Advance Information

MPC190TS/D
Rev. 0.2, 2/2003

MPC190 Security Processor
Technical Summary

This document provides an overview of the MPC190 security processor, including a brief development history, target applications, key features, typical system architecture, device architectural overview, and a performance summary.

1 Development History

The MPC190 belongs to the Smart Networks platform's S1 family of security processors developed for the commercial networking market. This product family is derived from security technologies Motorola has developed over the last 30 years, primarily for government applications. The fourth-generation execution units (EU) have been proven in Motorola semi-custom ICs and in the MPC180, the first product in Motorola's security processor line.

2 Typical Applications

The MPC190 is suited for applications such as the following:

- Edge routers
- DSLAMS
- Broadband access equipment
- eCommerce severs
- Wireless base stations
- WAP Gateways

3 Features

The MPC190 is a flexible and powerful addition to any networking or computing system supporting PCI. The MPC190 is designed to off-load computationally intensive security functions—such as key generation and exchange, authentication, and bulk encryption—from PowerQuicc II™ communications processors with integrated PCI (the MPC8265A and the MPC8266A) or from any processor through the use of a PCI bridge chip.

The MPC190 is optimized to process all the algorithms associated with IPSec, IKE, WTLS/WAP and SSL/TLS. In addition, the MPC190 is the only security processor on the market (other than the MPC180) capable of executing the elliptic curve cryptography that is especially important for secure wireless communications.

MPC190 features include the following:

- 6 Public key execution units (PKEUs) that support the following:
  - RSA and Diffie-Hellman
    - Programmable field size 80- to 2048-bits
    - RSA-1024-64 key exchange in 2.0ms
    - 520 IKE handshakes/second
  - Elliptic curve operations in either F(2) m or F(p)
    - Programmable field size from 55- to 511-bits
    - ECC key exchange (155 bit key) in 5.7ms
    - 1000 IKE handshakes/second
- 3 Data encryption standard execution units (DEUs)
  - DES
  - 3DES
  - Two key (K1, K2, K1) or Three Key (K1, K2, K3)
  - ECB and CBC modes for both DES and 3DES
- 3 Message digest execution units (MDEUs)
  - SHA-1 with 160-bit message digest
  - MD4 or MD5 with 128-bit message digest
  - HMAC with either algorithm
- ARC four execution unit (AFEU)
  - Implements a stream cipher compatible with the RC4 algorithm
  - 40- to 128-bit programmable key
- Random number generator (RNG)
- PCI 2.2 compliant external bus interface, with master/slave logic.
  - 32-bit address/64 -bit data, 66MHz
  - 32-bit address/32 –bit data mode
- 9 Crypto-channels, each supporting multi-command descriptor chains
  - Static and/or dynamic assignment of crypto-execution units via an integrated controller
  - Buffer size of 2KBytes for each crypto-channel
- 1.8v supply, 3.3v I/O
- 252 MAP BGA,
- 2.0W power dissipation
- HiPerMOS4 0.25µm process

4 Typical System Architecture

The MPC190 is designed to integrate easily into systems using PCI, including systems built with processors with integrated PCI bridges, such as the Motorola MPC8265A, as shown in Figure 4-1. The external processor accesses the MPC190 through its device drivers using system memory for data storage. The MPC190 resides in the PCI address map of the processor; therefore, when an application requires cryptographic functions, it creates descriptors for the MPC190, defining the cryptographic function to be performed and the location of the data. The MPC190's PCI-mastering capability permits the host processor to set up a crypto-channel with a few short register writes, leaving the MPC190 to perform reads and writes

on system memory to complete the required task. Alternatively, all the execution units' registers are available for direct read and write through the PCI interface.

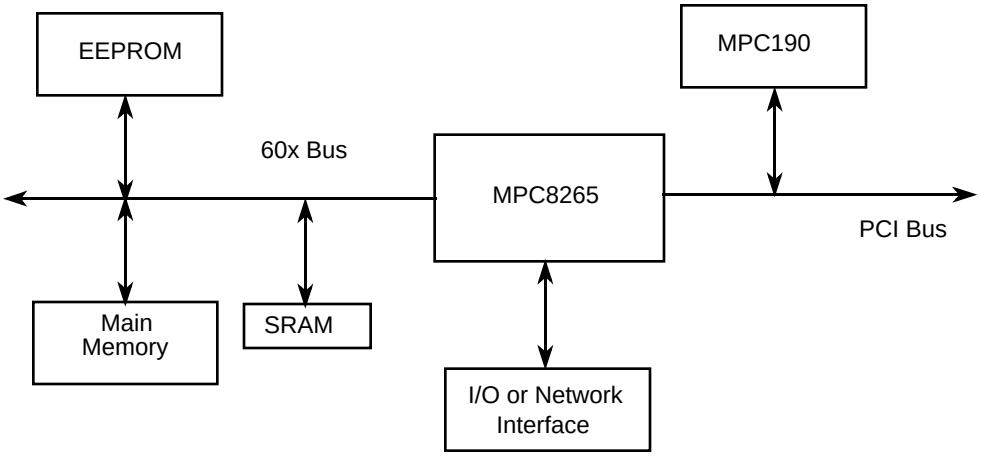

Figure 4-1. MPC190 Connected to PowerQuicc II PCI Bus

Figure 4-2 shows a configuration with the MPC190 communicating with the host processor via a PCI bridge, such as the MPC107.

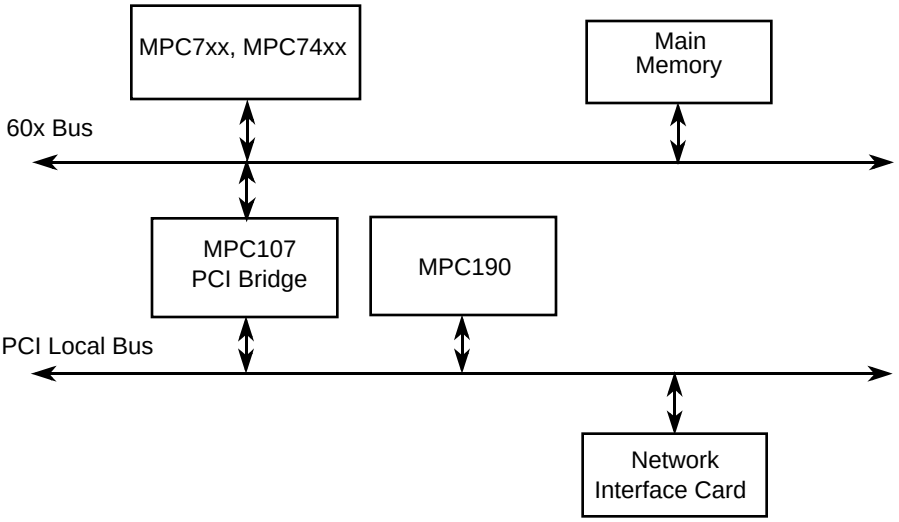

Figure 4-2. MPC190 Connected to PowerPC Host CPU Via Bridge

5 Architectural Overview

A block diagram of the MPC190 internal architecture is shown in Figure 5-3. The PCI bus interface (PCI I/F) module is designed to transfer 32-bit or 64-bit words between the PCI v2.2-compliant bus and any register inside the MPC190. The MPC190 controller decodes descriptor headers (See 6.6, "Crypto-Channels.") and writes them to the appropriate crypto-channel input buffer. The crypto-channel then processes the data pointers within the data packet descriptor and, via the PCI/IF module, initiates PCI bus mastering to transfer additional data and instructions from memory, as specified by the services

requested in the descriptor header. As data is processed, it is written to the individual execution units' output buffers and then, via the PCI/IF module, the processed data is written back to system memory.

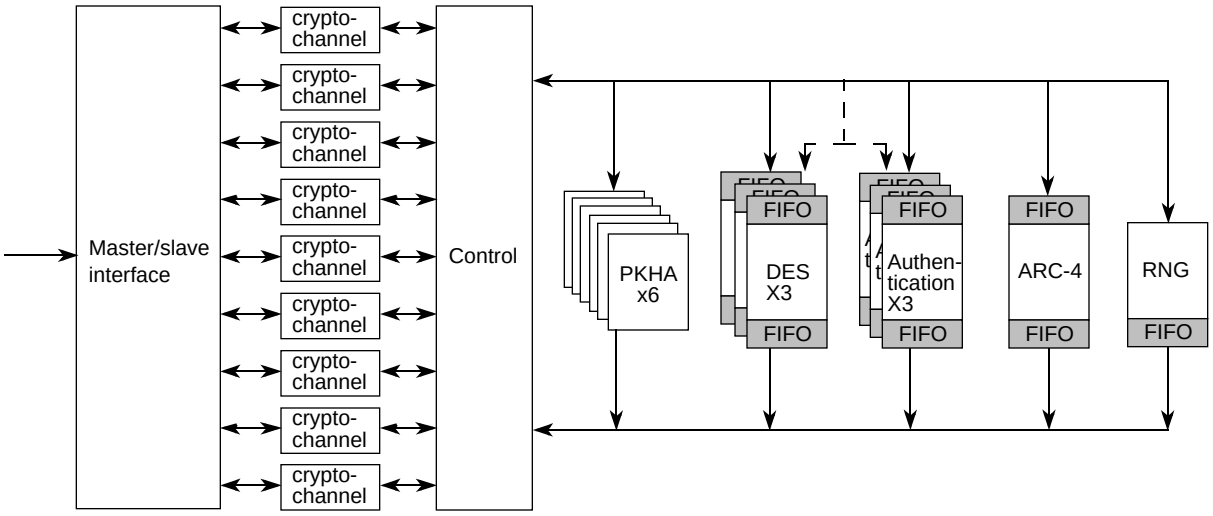

Figure 5-3. MPC190 Block Diagram

6 Data Packet Descriptors

As an IPSec accelerator, the MPC190's controller has been designed for easy use and integration with existing systems and software. All cryptographic functions are accessible through data packet descriptors, some of which have been defined as multifunction to facilitate IPSec applications. A data packet descriptor is diagrammed in Table 6-1.

Table 6-1. Example Data Packet Descriptor

Field Name	Value/Type	Description
DPD_DES_CTX_CRYPT	tbd	Representative header for "DES using Context to Encrypt"
LEN_CTXIN PTR_CTXIN	length pointer	Number of bytes to be written Pointer to context (IV) to be written into DES engine
LEN_KEY PTR_KEY	length pointer	Number of bytes in key Pointer to block cipher key
LEN_DATAIN PTR_DATAIN	length pointer	Number of bytes of data to be ciphered Pointer to data to perform cipher upon
LEN_DATAOUT PTR_DATAOUT	length pointer	Number of bytes of data after ciphering Pointer to location where cipher output is to be written
LEN_CTXOUT PTR_CTXOUT	length pointer	Length of output context (IV) Pointer to location where altered context is to be written
nul length nul pointer	length pointer	Zeroes for fixed length descriptor filter Zeroes for fixed length descriptor filter
nul length nul pointer	length pointer	Zeroes for fixed length descriptor filter Zeroes for fixed length descriptor filter
LEN_NEXT PTR_NEXT	length pointer	Length of next data packet descriptor (bytes) Pointer to next data packet descriptor

Each data packet descriptor contains the following:

- Header—The header describes the required services and encodes information that indicates which EUs to use and which modes to set.
- Seven data length/data pointer pairs—The data length indicates the number of contiguous bytes of data to be transferred (not to exceed 2048). The data pointer indicates the starting address of the data, key, or context in system memory.

A data packet descriptor ends with a pointer to the next data packet descriptor. Therefore, once a descriptor is processed and if the value of this pointer is non-zero, it is used to request a PCI burst read of the next descriptor.

Processing of the next descriptor (and whether or not an interrupt is generated) is determined by the programming of crypto-channel's configuration register. Two modes of operation are supported:

- Interrupt at end of descriptor
- Interrupt at end of descriptor chain

The crypto-channel requests a write-back of the descriptor header after processing a data packet descriptor. The value written back is identical to that of the header, with the exception that a DONE field is set.

Occasionally, a descriptor field may not be applicable to the requested service. For example, if using DES in ECB mode, the contents of the IV field do not affect the result of the DES computation. Therefore, when processing data packet descriptors, the crypto-channel skips any pointer that has an associated length of zero.

6.1 PCI Interface

The PCI interface manages communication between the MPC190's internal execution units and the PCI bus. The interface is memory mapped; therefore, PCI target accesses and PCI initiator writes from the MPC190 must be addressed on 32-bit double-word (DWORD) boundaries. The MPC190 performs PCI initiator reads on byte boundaries and assigns the data to DWORD boundaries as appropriate.

The PCI v2.2-compliant PCI interface supports 32-bit address and data transfers and up to 66 MHz / 64-bit data transfers. External support circuitry is required for voltage level conversion when connected to a 33 MHz / 32 bit 5V bus. The user should only be concerned with the external timing between the PCI interface and the external bus; internal timing is maintained by the PCI Interface.

6.2 MPC190 Controller

The MPC190 controller manages on-chip resources, including individual execution units (EUs), FIFOs, the PCI Interface, and the internal buses that connect all the various modules. The controller receives service requests from the PCI interface and various crypto-channels, and schedules the required activities. The controller can configure each of the on-chip resources in three modes:

- Host-controlled mode—The host is directly responsible for all data movement into and out of the resource.
- Static mode—The user can reserve a specific execution unit to a specific crypto-channel.
- Dynamic mode—A crypto channel can request a particular service from any available execution unit.

6.3 Host-Managed Register Access

All EUs can be used entirely through register read/write access. It is strongly recommended that read/write access only be performed on a EU that is statically assigned to an idle crypto-channel. Such an assignment is the only method for the host to inform the controller that a particular EU is in use.

6.4 Static EU Access

The Controller can be configured to reserve one or more EUs to a particular crypto-channel. Doing so permits locking the EU to a particular context. When in this mode, the crypto-channel can be used by multiple descriptors representing the same context without unloading and reloading the context at the end of each descriptor. This mode presents considerable performance improvement over dynamic access, but only when the MPC190 is supporting few (or one) contexts. Static EU access can also be used to reserve one particular public key execution unit (PKEU) for one type of computation. For example, one PKEU could be reserved for all private key RSA operations using prime P, and the other could be reserved for all computations using prime Q. Again, this presents a performance improvement because all fixed parameters can remain within the reserved PKEUs. This reduces the overhead of loading and unloading contexts and therefore improves performance. However, this is only a performance improvement if the lack of dynamically available PKEUs does not become a bottleneck in key agreement protocols.

6.5 Dynamic EU Access

Processing begins when a data packet descriptor pointer is written to the next descriptor pointer register of one of the crypto-channels. Prior to fetching the data referred to by the descriptor and based on the services requested by the descriptor header in the descriptor buffer, the controller dynamically reserves usage of an EU to the crypto-channel. If all appropriate EU units are already dynamically reserved by other crypto-channels, the crypto-channel stalls and waits to fetch data until an appropriate EU is available.

If multiple crypto-channels simultaneously request the same EU, the EU is assigned on a round-robin basis. Once the required EU has been reserved, the crypto-channel fetches and loads the appropriate data packets, operates the EU, unloads data to system memory, and releases the EU for use by another crypto-channel. If a crypto-channel attempts to reserve a statically-assigned EU (and no appropriate EUs are available for dynamic assignment), an interrupt is generated and status indicates illegal access. When dynamic assignment is used, each encryption/decryption packet must contain context that is particular to the context being supported.

6.6 Crypto-Channels

The MPC190 includes nine crypto-channels that manage data and EU function. Each crypto-channel consists of the following:

- Control registers containing information about the transaction in process
- A status register containing an indication of the last unfulfilled PCI request
- A pointer register indicating the location of a new descriptor to fetch
- Buffer memory used to store the active data packet descriptor (See 6, “Data Packet Descriptors.”)

Crypto-channels analyze the data packet descriptor header and request from the controller the first required cryptographic service. After the controller grants access to the required EU, the crypto-channel and the controller perform the following steps:

1. Set the appropriate Mode bits available in the EU for the required service.

2. Fetch context and other parameters as indicated in the data packet descriptor buffer and use these to program the EU.
3. Fetch data as indicated and place in either the EU's input FIFO or the EU itself (as appropriate).
4. Wait for EU to complete processing.
5. Upon completion, unload results and context and write them to external memory as indicated by the data packet descriptor buffer.
6. If multiple services requested, go back to step 2.
7. Reset the appropriate EU if it is dynamically assigned. Note that if statically assigned, a EU is reset only upon direct command written to the MPC190.
8. Perform descriptor completion notification as appropriate. This notification comes in one of two forms—interrupt or header writeback modification—and can occur either at the end of every descriptor or at the end of a descriptor chain.

7 Execution Units (EUs)

"Execution unit" is the generic term for a functional block that performs the mathematical permutations required by protocols used in cryptographic processing. The EUs are compatible with IPsec, WAP/WTLS, IEEE 1363, and Java Security processing, and can work together to perform high level cryptographic tasks.The MPC190's execution units are as follows:

- PKEU for computing asymmetric key mathematics, including Modular Exponentiation (and other Modular Arithmetic functions) or ECC Point Arithmetic
- DEU for performing block symmetric cryptography
- AFEU for performing RC-4 compatible stream symmetric cryptography
- MDEU for hashing data
- RNG for random number generation

7.1 Public Key Execution Unit (PKEU)

The PKEU is capable of performing many advanced mathematical functions to support both RSA and ECC public key cryptographic algorithms. ECC is supported in both F(2)m (polynomial-basis) and F(p) modes. This EU supports all levels of functions to assist the host microprocessor to perform its desired cryptographic function. For example, at the highest level, the accelerator performs modular exponentiations to support RSA and performs point multiplies to support ECC. At the lower levels, the PKEU can perform simple operations such as modular multiplies.

7.1.1 Elliptic Curve Operations

The PKEU has its own data and control units, including a general-purpose register file in the programmable-size arithmetic unit. The field or modulus size can be set in increments of 32 between 80 and 511 bits, supporting a wide range of cryptographic security levels. Because processing time is determined by field or modulus size, larger field / modulus sizes results in greater security but lower performance. For example, a field size of 160 is roughly equivalent to the security provided by 1024 bit RSA. A field size set to 208 roughly equates to 2048 bits of RSA security.

The PKEU block contains routines implementing the atomic functions for elliptic curve processing—point arithmetic and finite field arithmetic. The point operations (multiplication, addition, and doubling) involve one or more finite field operations, which are addition, multiplication, inverse, and squaring. Point add and double each use of all four finite field operations. Similarly, point multiplication uses all EC point operations as well as the finite field operations. All these functions are supported both in modular arithmetic as well as

polynomial basis finite fields. The local control unit makes the necessary calls to the finite field blocks such that either of the two point operations are executed properly.

7.1.2 Modular Exponentiation Operations

The PKEU is also capable of performing ordinary integer modulo arithmetic. This arithmetic is an integral part of the RSA public key algorithm; however, it can also play a role in the generation of ECC digital signatures and Diffie-Hellman key exchanges.

Modular arithmetic functions supported by the MPC190's PKEU include the following:

- R 2 mod N
- A' E mod N
- (A x B) R -1 mod N
- (A x B) R -2 mod N
- (A+B) mod N
- (A-B) mod N

Where the following variable definitions:

  - A' = AR mod N
  - N is the modulus vector
  - A and B are input vectors,
  - E is the exponent vector
  - R is 2 s, where s is the bit length of the N vector rounded up to the nearest multiple of 32.

The PKEU can perform modular arithmetic on operands up to 2048 bits in length. The modulus must be larger than or equal to 129 bits. The PKEU uses the Montgomery modular multiplication algorithm to perform core functions. The addition and subtraction functions exist to help support known methods of the Chinese Remainder Theorem (CRT) for efficient exponentiation.

7.2 Data Encryption Standard Execution Unit (DEU)

The DES execution unit (DEU) performs bulk data encryption/decryption, in compliance with the Data Encryption Standard algorithm (ANSI x3.92). The DEU can also compute 3DES and extension of the DES algorithm in which each 64-bit input block is processed three times. The MPC190 supports 2 key (K1=K3) or 3 key 3DES.

The DEU operates by permuting 64-bit data blocks with a shared 56-bit key and an initialization vector (IV). The MPC190 supports two modes of IV operation: ECB (Electronic Code Book) and CBC (Cipher Block Chaining).

7.3 Arc Four Execution Unit (AFEU)

The AFEU accelerates a bulk encryption algorithm compatible with the RC4 stream cipher from RSA Security, Inc. The algorithm is byte-oriented, meaning a byte of plain text is encrypted with a key to produce a byte of ciphertext. The key is variable length and the AFEU supports key lengths from 40 to 128 bits (in byte increments), providing a wide range of security strengths. RC4 is a symmetric algorithm, meaning each of the two communicating parties share the same key.

7.4 Message Digest Execution Unit (MDEU) Module

The MDEU computes a single message digest (or hash or integrity check) value of all the data presented on the input bus, using either the MD4, MD5 or SHA-1 algorithms for bulk data hashing.

- SHA-1 is a 160 bit hash function, specified by the ANSI X9.30-2 and FIPS 180-1 standards.
- The MD4/MD5 generates a 128 bit hash, and the algorithm is specified in RFC 1321.
- The MDEU also supports HMAC computations, as specified in RFC 2104.

7.5 Random Number Generator (RNG)

The RNG is a digital integrated circuit capable of generating 32-bit random numbers. It is designed to comply with FIPS 140-1 standards for randomness and non-determinism.

Because many cryptographic algorithms use random numbers as a source for generating a secret value (a nonce), it is desirable to have a private RNG for use by the MPC190. The anonymity of each random number must be maintained, as well as the unpredictability of the next random number. The FIPS-140 compliant private RNG allows the system to develop random challenges or random secret keys. The secret key can thus remain hidden from even the high-level application code, providing an added measure of physical security.

8 Performance Estimates

Bulk encryption/authentication performance estimates shown in Table 8-1 include data/key/context reads (from memory to MPC190), security processing (internal to MPC190), and writes of completed data/context to memory by MPC190, using typical 64-bit, 66MHz PCI system overhead.

Table 8-1. Estimated Bulk Data Encryption Performance (Mbps)

	DES CBC	3DES CBC	ARC4	MD5	SHA-1	3DES/ HMAC-MD5
64 byte	191	180	107	191	175	69
128 byte	337	284	186	363	327	130
256 byte	532	410	295	549	481	231
512 byte	784	539	428	736	629	357
1024 byte	1026	639	552	897	747	488
1536 byte	1139	680	630	968	796	558

The MPC190 supports single pass processing of encryption/message authentication.

9 Revision History

Table 9-1 summarizes the revision history of this document.

Table 9-1. Revision History

Revision No.	Substantive Change(s)
0	Initial release.
0.1	Added revision history.
0.2	Updated with new template

THIS PAGE INTENTIONALLY LEFT BLANK

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140
(800) 441-2447

JAPAN:

Motorola Japan Ltd.
SPS, Technical Information Center
3-20-1, Minami-Azabu Minato-ku
Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.
Silicon Harbour Centre, 2 Dai King Street
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

(800) 521-6274

HOME PAGE:

www.motorola.com/semiconductors

MPC190TS/D