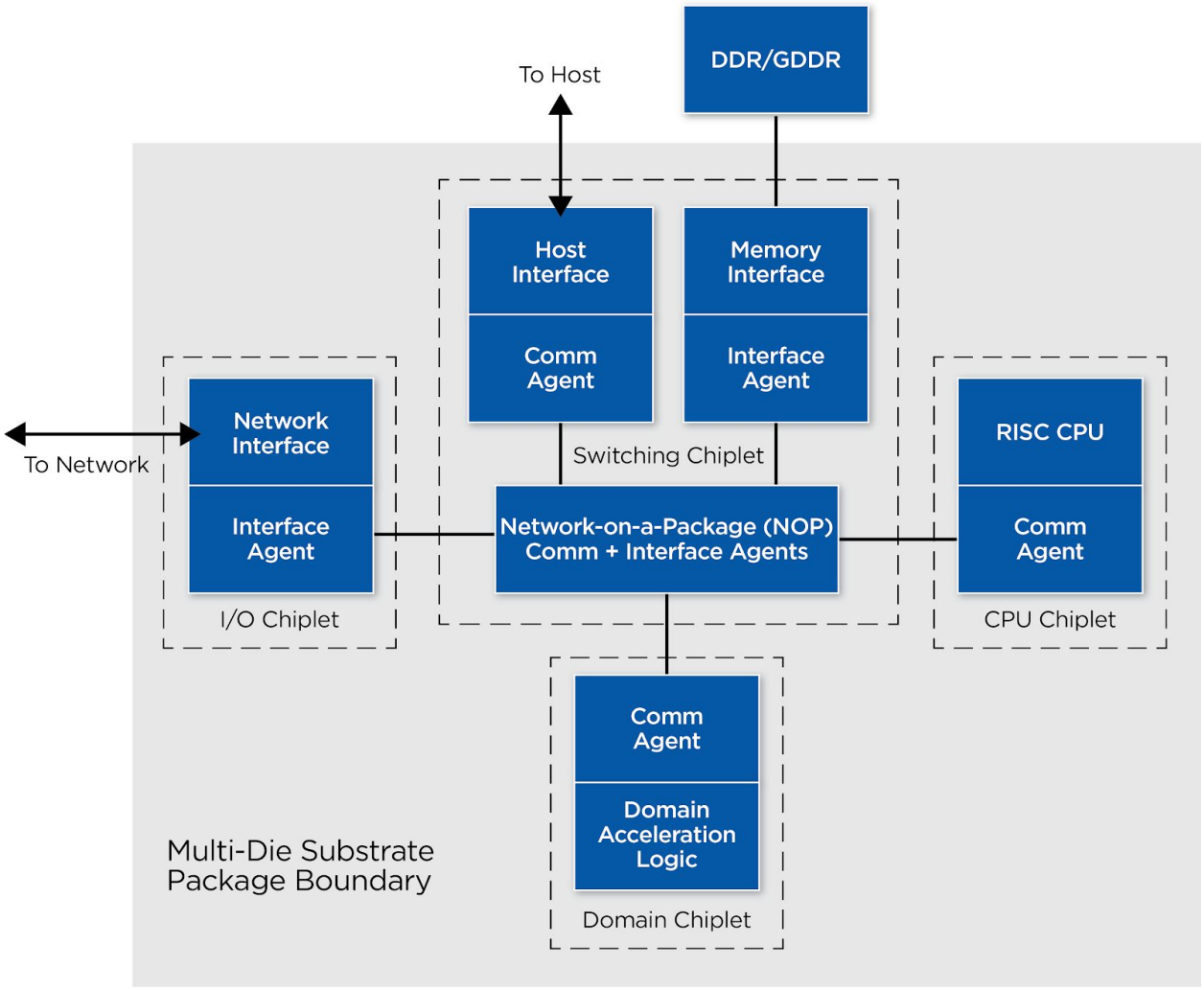



Figure 18: Reference multi-die architecture

The objective is to enable applications to be developed and executed on the reference multi-die architecture similar to the way they are developed on a monolithic architecture. One additional complexity is that unlike a monolithic IC, the die may actually be produced by different vendors.

To present a monolithic IC-like development environment, the multi-die architecture must offer a memory model and performance comparable to a monolithic architecture. The memory model needs to be implemented over an open transaction layer to enable easy interoperability across die from different multiple vendors. Finally, performance must be achieved with a PHY connectivity technology that enables the architecture to be implemented with cost-efficient organic substrate packaging technology.

In summary, to compensate for the difference between a monolithic and heterogeneous implementations, the reference architecture has to develop a complete protocol stack that addresses the following issues:

1. A network layer to move data around with three sublayers
    a. The PHY layer: Protocols for physical inter-die communication
    b. The link layer: A protocol for logical communication between physically connected die
    c. A routing layer: A protocol to route transactions between devices not directly connected
2. A memory layer that is the infrastructure for all processing activity in the ODSA
    a. Transaction Layer: A common transaction model for moving data between processing elements
    b. Sublayer 2: Protocols for memory management across processing components
3. The application layer: Firmware and software to integrate the accelerator with a host

With this approach, higher layers can be preserved even as the design evolves. The rest of this section develops the components of the reference multi-die architecture.

## 5.2.1 Memory Layer

The memory layer serves as the infrastructure for all communication between computing processes in the ODSA. Host-accelerator communication is through memory as well as any communication between logical elements. The ODSA supports memory managements through a common model for memory transactions.

### 5.2.1.1 Common Transaction Model

In current systems, the process for off-die communication is different from that for intra-die communication. In multi-chip packages, cross-chiplet communication needs to be functionally identical to the process for intra-chiplet communication, though it may be less performant. The common transaction model is the interface between the PHY layer and the memory management protocols. The common transaction model abstracts the specifics of the physical communication from the accelerator applications and the memory transaction protocols supporting the applications. This section outlines the requirements for the common transaction model and current protocols that may meet those requirements.

Accelerator applications need to support a wide range of data transfer sizes for different applications, each with differing latency requirements:

1. ML applications require the transfer of several gigabytes of data for each object processed by an accelerator.
2. In networking applications, object sizes may range from 64-1500B and each object may need control data of about 1-2KB
3. Across all applications, cache coherence protocols may need very low-latency transfers as small as a cache line.

ODSA implementations are expected to be assembled by combining chiplets from multiple vendors. In such systems, not all devices with communication agents will have the same

capabilities. Secondly, the number of inter-chiplet physical links is expected to be an important resource. Heterogeneity at scale creates specific requirements.

1. In large systems with more than one switch, transactions may need to be routed. Even with buffers at links, simple routing at switches is desirable.
2. Read and write memory transactions that cross chiplet boundaries should not stall and block the use of an inter-chiplet link. That is, completion should be guaranteed.
3. Enable a device discovery or registration system that enumerates the specific capabilities of each chiplet.

A split-transaction credit-based protocol based on messages with support for burst data and size-limited posted writes can meet these requirements. The specifics of the protocol remain to be developed. The proposal is to use existing protocols either directly in their entirety or as a starting point for development. Specifically, to use PCI Express and the new open Tile-Link transaction models as starting points. PCI Express has the advantage of being widely used in current devices. Reusing PCI Express will give the ODSA an ability to reuse a large number of legacy die. PCI Express has the constraint of being oriented to device-to-root transfers. When inserted into a host, a PCI Express system on the host may need to coexist with the host-based PCI Express system, increasing complexity. Further study will be required to completely specify the common transaction model.

#### 5.2.1.2 Memory Management Protocols

The ODSA will run multiple types of memory management protocols on the common transaction model to support a diverse set of accelerator applications. The ODSA envisions supporting two types of memory read and write flows within memory.

**Coherent Flows**: All processing elements with coherency agents maintain a consistent hardware-supported view of coherent memory space. Any reads will produce the most recent values. Any writes will invalidate any physically distributed copies. Coherency protocols expend latency (hidden to the software developer) to maintain coherent state. For this reason, performance is best when the physical area over which coherence has to be maintained is limited. As with the CTM layer, the ODSA will look to reuse an existing memory synchronization protocol such as CCIX or TileLink.

**Non-Coherent Flows:** Many accelerator applications require the predictable transfer of large blocks of data. For example, inferencing flows for ML require the transfer of a large set of weight data. Non-coherent flows may be better suited to the large scale data transfers required by many applications. Netronome uses a novel ISF (instruction-driven switch fabric) protocol for non-coherent intra-chip data transfers. The protocol uses a credit-based link layer over distributed near-neighbor connections. The ODSA will support the use of the ISF protocol over the common transaction model.

### 5.2.2 Network Infrastructure

The ODSA needs to provide a network infrastructure to enable the common transaction model.

#### 5.2.2.1 Data Link Layer

The physical communication protocol technology between chiplets in a multi-die package is still evolving. The choice of a specific Phy protocol for a system is driven by a combination of performance requirements and cost constraints. For example, for a part without significant performance requirements, it may be more economical to reuse an existing die with PCI Express connectivity than to develop a new product with short-reach SerDes connectivity. Similarly, certain functions may evolve to be uniquely supported by specific protocols. As an example, the XSR protocol may be the standard with which chiplets that terminate optical connections connect to a system.

As previously discussed, the ODSA proposes that all Phy layer protocols present a PIPE interface to the higher layers.

#### 5.2.2.2 Transaction Routing

As in TileLink all message routing in the ODSA will be by address. The ODSA expect  to support both static and dynamic routing algorithms.

### 5.2.3 External Interfaces

The ODSA will also support multiple types of external interfaces. These interfaces typically involve a significant amount of analog circuitry that do not benefit as much as digital logic from a process node feature size reduction. Hence, they may be best implemented as separate chiplets.

Data on external interfaces will be delivered on interface agents that do not support the common transaction model. The Phy layer for each interface will be specific to the type of interface. Further, data I/O to/from these interfaces is expected to occur through the switching chiplet, enabling them to appear as pooled resources to the rest of the system.

1. Network Interface: The network interface is expected to be implemented as a separate chiplet so the network I/O bandwidth can be changed independently of the rest of the system. The network interface may either be an optical or electrical interface. Data from the interface is directed to the switching chiplet through a standard OIF interface, such as XSR. The data on this interface is not expected to be delivered on the common transaction model.

2. Host interface: The host interface is expected to be a later generation PCI interface. Some applications may require that the memory in the accelerator be coherent with the

host's memory as well. Therefore, unlike the network interface, the host interface will also need to support the common transaction model. We expect the host interface to be integrated with the switching chiplet.

3. Memory interface(s): The switching chiplet is expected to support an external memory interface. To meet latency and/or throughput requirements, the RISC CPU and accelerator chiplets may each have their own interfaces. It is possible that for some applications, a switching chiplet with a High-Bandwidth Memory (HBM) interface and high-bandwidth USR Phy layer may meet the throughput and latency requirements of both the RISC CPU and the accelerator logic. That is, external memory can be a pooled resource accessed through the switching chiplet.

### 5.2.4 Software Execution Model

Effective DSAs require the integrated development of silicon, systems, firmware and software. Two software issues relative to DSA development that have received a lot of recent attention: domain-specific languages and the offload model will be discussed.

There are two broad models used to offload functionality from a host to the DSA, transparent and directed offload. Both models also require a driver on the host that is used to manage the DSA from the host.

#### 5.2.4.1 Domain-Specific Languages

While DSAs are programmable and not fixed-function ASICs, almost all DSAs contain programmable functions unique to the application domain. One business concern with DSAs is the need to create code custom to a specific DSA that is not portable to other devices implementing similar functionality. For example, code for one machine-learning IC is not executable on another machine learning IC.

Domain-specific languages address this issue. These languages specify acceleration primitives common to a domain. DSAs execute code in a DSL by compiling the program to the hardware. Performance is achieved by supporting the performant implementation of language primitives. Code written in a DSL is portable to any accelerator that has a compiler for the DSL. DSLs enable performance to be combined with code portability across multiple devices. Commercially, TensorFlow from Google for machine learning is the best known DSL. TensorFlow is the DSL used to program the TensorFlow ASIC. TensorFlow was also used to program a portable VM-based neural accelerator. Two DSLs, P4 and eBPF, have seen increased use in networking applications. P4 was originally designed for core switching applications and then extended to server data center applications. For example, P4 has been used to implement the Gen-Z memory-aggregation switching protocol. eBPF was designed to define custom networking in the Linux kernel and has recently seen renewed interest as a simple model for offload.

The ODSA will support the use of DSLs. For example, both P4 and eBPF for networking applications. The ODSA will leverage Netronome's open source contributions in these areas.

#### 5.2.4.2 Host Software

In most acceleration applications, the datapath running on the DSA is coupled with a datapath on a general purpose CPU. In the ODSA, this may be the RISC CPU in the ODSA, or the host to which the ODSA is attached. The ODSA will support two models for host interaction:

- Explicit Offload Datapath Programming: The accelerator datapath is functionally separate from the host datapath. The datapath on the accelerator is programmed separately by the developer. The host controls the datapath through an API. Data is sent from the host to the accelerator datapath using explicit function calls on the host. Two examples of this model are the first generation TensorFlow ASIC from Google and P4-programmed datapath on Agilio SmartNICs from Netronome. This approach will require an open-source interface from the host processor to the offload datapath. The P4 Runtime is an example of this type of interface.

- Implicit Offload Datapath Programming: In this model, the accelerator is not invoked or programmed explicitly within the datapath. External control switches execution from the host to the accelerator. Most commonly this is achieved by the programmer developing the target functionality in a VM. This approach requires greater open source support from the host operating system. Both the host and the acceleration logic will need to support any DSL used for the application. The application is accelerated by moving the VM from the host to the accelerator. Two examples of this model are eBPF offload from Netronome and the TensorFlow VM from the University of Washington.

The ODSA will support both explicit and implicit offload datapath programming models.

# 6 ODSA Implementation

To demonstrate the benefits of the ODSA, the proposal is to implement a prototype that supports both coherent and non-coherent memory transactions. To demonstrate the benefits of an open architecture, the ODSA will implement a PoC with current versions of these chiplets.

## 6.1 Prototype Configuration and Chiplets

The ODSA proposes to implement the reference architecture in Figure 19 with a Kandou USR as the inter-chiplet connection protocol, on a low-cost organic or glass substrate. On the USR, we will implement both a coherent memory protocol and extend the ISF to inter-chiplet non-coherent transactions. The Link layer will implement the common model for memory transactions.

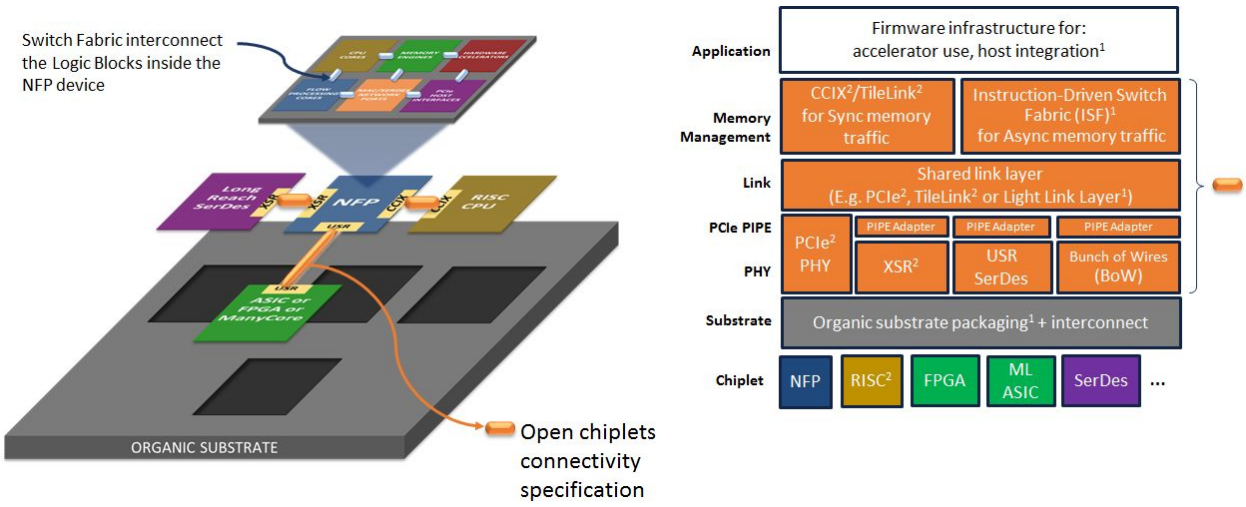



Figure 19: ODSA Reference Architecture implementation with USR SerDes and Glass substrate

We briefly review chiplets to be used in developing the PoC and final ODSA prototype.

### 6.1.1 Network Flow Processor

The inter-chiplet switching, and host interface will be provided by a Netronome NFP. The PoC will use the NFP-6000 and use its four PCIe interfaces to provide connectivity to other processing elements and the host. The prototype and PoC will also use the extensive set of open source drivers and offload firmware and host software available with the NFP family.

### 6.1.2 Fiber to the Package Optics

When development is complete, we will use the Fiber to the package optics solution discussed in Section 3.

### 6.1.3 FPGA

FPGAs provide reconfigurable hardware on which custom hardware, including DSAs, can be dynamically implemented. When a high-volume of identical accelerators are needed, custom logic is often more efficient. There are many cases when FPGAs can be the prefered solution including when a datapath can be optimized for specific instances, or when the volume is lower, or when you want to reuse the same silicon for multiple DSAs at different times. FPGAs are also very efficient for simpler transformations of data encoding between acceleration phases to enable streaming together otherwise incompatible accelerators.

A candidate for the prototype is the Achronix Speedster22i HD1000 FPGA. The HD1000 provides 700,000 LUTs, 700,000 Registers, and 10MBytes of reconfigurable RAM blocks for implementing custom logic. It also provides a 100 Gigabit Ethernet MAC and PHY, DDR3 controller. The HD1000 would be connected via a PCI Express Gen 3 connection as this legacy device does not have CCIX support.

### 6.1.4 RISC CPU

The PoC and prototype will use application processors based on the U7 series RISC-V cores from SiFive.

## 6.2 Non-coherent Transaction Model PoC

We aim to develop the first ODSA PoC primarily out of die currently in production or development, that is without requiring a new tapeout. The objective of the PoC is to demonstrate the value of the abstraction provided by the ODSA and to serve as a platform for application and protocol development. Specifically, the PoC will demonstrate applications in networking and inferencing.

The PoC will be built from the following components:

| Function | Component | Status |
|---|---|---|
| Switching | Netronome NFP-6000 | Production |
| RISC CPU | SiFive FU700 | Production |
| Connectivity | LR PHY | Production |
| | Fiber and microcoax to the package | Development |
| Accelerator | FPGA | Development |

The PoC will support the following protocol stack:

| Layer | Implementation |
|---|---|
| Packaging substrate | Organic/Glass |
| PHY | PCIe |
| Link | PCIe |
| Transaction model | Descriptor-based data transfer |
| Memory transaction | Non-coherent |
| Switching protocol | P4, eBPF |
| Application | Networking |
| | Inferencing |

### 6.2.1 Prototype Packaging

Since the total number of interconnections in between the NFP die and the other dice are less than 100, it is very easy to implement on an organic substrate and then package the multi-die in a MCM package. Using microstrip line and stripline to route high speed signals and other low speed I/Os, a 6 layer (2+2+2) substrate might be enough to complete the routing task. Today, a 14 layer (6+2+6) package which offers up to four high speed routing layers is easy to make. Even a 20 layer (9+2+9) package can be in production.

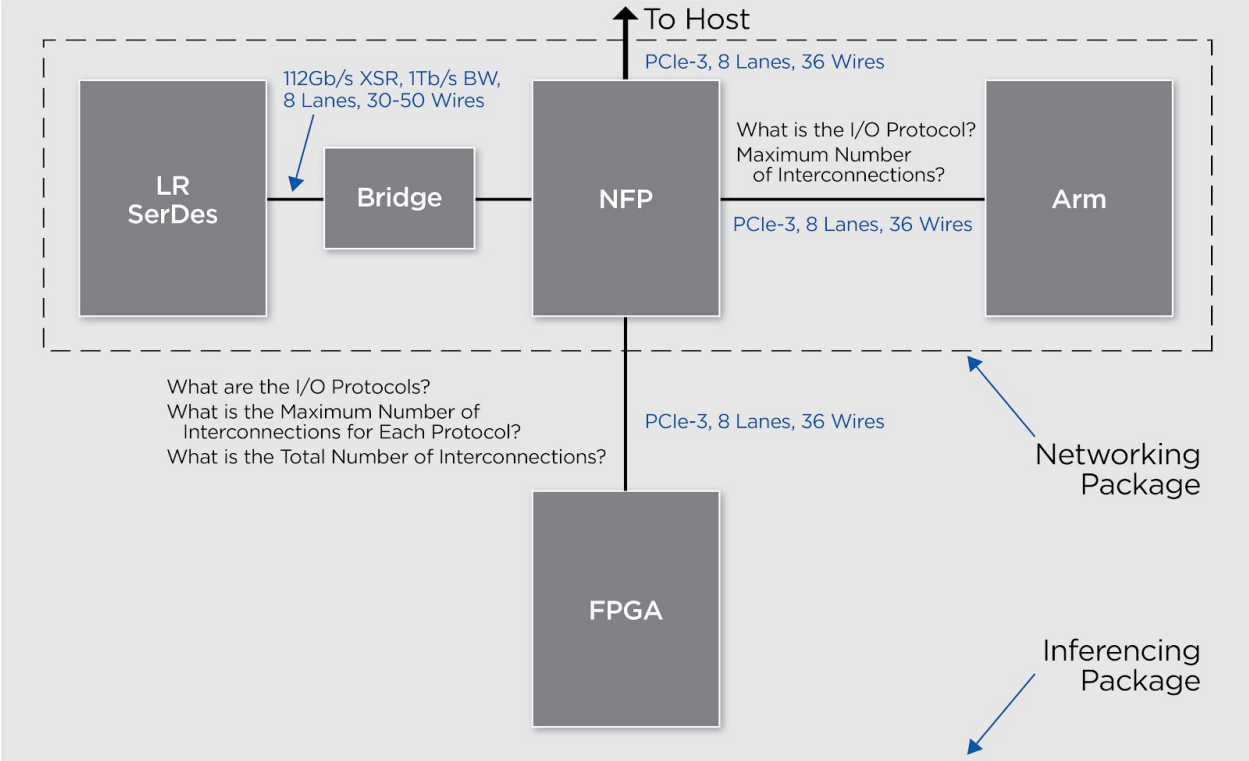



Figure 20. Networking and inferencing package on demo system

# 7 ODSA Deployment/Use Model

We envision a wide range of applications for the ODSA in high-performance networking in a wide range of applications.

## 7.1 High-Performance Scientific Networking

The Energy Sciences Network (ESnet) is a high-performance, unclassified network built to support scientific research. Funded by the U.S. Department of Energy's Office of Science (SC) and managed by Lawrence Berkeley National Laboratory, ESnet provides services to more than 40 DOE research sites, including the entire National Laboratory system, its supercomputing

facilities, and its major scientific instruments. ESnet also connects to 140 research and commercial networks, permitting DOE-funded scientists to productively collaborate with partners around the world.

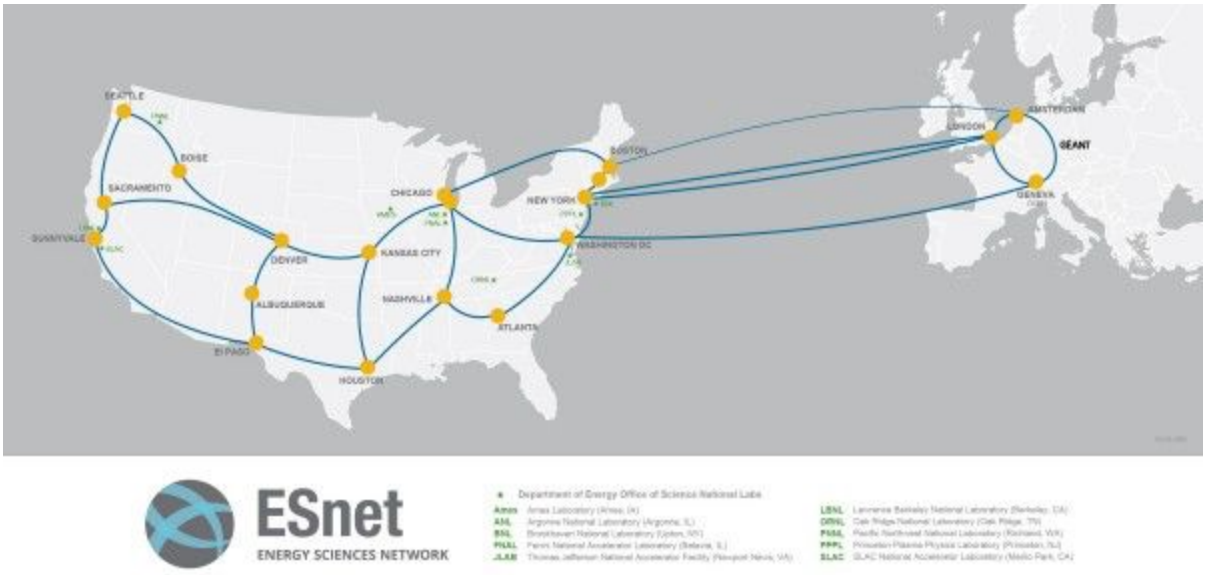



Figure 21: ESNet Network

DOE operates a number of the largest supercomputers in the world, as well as interconnecting large scale scientific instruments like the Large Hadron Collider with 100s of thousands of distributed computing cores. This mission drives the need for high-bandwidth networking, transferring 100s of petabytes of data every month.

A common design pattern for these facilities is to have a set of Data Transfer Nodes which connect to the WAN, in order to act as gateways for large scale data transfer. These nodes need to be performant in order to sufficiently load traffic into the WAN. ESnet's present generation architecture (ESnet-5) was deployed in 2009 and has been supplying 100Gb/s of WAN bandwidth for petabyte data transfers. The next generation of ESnet (ESnet-6) will scale this WAN capacity to over 1Tb/s on a nationwide footprint. In the next 5 to 10 years, we will be moving Exabytes per month, driven by growth in scientific instrument sensors, as well as Exascale computing facilities. Building data transfer nodes at such a scale is a critical requirement.

Data Transfer Nodes are located on campus at scientific facilities. This could be one of the National Laboratories hosting a DOE supercomputer, or it could be a site that hosts a large scale scientific instrument like a Cyclotron, or other particle accelerator or X-Ray light source.

The ESnet-6 architecture places compute and storage capabilities that are closely attached to the WAN backbone. Some of these nodes follow the role of data transfer nodes ( DTNs) whilst

others are hosted in high value peering locations such as Equinix or Level-3 peering locations. Due to the high cost of space and power in these premium locations, it is really important to place 100G storage nodes that are very low-profile, and low-power. The projection for these nodes is to supply a petabyte of storage, and networking in a compact 2U or 4U form factor.

NERSC Supercomputer at LBNL vs. Commercial Colo Cage


Figure 22: Supercomputer facilities

For a sense of scale, it can be seen that large warehouse sized facilities are made available at campuses housing supercomputers. However for colocation spaces in high demand peering locations like NYC, every unit of rack space matters. In the following sections we will describe the architectural implications of each.

### 7.1.1 Anatomy of a Data Transfer Node

In large science facilities like supercomputers, the primary traffic on the WAN is related to file copies at colossal scale. It is common to move petabytes of data into the compute facility for a single computational run. Highly-optimized ftp servers are deployed for this purpose. These servers are referred to as Data Transfer Nodes ( DTNs ). DTNs are built using standard networking, server and storage hardware in order to provide sustained data transfers at 100Gb/s into the wide area network. For example allowing us to move 1PB of data from Chicago to Berkeley. The primary design objective of these DTNs are as follows:

1. Best-in-class performance from off the shelf components
2. High density and low power from standard SSD and PCIe devices
3. 2U and 4U computer cases, placed in racks for scale

This leads to a complete system solution filling one or more racks, consuming several kW of power, and being placed next to a supercomputer that is the size of a warehouse with a MW power budget. Chiplets in the form of standard industry components (nVME , NIC , M.2) will

help drive the speed and capacity of next gen DTNs. However the really interesting opportunity for modules shines in the use case described next.

### 7.1.2 Anatomy of a Dense Network Cache

As part of a large science data transfer network, the use of network caches and CDNs is a critical component that significantly reduces the amount of storage built into end sites. If the network is unable to provide data quickly, the sites have no choice but to make local copies of data sets. This adds up quickly when datasets are 1PB and larger. CDNs and extreme network bandwidth are required in order to provide data on-demand.

Squeezing CDN storage into a tiny Manhattan colocation space at Equinix or Level 3 is an extreme challenge for both space and power. Instead of deploying a rack with kW power budgets, we are looking at 2U or 4U of total space with a few hundred watts of power.

Packaging can be the first line of attack. It is possible to build specialized CDN nodes which leverage laptop-scale components, which integrate networking ASICs into the same PCB as the CPU and storage, and pack as much storage as possible into the available space. However, if the same system level components are used and simply packing them more tightly, the total power does not scale down, instead the power density increases in terms of Watts/Rack Unit. Chiplets provide the opportunity to tackle the power equation.

CPUs, storage and networking are all pushing the interconnect limits with 50GHz SerDes connections. A node that incorporates flash storage, a set of high power CPUs in order to run FTP, and some NIC cards to produce 100G scale IP connections can easily have 500 or more 50G SerDes links, many of which might carry the extra burden of re-timers in order to drive PCB traces for such a loosely packed system. With chiplets it is possible to move from PCB scale to module scale. Collapsing the I/O requirements dramatically. Ultra-low power SerDes driving module traces, the elimination of retimers and skew, the reduction in voltage drive levels and interconnect resistance is a game-changing methodology for reducing power in applications like ultra compact CDN nodes.

### 7.1.3 Chiplets for Fast Disk to NIC Use Cases

In the previous sections the power and density reduction potential for chiplets to provide much better interconnect for system components was described. Another, and possibly most significant optimization possible with chiplets is to build domain-specific architectures that are not driven by the general purpose “Linux computer with peripherals” model that the present hardware and software industry fits into.

If the data transfer problem is defined to be the task of “running FTP,” then it immediately leads to placing a high-performance CPU into the datapath that can copy files from storage devices onto a socket connection to the network. This is an incredibly wasteful use of a general purpose CPU acting as a broker to bridge storage and networking. We can define the task of placing

storage data efficiently onto the WAN as a “domain-specific” use case that is a moderately small subset of the server marketplace.

With the lower NREs and the ability to perform multi-vendor die integration associated with chiplets, it is possible for system vendors to piece together a more optimal solution for this domain-specific purpose. Eliminating CPUs from the datapath can easily lead to a 50% or better power reduction, which for practical purposes is not available to us when plugging together standard components.

There are countless similar domain-specific use cases that exist in the IT ecosystem. Chiplets become the new “PCB” for system design, keeping in mind that silicon for chiplets is different from silicon for PCBs primarily due to the I/O technology required for each.

# 8 Business Models

The vision of chiplets is a broad ecosystem of thousands of interoperable chiplets built in various foundries that deliver a wide variety of functionality for lower costs, faster time-to-market and more cost-effective innovation. Business models will need to support this vision.

In order for this method to succeed, novel business models need to be established. Integrated ASIC providers have already put in place effective models for integration of High Bandwidth Memory (HBM) modules, memory devices and Known Good Die (KGD) systems. This model can be extended to provide much more complex integration with components from multiple sources. The illustrations below outline such a business model, structured with ‘owners’ for various components.

The chiplet model also can benefit overall investment cost. If a company has real value in developing a machine learning (ML) accelerator, for example, it may not make sense for them to develop a network interface for every possible system. Being able to pull this network interface into the design from a selection of available components reduces the investment that needs to be made developing and building hardware for the network interface. Conversely, a company that builds these network interface chiplets benefits from increased volume, amortizing their investment over a greater revenue stream.

## 8.1 Workflow with Chiplets

The example in Figure 23 shows options for building a component where an RF design house may need to integrate additional complex logic function along with analog IP in various nodes. In this case, the RF design house develops the ‘product’ leveraging partnerships with ASIC

providers to build interface IP and function into a portion of the product.

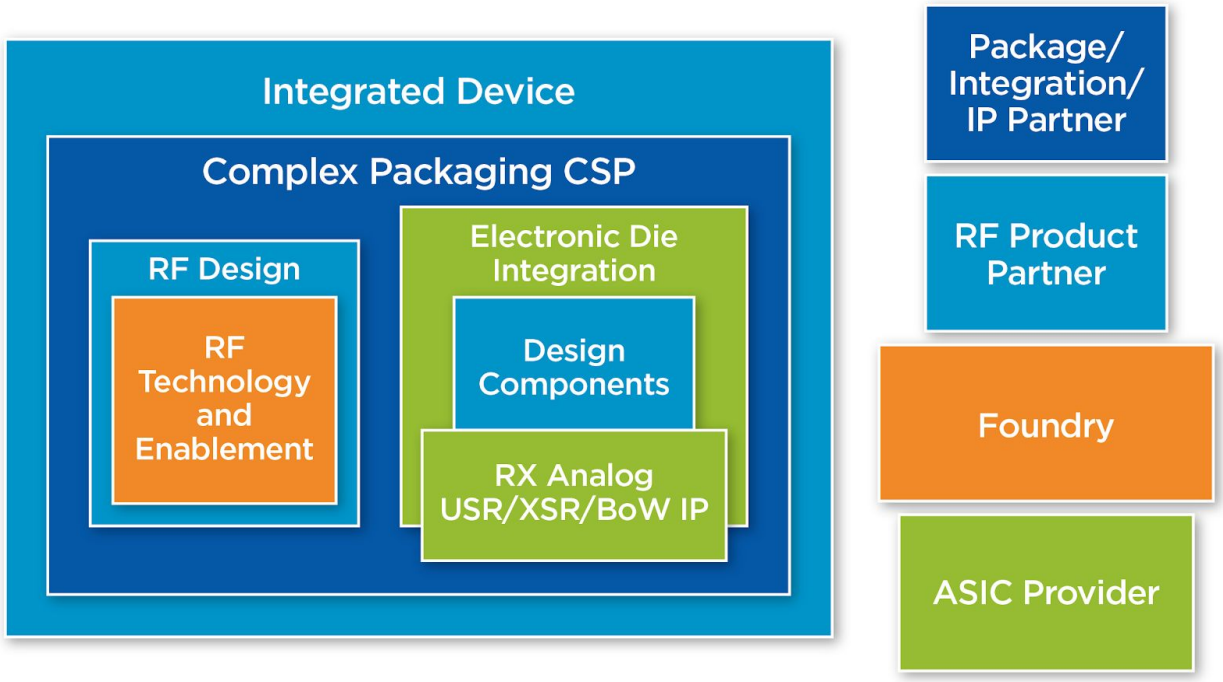



Figure 23: New RF component design flow with standard IP for inter-die interface

The next level up of assembly is shown in the following diagram where an ASIC provider engages with Outsourced Semiconductor Assembly and Test companies (OSATs) to design and assemble MCM packages using components consigned from the RF provider above along with an OEM customer ASIC function.

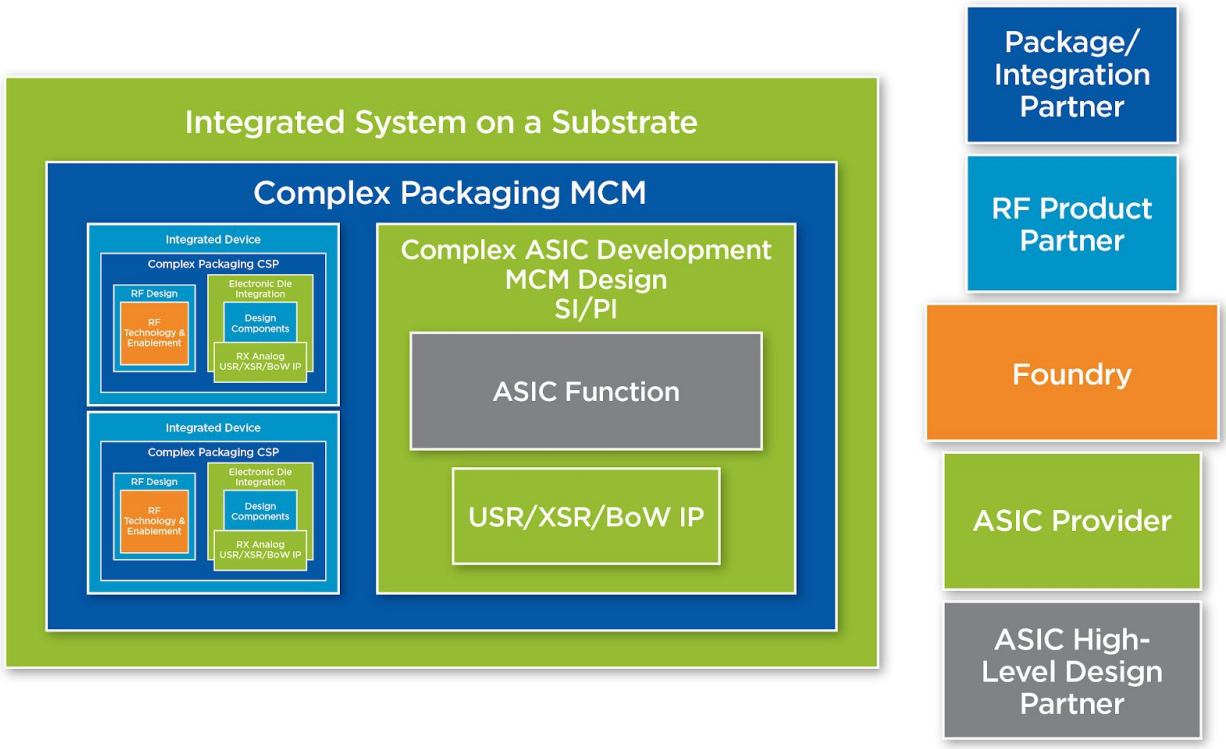



Figure 24: Chiplet-based design flow with OSATs

Of course these complex models need to have details established to enable a smooth solution to pricing procurement and test requirements, but many of those paths are becoming much more familiar as the industry supplies more and more complex components in the module.

A structural issue with multi-vendor die ecosystems is the practice of delivering KGD in wafer form. Traditional methods involve marking the die that failed test with a dot of ink (or making an entry in a database) and then delivering the marked wafer. The marked wafer is then diced and mounted into packages.

An issue with using this flow in a multi-vendor ecosystem is that it exposes the yield of each wafer to the customers of the dies. Chip yield is often a closely guarded fact for semiconductor manufacturers.

There are at least three solutions to this concern that can be used when enabling a multi-vendor ecosystem:

1. Remove the dies from the diced wafer and deliver them on a carrier
2. Use a trusted, contractually bound third party
3. Use pre-packaged, pre-tested chips provided in a chip-scale package

Each of these solutions addresses the concern of yield exposure. These solutions provide different ways to deliver known-good devices for integration.

## 8.2 Silicon Intellectual Property

Over the last decade Systems on Chip (SoCs) have become the predominant way that computing power is delivered in all but large-scale, data center servers. PCs, notebooks, mobile phones, network gateways and a wide variety of embedded systems have all adopted SoCs. SoCs are typically composed of a variety of pre-designed logic blocks called Silicon IPs.

Traditionally, licensors of silicon IP have offered their technology through a combination of upfront license fees and per unit royalties. They will provide verification tests and expect limits on the liabilities associated with errors in their designs. An IP vendor will seek to amortize the cost of IP development across multiple customers, achieving a lower price tag for a specific function than an SoC vendor could obtain if they developed the function themselves and were the only user of the function.

Interconnect IP has the unique requirement that the same IP must work on both sides of a link. System architects and procurement managers will insist that the IP has flexibility to be ported to different foundries, that porting is cost-effective, and ultimately that interconnect IP is available from multiple IP vendors or can be independently developed by chiplet providers. The business model will require that interconnect IP contributions are available and accessible across the chiplet ecosystem. This may be accomplished through existing standards groups like the OIF or IEEE, an existing industry consortium like the Open Compute Platform (OCP), or through a new consortium like the USR Alliance.

## 8.3 New Opportunities for Chiplet Integration?

Chiplet integration will require a different business model than that used for silicon IP. The reason for this is that chiplets, unlike silicon IP, will need to be processed, manufactured and quality guaranteed for years if not decades.

Large semiconductor companies will likely continue to be vertically integrated with the capability to design, build, assemble and test their own chiplet-based MCM solutions. Smaller companies that design specific task oriented chiplets will not likely have this capability and will instead rely on foundries or packaging houses to integrate and test in the MCM.

Chiplet companies will need to provide manufacturing guarantees to their customers regarding the lifetime of the chiplet based on what the foundry is willing to support. Alternatively, to better assure supply to an end customer, the chiplet provider may offer the transfer of manufacturing rights to the multi-chip module developer in exchange for royalty payments.

Initially, it might make sense for foundries and/or packaging houses to take responsibility for not only manufacturing chiplets on behalf of chiplet developers, but also accepting responsibility for the operational business of offering qualified chiplet die to MCM integrators and then providing

license or royalty fees back to chiplet vendors based on end-unit sales. The foundry or packaging house would build a catalog of chiplet devices that could be selected for integration into an MCM.

Over time, as the inventory of different chiplets expands, systems companies will want flexibility to mix and match chiplets from different foundries. It is easy to imagine that new companies could form that would architect new chiplet-based MCM solutions and leverage the best solutions from across the industry.

Large semiconductor vendors may adopt a model of developing chiplet technology for their own devices where they in-source 75% to 80% of the technology and only outsource specific technologies where it is non-strategic or does not make economic sense to develop themselves. Examples of technologies that would be candidates for outsourced chiplet development include: memory technology, SerDes technology, FPGA technology and DSA technology. Large semiconductor vendors may then establish their own captive ecosystem of packaging, interconnect technology and chiplet vendors.

Chiplet design will be similar to current SoC designs but the price tags will likely be smaller than those for fully integrated SoC devices. Because they are implemented in silicon, they will need to have test programs developed for known good die testing as well as testing within final packaged assemblies. Provisions will have to be made for respins and mechanisms will need to be established to track silicon revisions throughout the final product life cycle.

## 8.4 Open Accelerators and Chiplets Will Drive New Ways of Working

One approach is to think of MCMs as the new PCB where an ecosystem of interoperable components, interconnects, protocols and software is required to assemble, test and deploy reliable multi-chip solutions.

MCMs are not new, and much of the existing semiconductor ecosystem can be comfortably leveraged for silicon design, KGD testing, package design and assembly. Business models generally don't have to change. However, when a MCM contains many chiplets, additional attention and specifications will be required for things like:

- Efficient link, protocol and software solutions specifically for USR applications
- USR standards and interoperability agreements
- Certification programs for USR components to validate interoperability
- Ecosystem alignment on chiplet and assembled MCM testing procedures
- High-volume MCM assembly and test
- Product warranties
- Field failure analysis and root cause identification

The ODSA Workgroup jointly initiated by Netronome, Achronix, GlobalFoundries, Kandou, NXP, Sarcina and SiFive has been formed to develop an open architecture and related specifications for developing chiplets that promise to reduce silicon development and manufacturing costs. The ODSA Workgroup will also begin to work through the details of viable business models to realize the chiplet vision.

# 9 Conclusion

To compensate for the end of Moore's Law, DSAs are needed to handle the workloads in the data center and at the network edge. However, current approaches to developing custom monolithic ASICs for DSAs are not economically viable. Heterogeneous systems, in which integrated ASICs are composed of chiplets from multiple process nodes and/or multiple vendors are one option to reduce development costs. Current approaches to developing complete systems are closed and proprietary.

Recently surveyed advances in interconnect and packaging technology and data transfer protocols significantly improve heterogeneous systems. These advances were used to propose a new open architecture the ODSA for DSAs. Unlike current standardization approaches, the ODSA proposes standards for the full stack needed to implement a DSA, including data transfer protocols. The proposal is to build a prototype implementation of the ODSA. How business models will need to evolve to support chiplet-based manufacturing flows were also reviewed.

One of the primary advantages of the ODSA architecture is the ability to decouple PHY interfaces from the physical die used for other processing functions. The transaction layer used between chiplets is the key enabler for this integration. By leveraging the ODSA model, developers are free to choose the optimal solution for each chiplet based on performance needs, IP availability, and cost. Developers can rapidly assemble best-of-breed accelerators from chiplets that support the ODSA.

# 10 About the ODSA Workgroup

The mission of the ODSA Workgroup is to develop an open architecture and related specifications for developing chiplets that promise to reduce silicon development and manufacturing costs. The architecture being developed in the ODSA Workgroup enables the chiplet-based silicon design to be composed using best-of-breed components such as processors, accelerators, and memory and I/O peripherals using optimal process nodes.

The ODSA Workgroup, founded by Achronix, GLOBALFOUNDRIES, Kandou, Netronome, NXP, Sarcina and SiFive, is in the genesis of its formation and is open for contributions and accepting new members to build a robust ecosystem. Companies and industry partners wishing to learn more, participate and become an integral part of the ODSA Workgroup can inquire further at

odsa@netronome.com. In addition to the members, this paper includes contributions from Aquantia, ESnet and SamTec.

# 11 References


1. R. Swaminathan, “2D to 3d architectures: back to the future,” presented at the IMAPS Device Packaging, 2018, p. 33.
2. L. Li et al., “3D SiP with Organic Interposer for ASIC and Memory Integration,” in 2016 IEEE 66th Electronic Components and Technology Conference (ECTC), Las Vegas, NV, USA, 2016, pp. 1445–1450.
3. A. M. Caulfield et al., “A Cloud-Scale Acceleration Architecture,” 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)
4. A. Yee, “A Comparison of Low Cost Interposer Technologies,” NVIDIA, 2013.
5. Graphcore: www.graphcore.ai
6. B. Lindsey, M. Leslie, and W. Luk, “A Domain-Specific Language for accelerated Multilevel Monte Carlo simulations,” in 2016 IEEE 27th International Conference on Application-specific Systems, Architectures and Processors (ASAP), London, United Kingdom, 2016, pp. 99–106.
7. Hennessy, John L. and Patterson, David A., “A New Golden Age for Computer Architecture: Domain-Specific Hardware/Software Co-Design, Enhanced Security, Open Instruction Sets, and Agile Chip Development,” June 4, 2018.
8. D. He, Z. Wang, and J. Liu, “A Survey to Predict the Trend of AI-able Server Evolution in the Cloud,” IEEE Access, vol. 6, 2018, pp. 10591–10602.
9. J. Choi, R. Lian, Z. Li, A. Canis, and J. Anderson, “Accelerating Memcached on AWS Cloud FPGAs,” Proceedings of the 9th International Symposium on Highly-Efficient Accelerators and Reconfigurable Technologies.
10. Jason Cong, Zhenman Fang, Yuchen Hao, and Glenn Reinman, “Supporting Address Translation for Accelerator-Centric Architectures,” presented at the 23rd IEEE Symposium on High Performance Computer Architecture (HPCA).
11. R. Nimaiyar et al., “An Inference Engine, Network Compiler + Runtime for Xilinx FPGAs,” in HotChips, 2018, p. 20.
12. K. Hazelwood et al., “Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective,” in 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA), Vienna, 2018, pp. 620–629.
13. Moein Khazraee, Lu Zhang, Luis Vega, and Michael Bedford Taylor, “Moonwalk: NRE Optimization in ASIC Clouds or, accelerators will use old silicon,” presented at the ASPLOS, Xi’an, China, 2017.
14. Lapedus, Mark, “Big Trouble At 3nm,” Big Trouble At 3nm, Semi-engineering June 21, 2018.
15. S. Zhang et al., “Cambricon-X: An accelerator for sparse neural networks,” in 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), Taipei, Taiwan, 2016, pp. 1–12.
16. Brad Benton, “CCIX, Gen-Z, OpenCAPI: Overview and Comparison,” presented at the OpenFabrics Alliance 13th Workshop, 2017.

17. N. P. Jouppi et al., “In-Data Center Analysis of a Tensor Processing Unit,” Hot Chips, August 22, 2017, p. 38.
18. D. S. Green, “Common Heterogeneous Integration and Intellectual Property (IP) Reuse Strategies (CHIPS),” p. 32.
19. T. Vijayaraghavan et al., “Design and Analysis of an APU for Exascale Computing,” in 2017 IEEE International Symposium on High Performance Computer Architecture (HPCA), Austin, TX, 2017, pp. 85–96.
20. S. M. Lee, J. H. Jang, J. H. Oh, J. K. Kim, and S. E. Lee, “Design of hardware accelerator for Lempel-Ziv 4 (LZ4) compression,” IEICE Electronics Express, vol. 14, no. 11, pp. 20170399–20170399, 2017.
21. NVidia, “DGX-1 V100 System Architecture”
22. A. Farmahini-Farahani, J. Ho Ahn, K. Morrow, and N. Sung Kim, “DRAMA: An Architecture for Accelerated Processing Near Memory,” IEEE Computer Architecture Letters, vol. 14, no. 1, January 2015, pp. 26–29.
23. V. Sze, Y.-H. Chen, T.-J. Yang, and J. Emer, “Efficient Processing of Deep Neural Networks: A Tutorial and Survey,” arXiv:1703.09039 [cs], Mar. 2017.
24. M. M. Ozdal, “Emerging Accelerator Platforms for Data Centers,” IEEE Design & Test, vol. 35, no. 1, February 2018, pp. 47–54,.
25. S. Ghose, K. Hsieh, A. Boroumand, R. Ausavarungnirun, and O. Mutlu, “Enabling the Adoption of Processing-in-Memory: Challenges, Mechanisms, Future Research Directions,” arXiv:1802.00320 [cs], February 2018.
26. T. Ajayi et al., “Experiences Using the RISC-V Ecosystem to Design an Accelerator-Centric SoC in TSMC 16nm,” 2017, p. 6.
27. E. Nurvitadhi et al, “Accelerating Recurrent Neural Networks in Analytics Servers: Comparison of FPGA, CPU, GPU, and ASIC.”
28. P. Vivet, “FROM 3D TECHNOLOGY TO 2.5D AND 3D MANY-CORE ARCHITECTURES,” presented at the MCSoC, 2016, p. 50.
29. S. Ramalingam, “HBM package integration: Technology trends, challenges and applications,” 2016 IEEE Hot Chips 28 Symposium, 2016, pp. 1–17.
30. V. G. Cook et al, “Graph Streaming Processor,” HotChips, 2017.
31. S. Shumarayev, “Heterogeneous Modular Platform.” HotChips, 2017.
32. H. Braunisch, A. Aleksov, S. Lotz, and J. Swan, “High-speed performance of Silicon Bridge die-to-die interconnects,” in 2011 IEEE 20th Conference on Electrical Performance of Electronic Packaging and Systems, San Jose, CA, USA, 2011, pp. 95–98.
33. “CCIX Specification,” www.ccixconsortium.com
34. “TileLink Specification,” SiFive Corporation.
35. N. P. Jouppi et al., “In-Datacenter Performance Analysis of a Tensor Processing Unit,” in Proceedings of the 44th Annual International Symposium on Computer Architecture - ISCA ’17, Toronto, ON, Canada, 2017, pp. 1–12.
36. M. Vesper, D. Koch, K. Vipin, and S. A. Fahmy, “JetStream: An Open-Source High-Performance PCI Express 3 Streaming Library for FPGA-to-Host and FPGA-to-FPGA Communication,” p. 10.

37. A. Arunkumar et al., “MCM-GPU: Multi-Chip-Module GPUs for Continued Performance Scalability,” in Proceedings of the 44th Annual International Symposium on Computer Architecture, ISCA ’17, Toronto, ON, Canada, 2017, pp. 320–332.
38. K. Flamm, “Measuring Moore’s Law: Evidence from Price, Cost, and Quality Indexes,” National Bureau of Economic Research, Cambridge, MA, w24553, Apr. 2018.
39. Z. Wu et al., “Modeling, design and fabrication of ultra-thin and low CTE organic interposers at 40μm I/O pitch,” in 2015 IEEE 65th Electronic Components and Technology Conference (ECTC), San Diego, CA, 2015, pp. 301–307.
40. M. Schlansker, J. Tourrilhes, S. Banerjee, and P. Sharma, “Network Function Virtualization and Messaging for Non-Coherent Shared Memory Multiprocessors,” p. 24.
41. S. S. Banerjee, M. el-Hadedy, C. Y. Tan, Z. T. Kalbarczyk, S. Lumetta, and R. K. Iyer, “On accelerating pair-HMM computations in programmable hardware,” in 2017 27th International Conference on Field Programmable Logic and Applications (FPL), Ghent, Belgium, 2017, pp. 1–8.
42. S. Ventrone, “Optimizing selection of GLOBALFOUNDRIES ASIC and Foundry to achieve optimal IC Solutions,” 2018, p. 17.
43. O. Mutlu, “Processing Data Where It Makes Sense: Enabling In-Memory Computation,” p. 192.
44. T. Nowatzki, V. Gangadhar, K. Sankaralingam, and G. Wright, “Pushing the Limits of Accelerator Efficiency While Retaining Programmability,” p. 13.
45. L. Wu, A. Lottarini, T. K. Paine, M. A. Kim, and K. A. Ross, “Q100: the architecture and design of a database processing unit,” in Proceedings of the 19th international conference on Architectural support for programming languages and operating systems - ASPLOS ’14, Salt Lake City, Utah, USA, 2014, pp. 255–268.
46. H.-C. Ng, S. Liu, and W. Luk, “Reconfigurable acceleration of genetic sequence alignment: A survey of two decades of efforts,” in 2017 27th International Conference on Field Programmable Logic and Applications (FPL), Ghent, Belgium, 2017, pp. 1–8.
47. Jongse Park, Hardik Sharma, Divya Mahajan, Joon Kyung Kim, Preston Olds, and Hadi Esmaeilzadeh, “Scale-Out Acceleration for Machine Learning,” presented at the 50th Annual IEEE/ACM International Symposium on Microarchitecture, 2017.
48. S. Sen, S. Jain, S. Venkataramani, and A. Raghunathan, “SparCE: Sparsity aware General Purpose Core Extensions to Accelerate Deep Neural Networks,” arXiv:1711.06315 [cs], November 2017.
49. Z. Ruan, T. He, B. Li, P. Zhou, and J. Cong, “ST-Accel: A High-Level Programming Platform for Streaming Applications on FPGA,” p. 8.
50. T. Nowatzki, V. Gangadhar, N. Ardalani, and K. Sankaralingam, “Stream-Dataflow Acceleration,” in Proceedings of the 44th Annual International Symposium on Computer Architecture - ISCA ’17, Toronto, ON, Canada, 2017, pp. 416–429.
51. C. Ding et al., “Structured Weight Matrices-Based Hardware Accelerators in Deep Neural Networks: FPGAs and ASICs,” p. 6.
52. M. Abadi et al., “TensorFlow: A system for large-scale machine learning,” p. 21.
53. S. Xie and M. B. Taylor, “The BaseJump Manycore Accelerator Network,” arXiv:1808.00650 [cs], Aug. 2018.

54. M. Bedford Taylor, “The Evolution of Bitcoin Hardware,” Computer, vol. 50, no. 9, 2017, pp. 58–66.
55. K. Lim, D. Meisner, A. G. Saidi, P. Ranganathan, and T. F. Wenisch, “Thin servers with smart pipes: designing SoC accelerators for memcached,” p. 12.
56. T. Chen et al., “TVM: An Automated End-to-End Optimizing Compiler for Deep Learning,” arXiv:1802.04799 [cs], February 2018.
57. T. Moreau, T. Chen, Z. Jiang, L. Ceze, C. Guestrin, and A. Krishnamurthy, “VTA: An Open Hardware-Software Stack for Deep Learning,” arXiv:1807.04188 [cs, stat], July 2018.
58. L. Jiang, M. Kim, W. Wen, and D. Wang, “XNOR-POP: A processing-in-memory architecture for binary Convolutional Neural Networks in Wide-IO2 DRAMs,” in 2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), Taipei, Taiwan, 2017, pp. 1–6.
59. OIF CEI 4.0 Interoperability Agreement http://www.oiforum.com/wp-content/uploads/OIF-CEI-04.0.pdf
60. TE Connectivity XLA Socket https://www.te.com/commerce/DocumentDelivery/DDEController?Action=srchrtrv&DocNm=1-1773956-5_XLA_Socket_Brochure&DocType=DS&DocLang=EN
61. Netronome NFP-6000. https://www.netronome.com/static/app/img/products/silicon-solutions/PB_NFP6000.pdf
62. J. Noguera et al, “Xilinx Project Everest: HW/SW Programmable Engine”, Hotchips, 2018.
63. Bitcoin mining ASICs https://en.bitcoin.it/wiki/List_of_Bitcoin_mining_ASICs
64. K. Guo et. al, “A Survey of FPGA Based Neural Network Accelerator.”
65. T. Moorthy and S. Gopalakrishnan, “IO and data management for infrastructure as a service FPGA accelerators.”
66. R. Zhao et al, “The Celerity Open Source RISC-V Tiered Accelerator Fabric: Fast Architectural Design Methodologies for Fast Chips.”
67. C. Nicol, “A Dataflow Processing Chip for Training Deep Neural Networks,” HotChips, 2017.
68. S. Tang, “A List of Chip/IP for Deep Learning,” https://medium.com/@shan.tang.g/a-list-of-chip-ip-for-deep-learning-48d05f1759ae
69. P4 Consortium, www.p4.org
70. M. Fleming, “A thorough introduction to eBPF,” https://lwn.net/Articles/740157/.
71. B. Vinnakota, “P4: driving innovation in server-based networking,” https://www.datacenterdynamics.com/opinions/p4-driving-innovation-in-server-based-networking/
72. N. Viljoen and J. Kicinski, “Comprehensive BPF offload,” “https://www.netdevconf.org/2.2/slides/viljoen-xdpoffload-talk.pdf