

Welcome to [E-XFL.COM](https://www.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	Obsolete
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	-40°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/product-detail/nxp-semiconductors/p5020nx1vnb

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.

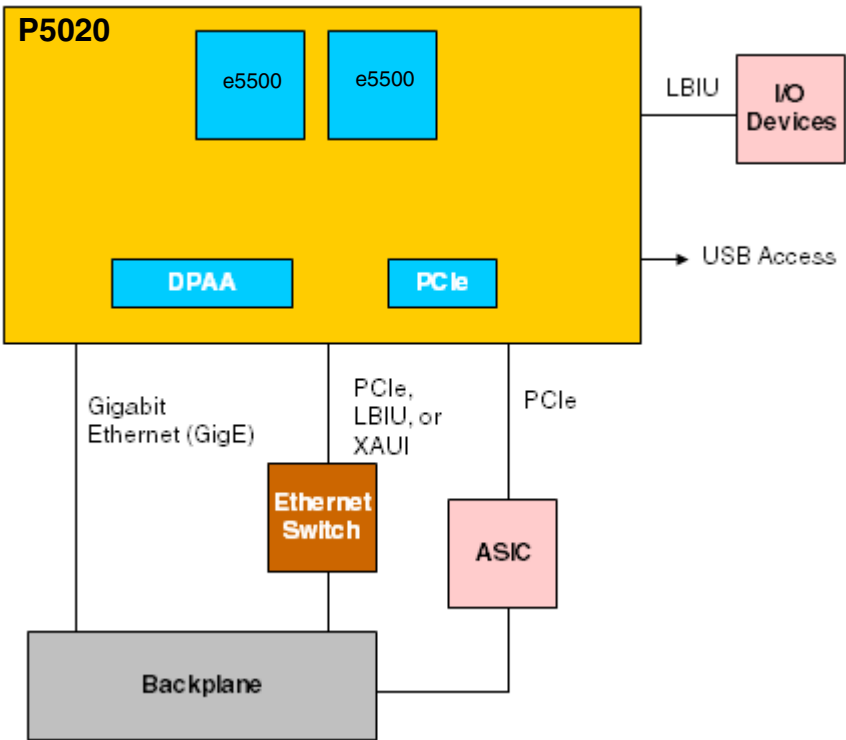


Figure 1. Control Plane Processor for a Router

1.2 DSP Farm Control Processor

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

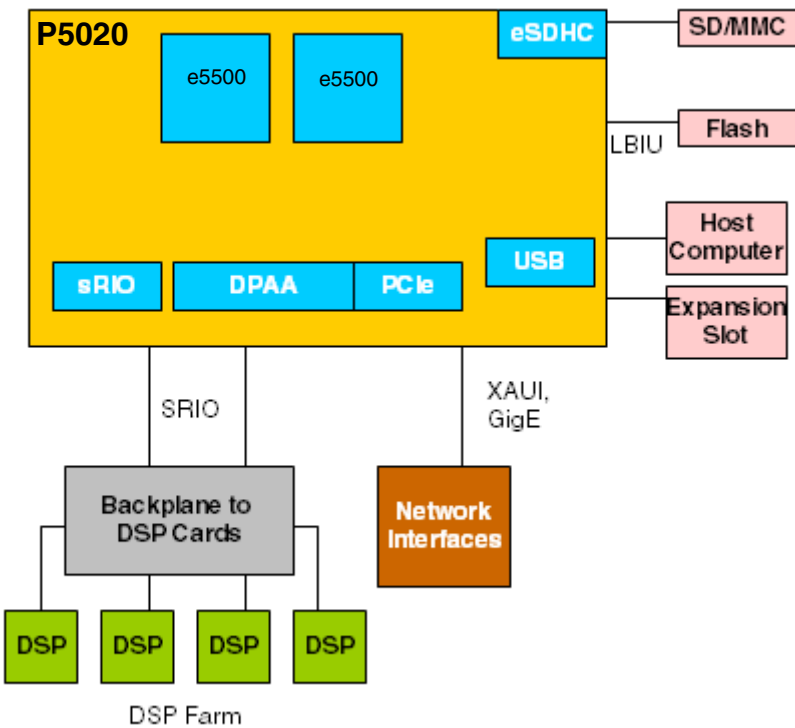


Figure 2. Control Plane Processor for a DSP Farm

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.

Features

- 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 I²C controllers
 - Two Dual DUARTs
 - Enhanced local bus controller (eLBC)
- 18 SerDes lanes to 5 GHz
- Multicore Programmable Interrupt Controller (MPIC)

3.6 e5500 Core and Cache Memory Complex

Each e5500 is a superscalar dual issue processor, supporting out-of-order execution and in-order completion, which allows the Power Architecture e5500 to perform more instructions per clock than other RISC and CISC architectures.

3.6.1 e5500 Core Features

- Up to 2.0 GHz core clock speed
- 36 bit physical addressing
- 64 TLB SuperPages
- 512-entry, 4-Kbyte pages front end
- 3 Integer Units: 2 simple, 1 complex (integer multiply and divide)
- 64-byte cache line size
- L1 caches, running at same frequency of CPU
 - 32-Kbyte Instruction, 8-way
 - 32-Kbyte Data, 8-way
 - Both with data and tag parity protection
- Supports data path acceleration architecture (DPAA) data and context “stashing” into the L1 data cache and the backside L2 cache
- User, supervisor, and hypervisor instruction level privileges
- New processor facilities
 - Hypervisor APU
 - Classic double precision floating point unit
 - Uses 32 64-bit floating-point registers (FPRs) for scalar single- and double-precision floating-point arithmetic
 - Replaces the embedded floating-point facility (SPE) implemented on the e500v1 and e500v2
 - Designed to comply with IEEE Std. 754™1985 FPU for both single- and double-precision operations
 - “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
 - Expanded interrupt model
 - Improved programmable interrupt controller (PIC) automatically ACKs interrupts
 - Implements message send and receive functions for interprocessor communication, including receive filtering
 - External PID load and store facility
 - Provides system software with an efficient means to move data and perform cache operations between two disjoint address spaces

- 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

Features

The 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 device 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).

3.6.5 Memory Complex

The P5020 memory complex consists of the two DDR controllers for main memory, and the memory controllers associated with the enhanced local bus controller (eLBC).

3.6.5.1 DDR Memory Controllers

The two DDR memory controllers have the following functionalities:

- Supports DDR3/3L SDRAM. The P5020 also supports chip-select interleaving within a controller. The memory interface controls main memory accesses and together the two controllers support a maximum of 64 Gbytes of main memory.
- Supports interleaving across controllers on bank, page, or cache line boundaries.
- The P5020 can be configured to retain the currently active SDRAM page for pipelined burst accesses. Page mode support of up to 64 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 P5020 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.

3.6.6 PreBoot Loader (PBL) and Nonvolatile Memory Interfaces

The PreBoot Loader (PBL) is a new logic module that operates similarly to an I²C 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 (I²C, eLBC, SPI, and SD/MMC) into fully initialized DDR or the 2-Mbyte CPC.
- 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 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.

- 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+

Features

- 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

3.9 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:

- Packet Distribution and Queue/Congestion Management
- Accelerating Content Processing

Features

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	<ul style="list-style-type: none"> 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	<ul style="list-style-type: none"> 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. 	<ul style="list-style-type: none"> 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

3.9.4.4 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 the previous SEC crypto-channels. DECOs are responsible for header and trailer processing, and managing context and data flow into the CHAs assigned to it for the length of an operation.

The DECOs can perform header and trailer processing, as well as single pass encryption/integrity checking for the following security protocols:

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

In prior versions of the SEC, the individual algorithm accelerators were referred to as Execution Units (EUs). In the SEC 4.2, these are referred to as Crypto Hardware Accelerators (CHAs) to distinguish them from prior implementations. Specific CHAs available to the DECOs are listed below.

- Advanced encryption standard unit (AESA)
- ARC four execution unit (AFHA)
- Cyclic redundancy check accelerator (CRCA)
- Data encryption standard execution unit (DESA)
- Kasumi execution unit (KFHA)
- SNOW 3 G hardware accelerator (STHA)
- Message digest execution unit (MDHA)
- Public key execution unit (PKHA)
- Random number generator (RNGB)

Depending on the security protocol and specific algorithms, the SEC 4.2's aggregate symmetric encryption/integrity performance is 5 Gbps, while asymmetric encryption (RSA public key) performance is ~5,000 1024b RSA operations per second.

The SEC 4.2 is also part of the QorIQ Trust Architecture, which gives the P5020 the ability to perform secure boot, runtime code integrity protection, and session key protection. The Trust Architecture is described in [Section 3.10, "Avoiding Resource Contentions Using the QorIQ Trust Architecture."](#)

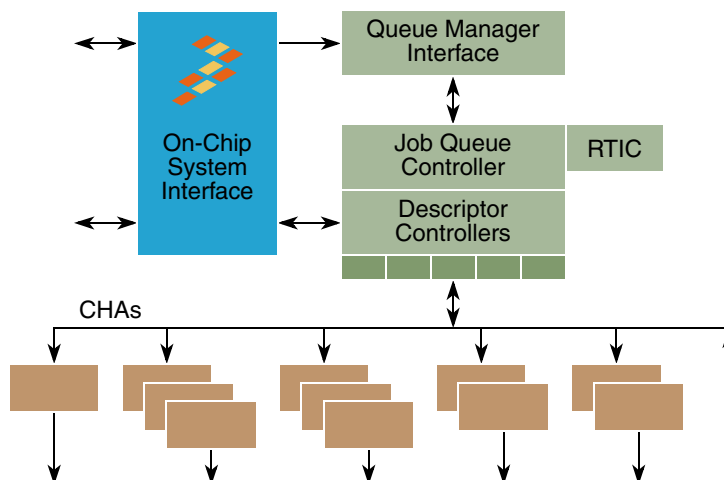


Figure 6. SEC 4.2 Block Diagram

3.9.4.5 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 ~ 5 Gbps.

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.

3.9.4.5.1 PME Regular Expressions (Regex)

The PME specifies patterns of bytes as regular expressions (regex). The P5020 (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.

3.9.4.5.2 PME Match Detection

Within the PME, match detection proceeds in stages. The key element scanner performs initial byte pattern matching, with handoff to the data examination engine for elimination of false positives through more complex comparisons. As the name implies, the stateful rule engine receives confirmed basic matches from the earlier stages, and monitors a stream for addition for subsequent matches that define an event pattern.

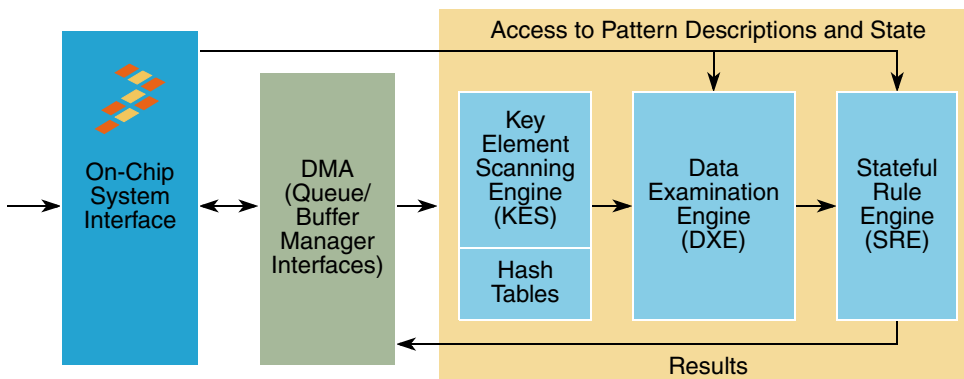


Figure 7. PME 2.1 Block Diagram

3.9.4.6 RapidIO Message Manager (RMan)

The RapidIO message manager (RMan) produces and consumes Type 8 Port-write, Type 9 Data Streaming, Type 10 Doorbells and Type 11 Messaging traffic and is capable of producing Type 5 NWRITE and Type 6 SWRITE transactions.

For inbound traffic, the RMan supports up to 17 open reassembly contexts as a arbitrary mix of Type 9, and Type 11 traffic.

As ingress packets arrives at the RMan, they are compared against up to 64 classification rules to determine the target queue. These rules support Type 8, 9, 10 and 11 transaction types. They may be wildcarded and are configured as masks over selected header fields. The following fields are maskable as part of each classification rule:

Transaction types:

- RapidIO port
- Source ID
- Destination ID
- Flow level

Type 9 messaging-specific fields:

- Class-of-service (CoS)
- StreamID

Type 11 messaging-specific fields:

- Mailbox

- Extended mailbox
- Letter

Should the packet remain unclassified, the traffic is retried with an error in the case of Type 10 and 11 traffic and dropped in the case of Type 9 traffic. Dropped traffic is logged and upon a threshold can assert an error interrupt.

Classification allows Type 9, 10 and 11 traffic to be distributed across 64 possible Frame queues. A single dedicated inbound Type 8 Port-write Frame queue is provided.

For all outbound traffic types (Type 8, 9, 10 and 11), the Datapath Acceleration Architecture allows a very large number of outbound Frame queues effectively limited by system, software and performance constraints.

3.9.4.7 RAID5/6 Engine

The P5020 includes a RAID5/6 Engine for storage applications, which significantly extends the capability and performance of earlier PowerQUICC RAID (XOR) functionality. The RAID5/6 Engine supports a variety of storage-related functions such as Move, Generate XOR, RAID 6 Parity, Fill and Check. The following table summarizes the functions supported by the engine.

Table 7. RAID5/6 Engine Supported Functions

Function	No. of Sources	No. of Destinations	Command Options	
			Scatter/Gather	DIF
No Op	—	—	—	—
Single Source Move	1	1	Y	N
Multicast Move	1	2	Y	N
Add DIF	1	1 or 2	Y	Y
Remove DIF	1	1 or 2	Y	Y
Update DIF	1	1 or 2	Y	Y
Generate Q Parity	2–16	1	Y	Y
Generate Q and Q Parity	2–16	2	Y	Y
Fill Pattern	—	1	Y	Y
Check Pattern	1	—	Y	Y
Fill LFSR	—	1	Y	N
Check LFSR	1	—	Y	N
Compare	2	—	Y	Y
Gather DIF	1	1	Y	Y

The RAID5/6 Engine supports commands with between 1 and 16 sources for relevant functions. A simple DMA move operation is supported along with a two-destination multicast move that duplicates the source data. Both of these simple operations are the foundation for commands that support Data Protection Information (DIF) insertion, updating and checking. A single RAID5/6 parity generate function is

Features

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

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 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 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](#)

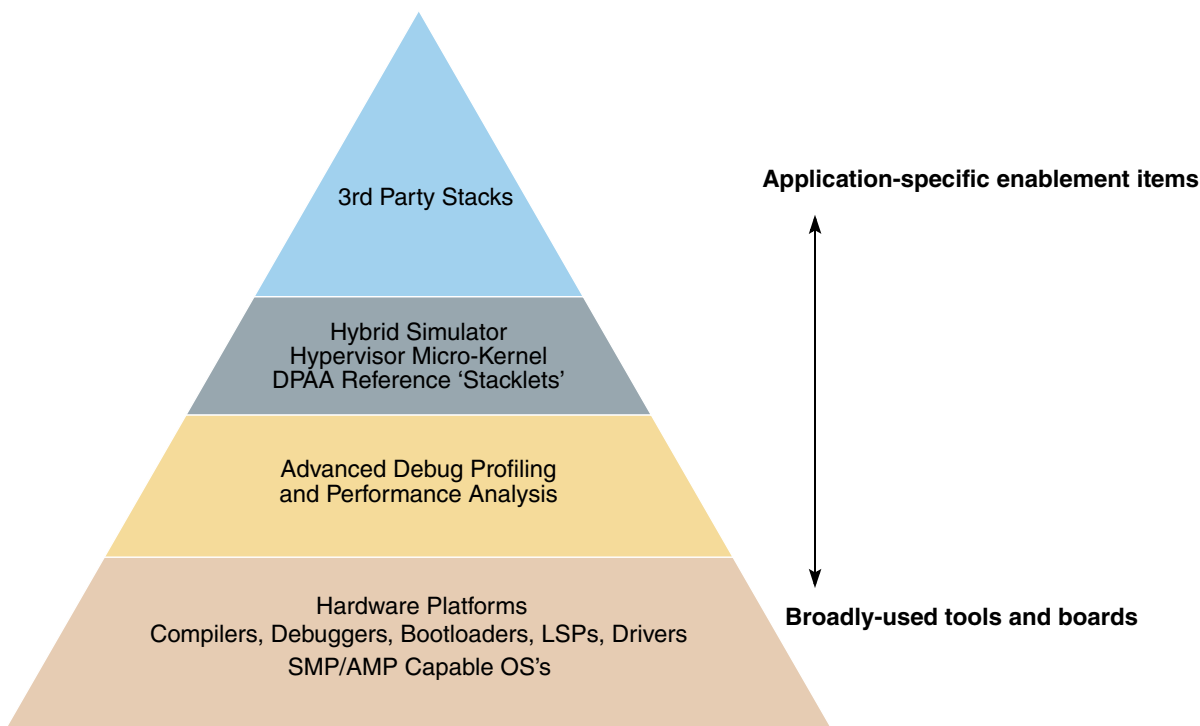


Figure 9. Levels of Developer Environment

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:

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: <http://www.reg.net/v2/webservices/Freescale/Docs/TermsandConditions.htm>.

Freescale, the Freescale logo, Altivec, C-5, CodeTest, CodeWarrior, ColdFire, C-Ware, Energy Efficient Solutions logo, Kinetis, mobileGT, PowerQUICC, Processor Expert, QorIQ, Qorivva, StarCore, Symphony, and VortiQa are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. Airfast, BeeKit, BeeStack, ColdFire+, CoreNet, Flexis, MagniV, MXC, Platform in a Package, QorIQ Qonverge, QUICC Engine, Ready Play, SafeAssure, SMARTMOS, TurboLink, Vybrid, and Xtrinsic are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2011-2013 Freescale Semiconductor, Inc.

