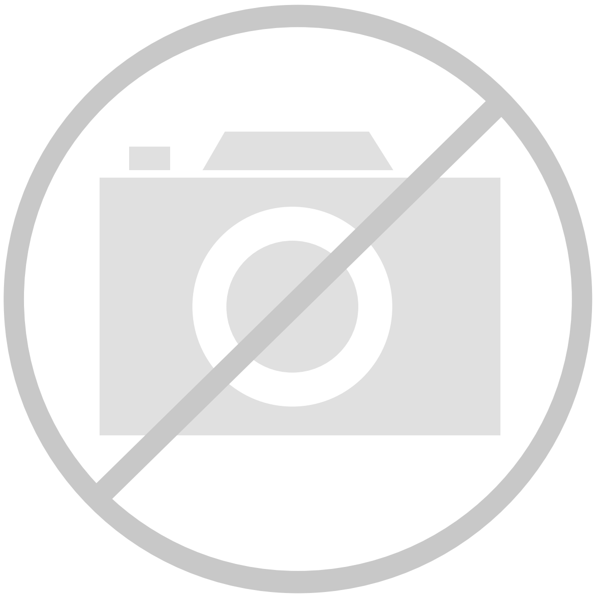


Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Active
Core Processor	-
Number of Cores/Bus Width	-
Speed	-
Co-Processors/DSP	-
RAM Controllers	-
Graphics Acceleration	-
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	-
Operating Temperature	-
Security Features	-
Package / Case	-
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=t4160nse7qtb

2 Summary of benefits

The T4 family of processors are ideal for combined control and data plane processing. A wide variety of applications can benefit from the processing, I/O integration, and power management capabilities. Similar to other QorIQ devices, the T4 family of processors' high level of integration offers significant space, weight, and power benefits compared to multiple discrete devices. Examples include:

- Service provider networking: RNC, metro networking, gateway, core/edge router, EPC, CRAN, ATCA, and AMC solutions.
- Enterprise equipment: router, switch services, and UTM appliances.
- Data centers: NFV, SDN, ADC, WOC, UTM, proxy, server appliance, and PCI Express (PCIe) offload.
- Storage controllers: FCoE bridging, iSCSI controller, and SAN controller.
- Aerospace, defense, and government: radar imaging, ruggedized network appliance, and cockpit display.
- Industrial computing: single-board computers and test equipment.

2.1 e6500 CPU core

The T4 family of processors are based on the Power Architecture® e6500 core. The e6500 core uses a seven-stage pipeline for low latency response while also boosting single-threaded performance. The e6500 core also offers high aggregate instructions per clock at lower power with an innovative "fused core" approach to threading. The e6500 core's fully resourced dual threads provide 1.7 times the performance of a single thread.

The e6500 cores are clustered in banks of four cores sharing a 2 MB L2 cache, allowing efficient sharing of code and data within a multicore cluster. Each e6500 core implements the Freescale AltiVec technology SIMD engine, dramatically boosting performance of heavy math algorithms with DSP-like performance.

The e6500 core features include:

- Up to 1.8 GHz dual threaded operation
- 7 DMIPS/MHz per core
- Advanced power saving modes, including state retention power gating

2.2 Virtualization

The T4 family of processors includes support for hardware-assisted virtualization. The e6500 core offers an extra core privilege level (hypervisor) and hardware offload of logical-to-real address translation. In addition, the T4 family of processors includes platform-level enhancements supporting I/O virtualization with DMA memory protection through IOMMUs and configurable "storage profiles" that provide isolation of I/O buffers between guest environments. Virtualization software for the T4 family includes kernel virtualization machine (KVM), Linux containers, and Freescale hypervisor and commercial virtualization software from vendors such as Enea®, Greenhills Software®, Mentor Graphics®, and Wind River.

2.3 Data Path Acceleration Architecture (DPAA)

The T4 family of processors enhance the QorIQ DPAA, an innovative multicore infrastructure for scheduling work to cores (phyiscal and virtual), hardware accelerators, and network interfaces.

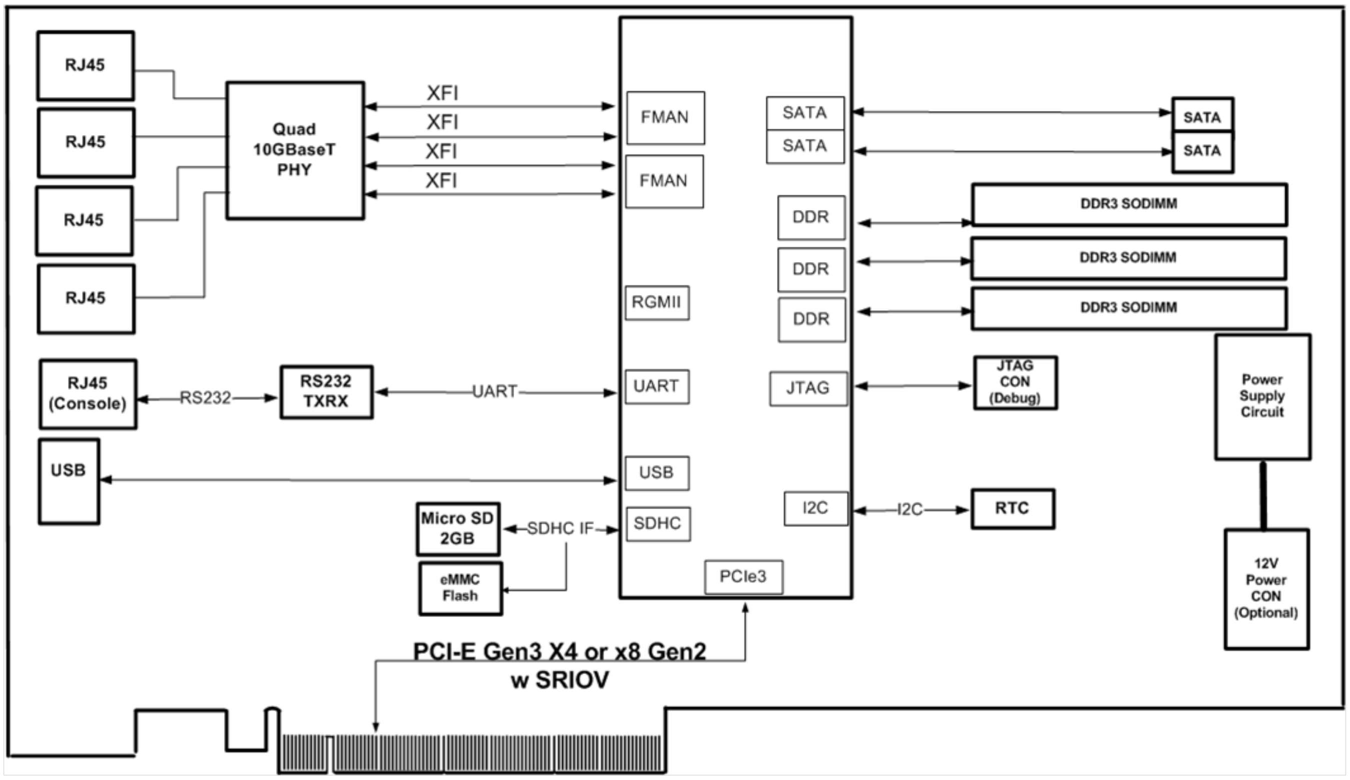


Figure 4. Intelligent network adapter

4 Multicore processing options

This flexible chip can be configured to meet many system application needs. The chip's CPUs (and hardware threads as virtual CPUs) can be combined as a fully-symmetric, multiprocessing, system-on-a-chip, or they can be operated with varying degrees of independence to perform asymmetric multiprocessing. High levels of processor independence, including the ability to independently boot and reset each core, is characteristic of the chip. The ability of the cores to run different operating systems, or run OS-less, provides the user with significant flexibility in partitioning between control, datapath, and applications processing. It also simplifies consolidation of functions previously spread across multiple discrete processors onto a single device.

While up to 24 Power Architecture threads (henceforth referred to as 'virtual CPUs', or 'vCPUs') offer a large amount of total, available computing performance, raw processing power is not enough to achieve multi-Gbps data rates in high-touch networking and telecom applications. To address this, this chip enhances the Freescale Data Path Acceleration Architecture (DPAA), further reducing data plane instructions per packet, and enabling more CPU cycles to work on value-added services as opposed to repetitive, low-level tasks. Combined with specialized accelerators for cryptography, pattern matching, and compression, the chip allows the user's software to perform complex packet processing at high data rates. There are many ways to map operating systems and I/O up to 24 chip vCPUs.

4.1 Asymmetric multiprocessing

As shown in this figure, the chip's vCPUs can be used in an asymmetric multi-processing model, with *n* copies of the same uni-processor OS, or *n* copies of OS 1, *n* copies of OS 2, and so on, up to 24 OS instances. The DPAA distributes work to the specific vCPUs based on basic classification or it puts work onto a common queue from which any vCPU can dequeue work.

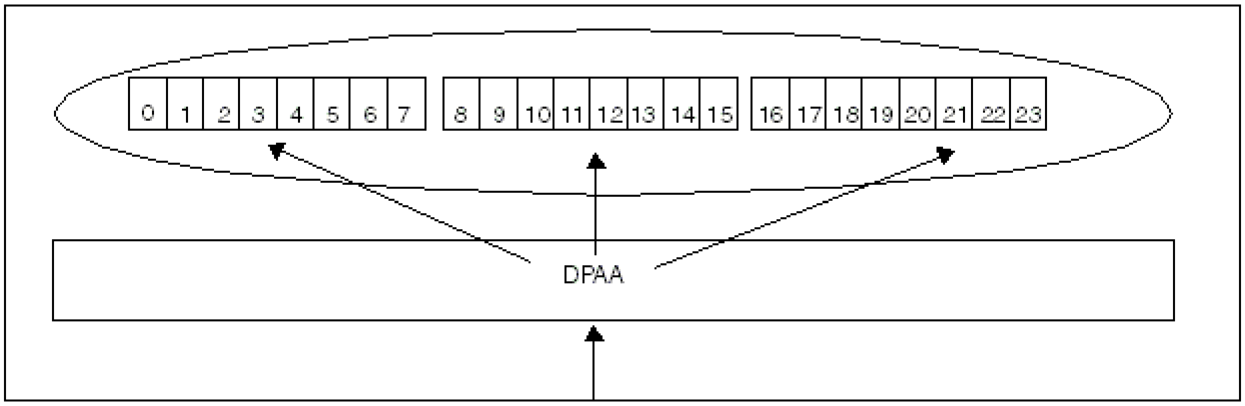


Figure 5. 24 vCPU AMP or SMP with affinity

4.2 Symmetric multiprocessing

Figure 5 also presents 24 vCPU SMP, where it is typical for data processing to involve some level of task affinity.

4.3 Mixed symmetric and asymmetric multiprocessing

This figure shows one possibility for a mixed SMP and AMP processing. Two physical CPUs (vCPUs 0-3) are combined in an SMP cluster for control processing, with the Datapath using exact match classification to send only control packets to the SMP cluster. The remaining virtual cores could run 20 instances of datapath software.

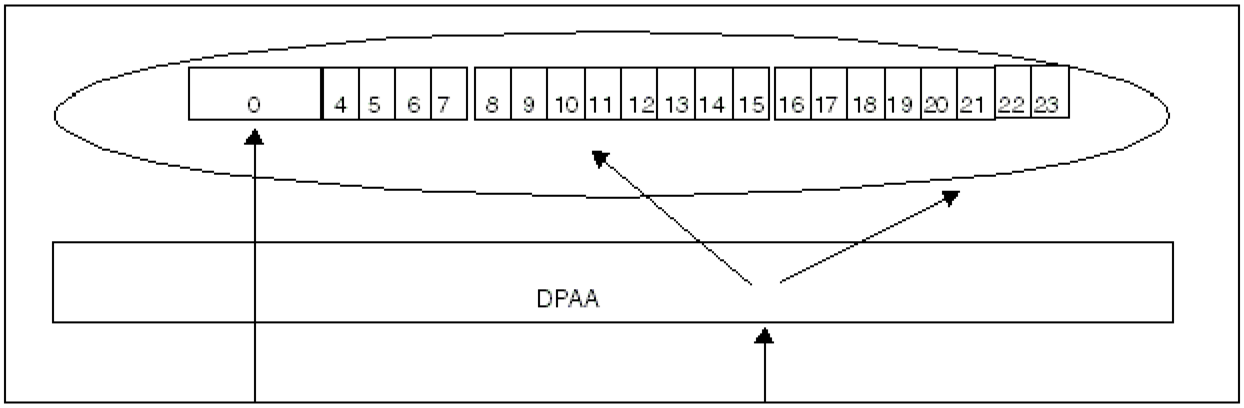


Figure 6. Mixed SMP and AMP option 1

This figure shows another possibility for mixed SMP and AMP processing. Two of the physical cores are run in single threaded mode; the remaining physical cores operate as four virtual CPUs. The Datapath directs traffic to specific software partitions based on physical Ethernet port, classification, or some combination.

- RegEx Pattern Matching Acceleration (PME 2.1) at up to 10 Gbps
- Decompression/Compression Acceleration (DCE 1.0) at up to 20 Gbps
- DPAA chip-to-chip interconnect via RapidIO Message Manager (RMAN 1.0)

- Up to 32 SerDes lanes at up to 10.3125 GHz
- Ethernet interfaces
  - Up to four 10 Gbps Ethernet XAUI or 10GBase-KR XFI MACs
  - Up to sixteen 1 Gbps Ethernet MACs
  - Up to two 1Gbps Ethernet RGMII MACs
  - Maximum configuration of 4 x 10 GE (XFI) + 10 x 1 GE (SGMII) + 2 x 1 GE (RGMII)
- High-speed peripheral interfaces
  - Up to four PCI Express 2.0 controllers, two supporting 3.0
  - Two Serial RapidIO 2.0 controllers/ports running at up to 5 GHz with Type 11 messaging and Type 9 data streaming support
  - Interlaken look-aside interface for serial TCAM connection at 6.25 and 10.3125 Gbps per-lane rates.
- Additional peripheral interfaces
  - Two serial ATA (SATA 2.0) controllers
  - Two high-speed USB 2.0 controllers with integrated PHY
  - Enhanced secure digital host controller (SD/MMC/eMMC)
  - Enhanced serial peripheral interface (eSPI)
  - Four I2C controllers
  - Four 2-pin or two 4-pin UARTs
  - Integrated Flash controller supporting NAND and NOR flash
- Three eight-channel DMA engines.
- Support for hardware virtualization and partitioning enforcement
- QorIQ Platform's Trust Architecture 2.0

5.3 Critical performance parameters

This table lists key performance indicators that define a set of values used to measure SoC operation.

Table 1. Critical performance parameters

Indicator	Values(s)
Top speed bin core frequency	1.8 GHz
Maximum memory data rate	1867 MHz (DDR3)[1], 1600 MHz for DDR3L • 1.5 V for DDR3 • 1.35 V for DDR3L
Integrated flash controller (IFC)	1.8 V
Operating junction temperature range	0-105 C
Package	1932-pin, flip-chip plastic ball grid array (FC-PBGA), 45 x 45mm

1. Conforms to JEDEC standard

5.4 Core and CPU clusters

This chip offers 12, high-performance, 64-bit Power Architecture, Book E-compliant cores. Each CPU core supports two hardware threads, which software views as a virtual CPU. The core CPUs are arranged in clusters of four with a shared 2 MB L2 cache.

This table shows the computing metrics the core supports.

Table 2. Power architecture metrics

Metric	Per core	Per cluster	Full device
DMIPS	10,800	43,200	129,600
Single-precision GFLOPs	18	72	Up to 216
Double-precision GFLOPs	3.6	14.4	Up to 42.4

The core subsystem includes the following features:

- Up to 1.8 GHz
- Dual-thread with simultaneous multi-threading (SMT)
  - Threading can be disabled on a per CPU basis
- 40-bit physical addressing
- L2 MMU
  - Supporting 4 KB pages
  - TLB0; 8-way set-associative, 1024-entries (4 KB pages)
  - TLB1; fully associative, 64-entry, supporting variable size pages and indirect page table entries
- Hardware page table walk
- 64-byte cache line size
- L1 caches, running at core frequency
  - 32 KB instruction, 8-way set-associative
  - 32 KB data, 8-way set-associative
  - Each with data and tag parity protection
- Hardware support for memory coherency
- Five integer units: 4 simple (2 per thread), 1 complex (integer multiply and divide)
- Two load-store units: one per thread
- Classic double-precision floating-point unit
  - Uses 32 64-bit floating-point registers (FPRs) for scalar single- and double-precision floating-point arithmetic
  - Designed to comply with IEEE Std. 754™-1985 FPU for both single and double-precision operations
- AltiVec unit
  - 128-bit Vector SIMD engine
  - 32 128-bit VR registers
  - Operates on a vector of
    - Four 32-bit integers
    - Four 32-bit single precision floating-point units
    - Eight 16-bit integers
    - Sixteen 8-bit integers
  - Powerful permute unit
  - Enhancements include: Move from GPRs to VR, sum of absolute differences operation, extended support for misaligned vectors, handling head and tails of vectors
- Supports Data Path Acceleration Architecture (DPAA) data and context "stashing" into L1 and L2 caches
- User, supervisor, and hypervisor instruction level privileges
- Addition of Elemental Barriers and "wait on reservation" instructions
- New power-saving modes including "drowsy core" with state retention and nap
  - State retention power-saving mode allows core to quickly wake up and respond to service requests
- Processor facilities
  - Hypervisor APU
  - "Decorated Storage" APU for improved statistics support
    - Provides additional atomic operations, including a "fire-and-forget" atomic update of up to two 64-bit quantities by a single access
  - Addition of Logical to Real Address translation mechanism (LRAT) to accelerate hypervisor performance
  - Expanded interrupt model

- Supports external SD bus voltage selection by register configuration
- Host will send 80 idle SD clock cycles to card, which are needed during card power-up, if bit INITA in the system control register (SYSCTL) is set

5.8 Universal serial bus (USB) 2.0

The two USB 2.0 controllers with integrated PHY provide point-to-point connectivity that complies with the USB specification, Rev. 2.0. Each of the USB controllers with integrated PHY can be configured to operate as a stand-alone host, and one of the controllers (USB #2) can be configured as a stand-alone device, or with both host and device functions operating simultaneously.

5.9 High-speed peripheral interface complex (HSSI)

This chip offers a variety of high-speed serial interfaces, sharing a set of 16 SerDes lanes. Each interface is backed by a high speed serial interface controller. This chip has the following types and quantities of controllers:

- Four 2.0 PCI Express controllers, two supporting 3.0
- Two Serial RapidIO 2.0
- Two SATA 2.0
- One Interlaken look-aside
- Aurora
- Up to sixteen Ethernet controllers with various protocols

5.9.1 PCI Express

Each of the chip's PCI Express controllers is compliant with the PCI Express Base Specification Revision 2.0. Two are additionally compliant with Revision 3.0 (8 GHz). Key features of each PCI Express controller include the following:

- Power-on reset configuration options allow root complex or endpoint functionality.
- The physical layer operates at 2.5, 5, or 8 Gbaud data rate per lane.
- x4, x2, and x1 link widths supported on all controllers
- Two controllers can support x8 link width
- Both 32- and 64-bit addressing
- 256-byte maximum payload size
- Full 64-bit decode with 40-bit wide windows
- Inbound INTx transactions
- Message signaled interrupt (MSI) transactions
- One PCI Express controller supports end-point SR-IOV
    - Two physical functions, each with 64 virtual functions
    - Eight MSI-X per virtual function

5.10.1 Packet distribution and queue/congestion management

This table lists some packet distribution and queue/congestion management offload functions.

Table 3. Offload functions

Function type	Definition
Data buffer management	Supports allocation and deallocation of buffers belonging to pools originally created by software with configurable depletion thresholds. Implemented in a module called the Buffer Manager (BMan).
Queue management	Supports queuing and quality-of-service scheduling of frames to CPUs, network interfaces and DPAA logic blocks, maintains packet ordering within flows. Implemented in a module called the Queue Manager (QMan). The QMan, besides providing flow-level queuing, is also responsible for congestion management functions such as RED/WRED, congestion notifications and tail discards.
Packet distribution	Supports in-line packet parsing and general classification to enable policing and QoS-based packet distribution to the CPUs for further processing of the packets. This function is implemented in the block called the Frame Manager (FMan).
Policing	Supports in-line rate-limiting by means of two-rate, three-color marking (RFC 2698). Up to 256 policing profiles are supported. This function is also implemented in the FMan.
Egress Scheduling	Supports hierarchical scheduling and shaping, with committed and excess rates. This function is supported in the QMan, although the FMan performs the actual transmissions.

5.10.2 Accelerating content processing

Properly implemented acceleration logic can provide significant performance advantages over most optimized software with acceleration factors on the order of 10-100x. Accelerators in this category typically touch most of the bytes of a packet (not just headers). To avoid consuming CPU cycles in order to move data to the accelerators, these engines include well-pipelined DMAs. This table lists some specific content-processing accelerators on the chip.

Table 4. Content-processing accelerators

Interface	Definition
SEC	Crypto-acceleration for protocols such as IPsec, SSL, and 3GPP RLC
PME	Regex style pattern matching for unanchored searches, including cross-packet stateful patterns
DCE	Compression/Decompression acceleration for ZLib and deflate

5.10.3 Enhancements of T4240 compared to first generation DPAA

A short summary of T4240 enhancements over the first generation DPAA (as implemented in the P4080) is provided below:

- Frame Manager
  - 2x performance increase (up to 25 Gbps per FMan)
  - Storage profiles.
  - HiGig (3.125 GHz) and HiGig2 (3.125 GHz and 3.75 GHz)
  - Energy Efficient Ethernet
- SEC 5.0
  - 2x performance increase for symmetric encryption and protocol processing

        - Up to 20 Gbps for IPsec @ Imix
    - 10x performance increase for public key algorithms
    - Support for 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3 (ZUC)
- DCE 1.0, new accelerator for compression/decompression
- RMan (Serial RapidIO Manager)
- DPAA overall capabilities
    - Data Center Bridging
    - Egress Traffic Shaping

5.10.4 DPAA terms and definitions

The QorIQ Platform's Data Path Acceleration Architecture (henceforth DPAA) assumes the existence of network flows, where a flow is defined as a series of network datagrams, which have the same processing and ordering requirements. The DPAA prescribes data structures to be initialized for each flow. These data structures define how the datagrams associated with that flow move through the DPAA. Software is provided a consistent interface (the software portal) for interacting with hardware accelerators and network interfaces.

All DPAA entities produce data onto frame queues (a process called enqueuing) and consume data from frame queues (dequeuing). Software enqueues and dequeues through a software portal (each vCPU has two software portals), and the FMan, RMan, and DPAA accelerators enqueue/dequeue through hardware portals. This figure illustrates this key DPAA concept.

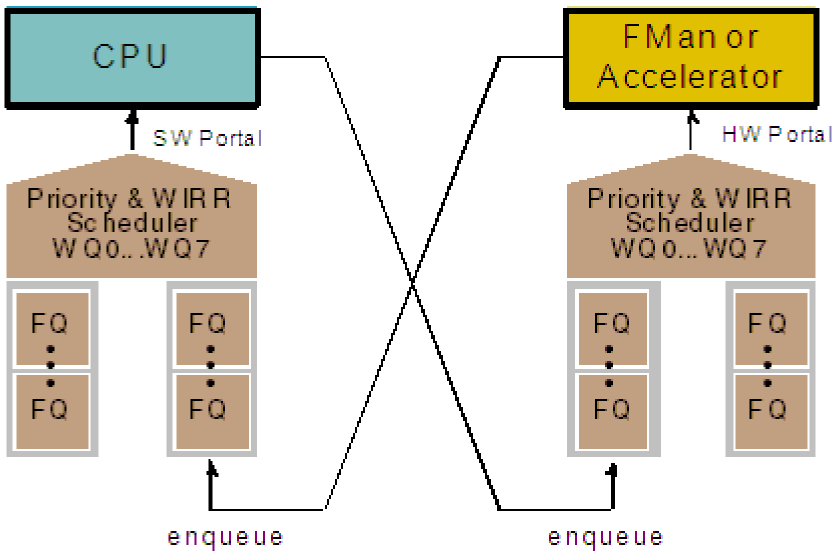

CPU
SW Portal
Priority & WIRR Scheduler WQ0...WQ7
FQ
FQ
FQ
FQ
enqueue
FMan or Accelerator
HW Portal
Priority & WIRR Scheduler WQ0...WQ7
FQ
FQ
FQ
FQ
enqueue

Figure 9. DPAA enqueuing and dequeuing

This table lists common DPAA terms and their definitions.

Table 5. DPAA terms and definitions

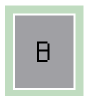
Term	Definition	Graphic representation
Buffer	Region of contiguous memory, allocated by software, managed by the DPAA BMan	

Table continues on the next page...

5.10.5.1.1 Receiver functionality: parsing, classification, and distribution

Each Frame Manager matches its 25 Gbps Ethernet connectivity with 25 Gbps (37.2 Mpps) of Parsing, Classification, and Distribution (PCD) performance. PCD is the process by which the Frame Manager identifies the frame queue on which received packets should be enqueued. The consumer of the data on the frame queues is determined by Queue Manager configuration; however, these activities are closely linked and managed by the FMan Driver and FMan Configuration Tool, as in previous QorIQ SoCs.

This figure provides a logical view of the FMan's processing flow, illustrating the PCD features.

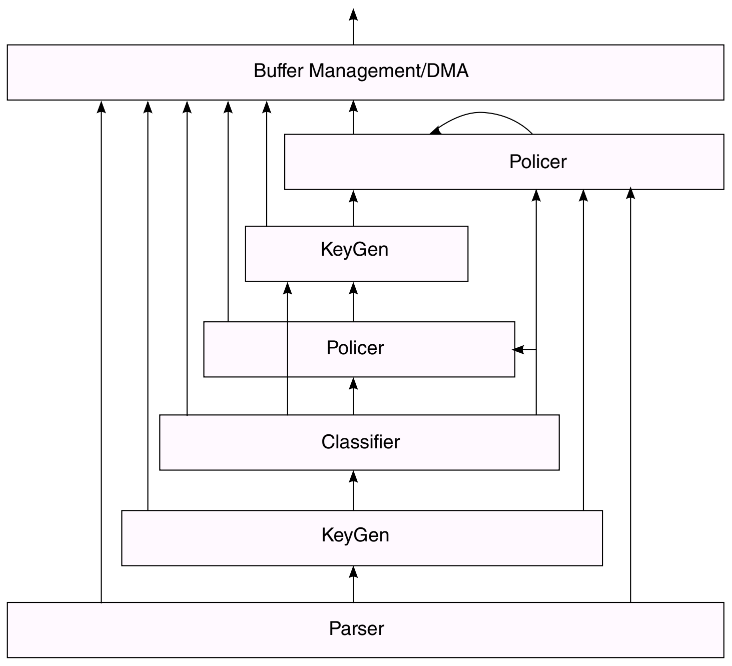


Figure 11. Logical view of FMan processing

Each frame received by the FMan is buffered internally while the Parser, KeyGen, and Classification functions operate.

The parse function can parse many standard protocols, including options and tunnels, and it supports a generic configurable capability to allow proprietary or future protocols to be parsed. Hard parsing of the standard protocol headers can be augmented with user-defined soft parsing rules to handle proprietary header fields. Hard and soft parsing occurs at wire speed.

This table defines several types of parser headers.

Table 6. Parser header types

Header type	Definition
Self-describing	Announced by proprietary values of Ethertype, protocol identifier, next header, and other standard fields. They are self-describing in that the frame contains information that describes the presence of the proprietary header.
Non-self-describing	Does not contain any information that indicates the presence of the header.

Table continues on the next page...

Table 6. Parser header types (continued)

Header type	Definition
	For example, a frame that always contains a proprietary header before the Ethernet header would be non-self-describing. Both self-describing and non-self-describing headers are supported by means of parsing rules in the FMan.
Proprietary	Can be defined as being self-describing or non-self-describing

The underlying notion is that different frames may require different treatment, and only through detailed parsing of the frame can proper treatment be determined.

Parse results can (optionally) be passed to software.

5.10.5.1.2 FMan distribution and policing

After parsing is complete, there are two options for treatment, as shown in this table.

Table 7. Post-parsing treatment options

Treatment	Function	Benefits
Hash	• Hashes select fields in the frame as part of a spreading mechanism. • The result is a specific frame queue identifier. • To support added control, this FQID can be indexed by values found in the frame, such as TOS or p-bits, or any other desired field(s).	Useful when spreading traffic while obeying QoS constraints is required
Classification look-up	• Looks up certain fields in the frame to determine subsequent action to take, including policing. • The FMan contains internal memory that holds small tables for this purpose. • The user configures the sets of lookups to perform, and the parse results dictate which one of those sets to use. • Lookups can be chained together such that a successful look-up can provide key information for a subsequent look-up. After all the look-ups are complete, the final classification result provides either a hash key to use for spreading, or a FQ ID directly.	• Useful when hash distribution is insufficient and a more detailed examination of the frame is required • Can determine whether policing is required and the policing context to use

Key benefits of the FMan policing function are as follows:

- Because the FMan has up to 256 policing profiles, any frame queue or group of frame queues can be policed to either drop or mark packets if the flow exceeds a preconfigured rate.
- Policing and classification can be used in conjunction to mitigate Distributed Denial of Service Attack (DDOS).
- The policing is based on the two-rate-three-color marking algorithm (RFC2698). The sustained and peak rates, as well as the burst sizes, are user-configurable. Therefore, the policing function can rate-limit traffic to conform to the rate that the flow is mapped to at flow set-up time. By prioritizing and policing traffic prior to software processing, CPU cycles can focus on important and urgent traffic ahead of other traffic.

Each FMan also supports PCD on traffic arriving from within the chip. This is referred to as off-line parsing, and it is useful for reclassification following decapsulation of encrypted or compressed packets.

FMan PCD supports virtualization and strong partitioning by delaying buffer pool selection until after classification. In addition to determining the FQ ID for the classified packet, the FMan also determines the 'storage profile.' Configuration of storage profiles (up to 32 per physical port) allows the FMan to store received packets using buffer pools owned by a single software partition, and enqueue the associated Frame Descriptor to a frame queue serviced by only that software partition.

The SEC 5.0 can perform full protocol processing for the following security protocols:

- IPsec
- SSL/TLS
- 3GPP RLC encryption/decryption
- LTE PDCP
- SRTP
- IEEE 802.1AE MACSec
- IEEE 802.16e WiMax MAC layer

The SEC 5.0 supports the following algorithms, modes, and key lengths as raw modes, or in combination with the security protocol processing described above.

- Public Key Hardware Accelerators (PKHA)
  - RSA and Diffie-Hellman (to 4096b)
  - Elliptic curve cryptography (1023b)
- Data Encryption Standard Accelerators (DESA)
  - DES, 3DES (2-key, 3-key)
  - ECB, CBC, OFB, and CFB modes
- Advanced Encryption Standard Accelerators (AESA)
  - Key lengths of 128-bit, 192-bit, and 256-bit
  - ECB, CBC, CTR, CCM, GCM, CMAC, OFB, CFB, xcbc-mac, and XTS
- ARC Four Hardware Accelerators (AFHA)
  - Compatible with RC4 algorithm
- Message Digest Hardware Accelerators (MDHA)
  - SHA-1, SHA-256, 384, 512-bit digests
  - MD5 128-bit digest
  - HMAC with all algorithms
- Kasumi/F8 Hardware Accelerators (KFHA)
  - F8, F9 as required for 3GPP
  - A5/3 for GSM and EDGE, GEA-3 for GPRS
- Snow 3G Hardware Accelerators (SNOWf8 and SNOWf9)
  - Implements Snow 3.0, F8 and F9 modes
- ZUC Hardware Accelerators (ZUCE and ZUCA)
  - Implements 128-EEA3 & 128-EIA3
- CRC Unit
  - Standard and user-defined polynomials
- Random Number Generator
  - Incorporates TRNG entropy generator for seeding and deterministic engine (SHA-256)
  - Supports random IV generation

The SEC 5.0 is designed to support bulk encryption at up to 40 Gbps, large packet/record IPsec/SSL at up to 30 Gbps, and 20 Gbps for IPsec ESP at Imix packet sizes. 3G and LTE algorithms are supported at 10 Gbps or more.

The SEC dequeues data from its QMan hardware portal and, based on FQ configuration, also dequeues associated instructions and operands in the Shared Descriptor. The SEC processes the data then enqueues it to the configured output FQ. The SEC uses the Status/CMD word in the output Frame Descriptor to inform the next consumer of any errors encountered during processing (for example, received packet outside the anti-replay window.)

    - All standard modes of decompression
    - No compression
    - Static Huffman codes
    - Dynamic Huffman codes
  - Provides option to return original compressed Frame along with the uncompressed Frame or release the buffers to BMan
  - Does not support use of ZLIB preset dictionaries (FDICT flag = 1 is treated as an error).
  - Base 64 decoding (RFC4648) prior to decompression

The DCE 1.0 is designed to support up to 8.8 Gbps for either compression or decompression, or 17.5 Gbps aggregate at ~4 KB data sizes.

5.10.6 DPAA capabilities

Some DPAA features and capabilities have been described in the sections covering individual DPAA components. This section describes some capabilities enabled by DPAA components working together.

5.10.6.1 Ingress policing and congestion management

In addition to selecting FQ ID and storage profile, classification can determine whether policing is required for a received packet, along with the specific policing context to be used.

FMan policing capabilities include the following:

- RFC2698: two-rate, three-color marking algorithm
- RFC4115: Differentiated service two-rate, three-color marker with efficient handling of in-profile traffic
- Up to 256 internal profiles

The sustained and peak rates, and burst size for each policing profile are user-configurable.

5.10.6.2 Customer-edge egress-traffic management (CEETM)

Customer-edge egress-traffic management (CEETM) is a DPAA enhancement first appearing in the T4240. T4240 continues to support the work queue and frame queue scheduling functionality available in the P4080 and other first generation QorIQ chips, but introduces alternative functionary, CEETM, that can be mode selected on a network interface basis to support the shaping and scheduling requirements of carrier Ethernet connected systems.

5.10.6.2.1 CEETM features

Each instance of CEETM (one per FMan) provides the following features:

- Supports hierarchical multi-level scheduling and shaping, which:
  - is performed in an atomic manner; all context at all levels is examined and updated synchronously.
  - employs no intermediate buffering between class queues and the direct connect portal to the FMan.
- Supports dual-rate shaping (paired committed rate (CR) shaper and excess rate (ER) shaper) at all shaping points.
  - Shapers are token bucket based with configurable rate and burst limit.
  - Paired CR/ER shapers may be configured as independent or coupled on a per pair basis; coupled means that credits to the CR shaper in excess of its token bucket limit is credited to the ER bucket
- Supports eight logical network interfaces (LNI)
  - Each LNI:
    - aggregates frames from one or more channels.
    - priority schedules unshaped frames (aggregated from unshaped channels), CR frames, and ER frames (aggregated from shaped channels)

        - applies a dual-rate shaper to the aggregate of CR/ER frames from shaped channels
        - can be configured (or reconfigured for lossless interface failover) to deliver frames to any network interface.
- Supports 32 channels available for allocation across the eight LNIs
- Each channel:
    - can be configured to deliver frames to any LNI.
    - can be configure to be unshaped or shaped; when shaped, a dual rate shaper applies to the aggregate of CR/ER frames from the channel.
    - has eight independent classes and eight grouped classes; grouped classes can be configured as one class group of eight or as two class groups of four.
    - supports weighted bandwidth fairness within grouped class groups with weights configured on a channel and class basis.
    - strict priority scheduling of the eight independent classes and the aggregate(s) of the grouped classe(s); the priority of each of the two class groups can be independently configured to be immediately below any of the independent classes.
    - is configurable such that each of the eight independent classes and two class groups can supply CR frames, ER frames or both when channel is configured to be shaped.
    - is configured independently.
- Each class:
    - has a dedicated class queue (CQ) with equivalent congestion management functionality available to FQs.
    - can have a dedicated or shared Congestion Management Record supports sufficient number of CMRs for all CQs to have a dedicated CMR, if desired.
    - can be flow-controlled by traffic-class flow control messages via portal; achieves backward compatibility with by allowing each of these 16 classes to be configured (per LNI) to respect one or none of the 8 on/off control bits within existing message format (as was defined for 8-class non-CEETM channels).
    - is identified via a "logical frame queue identifier" to maintain semantic compatibility with enqueue commands to frame queues (non-CEETM queues).
    - supports the identification of intra-class flows (logically equivalent to FQs but not queued separately) in order to apply static context (Context_A and Context_B) to frames as they are dequeued from CQs; this provides functionality equivalent to that available when a frame is dequeue from a frame queue (non-CEETM queues).

5.10.6.2.2 CEETM configuration

The CEETM configuration, shown in Figure 13, is very asymmetrical and is intended to demonstrate the degrees of configurability rather than an envisioned use case.

NOTE

The color green denotes logic units and signal paths that relate to the request and fulfillment of committed rate (CR) packet transmission opportunities. The color yellow denotes the same for excess rate (ER). The color black denotes logic units and signal paths that are used for unshaped opportunities or that operate consistently whether used for CR or ER opportunities.

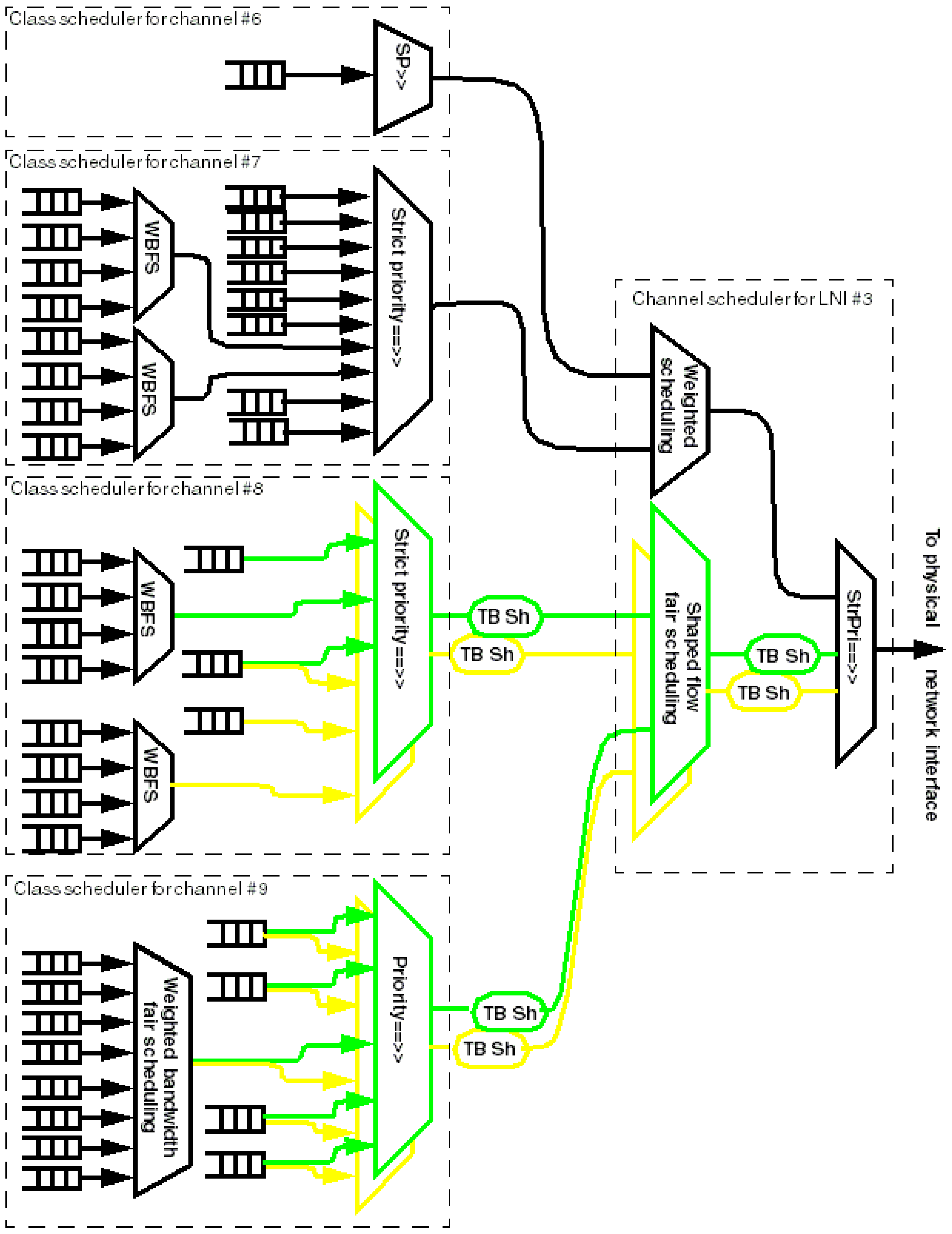

Class scheduler for channel #6
SP>>
Class scheduler for channel #7
WBFS
WBFS
Strict priority==>>
Class scheduler for channel #8
WBFS
WBFS
Strict priority==>>
TB Sh
TB Sh
Class scheduler for channel #9
Weighted bandwidth fair scheduling
Priority==>>
TB Sh
TB Sh
Channel scheduler for LNI #3
Weighted scheduling
Shaped flow fair scheduling
TB Sh
TB Sh
StrPri==>>
To physical network interface

Figure 13. CEETM scheduler: illustrative configuration scenario

Figure 13 illustrates the following scenario:

- Channels #6, #7, #8 and #9 have been configured to be scheduled by the channel scheduler for LNI#3 (for example, all the packets from these channels are directed to the physical network interface configurably coupled to LNI#3).
- Channels #6 and #7 have been configured to be "unshaped." Packets from these channels will not be subjected to shaping at the channel level and will feed the top priority level within the LNI, which is also not subjected to shaping. Their class schedulers will not distinguish between CR and ER opportunities.
- Channels #8 and #9 have been configured to be "shaped." Their class schedulers will distinguish between CR and ER opportunities. The CR/ER packets to be sent from each channel shall be subjected to a pair of CR/ER token bucket shapers specific to that channel. The aggregate of CR/ER packets from these channels are subject to a pair of CR/ER token bucket shapers specific to LNI#3.
- Channel #6 has only one class in use. That class queue behaves as if it were a channel queue and as a peer to Channel #7. Unused classes do not have to be configured as such; they are simply not used.
- Channel #7 has all 16 classes in use.
  - The group classes have been configured as two groups (A and B) of four classes.
  - The priority of the groups A and B have both been set to be immediately below independent class 5. In a case of similar configuration group A has higher priority than group B.
- Channel #8 has three independent classes and two groups of four grouped classes in use.
  - The priorities of the class groups A and B have been set to be immediately below independent class 0 and class 2 respectively.
  - Independent class 0 and class group A have been configured to request and fulfill only CR packet opportunities.
  - Independent class 1 has been configured to request and fulfill both CR and ER packet opportunities.
  - Independent class 2 and class group B have been configured to request and fulfill only ER packet opportunities.
- Channels #9 has four independent classes and one group of eight grouped classes in use.
  - The group classes have been configured as one group (A) of eight classes.
  - All independent classes and the class group (A) have been configured to request and fulfill both CR and ER packet opportunities.

Benefits of the CEETM include the following:

- Provides "virtual" ports for multiple applications or users with different QoS/CoS requirements which are sharing an egress interface
- Supports DSCP capable scheduling for the following virtual link with configurable combinations of strict priority and weighted scheduling
  - Weighted scheduling closely approximating WFQ
- Supports traffic shaping
  - dual rate shaping of the virtual links
- Supports aggregating traffic from multiple virtual links and shaping this aggregate
- Hierarchical scheduling and shaping
- Class-based scheduling and dual rate shaping
- Supports a subset of the IEEE Data Center Bridging (DCB) standards

5.10.6.3 Data Center Bridging (DCB)

Data Center Bridging (DCB) refers to a series of inter-related IEEE specifications collectively designed to enhance Ethernet LAN traffic prioritization and congestion management. Although the primary objective is the data center environment (consisting of servers and storage arrays), some aspects of DCB are applicable to more general uses of Ethernet, within and between network nodes.

The SoC DPAA is compliant with the following DCB specifications :

- IEEE Std. 802.1Qbb: Priority-based flow control (PFC)
  - PAUSE frame per Ethernet priority code point (8)
  - Prevents single traffic class from throttling entire port
- IEEE Std. 802.1Qaz: Enhanced transmission selection (ETS)
  - Up to three Traffic Class Groups (TCG), where a TCG is composed of one or more priority code points
  - Bandwidth allocation and transmit scheduling (1% granularity) by traffic class group
  - If one of the TCGs does not consume its allocated bandwidth, unused bandwidth is available to other TCGs

5.11 Resource partitioning and QorIQ Trust Architecture

Consolidation of discrete CPUs into a single, multicore chip introduces many opportunities for unintended resource contentions to arise, particularly when multiple, independent software entities reside on a single chip. A system may exhibit erratic behavior if multiple software partitions cannot effectively partition resources. Device consolidation, combined with a trend toward embedded systems becoming more open (or more likely to run third-party or open-source software on at least one of the cores), creates opportunities for malicious code to enter a system.

This chip offers a new level of hardware partitioning support, allowing system developers to ensure software running on any CPU only accesses the resources (memory, peripherals, and so on) that it is explicitly authorized to access. This section provides an overview of the features implemented in the chip that help ensure that only trusted software executes on the CPUs, and that the trusted software remains in control of the system with intended isolation.

5.11.1 Core MMU, UX/SX bits, and embedded hypervisor

The chip's first line of defense against unintended interactions amongst the multiple CPUs/OSes is each core vCPU's MMU. A vCPU's MMU is configured to determine which addresses in the global address map the CPU is able to read or write. If a particular resource (memory region, peripheral device, and so on) is dedicated to a single vCPU, that vCPU's MMU is configured to allow access to those addresses (on 4 KB granularity); other vCPU MMUs are not configured for access to those addresses, which makes them private. When two vCPUs need to share resources, their MMUs are both configured so that they have access to the shared address range.

This level of hardware support for partitioning is common today; however, it is not sufficient for many core systems running diverse software. When the functions of multiple discrete CPUs are consolidated onto a single multicore chip, achieving strong partitioning should not require the developer to map functions onto vCPUs that are the exclusive owners of specific platform resources. The alternative, a fully open system with no private resources, is also unacceptable. For this reason, the core's MMU also includes three levels of access permissions: user, supervisor (OS), and hypervisor. An embedded hypervisor (for example, KVM, XEN, QorIQ ecosystem partner hypervisor) runs unobtrusively beneath the various OSes running on the vCPUs, consuming CPU cycles only when an access attempt is made to an embedded hypervisor-managed shared resource.

The embedded hypervisor determines whether the access should be allowed and, if so, proxies the access on behalf of the original requestor. If malicious or poorly tested software on any vCPU attempts to overwrite important device configuration registers (including vCPU's MMU), the embedded hypervisor blocks the write. High and low-speed peripheral interfaces (PCI Express, UART), when not dedicated to a single vCPU/partition, are other examples of embedded hypervisor managed resources. The degree of security policy enforcement by the embedded hypervisor is implementation-dependent.

In addition to defining regions of memory as being controlled by the user, supervisor, or hypervisor, the core MMU can also configure memory regions as being non-executable. Preventing CPUs from executing instructions from regions of memory used as data buffers is a powerful defense against buffer overflows and other runtime attacks. In previous generations of Power Architecture, this feature was controlled by the NX (no execute) attribute. In new Power Architecture cores such as the e6500 core, there are separate bits controlling execution for user (UX) and supervisor (SX).

5.11.2 Peripheral access management unit (PAMU)

MMU-based access control works for software running on CPUs; however, these are not the only bus masters in the SoC. Internal components with bus mastering capability (FMan, RMan, PCI Express controller, PME, SEC, and so on) also need to be prevented from reading and writing to certain memory regions. These components do not spontaneously generate access attempts; however, if programmed to do so by buggy or malicious software, any of them could read or write sensitive data registers and crash the system. For this reason, the SoC also includes a distributed function referred to as the peripheral access management unit (PAMU).

PAMUs provide address translation and access control for all non-CPU initiators in the system. PAMU access control is based on the logical I/O device number (LIODN) advertised by a bus master for a given transaction. LIODNs can be static (for example, PCI Express controller #1 always uses LIODN 123) or they can be dynamic, based on the ID of the CPU that programmed the initiator (for example, the SEC uses LIODN 456 because it was given a descriptor by vCPU #2). In the dynamic example, the SoC architecture provides positive identification of the vCPU programming the SEC, preventing LIODN spoofing.

5.11.3 IO partitioning

The simplest IO configuration in chips running multiple independent software partitions is to dedicate specific IO controllers (PCI Express, SATA, Serial RapidIO controllers) to specific vCPUs. The core MMUs and PAMUs can enforce these access permissions to insure that only the software partition owning the IO is able to use it. The obvious problem with this approach is that there are likely to be more software partitions wanting IO access than there are IO controllers to dedicate to each.

Safe IO sharing can be accomplished through the use of a hypervisor; however, there is a performance penalty associated with virtual IO, as the hypervisor must consume CPU cycles to schedule the IO requests and get the results back to the right software partition.

The DPAA (described in Data Path Acceleration Architecture (DPAA)") was designed to allow multiple partitions to efficiently share accelerators and IOs, with its major capabilities centered around sharing Ethernet ports. These capabilities were enhanced in the chip with the addition of FMan storage profiles. The chip's FMans perform classification prior to buffer pool selection, allowing Ethernet frames arriving on a single port to be written to the dedicated memory of a single software partition. This capability is fully described in Receiver functionality: parsing, classification, and distribution."

The addition of the RMan extends the chip's IO virtualization by allowing many types of traffic arriving on Serial RapidIO to enter the DPAA and take advantage of its inherent virtualization and partitioning capabilities.

The PCI Express protocol lacks the PDU semantics found in Serial RapidIO, making it difficult to interwork between PCI Express controllers and the DPAA; however, PCI Express has made progress in other areas of partition. The Single Root IO Virtualization specification, which the chip supports as an endpoint, allows external hosts to view the chip as multiple two physical functions (PFs), where each PF supports up to 64 virtual functions (VFs). Having multiple VFs on a PCI Express port effectively channelizes it, so that each transaction through the port is identified as belonging to a specific PF/VF combination (with associated and potentially dedicated memory regions). Message signalled interrupts (MSIs) allow the external Host to generate interrupts associated with a specific VF.

5.11.4 Secure boot and sensitive data protection

The core MMUs and PAMU allow the SoC to enforce a consistent set of memory access permissions on a per-partition basis. When combined with an embedded hypervisor for safe sharing of resources, the SoC becomes highly resilient to poorly tested or malicious code. For system developers building high reliability/high security platforms, rigorous testing of code of known origin is the norm.

For this reason, the SoC offers a secure boot option, in which the system developer digitally signs the code to be executed by the CPUs, and the SoC insures that only an unaltered version of that code runs on the platform. The SoC offers both boot time and run time code authenticity checking, with configurable consequences when the authenticity check fails. The SoC also supports protected internal and external storage of developer-provisioned sensitive instructions and data. For example, a system developer may provision each system with a number of RSA private keys to be used in mutual authentication and key exchange. These values would initially be stored as encrypted blobs in external non-volatile memory; but, following secure boot, these values can be decrypted into on-chip protected memory (portion of platform cache dedicated as SRAM). Session keys, which may number in the thousands to tens of thousands, are not good candidates for on-chip storage, so the SoC offers session key encryption. Session keys are stored in main memory, and are decrypted (transparently to software and without impacting SEC throughput) as they are brought into the SEC 5.0 for decryption of session traffic.

5.13 Debug support

The reduced number of external buses enabled by the move to multicore chips greatly simplifies board level lay-out and eliminates many concerns over signal integrity. Even though the board designer may embrace multicore CPUs, software engineers have real concerns over the potential to lose debug visibility. Despite the problems external buses can cause for the hardware engineer, they provide software developers with the ultimate confirmation that the proper instructions and data are passing between processing elements.

Processing on a multicore chip with shared caches and peripherals also leads to greater concurrency and an increased potential for unintended interactions between device components. To ensure that software developers have the same or better visibility into the device as they would with multiple discrete communications processors, Freescale developed an Advanced Multicore Debug Architecture.

The debugging and performance monitoring capability enabled by the device hardware coexists within a debug ecosystem that offers a rich variety of tools at different levels of the hardware/software stack. Software development and debug tools from Freescale (CodeWarrior), as well as third-party vendors, provide a rich set of options for configuring, controlling, and analyzing debug and performance related events.

6 Conclusion

Featuring 24 virtual cores, and based on the dual-threaded e6500 Power Architecture core, the T4240 processor, along with its 16 (T4160) and 8 (T4080) virtual-core variants, offers frequencies up to 1.8 GHz, large caches, hardware acceleration, and advanced system peripherals. All three devices target applications that benefit from consolidation of control and data plane processing in a single chip. In addition, each e6500 core implements the Freescale AltiVec technology SIMD engine, dramatically boosting the performance of math-intensive algorithms without using additional DSP components on the board. A wide variety of applications can benefit from the processing, I/O integration, and power management offered for the T4 series processors. Similar to other QorIQ devices, the T4 family processors' high level of integration offers significant space, weight, and power benefits compared to multiple discrete devices. Freescale also offers fully featured development support, which includes the QorIQ T4240 QDS Development System, QorIQ T4240 Reference Design Board, Linux SDK for QorIQ Processors, as well as popular operating systems and development tools from a variety of vendors. See the Freescale website for the latest information on tools and SW availability.

For more information about the QorIQ T4 family, contact your Freescale sales representative.

Appendix A T4160

A.1 Introduction

The T4160 is a lower power version of the T4240. The T4160 combines eight dual threaded Power Architecture e6500 cores and two memory complexes (CoreNet platform cache and DDR3 memory controller) with the same high-performance datapath acceleration, networking, and peripheral bus interfaces.

This figure shows the major functional units within the chip.

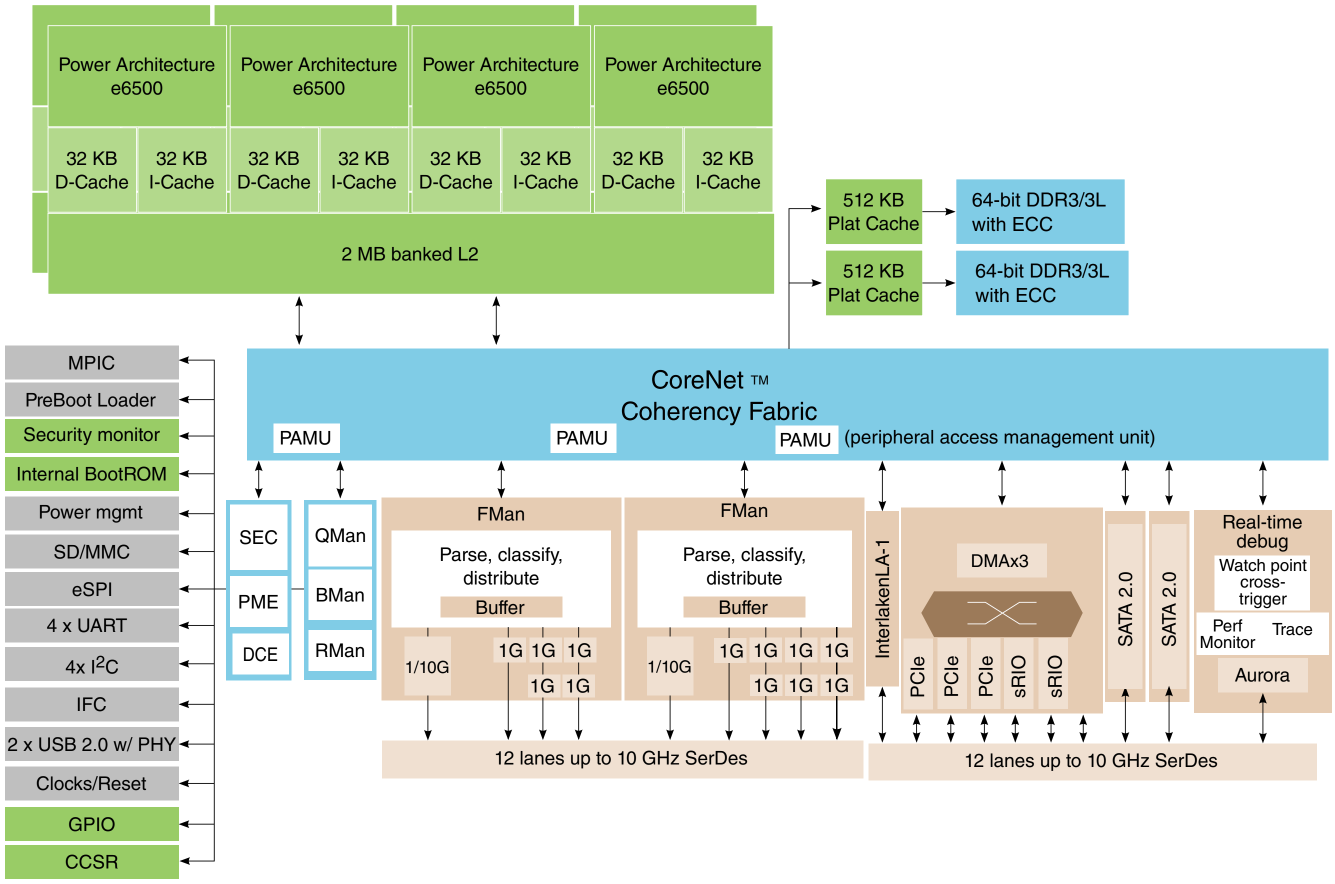


Figure A-1. T4160 block diagram

A.2 Overview of differences between T4240 and T4160

Table A-1. Differences between T4240 and T4160

Feature	T4240	T4160
Cores		
Number of physical cores	12	8
Number of threads	24	16
Number of clusters	3	2
Memory subsystem		
Total CPC memory	3 x 512 KB	2 x 512 KB
Number of DDR controllers	3	2
Peripherals		
Number of Frame Managers	2	2
Total number of Anyspeed MACs	8 per Frame Manager	6 (FMan1) and 8 (FMan2)

Table continues on the next page...

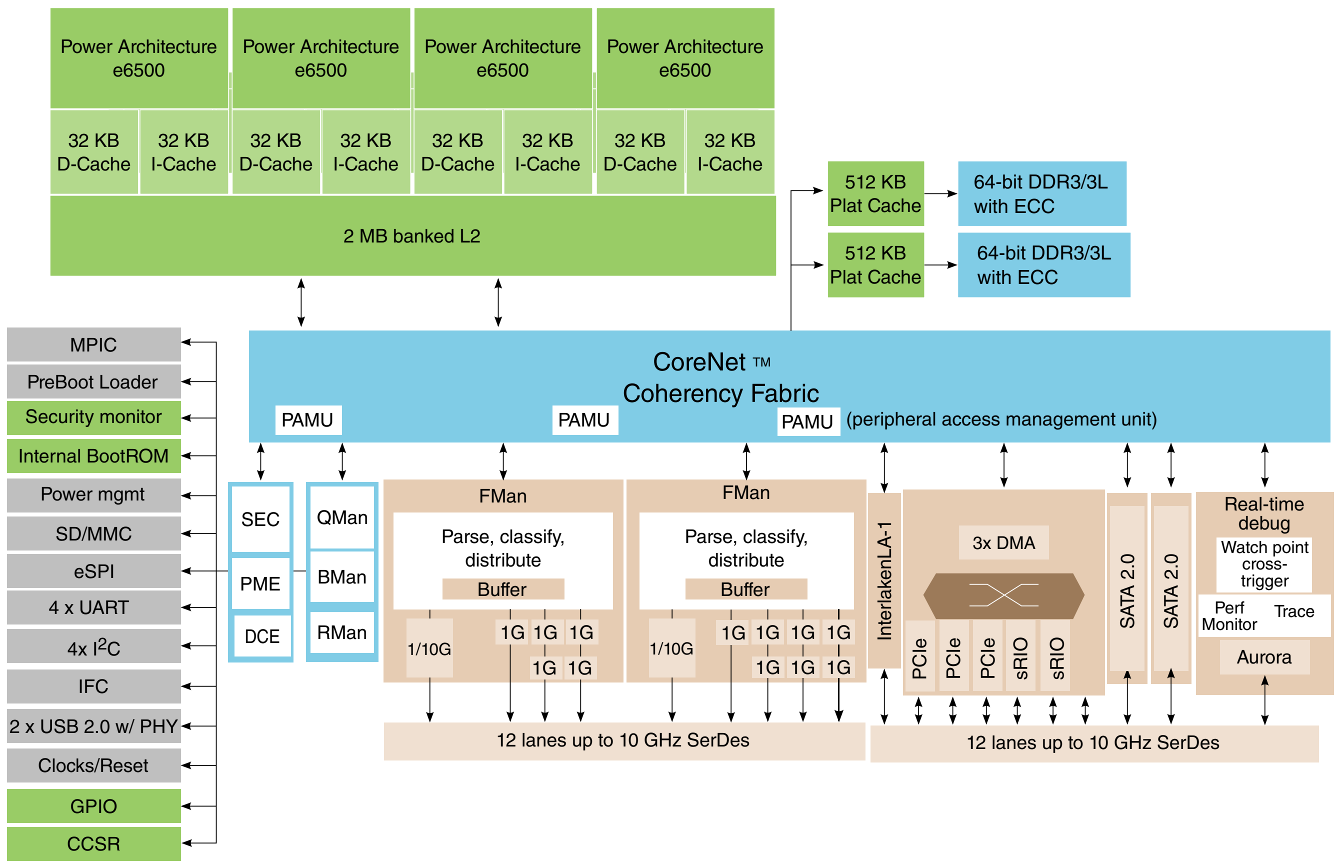


Figure B-1. T4080 block diagram

B.2 Overview of differences between T4160 and T4080

Table B-1. Differences between T4160 and T4080

Feature	T4160	T4080
Cores		
Number of physical cores	8	4
Number of threads	16	8
Number of clusters	2	1

Appendix C Revision history

C.1 Revision history

This table provides a revision history for this document.