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	PowerPC e5500
Number of Cores/Bus Width	2 Core, 64-Bit
Speed	2.0GHz
Co-Processors/DSP	Security; SEC 4.2
RAM Controllers	DDR3, DDR3L
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	1Gbps (5), 10Gbps (1)
SATA	SATA 3Gbps (2)
USB	USB 2.0 + PHY (2)
Voltage - I/O	-
Operating Temperature	0°C ~ 105°C (TA)
Security Features	Boot Security, Cryptography, Secure Fusebox, Secure JTAG, Secure Memory, Tamper Detection
Package / Case	1295-BBGA, FCBGA
Supplier Device Package	1295-FCPBGA (37.5x37.5)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=p5020nse1qmb

1 P5020 Application Use Cases

1.1 Router Control Processor

The following figure shows the P5020 in a linecard control plane application, where the linecard is part of a high-end network router.

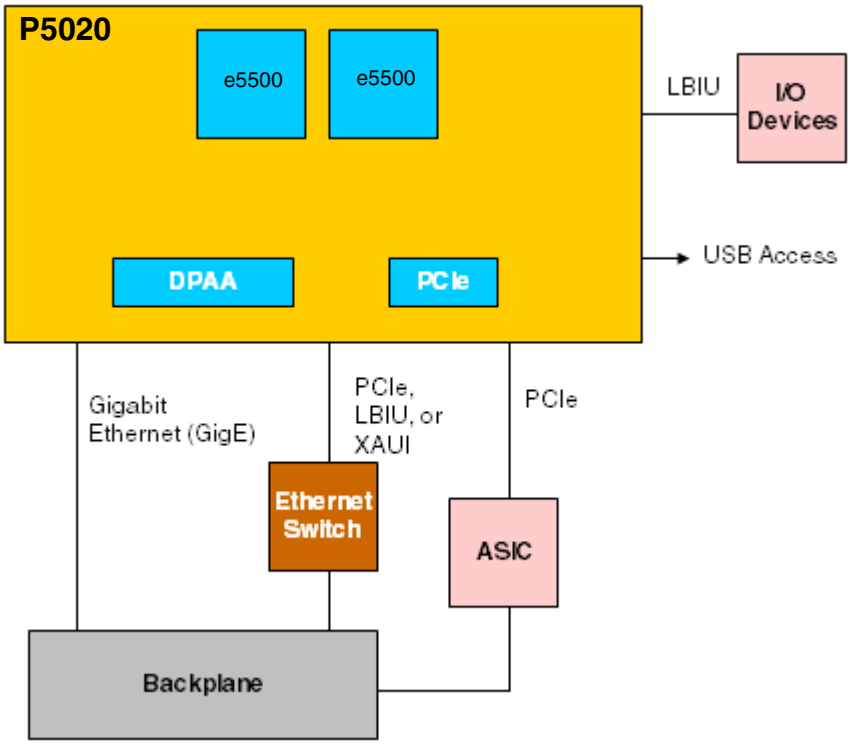

Figure 1. Control Plane Processor for a Router

1.2 DSP Farm Control Processor

The following figure shows a DSP farm enabled by the P5020 utilizing serial RapidIO.

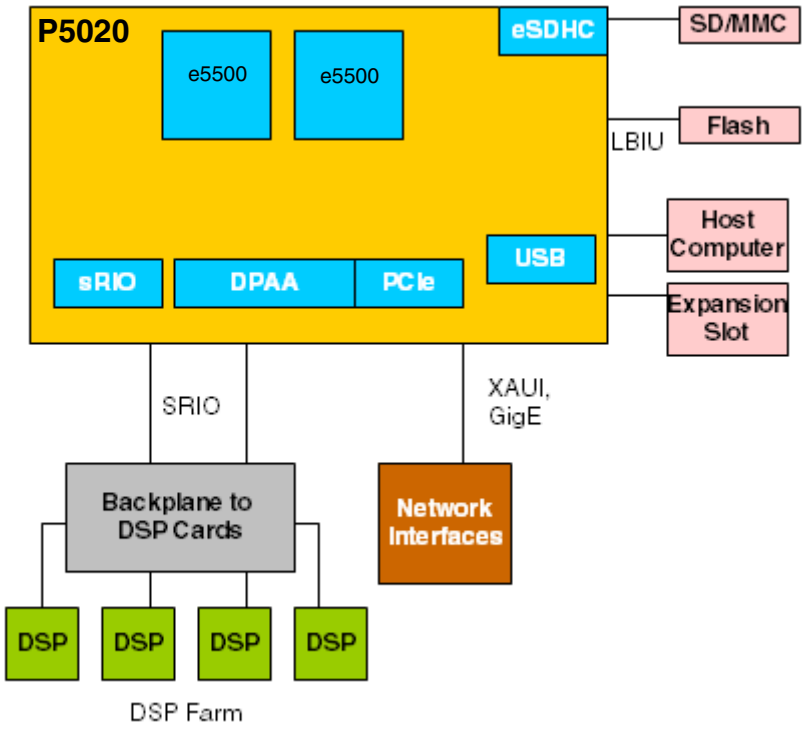

Figure 2. Control Plane Processor for a DSP Farm

1.3 SAN RAID 6 Controller

The following figure shows a RAID-enabled Disk Array Controller in an redundant active-active system for block-oriented storage systems. The P5020 Data Path Acceleration Architecture (DPAA) accelerates RAID 5/6 calculations and low-overhead data movement while optionally supporting data-at rest encryption and Data Integrity Field support.

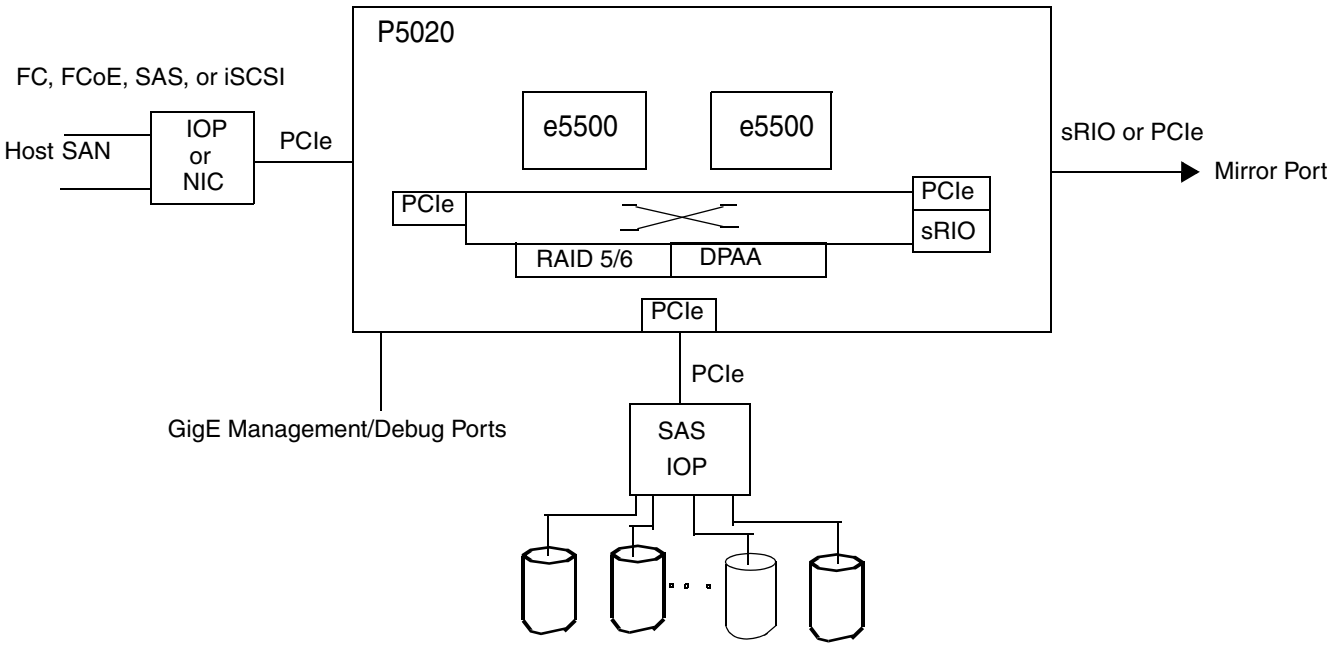

Figure 3. SAN RAID Controller

2 P5020 Dual-Core Processing Options

The device 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 device cores running on an OS:

- Symmetric multi-processing
- Cooperative asymmetric multi-processing
  - — Two copies of the same OS that are non-SMP enabled
  - — Two separate operating systems

2.2 Running OS-Less Using a Simple Scheduler

It is also possible for one or more cores to run OS-less, using a simple scheduler. This is a likely scenario when cores are performing datapath operations with bounded real-time requirements. This use case is greatly enhanced by the provisioning of a 512-Kbyte private back-side L2 cache for each e5500 core. These caches can operate as a traditional unified cache, or be set to operate as instruction only, data only, or even locked and used as memory-mapped SRAM.

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 P5020. 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

  - Supervisor
  - Hypervisor
- Independent boot and reset
- Secure boot capability

- Two 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
- Two 64-bit DDR3/3L SDRAM memory controllers with ECC and interleaving support
- Datapath acceleration architecture (DPAA) incorporating acceleration for the following functions:
  - 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)
  - RapidIO™ messaging manager (RMan)
  - RAID5/6 Engine
    - Support for XOR and Galois Field parity calculation
    - Support for data protection information (DPI)
- Ethernet interfaces
  - One 10 Gbps Ethernet (XAUI) controller
  - Five 1 Gbps or four 2.5 Gbps Ethernet controllers
- High speed peripheral interfaces
  - Four PCI Express 2.0 controllers/ports running at up to 5 GHz
  - Two serial RapidIO 2.0 controllers/ports (version 1.3 with features of 2.1) running at up to 5 GHz with Type 11 messaging and Type 9 data streaming support
- Additional peripheral interfaces
  - Dual SATA supporting 1.5 and 3.0 Gb/s operation
  - Two USB 2.0 controllers with integrated PHY
  - SD/MMC controller (eSDHC)
  - Enhanced SPI controller
  - Four I^2C controllers
  - Two Dual DUARTs
  - Enhanced local bus controller (eLBC)
- 18 SerDes lanes to 5 GHz
- Multicore Programmable Interrupt Controller (MPIC)

- Two 4-channel DMA engines

3.3 P5020 Benefits

The P5020's e5500 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 e5500 core, is a defining characteristic of the device. 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 Data Path Acceleration Architecture (DPAA) Benefits

While the two 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 P5020 uses Freescale's Data Path Acceleration Architecture (DPAA) (see Section 3.9, "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 P5020 allows the user's software to perform complex packet processing at high data rates.

3.5 Critical Performance Parameters

The following table lists key performance indicators that define a set of values used to measure P5020 operation.

Table 1. P5020 Critical Performance Parameters

Indicator	Values(s)
Top speed bin core frequency	2.0 GHz
Maximum memory data rate	1.3 GHz (DDR3/3L)[1] • 1.5-V for DDR3 • 1.35-V for DDR3L
Local bus	• 3.3 V • 2.5 V • 1.8 V
Operating junction temperature range	0–105 C
Package	1295-pin FC-PBGA (flip-chip plastic ball grid array)

Notes:
1 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

3.6.2 512-Kbyte Private Backside Cache

- Each e5500 core features a 512-Kbyte private backside L2 cache running at the same frequency of CPU. The caches support:Write Back, pseudo LRU replacement algorithm
- Tag parity and ECC data protection
- Eight-way, with arbitrary partitioning between instruction and data. For example, 3-ways instruction, 5-ways data, and so on.
- Supports direct stashing of datapath architecture data into cache

3.6.3 CoreNet Platform Cache (CPC)

The QorIQ P5020 also contains 2x1-Mbyte of shared CoreNet platform cache, with the following features:

- Configurable as write back or write through
- Pseudo LRU replacement algorithm
- ECC protection
- 64-byte coherency granule
- Two cache line read 1024 bits per cycle at 800 MHz, 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)

3.6.4 CoreNet Fabric and Address Map

The CoreNet fabric is Freescale's next generation 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 800 MHz
- The CoreNet fabric has been designed to overcome bottlenecks associated with shared bus architectures, particularly address issue and data bandwidth limitations. The P5020'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 device having lower average memory latency

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 P5020s. See Section 3.6.7, "Enhanced Local Bus Controller."
- The serial memory controllers may be accessed by software running on the CPUs following boot; they are not dedicated to the PBL. See Section 3.6.7.1, "Serial Memory Controllers."

3.6.7 Enhanced Local Bus Controller

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 32-bit data bus operating at up to 93 MHz
- Eight chip selects for eight external slaves
- Up to eight-beat burst transfers
- 8-, 16-, or 32-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-, or 32-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.

3.6.7.1 Serial Memory Controllers

In addition to the parallel NAND and NOR flash supported by means of the eLBC, the P5020 supports serial flash using SPI and SD/MMC/eMMC card. The SD/MMC/eMMC controller includes a DMA engine, allowing it to move data from serial flash to external or internal memory following straightforward initiation by software.

3.7 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:

- Compatible with USB specification, Rev. 2.0
- Supports full-speed (12 Mbps), and low-speed (1.5 Mbps) operations
- Supports the required signaling for the USB transceiver macrocell interface (UTMI).The PHY interfacing to the UTMI is an internal PHY.
- Both controllers support operation as a stand-alone USB host controller
  - Support USB root hub with one downstream-facing port
  - Enhanced host controller interface (EHCI)-compatible
- One controller supports operation as a stand-alone USB device
  - Supports one upstream-facing port
  - Supports six programmable USB endpoints

The host and device functions are both configured to support all four USB transfer types:

- Bulk
- Control
- Interrupt
- Isochronous

3.8 High-Speed Peripheral Interface Complex

All high-speed peripheral interfaces connect via 18 lanes of 5-GHz SerDes to a common crossbar switch referred to as OCeaN. Two high-speed I/O interface standards are supported: PCI Express (PCIe), and Serial RapidIO (sRIO). The P5020 integrates the following:

- Four PCIe controllers
- Two Serial RapidIO controllers
- RapidIO message manager (RMan).

3.8.1 PCI Express Controllers

Each of the four PCIe interfaces is compliant with the *PCI Express Base Specification Revision 2.0*. Key features of the PCIe interface include the following:

- Power-on reset configuration options allow root complex or endpoint functionality.
- The physical layer operates at 2.5 or 5 Gbaud data rate per lane.
- Receive and transmit ports operate independently, with an aggregate theoretical bandwidth of 32 Gbps.

- x8, x4, x2, and x1 link widths supported
- Both 32- and 64-bit addressing and 256-byte maximum payload size
- Full 64-bit decode with 36-bit wide windows
- Inbound INTx transactions
- Message Signaled Interrupt (MSI) transactions

3.8.2 Serial RapidIO

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
- Support for SWRITE, NWRITE, NWRITE_R and Atomic transactions
- Receiver-controlled flow control
- RapidIO error injection
- Internal LP-serial and application interface-level loopback modes

3.8.2.1 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

3.8.3 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+

3.9.1 Packet Distribution and Queue/Congestion Management

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

Table 2. 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.

3.9.2 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. The following table lists some specific content-processing accelerators on the P5020.

Table 3. 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 3.9.4.4, “Security Engine (SEC 4.2)
- Section 3.9.4.5, “Pattern Matching Engine (PME 2.1)

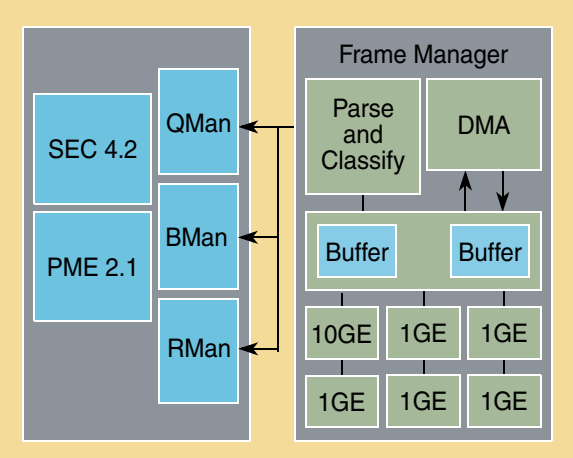

Figure 5. QorIQ Data Path Acceleration Architecture (DPAA)

3.9.4.1 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. 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.

3.9.4.1.1 FMan Network Interfaces

The FMan integrates five data path, tri-speed Ethernet controllers (dTSECs) and one 10-Gbit Ethernet controller.

Note that the more basic parsing and filing capability found in prior PowerQUICC eTSECs is removed from the MACs themselves, and aggregated in the more flexible and robust parsing and classification logic described in Section 3.9.4.1.2, “FMan Parse Function.”

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); the 10-Gbit MAC is a single-speed full duplex. It supports IEEE 802.3 full-duplex flow control (automatic PAUSE frame generation or software-programmed PAUSE frame generation and recognition).

When all SERDES are otherwise allocated, it is possible to enable two of dTSECs by means of RGMII or RMII physical interfaces.

3.9.4.1.2 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 the following table.

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.

3.9.4.1.3 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	• Hashes selected 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 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.

3.9.4.2 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 composed of a user-configured number of frame queues.

3.9.4.3 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

supported which calculates Galois field (GF) based parity calculation for (where MULT = 1 performs simple XOR) up to 16 sources. A variant supports calculation of two GF multiplies for use in calculating XOR and RAID 6 Parity simultaneously without reading the input data twice. This command calculates two GF multiplications across the sources and writes them to two destinations. The GF primitive polynomial is programmable and thus supports common polynomials such as 0x11D and 0x14D.

In addition to classic storage acceleration, the RAID5/6 Engine provides some additional helpful functions including the ability to fill or check a region based on a 128-bit value, incrementing value or using a LSFR algorithm. A compare function is provided that compares two regions of memory and reports the result to a result queue.

The RAID5/6 Engine supports ANSI T10 Data Protection Information and is capable of checking, adding, removing and updating the Data Integrity Fields (DIF). All Reference and Application Tags seen during an operation may be set to an initial value or that value can be incremented as blocks are processed by the engine. Reference Tag, Application Tag can be configurable disabled/enabled from DIF function on per command basis. It also supports IP checksum-based guard generation and checking (RFC 793), in addition to the T10 CRC based guard.

3.10 Avoiding Resource Contentions Using the QorIQ Trust Architecture

Consolidation of discrete CPUs into a single, multicore SoC and potential repartitioning of legacy software on those cores introduces many opportunities for unintended resource contentions to arise, but the QorIQ Trust Architecture can reduce the risk of these issues.

3.10.1 QorIQ Trust Architecture Benefits

A system may exhibit erratic behavior if the multiple CPUs do not effectively partition and share system resources. While it can be challenging to prevent unintended resource contention, stopping malicious software is much more difficult. 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.

The P5020 offers a new level of hardware partitioning support, allowing system developers to ensure software running on any CPU only accesses the resources (memory, peripherals, etc.) that it is explicitly authorized to access. This may not seem like a challenge in an SMP environment, because the OS performs resource allocation for the applications running on it. However, it is a very difficult problem to overcome in AMP environments where there may be multiple instances of the same OS, or even different OSes running on the various CPU cores. Even OS protections in an SMP system may be insufficient in the presence of malicious software.

3.10.2 e5500 Core MMU and Embedded Hypervisor

The P5020's first line of defense against unintended interactions amongst the multiple CPUs/OSes is each 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, peripheral device, and so on) 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, 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 SoC, achieving strong partitioning should not 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 core MMU also includes embedded Hypervisor extensions.

Each core 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 P5020 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.

3.10.3 Peripheral Access Management Unit (PAMU)

The P5020 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.

3.10.4 Secure Boot and Sensitive Data Protection

The core MMUs and PAMU allow the device 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 P5020 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.

3.10.4.1 Secure Boot Option

The system developer digitally signs the code to be executed by the CPU coming out of reset, and the device ensures that only an unaltered version of that code runs on the platform. The P5020 offers both boot time and run time code authenticity checking and configurable consequences when the authenticity check fails.

3.10.4.2 Sensitive Data Protection Option

The P5020 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 device 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.

3.11 Advanced Power Management

The P5020's advanced power management capabilities are based around fine-grained static clock control and software-controlled dynamic frequency management.

3.11.1 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 P5020 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 P5020 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 device. 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.

3.11.2 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).

3.11.3 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 device achieves some critical level, external temperature detection circuitry can drive a high-priority interrupt into the P5020, 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 G Hz 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 core.

3.12 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 HW 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 P5020 as they would with multiple discrete Freescale communications processors, Freescale developed the debug architecture shown in the following figure.

— Generates trace messages to Nexus Port Controller
— 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 P5020.

The various levels of the developer environment are shown in Figure 9, 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, as follows:

- 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

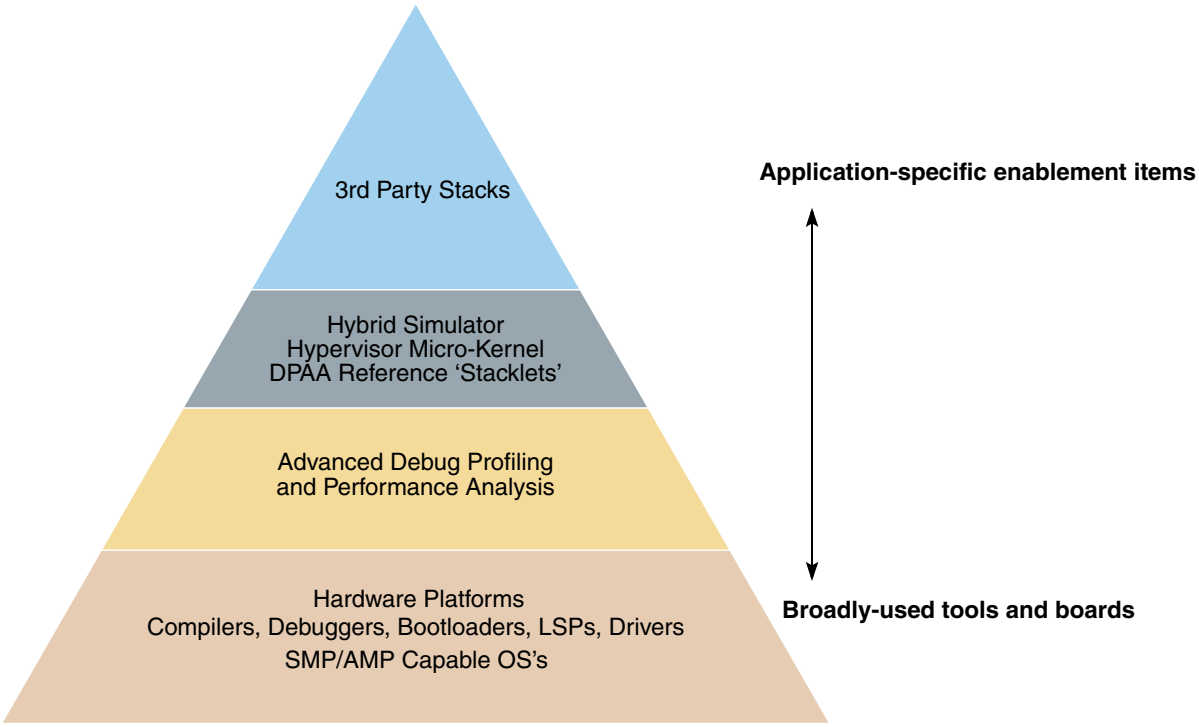

Figure 9. Levels of Developer Environment

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 device.

4.1.3 SMP/AMP Capable OS's

Open source tools will be part of an overall P5020 development board Linux support package, which will include AMP and SMP versions of the Linux OS, and P5020 drivers for the accelerators and networking and peripheral interfaces featured in the P5020. 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 P5020.

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 P5020. The hybrid simulator allows the user to switch between "fast functional mode" and "detailed performance mode" with capabilities that include the following:

- Global visibility
- Determinism
- Bug reproducibility
- Reverse execution
- Special abilities to detect race conditions
- Ability to detect race conditions

4.3.2 Hypervisor Micro-Kernel

The P5020's e5500 cores offer a new embedded Hypervisor capability to address the need for a single operating system performing coordination and access control functions, managing shared resources in an efficient manner. The embedded Hypervisor provides the software layer needed to manage the operating systems and supervisor-level applications as they access shared resources. Recognizing that each developer's system design may call for a different partitioning of resources, and involve different combinations of OSes and RTOSes, Freescale and our ecosystem partners will provide reference implementations of the embedded Hypervisor's peripheral virtualization and access control which the developer can modify to match unique system requirements.

4.3.3 DPAA Reference "Stacklets"

It is expected that some CPUs will be dedicated as datapath processors, working closely with the DPAA. Freescale will provide reference protocol "stacklets," optimizing performance critical regions of protocol processing and their interaction with the DPAA hardware.

4.4 Top Level of the Pyramid: Application-Specific Enablement

This category includes 3rd-party stacks optimized for DPAA, RegEx, AV TCP, IPv4/6, IPsec/SSL.

Many of the expected applications for the P5020 involve network protocol processing. Partitioning between control CPUs and datapath CPUs, and developing the protocol processing firmware which runs on the datapath CPUs is an area for significant value added services for Freescale partners at the top level of the enablement pyramid. OEMs wishing to engage with these partners can realize significant "time-to-performance" advantages.

5 Document Revision History

The following table provides a revision history for this product brief.

Table 8. Revision History

Revision	Date	Substantive Change(s)
1	02/2013	Modified USB Specification, Section 3.7, "Universal Serial Bus (USB) 2.0."
0	12/2011	Initial public release.