AMD PENSANDO

WHITE PAPER | PERFORMANCE REPORT

ARCHITECTURE MATTERS

Performance of a Multi-Stage SDN Pipeline on Arm® vs AMD Pensando™ Programmable Silicon

Abstract

Public cloud providers implement complex network functions in every server to provide tenant isolation, security, and metering. To offload those services from the server CPU, data processing units (DPUs) have emerged as a third processor being deployed in servers to handle these network functions and other services. DPUs can have architectures based on FPGAs, ASICs, general-purpose cores (e.g., Arm®), or some combination of those components. This paper shows that an AMD Pensando™ DPU, using a domain-specific architecture with programmable silicon, can provide higher performance and scale compared to DPU architectures that rely on general-purpose cores to perform much of the work.

Background

CPU performance increases have long ago made running a single workload on a physical server very inefficient. Virtualization solved the efficiency problem by providing the ability to combine many servers and applications onto fewer pieces of physical hardware, but this advancement in technology created its own set of challenges.

The primary challenge of placing multiple users and applications on a single server is isolation and security: keeping different applications and their users separated from one another while still providing each environment with adequate resources. There also needed to be a clear separation of access and duties; that is, the infrastructure administrators should not have access to the application and customer data and each set of administrators should be limited to making changes in their own operational area. There also needed to be a set of tools for monitoring and troubleshooting.

This required a shift in architecture, and the way network services have been deployed had to change. The resulting change was the distribution of services like firewall, load balancing, microsegmentation, networking services, and encryption across the infrastructure. Instead of being centralized within the data center, the new model distributes them on the server edge, closer to the application.

The logical place for this to occur is on the server, within the virtualization software layer—but this typically comes at the cost of processing power on the server itself. Ironically, some of the efficiencies gained by virtualization were lost to these services now running on the server itself. In addition to being less efficient, running the services in a virtualization layer on general-purpose compute was less performant than running them on specialized hardware.

Cloud service providers (CSPs) and hyperscalers were among the first to identify this issue. Studies by Google show that 30% of server resources are consumed by a "data center tax", characterizing a set of common building block services related to infrastructure.[1] Facebook research showed that up to 82% of CPU cycles can be spent on common operations that are not core to the application logic.[2] These observations drove the creation of a new set of PCIe®-connected hardware devices that could run inside the servers to offload the microservices and free up server CPU for applications that drive revenue (such as more customer workloads, in the case of CSPs).

This new set of hardware devices has evolved into a new type of processor that lives within servers, known as the *data processing unit* (DPU). DPUs have evolved quickly and there are various architectures. Most vendors combine some form of general-purpose CPU cores with various hardware accelerators and networking components together to build a DPU.

A key factor in the flexibility and performance of a DPU is the architecture. Many DPUs started using general-purpose CPU cores as the primary element to provide services. While CPUs are highly flexible, they are not purpose-built for the types of services and applications running on the DPU. The addition of hardware accelerators helps, but if a new feature is needed, the general-purpose CPU cores will again be tasked with providing the service.

The most performant and flexible architecture will be one that incorporates general-purpose CPU cores, hardware accelerators, and programmable networking components. Figure 1 and Figure 2 illustrate two DPUs: one with basic network connectivity, hardware accelerators, and 16 general-purpose CPU cores, and the other providing a programmable network pipeline, hardware accelerators, and 16 general-purpose CPU cores.

ARCHITECTURE MATTERS: Comparison of DPU Hardware Strategies

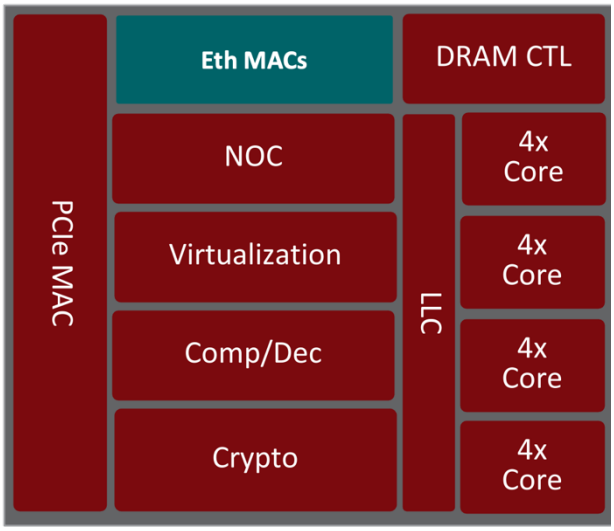

Figure 1. CPU core-centric architecture

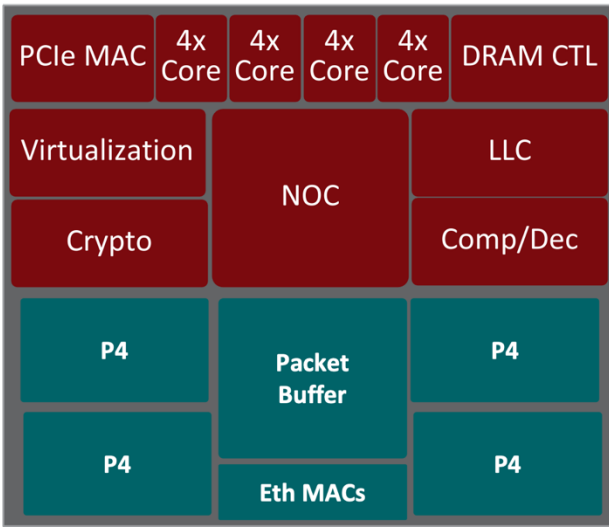

Figure 2. Programmable silicon-centric architecture

To prove the advantage of a programmable silicon architecture we will compare the performance of software-defined networking (SDN) services running on these two architectures.

Test Scenario

The test is to build a multi-stage pipeline that provides the basic features of an SDN network service, one that is commonly offloaded onto DPUs. We can then compare the performance of this service running completely on general-purpose cores to its performance running in the P4 ASICs of the AMD Pensando 2nd generation DPU (also known by its codename “Elba”).

One of the most common network pipelines used in virtualized environments is an overlay network with stateful security control and policing. This pipeline provides network isolation, metering, security and statistics collection in a muti-tenant environment.

For the general-purpose CPU core architecture, the pipeline is developed in C++ and constructed to maximize performance by running the services in user space, using the Data Path Development Kit (DPDK) and Vector Packet Processing (VPP). The DPU

tested comprised 16 Arm cores running at 3.0 GHz, with 12 of those cores dedicated for the pipeline.

The 8-stage pipeline takes input on one port of the DPU, decapsulates the frame, provides input security, polices the traffic, forwards the frame, provides output security, encapsulates the frame and then outputs it on the other port. Figure 3 shows the pipeline's stages along with the VPP nodes used to perform the various actions along the pipeline.

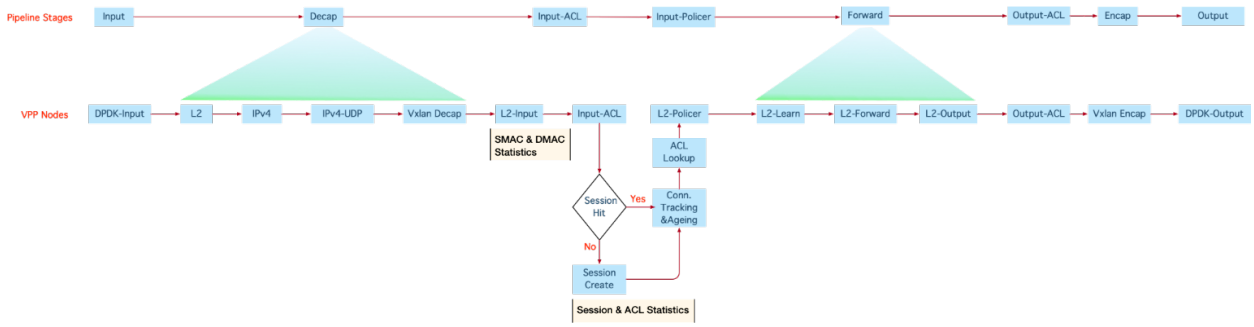

Figure 3. Example SDN Overlay Pipeline with stateful security

When a DPU provides services between ports, this is known as a *bump-in-the-wire* (BITW) implementation. This is a common use case in a host where the DPU is completely isolated from the host OS and is managed through a separate network connection. Another example would be an appliance where the DPU is responsible for the data plane services and communicates with the appliance CPU via PCIe for control plane and management services. Figure 4 shows how a host-based and appliance-based BITW may be implemented to provide distributed services at the server edge.

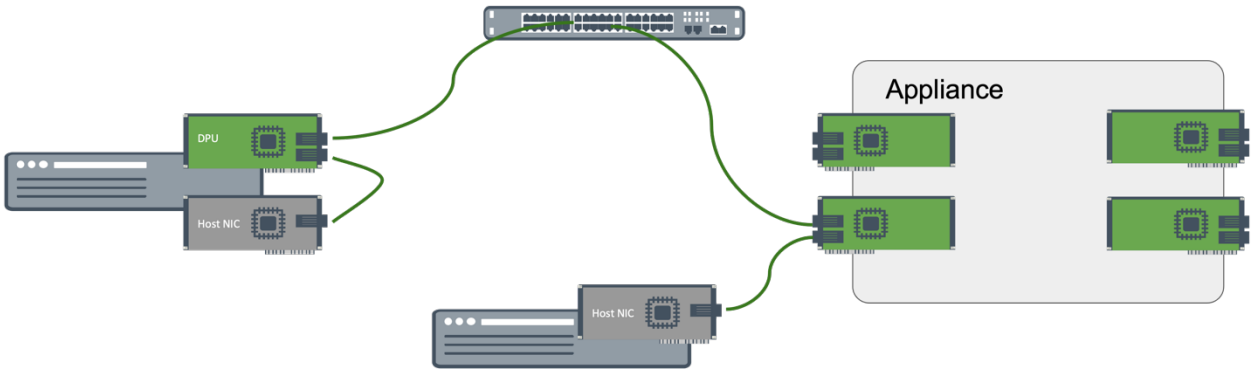

Figure 4. Common BITW implementations

To test the performance of the general-purpose CPU-based DPU, the VPP pipeline was tested on an AMD Pensando Elba DPU, using *only* the A72 Arm cores to process traffic. Traffic was sent between two ports on an IXIA server. These ports were connected at 100 Gb/s and the IXIA sent various frame sizes up to 5 million unique flows to one uplink port at line rate and measured the performance at the other uplink port. Figure 5 shows the test rig setup.

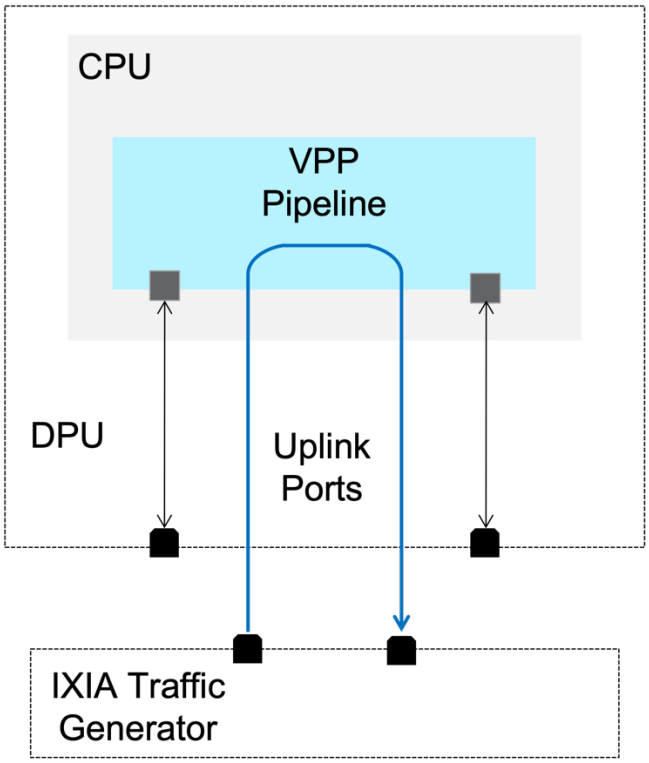

Figure 5. Test configuration

Testing Results

To understand how the various features in the pipeline impacted performance, testing was performed by enabling the features in a cumulative fashion starting with the most basic pipeline which performs basic forwarding functions of VXLAN encapsulated packets. The test was repeated with more services added incrementally until the full pipeline, including tunnel endpoint services, were being handled on the DPU. Table 1 describes the feature scale used during testing of the VPP pipeline.

Table 1. VPP Testing Scale

Traffic Flows	5,000,000
Input Access Control List Entries	3,000
Source MAC (local mappings)	1,000
Input Policers (each source MAC gets a policer)	1,000
Tunnel End Points	1,000

Figure 6 depicts the baseline performance as features are enabled in the VPP pipeline. This data shows that performance drops as additional services are added. The performance of the full pipeline for 1M flows of 128-byte packets was 7.1 million packets per second (MPPS). The greatest performance drop occurred with the addition of ACLs to the pipeline.

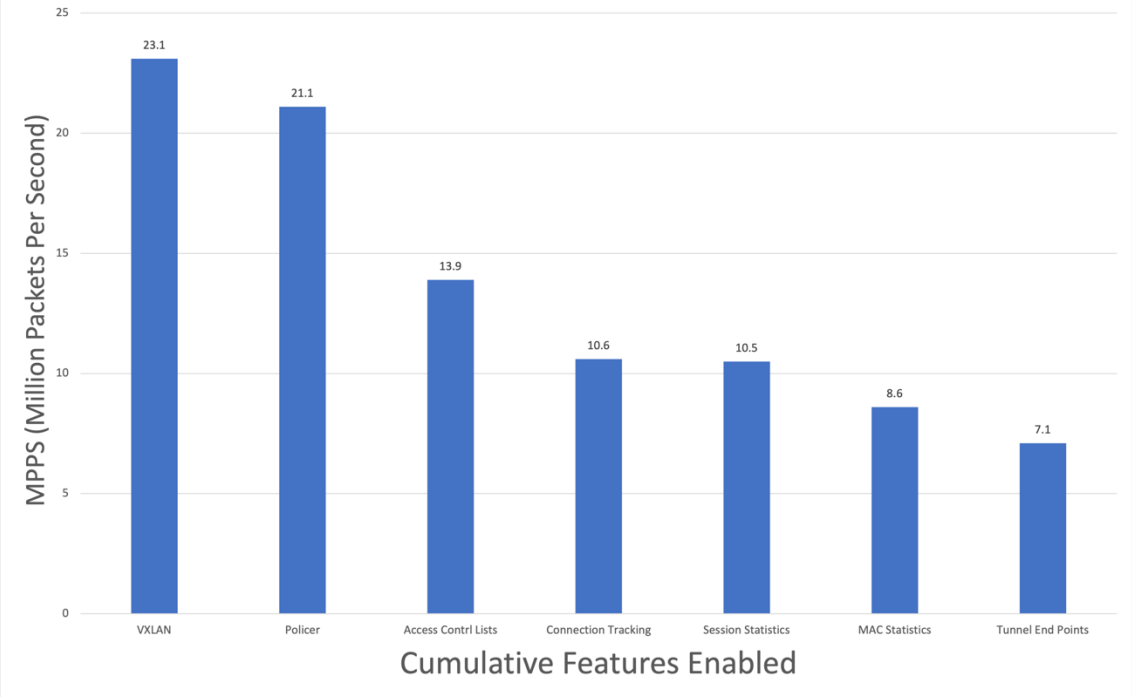

Figure 6. PPS performance

This drop is a result of the number of instructions that the CPU cores must execute because of the added features being enabled. Figure 7 shows the increase in instructions per packet as various features are enabled. With all features enabled, there are ~900 instructions that must be executed for each packet. This puts a great deal of stress on the CPU cores.

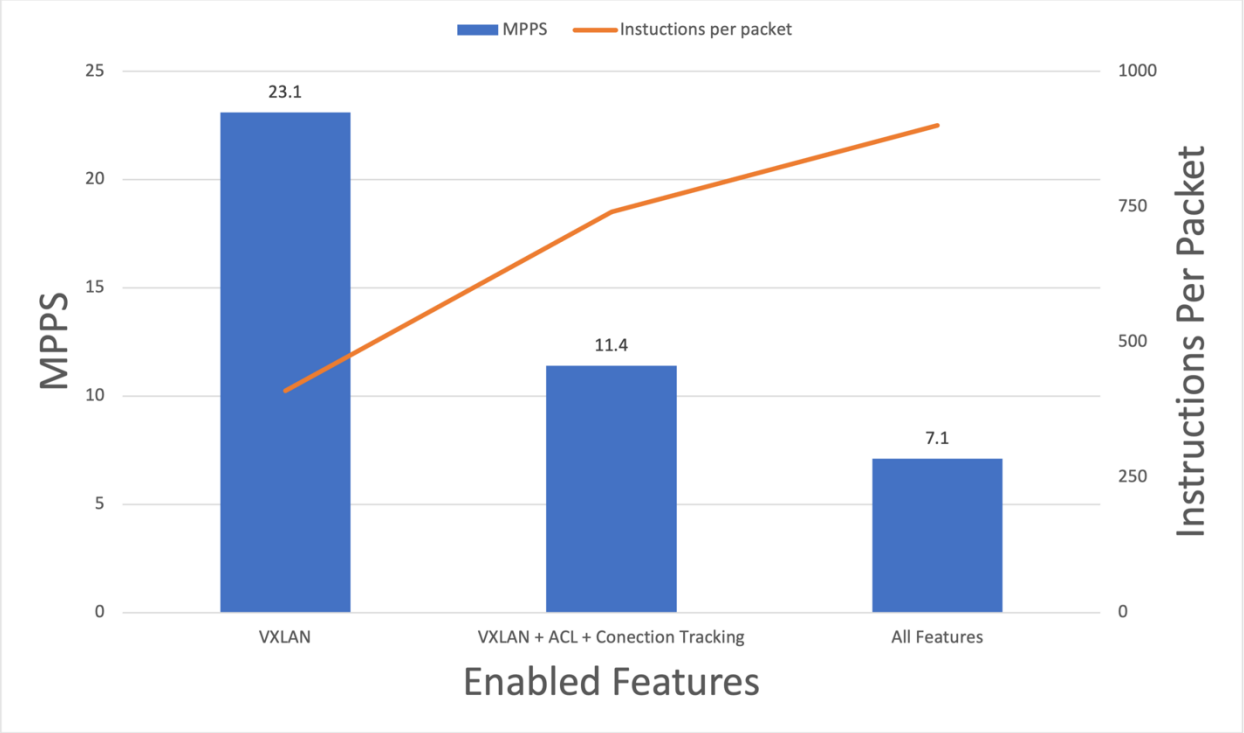

Figure 7. Performance with instructions per packet

Another key metric for a DPU is performance as flow scales. To determine how the number of unique flows impacted performance, the full pipeline was tested for an increasing number of flows. Figure 8 shows that from 1,000 to 1,000,000 flows performance decreases by ~15%, but then remains steady from 1,000,000 to 5,000,000 flows.

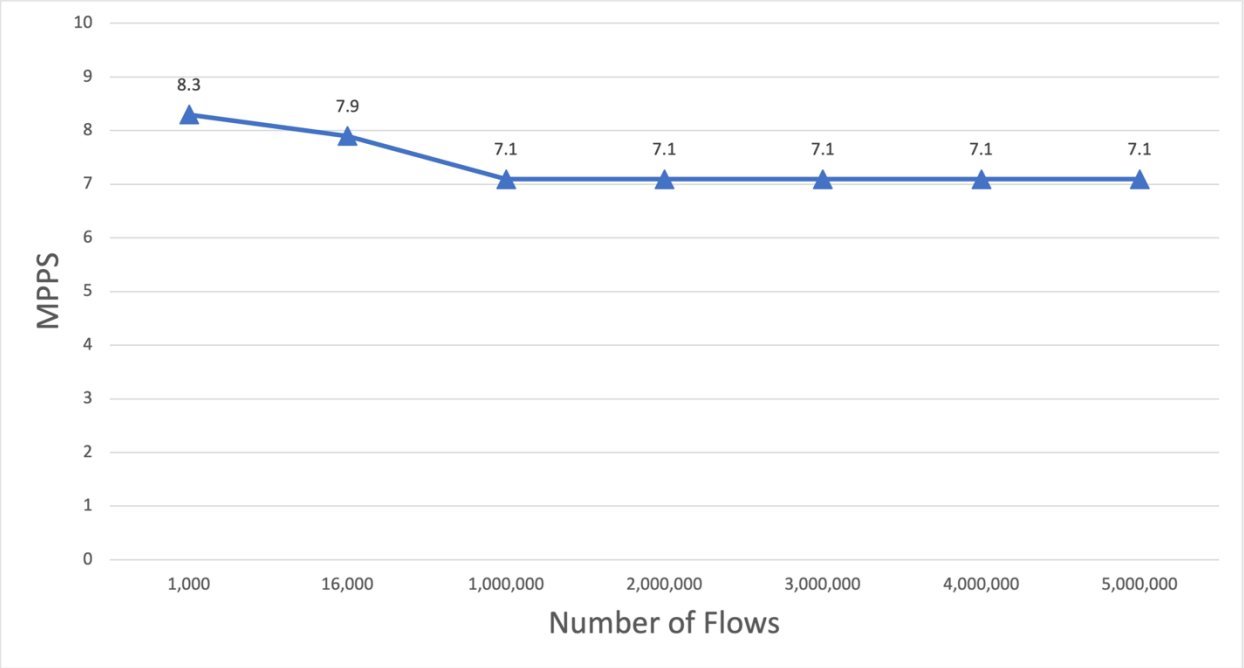

Figure 8. Flow scale performance

One of the most important metrics for a server that is providing a front end for an unknown volume of users is connections per second (CPS), showing how many unique connections can be set up each second. This is especially important for applications that may be hosted in the cloud as a front end for applications such as e-commerce or gaming. For the full pipeline, the general-purpose CPU architecture achieved 40,000 CPS.

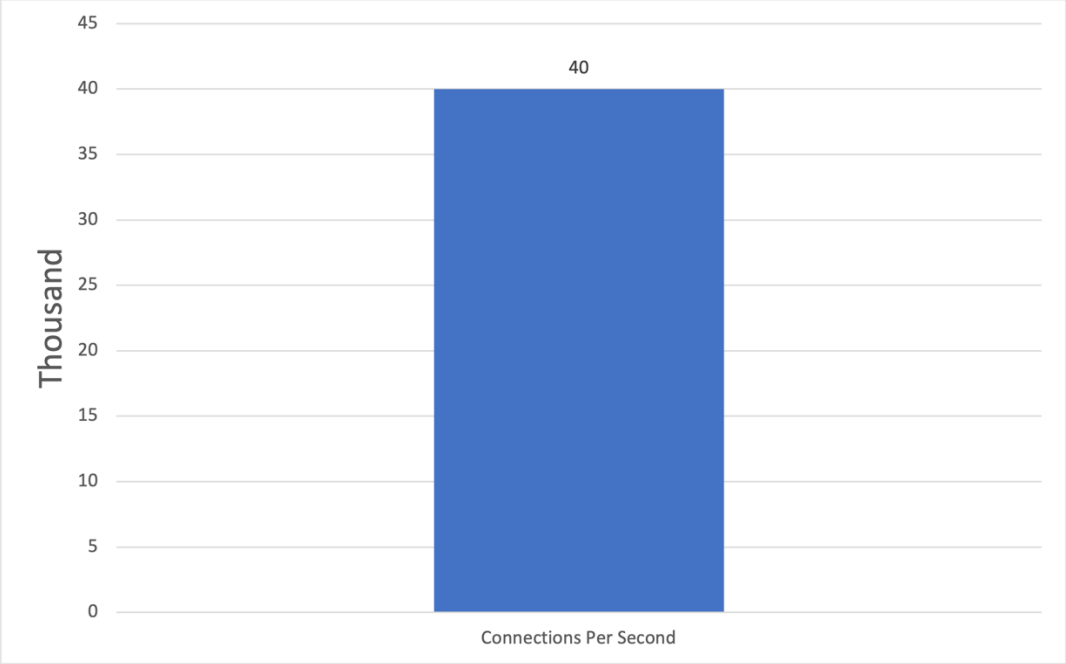

Figure 9. CPS performance for full pipeline

The baseline set of tests were captured with a 128-byte packet size, but it is also important to understand how the performance would be impacted by other packet sizes. Figure 10 shows that as the packet size increases, there is a decline in overall performance.

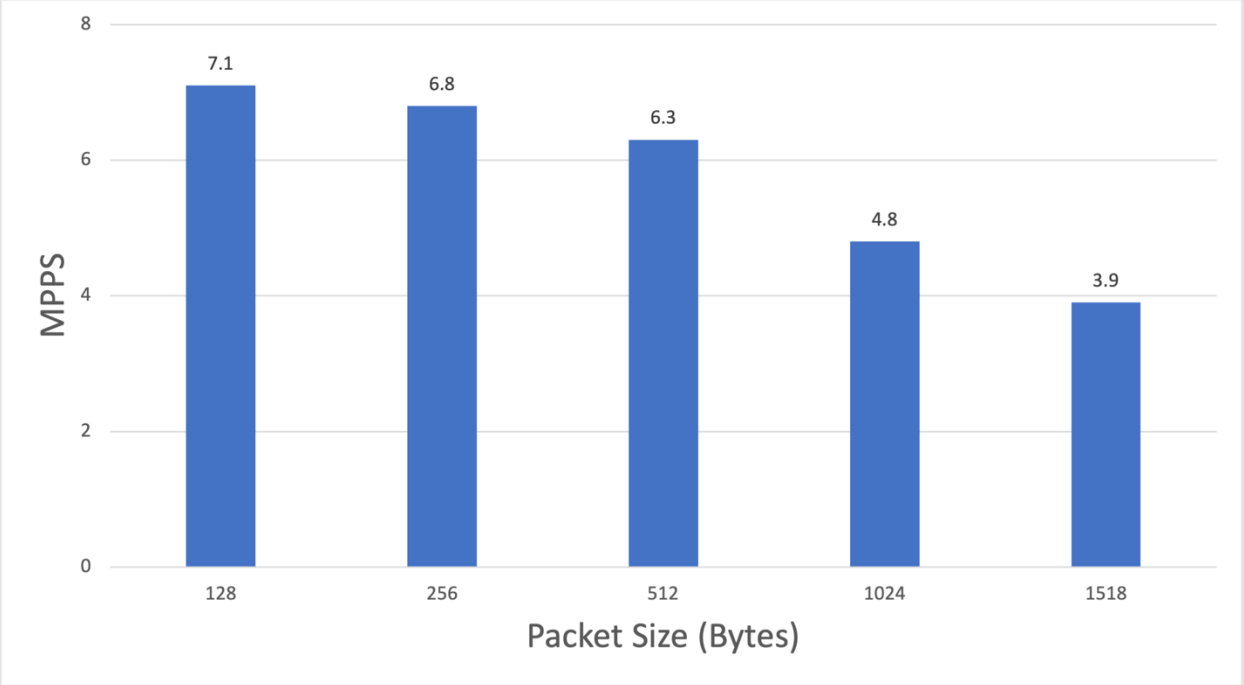

Figure 10. Performance across packet sizes

AMD Pensando DPU Comparisons

Using the same test setup for a BITW pipeline, we can compare the results of the AMD Pensando architecture using its match processing units (MPUs, described below) of the P4 programmable architecture to execute the features in its fast path pipeline. Figure 11 and Figure 12 show that the AMD Pensando DPU provides 7 times the PPS and 100 times the CPS of a DPU that relies on general-purpose cores to provide SDN services.

ARCHITECTURE MATTERS: Comparison of DPU Hardware Strategies

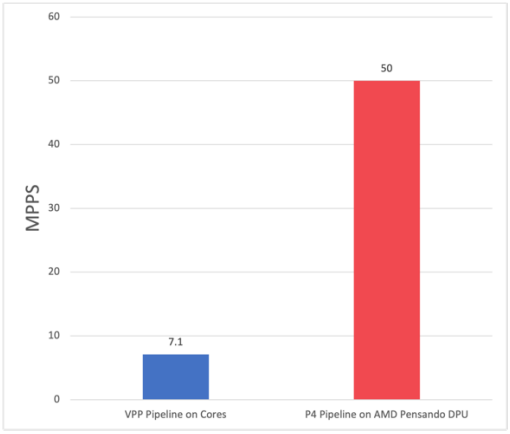

Figure 11. Packets per second comparison

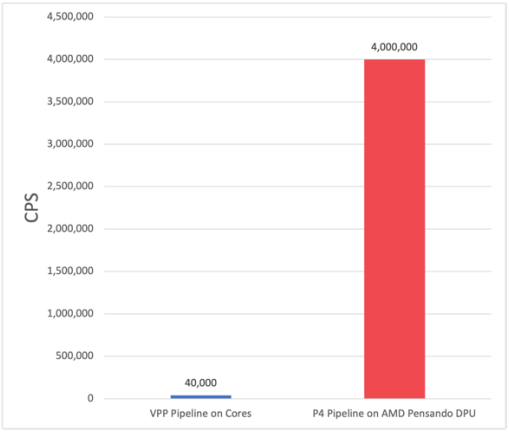

Figure 12. Connections per second comparison

In the case of the AMD Pensando Elba DPU, the pipeline tested was a 32-stage production pipeline developed for a major cloud provider. This pipeline includes all the features of the VPP. The results for this pipeline were measured at maximum scale as indicated in Table 2.

Table 2. AMD Pensando scale for testing

Total Flows	64,000,000 flows / 32,000,000 sessions
Total Connections Per ENI	Up to 32,000,000
Total Endpoint Mappings	8,000,000
Total ENI	100 minimum
Total Prefixes (Routing/Metering/Policy)	54,000,000
Total Access Control Lists (ACLs)	640,000 rules
Total ACLs per ENI	100,000 IP prefixes 10,000 source/destination ports
Total Network Address Translation	64,000

AMD Pensando Software-in-Silicon Architecture

The differentiator in performance is the architecture. A DPU that relies heavily on general-purpose cores is not the best architecture for offloading services. This class of DPU does provide the benefit of moving services off the host cores, and when combined with other accelerators they can provide fast path services, but they lack the ability to provide agility and scale for fast path services. For services that are not available in the DPU's hardware accelerators, the functionality must be executed in the general-purpose cores.

In contrast, the AMD Pensando DPU architecture is purpose-built to accelerate services in silicon. The programmable silicon consists of *match processer units* (MPUs) capable of executing a variety of services. Using a toolchain that is centered around the P4 language, the AMD Pensando DPU can be programmed to accelerate a variety of services and protocols. Due to its programmable nature, it can also be programmed to handle new protocols or customized frame types that include metadata for special handling.

The general-purpose Arm cores in the AMD Pensando architecture are used primarily for control plane and exception packets. Figure 13 shows the architecture of the AMD Pensando DPU. The key central components of the DPU are the MPUs; this is where the software-in-silicon is executed and provides accelerated (fast path) services. Note that the system memory is connected to both the general-purpose cores and the domain-specific MPUs. This allows features that are typically handled by general-purpose cores, like connection tracking, to be implemented on the MPUs and executed directly in the pipeline.

ARCHITECTURE MATTERS: Comparison of DPU Hardware Strategies

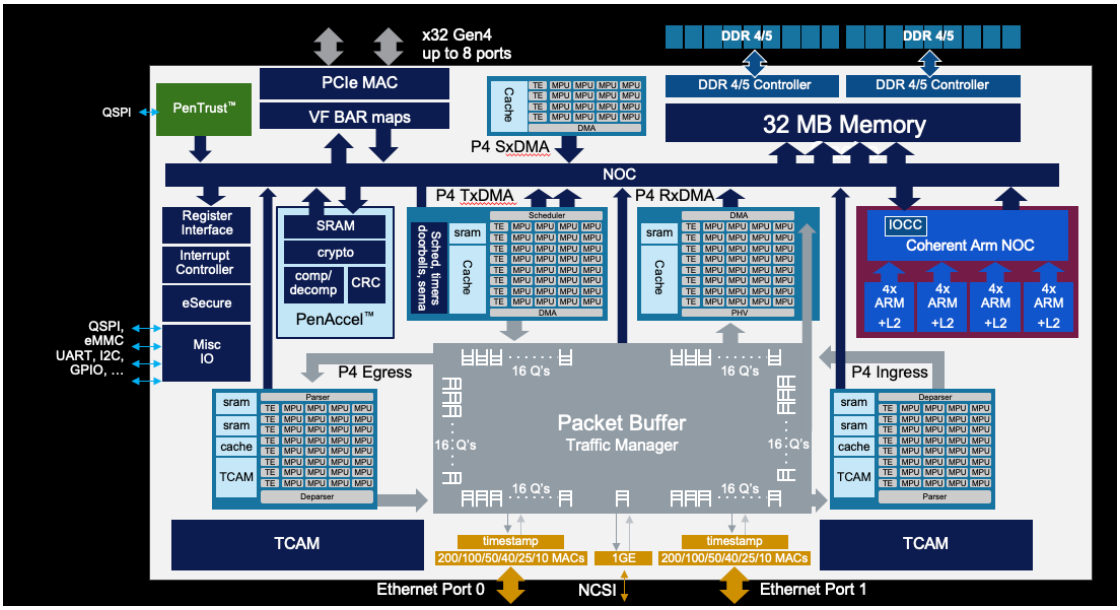

Figure 13. AMD Pensando Architecture

A *pipeline* is programmed in P4 and runs in the elements that connect to the packet buffer, as depicted in Figure 14. A pipeline can include various stages that may provide one or more functions. Each stage can run in parallel on different MPUs, and as packets move through the pipeline, they are not waiting on clock cycles that may be servicing a different stage. In addition, all the MPUs have access to the same memory system, so that a packet passing from one stage to the next is not being copied to a new memory location.

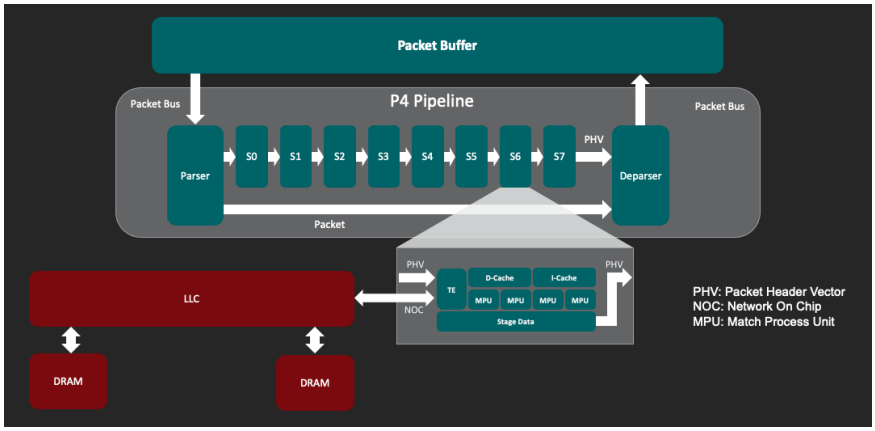

Figure 14. Programable pipeline

Using this architecture, the AMD Pensando Elba DPU can achieve much better performance and scale than it can when using only Arm A72 cores for pipeline processing. The architecture is also fully programmable, so that the consumer can choose what features are implemented on the silicon as opposed to being limited to what accelerated features are available with DPUs that use a fixed architecture. The programmability provides both agility and investment protection because accelerating new protocols in the silicon does not require a change in hardware.

ARCHITECTURE MATTERS: Comparison of DPU Hardware Strategies

Summary

As virtualization increased server efficiencies in the data center, more networking services were offloaded to the server. DPUs emerged as a third processor in CSP and hyperscaler data centers to offload these services. Solutions like the VMware Distributed Service Engine and the HPE Aruba CX 10000 Smart Switch are bringing these processors into the Enterprise data center. In addition, DPUs are starting to be used in security and storage appliances to gain efficiency and improve performance and scale.

Like any technology, DPUs have evolved since their inception. Initially most relied on general-purpose cores and some hardware acceleration to offload virtualization services. Since then, many DPUs have begun to include more hardware acceleration including networking offloads. but these accelerators are static and when they reach a scale limit or a new function needs to be implemented, they fall back to using the general-purpose cores. This results in a major hit in performance.

The AMD Pensando DPU did not evolve from an existing set of products. Instead, it was intentionally architected to offload and accelerate multiple services simultaneously with the scale and performance demanded in the world's largest data centers. The AMD Pensando DPU architecture has a proven track record in multiple CSP environments, with hundreds of thousands of DPUs in deployment.

Learn More

- Product Brief: AMD Pensando 2nd Generation ("Elba") Distributed Processing Unit
- VMware vSphere Distributed Services Engine with AMD Pensando DPU
- HPE Aruba Networking CX 10000 Series Switch with AMD Pensando
- Other AMD Pensando resources on the AMD Documentation Hub

AMD
together we advance_networking

Disclaimer

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

amd.com/pensando

PWP24002

End Notes

[1] https://storage.googleapis.com/pub-tools-public-publication-data/pdf/44271.pdf

[2] https://research.facebook.com/publications/accelerometer-understanding-acceleration-opportunities-for-data-center-overheads-at-hyperscale/