White Paper

Introduction to SFC91xx ASIC Family

Steve Pope, PhD,
Chief Technical Officer,
Solarflare Communications

David Riddoch, PhD,
Chief Software Architect,
Solarflare Communications

The 91xx family of ASICs is Solarflare's 4th generation Ethernet controller. It builds upon the highly successful 90xx range of controllers offering:

- Higher performance, through a PCIe 3.0 host interface, support for 40G Ethernet interfaces, and an all new internal data-path micro-architecture.
- Improved offload capability, through the addition of hardware based TCP segmentation and reassembly offloads, VLAN and VxLAN and FCOE offloads.
- Improved flow processing capability, through the addition of dedicated parsing, filtering, traffic shaping and flow steering engines which are capable of operating flexibly and with an optimal combination of a full hardware data plane with software based control plane.
- Improved switching capability, through the addition of a hardware switch fabric on the silicon, capable of steering any flows based on Layer2, Layer3 or application level protocol between physical and virtual interfaces and fully supporting a software defined network control plane with DCB/PCI-IOV virtualization acceleration for high-performance operating systems and virtual appliances via physical or virtual functions.
- Improved time stamping capabilities, through the addition of a fully integrated time-stamp unit, the 91xx ASIC can be used to generate high-precision hardware timestamps both as packet meta-data and inserted into frame data.

In financial services, high performance TCP and UDP processing is an important attribute for applications such as market data feed handling, exchange gateways, order routers, direct market access and algorithmic trade execution. For these applications, there are a number of specific enhancements which have been made and productized through the Solarflare SFN7xxx cards. These enhancements are described in this white-paper. The following documents are available from Solarflare and contain further information:

SF-105918-CD	Introduction to OpenOnload White Paper
SF-104474-CD	Onload User Guide
SF-103837-CD	Solarflare Server Adapter User Guide

Many other features of the SFN7xxx cards not described in this paper are useful in other environments such as enterprise virtualization, broadcast video, big data analytics, HPC and web appliances.

Reduced Latency

The 91xx ASIC features a new higher speed internal data-path and also operates at PCIe 3.0 bus speeds[1]. These features combine to significantly improve system latency, as shown in the following table for a number of micro-benchmark comparisons between the SFN6122F and SFN7122F adapters.

Test	SFN6122F Latency (us)	SFN7122F Latency (us)	Improvement (%)
Netperf RR UDP	2.2	1.7	23
sfnt-pingpong UDP	2.2	1.7	23
Netperf RR TCP	2.4	1.8	25
sfnt-pingpong TCP	2.4	1.8	25
EFPIO UDP	N/A	1.3	-

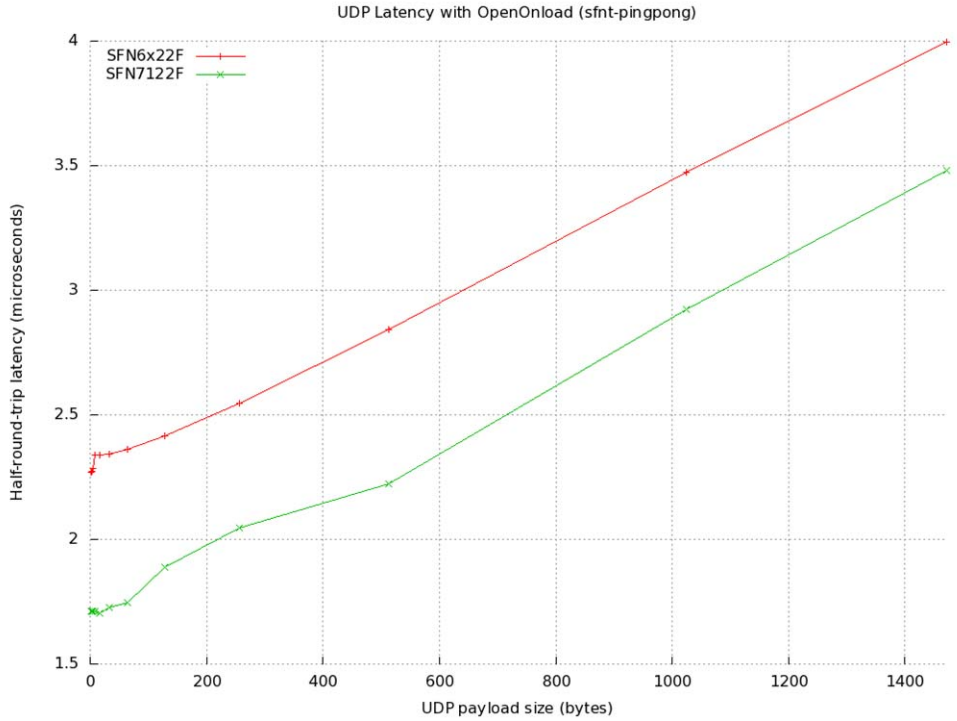

Figure 1: Latency reduction of SFN7122F compared with SFN6122F Adapters.

Figure 1 illustrates the SFN7122F latency reduction over a range of packet sizes. This reduced latency improves tick to trade times, helps market making algorithms and reduces queuing, improving system resilience to micro-bursts.

[1] Expressed as raw transfers/second PCIe 3.0 operates at 8GT/s compared with PCIe 2/0 systems at 5GT/s. However, PCIe 3.0 supports a more efficient symbol encoding scheme (128b/130b rather than 8b/10b) at its physical layer which together with the higher operating speed, results in a doubling of the link bandwidth to ~1GB/s per lane per direction.

Replication and Switching of Multicast Packet Flows

This is a common requirement for applications where independent threads are required to subscribe to the same multicast flows arriving from the network. Previously host software would be required to copy such flows to the different application threads, for example using the Open-Onload stack sharing mechanism. However with the 91xx series ASIC, this operation can now be performed entirely within the ASIC, resulting in the elimination of software copies and any inter-process synchronization.

Figure 2 shows an IP multicast flow which has been subscribed to by two distinct user-level processes. Each process is linked with the *libonload.so* protocol library which during the handling of a multicast join operation will insert a filter at the NIC to request the IP flow be delivered to its RX descriptor ring. For each installed filter, the NIC will replicate the incoming frames of the flow and deliver independently to each of the RX rings.

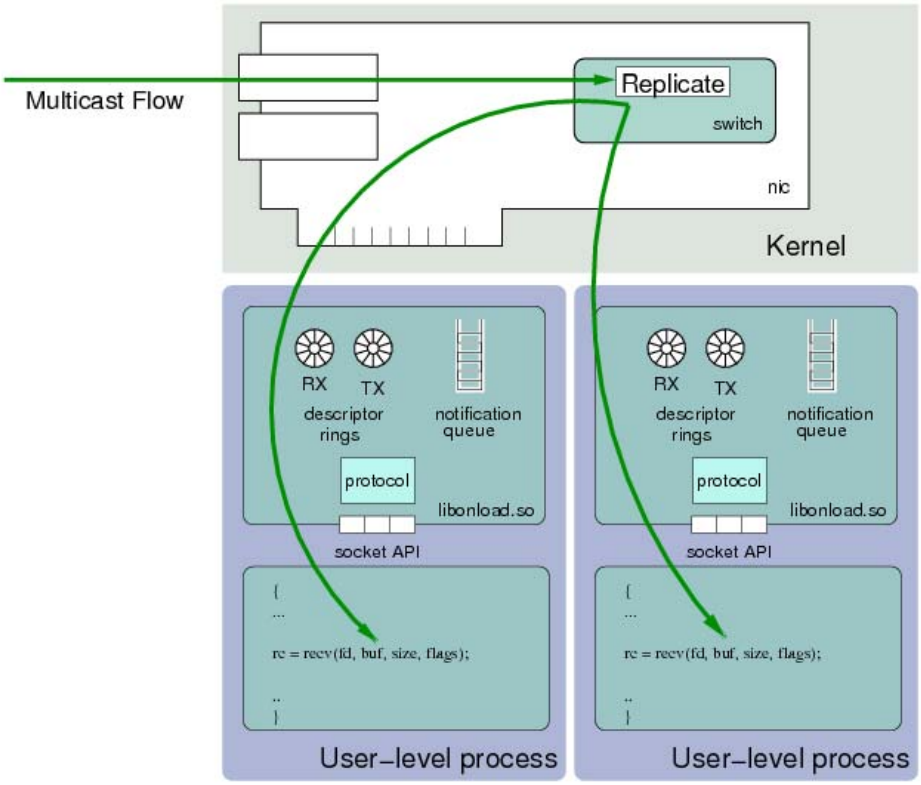

Figure 2: Multicast Replication on ingress to separate user level processes.

This switching feature is fully symmetric, for example as shown in Figure 3, frames arriving at the NIC from a transmitting process on the same host as a consuming application can be replicated and switched both onto the physical Ethernet port as well as back to the host. The operation takes place without any software cross-talk between the applications. Additionally,

filtering operations can be specified flexibly from any of the frame header bits. This would allow for example, applications to subscribe to multicast feeds which differ only by VLAN.

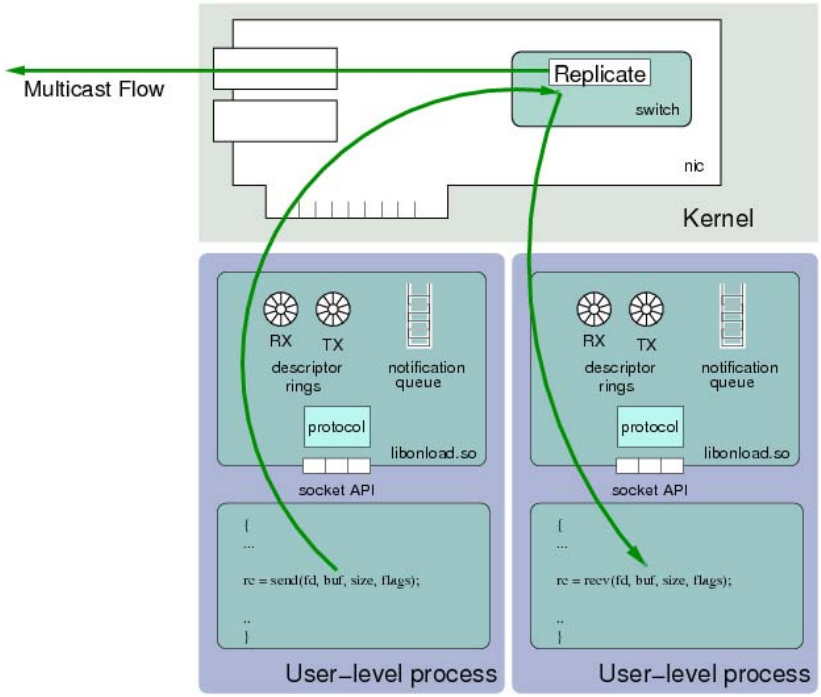

Figure 3: Acceleration of multicast transmission with locally subscribed receivers.

Microburst Resiliency through Unified Switch Architecture

The 91xx ASIC switch behaves as a non-blocking switch fabric with fully virtualized internal buffer management. This feature has the benefit that all of the data-path buffer memory within the ASIC can be efficiently and flexibly managed to cope with the demands of all the physical and virtual ports of the device. For example, consider the scenario of a heavy burst of traffic arriving on a port and (perhaps due to host memory bandwidth limitations) which cannot be delivered to the host at line-rate. This scenario can be efficiently handled by allowing one port to use buffering which may be transiently available because another port is relatively idle. The scheduling decisions regarding the allocation of the buffers are fully under the ASIC's firmware control, enabling sophisticated memory management algorithms to be deployed. This, together with the ability of the 91xx data-path to deliver to the host at a sustained data-rate greater than 60Gb/s provide significantly improved resilience during peak traffic conditions.

Scalable Address Translation Service

In order to ensure system integrity whilst providing unprivileged address space access to DMA capable virtualized hardware, all Solarflare ASICs support an address translation service (ATS) between application virtual memory and the PCIe bus physical addresses required by the ASIC for DMA operations. While this function is now supported by many IOMMU enabled server motherboard chipsets, it is the case that driver support for IOMMU remains immature for many operating systems and many Solarflare customers prefer to use the address translation service provided on the Solarflare ASICs with mature driver support.

On older generations of Solarflare ASICs, each address translation entry mapped a 4KB or 8KB page size. This enabled fine grained scatter gather operations and is a natural size for many operating systems. However, the small page size consumed a large number of ATS entries within the ASIC (typically 30K per Onload stack instance) and exhaustion of the ATS (also known as buffer table) resource was typically the first limit to scaling Onloaded processes. If no ATS entries are available, Onload is unable to allocate DMA coherent buffers and must resort to processing network flows via kernel resources and therefore with reduced performance.

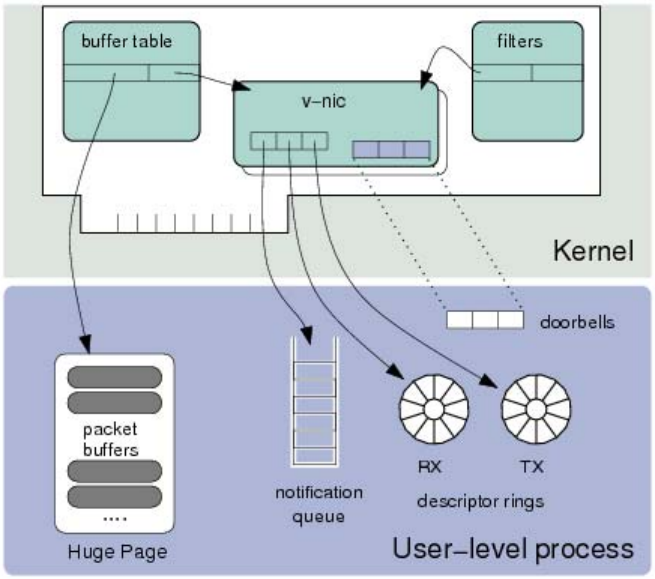

Figure 4: Mapping onto a Huge Page fragmented into a large number of packet buffers.

However, as shown in Figure 4, the ATS implementation within the 91xx ASIC is able to map up to 4MB of address space per entry. When used in conjunction with Huge Page operating system support, each ATS entry may now map onto a large number of MTU sized buffers, potentially enabling a 50-100x increase in the number of Onload stacks which can be allocated concurrently per host.

PIO and Templates

The 91xx ASICs are able to support programmed IO (PIO) operations as well as Direct Memory Access (DMA) models of data transfer from the host. When using PIO, the CPU transfers data directly over the PCIe bus via load/store operations through non-cacheable memory. This provides the lowest possible latency. Conversely, DMA is the more traditional model of data transfer, also supported on Solarflare ASICs whereby data to be transferred is identified to the NIC by the CPU by means of descriptor rings and doorbell writes, but the actual transfers are handled asynchronously by the ASIC. The Solarflare implementation of PIO is designed around a concept of *templated* transmission.

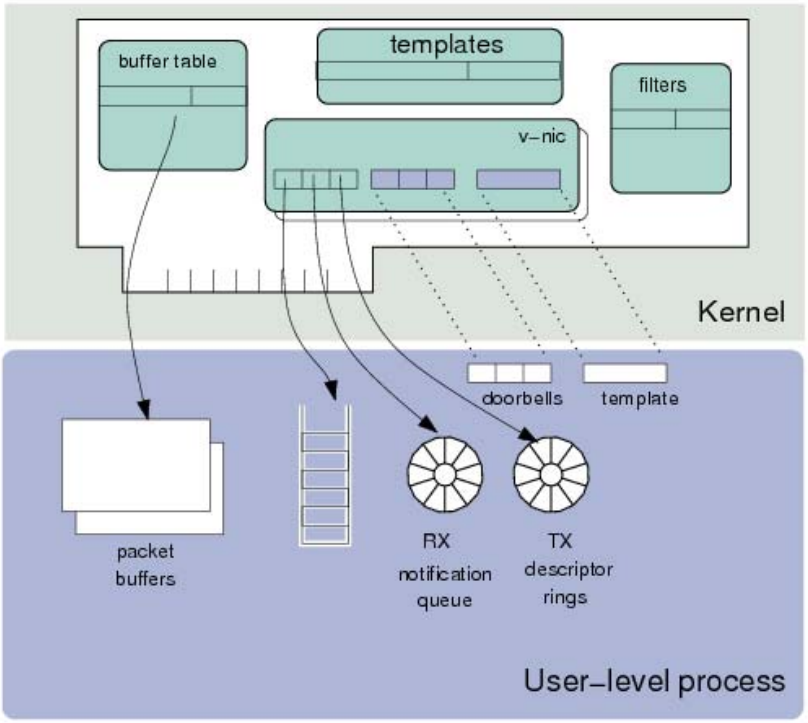

Figure 5: Template resources attached to a v-nic.

A template (Figure 5) is an MTU sized buffer which exists on the network interface and which can be mapped to and owned by host software (for example, a user-space endpoint). Software may access the template as a scratch pad, though typically will write data which is intended for transmission by the ASIC. Software can also form descriptors which refer to the message template as a source of transmission, just as a descriptor would normally refer to regions of host memory. The ASIC when servicing a given DMA descriptor ring will process descriptors in the order by which they were posted, pulling data from the host memory buffers or the ASIC templates as required.

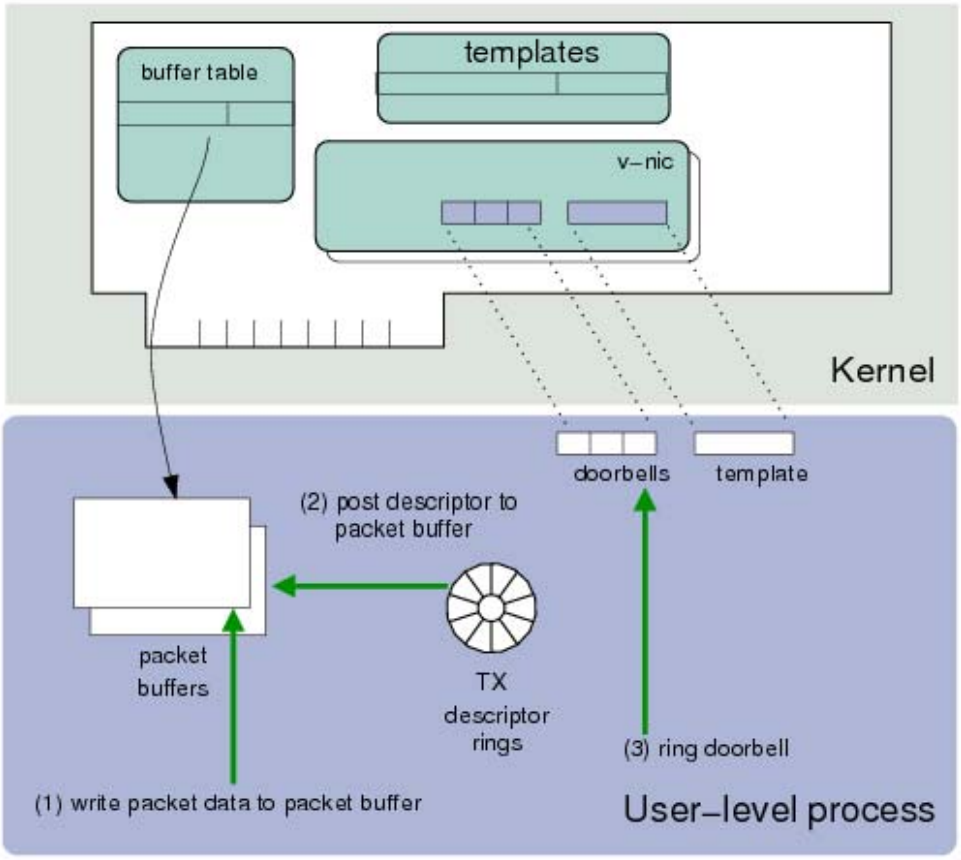

Figure 6: DMA transmission from host resident packet buffers.

Figure 6 shows the steps required to initiate transmission of a frame using conventional DMA. Firstly the packet to be transmitted is constructed in packet buffers which are mapped both into the address space of the process and IO mapped through to the NIC. Secondly a descriptor is posted to the TX descriptor ring. This entry is a pointer to the packet buffer in a private address space valid only between this process and the buffers

on the NIC which have been allocated to the descriptor ring. Thirdly a doorbell is written through a non-cached memory mapping onto the NIC. The doorbell indicates both the particular TX ring which requires attention and may include other information such as the first descriptor(s) on the ring, avoiding avoids the requirement for the ASIC to pull a descriptor from host memory when transmission has been requested. The NIC schedules the TX ring and if necessary reads the relevant descriptors. The NIC then reads the packet buffers to be transmitted using DMA.

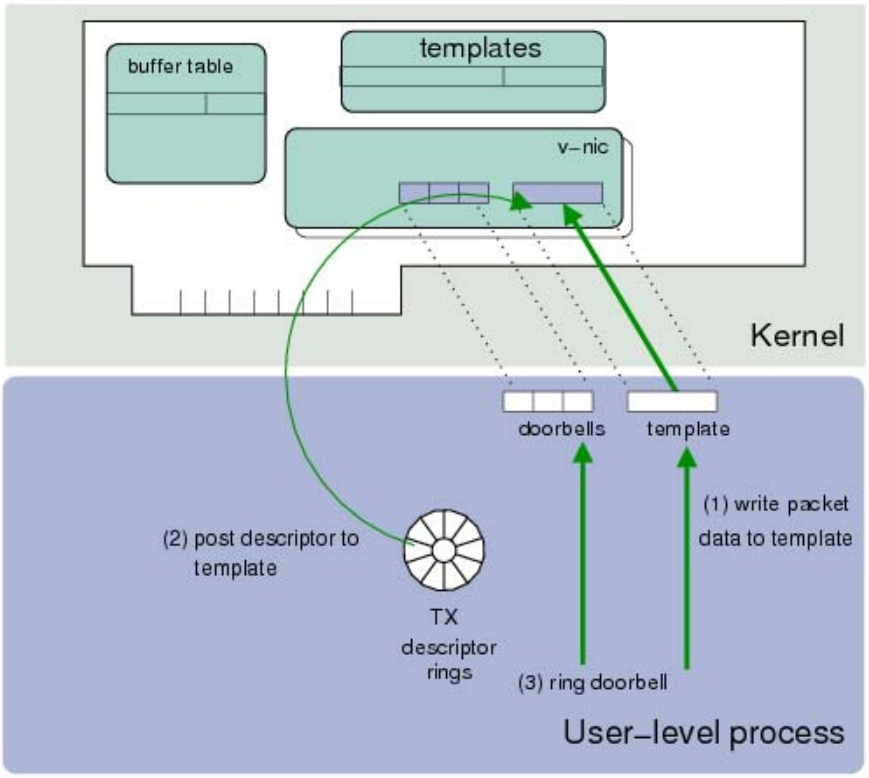

Figure 7: PIO transmission using NIC resident templates.

Figure 7 shows frame transmission using NIC resident templates. Firstly PIO (CPU load-store operations) is used to transfer packet data to the ASIC template which is memory mapped into the process address space. Secondly a descriptor is posted to the TX descriptor ring. This descriptor ring entry references the template (which must have been associated with the TX ring). Thirdly a doorbell is written just as in the DMA case. When

the NIC processes the TX descriptor ring, packet data is accessed from the template just as if it were a host resident packet buffer. In both PIO and DMA cases, packet transmission is indicated via an event notification, which also indicates that either the packet buffer or template is available for re-use. Descriptors to templates may be interleaved with descriptors to packet buffers; the wire order of frames is given by the order by which the respective descriptors have been posted onto the TX ring.

The ASIC templates can be used for a number of other host software performance optimizations. For example, if some or all of the packet data is known in advance of the required time to transmit, then host software can push this data in advance to the template. At the time of transmission, only the final portions of packet data need be transferred over the PCIe bus on the time critical path.

Also, following a transmission, it is only required to push the packet differences to the template for a subsequent transmission, again resulting in a reduction of data required to traverse the IO bus and latency savings.

If an application is required to perform a unicast fan-out function, where a single unicast message must be delivered to a number of network endpoints, it is possible to use the template mechanism. Packet data is written once to the template and subsequent packets are transmitted by updating just the packet headers.

Templates are supported at the Solarflare user-space virtual interface (via the ef_vi API) and are currently used by the OpenOnload library for low-latency PIO transmission of packets. See document SF-104474-CD for more details.

The OpenOnload extensions library is also enhanced to expose Template operations, enabling applications accessing the network at the POSIX socket abstraction level to take also advantage of some[2] of the features.

Timestamping

The 91xx ASIC will timestamp every packet on ingress or egress at the Ethernet MAC interface. Solarflare 71xx and 73xx NICs are fitted with a temperature compensated oscillator (TCXO) which enables these timestamps to be taken with high precision. Timestamp information is carried through the 91xx ASIC data-path as meta-data, through the internal switch fabric through to the micro-engines which are responsible for packet dispatch and notification reporting. The timestamp information may be presented to host software either conventionally, as part of the descriptor completion events, or alternatively by insertion into the Ethernet frame itself.

This timestamp feature is used with the Solarflare enhanced IEEE 1588 software daemon to synchronize the NIC oscillator to a network master clock with high precision. As when used with the older generation Solarflare ASICs, there is driver support to also discipline

[2] The re-use of a transmitted template is not possible with the current Onload extension API.

the server's own oscillator to the network disciplined NIC oscillator, thereby enabling accurate software time-stamping. Of course the system oscillator has a much lower precision than the Solarflare TXCO and so for many situations hardware based timestamps are preferable.

The combination of the precise time stamping and packet replication features of the 91xx ASICs is very useful when used in conjunction with the SolarCapture application. Here packets which arrive at a host and are destined for application processing can be time-stamped in hardware and replicated, one copy being delivered to the application for processing, another copy being captured by the SolarCapture application. Rather than connecting a physical appliance, or configuring a SPAN-port on a switch, every server in the data-center can be provisioned as a capture appliance, right at the point that application processing is taking place.

Virtualisation

The 91xx ASIC has an improved model of virtualization in that there is no longer a hard distinction between drivers attaching to physical or virtual functions and that there is no longer a requirement for a *master* driver in the system. This property means that all resource allocation and the entire ASIC control plane is managed by the ASIC itself without there being any requirement for communication between device drivers and thereby making it very easy to support multiple driver stacks which may include virtualized driver stacks.

Because each driver can request flow-filtering and other hardware resources directly from the ASIC, it now becomes possible for example shown in Figure 8, to run Open Onload within a guest operating system in a virtualized environment. Each guest OS is completely independent from the other and received dedicated hardware resources to directly access the network. Using Open Onload in this manner, both the hypervisor and the operating system are bypassed, enabling ultra high-performance while maintaining the manageability of a virtualized environment.

When used in this manner, network flows can be processed by OpenOnload within a guest with only slight performance degradation compared with running in a bare-metal configuration. The switching capability of the ASIC allows broadcast/multicast traffic to be replicated where necessary for the guest operating systems.

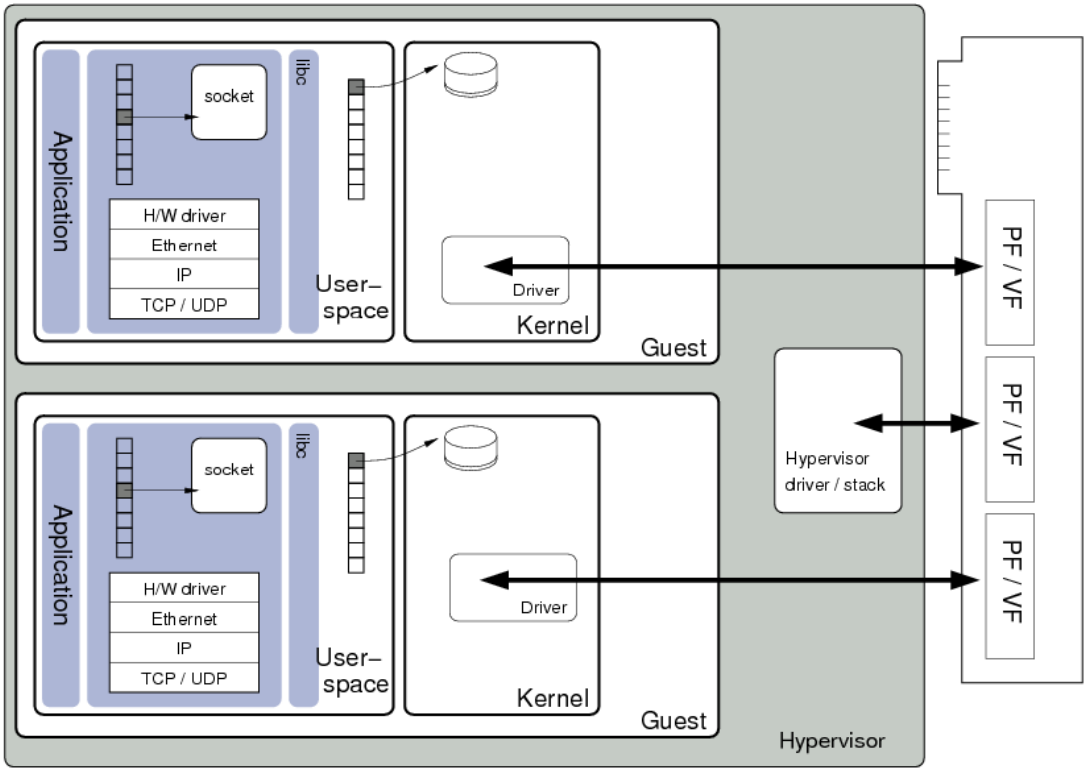

Figure 8: Support for multiple independent driver stacks in a virtualized environment.

In Summary

The 91xx family of ASICs represents a significant step forward with respect to performance and feature capabilities. This whitepaper has described some of the capabilities of the ASIC with emphasis on high-performance financial services applications. There is however broad applicability for the device in many other verticals such as scientific computing, enterprise virtualization and storage.

About the Authors

Steve Pope holds a Ph.D. in Networks and Operating Systems from the University of Cambridge. He is currently CTO of Solarflare, responsible for technology direction, system architecture and intellectual property.

David Riddoch is the Chief Software Architect of Solarflare, which he co-founded in July 2002. Previously, David was the architect and lead developer of the software for the CLAN high performance network project at AT&T Laboratories Cambridge. David holds a first class degree in computer science and a Ph.D. in high performance networking from the University of Cambridge.