

Intelligent Network Applications for 10Gbps and Beyond

Traditionally, network switches and routers have acted solely on packet header information (Layer 2-4 in the OSI model), oblivious to the payload being carried. However, with continued attacks on existing protocols and new risks with emerging applications, intelligent networking requires the processing of the contents of network packets (Layer 4-7) as well as the headers. These intelligent networking functions can appear in standalone “bump-in-the-wire” appliances (i.e., the device is inserted in the link between two switches or routers), or increasingly common, in blades inside enterprise and service provider switches. In either case, the applications are similar and include:

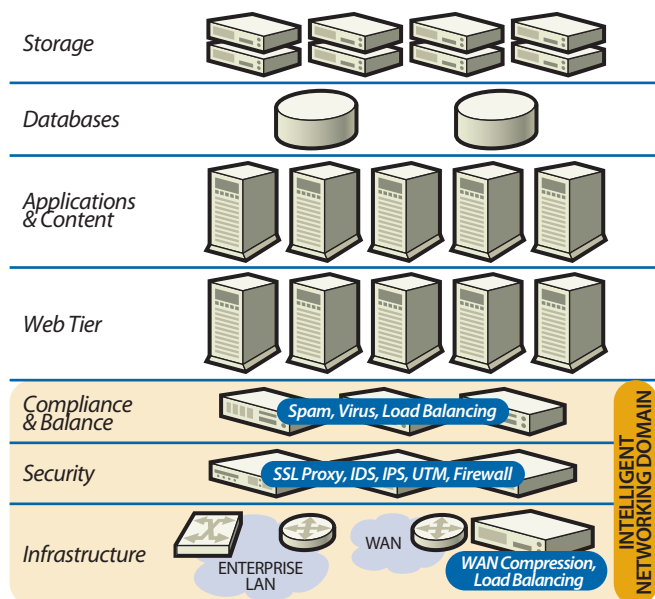
- Stateful firewalls and Unified Threat Management (UTM);
- Application and web firewalls;
- Optimizers, such as load balancers;
- WAN optimizer/compressors;
- Compliance products that monitor network content;
- Intrusion Detection and Prevention (IDS/IPS).

THE NEED

The growth in rich-media applications, such as YouTube and Skype™, the emergence of web-based hosted applications in enterprises and between enterprises has caused a massive growth in data center capacity both in service providers and enterprises, all while the greater availability of affordable broadband connectivity has enabled enterprises to achieve significant savings by consolidating data centers. These new “macro” data centers need a new class of intelligent networking element capable of scaling to 10Gbps and beyond, and of handling the millions of simultaneous flows present on these high-capacity trunks.

(continued)

Figure 1. The need for intelligent network flow processing is growing rapidly in high-speed Ethernet and IP networks, along with the increasing number of network and security applications embedded within the network.



EXECUTIVE OVERVIEW

The Internet has become the backbone of commerce, entertainment and information exchange. It is still evolving and new business models, usage models and technologies are deployed daily.

New technologies that are having an impact include Web 2.0, hosted applications, rich-media sites and real-time communication. While these applications drive a continued need for increased bandwidth, greater reliability and lower latency from the network, simultaneously security requirements dictate that the network must have significantly greater application awareness.

This whitepaper focuses on the requirements for next-generation network appliances in intelligently processing traffic for the growing real-time, interactive and transactional applications. We define the term “intelligent networking” to encompass the needs of these applications.

THE SIMPLEST INTELLIGENT NETWORKING APPLICATION

Intelligent networking functions are typically deployed as a “bump-in-the-wire”: network traffic arrives on one side, is processed and then emanates from the other. (Usually the arrangement is symmetrical with traffic flow in the opposite direction, too.) In the case of embedded cards within network switches while the data arrives from and is delivered to the chassis backplane, logically, the same configuration holds.

In Figure 2, we illustrate the architecture of a typical “appliance.” It shows a network adapter card with two (10Gbps) ports installed in a typical dual core motherboard. If this is deployed in a “bump-in-the-wire” configuration, the memory bandwidth required to get the packets in and out of memory is 40Gbps. This is a substantial portion of the memory bandwidth in current motherboard designs, and at this point NO processing of the packet data has yet occurred.

Consider a simple proxy service, in which software on the processor will terminate one TCP connection, manipulate the data and create a new TCP connection out the other side. Even with benign assumptions (1460 byte packets, TCP checksum offload and zero copy), we find that the processing requirement is equivalent to five 2Ghz IA cores and the memory bandwidth required up to 250Gbps.

Furthermore, there is also a well-known and documented problem as we increase the number of flows being processed. For each TCP flow, software will maintain about 700B (bytes) of flow state; with a million flows we have 700MB. The size of this data structure and the manner in which it is used largely nullifies any of the usual cache benefits.

What requirements can be drawn from this?

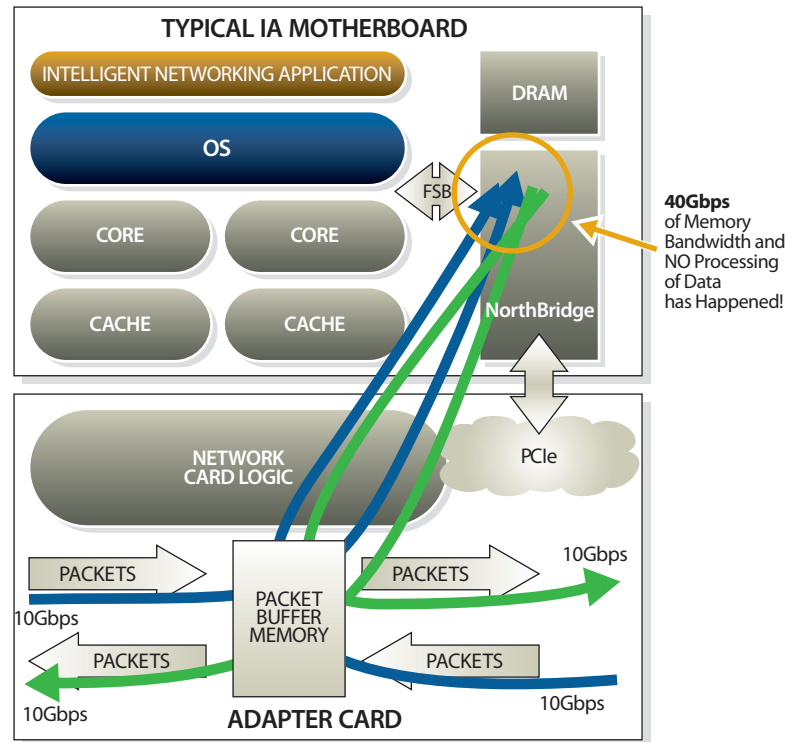
- In order to process packets at the required rates, functionality must be offloaded from the main application processors. However, the offload required is application-specific, a programmable, flexible offload function is needed.
- Memory and cache management must be designed with the intelligent network application usage in mind.

HETEROGENEOUS INTEL ARCHITECTURE (IA) AND IXP SYSTEMS

The industry-standard IA platform continues to deliver the best price/performance for running general purpose application code. Cost-effective networking appliances in the future will need to exploit the rapid technological pace and volume achieved in the IA platform to deliver effective solutions. However, as a packet handling engine, the IA has a number of issues noted above:

- Large numbers of independent flows, composed of many small packets, result in severely degraded cache performance;
- Manipulation of state in the system memory generates significant amounts of memory traffic during which the processor is stalled, effectively doing no work;

Figure 2. Network bandwidth-to-host-CPU ratio in typical appliance architectures.



- The flexibility and programmability of the IO architecture results in a significant amount of processor cycles dedicated solely to moving data in and out of the system memory; cycles which are not available to application code.

The IXP network processor has been designed to address these specific issues:

- It features a highly parallel architecture incorporating specifically designed packet processing RISC microengines (ME) dimensioned to handle the expected number of packet flows;
- The parallelism in the architecture, combined with multiple threads per ME, allows the hardware to continue to perform useful work, even while many threads are awaiting responses from the system memory;
- The highly stylized handling of packet flows is supported in hardware with specific hardware queue mechanisms minimizing the software overhead.

However, the IXP is not a great general-purpose computing platform. There is no straightforward way to abstract the capabilities of the IXP into a common programming language that would make the task of programming it easy for application programmers.

An effective system solution can thus be achieved through a combination of the IXP being used for packet processing and data movement, with the IA focused on the execution of the code of the network application. This combination concisely addresses the requirements as illustrated in Figure 3, our example proxy application.

In general, the industry is moving to an architecture in which the IA is the application workhorse, with common, though specialized, functions implemented in programmable

HW accelerators. Graphics Processing Units (GPUs) which are more effective than CPUs at complex graphics rendering, are the common and well-established example, while crypto, Java™ and XML co-processors are emerging. The IXP will take its place amongst these as an intelligent networking coprocessor. Besides offering increased performance through specialization, such solutions also offer much lower power consumption than CPU-only designs. This is increasingly important as devices continue to shrink and on-chip power density becomes a limiting factor.

The efficiency, capabilities and programmability of these heterogeneous multiprocessor solutions is dictated by the manner in which the different processors can communicate and share memory. If the processors can share some cache-coherent memory, and with network IO tightly coupled into this memory system, the latency for packets arriving can be minimized, undergoing packet-level processing by the IXP, application processing by the IA and, finally, departing the system.

WHY QUICK PATH INTERCONNECT (QPI)?

The IA platform has been a major success because of its openness to the addition of third-party hardware. This has been achieved through a series of standard-defined interfaces, USB, PCI, AGP, and, most recently, PCI Express (PCIe).

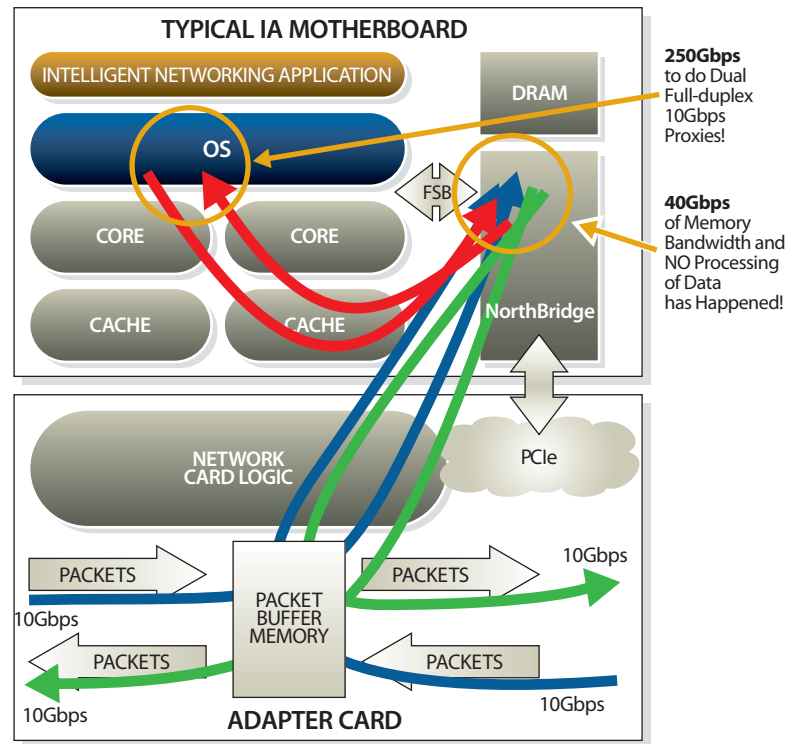
PCIe is widely used to build heterogeneous multiprocessors. For example, the addition of Netronome Flow Engine (NFE) cards to an industry-standard server provides a very flexible combination of IA and IXP processing for network appliances. However, the use of PCIe to build a heterogeneous multiprocessor has its limitations:

- The processors are forced into a message-passing paradigm, where any data that might be required by the IA is copied into its memory, whether it is ever used or not;
- Similarly maintaining synchronized state between the IXP and IA involves many message and memory operations across the PCIe links.

Today, these architectures are capable of supporting several Gbps of throughput, but will not scale for future requirements. The solution to this problem is well known: build a more tightly coupled system based on NUMA principles in which the processors share memory that can be maintained by hardware as cache-coherent, but maintain locality of reference. This allows system scalability far and above shared bus designs. To date, most heterogeneous system designs for networking appliances have used RISC processors for the simple reason that the commercial environment exists to support such designs. The appropriate memory bus interface is clearly defined and supported, and the design process is enabled through tools and libraries that make the design straightforward.

With the introduction of QPI, Intel® has defined an industry standard and support program that enables the design of the required tightly coupled heterogeneous IA and IXP multiprocessors that will enable the next generation of networking applications and scale well beyond 10Gbps.

Figure 3. An example proxy application on Netronome-enabled hardware.



NETRONOME FLOW MANAGEMENT & VIRTUALIZATION AS KEY ENABLER

While the IA/IXP heterogeneous multiprocessor enables great efficiency in handling packets flows, it is a requirement of the system software to then provide the interfaces and runtime implementation that enables network applications to realize the performance improvements available.

The challenge is to provide an environment that is functionally equivalent to the familiar application programming model, but exploits the underlying IA and IXP capabilities. In the case of embedded network applications, the programming environment is primarily Linux®, and the underlying capabilities of the platform are exposed to the application programmer through the components of the Netronome Flow Manager (NFM) and virtualization software.

The NFM is composed of Linux libraries, drivers and server processes which encapsulate the complexity of IA/IXP interaction. The NFM offers capabilities in a variety of ways. For example, the NFM driver components offer simple legacy interfaces, such as virtual NICs, together with capabilities such as TCP acceleration offload and zero copy, while the NFM library interface provides highly programmable event driven callbacks.

Where required, the NFM provides a virtualization layer to enable the support of multiple Linux instances for performance, security and fault isolation.

Through the support of Linux, the platform of choice for network application programmers, and offering acceleration through existing interfaces, legacy applications are immediately able to exploit the improved platform performance, while in the future, moving to the higher-level NFM library and shared memory interfaces offers even more performance headroom.

The NFM plays a critical role in assigning processors and configuring the OS scheduler(s) to operate optimally in response to the various networking events that are conveyed from the IXP to the IA for processing. Such an approach is possible in intelligent networking applications where the applications and workloads can be highly characterized, and the generality and flexibility of general purpose dynamic schedulers can be sacrificed for increased performance for these specific applications.

CURRENT NETRONOME SOLUTIONS

Netronome delivers these benefits in current products, as described in this brief overview.

The IXP devices used by Netronome contain 16 MEs (microengines) or small cores. Each device has three DRAM channels and four QDR SRAM channels leading to significant memory bandwidth. Each ME in turn has eight HW threads.

The MEs do not have caches. Instead, there is a significant number of registers for local variables with threads that can issue reads/writes to the memories with HW handing over control to the next ready-to-run thread. In this manner, memory latency can be hidden while maintaining performance linearity over the complete memory space. This means that the aforementioned TCP proxy can be built to linearly scale over millions of flows.

Netronome uses these devices in a series of Accelerator PCIe cards for IA/X86 systems with Netronome-developed specialized SW and FPGAs to allow for tight coupling of IA and IXPs. This architecture is illustrated in Figure 4.

A key aspect is the high-performance, low latency message-passing infrastructure that Netronome has built between the ME cores (and threads) and the IA cores allowing for cooperative processing of Network data. As can be seen in Figure 4, this is physically achieved by adding an FPGA on one of the SRAM ports (QDR) to achieve the interconnect.

An extensive acceleration software infrastructure and APIs have been built to allow customers to use ME core acceleration without the need for programming the ME cores. The software infrastructure referred to is called the Netronome Flow Manager (NFM) and more information can be obtained from Netronome's website. If a customer desires to program the MEs, a C compiler and Assembler is available along with a message-passing library and zero copy APIs to facilitate communication with the IA cores.

In conclusion, Netronome has built a complete Intelligent Networking acceleration suite for cooperative processing of Network data that scales to 10Gbps. In other words, a heterogeneous architecture platform.

Niel Viljoen, is the CEO and a co-founder of Netronome Systems.

Derek McAuley is the Chief Scientist of Netronome Systems.

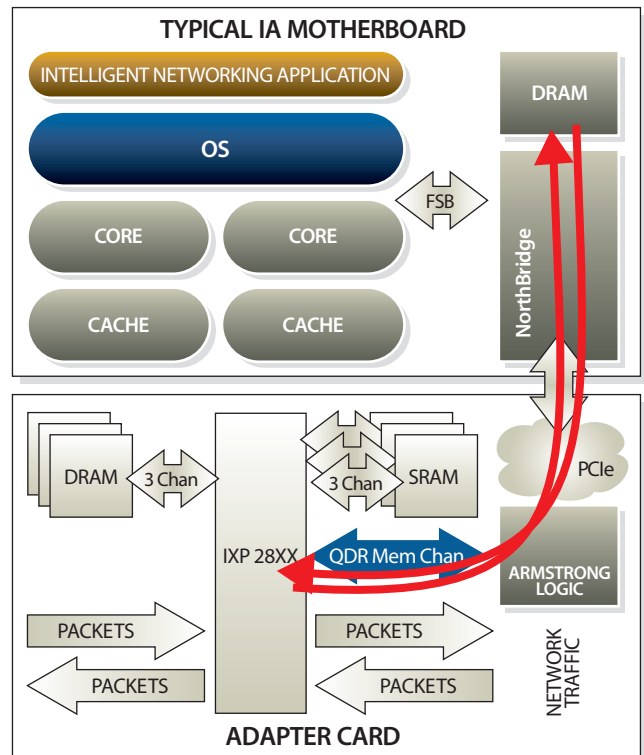


Figure 4. Netronome NFE PCIe product architecture.

TM Netronome, and "Total Performance. Total Control." are trademarks of Netronome Systems, Inc.

All other trademarks are the property of their respective owners.

© 2007 Netronome Systems, Inc. All rights reserved. Specifications are subject to change without notice.



Total Performance. Total Control.™

Netronome has operations in:
USA (Pittsburgh [HQ], Santa Clara & Boston),
UK (Cambridge),
Malaysia (Penang),
South Africa (Centurion) and
China (Shenzhen, Hong Kong)
info@netronome.com
 +1 877 NETRO A-Z
www.netronome.com