Title: **Virtual Accelerator Engines**
Author: M. Miller
Date: 2020, 04, 03

Document Number:
WP_VAE_VIRTUAL ACCELERATOR ENGINES-200403

Virtual Accelerator Engines

By Michael Miller
CTO, MoSys, Inc.

Introduction

For years, systems architects have been playing cat and mouse by chasing system log jams when it comes to memory, storage and compute bottlenecks. These factors are interdependent; speeding up one impacts the others and adjusting only one may not provide any benefit to the application. Initially, the primary bottleneck was found in the processor clock rate and that was the primary focus for design. When processors exceeded the clock rate of memory, caches were born, followed by multiple CPUs sharing the same data. Over the years, power/performance limitations in process technology have slowed clock rate advances down which led to the integration of multiple cores onto the same silicon.

As new bottlenecks emerged, the industry went through a phase of disaggregating compute and memory. Virtual Machines and containers residing in massive cloud datacenters became a way to provision centralized compute resources. While the core clock rates and effective compute cycles across many cores have both increased dramatically based on all this innovation, there are still fundamental issues still to be solved including:

- ❖ Latency to main memory has stayed virtually unchanged
- ❖ Big and growing datasets are creating a huge challenge
- ❖ Requirement for higher random access to data is becoming the norm
- ❖ Critical bottlenecks for these newer applications that cannot be easily cached

The data in the network today used for connections, analytics, security, etc., has become more unstructured. When it was a spreadsheet, users had nice columns and rows of data they could load as a burst. SQL database tables were rectangular rows and columns too. Now, the structures behind leading favorite social media, entertainment and commerce applications utilize data better organized as a network or as a graph of data nodes. Thus, random access will increasingly rule the day. The industry can no longer rely on the nice cache line burst paradigm to save the day.

The trend in the industry now is to leverage heterogeneous workload accelerators on specialized cards using GPUs and FPGAs with memory. Ultimately, this is all about moving common workloads and datasets closer to specialized compute engines. In order to handle the vast amounts of data and the growing need for lower latency, the next step will be to push compute back out onto the edge of the network and drive customized functions into hardware structures like smart memories with in-memory compute capability. This will improve processing throughput performance and power efficiency and further reduce latency for common functions.

MoSys…An Industry Leader in Intelligent Memory Architectures

Mosys is a leader in solving memory bottlenecks such as bandwidth and memory access and, as such, recognized early on the industry need to embed in-memory functions such as compute, into powerful memory architectures in silicon devices.

Using high speed random-access embedded memory (1T-SRAM), serial interconnect up to 25G per lane (bandwidth up to 800G), and in-memory acceleration functions to speed up common Read-Modify-Write (RMW), the MoSys Bandwidth Engine family (BE2, BE3) of Silicon Accelerator Engines solve the fundamental memory access bottlenecks.

In-memory compute functions in the Programmable HyperSpeed Engine (PHE), another Silicon Accelerator Engine, expand the capability by incorporating 32 RISC cores with tightly coupled 1T-SRAM for the highest throughput and lowest latency between memory and compute. The PHE family includes architectural advances such as multi-port capability and integrated ultra-fast shared processor memories which complement the large 1T-SRAM array on board to further accelerate performance and minimize latency in both internal and off-chip memory access allowing MoSys, or customers, to embed functions to increase performance.

The high random-access rate of the MoSys 1T-SRAM, low tRC and smaller granular access sizes (72b & 144b) make the MoSys memory technology ideally suited for today’s more unstructured randomly organized data. The 32 RISC engines tightly coupled to the low latency 1T-SRAM provide for the ideal algorithm engine to implement complex accelerator engines. The question presented is how to best make use of this accelerator technology.

Virtualized Accelerator Engines

The Problem:

Virtual Machines, Virtual Networks, Cloud Storage and container technology were developed to allow cloud-based datacenters to more efficiently deploy compute resources. For true software code portability, there is a minimum set of hardware functions that must be available on all machines. This same strategy is used in Software Defined Networks (SDN) where the control plane software has a requirement for a minimum set of hardware functions that must exist. It can be restrictive to introduce accelerator engines in such systems because any dependencies on the accelerator engine may limit the number of available platforms that higher level software can run on. This is especially true when the accelerators each has its own software interface.

So, today the problem is, depending upon the performance required, when an accelerator is chosen, the software must be changed to accommodate that specific accelerator’s interface. This becomes a large impediment as to how far an accelerator can be deployed or how broadly an application can take advantage of accelerators. Furthermore, today’s engineers need to focus on the higher-level functionality of new emerging network-based applications, not firmware and how some Instruction Set Architecture interacts with memory in order to implement the best filtering algorithm.

The Solution:

The solution then is to virtualize the accelerator engine by creating a fixed abstract functionality and API for a specific function across different hardware environments. The virtualized function can run on the CPU alone, FPGA only, or higher performance solutions based on an FPGA with an attached accelerator IC. Based on a single software model, the same virtualized function can be available in a broader set of environments and scaled extensively across a wide performance range.

MoSys Creates the Virtual Accelerator Engine Platform

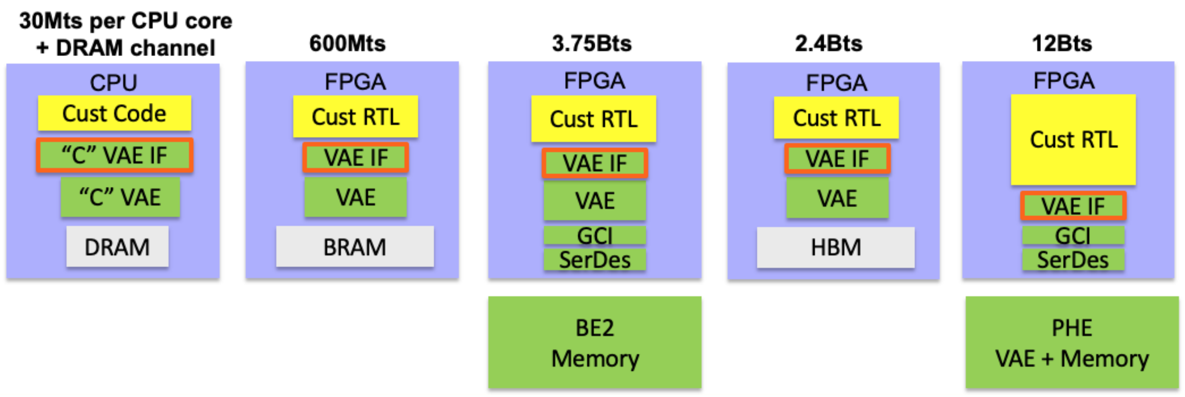
MoSys has developed embeddable Virtual Accelerator Engines (VAE) which can span implementations including software versions executing on a CPU, optimized FPGA solutions, and highly accelerated solutions using FPGAs with MoSys Accelerator Engine ICs such as the MoSys PHE.

There are several key attributes which are at the core of a MoSys VAE Platform's value proposition that are discussed in the following sections:

- ❖ A single Virtual Architecture
- ❖ Single Common API (VAE API)
- ❖ Single Common RTL Module Interface (VAE IF)
- ❖ Adaptation Layer software to higher level existing API

A Single Virtual Architecture

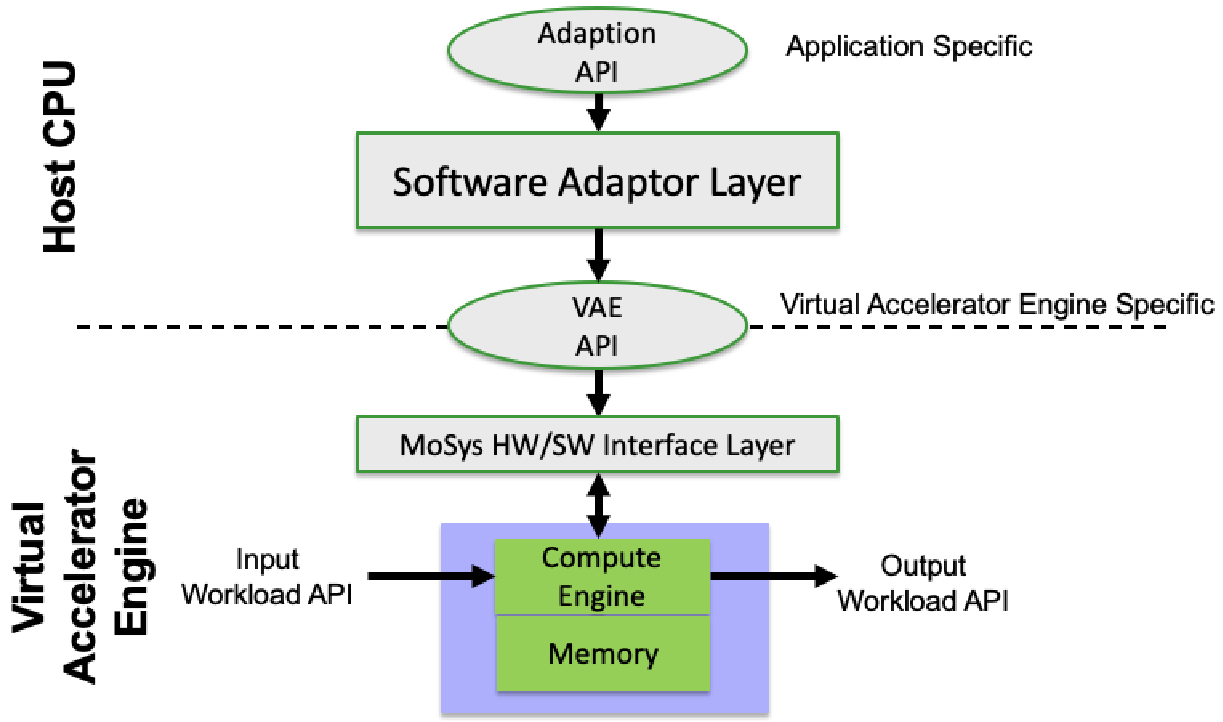
MoSys VAEs software interface and functions are the same across all implementations. This allows for a wide range of instantiations with a large upside range of performance from 30M operations per second with a single standard CPU core and DRAM to more than 12B operations (read + writes) per second using PHE. The highest performance gains are driven mostly by improving the effective random-access memory rate and represent a range of more than a 400x in performance boost. Alternatively, using a solution with DRAM and HBM will deliver the highest density with system blocks of memory measured in 100s of Gigabytes compared to faster but smaller BE or PHE solutions with 100s of Megabytes. This allows the designer to trade off memory speed vs. capacity but maintain the same basic software structure.



Each VAE platform is designed to support a specific application function such as classify input vectors, indexing databases, or performing optimized calculations where the base function can be generalized enough to fit many types of specific applications.

Single API

Each VAE is implemented with a common API for configuring, managing and presenting workloads to the accelerator. A common API will allow software engineers to deliver the specific accelerator function across platforms without changing the system software structure; only the hardware implementation will change, but in FPGA-based systems, the I/F will also be common.



In most cases, there will be a Software Adaptation Layer that runs on the host that maps from the application layer API to the VAE API. The adaptation layer will perform tasks of mapping or translating from a specific data structure such as machine learning classifier or header look up key into the data structures executed on the available hardware available for the VAE.

Single Common RTL Module Interface

Most VAE platforms will have FPGA-based products each having a different performance capability. In these cases, the RTL interface across these FPGA products will have a Common RTL logic specification (VAE IF). Designing to a common hardware interface allows for easier migration of performance and capacity. Additionally, the designer can create a wider range of product offerings and the opportunity for easier implementation of add-on acceleration modules. At the same time, this provides future proofing because as new silicon becomes available with a VAE port, the designer can drop it into the same logical "socket".

In the end, this leverages and protects the software investment, and allows performance scaling over multiple hardware environments.

Adaptation Layer Software to Higher Existing API

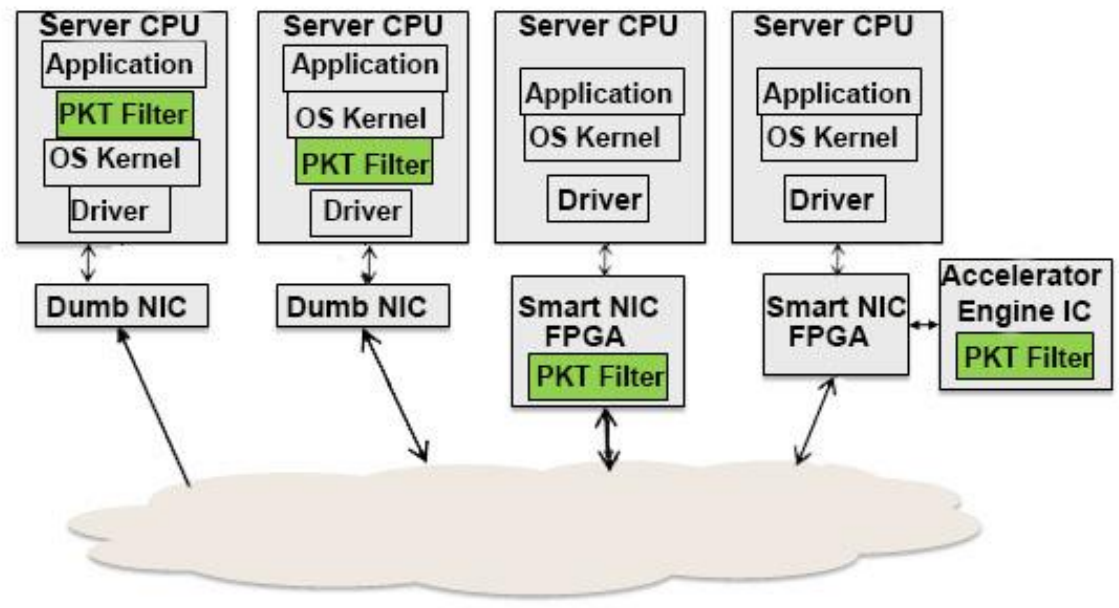
New software can utilize the VAE API directly, achieving the highest level of performance and portability. Often the designer must work with existing software, which will require a different method to address.

Two possible approaches for integrating into existing software:

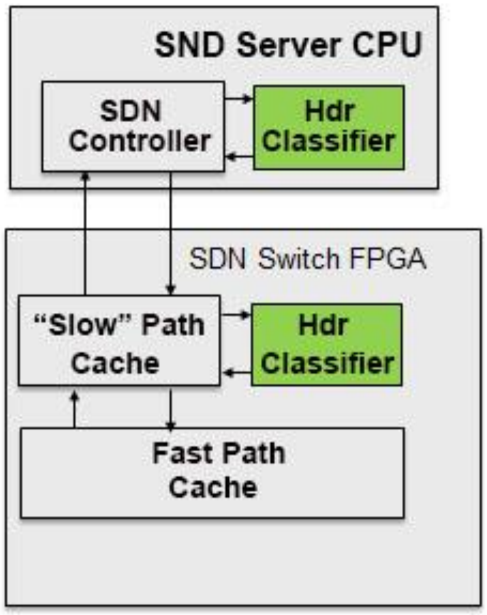
- ❖ Call the virtualized accelerator engine API directly or,
- ❖ Create an Adaptation Layer of software with its own specific API. This software adaptation code takes the current software output and adapts the data to execute the virtualized accelerator engine. Depending on the platform application product, MoSys may supply this Adaptor software.

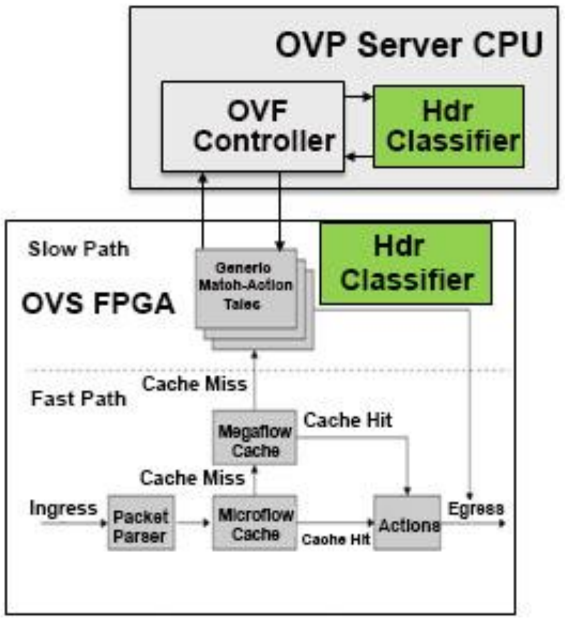
The adaptation code has an API that is customer-specific and allows the customer to protect the transportability of the code to different hardware environments based on application performance needs.

For example, a packet filter could be implemented as a Virtual Accelerator and be deployed at different points in the stack and different points in the network, all with the same software control API where some instances are pure software on a CPU all the way to hardware accelerators. Having the same common API enables portability of higher-level platform functionality. As new HW accelerator technology emerges there is an increased probability that it can be utilized without disrupting the layers above, thus providing a future proof path to increased performance scaling and capacity.



Utilizing the concept of a common API and a virtual function, the VAE architecture can be used in hierarchical systems where parallel paths processing packets from fast paths to slower paths exist. The benefit of the common API is that the same software stack can be used to control each of the paths in the hierarchy in a unified fashion.





Target Applications for Virtualized Accelerator Engines

MoSys virtualized accelerator engine technology and products are targeted initially towards accelerating embedded storage, search and classification applications which require high random-access rate of finite data structures. In many systems today, there is a hierarchy of databases that may range from large to small such as: large structures in non-volatile memory, "in-memory" databases held in DRAM, and smaller subset databases held in embedded memory close to the decision-making logic. In many cases, the embedded memory with in-memory processing may be clustered to create larger aggregate storage giving benefits of both size and performance.

Embedded functions that can benefit from this technology include:

- Key value pair databases
- Networking search functions
- Machine Learning
- Computation
- Algorithm acceleration
- Security analysis
- And more

MoSys will focus on:

- ❖ Embedded applications
- ❖ Apps that require high random-access rate data
- ❖ Applications requiring low latency

Target markets where these functions are used include:

- ❖ Smart NICs
- ❖ Security appliances
- ❖ Network hardware
- ❖ Datacenter acceleration cards
- ❖ 5G edge compute
- ❖ Aerospace and &Defense
- ❖ High-performance computing
- ❖ High speed test and measurement equipment
- ❖ And more

Summary

As more and more applications rely on analysis of unstructured and random data, compute accelerators will continue to gain importance and will migrate to the edge of the network in order to address the need for reducing latency and increasing processing throughput for common functions.

MoSys virtualized accelerator engine technology and products are designed to accelerate embedded storage, search and classification applications which require high random-access rate of finite data structures while insulating today's engineers from the minutiae of programming firmware and hardware.

Since the API is common, the same virtualized accelerator engine software can be used from the central part of a system and its use can be replicated to the edges, making it a cost-effective solution for broad deployment thus achieving a high degree of platform portability.

Finally, the virtualized function and the common API work together to provide for future proofed road maps that can incorporate higher levels of hardware acceleration, future improvements in silicon technology and expanded capacity ranges.

###