# The Case for Intel® FM10000 in KVM Acceleration

## A Proof of Concept with OpenVSwitch Offload

May. 2016

# General

The use of OpenVSwitch in OpenStack enabled KVM environments is becoming a commonplace. Being a strictly software implementation, OpenVSwitch is flexible to exhibit an impressive set of capabilities, addressing various requirement typical to carrier networks (visibility, policing, QoS, steering, tunneling), as well as many and other such demand of network virtualization and NFV (VLAN, VxLAN). However, basic table lookup and packet forwarding, let alone more complex forwarding decision making, all come with a good deal of CPU utilization needs, hurdling scale up in bandwidth and total number of VMs. But not just packet processing per se poses the highest burden. It is those extremely frequent interrupts issued times and again as packets traverses either northbound-

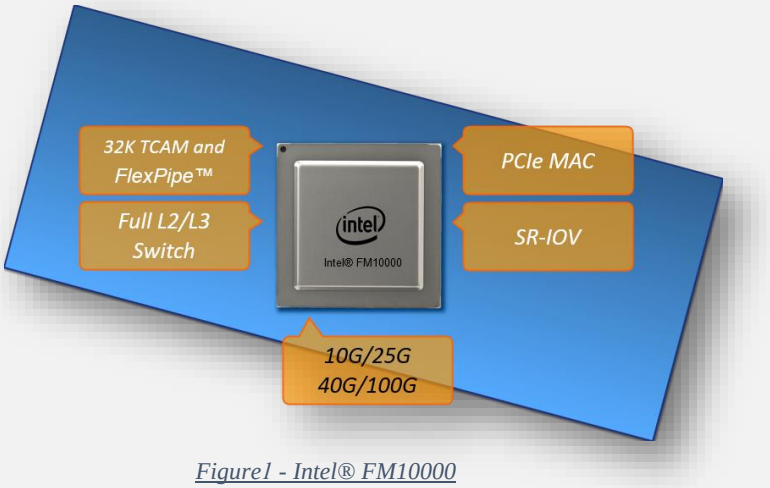


*Figure1 - Intel® FM10000*

southbound or eastbound-westbound, quickly choking up CPU. In fact, software switch exhibit non-linear performance drop as lookup tables increase [1]. This is the main drive to move forwarding tasks off CPU, down to a purposely built switching silicon. Close to 35% of CPU power could be saved and be freed for business logic processing.

Silicom targets to demonstrate how an FM10000 based adapter is an integral, logical and natural extension and enhancement to DPDK based OVS, allowing for (1) inter VM forwarding and live migration; and (2) Flows forwarding or dropping without ever bothering the host's OVS.

This paper demonstrates two approaches for enhancing overall virtual switching performance. Both share the common trait of interrupts mitigation. Both take advantage of Intel® FM10000 switching silicon. Taking yet another advantage of the latter and its integrated PCIe MACs, Silicom built a line of PCIe adapter based on it, featuring wide range of 10Gbps, 25Gbps, 40Gbps and 100Gbps links, with fully fledged ASIC switching engine, assisted with large TCAM, all in acceptable size and power envelope.

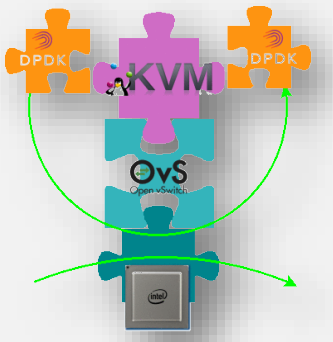


*Figure2 - OVS Offload*

# Intel® FM10000

Featuring rich data path in one die, Intel® FM10000 offers a wide range of PCIe MAC host interfaces, SR-IOV capabilities, fully functional switching fabric with FlexPipe™ engine and 32K TCAM, and set of SERDES interfaces to accommodate up to 100Gbps links. To understand and demonstrate FM10000 potential value as an offload engine for virtual switching, Silicom demonstrates a 3 steps proof of concept tests. All tests share common high level structure of an east-to-west inter VM communication and traffic forwarding. Silicom offer a wide range of form factors of PCIe network adapters, based on FM10000, and the one chosen to these tests features four 10GbE external ports and one PCIe Ethernet host interface [2].

Two major concerns are common when it comes to virtual switching in general. One is the inter VM (east-west) communication. It involves a lot of costly operations that the hypervisor should perform. Copying data across VNICs, forwarding table lookup, let alone header stripping of prepending (tunneling), or other types of traffic management and policing, all pose cycles consuming requirements to the general purpose CPU. Moreover, visibility of the traffic that is forwarded across VM, between one another become also a concern. Tapping those virtual wires is simple (after all, it is software), but on the same time, exponentially costly.

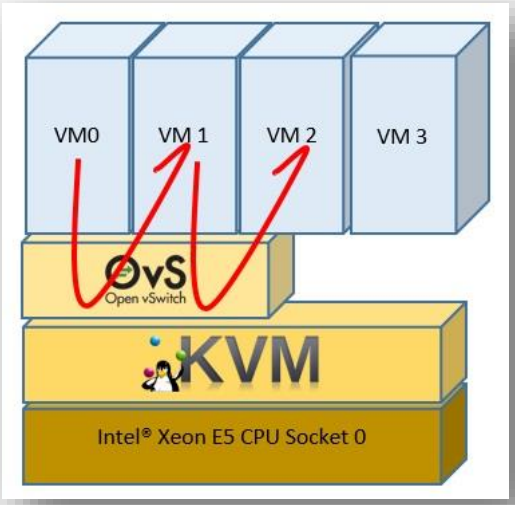


*Figure3 - OVS Forwarding*

The other concern involves VM live migration. Essentially, VM live migration is achieved by copying memory content of a VM in its source location, to its new incarnation in destination environment (hypervisor). Due to the volatile nature of traffic coming into a VM, VNIC memory synchronization is one of the trickiest parts of VM live migration.

The 3 steps approach of the PoC performed by Silicom is aimed, on one hand, at demonstrating how efficient FM10000 can be with switching and forwarding offload, and on the other, at addressing those concerns:

- Step 1, OVS forwarding – Testing and measuring native software bridge forwarding;
- Step 2, SRIOV – Testing and measuring VF assigned in pass through mode to a VM;
- Step 3, DPDK – Measuring switching with kernel bypass, eliminating interrupts.

Step 1 is brought herein as a reference. The native architectural approach of OVS switching is, on one hand, simple, with native OS bridging and tap device, with virtio-net based front end VM network device. On the other hand, this approach is far from being optimal in performance. Step 2 is brought to demonstrate the other extreme, which is forwarding that is performed entirely by switching ASIC on behalf of the VM, with no software involved in the data path. Step 3 is brought as a reference for software based data path that eliminates interrupts, while preserving the ability of live migration (by using virtio-net, for instance), while leaving a lot of space for offloading forwarding tasks to FM10000 mechanisms.

# Forwarding Tests Results

Test setup was simple. A standard Xeon based server was equipped with FM10000 based PCIe adapter. KVM with OVS was installed on the server, and seven virtual machines instances were set, each configured to forward the received traffic onwards to the next, while the last one VM in chain was configured to send traffic back to test equipment. On first iteration, one VM was tested. Next, two VMs were tested, etc.

| | 1 VM | 2 VMs | 3 VMs | 4 VMs | 5 VMs |
|---|---|---|---|---|---|
| SR-IOV | 9.621 Gbps | 9.618 Gbps | 9.613 Gbps | 9.608 Gbps | 5.679 Gbps |
| OVS | 0.522 Gbps | | | | |

*Table1 - SR-IOV and OVS Tests Results*

Native OVS forwarding appear to be a very low reference point, with well below one Giga bit per second of forwarding capability. At first, these results seemed to be fouled in some manner, but later on it became clear that these are consistent results. Same traffic pattern was employed with SR-IOV configuration, in which case, forwarding much closer to true wire speed.

It is interesting to point out, that, the forwarding mechanism in the SR-IOV setup that actually performed the forwarding is the physical function switching (eswitch) mechanism that enables MAC/VLAN based switching across virtual functions. One more interesting point to notice is how the SR-IOV tests results adhere to the PCIe bandwidth limitation. The effective bandwidth of a 8 lanes PCIe v3 bus is about 45-47gpbs, which reflects very nicely in the total sum of the tested bandwidth.

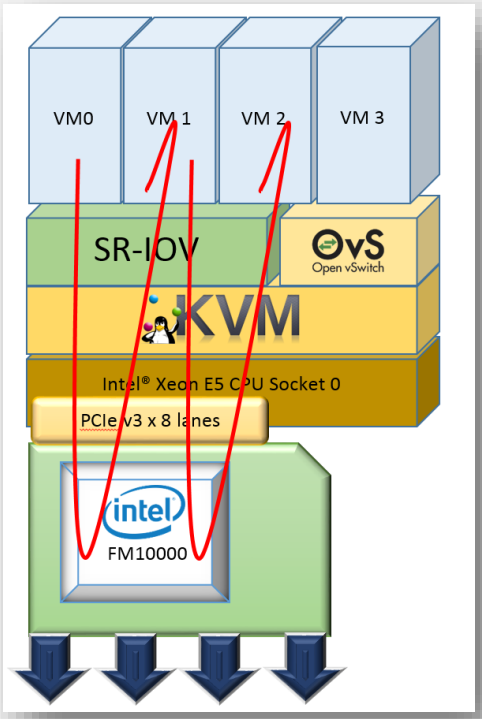


*Figure4 - SR-IOV with FM1000 Inter VM Forwarding*

# Next Step in PoC

Having demonstrated the value of bypassing the software switching and forwarding mechanisms, and the outcome of reducing the number of interrupts in the system, the natural next step in this series of tests would be implementing this traffic forwarding scheme over a DPDK OVS implementation, accelerated by FM10000. The advantages of such setup are many:

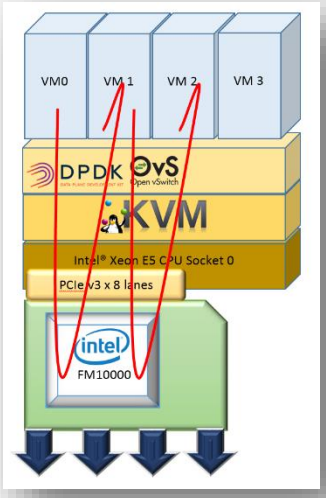


*Figure5 - DPDK OVS with FM10000*

1) Virtio – the use of virtio-net would adhere to the basic requirements of VM live migration

2) Kernel bypass in the data path – DPDK enabled data path by its nature, is poll mode based, thus eliminating system interrupts to the minimum.

3) User space – DPDK OVS daemon and forwarding logic is implemented in user space, so as the control plane of FM10000 switching fabric. That way, it would be feasible to implement netdev provider to couple the FM10000 to the DPDK OVSD, and data path provider as well.

Test setup included SuperMicro X9DR3-F based sever with Xeon E5-2650 @ 2.6 GHz, and Spirent test center for traffic generation.

## REFERENCES

[1] Removing Roadblock from SDN: OpenFlow Software Switch Performance on Intel DPDK, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6680560&tag=1

[2] PE310G4DBIR http://www.silicom-usa.com/10_Gigabit_Content_Director_Server_Adapter_PE310G4DBiR_54