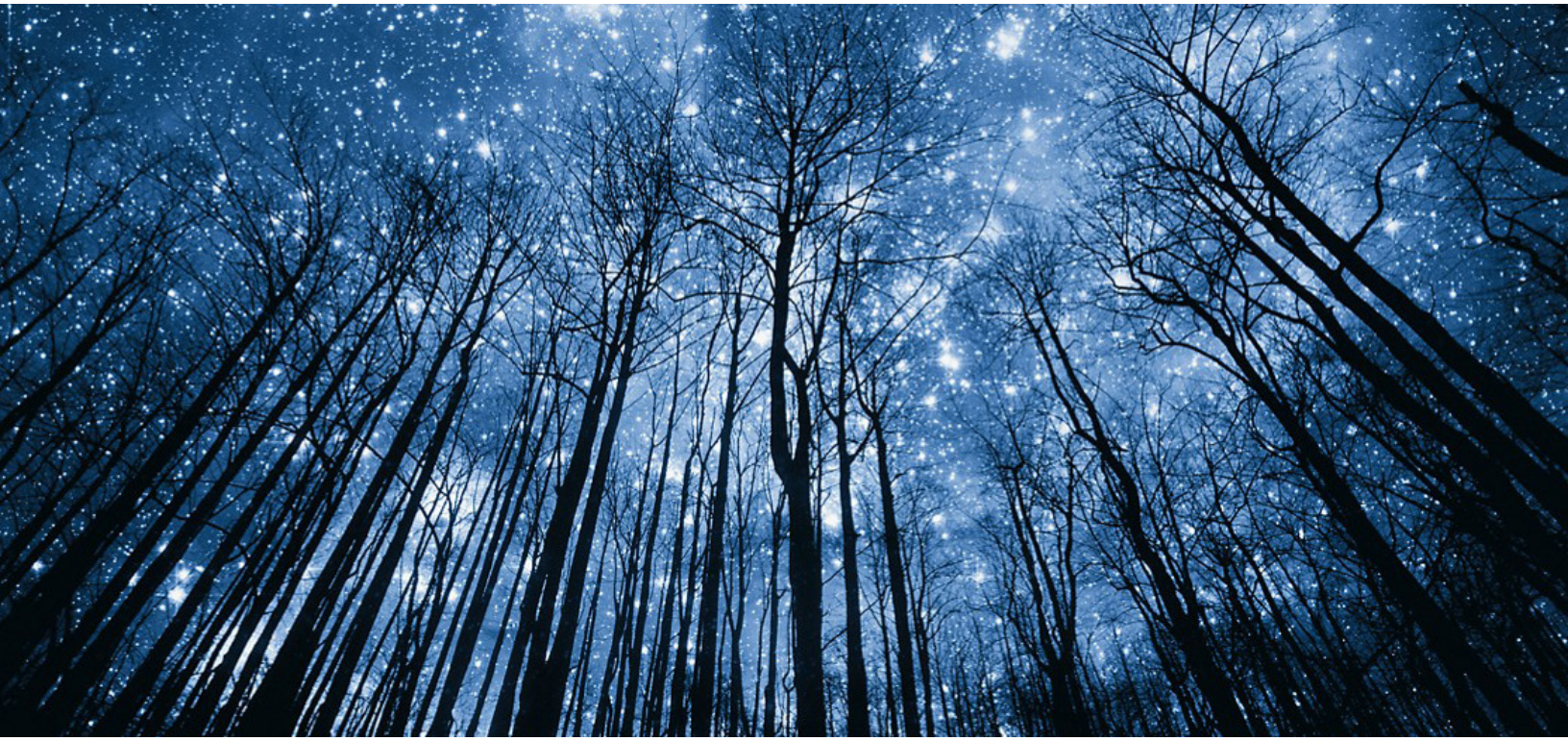
DELL Technologies

CXL – GAMECHANGER FOR THE DATA CENTER



Samriddhi Jaiswal
Customer eXperience Engineer
Dell Technologies

DELL Technologies
Proven Professional

Knowledge Sharing Article

Table of Contents

Abstract

The demand for data storage and processing power has skyrocketed as the world becomes increasingly digital. *We are adopting various use cases, such as gene sequencing, autonomous driving, and Fraud Detection for secure transactions where advanced workloads, like AI and ML models, are being utilized to analyze data and produce better output. For an organization to keep up with this and continue to innovate in these fields, it must have a robust infrastructure capable of handling the high-performance computing (HPC) requirements of these workloads. T*he industry is introducing a transformative change in data center architecture that is hoped to bring substantial advances in performance, efficiency, and cost. The traditional server architecture, which primarily focuses on computation, is giving way to prioritizing data and fully embracing it. This evolution is bound to change how we think about data centers and the technologies that power them, and it's a thrilling time to be a part of this transformation.

The data center is shifting from a model where each server has dedicated processing and memory—as well as networking devices and accelerators—to a disaggregated “pooling” paradigm that intelligently matches resources and workloads.

Compute Express Link (CXL) introduces a revolutionary architectural shift, bringing new possibilities to data center performance, efficiency, and cost. CXL is an open-source interconnect for memory to connect to processing in servers and storage. It also allows the pooling of memory made by multiple vendors’ products, and that can be connected directly to processors in CPU, GPU, and DPU form, as well as to smart NICs and computational storage. *C*XL 3.0 also enables shared memory to be distributed among different hosts in a system, allowing memory to become separated, like how storage is currently utilized.

This paper hopes to embed an understanding of CXL while also highlighting the following:

1. Knowledge of the current landscape and history
2. CXL protocols and iterations
3. Cases strongly highlight the driving benefits, such as a need for large memory capacities and heterogeneous computing, which are critical requirements for the performance of high-capacity workloads such as AI.

The vision to create component pooling could provide the resources needed for AI, HPC, and edge computing, and the CXL Consortium, with Dell EMC as a board member, understands this. With CPUs, GPUs, FPGAs, and network ports being pooled, entire data centers might behave like a single system, potentially bringing a dramatic paradigm shift.

Introduction

With the rise of AI and ML, we are working on better ways to manage the increasing amount of data. Businesses require applications that can handle the volume and velocity of data generated to stay competitive. According to Statista research, the total amount of data created, captured, copied, and consumed globally, was 64.2 zettabytes in 2020. Over the following years up to 2025, global data creation is projected to grow to more than 180 zettabytes (1). Futuristic solutions such as edge computing, heterogeneous computing, distributed storage, and data lake architectures require utilizing resources that can keep up with this pace of growth without causing any bottlenecks.

What are the current challenges in computing?

The rapid pace of technological advancement has led to a growing demand for fast, efficient data centers. However, as the industry approaches the physical limitations of Moore's Law (2), the challenges of maintaining this exponential growth have become increasingly apparent. Systems optimized for generic tasks seemingly fail as newer workloads require more computational power, memory, and specialized hardware accelerators. As a result, traditional computing systems need to be improved.

With the expansion of big data and the corresponding increase in processing demands, traditional data center architecture has strived to mitigate the strain on processing power by scaling up CPU cores. This results in a mismatch between the available memory bandwidth and the processing demands of the added cores. Referred to as a "Memory Wall" (3), it creates a bottleneck where the processor cannot access data from memory fast enough to keep up with its execution rate, leading to a reduction in overall system performance.

Is heterogeneous computing the solution?

Heterogeneous computing (HC) uses different types of processors or cores in a system to perform various tasks. This can include using different types of central processing units (CPUs), graphics processing units (GPUs), digital signal processors (DSPs), field-programmable gate arrays (FPGAs), or other specialized processing units.

It is becoming increasingly common in many fields, particularly high-performance computing, data centers, and artificial intelligence (AI) applications.

1. In high-performance computing, heterogeneous computing performs complex simulations, modeling, and data analysis tasks. For example, a supercomputer might use a combination of CPUs, GPUs, and other specialized processing units to perform complex simulations of weather patterns, protein folding, or large-scale data analysis.
2. In data centers, heterogeneous computing is used to improve the performance and efficiency of the servers and the data center. For example, a data center might use a combination of CPUs and GPUs to perform machine learning, data analytics, and video encoding.
3. In AI applications, heterogeneous computing performs tasks such as image and speech recognition, natural language processing, and autonomous systems. For example, a self-driving car might use a combination of a CPU, a GPU, and specialized digital signal processors (DSPs) to perform tasks such as image recognition, sensor processing, and control systems.

A simple way to think about this is to compare it to a toolbox, where each tool is specialized for a

specific task, but together they can help complete a wide range of projects. Similarly, in HC, different types of processors and accelerators are combined to complete various computing tasks, resulting in a more versatile and powerful system. This analogy should help you further understand how it can be used. By using different types of cores, HC can help to balance the load across different resources.

Although using multiple processors as a homogenous system can help increase your bandwidth and reduce the risk of running into the "Memory Wall" bottleneck, it also adds significant complexity due to the current architecture's inability to decouple memory from its respective CPU or accelerator. One way this complexity can manifest is that different types of cores in the same system can make it harder to optimize the memory hierarchy and communication between cores, which can introduce additional latency and overhead.

As the industry progresses toward next-gen data center architecture, there is an increasing need for a memory subsystem that emulates the flexibility and scalability of storage. Allowing memory to be decoupled from the CPU and seamlessly integrated into a software-defined, composable infrastructure can enable data center managers to allocate dynamically and provision memory resources in alignment with the evolving needs of their workloads.

The Compute Express Link (CXL) interface is an example of the innovation the industry needs to realize next-generation data centers and data management. CXL enables faster and more efficient data transfer between the processor and peripheral devices, memory mapped I/O, memory coherency and consistency, and power management. It also allows a diverse mix of architectures deployed in CPU, GPU, FPGA, smart NICs, and other accelerators to work together to process data.

What is CXL?

The CXL Consortium defines CXL succinctly as an “industry-supported cache-coherent interconnect for processors, memory expansions, and accelerators.” (4) Before we get into more details, understanding cache coherent interconnect is crucial.

What is Cache Coherent?

Let’s start with the basics. Cache allows the processor to access the data more quickly, as it doesn't have to wait for the main memory, also known as RAM (Random Access Memory). In HC, multiple copies of shared data can exist in a multiprocessor system that uses each processor's shared and individual cache memory. One copy is stored in the main memory, while the other copies are stored in the cache memory of each processor that requested it. When one of the copies of the data is modified, the other copies must be updated to reflect the change. If this is not done, the wrong data will be stored and read, leading to invalid results, and crashing the program.

A cache coherent interconnect is a technology that ensures that the data stored in multiple local memory cache locations of different processors or cores in a multi-core system is **consistent and up to date** without explicit communication between the processors.

One way to maintain coherency is for the CPU to constantly "snoop" the PCIe interconnect, which means it continually monitors the interconnect traffic to check if any other device has updated a

piece of data that is also cached in the CPU's local memory. (5) This can be a resource-intensive task and negatively impact performance.

To reduce the burden from the CPU, it is beneficial to place resource-intensive tasks on a separate interconnect, which is responsible for maintaining cache coherency. When a processor modifies a block of data in its local cache, the interconnect propagates this change to the main memory and all caches in the system.

CXL leverages an optimized command and feature set to facilitate high-speed, efficient data transfer between devices utilizing parallel command queues. Utilizing a low-latency, high-bandwidth interconnect, CXL enables rapid data transfer speeds. Additionally, it incorporates advanced capabilities such as prioritization of commands, error mitigation, and power management to enhance system reliability and efficiency.

The question now becomes are we eliminating PCIe moving forward? Not at all!

CXL is not replacing PCI Express (PCIe) but instead is building upon it and providing additional features and capabilities. **CXL uses the same physical layer as PCIe and is fully backward compatible with it, meaning that CXL devices can work with existing PCIe infrastructure** (6). CXL also has its software stack, which enables memory mapped I/O, memory coherency, and consistency. It utilizes the high-speed data transfer capabilities of the PCIe Gen6 interface, which allows for a single controller to have a 64GT/s x16 link. Additionally, CXL allows splitting this link into multiple smaller links, such as x8 and x4, to increase the overall bandwidth available to the system.

The ability to customize is crucial in this scenario and is a vital takeaway, as it allows customers to choose how they want to use the system. They can use a PCIe slot to plug in either a PCIe device or a CXL device. This results in cost savings for the business as it allows for tailored decisions for its systems.

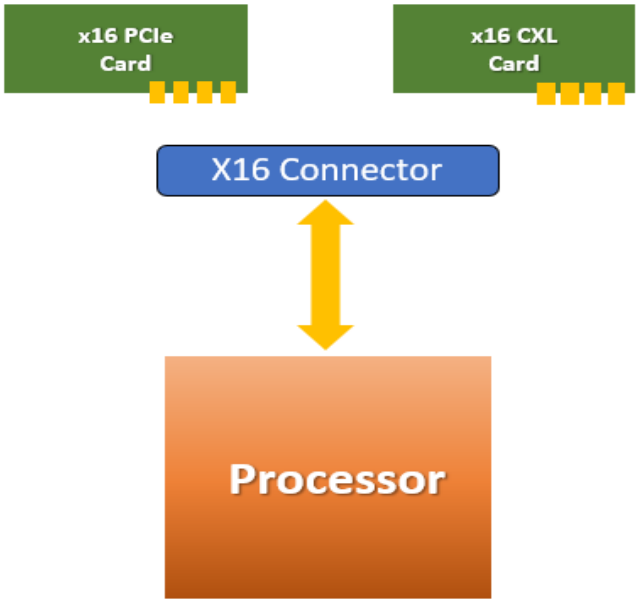


*Fig 1- CXL uses a flexible processor Port that can **auto-negotiate** to either the standard PCIe transaction protocol or the alternate CXL transaction protocols.*

Keeping this in mind, since PCIe is a fundamental part of CXL, it will inherently improve with it as it improves and comes up with newer specifications.

CXL protocols

CXL defines protocols that enable communication between a host processor and attached devices. It defines three main protocols: CXL.io, CXL.cache, and CXL.memory. CXL uses a single link to transmit data using three different protocols simultaneously (called multiplexing). Each of these protocols serves a specific purpose and provides different capabilities to enable the coherent sharing of memory resources between computing devices.

1. **CXL.io**: This is required for initialization device discovery, configuration, direct memory access for data movement, etc.
2. **CXL.cache**: This is designed for more specific applications, enabling accelerators to access and cache host memory for improved performance efficiently. It defines interactions between a Host and a device and enables memory mapped I/O, which allows devices to access host memory as if it were local to the device. This protocol can improve performance by reducing the need for data to be copied between the host and device memory.
3. **CXL.memory**: This protocol enables memory coherency and consistency. It is designed to allow the host, such as a processor, to access device-attached memory using load/store commands. Think read date (i.e., load) from memory or write data (i.e., store) to memory. The host CPU acts as a requester, requesting access to the memory, while the specialized device acts as a subordinate, providing access to the memory. This memory expansion allows the host to access more than what is physically possible with traditional systems.

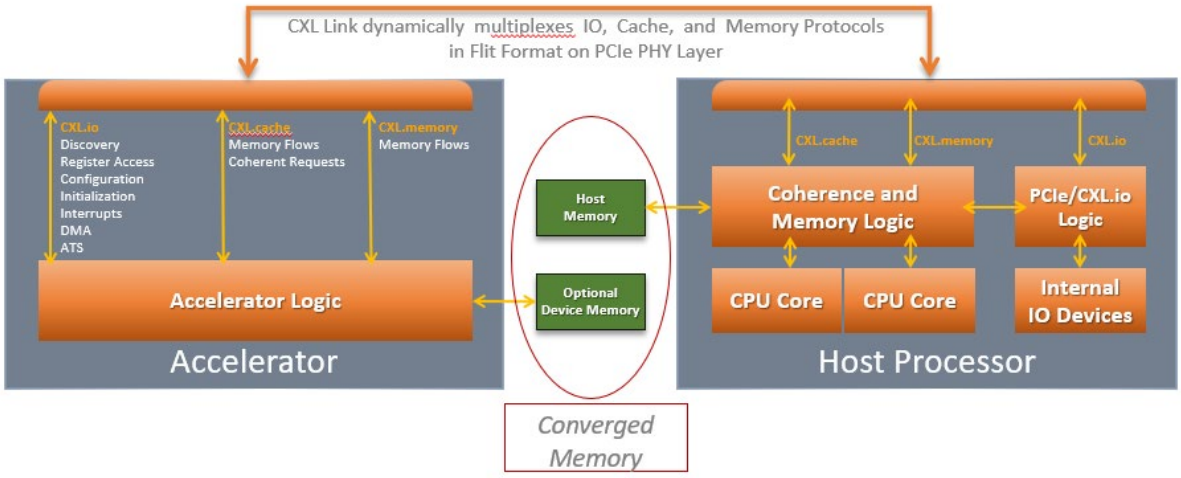


Fig 2 - CXL transaction layer comprises three dynamically multiplexed protocols on a single Link

All these protocols (7) work together to enable the coherent sharing of memory resources between computing devices, e.g., a CPU host and an AI accelerator. These simplify programming by allowing communication through shared memory. Furthermore, CXL also provides security features as all

three protocols are secured via Integrity and Data Encryption (IDE) which is essential in today's data centers.

These attributes make CXL an ideal solution for high-performance computing and data center environments where low latency, high-bandwidth interconnectivity, and memory sharing are paramount.

Unveiling the various CXL specifications -

The CXL Consortium has introduced three specifications over the past few years.

CXL 1.0 and CXL 1.1:

CXL 1.0 uses multiple protocols built on top of the PCIe 6.0 standard, allowing for coherent memory across host processors and other devices. This means that devices with CXL 1.0 can take advantage of the high speeds and performance of PCIe 6.0 while also ensuring that memory is consistent across the system. CXL 1.1 added compliance testing for reliability, availability, and serviceability (RAS) and guaranteed backward compatibility with CXL 1.0.

CXL aims to create fully composable computing systems and memory pooling in data centers. However, not all data center components can be replaced at once. CXL 1.1 focuses on accelerators and memory expansion use cases, laying the foundation for future developments.

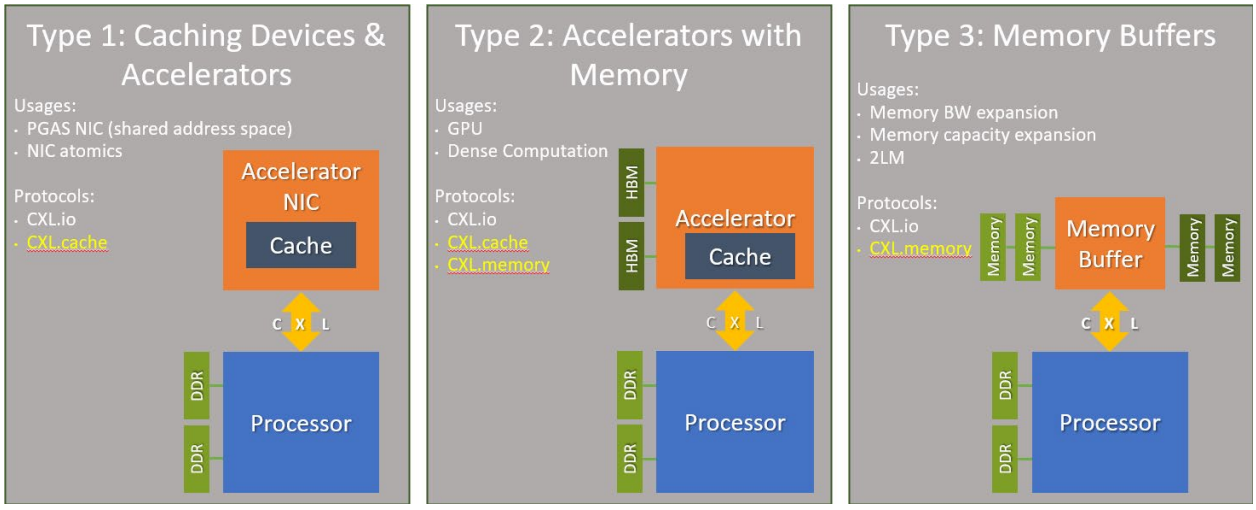


Figure 3- Use cases enabled through CXL 1.0/1.1

In the above diagram (8), CXL 1.0 and 1.1 highlights a direct connect between a host and a device, and it targeted three types of usage models. The left illustrates a Type -1 device that needs caching semantics for usages like smart NICs to deliver better performance. One is PGAS (Partitioned Global Address Space) NIC, designed to provide high-performance communication between nodes in a parallel computing environment (such as HPC). CXL provides low-latency communication between nodes and enables them to access and manipulate the shared memory space, particularly useful for workloads such as machine learning and scientific simulations.

Moving on to Type -2 devices (middle illustration), examples include GP GPU and FGPA dense computation with a local memory attached. They would be expected to implement all three protocols. In this scenario, GP GPUs require high-bandwidth and low latency between the GPU and

other devices, such as storage and networking devices (CXL.io). They need to share cache memory with other devices, such as the CPU or other accelerators (CXL.cache), helping improve the performance of workloads that require tremendous data to be transferred between devices. Lastly, CXL enables the GPU to share main memory with other devices (CXL.memory) for workloads that require a lot of memory, such as big data processing and high-performance computing.

For Type -3 devices (extreme right of illustration), the usage would be scenarios where Memory Bandwidth expansion and Tiered memory (including storage class memory) are necessary. Type -3 devices would need to implement CXL.io and CXL.memory semantics. The memory will be mapped to system memory as cacheable memory. The host processor orchestrates the cache coherency and relieves the devices from having to implement complex coherency flows. Think of these as your memory modules. They provide memory capacity that is persistent, volatile, or a combination.

CXL 2.0

CXL 2.0 introduced switching and memory pooling and improved RAS within the standard. These are some interesting use cases.

- Switching allows multiple host CPUs to share and allocate devices depending on the workload, creating fully composable data centers.
- This new specification enabled memory pooling where host CPUs are connected to high-capacity memory devices and only allocate the necessary memory per workload, preventing over-provisioning. This applies to data accelerators, SmartNICs, and other composable devices.
- It also adds link-level integrity and data encryption, ensuring that traffic on the CXL link is secure.

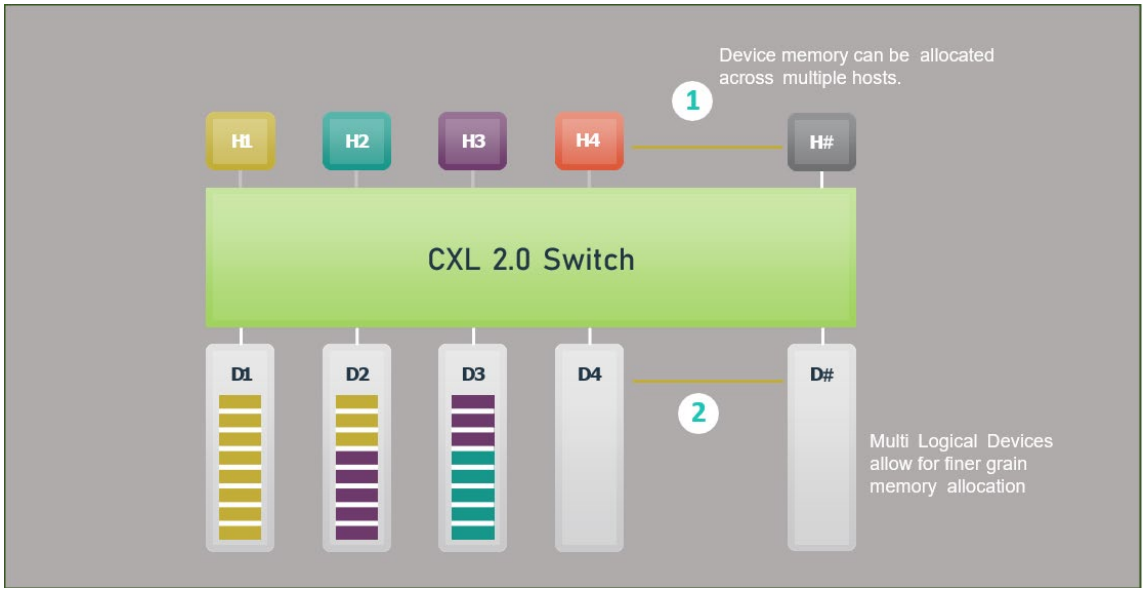


Fig 4 – CXL 2.0 Memory Pooling Feature

CXL 3.0

CXL 3.0, released in August 2022, is based on PCIe 6.0, and doubles the transfer rate to 64GT/s without increasing latency. It includes multi-level switching, allowing systems to level CXL 3.0 switches in either a cascading or fanout design. It also provides for dynamic capacity devices so that a switch can consist of different devices with different protocol types.

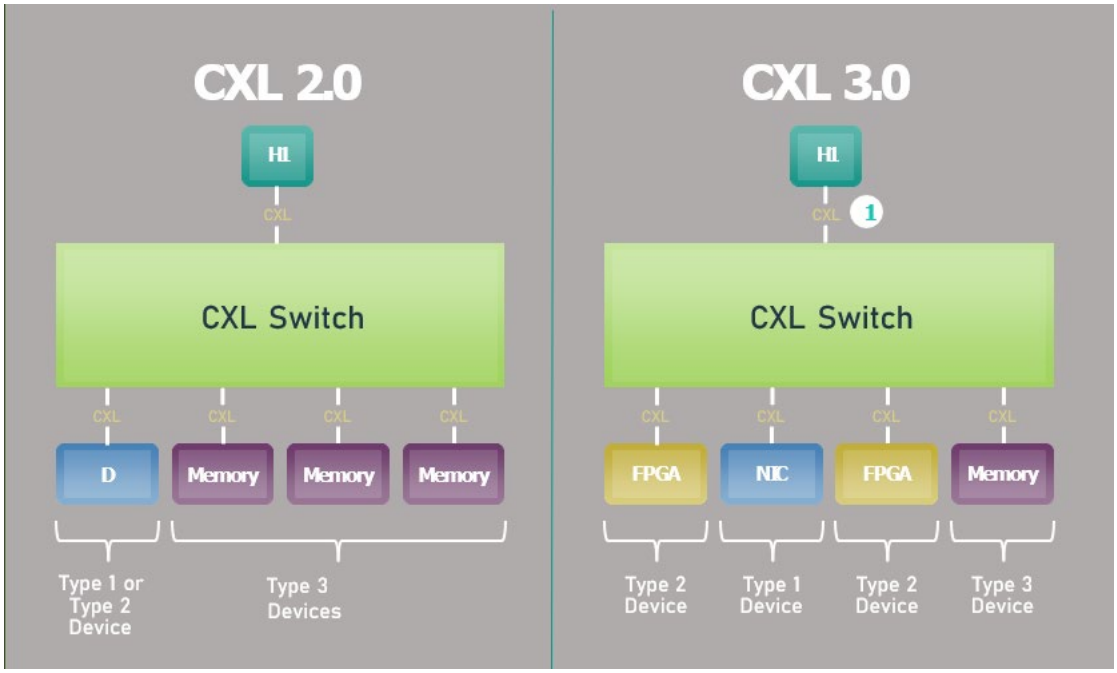


Fig 5- Each host's root port can connect to more than one device type

One of the critical features of CXL 3.0 is memory sharing, which builds upon CXL 2.0's memory pooling feature. Memory sharing allows more than one host to access a common section of memory simultaneously, enabling multiple machines to store, compute, and move data coherently.

Another critical feature of CXL 3.0 is the introduction of CXL fabric. This allows devices in the data center to connect and communicate in a way that is more flexible and efficient than previous methods. The CXL fabric uses a port-based routing technology, allowing up to 4,096 different devices to connect. These devices include CPU Hosts, GPUs, and memory devices. It also allows devices to communicate directly with each other without needing to go through a central "shared host." This dramatically improves performance and reduces latency.

This current specification is also compatible with previous versions of the CXL standard, which means that it can be used in existing data centers without requiring a complete overhaul.

The CXL standard is designed to make it easier for data centers to manage and process large amounts of data by making memory and storage more flexible and efficient. Utilizing memory sharing and Global Fabric Attached Memory (GFAM) devices, **the CXL protocol allows for the decoupling of memory from traditional host CPU dependencies, enabling data mobility and scalability previously unattainable.**

Memory is no longer a constraint.

As next-generation heterogeneous computing and composable data centers have the potential to revolutionize how we manage large amounts of data, CXL also can disrupt the traditional memory and storage hierarchy.

For the past several decades, the way data centers store and retrieve data has remained relatively unchanged. Typically, data is stored in storage devices with a higher capacity, but it also comes with increasing latency the further it is from the processor. On the other hand, as we move up the memory and storage hierarchy, memory has lower latency but also higher bandwidth. Additionally, there is persistent memory and various other options in between.

However, an in-depth understanding of CXL-enabled memory pooling for HPC systems and workloads still needs to be included.

When a workload uses compute and memory resources in a way that is different from the fixed resource configuration, it can result in those resources being underutilized. One solution to this problem is to disaggregate memory and compute resources. This means that the compute resources are not tightly tied to a specific amount of memory but instead have access to a pool of memory resources that can be allocated as needed (9). This allows for better resource utilization and the memory resources to be upgraded or maintained independently of the compute resources, potentially reducing the overall TCO.

With CXL, we now have the option for a new type of memory called far memory. This type of memory is slower and has higher latency but has higher capacity than traditional near memory. However, it also has faster speeds and higher bandwidth than traditional storage options. Furthermore, it is more cost-effective thanks to its memory pooling capabilities. *The latency of far memory is insignificant when considering its capacity benefits.* As an alternative to far memory, one could stack near memory to increase density, increasing the cost per bit. Far memory offers a balance of performance and capacity for many workloads, especially for data processing and AI/ML workloads where hot data components are accessed more efficiently.

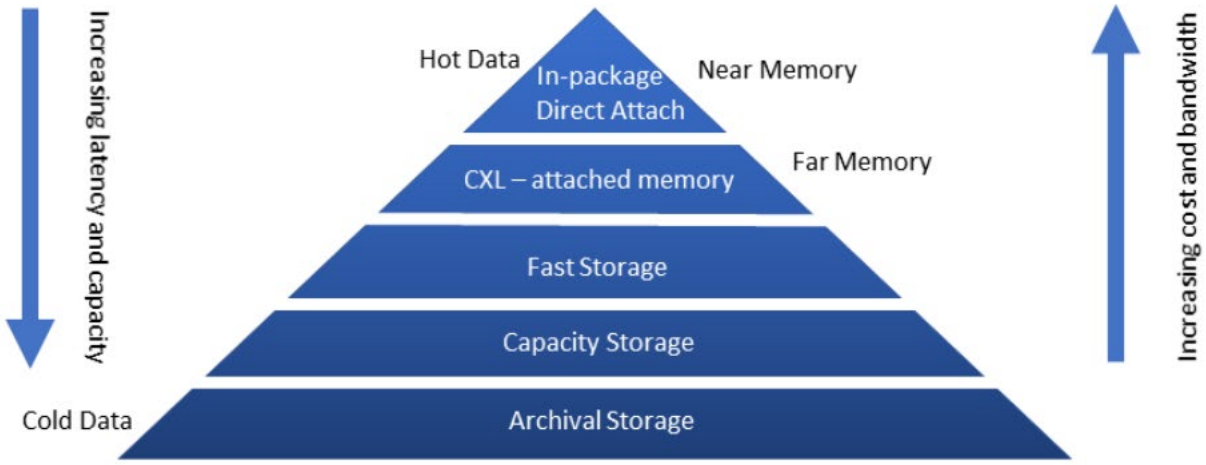


Fig 6.- Memory and storage hierarchy (10)

It enables greater flexibility in platform architecture, allowing for a CXL-enabled system with a scale-out architecture to have the memory benefits of a scale-up architecture. Scale-up architecture typically refers to larger systems with high memory capacity and performance. Historically, data centers with scale-out architecture use many smaller machines and either spread memory across multiple devices with high-latency interfaces or break up the application into smaller parts. With CXL, devices can be composed to build optimized infrastructure while maintaining the scalability and flexibility of a scale-out architecture.

By making memory and cache coherent and consistent across the host CPU and heterogeneous devices, CXL frees up resources within the system. This flexibility opens new possibilities for data centers, depending on the specific workload and application. It combines the benefits of both scale-up and scale-out architectures. CXL also has the advantage of higher bandwidth, as it is built on PCIe and offers speeds of up to 64GT/s without additional latency, which is particularly beneficial for hyperscale's.

How does CXL enable heterogeneous computing?

Heterogeneous computing utilizes multiple types of processors or cores in a system to improve performance or energy efficiency. These processors have specialized capabilities to manage specific tasks, and this approach breaks away from traditional processor design, offering new challenges and opportunities in high-performance computing. This method will continue to add more cores and specialized hardware features such as transactional memory, random number generators, and improved vector extensions. It will improve efficiency and performance in big data, mobile, and graphics markets.

Combining traditional processors with unconventional cores, such as custom logic, field-programmable gate arrays (FPGAs), or general-purpose graphics processing units (GPUs), can achieve greater energy efficiency. Instead of relying on a single CPU or GPU, heterogeneous architectures include an application-specific integrated circuit (ASIC) or FPGA to perform highly specialized processing tasks.

The main advantage of heterogeneous computing is that it achieves performance gain through parallelism rather than the clock frequency.

CXL enables the shift towards heterogeneous computing by providing an industry-standard protocol widely supported by major players in the industry. This allows for a common interconnect connecting different types of chips, making it possible to seamlessly integrate a diverse range of components within a single system.

As each new specification built on the features of the previous one, it provided new insights and domains of usage. Primarily, it allows devices to communicate in a peer-to-peer fashion. In CXL 3.0, as previously mentioned, devices could not directly access each other's memory without going through a host. Doing this helps reduce latency and means devices are not saturating host-to-switch bandwidth with these requests.

So Multi-device connection to the processor and communication with each other using the CXL protocols enable the devices to share data, resources, and work together to perform different tasks. **This fulfills the primary idea of HC while also maintaining the accuracy of cache coherency**.

For example, CXL.io devices provide I/O functionality, such as storage and networking, during CXL.memory devices can provide main memory, and CXL.cache devices can accelerate the system's performance as cache memory and accelerators. **Additionally, CXL allows the use of memory mapped I/O, which enables peripheral devices to be accessed using the same memory addressing used for main memory.** This makes the system more flexible and allows the operating system to manage and access the devices connected to the bus in a unified way, regardless of their type.

CXL improves performance and efficiency by eliminating duplicate data and optimizing memory resources, and it provides cache coherence that ensures shared data remains consistent across multiple local memory locations. This eliminates the need for software consistency and reduces the complexity of programming. This provides a way for different accelerators and devices to communicate and share resources more efficiently and effectively. Unlike other proprietary technologies such as NVLink and OpenCAPI, (11) CXL is an open standard, which means it is not tied to any vendor or technology. This makes it more versatile and flexible, allowing a wide range of devices and accelerators from different vendors to work together seamlessly.

Memory Pooling for Data Centers

With CXL, memory can be disaggregated, meaning it can be separated from the CPU and attached to a separate device. This allows for greater flexibility and scalability in data centers by allowing multiple servers to access the same pool of memory.

For example, in a data center with multiple servers, each server may have its local memory (DRAM), but it could run out of memory capacity to manage a specific task. With CXL memory pooling, the server can access memory from the pool of memory, which is shared among the servers. This allows the server to access more memory than is physically possible with traditional systems. When the task is completed, the memory can be released back to the pool for other servers. This allows for more efficient use of memory resources and can reduce costs.

Another example would be a data center where many virtual machines are running, and the memory resources are oversubscribed. CXL memory pooling allows the data center administrator to provision the memory resources to the virtual machines that need it when they need it. This ensures that the virtual machines have the memory resources they need to run effectively and allows the data center administrator to optimize the memory resource usage.

CXL memory pooling also enables memory expansion and greater memory bandwidth than traditional systems. It allows for a large pool of memory to be shared among multiple servers, and it is designed to be secure, with all three protocols (CXL.io, CXL.cache, and CXL.memory) secured via Integrity and Data Encryption (IDE). This allows for confidentiality, integrity, and replay protection of the shared memory pool.

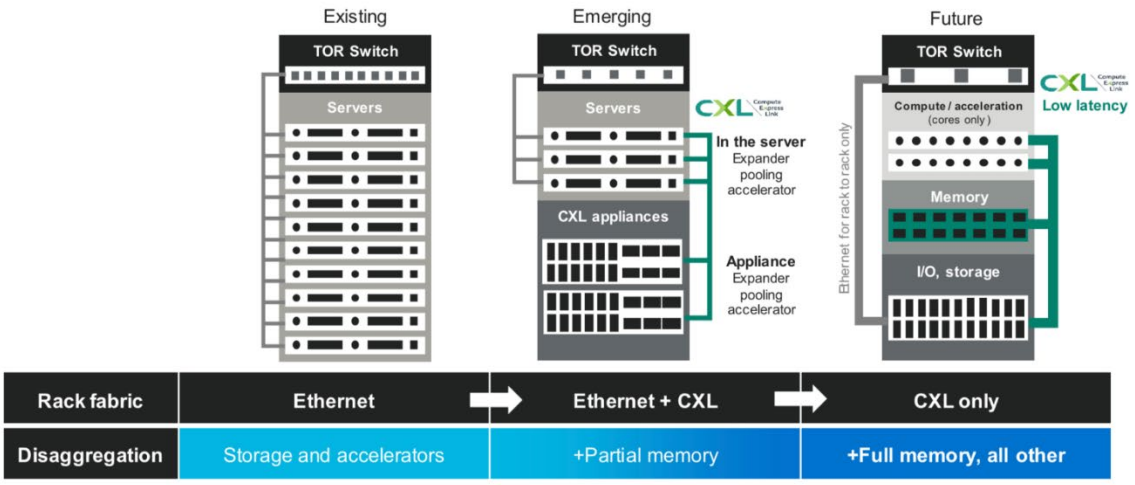


Fig 7 – Proposed view of how disaggregated memory will look in the coming years.

Overall, CXL memory pooling allows for greater flexibility and scalability in data centers by allowing multiple servers or devices to share memory resources, enables memory expansion and greater memory bandwidth, and is designed to be secure. It allows for more efficient use of memory resources and can reduce costs, making it more cost-effective for data centers.

Revolutionizing our Data Centers

CXL is revolutionizing data centers by enabling memory pooling and composability, which leads to increased performance, scalability, and flexibility. Let us quickly recap its features in the context of data centers –

- CXL allows for direct data transfer between accelerators, eliminating the need for host memory as an intermediary step. This allows for faster, more efficient communication between different compute units.
- Data centers can now better utilize the resources available to them by dynamically allocating memory and computing resources based on specific workloads' needs.
    - The specific needs that CXL addresses in data centers include the ability to allocate and manage memory resources dynamically, support different types of memory and memory hierarchies, and provide cache coherency across different types of devices.
    - Applications that can benefit from these features include data processing, big data analytics, artificial intelligence and machine learning, scientific simulations, and other high-performance computing workloads.
- CXL also enables composability in data centers, allowing for constructing memory subsystems that meet the specific needs of different applications.
    - For example, an application that requires large amounts of memory with fast access times, such as an artificial intelligence or machine learning workload, would benefit from a memory subsystem that includes high-speed memory, such as HBM or DDR4.
    - On the other hand, an application that requires large amounts of storage with lower access times, such as a significant data analytics workload, would benefit from a memory subsystem that includes high-capacity memory, such as NAND flash or Intel Optane.

    - This support for memory pooling allows for more cost-effective solutions by enabling the sharing of memory resources between different jobs or ranks, increasing overall utilization.
- CXL's support for security features like memory encryption and access control allows for better protection of sensitive data and improved security in data centers.
- Additionally, its open standards make it an attractive option for data centers, as it allows for integrating different vendors' hardware and accelerators, improving flexibility and scalability.

Conclusion

CXL technology is set to disrupt the data center industry, fundamentally changing how we think about and utilize memory resources. Its ability to enable memory coherency, on-demand resource allocation, and various other benefits for diverse workloads and applications make it a **game-changer for how data centers operate**. The high-bandwidth, low-latency interconnect provided by CXL allows different types of devices to be connected to the processor and communicate with each other, opening new possibilities for data processing and AI/ML workloads.

However, it is important to note that the full potential of CXL is yet to emerge, and its adoption heavily depends on the industry's continued investment in developing and standardizing the technology. The CXL consortium, which includes major players like Dell EMC, is investing in developing and standardizing this technology.

As CSPs and large enterprises strategize the next generation of data centers, they should consider the potential benefits of CXL and weigh the risks and investments involved in its adoption.

But make no mistake, the future of data centers is on the brink of a significant shift, and CXL technology is leading the charge. The possibilities are endless, and the potential for increased efficiency and cost savings is undeniable.

The data center industry is on the cusp of a transformation, and CXL is at the forefront of this change. The future is exciting and adopting CXL is a step toward a new era of on-demand computing.

Fig 8 – View of members of the CXL Consortium.

References

1. **CSM.** https://www.csm.tech/glemerging-technologies/offering/artificial-intelligence-machine-learning/.
2. **Forbes.** The True Nature Of Moore's Law – Driving Innovation For The Next 50 Years.
3. **AGM Sigarch.** Memory-centric Computing Systems: What's Old Is New Again.
4. **CXL Consortium.** *Compute Express Link™: The Breakthrough CPU-to-Device Interconnect.*
5. **IBM.** *Cache Coherency - https://www.ibm.com/docs/en/aix/7.2?topic=architecture-cache-coherency.*
6. *CXL 2.0 - https://docs.wixstatic.com/ugd/0c1418_d9878707bbb7427786b70c3c91d5fbd1.pdf.* **Sharma, Dr. Debendra Das.**
7. ***Compute Express Link (CXL) – Everything You Ought To Know -https://www.logic-fruit.com/blog/cxl/compute-express-link-cxl/.***
8. ***Introduction to Compute Express Link (CXL): The CPU-To-Device Interconnect Breakthrough (https://www.computeexpresslink.org/post/introduction-to-compute-express-link-cxl-the-cpu-to-device-interconnect-breakthrough).*** **CXL Consortium.**
9. ***Design and Analysis of CXL Performance Models for Tightly-Coupled Heterogeneous Computing.*** **Anthony M Cabrera, Aaron R Young, and Jeffrey S Vetter.**
10. ***CXL: ENABLING A HETEROGENEOUS, COMPOSABLE, NEXT-GENERATION DATA CENTER.*** **Moor Insights and Strategy.**
11. ***Evaluating Emerging CXL-enabled Memory.*** **Jacob Wahlgren, Maya Gokhale, Ivy B. Peng.**