# Freescale Semiconductor, Inc.

**MOTOROLA**
*intelligence everywhere*™

*digitaldna*™

This document details all known silicon errata for the MPC190 and its derivatives. Table 1 provides a revision history for this document.

**Table 1. Document Revision History**

| Rev. No. | Substantive Changes |
|----------|---------------------|
| 0 | Initial release. |
| 1 | Added Errata 8. |
| 2 | Added Errata 9–11. |
| 3 | Added Errata 12. Updated template. |

Table 2 provides a revision history of the MPC190 silicon.

**Table 2. Silicon Revision History**

| Silicon Revision | Applicable Errata/Enhancements |
|------------------|-------------------------------|
| PPC190VF | 1–8, 12 |
| XPC190VFA | 3–6, 8, 12 |
| XPC190VFB | 3–6, 9–12 |

**For More Information On This Product,**
**Go to: www.freescale.com**

## Errata No. 1: $\overline{\text{REQ64}}$ Internal Delay

### Detailed Description and Projected Impact:

The PPC190 does not sample the PCI $\overline{\text{REQ64}}$ signal at the correct time when operating at 66 MHz. For a 64-bit PCI bus (any clock speed), the system board asserts the $\overline{\text{REQ64}}$ during RESET to all 64-bit slots. The PCI device samples $\overline{\text{REQ64}}$ on the negation of reset to determine the width of the slot that it is inserted into (32 or 64 bits). If a 64-bit PCI device determines that it is in a 32-bit slot it must terminate the remaining 32 bits so that they do not float and consume excessive power.

When operating at 66 MHz (M66EN high), the MPC190 internally delays its reset, which causes it to sample $\overline{\text{REQ64}}$ too late. Thus, the MPC190 thinks it is in a 32-bit slot. To terminate the remaining 32 bits, the MPC190 drives these signals continuously, which causes fighting buffers when other PCI components try to do 64-bit operations.

### Work Around:

- Run at 33 MHz.
- Use only in a 32-bit slot.
- Delay the negation of $\overline{\text{REQ64}}$ at RESET past the worst-case time for the MPC190 to sample it.

### Projected Solution:

Fixed in XPC190VFA.

## Errata No. 2: RNG Static Mode Operation

### Detailed Description and Projected Impact:

The PPC190 allows for two modes of execution unit operation—static and dynamic. In static mode, a particular execution unit is assigned to a crypto-channel. Static descriptors are used to perform multiple operations with that execution unit without changing keys or context, thus, reducing context switching overhead.

When assigned statically, the PPC190 random number generator (RNG) reliably produces the first 64-bit random number, but then exhibits non-random behavior, including the possibility of locking up. Because there is no context switching associated with requesting random numbers from the RNG, use of static mode offers the user no performance advantage. Dynamic mode allows the user to cause the RNG to output up to 2048 bytes of random values in a single descriptor, and multiple dynamic descriptors can be chained if the user needs more than 2048 bytes.

### Work Around:

- Use dynamic mode descriptors only for RNG operations.
- MPC190 driver version 040302 contains work around to reset RNG when used in static mode.

### Projected Solution:

Fixed in XPC190VFA.

**For More Information On This Product,**
**Go to: www.freescale.com**

## Errata No. 3: PCI Configuration Header—Base Address Register 0[3]

### Detailed Description and Projected Impact:

Bit 3 (taccess) is also known as the prefetchable attribute bit. This bit is hardwired to 1, indicating that the PPC190 is prefetchable.

Although it is memory mapped, most of the PPC190 is not 'well-behaved' memory, and should not be prefetched to avoid destructive reads. This bit in BAR0 does not control the PCI bus controller, it merely advises it that the PPC190VF is prefetchable. System software should configure the PCI bus controller to ignore the value of the taccess bit as read from the PPC190 during initialization, and substitute a value of zero, for 'non-prefetchable.'

### Work Around:

None

### Projected Solution:

Warning added to *MPC190 User's Manual*. Cacheable memory spaces are generally defined in the memory controller without regard to this bit.

**For More Information On This Product,**
**Go to: www.freescale.com**

## Errata No. 4:  AFEU Context-Dump Issue

### Detailed Description and Projected Impact:

Using a context-dump static descriptor on the AFEU leaves it in an undefined state. If the AFEU is not reset (because it is in static mode) and is not loaded with new context (which is the intended behavior), then the it is left expecting new context, and will not start up correctly when data is pushed into the FIFO.

This issue is observed when the current drivers try to break a >2048-byte block of data into two identical descriptors of less than 2048 bytes, and both descriptors try to unload context. This operational procedure is outside intended use models, but is not expressly prohibited by the specification.

### Work Around:

None

### Projected Solution:

Do not fix. Document this issue as an illegal mode. Changes to be made in *MPC190 User's Manual*.

## Errata No. 5: AFEU Context Loading and Unloading Erroneously Loaded into MDEU Input FIFO When Snooping

### Detailed Description and Projected Impact:

The MDEU works on the same data as AFEU/DEU by 'snooping.' The channel tells the controller to tell the MDEU to pick up the same data written to/read from the encryption EU FIFO. In normal operation, after the first ARC-4 key permute, AFEU context is preserved from packet to packet by unloading and reloading the S-box contents. This is done through the AFEU FIFOs.

When the AFEU outputs data and context to the FIFO, the channel tells the controller to tell the MDEU to snoop context as well as snooping actual data.

The MDEU does not process the context (it correctly limits its processing to the byte count provided by the descriptor), so a correct authentication result is output by the MDEU. Upon completion, however, the MDEU generates an input FIFO non-empty error, indicating it has unprocessed data in its input FIFO (the unneeded AFEU context).

### Implications for IPSec:

IPSec can use ARC-4, but typically does not. When dynamic mode ARC-4-HMAC (MD-5 or SHA-1) is selected, the HMAC is performed on the ciphertext, plus part of the header, as it is with a DES or 3DES encrypt. In this situation, our current method of in-snooping and out-snooping works, except for the FIFO non-empty error received at the end of a dynamic descriptor. This error could be masked, and the reset between descriptors would clear the MDEU input FIFO.

There is non-issue in static mode IPSec using ARC-4, since the AFEU context would not be unloaded. Also, there is a non-issue in debug mode, because snooping does not work in debug mode.

### Implications for SSL 3.1/TLS:

SSL 3.1/TLS processing requires the HMAC to be performed on the plaintext before encryption of the plaintext + HMAC. The MPC190 will require three descriptors to do this, although two options exist for actual data and descriptor flow. The potential for MDEU snooping to pull-in AFEU context can be avoided by the method of descriptor chaining.

### Work Around:

None

### Projected Solution:

Do not fix. Documentation will be updated to advise users of this issue, and a low-level driver will implement descriptor chaining methods to perform IPSec or SSL in a manner that does not require masking an MDEU 'input FIFO non-empty' error. Masking the input FIFO non-empty error will also be documented as a work around.

**For More Information On This Product,
Go to: www.freescale.com**

## Errata No. 6: 2Key 3DES Parity Error

### Detailed Description and Projected Impact:

The DEU supports single DES and triple DES using either 2Key (112-bit key) or 3Key (68-bit key) modes. The type of triple DES keying is determined by the number of bytes a key specifies in the descriptor.

While the DEU correctly determines, based on an internal 'keysize' register, whether to reuse key1 or use key3 for the third key, it always computes incorrect parity on key3. If keysize specifies 2-key triple DES, then the key3 register is left uninitialized, which will result in a KEY PARITY ERROR once the DEU is signaled to begin ciphering.

### Work Around:

- Disable key parity checking.
- If 2Key 3DES is desired, configure DEU for 3Key 3DES, but write key3 same as key1.

### Projected Solution:

Do not fix. The 2Key 3DES is a rarely used mode. Document this as an errata, with the above suggested work arounds. Neither work around has a noticeable effect on system performance. A wording change is being made to the *MPC190 User's Manual*. A low-level driver should be implemented, such that using 3Key 3DES (and writing key1 to both key1 and key3 when 2Key is selected) is transparent to the user.

## Errata No. 7:  MDEU Autopad

### Detailed Description and Projected Impact:

Autopadding for MD-5 and SHA-1 does not function reliably. Performing the suggested work around will lower system performance slightly.

### Work Around:

All packets must be padded at the application level. All descriptors with autopadding must not be used.

### Projected Solution:

Fixed in XPC190VFA.

**For More Information On This Product,**
**Go to: www.freescale.com**

## Errata No. 8: Controller Lock-Up Condition

### Detailed Description and Projected Impact:

The MPC190 is designed to act as a PCI 2.2-compliant master/slave, however, there are conditions under which a slave write to the MPC190 can cause the controller to lock up. The controller is the sole master within the MPC190. It translates all internal requests (from the crypto-channels) into external PCI bus requests, via the PCI interface block. For example, when the MPC190 has granted the PCI bus to perform a read, the controller provides the PCI interface block with the transaction parameters, then as data is received, it buffers and writes the data to the targeted internal address.

Most MPC190 bus transactions are short, related to the fetching of descriptors, keys, IVs, and the write back of HMACs and status. These transactions can be run before a nominal PCI latency timer value would expire. (The recommended setting for the MPC190 latency timer is 0x20, corresponding to 32 PCI cycles. In a 32-bit system, this 'time slice' is sufficient for the MPC190 to fetch descriptors, keys, and context without being preempted by the loss of BUS_GNT.) Long transactions by the MPC190 are typically related to data fetches. With a time slice of 32 PCI cycles, it is expected that the MPC190 would be preempted one or more times during the fetch of typical packet payloads.

When the MPC190 is preempted by another PCI bus master (MPC190 loses GNT, the other master gets it), the controller and PCI block suspend the current transaction, and begin re-arbitration for the bus in order to complete the existing transaction. This process works in all situations except when the other master uses its GNT to perform an immediate read or write of the MPC190 internal registers.

The PCI 2.2 specification allows GNT to be given to the other master well in advance of that master claiming ownership of the PCI bus. From the time it is given GNT, the other master watches for the MPC190 to complete its final data phase, and on the very next cycle, it can drive the address and transaction attributes on the PCI bus. When the MPC190 decodes its own address on the PCI bus, it asserts DEVSEL to claim the transaction as a target, and the other master can perform a read or write to MPC190 internal registers. In some cases, this slave access to the MPC190 internal registers collides with data from the MPC190 buffered read, causing the MPC190 controller to lose count of the remaining bytes to be fetched, and preventing it from properly initiating the next transaction.

This scenario is most likely to be encountered when the host processor is using multiple channels on the MPC190, and preempts a data read by one channel in order to setup or check status on a second channel. This could also be triggered by the read of a status register to determine if a single channel's interrupt signified ERROR or DONE. The fact that the operations which have been determined to cause this controller lock up have only been recently observed, suggests that many PCI masters do not take control of the bus as fast as PCI 2.2 allows them to do.

### Work Around:

According to the PCI specification, a master can start a transaction on the cycle following the last data phase of the previous master. While this does not appear to be the general case, the MPC8245 (Motorola's integrated communications processor with an internal PCI bridge), and likely other PCI devices are capable of generating the immediate read or write of MPC190 internal registers. To prevent the MPC190 from being preempted, the MPC190 must only perform single reads. This can be accomplished by setting the latency timer to 0x00, or by setting the single target read bit in the MPC190 controller.

### Projected Solution:

A more robust buffering scheme has been determined, and will be implemented on the MPC190VFB.

## Errata No. 9:    Enhancement: PKEU Mode Register Change

**Detailed Description and Projected Impact:**

The PKEUs in the MPC190VFB has been upgraded to reduce the number of descriptors needed to perform sign and verify operations. Specifically, a PKEU mode has been added which consolidates three lower level operations ($R^2$ mod N, mod mult, and mod exp) into a single routine called 'single step exponentiation.' See Section 8.1.2, "PKEU Mode Register," in the *MPC190 User's Manual,* Rev. 2.2, for additional details. This enhancement can significantly reduce CPU overhead associated with public key operations. An updated reference device driver using this more efficient PKEU routine will also be released in Q303.

**For More Information On This Product,**
**Go to: www.freescale.com**

## Errata No. 10:  Enhancement: Interrupt Mask Register (IMR) Change

### Detailed Description and Projected Impact:

The default setting of all XPC190 interrupt sources is 'unmasked.' To assist the user in avoiding multiple unintended interrupts prior to proper configuration of the 190, a new bit (bit 63, GIE) has been added to the controller's interrupt mask register—global interrupt enable. This bit, which resets to 'disabled,' allows the user to selectively mask individual interrupt sources in the interrupt mask register before enabling the remaining unmasked interrupt sources. See Section 5.1.3, "Interrupt Mask Register," in the *MPC190 User's Manual,* Rev. 2.2, for additional details.

## Errata No. 11:  Enhancement: Master Control Register (MCR) Change

### Detailed Description and Projected Impact:

To assist the user (and device driver) in determining which revision of XPC190 silicon is available to the system, a revision ID field has been added to the 190 master control register. This bit (32) will reset to 0 in the PPC190VF and XPC190VFA, and will reset to 1 in the XPC190VFB. Note that there is also a separate ID register in the XPC190, however, this ID register does not change from VF to VFA to VFB, and cannot be used by the driver to determine whether software work arounds to specific errata can be disabled, and enhancements 9–10 can be utilized. See Section 5.1.7, "Master Control Register," in the *MPC190 User's Manual,* Rev. 2.2, for additional details.

**For More Information On This Product,
Go to: www.freescale.com**

## Errata No. 12:  PKEU Address Error

### Detailed Description and Projected Impact:

Not all addresses in the PKEU map to functional registers or memories. The PKEU address error is meant to advise the user that a read or write has occurred to a hole in the PKEU address map, however, the PKEU signals address error (via the PKEU interrupt status register) for all accesses to the PKEU, regardless of whether the address falls into a hole.

### Work Around:

Mask all PKEU address errors via the PKEU interrupt control register. This is not a common error, and a true address error (read or write to a hole in the PKEU address map) will eventually cause a system timeout. When acting as a bus master, the MPC190 should never generate an address error.

### Projected Solution:

Will not be fixed. A warning will be added to the *MPC190 User's Manual*. The MPC190 device driver automatically masks this bit.

**For More Information On This Product,
Go to: www.freescale.com**

**Freescale Semiconductor, Inc.**

**THIS PAGE INTENTIONALLY LEFT BLANK**

Freescale Semiconductor, Inc.

**For More Information On This Product,
Go to: www.freescale.com**

**Freescale Semiconductor, Inc.**

**THIS PAGE INTENTIONALLY LEFT BLANK**

Freescale Semiconductor, Inc.

**For More Information On This Product,**
**Go to: www.freescale.com**

# Freescale Semiconductor, Inc.

**Freescale Semiconductor, Inc.**

**HOW TO REACH US:**

**USA/EUROPE/LOCATIONS NOT LISTED:**

Motorola Literature Distribution
P.O. Box 5405, Denver, Colorado 80217
1-480-768-2130
(800) 521-6274

**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

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

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola 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 which may be provided in Motorola 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. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

**MOTOROLA**

MPC190CE/D

**For More Information On This Product,
Go to: www.freescale.com**