

The Evolution to Network Flow Processing

Enabling the deployment of Intelligent Networks at 10G and beyond

In a typical enterprise or carrier network environment, network traffic is sourced by many users and applications, and, as a result, tends to be uncorrelated. This aggregated multiplexed packet stream may consist of packets belonging to real-time applications, such as video conferencing and VoIP, and others belonging to non-real-time applications, like Email and text messaging. In order to achieve the level of performance required, the network equipment needs to first classify this packet stream into flows before they are presented to the multi-core CPU sub-system. At 10Gbps line rate, the processing sub-system is required to process, inspect, detect potential threats, and apply specific security rules, all to millions of flows.

To enable service providers and network equipment vendors to meet the high-performance challenge, a new multi-chip, multi-core heterogeneous processor architecture is required. In this architecture, a Network Flow multi-core Processor (NFP), optimized for L2-L7 processing, works in conjunction with one or more general-purpose multi-core CPUs. Designs based on this architecture will allow equipment providers to deliver high-performance, flexible and field-programmable systems that enable service providers to generate more revenue per user over a longer product life cycle.

THE NETWORK IS EVOLVING

For years, service providers, IT professionals, networking vendors, and forecasters have been predicting the convergence of services and of the underlying network. Initially, it was the convergence of TDM voice and packet data networks into one unified network. Later, with the advent of broadband, wireless and video, this convergence has evolved into a quad-play offering (Data, Voice, Video and Wireless) by service providers, and a unified network using the Internet Protocol (IP), Ethernet transport, and the Session Initiation Protocol (SIP), along with security measures to ensure that users are authenticated and the data remains intact. (Data refers to any digitized service, including voice and video.) As a result of these trends, the level of intelligence required in all network nodes is increasing, with some nodes requiring vastly increased programmable intelligence to support complex infrastructure applications while operating at ever-increasing line rates.

A high level view of the converged network topology is depicted in Figure 1. The enterprise accesses the service provider network over a WAN connection to the service provider's Point of Presence (PoP) in the local office or a co-location space, which is usually an access aggregation equipment for copper, cable, fiber, and wireless access. Moving upstream toward the core of the network, an edge device (e.g. router, multi-service provisioning platform,

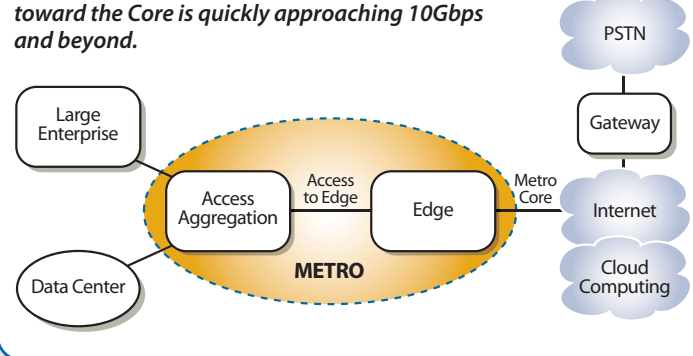
EXECUTIVE OVERVIEW

Network traffic in both enterprise and carrier networks continues to rise, driving the bandwidth requirements and line rates to 10Gbps today, and expecting to grow to 40Gbps and beyond in a few years. With the need for application awareness, content inspection, and security processing, the amount of processing power within the network infrastructure at these ever-increasing line rates grows exponentially.

To enable service providers and network equipment vendors to meet the high-performance challenge, a new multi-chip, multi-core heterogeneous processor architecture is required. In this architecture, a Network Flow multi-core Processor (NFP), optimized for L2-L7 processing, works in conjunction with one or more general-purpose multi-core CPUs. Designs based on this architecture will allow equipment providers to deliver high-performance, flexible and field-programmable systems that enable service providers to generate more revenue per user over a longer product life cycle.

This paper discusses this new class of flow-processing architecture and its role in the new heterogeneous multi-processing architecture.

Figure 1. High-level converged network topology. Line-speed performance from the Access Aggregation toward the Core is quickly approaching 10Gbps and beyond.



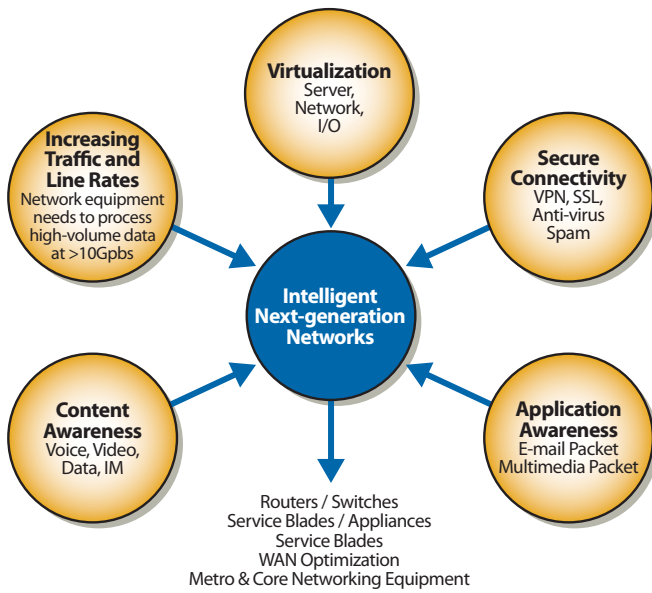


Figure 2. There are many forces contributing to the need for next-generation intelligent networks. Such networks need to be content- and application-aware, secure, and operate at a sustainable 10Gbps and beyond.

broadband access server) is used for service definition and aggregation, including service prioritization, QoS and security. A high-speed core transport network leads to the Internet, the emerging cloud computing, and to the PSTN through a media gateway.

Figure 2 depicts the major forces behind the intelligent next-generation networks, for enterprise and carrier networks.

Network Traffic is growing dramatically driven by consumer broadband, corporate traffic, and newer IP-based services, such as IP Video, and IPTV. The result is that network traffic in the enterprise and carrier network has exploded in recent years. With the explosion of Internet-based services, the bandwidth requirements at every point in the network (and, as a result, line

speeds) have been quadrupling every five years. This staggering rate of growth is even higher in enterprise networks and data centers. Figure 3 depicts the migration to 10Gbps and beyond.

Content-awareness is another force behind the need for network intelligence, as digitized network traffic represents not only data, but also voice, video, IM and other services. An intelligent network needs to be content-aware to best implement traffic filtering, prioritization and QoS, and help service providers meet their committed SLAs.

In addition, the intelligent network needs to be aware of the application, such as Email, Web services and multi-media services. The feature is also needed for application-level switching for load balancers and application-level security appliances.

Security is another major force fueling the need for intelligent networks. Increasing security requirements are driving the need for high-performance deep packet inspection and security processing. Network security has always been important, but recent trends, such as Software as A Service (SAS) and Web 2.0 applications, result in more “sensitive” information being carried over the network, and this needs to be secured. Security requirements are relevant to residential, enterprise and carrier networks alike. Some of the key market drivers are:

1. Increase in number of attacks
2. The rise in volume of un-productive or malicious traffic
3. Productivity-crippling security-related downtime
4. Regulatory or compliance repercussions for not protecting electronic assets
5. Loss of data — fueling investment in data encryption, intrusion prevention, and access control.

Finally, virtualization trends are fueling higher-performance, consolidation, lower power, and efficiency. The data center has been experiencing consolidation with stronger needs for better control for security and management. Virtualization of the servers in the data center has been on the rise in order to increase effective utilization and sharing across network resources,

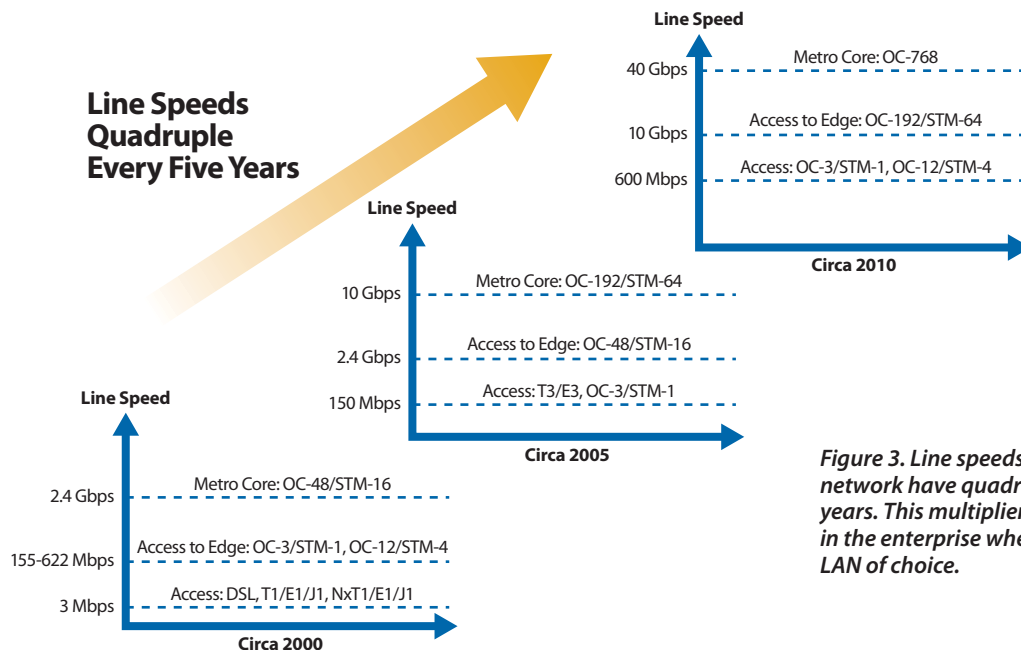


Figure 3. Line speeds in the carrier network have quadrupled every five years. This multiplier is even higher in the enterprise where Ethernet is the LAN of choice.

while reducing the Total Cost of Ownership (TCO). The next data center resources to experience virtualization are the underlying network (LAN and SAN) and the network I/O.

To support today's high-performance servers and network security appliances, IT managers and security architects are increasingly using virtualization to maximize the utilization of valuable server resources, including CPU cycles, server rack space, power, and cooling.

MEETING THE REQUIREMENTS

Meeting the challenges outlined above warrants a new approach to the development of the high-performance systems required by the intelligent network. Such systems need to be capable of handling traffic at L2-L7, and performing intelligent processing on the traffic at any level, while sustaining throughputs of 10Gbps and higher. The combination of Network Flow Processor(s) with general-purpose multi-core CPUs provides a platform that can satisfy these requirements. The NFP is a specialized, multi-core device that handles lower-layer (L2-L4) processing and accelerates higher-layer (L4-L7) processing, and is closely coupled to the general-purpose CPU(s) in the system. This resulting multi-chip heterogeneous multi-core processing architecture is required for 10G systems and beyond.

Lower-layer (L2-L4) Packet Processing

As previously mentioned, the NFP needs to handle L2-L4 processing, but at very high rate. These packet processing functions can be divided into ingress and egress processing:

Ingress processing encompasses all operations performed for incoming packets:

- Error detection and correction
- Security verification, including blocking incoming flows, if needed
- De-encapsulation and de-tunneling — e.g. VPNs and IP over MPLS
- Flow classification
- Traffic measurement and policing
- Address lookup and packet forwarding
- Header modifications and NFP splicing
- Packet reassembly and flow termination
- Forwarding, queuing, and scheduling

Egress processing encompasses all operations performed for outgoing packets:

- Addition of error detection code
- Address lookup and packet forwarding
- Packet segmentation
- Traffic shaping
- Timing, scheduling, queuing, and buffering
- Header modification
- Encapsulation and tunneling
- Output security processing

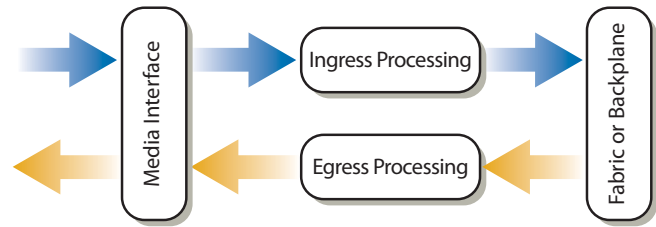


Figure 4. Ingress processing handles incoming packets/flows from a media interface to a second media interface, a switching fabric, or a backplane. Egress processing is the reverse process.

Although the functions outlined above can be executed using a general-purpose CPU, the speed under which it has to be executed requires the use of dedicated packet processing engines, as found in the NFP.

Optimizing Packet Processing

The use of flow classification is one way to optimize packet processing performance. Instead of dealing with incoming traffic on a packet-by-packet basis, the NFP needs to classify the traffic into flows. For example, an on-chip function is required to perform a hash function on the incoming packet header and, as a result, determine a flow designation. This function can also maintain the state information and action list for the flow to be performed on subsequent packets belonging to that same flow.

Another way to optimize packet processing performance is through parallel processing. As most of the packets arriving at a network device (within a short time window) belong to different flows, they then must be processed independently of each other. The NFP exploits this parallelism by using multiple packet processing engines running in parallel with each PPE handling a different packet. This mechanism helps to considerably increase the processing budget per packet.

An additional way to increase the processing budget per packet is to use multiple PPEs in a pipeline with each PPE performing one specific task on the packet. In this case, the packet processing task is decomposed into subtasks with each pipeline stage handling one subtask. Some of the pipeline stages may be fixed-function hardware, while others may be programmable engines.

It is best to use a combination of all three optimization techniques listed above (flow classification, parallel, and pipeline processing) simultaneously for best utilization of on-chip resources and to meet stringent performance requirements.

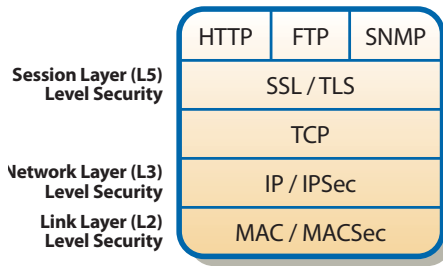
Higher-layer (L4-L7) Acceleration, including Security

As we focus on high-speed networks, the need to perform deep packet inspection at line rate requires the host CPU to use the assistance of external acceleration devices. One such example requiring external acceleration is security protocols processing. In this case, the processing architecture needs to support many levels of security protocols.

MACSec (or IEEE802.1ae) is the security protocol specified by IEEE802 committee for Ethernet layer 2 (MAC layer) level security. IPsec is the network layer (L3) level security and is most used for remote connectivity for telecommuting and branch-to-branch connectivity over a VPN connection. In addition, SSL or TLS — the Secure Sockets Layer (SSL) and the Transport Layer Security (TLS) — provide session layer (L5) level security and is primarily used for VPN and for Web access.

As shown in Figure 5, more than one layer of security processing can be active at the same time. As a result, the NFP needs to be able to support running SSL over MACSec, or IPsec over MACSec.

Figure 5.
The various layers of security protocols are shown. The NFP should be able to process SSL over MACSec or IPsec over MACSec packets.



CONFIGURATION FLEXIBILITY & PERFORMANCE REQUIREMENTS

In-line or Look-aside

The system architecture should be flexible to allow a system designer to connect the host CPU to the NFP in either an in-line or look-aside mode. In the look-aside mode, the general-purpose CPU is the master and can choose the type of tasks that need to be accelerated by an attached co-processor, like the NFP.

When operating in-line, the NFP is the master — it pre-processes data, and send only relevant packets to the CPU. This mode of operation can take advantage of the “cut-through” capabilities of the NFP, allowing an ingress packet to be forwarded to the appropriate egress port without the intervention of the general-purpose CPU.

The NFP needs to have the appropriate interfaces to support both modes. The on-chip network media and the fabric interfaces enable various system configurations where the NFP can reside on an acceleration card performing various ingress processing tasks for the host. As a result, the NFP can reside between the network port and the switching fabric, between the switching fabric and an output network port, or simply attach to the switching fabric.

The Need for Multi-threading

The NFP needs to employ multiple programmable Packet Processing Engine (PPE) cores capable of handling low-level performance-intensive tasks, such as ingress packet processing, header processing, table lookup, packet forwarding, and egress packet processing. The PPE core needs to employ a pipelined RISC multi-threaded architecture operating at high speed by having operands reside in local general-purpose registers. As the PPE core supports many threads, the PPE switches context to a new thread when a thread stalls for external memory access.

The protocol processing tasks performed by a packet processing thread are decomposed into modular functional blocks which are isolated from the data flow mechanisms between them to focus only on protocol-specific packet processing tasks. This also allows reuse of the functional blocks code. Each packet processing thread caches the packet header and descriptor in local memory or registers for use by the different functional blocks.

The NFP needs to utilize an application programming model that optimizes performance, enhances code re-use, and

allows applications to easily scale to higher data rates. It is best to separate hardware-specific code, such as receive, transmit, scheduling, and queue management, from network protocol-specific packet processing code, with the two codes running on different packet processing engines. Such separation enhances the portability and reuse of the packet processing code. Packet processing threads use a run-to-completion model for execution where a thread executes a series of tasks on a single packet before moving on to the next packet. Multiple packet processing threads across several packet processing engines run in parallel to scale-up the performance. This model is flexible and allows for easy addition of more functionality in the factory or in the field.

Memory Bandwidth Requirements

The NFP needs to support a highly optimized and flexible packet buffer architecture that enables applications to run at very high data rates. Such architecture should provide efficient mechanisms to allocate and free buffers, access both packet descriptor and packet data information, and add / delete packet headers. The architecture should also support packet queuing, multi-cast, and jumbo packets.

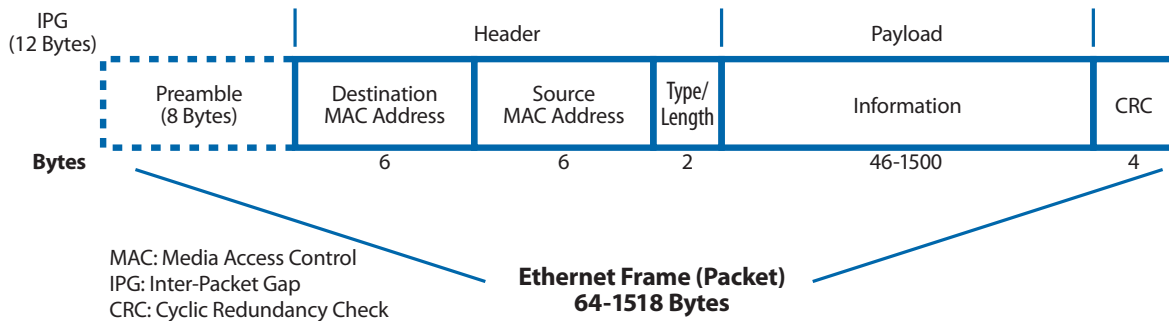
In order to move packets around and meet the 10G system requirements, the media interfaces, internal busses, and memory controllers need to operate at a very high bandwidth. A typical networking application using the store-and-forward model encounters four data moves:

1. Move packet from media interface into memory
2. Read the packet headers from memory
3. Process packet headers and write them to memory
4. Move packet from memory to the media interface for transmit

This implies that the memory (typically DDR2/DDR3 SDRAM) bandwidth needs to be four times (4x) the line rate to satisfy packet movement’s requirement. As a result, SDRAM bandwidth needs to be 40Gbps for a 10Gbps line rate. In fact, the memory bandwidth needs to be even higher (along the order of 80Gbps) to accommodate other types of accesses to memory, such as the need to read data structures (such as tables) and per-packet state information. In addition, all chip resources including busses, on-chip memory, and registers, need to be designed with such very high bandwidth in mind. These on-chip resources should also implement ECC (Error Correcting Codes) and or parity throughout, due to the existence of soft errors caused by Alpha particles and Neutron/Cosmic flux.

Performance Calculations

Since most networks use Ethernet frames as the underlying transport, we will use such frames as the basis to compute network performance. Figure 6 depicts a typical Ethernet frame. The frame has a minimum length of 64 bytes and a maximum length of 1518 bytes. The table in figure 6 shows the performance numbers in packets per second (pps) for 1G and 10G Ethernet. A maximum pps figure is usually calculated when all Ethernet frames are of 64-bytes in length, or the minimum size frame. Because of the Inter-Packet Gap (12 bytes) and pre-amble (8 bytes), the minimum packet size becomes 84 bytes. Hence, for a 10G Ethernet, the pps is calculated as follows:



	PPS for 64B	Time to Process 64B (nS)	PPS for 1518B	Time to Process 1518B (nS)
1000Base-T (1G Ethernet)	1,488 M	512	8,234	12,144
1000Base-T (10G Ethernet)	14,881 M	51.2	823.45	1,214

Figure 6. Ethernet frame format depicting a frame of minimum size of 64 byte and a maximum size.

Number of packets per second (PPS) = Line rate / number of bits in a packet, or

$$\frac{10 \times 10^9}{84 \times 8} = 14,881 \text{ million packets per second}$$

In addition, the table also depicts the per-packet time budget required to process a packet. At 10G Ethernet, the NFP has only 51.2 nano seconds to process the entire packet. This short time budget to process each packet was calculated based on a 64-byte packet without the inclusion of the 20-bytes IPG and pre-amble ($64 \times 8 \times 10^{-10}$).

CURRENT SOLUTIONS ARE INADEQUATE

In order to make the system scalable, it is a good practice to keep application-level processing separate from the L2-L7 data plane processing. General-purpose multi-core processors focus on application processing, while the NFP focuses on data plane processing.

Current solutions, such as network processors and communication processors, are inadequate in meeting tomorrow's challenge of 10Gbps and beyond, especially when there is an additional need to perform higher-layer processing. Network processors are traditionally L2-L4 processors and usually lack the much-needed high-layers processing acceleration. Communication processors, with on-chip multi-core general-purpose CPUs combined with fixed-function hardware acceleration engines on a single chip, can handle both application and data plane processing but are limited to well below 10Gbps when performing full packet inspection at minimum packet size (Figure 7). With the demand to perform deep packet inspection at line rates, this single chip solution can't keep up with the processing demands. A two-chip solution forming the heterogeneous architecture is needed. This approach also takes

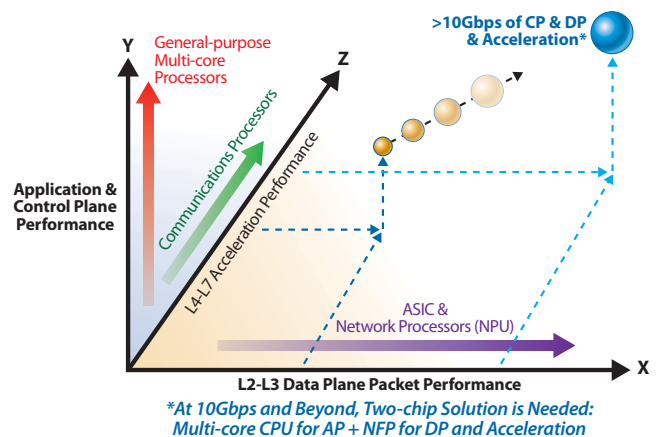


Figure 7. The need to process packets at L2-L7 requires a two-chip heterogeneous multi-processing solution at sustained line rates of 10Gbps and above.

advantage of advances in general-purpose embedded CPU architecture and technology over an evolution roadmap that is very different from that of high-performance programmable packet processing engines. The result is a choice for the system designer to use the best of both worlds.

To further examine this point, one should focus on the need to meet network I/O requirements at various performance and power requirements (Figure 8). At the low-end, an I/O optimized single-core general-purpose processor executing application code can suffice. As performance requirements increase, a multi-core general-purpose CPU can execute I/O-specific software library functions. Fixed-function integrated acceleration functions are integrated with the multi-processor to meet higher performance and power efficiency requirements. At the high-end, an NFP-like device is needed as the intelligent multi-core co-processor to off-load the general-purpose multi-core CPU — again to meet the stringent requirements for performance and power efficiency.

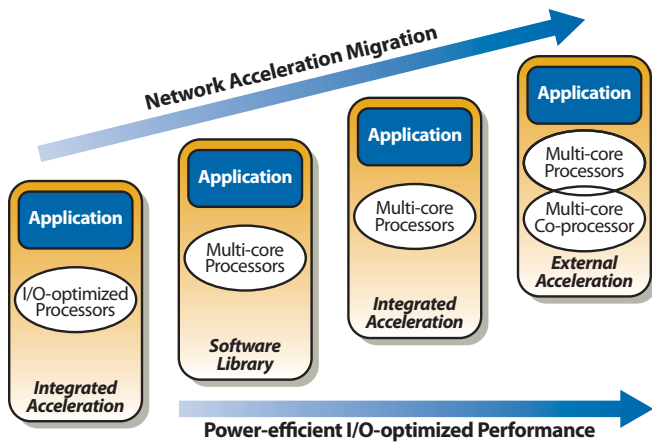


Figure 8. The challenge to meet network I/O varies with performance and power requirements. At the high-end, an external acceleration device, such as the multi-core NFP, is needed in conjunction with the host multi-core CPU.

Cache-based Architecture — Performance Limitations

Many general-purpose, cache-based, multi-core CPUs have tried to fill the high-performance processing gap, but their performance is limited to well below 10Gbps sustained line rate. Cache misses force the CPU cores to starve for memory accesses, where main memory latency is way too slow compared to cache memory. When adding more CPU cores to the mix the problem is actually compounded - resulting in even more cache misses.

Traditional general-purpose CPUs have relied on caches to work around memory latency issues. Cache mechanisms exploit temporal locality (locality in time) and special locality (locality in space) in data access patterns to optimize memory latency times. Traditional caches, however, do not work well for handling high-speed network traffic for the following reasons:

1. Packet data and per-packet state are completely new and different for every packet. As a result, they will need to be moved in to the cache every time a new packet is received.

2. Most packets processed by an intermediate node typically belong to unrelated flows. In fact, the higher the data rate, the lower the probability that the node will process packets belonging to the same flow (or connection) within a small time window.

It should be noted that whereas a memory read cycle of a cache can be less than 100 CPU cycles, a memory read from main memory can be well over 300 cycles when the CPU is operating at 1GHz. For reasons outlined above, external accelerators optimized for data plane processing at high data rates typically do not include a traditional cache.

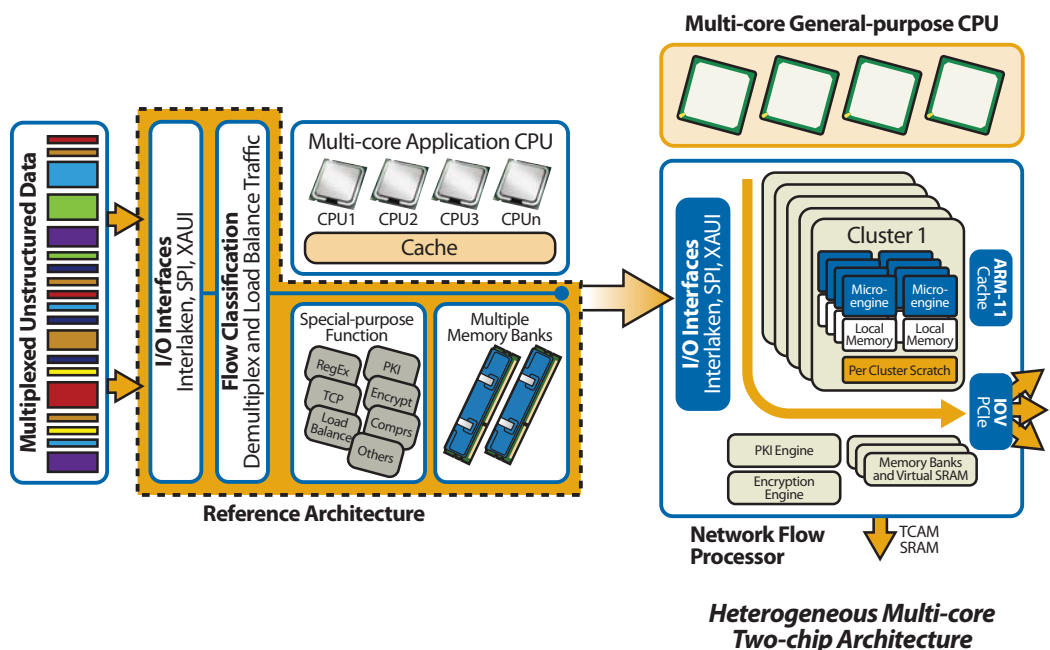
THE NEED FOR A HETEROGENEOUS ARCHITECTURE

If we examine the reference architecture needed at very high line rates, one would find the various processing blocks depicted on the left of figure 9. In the Heterogeneous multi-processing architecture, many of the regular processing tasks that do not work well in heavily cached architectures are mapped into small multi-threaded processors that have large numbers of memory transactions occurring in parallel (Figure 9 on the right). These processors do not have all the additional transistors that are required in the big processors and, therefore, consume less power per instruction and, in addition, map better onto workloads that do not cache well. In the heterogeneous architecture, these processors are combined with traditional multi-core processors that run the bulk of the application code.

In order to make the system scalable, it is a good practice to keep the application processing separate from the L2-L7 data plane processing. General-purpose, multi-core processors focus on application processing, while the NFP focuses on data plane processing.

The NFP integrates multiple specialized processor cores (the PPEs), security-specific processors, and a single general-purpose CPU onto one piece of silicon to meet the requirements of system designs at 10Gbps and above. The on-chip CPU handles tasks related to the NFP chip. Such tasks include bootstrapping, exception handling, diagnostics and debugging.

Figure 9. The reference architecture (left) depicts the need for a multi-core application general-purpose CPU and special-purpose acceleration functions. In the heterogeneous model, a two-chip solution is used — the NFP and a general-purpose CPU. The NFP integrates multiple programmable PPE cores, security processors, and an optimized I/O and memory interfaces.



A high-level programming model for the heterogeneous multi-processing architecture is needed. This needs to take advantage of programming in high-level languages and use open-source tools, while allowing the system designer to customize the PPE for best power/performance combination.

A networking application can be compiled to take advantage of the underlying multi-processing heterogeneous system, making use of threads on the general-purpose multi-core CPU and those on the many packet processing engines (Figure 11). This model can take advantage of the “pool of threads” approach. In this approach, a thread, or a processing resource, can be assigned on-demand in response to the traffic’s varying processing requirements — for example to examine an IPSec tunnel. Once a thread completes its tasks, it is immediately added to the pool and is available for work again.

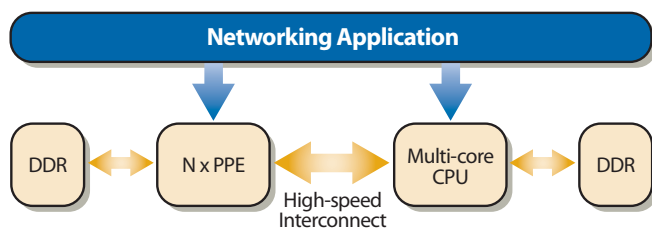


Figure 10. High-level programming in a heterogeneous multi-processing model: A pool of threads from the many-cores NFP and the general-purpose CPU are available for parallel processing.

Applying the Heterogeneous Model

The heterogeneous multi-processing model focuses on high-performance systems reaching 10Gbps and beyond. In this configuration (Figure 12), the NFP is used in an in-line mode. Flow classification rules are loaded into the NFP. Incoming packets from a media network interface are classified in flows, based on pre-loaded classification rules, and can be directly forwarded to a second media port, switching fabric, or backplane.

The NFP is capable of mirroring traffic to the CPU on a need basis. This way, the CPU utilization is reduced, allowing the system designer to use a less-powerful CPU or run additional applications on the same CPU. The NFP executes a load balancing algorithm when sending network traffic to the attached multi-core CPU. This way, it creates a parallel processing environment by ensuring that the various cores are utilized in a balanced and efficient way, allowing the multi-core CPU to execute multiple instances of single-threaded applications.

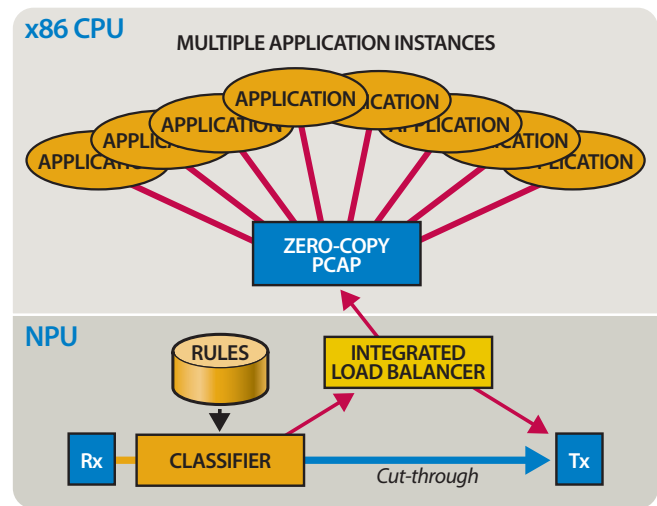


Figure 11. The NFP is shown in an in-line mode in a heterogeneous multi-processing system. When sending packets to the multi-core CPU, the NFP executes a load balancing algorithm to ensure that the various cores are utilized in a balanced and efficient way.

Summary

With the explosion of Internet traffic, including bandwidth-hungry video, and the ever-increasing need for security, communication equipment needs to be content- and application-aware at increasingly higher and higher-speeds. This speed is approaching 10Gbps at many points in the enterprise and carrier networks. Traditional network communication processors are not adequate to meet the L2-L7 requirements at such sustained line speeds. A new architecture is needed to meet the performance and power requirements: the heterogeneous multi-core, multi-processing architecture. In this architecture a multi-core, multi-threaded processor (the Network Flow Processor or NFP) is used in conjunction with a general-purpose multi-core CPU. This heterogeneous architecture will allow equipment providers to address these challenges by delivering high-performance field-programmable products enabling service providers to garner more revenue-per-user over a longer product life cycle.

Nabil Damouny is the senior director of marketing at Netronome Systems, and can be reached at nabil.damouny@netronome.com.

TM Netronome, the Netronome Logo, and “Intelligent to the Core.” are trademarks of Netronome Systems, Inc. All other trademarks are the property of their respective owners. Copyright © 2008 Reed Business Information, a division of Reed Elsevier Inc. Reprinted with permission. (This paper was originally published in EDN on-line on December 5, 2008.) 5-09



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
 +1 877 638 7629
netronome.com