



Welcome to [E-XFL.COM](http://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	<a href="https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=p2040nse7mmc">https://www.e-xfl.com/pro/item?MUrl=&amp;PartUrl=p2040nse7mmc</a>

## 2 P2040 Multicore Processing Options

The four P2040 cores can run either on an OS or run OS-less using a simple scheduler.

### 2.1 Running on an OS

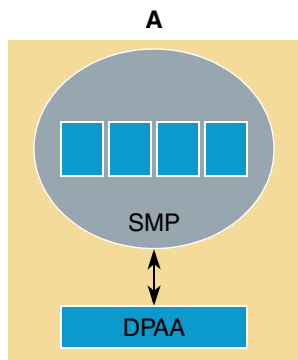
There are different multi-processing options with the P2040 cores running on an OS:

- Four-core, asymmetric multi-processing
  - Four copies of the same uni-processor operating system
  - or
  - Up to four different uni-processor operating systems
- Four-core, symmetric multi-processing (SMP)
- Mixed symmetric and asymmetric multi-processing. For example,  $N$  cores running in SMP mode, while the remainder of the cores operate asymmetrically with up to  $4-N$  different OSes

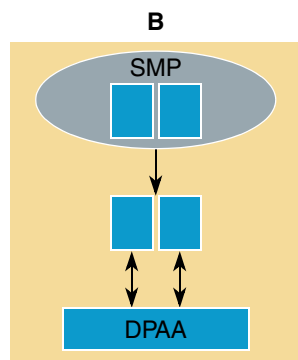
CPU cores operating asymmetrically can be run at asynchronous clock rates. Each processor can source its input clock from one of the multiple PLLs inside the P2040. This allows each core to operate at the minimum frequency required to perform its assigned function, saving power. The cores are also capable of running at half and quarter ratios of their input PLL frequency, and can switch between PLLs and ratios nearly instantaneously. This allows lightly utilized CPUs to be slowed (under software control) for power savings, rather than performing more complex task migration operations.

### 2.2 DPAA Multicore Processing Use Cases

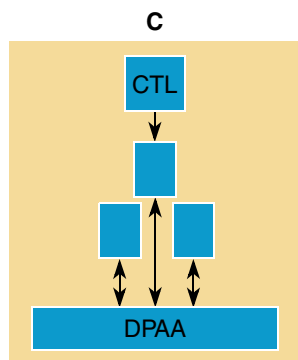
[Figure 3](#) shows several multicore processing use cases and the potential interaction with the Data Path Acceleration Architecture (DPAA).



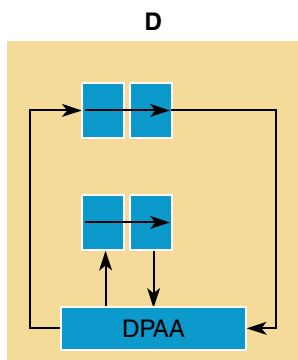
All CPUs are running a single operating system, with any specialization of CPU function occurring through OS techniques such as Task Affinity. The I/Os and acceleration hardware are under the control of the SMP OS. Typically all CPUs operate at the same frequency.



Some number of the cores are operated as an SMP cluster, most likely running high complexity control plane operations. The control plane configures and manages the remaining processors, which are running individual copies of an RTOS or scheduler to perform dataplane operations. In this use case, the SMP CPUs typically operate at the same frequency, the remaining CPUs can run at a different frequency from the SMP CPUs, and even from each other.



A single CPU is used as the control processor, configuring and managing the other three processors, which are running individual copies of an RTOS or scheduler, as in B. CPU operating frequencies are an independent parameter.



All CPUs are used for datapath operations, here shown as two sets of pipelined functions, each interacting independently with the I/Os and accelerators. Operating frequencies for each CPU in the pipeline can be set independently.

**Figure 3. CPU Usage Use Cases**

Figure 4 shows an additional use case, which involves the use of one of the CPUs as an I/O processor. The DPAA can greatly simplify and accelerate processing for packets entering the system by means of the

Ethernet interfaces. For systems requiring external ASICs or legacy network interface cards in the high-performance datapath, system developers can allocate a CPU to help interwork between the native data buffers used by PCI Express- or Serial RapidIO-based network interfaces and the data buffers used by the datapath acceleration hardware.

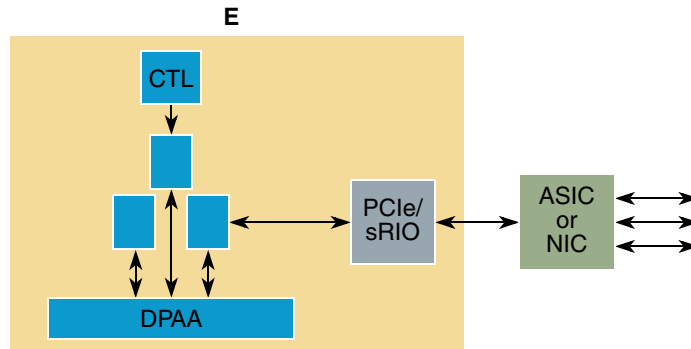


Figure 4. IO Processor Managing PCIe/Serial RapidIO-Based Network Interfaces

### 3 P2040 Features

#### 3.1 P2040 Block Diagram

Figure 5 shows the major functional units within the P2040.

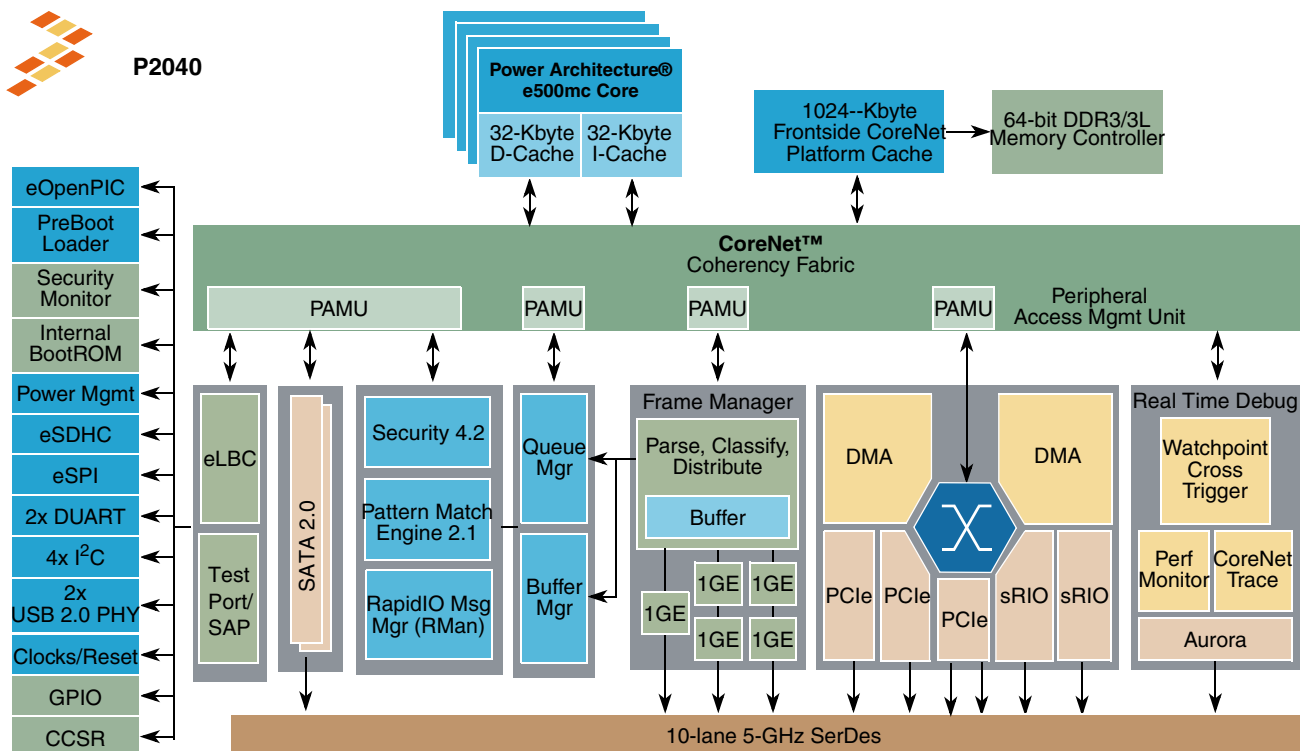


Figure 5. P2040 Preliminary Block Diagram

## 3.2 P2040 Features Summary

The P2040 SoC includes the following functions and features:

- Four e500mc cores built on Power Architecture technology
- Three levels of instructions:
  - User
  - Supervisor
  - Hypervisor
- Independent boot and reset
- Secure boot capability
- 1-Mbyte shared CoreNet platform cache (CPC)
- Hierarchical interconnect fabric
  - CoreNet fabric supporting coherent and non-coherent transactions with prioritization and bandwidth allocation amongst CoreNet end-points
  - Queue manager fabric supporting packet-level queue management and quality of service scheduling
- One 64-bit DDR3/3L SDRAM memory controller with ECC and chip-select interleaving support
- Data Path Acceleration Architecture (DPAA) incorporating acceleration for the following functions:
  - Frame management for packet parsing, classification, and distribution
  - Queue management for scheduling, packet sequencing, and congestion management
  - Hardware buffer management for buffer allocation and de-allocation
  - Encryption/decryption (SEC 4.2)
  - RegEx pattern matching (PME 2.1)
- Ethernet interfaces
  - Five 1 Gbps or four 2.5 Gbps Ethernet controllers
- High speed peripheral interfaces
  - Three PCI Express 2.0 controllers/ports running at up to 5 GHz
  - Two Serial RapidIO® controllers/ports (version 1.3 with features of 2.1) running at up to 5 GHz
    - RapidIO message manager (RMan) with Type 5–6 and Type 8–11 support
  - Dual SATA 2.0 interfaces
- Additional peripheral interfaces
  - Two USB 2.0 controllers with integrated PHY
  - SD/MMC controller (eSDHC)
  - Enhanced SPI controller
  - Four I<sup>2</sup>C controllers
  - Dual DUARTs

- 10 SerDes lanes to 5 GHz
- Enhanced local bus controller (eLBC)
- Multicore programmable interrupt controller (MPIC)
- Two 4-channel DMA engines

### 3.3 Benefits of the P2040

The P2040’s e500mc cores can be combined as a fully-symmetric, multi-processing, system-on-a-chip, or they can be operated with varying degrees of independence to perform asymmetric multi-processing. Full processor independence, including the ability to independently boot and reset each e500mc core, is a defining characteristic of the P2040. 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.

### 3.4 Benefits of the Data Path Acceleration Architecture (DPAA)

While the four Power Architecture cores offer a major leap in available processor performance in many throughput-intensive, packet-processing networking applications, raw processing power is not enough to achieve multi-Gbps data rates. To address this, the P2040 uses Freescale’s Data Path Acceleration Architecture (DPAA) (see [Section 3.11, “Data Path Acceleration Architecture \(DPAA\)”](#)), which significantly reduces data plane instructions per packet, enabling more CPU cycles to work on value-added services rather than repetitive low-level tasks. Combined with specialized accelerators for cryptography and pattern matching, the P2040 allows the user’s software to perform complex packet processing at high data rates.

### 3.5 P2040 Critical Performance Parameters

[Table 1](#) lists key performance indicators that define a set of values used to measure P2040 operation.

**Table 1. P2040 Critical Performance Parameters**

Indicator	Values(s)
Top speed bin e500mc core frequency	1.2 GHz
Maximum memory data rates	1.2 GHz (DDR3/3L) <sup>1</sup> <ul style="list-style-type: none"> <li>• 1.5-V for DDR3</li> <li>• 1.35-V for DDR3L</li> </ul>
Local bus	<ul style="list-style-type: none"> <li>• 3.3 V</li> <li>• 2.5 V</li> <li>• 1.8 V</li> </ul>
Operating junction temperature range	0–105 C with the option for –40 to 105 C
Package	780-pin FC-PBGA (flip-chip plastic ball grid array)

**Notes:**

<sup>1</sup> Conforms to JEDEC standard

- Eliminates the need to copy data from a source context into a kernel context, change to destination address space, then copy the data to the destination address space or alternatively to map the user space into the kernel address space

### 3.6.2 CoreNet Platform Cache (CPC)

The P2040 contains 1-Mbyte of shared CoreNet platform cache (CPC). The key features of the CPC include the following:

- Configurable as write-back or write-through
- Pseudo LRU replacement algorithm
- ECC protection
- 64-byte coherency granule
- 1 cache line read 64 bytes per cycle at 600 MHz, 0.3 terabits/sec read bandwidth
- 32-way cache array configurable to any of several modes on a per-way basis.
  - Unified cache, I-only, D-only
  - I/O stash (configurable portion of each packet copied to CPC on write to main memory)
    - stashing of all transactions and sizes supported
    - explicit (CoreNet signalled) and implicit (address range based) stash allocation
  - Addressable SRAM (32-Kbyte granularity)

### 3.7 CoreNet Fabric and Address Map

The CoreNet fabric is Freescale's next generation Front-side Interconnect Standard for multicore products, and provides the following:

- A highly concurrent, fully cache coherent, multi-ported fabric
- Point-to-point connectivity with flexible protocol architecture allows for pipelined interconnection between CPUs, platform caches, memory controllers, and I/O and accelerators at up to 600 MHz
- The CoreNet fabric has been designed to overcome bottlenecks associated with shared bus architectures, particularly address issue and data bandwidth limitations. The P2040's multiple, parallel address paths allow for high address bandwidth, which is a key performance indicator for large coherent multicore processors
- Eliminates address retries, triggered by CPUs being unable to snoop within the narrow snooping window of a shared bus. This results in the P2040 having lower average memory latency

The flexible P2040's 36-bit, physical address map consists of local space and external address space. For the local address map, 32 local access windows (LAWs) define mapping within the local 36-bit (64-Gbyte) address space. Inbound and outbound translation windows can map the P2040 into a larger system address space such as the RapidIO or PCIe 64-bit address environment. This functionality is included in the address translation and mapping units (ATMUs).

## 3.8 Memory Complex

The P2040 memory complex consists of one DDR controller for main memory, and the memory controllers associated with the Enhanced Local Bus Controller (eLBC).

### 3.8.1 DDR Memory Controller

The P2040 DDR memory controllers have the following functionalities:

- Supports DDR3/3L SDRAM. The P2040 supports chip-select interleaving within a controller.
- The P2040 can be configured to retain the currently active SDRAM page for pipelined burst accesses. Page mode support of up to 32 simultaneously open pages can dramatically reduce access latencies for page hits. Depending on the memory system design and timing parameters, page mode can save up to 10 memory clock cycles for subsequent burst accesses that hit in an active page.
- Using ECC, the P2040 detects and corrects all single-bit errors and detects all double-bit errors and all errors within a nibble.
- Upon detection of a loss of power signal from external logic, the DDR controllers can put compliant DDR SDRAM DIMMs into self-refresh mode, allowing systems to implement battery-backed main memory protection.
- Supports initialization bypass feature for use by system designers to prevent re-initialization of main memory during system power-on after an abnormal shutdown
- Supports active zeroization of system memory upon detection of a user-defined security violation

### 3.8.2 PreBoot Loader (PBL) and Nonvolatile Memory Interfaces

The PreBoot Loader (PBL) is a new logic module that operates similarly to an I<sup>2</sup>C boot sequencer but on behalf of a larger number of interfaces.

The PBL's functions include the following:

- Simplifies boot operations, replacing pin strapping resistors with configuration data loaded from nonvolatile memory
- Uses the configuration data to initialize other system logic and to copy data from low speed memory interfaces (I<sup>2</sup>C, eLBC, SPI, and SD/MMC) into fully initialized DDR or the 1-Mbyte front-side cache.
- Releases CPU 0 from reset, allowing the boot processes to begin from fast system memory.

The nonvolatile memory interfaces accessible by the PBL are as follows:

- The eLBC may be accessed by software running on the CPUs following boot; it is not dedicated to the PBL. It also can be used for both volatile (SRAM) and nonvolatile memory as well as a control and low-performance data port for external memory-mapped devices. See [Section 3.8.2.1, “Enhanced Local Bus Controllers \(eLBC\).”](#)
- The serial memory controllers may be accessed by software running on the CPUs following boot; they are not dedicated to the PBL. See [Section 3.8.2.2, “Serial Memory Controllers.”](#)



### 3.8.2.1 Enhanced Local Bus Controllers (eLBC)

The enhanced local bus controller (eLBC) port connects to a variety of external memories, DSPs, and ASICs.

Key features of the eLBC include the following:

- Multiplexed 32-bit address and 16-bit data bus operating at up to 75 MHz
- Four chip selects for four external slaves
- Up to eight-beat burst transfers
- 8-, 16-bit port sizes controlled by an internal memory controller
- Three protocol engines on a per-chip-select basis
- Parity support
- Default boot ROM chip select with configurable bus width (8-, 16-bit)
- Support for parallel NAND and NOR flash

Three separate state machines share the same external pins and can be programmed separately to access different types of devices. Some examples are as follows:

- The general-purpose chip-select machine (GPCM) controls accesses to asynchronous devices using a simple handshake protocol.
- The user-programmable machine (UPM) can be programmed to interface to synchronous devices or custom ASIC interfaces.
- The NAND flash control machine (FCM) further extends interface options.
- Each chip select can be configured so that the associated chip interface is controlled by the GPCM, UPM, or FCM controller.

All controllers can be enabled simultaneously. The eLBC internally arbitrates among the controllers, allowing each to read or write a limited amount of data before allowing another controller to use the bus.

### 3.8.2.2 Serial Memory Controllers

In addition to the parallel NAND and NOR flash supported by means of the eLBC, the P2040 supports serial flash using SPI and SD/MMC card interfaces. The SD/MMC controller includes a DMA engine, allowing it to move data from serial flash to external or internal memory following straightforward initiation by software.

## 3.9 Universal Serial Bus (USB) 2.0

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

Key features of the USB 2.0 controller include the following:

- Complies with USB specification, Rev. 2.0
- Supports high-speed (480 Mbps), full-speed (12 Mbps), and low-speed (1.5 Mbps) operations

## 3.10.2 Serial RapidIO Interfaces

### 3.10.2.1 Serial RapidIO Interface

The Serial RapidIO interface is based on the *RapidIO Interconnect Specification, Revision 1.3* with features from 2.1. RapidIO is a high-performance, point-to-point, low-pin-count, packet-switched system-level interconnect that can be used in a variety of applications as an open standard. The rich feature set includes high data bandwidth, low-latency capability, and support for high-performance I/O devices as well as message-passing and software-managed programming models. Receive and transmit ports operate independently, and with 2 x 4 Serial RapidIO controllers, the aggregate theoretical bandwidth is 32 Gbps.

Key features of the Serial RapidIO interface unit include the following:

- Support for *RapidIO Interconnect Specification, Revision 1.3* (all transaction flows and priorities)
- 1x, 2x, and 4x LP-serial link interfaces, with transmission rates of 2.5, 3.125, or 5.0 Gbaud (data rates of 2.0, 2.5, or 4.0 Gbps) per lane.
- Auto-detection of 1x, 2x, or 4x mode operation during port initialization
- 34-bit addressing and up to 256-byte data payload
- Receiver-controlled flow control
- RapidIO error injection
- Internal LP-serial and application interface-level loopback modes

### 3.10.2.2 RapidIO Message Manager (RMan)

The key features of the RapidIO message manager (RMan) include the following:

- Manages two inbox/outbox mailboxes (queues) for data and one doorbell message structure
- Can multi-cast a single-segment 256-byte message to up to 32 different destination DevIDs
- Has four outbound segmentation units supporting RapidIO Type 5–6 and Type 8–11

### 3.10.3 Serial ATA (SATA) 2.0 Controllers

The key features of each of the two SATA include the following:

- Designed to comply with Serial ATA 2.6 Specification
- Supports host SATA I per spec Rev 1.0a
  - OOB
  - Port multipliers
  - ATAPI 6+
  - Spread spectrum clocking on receive
- Support for SATA II extensions
  - Asynchronous notification
  - Hot plug including asynchronous signal recovery
  - Link power management

- Native command queuing
- Staggered spin-up and port multiplier support
- Support for SATA I and II data rates (1.5 and 3.0 Gbaud)
- Standard ATA master-only emulation
- Includes ATA shadow registers
- Implements SATA superset registers (SError, SControl, SStatus)
- Interrupt driven
- Power management support
- Error handling and diagnostic features
  - Far end/near end loopback
  - Failed CRC error reporting
  - Increased ALIGN insertion rates
  - Scrambling and CONT override

### 3.11 Data Path Acceleration Architecture (DPAA)

The DPAA provides the infrastructure to support simplified sharing of networking interfaces and accelerators by multiple CPU cores. These resources are abstracted into enqueue/dequeue operations by means of a common DPAA Queue Manager (QMan) driver. Beyond enabling multicore resource sharing, the DPAA significantly reduces software overheads associated with high-touch packet-forwarding operations. Examples of the types of packet-processing services this architecture is optimized to support are as follows:

- Traditional routing and bridging
- Firewall
- VPN termination for both IPsec and SSL VPNs
- Intrusion detection/prevention (IDS/IPS)
- Network anti-virus (AV)

The DPAA generally leaves software in control of protocol processing, while reducing CPU overheads through off-load functions, which fall into two, broad categories:

- [Section 3.11.1, “Packet Distribution and Queue/Congestion Management”](#)
- [Section 3.11.2, “Accelerating Content Processing”](#)

### 3.11.1 Packet Distribution and Queue/Congestion Management

Table 2 lists some packet distribution and queue/congestion management offload functions.

**Table 2. P2040 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.

### 3.11.2 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. Table 3 lists some specific content-processing accelerators on the P2040.

**Table 3. P2040 Content-Processing Accelerators**

Interface	Definition
SEC 4.2	Crypto-acceleration for protocols such as IPsec, SSL, and 802.16
PME 2.1	Regex style pattern matching for unanchored searches, including cross-packet stateful patterns

**Note:** Prior versions of the SEC and PME are integrated into multiple members of the PowerQUICC and QorIQ family. Both of these engines have been enhanced to work within the DPAA, and also upgraded in both features and performance.

- Section 3.11.4.5, “Pattern Matching Engine (PME 2.1)”

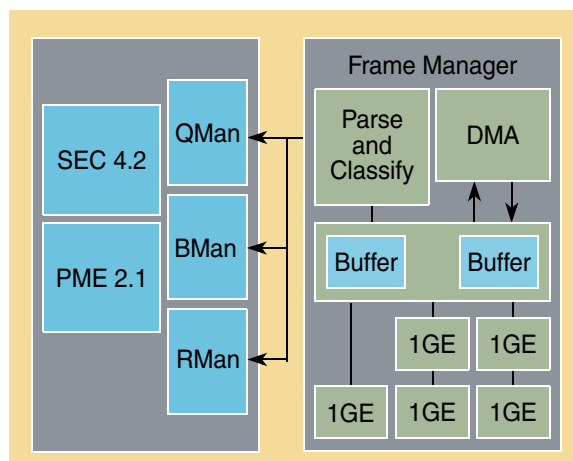


Figure 6. QorIQ Data Path Acceleration Architecture (DPAA)

### 3.11.4.1 Frame Manager (FMan)

The Frame Manager (FMan) combines the Ethernet network interfaces with packet distribution logic to provide intelligent distribution and queuing decisions for incoming traffic at line rate (7.5 Mpps). This integration allows the FMan to perform configurable parsing and classification of the incoming frame with the purpose of selecting the appropriate input frame queue for expedited processing by a CPU or pool of CPUs.

#### 3.11.4.1.1 FMan Network Interfaces

The P2040 FMan integrates five datapath, tri-speed Ethernet controllers (dTSECs). The Ethernet controllers support the following:

- Programmable CRC generation and checking
- RMON statistics
- Jumbo frames of up to 9.6 Kbytes

They are designed to comply with IEEE Std 802.3@, IEEE 802.3u, IEEE 802.3x, IEEE 802.3z, IEEE 802.3ac, IEEE 802.3ab, and additionally the 1Gbps MACs support IEEE-1588 v2 (clock synchronization over Ethernet).

The dTSECS are capable of full- and half-duplex Ethernet support (1000 Mbps supports only full duplex). It supports IEEE 802.3 full-duplex flow control (automatic PAUSE frame generation or software-programmed PAUSE frame generation and recognition).

The FMan can support line rate parsing and classification on an aggregate of 12 Gbps.

#### 3.11.4.1.2 FMan Parse Function

The primary function of the packet parse logic is to identify the incoming frame for the purpose of determining the desired treatment to apply. This parse function can parse many standard protocols,

including options and tunnels, and supports a generic configurable capability to allow proprietary or future protocols to be parsed.

There are several types of parser headers, shown in [Table 5](#).

**Table 5. 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. 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.

### 3.11.4.1.3 FMan Distribution and Policing

After parsing is complete, there are two options for treatment (see [Table 6](#)).

**Table 6. Post-Parsing Treatment Options**

Treatment	Function	Benefits
Hash	<ul style="list-style-type: none"> <li>Hashes selected fields in the frame as part of a spreading mechanism</li> <li>The result is a specific frame queue identifier.</li> <li>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).</li> </ul>	Useful when spreading traffic while obeying QoS constraints is required
Classification look-up	<ul style="list-style-type: none"> <li>Looks up certain fields in the frame to determine subsequent action to take, including policing</li> <li>The FMan contains internal memory that holds small tables for this purpose.</li> <li>The user configures the sets of lookups to perform, and the parse results dictate which one of those sets to use.</li> <li>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.</li> </ul>	<ul style="list-style-type: none"> <li>Useful when hash distribution is insufficient and a more detailed examination of the frame is required</li> <li>Can determine whether policing is required and the policing context to use</li> </ul>

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 for mitigating Distributed Denial of Service Attack (DDOS).
- The policing is based on two-rate-three-color marking algorithm (RFC2698). The sustained and peak rates as well as the burst sizes are user-configurable. Hence, the policing function can

rate-limit traffic to conform to the rate the flow is mapped to at flow set-up time. By prioritizing and policing traffic prior to software processing, CPU cycles can be focused on the important and urgent traffic ahead of other traffic.

### 3.11.4.2 Queue Manager (QMan)

The Queue Manager (QMan) is the main component in the DPAA that allows for simplified sharing of network interfaces and hardware accelerators by multiple CPU cores. It also provides a simple and consistent message and data passing mechanism for dividing processing tasks amongst multiple CPU cores. The QMan features are as follows:

- Common interface between software and all hardware
  - Controls the prioritized queuing of data between multiple processor cores, network interfaces, and hardware accelerators
  - Supports both dedicated and pool channels, allowing both push and pull models of multicore load spreading
- Atomic access to common queues without software locking overhead
- Mechanisms to guarantee order preservation with atomicity and order restoration following parallel processing on multiple CPUs
- Two-level queuing hierarchy with one or more Channels per Endpoint, eight work queues per Channel, and numerous frame queues per work queue
- Priority and work conserving fair scheduling between the work queues and the frame queues
- Lossless flow control for ingress network interfaces
- Congestion avoidance (RED/WRED) and congestion management with tail discard and up to 256 congestion groups with each group having user-configured number of frame queues.

### 3.11.4.3 Buffer Manager (BMan)

The buffer manager (BMan) manages pools of buffers on behalf of software for both hardware (accelerators and network interfaces) and software use. The BMan features are as follows:

- Common interface for software and hardware
- Guarantees atomic access to shared buffer pools
- Supports 32 buffer pools. Software and hardware buffer consumers can request both different size buffers and buffers in different memory partitions.
- Supports depletion thresholds with congestion notifications
- On-chip per pool buffer stockpile to minimize access to memory for buffer pool management
- LIFO (last in first out) buffer allocation policy that optimizes cache usage and allocation

### 3.11.4.4 Security Engine (SEC 4.2)

The SEC 4.2 is QorIQ's fourth generation crypto-acceleration engine. In addition to off-loading cryptographic algorithms, the SEC 4.2 offers header and trailer processing for several established security protocols. The SEC 4.2 includes several Descriptor Controllers (DECOs), which are updated versions of

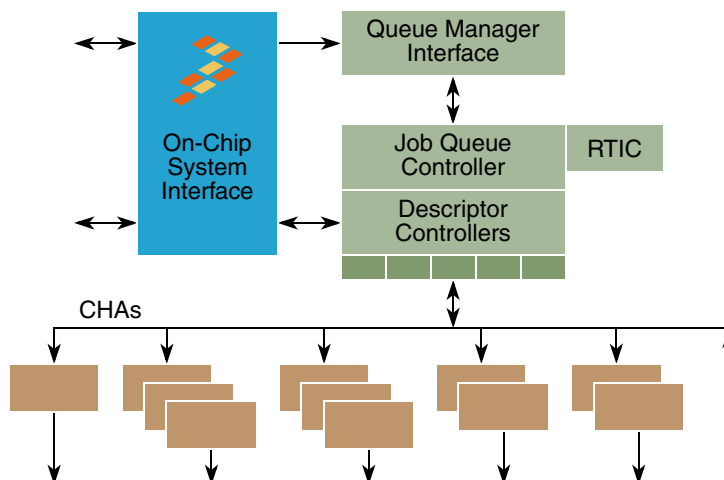


Figure 7. SEC 4.2 Block Diagram

### 3.11.4.5 Pattern Matching Engine (PME 2.1)

The PME is a self-contained hardware module capable of autonomously scanning data from streams for patterns that match a specification in a database dedicated to it. The PME 2.1 is an updated version of the PME used in previous members of the PowerQUICC family. Specific updates include the following:

- QMan interface supporting the DPAA Queue Interface Driver
- 2x increase in the number of patterns supported (16 Kbytes to 32 Kbytes)
- Increase in number of stateful rules supported (8 Kbytes to 16 Kbytes)
- Raw scanning performance is ~ 4Gbps.

Patterns that can be recognized, or “matched,” by the PME are of two general forms:

- Byte patterns are simple matches such as “abcd123” existing in both the data being scanned and in the pattern specification database.
- Event patterns are a sequence of multiple byte patterns. In the PME, event patterns are defined by stateful rules.

#### 3.11.4.5.1 PME Regular Expressions (Regex)

The PME specifies patterns of bytes as regular expressions (regex). The P2040 (by means of an online or offline process) converts Regex patterns into the PME’s pattern specification database. Generally, there is a one-to-one mapping between a regex and a PME byte pattern. The PME’s use of regex pattern matching offers built-in case-insensitivity and wildcard support with no pattern explosion, while the PME’s NFA-style architecture offers fast pattern database compilation and fast incremental updates. Up to 32,000 regex patterns are supported, each up to 128 bytes long. The 32,000 regex patterns can be combined by means of stateful rules to detect a far larger set of event patterns. Comparative compilations against DFA style regex engines have shown that 300,000 DFA pattern equivalents can be achieved with ~8000 PME regexes with stateful rules.



### 3.12.2 e500mc MMU and Embedded Hypervisor

The P2040's first line of defense against unintended interactions amongst the multiple CPUs/OSes is each e500mc core's MMU, which are configured to determine which addresses in the global address map the CPU is able to read or write. If a particular resource (such as a portion of memory or a peripheral device) is dedicated to a single CPU, that CPU's MMU is configured to allow access to those addresses (on 4-Kbyte granularity); other CPU MMUs are not configured for access to the other CPU's private memory range. When two CPUs need to share resources, both of their MMUs are configured to 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 SoC, achieving strong partitioning shouldn't require the developer to map functions onto cores 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 e500mc MMU also includes embedded Hypervisor extensions.

Each e500mc MMU supports three levels of instructions:

- User
- Supervisor (OS)
- Hypervisor: An embedded Hypervisor micro-kernel (provided by Freescale as source code) runs unobtrusively beneath the various OSes running on the CPUs, 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 core attempts to overwrite important device configuration registers (including CPU MMUs), the embedded Hypervisor blocks the write. Other examples of embedded Hypervisor managed resources are high- and low-speed peripheral interfaces (PCIe, UART) if those resources are not dedicated to a single CPU/partition.

### 3.12.3 Peripheral Access Management Unit (PAMU)

The P2040 includes a distributed function collectively referred to as the peripheral access management unit (PAMU), which provides address translation and access control for all bus masters in the system (PME, SEC, FMan, and so on). The PAMU access control can be one of the following:

- Absolute—The FMan, PME, SEC, and other bus masters can never access memory range XYZ.
- Conditional—Based on the Partition ID of the CPU that programmed the bus master

Being MMU-based, the embedded Hypervisor is only able to stop unauthorized software access attempts. Internal components with bus mastering capability also need to be prevented from reading and writing to specific memory regions. These devices do not spontaneously generate access attempts, but, if programmed to do so by buggy or malicious software, any of them could overwrite sensitive configuration registers and crash the system.

### 3.12.4 Secure Boot and Sensitive Data Protection

The e500mc MMUs and PAMU allow the P2040 to enforce a consistent set of memory access permissions on a per-partition basis. When combined with embedded Hypervisor for safe sharing of resources, the P2040 becomes highly resilient when poorly tested or malicious code is run. For system developers building high reliability/high security platforms, rigorous testing of code of known origin is the norm.

#### 3.12.4.1 Secure Boot Option

The system developer digitally signs the code to be executed by the CPU coming out of reset, and the P2040 ensures that only an unaltered version of that code runs on the platform. The P2040 offers both boot time and run time code authenticity checking and configurable consequences when the authenticity check fails.

#### 3.12.4.2 Sensitive Data Protection Option

The P2040 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 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 P2040 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 4.2 for decryption of session traffic.

### 3.13 Advanced Power Management

The P2040's advanced power management capabilities are based around fine-grained static clock control and software-controlled dynamic frequency management.

#### 3.13.1 Saving Power by Managing Internal Clocks

Dynamic voltage and frequency scaling (DVFS) are useful techniques for reducing typical/average power and maximizing battery life in laptop environments, but embedded applications must be designed for rapid response to bursts of traffic and max power under worst-case environmental conditions. While the P2040 does not implement DVFS in the PC sense, it does actively manage internal clocks to avoid wasting energy. Clock signals are disabled to idle components, reducing dynamic power. These blocks can return to full operating frequency on the clock cycle after work is dispatched to them.

The P2040 also supports (under software control) dynamic changes to CPU operating frequencies and voltages. Each CPU sources its input clock from one of two independent PLLs inside the P2040. Each CPU can also source its input clock from an integer frequency divider from two of the three independent PLLs. CPUs can switch their source PLL, and their frequency divider glitchlessly and nearly instantaneously. This allows each core to operate at the minimum frequency required to perform its assigned function, saving power.

### 3.13.2 Turning Off Unneeded Clocks

Fine-grained static control allows developers to turn off the clocks to individual logic blocks within the SoC that the system has no need for. Based on a finite number of SerDes, it is expected that any given application will have some Ethernet MACs, PCIe, or Serial RapidIO controllers inactive. These blocks can be disabled by means of the DEVDIS register. Re-enabling clocks to a logic block requires an SoC reset, which makes this type of power management operation infrequent (effectively static).

### 3.13.3 Avoiding Full System Failure Due to Thermal Overload

Changing PLL frequency dividers (/2, /4) can be used to achieve large and rapid reductions in dynamic power consumptions, and with the help of external temperature detection circuitry, can serve as a thermal overload protection scheme. If the junction temperature or system ambient temperature of the P2040 achieves some critical level, external temperature detection circuitry can drive a high-priority interrupt into the P2040, causing it to reduce selected CPU frequencies by half or more. This allows the system to continue to function in a degraded mode, rather than failing entirely. This technique is much simpler than turning off selected CPUs, which can involve complex task migration in an AMP system. When system temperatures have been restored to safe ranges, all CPUs can be returned to normal frequency within a few clock cycles.

When less drastic frequency changes are desired, software can switch the CPU to a slower speed PLL, such as 1 GHz versus 1.5 GHz. Many cores could be switched to a slower PLL during periods of light traffic, with the ability to immediately return those cores to the full rate PLL should traffic suddenly increase. The more traditional Power Architecture single-core power management modes (such as Core Doze, Core Nap, and Core Sleep) are also available in the e500mc.

## 3.14 Debug Support

The reduced number of external buses enabled by the move to multicore SoCs greatly simplifies board level lay-out and eliminates many concerns over signal integrity. While 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 SoC with shared caches and peripherals also leads to greater concurrency and an increased potential for unintended CPU interactions. To ensure that software developers have the same or better visibility into the P2040 as they would with multiple discrete devices, the P2040 implements the debug architecture shown in this figure.

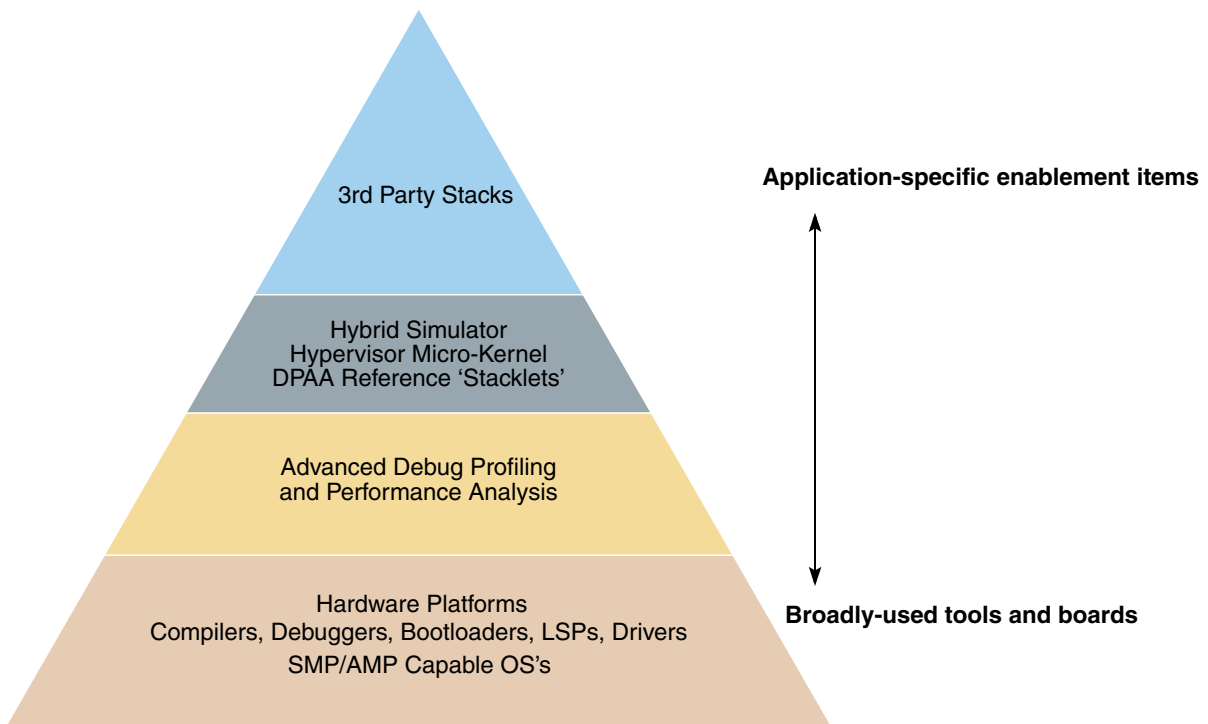
- Generates trace messages to Nexus port controller (NPC)
- Supports filtering of accesses of interest
  - Data Address Compare (4)
  - Data Value Compare (2)
  - Transaction Attribute Compare (2)

## 4 Developer Environment

Software developers creating solutions with the Power Architecture technology have long benefited from a vibrant support ecosystem, including high quality tools, OSes, and network protocol stacks. Freescale is working with our ecosystem partners to ensure that this remains the case for multicore, Power Architecture-based products, including the P2040.

The various levels of the developer environment are shown in [Figure 10](#), with the more broadly used tools and boards at the base of the pyramid, and increasingly application-specific enablement items at the top. Each level is described further in the following subsections:

- [Section 4.1, “Base of the Pyramid: Broadly-Used Tools and Boards”](#)
- [Section 4.2, “First Level of the Pyramid: Debug and Performance Analysis”](#)
- [Section 4.3, “Second Level of the Pyramid: Simulation, Hypervisor, and DPAA Reference “Stacklets”](#)
- [Section 4.4, “Top Level of the Pyramid: Application-Specific Enablement”](#)



**Figure 10. Levels of Developer Environment**

## 4.1 Base of the Pyramid: Broadly-Used Tools and Boards

### 4.1.1 Hardware Platforms

This category includes both development systems and the reference designs. Development systems are available from both Freescale and our partners, with some partner systems being offered with form factors and BOMs to support use as reference designs. Freescale development systems are supported by the open source GNU tool set including compilers, linkers, and debuggers.

### 4.1.2 Compilers, Debuggers, Bootloaders, LSPs, Drivers

In active partnership with the open source community and Linux distribution and support suppliers, these tools will be updated to fully and efficiently support the P2040.

### 4.1.3 SMP/AMP Capable OS's

Open source tools will be part of an overall P2040 development board Linux support package, which will include AMP and SMP versions of the Linux OS, and device drivers for the accelerators and networking and peripheral interfaces featured in the P2040. AMP Linux support will include the ability to boot multiple instances of Linux on different cores. Power Architecture ecosystem partners are committed to providing board support packages for the P2040.

## 4.2 First Level of the Pyramid: Debug and Performance Analysis

### 4.2.1 Advanced Debug

Advanced debug supports real-time trace analysis. It allows the developer to perform initial system bring-up and development, and is required to deal with the special challenges of software debugging and performance analysis in multicore systems.

### 4.2.2 Profiling and Performance Analysis

Freescale will bring tools support for profiling and performance analysis (such as enhanced statistics gathering) to the market both by means of our CodeWarrior line of tools and in partnership with industry standard tools suppliers.

## 4.3 Second Level of the Pyramid: Simulation, Hypervisor, and DPAA Reference “Stacklets”

### 4.3.1 Hybrid Simulator

In conjunction with Virtutech, Freescale will provide a hybrid simulator that combines both functional and performance measurement models of the P2040. The hybrid simulator allows the user to switch between “fast functional mode” and “detailed performance mode” with capabilities that include the following: