MCF5475DE
Rev. 7, 03/2013

MCF5475 Chip Errata

Silicon Revision: All

Supports: MCF5470, MCF5471, MCF5472, MCF5473, MCF5474, and MCF5475

Introduction

This document identifies implementation differences between the MCF547*x* processors and the description contained in the *MCF5475RM: MCF547x Reference Manual.* Refer to freescale.com/ColdFire for the latest updates.

Summary of MCF547x Errata

All current MCF547*x* devices are marked as L14S mask set. The date code on the marking can be used to determine which errata have been corrected on a particular device as shown in Table 1. The datecode format is XXXYYWW, where YY represents the year and WW represents the work week. The three leading digits can be ignored.

Table 1. Summary of MCF547*x* Errata

Errata	Module Affected	Date Errata Added	Date Code Affected?		
			< XXX0445	≥ XXX0445 & < XXX0632	≥ XXX0632
SECF052	FlexBus	3/1/04	Yes	No	No
SECF053	PCI	3/1/04	Yes	No	No
SECF054	PCI	3/1/04	Yes	No	No
SECF057	PCI	3/31/04	Yes	No	No
SECF059	USB	8/17/04	Yes	No	No
SECF060	Reset	8/17/04	Yes	No	No
SECF062	PCI	8/17/04	Yes	No	No
SECF063	PCI	8/17/04	Yes	No	No
SECF065	PCI	8/17/04	Yes	No	No

Table continues on the next page...

Table 1. Summary of MCF547*x* Errata (continued)

Errata	Module Affected	Date Errata Added	Date Code Affected?		
			< XXX0445	≥ XXX0445 & < XXX0632	≥ XXX0632
SECF067	USB, PSC, FEC	8/17/04	Yes	No	No
SECF068	PCI	8/17/04	Yes	No	No
SECF070	PLL	2/28/2005	Yes	Yes	No
SECF071	PCI Arbiter	2/28/2005	Yes	Yes	No
SECF072	USB	2/28/2005	Yes	Yes	No
SECF076	USB	2/13/2006	Yes	Yes	No
SECF079	PCI	2/13/2006	Yes	Yes	No
SECF081	USB	5/2/2006	Yes	Yes	No
SECF082	USB	5/2/2006	Yes	Yes	No
SECF055	PCI	3/1/2004	Yes	Yes	Yes
SECF056	FlexCAN	3/1/2004	Yes	Yes	Yes
SECF058	PCI	8/17/2004	Yes	Yes	Yes
SECF061	DSPI	8/17/2004	Yes	Yes	Yes
SECF064	FEC	8/17/2004	Yes	Yes	Yes
SECF066	PCI	8/17/2004	Yes	Yes	Yes
SECF069	PCI	8/17/2004	Yes	Yes	Yes
SECF010	FEC	9/14/2004	Yes	Yes	Yes
SECF073	PCI	2/13/2006	Yes	Yes	Yes
SECF074	PCI	2/13/2006	Yes	Yes	Yes
SECF075	PCI	2/13/2006	Yes	Yes	Yes
SECF077	FlexBus	2/13/2006	Yes	Yes	Yes
SECF078	PCI	2/13/2006	Yes	Yes	Yes
SECF080	PCI	2/13/2006	Yes	Yes	Yes
SECF083	Cache V4	8/22/2006	Yes	Yes	Yes
SECF084	FlexBus	11/6/2006	Yes	Yes	Yes
SECF085	USB	12/4/2007	Yes	Yes	Yes
SECF135	PCI	10/28/2008	Yes	Yes	Yes
SECF175	FEC, PSC, PCI	4/21/2010	Yes	Yes	Yes
SECF219	SIU	3/12/2013	Yes	Yes	Yes

The table below provides a revision history for this document.

Table 2. Document Revision History

Rev. No.	Date	Substantive Changes
5	10/2008	Added XL Bus errata
6	4/2010	Added SECF175-FIFO
7	03/2013	Added SECF219-SIU

SECF010: FEC Interrupts will not Trigger on Consecutive Transmit Frames

Errata type: Silicon
Affects: FEC
Description: The late collision (LC), retry limit (RL), and underrun (UN) interrupts do not trigger on consecutive transmit frames. For example, if back-to-back frames cause a transmit underrun, only the first frame generates an underrun interrupt. No other underrun interrupts are generated until a frame is transmitted that does not underrun or the FEC is reset.

Workaround: Because late collision, retry limit, and underrun errors are not directly correlated to a specific transmit frame, in most cases a workaround for this problem is not needed. If a workaround is required, there are two independent workarounds:

- Ensure that a correct frame is transmitted after a late collision, retry limit, or underrun errors are detected.
- Perform a soft reset of the FEC by setting ECR[RESET] when a late collision, retry limit, or underrun errors are detected.

Fix plan: Currently, there are no plans to fix this.

SECF052: Multiplexed Address/Data Bus Address Hold Time Violation

Errata type: Silicon
Affects: FlexBus
Description: The multiplexed address/data bus does not provide enough hold time to latch the address phase with transfer start. If transfer start is used as the latch enable, the address phase of the multiplexed bus provides 0ns of hold time.

Workaround: The external latch can use a transfer start signal gated with the input clock to generate an early latch enable or registered clock edge.

Workaround: A common latch available (the 16374 series) latches the input data on the rising edge of an active-high latch enable (LE) signal. For these latches, an adequate latch enable (LE) signal can be generated by simply inverting the transfer start ($\overline{\text{TS}}$) signal.

Fix plan: Fixed in datecodes XXX0445 and later, by the following two changes:

- The $\overline{\text{TS}}$ signal was inverted to create an active high address latch enable (ALE) signal. ALE replaced the $\overline{\text{TS}}$ signal.
- The address setup option for the FlexBus chip selects was changed. On the first revision of silicon, address setup only delayed the assertion of chip select; the valid address was only held for one clock. For new revisions, address setup increases the address valid time as well as delaying the assertion of chip select. For example, if three clocks of address setup are selected, the full 32-bit address is valid for four clock cycles.

SECF053: External PCI Single Write XL Bus PCI Read Collision Error Under Hidden Arbitration

Errata type: Silicon

Affects: PCI

Description: A PCI read problem can occur when an external PCI write across the target interface of the PCI controller and a read to PCI across the XL Bus collide. The condition occurs when the arbiter has awarded the controller GNT# during the external PCI write transaction (hidden arbitration). The problem occurs for single-beat XTPCI writes only. An external PCI master writes a single beat of data to PCI_BAR0 or PCI_BAR1 space on the PCI bus while, internally, an XL Bus master is attempting to read from PCI. The target write is posted and stored in a buffer on the rising edge of the last PCI clock of the externally-mastered transaction. (The last cycle of the PCI transaction is the single cycle of data transfer for the write transaction.) On the next XL Bus clock, the XLB slave interface of the PCI controller retries the XL Bus read to enforce the write-before-read ordering requirement for PCI bridges. The read is issued a retry so the inbound write can proceed to the XL Bus. In the PCI clock domain, however, the initiator interface of the PCI controller was already awarded grant (hidden arbitration) and does not register the posted inbound write until the next PCI clock. It is in this cycle, the cycle before the PCI clock domain registers the posted target write, that the PCI controller is triggered to begin the read on PCI that was recently issued a retry on XLB. The trigger for the PCI in this cycle began for all of the following reasons:

- The PCI controller recognizes that the externally mastered transaction is done ($\overline{\text{DEVSEL}}$ deasserted).
- The PCI controller received grant (hidden arbitration).
- The XL Bus address tenure remains pending in the PCI clock domain. (The retry is not registered until the next PCI clock.)

The result of the collision is extra read data in the initiator read buffer. The inbound write completed on the XL Bus and the XLB master retries the same read to PCI. Data from the mastered PCI read that should not have happened (same address) is sent back to the XLB master for this read while another mastered PCI read begins at the same address. The data sent back for this retried XL Bus read is good, but the next read gets data from the previous read and so on.

Workaround: There are no software workarounds for this erratum.

The following are ways to avoid the condition:

- Never do single-beat writes from an external PCI master to the ColdFire processor. Always burst writes to the processor (>1-beat).
- Do all reads from processor to PCI across SCPCI.
- Use an external PCI arbiter that only rearbitrates and awards GNT# to the PCI controller when the PCI bus goes idle. (No hidden arbitration.)

Fix plan: Fixed in datecodes XXX0445 and later.

SECF054: Disconnected PCI Read Bursts When the MCF5487x Is Master Can Cause Corruption When Followed By Target Access

Errata type: Silicon
Affects: PCI
Description: If the processor generates a PCI read burst from its XL Bus to PCI interface and it is disconnected, corruption can be caused by an external master access. The external master access causes the PCI controller to flush its read FIFO and ARTRY the read on the XL Bus. However, the PCI controller continues to attempt the read at the address at which it was disconnected. When the internal master retries the read, it can get data fetched from the wrong address and the FIFO become out-of-sync.

Workaround: There are no software workarounds for this bug. To avoid encountering the problem, do not generate accesses on the XL Bus that can cause burst reads on the PCI bus. This includes 64-bit accesses on the XL Bus.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF055: Type 1 Configuration 32-bit Accesses Using PCICAR

Errata type: Silicon
Affects: PCI
Description: As shown below, an unexpected 2-beat transfer is issued on the PCI bus when a Type 1 configuration PCI cycle is performed using the Configuration Mechanism and more than 3 bytes are being transferred. The byte enables are correctly driven, but a second false data transfer is issued.

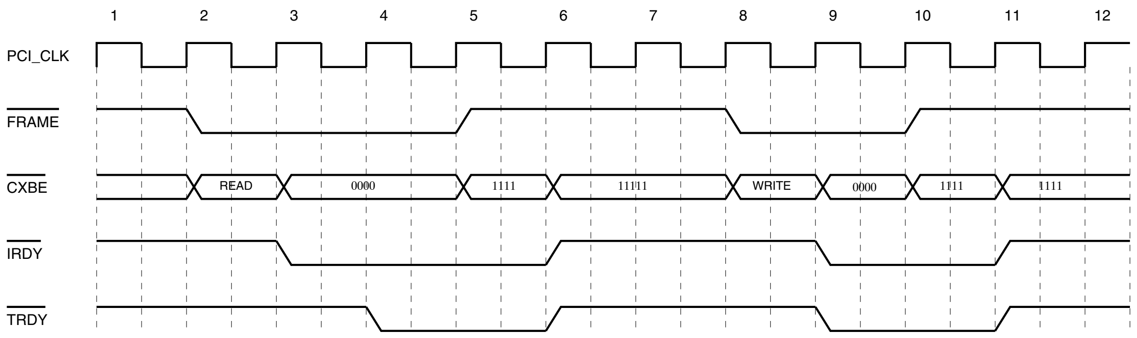

Figure 1. PCI bus 2-beat transfer

For writes, there is no harm done other than one or more (if a disconnect occurs) wasted PCI cycles for the false data transfer. During reads, two beats of data, the good data and the false data, are pushed into an internal read data FIFO and only one is popped off the FIFO. The first 32-bit read completes as expected. Any read following this read, causes the transaction on the PCI bus as expected. However, internally the data returned for this next read is not new data but what remained in the FIFO from the first two-beat read, the second false data transfer. The addition of the false data in the read FIFO causes subsequent reads to be out-of-sync with the data in the FIFO.

Workaround: Only issue byte or 2-byte type 1 configuration reads. Addresses at byte offsets retrieve the correct bytes. This avoids the two-beat read transactions.

Workaround: Use the SCPCI RX interface to issue 32-byte type 1 configuration reads to PCI.

Workaround: After issuing a 32-bit type 1 configuration read via the configuration mechanism (using PCICAR), clear the read data FIFO of the false data by immediately issuing an XLB transaction that results in a master abort. For instance, after a type 1 configuration read (to bridge), change the bus number in the PCICAR to an unused bus and issue another type 1 configuration transaction (read or write). A read returns all 1s. The master abort status bits flag, PCISCR[MA] and PCIISR[IA]. If subsequent configuration reads across the bridge are needed, reprogram the bus number to the valid value and continue interleaving the master abort transaction between each.

Fix plan: Currently, there are no plans to fix this.

SECF057: PCI Target Interface Prefetch from Slow Memory

Errata type: Silicon
Affects: PCI
Description: When the PCI controller prefetches target read data, an error can occur if internal memory inserts four or more wait states between data transfers of an internal burst. Prefetching applies to a hit in PCI_BAR1 space and is performed in response to a PCI memory-read-multiple command or if the prefetch software bit, PCITCR[P], is set. If the PCI target read crosses a 32-byte line boundary in prefetchable space, the next 32-byte line is fetched from internal memory and stored in a prefetch buffer in the target interface of the PCI controller. If the PCI read master terminates on the 0x18 longword of prefetched read data (the second to last longword of a 32-byte access) and the XL Bus transfer of this last beat of internal memory occurs at the same time, the next PCI read transaction can see corrupt data. The prefetch buffer resets and continues to control data transfer to the PCI bus. The actual data sent back on the next read is data from the prefetch buffer.

To meet the above requirements, the initial PCI read must terminate normally, by deasserting $\overline{\text{PCIFRAME}}$ on the last data phase, at the 0x18 longword address of the prefetch line. Therefore, the PCI transfer must have crossed a 32-byte boundary. Only reads bursts of 8-beat or more can end on the 0x18 longword and have also crossed a 32-byte boundary. (For example, read data of a PCI burst 0x18, 0x20, 0x24, 0x28, 0x2c, 0x30, 0x34, 0x38).

Workaround: Map PCI_BAR1, prefetchable memory, to DRAM only. DRAM inserts no more than one wait state between data transfers of the same burst.

Workaround: When target prefetch is enabled to slow prefetchable memory like FlexBus, align external PCI read bursts to avoid ending bursts at misaligned, 0x18, longwords.

Workaround: When target prefetch is enabled to slow prefetchable memory like FlexBus, do not allow 8-beat or longer external PCI read bursts.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF058: $\overline{\text{PCIPERR}}$ Implemented as an Open-Drain Signal

Errata type: Silicon
Affects: PCI

Description: The PCI parity error signal, $\overline{\text{PCIPERR}}$, is used for reporting data parity errors during all PCI transactions except a special cycle. $\overline{\text{PCIPERR}}$ is driven low two clock periods after the data phase with bad parity. It is driven low for a minimum of one clock period. $\overline{\text{PCIPERR}}$ is shared among all PCI devices and is driven with a tri-state driver. A required on-board pull-up resistor ensures the signal is sustained in an inactive state when no device is driving it.

After being asserted low, $\overline{\text{PCIPERR}}$ should be driven high one clock before being tri-stated to restore the signal to its inactive state. This would ensure the signal does not remain low in the following cycle because of a slow rise due to the pull-up.

The $\overline{\text{PCIPERR}}$ signal on the processor is implemented as open-drain. When the PCI controller detects a parity error, it synchronously asserts $\overline{\text{PCIPERR}}$, but instead of driving the signal high one clock before being tri-stated, it is tri-stated immediately. Therefore, when $\overline{\text{PCIPERR}}$ should negate after being driven asserted, it floats high asynchronously through the on-board pull-up resistor. The pull-up resistor may take more than one PCI clocks to restore the $\overline{\text{PCIPERR}}$ signal to its inactive state. The result is that after an initial parity error is detected by the PCI controller and reported on the $\overline{\text{PCIPERR}}$ signal, erroneous subsequent cycles of $\overline{\text{PCIPERR}}$ assertion may occur. The subsequent cycles of $\overline{\text{PCIPERR}}$ assertion would erroneously indicate subsequent beats of data parity errors.

Workaround: The parity error response bit in the PCI status/command register, PCISCR[PER], controls the processor's response to parity errors. Leave this bit cleared and the process never asserts $\overline{\text{PCIPERR}}$. The parity error detected status bit sets when a parity error is detected.

Fix plan: Currently, there are no plans to fix this.

SECF059: USB 2.0: UTMI Interface Timing Problems Between PHY/Controller

Errata type: Silicon
Affects: USB
Description: The USB 2.0 device controller and integrated PHY do not communicate correctly resulting in a non-functional USB device.

Workaround: No workaround.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF060: $\overline{\text{RSTO}}$ Does Not Assert During the Assertion of $\overline{\text{RSTI}}$

Errata type: Silicon
Affects: Reset
Description: During the assertion of $\overline{\text{RSTI}}$, the $\overline{\text{RSTO}}$ signal is driven high-impedance instead of being asserted. After $\overline{\text{RSTI}}$ negates, $\overline{\text{RSTO}}$ is asserted for 50,000 input clock cycles.

Workaround: No workaround.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF061: DSPI FIFO Underflow Ends Continuous Chip Select

Errata type: Silicon
Affects: DSPI

Description: When the DSPI is in master mode, if the transmit FIFO queue underflows during a continuous PCS transfer (CONT=1) then the peripheral chip select (PCS) negates, ignoring the continuous PCS, and the frame ends prematurely.

Workaround: Ensure that no underflow occurs during a multi-word transfer with continuous PCS. This is system dependent but may require:

- Adjusting DMA priorities
- If possible, having the transmit FIFO store the data for the complete transfer (multiple frame) before initiating a continuous PCS transfer.

Fix plan: Currently, there are no plans to fix this.

SECF062: PCI Type 0 Configuration Register Access

Errata type: Silicon
Affects: PCI
Description: The PCI Type 0 Configuration registers are not accessible to the processor through the PCI bus.

Workaround: Access the PCI Type 0 Configuration information by the internally memory mapped registers.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF063: PCI Target Interface Does Not Expect False Writes on the First Data Beat

Errata type: Silicon
Affects: PCI
Description: When the PCI controller has received a transaction in PCI target mode, indicated by a hit in PCI_BAR0 or PCI_BAR1 space, it expects data transfer on the first beat of a write transaction. If a write is issued with all byte enables not enabled on the first data beat of the transaction ($\overline{\text{PCICXBE}}$[3:0] = 1111) and the last target transaction was a read, the target interface responds, but subsequently hangs the controller.

Workaround: To avoid hanging the PCI controller, do not issue write transactions to the processor on PCI where the first data transferred has no $\overline{\text{PCICXBE}}$[3:0] signals active.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF064: FEC FIFO Status TXW Bit Erroneously Asserts on Tx Collision

Errata type: Silicon
Affects: FEC
Description: When the FEC is operating in half-duplex mode and a Tx collision occurs, the TXW bit in the FECTFSR can assert erroneously.

Workaround: Always mask the TXW condition by setting the FECTFCR[TXW_MASK] bit.

Fix plan: Currently, there are no plans to fix this.

SECF065: Retry Error Locks the PCI Bus

Errata type: Silicon
Affects: PCI
Description: When the Maximum Retries setting in the PCIICR is programmed to a non-zero value, the number of retry-disconnects received by a target during a write to an external PCI target device is counted. If the number of retries reaches the value stored in the Maximum Retries field, the write is aborted. If, when the retry limit is reached and grant to the PCI controller has deasserted, the PCI controller exhibits a broken master response. It asserts its request to again use the PCI bus, but when it is subsequently awarded grant, it does not issue a PCI transaction. The 16 PCI clock timeout can occur and the arbiter can thereafter ignore the controller's request. Because there is no mechanism to force the deassertion of the PCI Controller's request signal, the initiator path can forever be ignored by the arbiter.

Workaround: Program the Maximum Retries counter to 0x00 to allow for infinite automatic retries of writes.

Workaround: Only use the Retry Counter when operating as master so that grant is never deasserted to the PCI Controller.

Workaround: Use the Retry Counter in debug mode knowing that the broken master condition can occur.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF066: PCI FIFO Receive Full Burst Programming

Errata type: Silicon
Affects: PCI
Description: The Full Burst mode of the PCI DMA Receive Interface is a special mode that can only be used under strict data control conditions. Setting the Full Burst bit in the Rx Transaction Control Register, PCIRTCR[FB], supersedes the Max_Beats setting. If Full Burst is set, no check of the Receive FIFO fullness is done and the PCI transaction is immediately started when Packet_Size register is written and the Rx controller gains the PCI bus. The PCI transaction continues with multiple data beats until the full packet is transferred.

It has been found, when Max_Beats is set to 0x1 or 0x2, instead of ignoring Max_Beats, the Full Burst bit is ignored, and a single-beat PCI read is issued. After this, the packet transmission halts and further PCI mastering transactions are blocked.

Workaround: If using the full burst mode (PCIRTCR[FB] is set) program the Max_Beats field to a value other than 0x1 or 0x2. Valid values for Max_Beats when the Full Burst bit is set are 0x0, 0x3, 0x4, 0x5, 0x6, and 0x7.

Fix plan: Currently, there are no plans to fix this.

SECF067: Simultaneous FIFO Read/Write Results in Data Loss

Errata type: Silicon
Affects: FEC, PSC, USB
Description: It is possible that, under certain conditions, the data in a peripheral’s FIFO could become corrupted. The affected peripherals are the FECs, PSCs, and USB. The data corruption can occur under the following conditions:

- The FIFO Read Pointer must be mis-aligned. This means that RP[1:0]!= 00.
- There must be less than 16 bytes and more than 12 bytes of data remaining in the FIFO.
- The byte count must also satisfy the following conditions, which depend on the FIFO Read Pointer alignment:
    - RP[1:0]= 01: byte count - 11 >= 4
    - RP[1:0]= 10: byte count - 10 >= 4
    - RP[1:0]= 11: byte count - 9 >= 4
- A simultaneous data read and write from the two different FIFO ports (peripheral and CPU/DMA ports) must occur. The data read operation must read (as a single 8-bit read or included in a larger 16- or 32-bit read) the byte where (RP[1:0] == 11). The data write operation must write (as a single 8-bit write or included in a larger 16- or 32-bit write) the byte where (WP[1:0] == 11).

Workaround: No workaround.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF068: PCI Target Read Prefetch Data Corruption

Errata type: Silicon
Affects: PCI
Description: The problem arises when an initial PCI read request from prefetchable memory times out on PCI, because it requires more than 16 PCI clocks for the processor to get read data internally. This could happen if another master or masters are using the internal bus and blocking PCI, or if accesses to a slow peripheral are happening on the internal bus. If a write transaction to the target interface occurs after the read on PCI is retried by the PCI controller, write data posts in the controller. If the initial read request attempts again following the write transaction on PCI and the internal read remains pending on the internal bus at the time the PCI read returns, data corruption can occur. The combination of additional prefetching and the posted writes corrupt the read data buffers.

Workaround: Do not enable target prefetch for PCI reads.

Workaround: Set the LD bit and use the DMA interface for access to external PCI target space.

Fix plan: Fixed in datecodes XXX0445 and later.

SECF069: PCI XL Bus Initiator Interface Does Not Support Misaligned Reads

Errata type: Silicon
Affects: PCI
Description: The CPU or other internal masters may access the PCI bus via the XL Bus initiator interface. When a misaligned read from external PCI target space is attempted across the XL Bus Initiator interface, the read is acknowledged but never finishes. This hangs the internal system.

Transfer types that produce the error are:

- 2-byte transfer at address 0x03
- 3-byte transfer at address 0x02
- 3-byte transfer at address 0x03
- 4-byte transfer at address 0x01
- 4-byte transfer at address 0x02
- 4-byte transfer at address 0x03
- 5-byte transfer at address 0x00

- 5-byte transfer at address 0x01
- 5-byte transfer at address 0x02
- 5-byte transfer at address 0x03
- 6-byte transfer at address 0x00
- 6-byte transfer at address 0x01
- 6-byte transfer at address 0x02
- 7-byte transfer at address 0x00
- 7-byte transfer at address 0x01

Workaround: Ensure that all transactions to PCI via the XL Bus Initiator Interface are aligned. The CPU automatically issues aligned transactions.

Fix plan: Currently, there are no plans to fix this.

SECF070: PLL Clock Misalignment

Errata type: Silicon
Affects: PLL
Description: The system PLL generates the clocks used by the CPU and all the integrated peripherals based on the input clock, CLKIN. Following a system reset, it is possible for one or more of these output clocks to be misaligned in relation to the others. This misalignment results in malfunctions when accesses across affected clock boundaries are attempted.

Workaround: A power-on self test can be used to detect a malfunction; however, one of the malfunctions is the failure to boot from a non-volatile device. Therefore, some applications require an external watchdog device capable of resetting the system if indication of a successful power-on self test is not received. The self test should access and verify the response from all the integrated peripherals and from all external memory devices.

Fix plan: Fixed in datecodes XXX0632 and later.

SECF071: PCI Bus Request Nevermind Condition

Errata type: Silicon
Affects: PCI
Description: If, while the PCI bus is idle, a PCI device asserts its request and then deasserts it after being granted the bus, a subsequent assertion of the request can be ignored. The arbiter parks with the last master to use the bus. Internally, the arbiter thinks it has parked with the device that asserted and deasserted the request. A new request from this device is ignored. If any other master asserts its request for two PCI cycles, the arbiter asserts grant for this new master. At this point, the incorrectly ignored master is serviced by the arbiter.

The internal PCI controller can nevermind a request for the PCI bus if the request is for an XL bus-to-PCI read and an access to local memory occurs during the request assertion.

Workaround:

Disable the internal PCI arbiter and use an external PCI arbiter.

Workaround:

Implement an external interval timer to pulse an unused $\overline{\text{PCIBR}}$ pin at time intervals shorter than the internal programmable XL bus timer. Initially drive the unused $\overline{\text{PCIBR}}$ line high (inactive) and pulse low for two PCI clock cycles to issue a dummy request to the internal PCI arbiter to get the arbiter out of its bad state.

Workaround:

If no other devices in the system request and never-mind as described, then the ColdFire processor is the only device that can enter this condition. There are three software workarounds to avoid the ignore condition:

- Use CommBus PCI interface for XL bus to PCI target reads
- Only issue external writes to local memory as bursts (2-beats of data or more)
- Program the external PCI master to read back the last write to local memory

Fix plan: Fixed in datecodes XXX0632 and later.

SECF072: USB Controller IN EP Data Loss

Errata type: Silicon
Affects: USB
Description: There is an erratum in the endpoint (EP) data management logic that causes data in an IN EP FIFO to be lost.

The error is encountered when:

- There is an endpoint, EPx, configured as an IN EP.
- There is another device on the USB that has an IN EP with the same logical endpoint number as EPx.
- The host issues an IN packet request to EPx of the other device on the USB at a time when the processor EPx FIFO contains between 1 and 4 bytes of data from the same frame (this can be the last few bytes of a large frame or an entire frame in itself, but there must only be data for one frame in the EPx FIFO.)

When these conditions are met, the controller reacts to transactions intended for another device on the bus. Specifically, the data is removed from the EPx IN FIFO and lost. Furthermore, if a valid OUT packet is sent to another endpoint, EPy, configured as an OUT endpoint on the processor, the EPy FIFO is corrupted with the data removed from the EPx FIFO.

Workaround: Isolate the processor USB device to a USB tree with devices that do not duplicate any logical IN endpoint numbers that exist on the ColdFire processor.

Fix plan: Fixed in datecodes XXX0632 and later.

SECF073: PCI XL Bus Initiator Interface Does Not Support Misaligned Writes

Errata type: Silicon
Affects: PCI
Description: When a misaligned write that crosses the 32-bit address boundary to external PCI target space is attempted across the XL bus initiator interface, the write is issued on the PCI bus as a single-beat write only. The second PCI beat of write data is left in the internal write FIFO. This beat can corrupt the next XIPCI and block reads across the controller in either direction.

Here are the transfer types that produce the error:

- 2-byte transfer at address 0x03
- 3-byte transfer at address 0x02
- 3-byte transfer at address 0x03
- 4-byte transfer at address 0x01
- 4-byte transfer at address 0x02
- 4-byte transfer at address 0x03

Workaround: Ensure that all transactions to PCI via the XL bus initiator interface are aligned. The CPU automatically issues aligned transactions.

Fix plan: Currently, there are no plans to fix this.

SECF074: PCI Target Split Writes That Return An Internal Error Can Hang the Target Interface

Errata type: Silicon
Affects: PCI
Description: During PCI target split writes, where byte enables are non-contiguous, a problem with the controller can occur if an internal error is returned. The internal bus does not support single transactions with non-contiguous bytes transferred. An example of split write is when the active low PCI byte enables are driven 0b0100. To transfer these 3 bytes, two back-to-back transactions on the internal bus occur; a 2-byte transfer followed by a 1-byte transfer. When a $\overline{\text{TEA}}$ is returned on first transfer of the two XL bus transfers, it requires to complete the PCI write. No matter what kind of target interface PCI transaction is occurring when the XLB transaction occurs, a target abort is returned. Additionally, the write data is removed from the internal write FIFO until data is transferred to the XL bus. The same write continues to attempt until the first part of the transfer completes without an XL bus error. Every time a $\overline{\text{TEA}}$ is returned on the first XLB transfer, a target abort is seen on the PCI bus if a target interface externally mastered PCI transaction is in progress.

Workaround: No workaround. An XL Bus $\overline{\text{TEA}}$ is a catastrophic internal error. In general, any problems that occur after a $\overline{\text{TEA}}$ in a system occurs is secondary to the reason the $\overline{\text{TEA}}$ occurred.

Fix plan: Currently, there are no plans to fix this.

SECF075: PCI Target Interface Cache Wrap Mode Support

Errata type: Silicon
Affects: PCI
Description: As PCI target, when the cache wrap register is programmed to a power of two, the PCI controller attempts to transfer data for cache wrap addressed PCI transactions (PCI_AD[1:0] = 0b10). Target writes are handled correctly, and data is transferred correctly to internal memory. Target reads, on the other hand, do not fetch correct data at the wrap. Wrapping does not occur internally and therefore, the data after the wrap boundary, if transferred, is bad.

Workaround: Leave the cache line size register programmed to 0x00. With the cache line size register programmed to 0x00, the processor treats the cache wrap mode as a reserved memory mode. The first beat of data is correctly returned and a disconnect without data occurs on the second data phase. A value of 0x00 for the cache line size register also affects XL bus initiator commands. If the XL bus transaction is a burst read and the cache line size register is not 0x8, a Memory Read command is issued instead of a Memory Read Line command.

Fix plan: Currently, there are no plans to fix this.

SECF076: USB Controller Unable to Resume the Bus

Errata type: Silicon
Affects: USB
Description: A USB device may resume a suspended bus if its remote wakeup capability has been enabled by the host. The USBCR[RESUME] bit is used to initiate resume signalling. However, this feature does not work properly, and the bus remains in the suspended state. There is no way for the USB device controller to resume a suspended bus.

Workaround: No workaround. The USB device should not indicate that it is capable of remote wakeup in its configuration descriptor.

Fix plan: Fixed in datecodes XXX0632 and later.

SECF077: FlexBus Hang in 4:1 Clock Ratio

Errata type: Silicon
Affects: FlexBus
Description: When the FlexBus is operating at FB:XLB=1:4 clock ratio and the internal bus pipeline is enabled (XARB_CFG[PLDIS] = 0), the FlexBus may cause a system hang.

Workaround: If the FlexBus is operating at FB:XLB=1:4 clock ratio and the internal bus pipeline is enabled (XARB_CFG[PLDIS] = 0), enable at least two cycles of Flexbus address hold (CSCRn[RDAH] > 00).

NOTE

The XLB pipeline functionality is disabled by default.

Fix plan: Currently, there are no plans to fix this.

SECF078: PCI Output Hold

Errata type: Silicon
Affects: PCI
Description: The PCI controller does not provide any output signal hold time. The PCI Local Bus Specification Rev. 2.2 (found at http://www.pcisig.com) requires 1ns of hold time when operating the bus at 66 MHz and 2ns of hold time when operating at 33 MHz. This output hold requirement, though not explicitly specified, is necessary to overcome the allowable PCI clock skew. Please refer to the t_{val} parameter in Table 4-6, Table 7-4, and footnote 8 under Table 7-4 of the PCI specification.

Workaround: The allowable PCI clock skew is the source of this issue. The PCI clocks to the processor and other PCI devices should be routed to ensure that the clock to the processor does not lead that of the other devices.

Fix plan: Currently, there are no plans to fix this.

SECF079: $\overline{\text{PCIRESET}}$ Not Asserted During $\overline{\text{RSTI}}$ Assertion

Errata type: Silicon
Affects: PCI
Description: $\overline{\text{PCIRESET}}$ should be asserted active low as soon as $\overline{\text{RSTI}}$ is detected as asserted. Instead, $\overline{\text{PCIRESET}}$ is not driven until $\overline{\text{RSTI}}$ is negated.

Workaround: Add a pull-down to the $\overline{\text{PCIRESET}}$ line.

Workaround: AND the $\overline{\text{RSTI}}$ line with $\overline{\text{PCIRESET}}$ out to generate a logic level low on the reset signal going to all PCI devices.

Fix plan: Fixed in datecodes XXX0632 and later.

SECF080: PCI Master Abort May Follow An Issued Target Abort

Errata type: Silicon
Affects: PCI
Description: If the PCI controller issues a target abort and another external access follows the target aborted transaction within the next two cycles, the controller does not acknowledge the access. The response seen by the external device is a master abort. If an external access is issued in more than 2 cycles following the target abort issued by the PCI controller, the response is normal.

Workaround: Target aborts are issued for fatal reasons and can be corrected by modifying software to enable target address translation or accessing existing allocated target memory. During software debug, expect that a master abort may follow a target abort.>

Fix plan: Currently, there are no plans to fix this.

SECF081: Corrupted Accesses to 8/16-bit USB Registers

Errata type: Silicon
Affects: USB
Description: Reads from the 8- and 16-bit registers in the USB 2.0 device controller can occasionally be corrupted due to a synchronization issue between two clock domains. When corruption occurs, the most likely case is that the most-significant bit of the registers is read as cleared when it should be set.

Workaround: The interface from the CPU to the 8- and 16-bit USB registers is a 16-bit bus. If 32-bit accesses are performed, the access is broken up into two 16-bit accesses. The timing of the first access is such that corruption never occurs. Therefore, a workaround is to access all 8- and 16-bit registers with a single 32-bit access ensuring that the desired register is accessed in the first 16-bits.

For example, to access the BRTR at address MBAR+0xB107, perform a 32-bit access at MBAR+0xB106 and disregard the excess data.

Fix plan: Fixed in datecodes XXX0632 and later.

SECF082: Invalid Response to OUT Request

Errata type: Silicon
Affects: USB
Description: If the USB 2.0 device controller is operating in full-speed mode and an OUT request (data from host to device) is received that is larger than the available space in the FIFO, the device should response with a NAK packet. However, in this case, the USB controller does not send a status packet back to the host. The host usually retries the OUT transaction three times and then reset the bus.

Workaround: Ensure that there is always enough room in an OUT endpoint's FIFO for a maximum sized packet.

Fix plan: Fixed in datecodes XXX0632 and later.

SECF083: Branch Cache Hit on Extension Word Causes Unexpected Fetch Address Change

Errata type: Silicon
Affects: Cache V4
Description: The Version 4 ColdFire processor instruction fetch pipeline (IFP) implements a small 8-entry branch cache to reduce branch misprediction penalty and improve performance in loops. A branch prediction error occurs when a taken branch instruction is executed, creating a branch cache entry. Then, this same code is executed again with the former branch instruction now appearing as an extension word for another opcode. When the interpretation of the code stream is dynamically changed, a branch prediction error may occur.

In the past, Freescale had suggested using a TRAPF (word or long) instruction to remove an unconditional branch in the following if/then/else construct:

```
if (a = b)
                    x = 1;
                    else x = 2;
                    compiles to the assembly code:
                    cmp.l   Da,Db       (a,b, are in reg Da, Db)
                    beq.b   label1
                    moveq   #2,Dx       (x is in reg Dx)
                    bra.b   label2
                    label1: <op1>               ("moveq #1,Dx", for example)
                    label2: <op2>
```

where a TRAPF (word or long) can be substituted for the bra.b branch instruction and the subsequent

`<op1>`

instruction then effectively appears as the extension word of the TRAPF. (If a TRAPF.l is used,

`<op1>`

could represent two 16-bit instructions).

The instruction buffer logic can become confused if any

```
<op1>
```

instruction is a taken branch instruction, but the likelihood of this usage is expected to be low. The replacement of a branch instruction using this TRAPF construct is completely benign for cases where

```
<op1>
```

instructions are not conditional branch instructions.

A failing verification test example was created that demonstrates this extremely rare sequence-related problem in the IFP instruction buffer control. In the failing example below, the "bra.b L %b9C" at address 0x1914C is replaced by a TRAPF.w to effectively hide the bne.b when the code is executed sequentially. When a branch to 0x1914E occurs; however, the "66EC" is interpreted as a bne.b and, if taken, may be written into the branch cache. Later, if this same code is executed sequentially and if a branch cache hit occurs on the extension word at address 0x1914E, the prediction would change the instruction stream and subsequent fetches would get the wrong instruction data.

```
1914C: 51FA short   0x51fa      # bra.b L%b9C
                    L%b9B:
                    1914E: 66EC bne.b   L%b9A+0
                    L%b9C:
                    19150: A38B mov.l   %acc1,%a3
```

The result is that incorrect instructions could be executed, leading to data corruption and/or system hangs. In the failing example, the wrong instruction was executed for PC=0x19150 (data from the bne.b target was incorrectly used).

Workaround: Don't use the TRAPF.{w,l} construct to remove an unconditional branch.

Workaround: Allow the TRAPF.{w,l} construct, but don't let <op1> contain a conditional branch.

Workaround: Disable the branch cache, CACR[BEC] = 0, and accept lower branch performance.

Fix plan: Currently, there are no plans to fix this.

SECF084: Extra Wait State Added to Non-Burst Write Cycles

Errata type: Silicon
Affects: FlexBus
Description: When a chip select is configured for bursting and auto-acknowledge (BSTR, BSTW, and AA set), one extra wait state is added to any non-burst write access on the chip select. Read accesses have the correct number of wait states on the chip select line as programmed.

Workaround: No workaround

Fix plan: Currently, there are no plans to fix this.

SECF085: USB Device Controller May Lose Data in FIFO RAM

Errata type: Silicon
Affects: USB

Description: Under some conditions, the USB device controller on the processor can lose data in the FIFO RAM. This can result in lost received data (OUT packets), truncated transmitted data (IN packets), and failure to respond to IN requests from the USB host.

Workaround: There is no workaround for these errors using the on-chip USB device controller on the ColdFire processor. An external USB device controller can be connected to the Flexbus or PCI. Alternatively, Freescale offers a wide range of products comparable to the MCF5487x featuring fully operational USB controllers.

Fix plan: Currently, there are no plans to fix this.

SECF135: XL Bus Accesses Can Hang After PCI Transactions

Errata type: Silicon
Affects: PCI
Description: When core PCI transactions that involve writes to configuration or I/O space are followed by a core line access to line addresses 0x4 and 0xC, core access to the XL bus can hang.

Workaround: Prevent PCI configuration and I/O writes from being followed by the described line access by the core by generating a known good XL bus transaction after the PCI transaction.

Create a dummy function which is called immediately after each of the affected transactions. There are three requirements for this dummy function.

1. The function must be aligned to a 16-byte boundary.
2. The function must contain a dummy write to a location on the XL bus, preferably one with no side effects.
3. The function must be longer than 32 bytes. If it is not, the function should be padded with 16- or 48-bit TPF instructions placed after the end of the function (after the RTS instruction) such that the length is longer than 32 bytes.

```
create dummy_func.s:
/*
*/
    .global _dummy_function
    .global dummy_function
.text

/*********************************************************************
* This routine is the dummy function for PCI errata */

dummy_function:
_dummy_function:
    .align 16         /* force function start to 16-byte boundary.
             Can be done in linker file also */
    clr.l d0
    move.l d0, 0x10000F0C    /* Must use direct addressing. write to EPORT
module
                xlbus -> slavebus -> eport, writing '0' to register has no
                effect */
    rts
    tpf.l #0x0
    tpf.l #0x0
    tpf.l #0x0
    tpf.l #0x0
    tpf.l #0x0

    .end
```

Fix plan: Currently, there are no plans to fix this.

SECF175: Simultaneous FIFO Read/Write Results in Data Corruption

Errata type: Silicon
Affects: FEC
Affects: PSC
Affects: PCI
Description: It is possible that, under certain conditions, the data in the following peripherals' receive FIFO could become corrupted:

- Fast Ethernet Controller (FEC) modules
- Programmable Serial Controller (PSC) modules
- Peripheral Component Interface (PCI)

For the FEC, FIFO data is used to update the FEC buffer descriptors, so the data corruption could affect packet data and/or descriptor information.

For the data corruption to occur, all of the following conditions must be met:

- The FIFO read or write pointer is misaligned (RP[1:0] ≠ 00 or WP[1:0] ≠ 00),
- The FIFO contains less than 16 bytes of data, and
- A simultaneous data read from the CPU/DMA and write by the peripheral to the two different FIFO ports occurs.

Because a simultaneous read and write is required for the problem to occur, this situation is rare in most use cases. Slower line rates decrease the probability of encountering this issue. For example, the probability of seeing the problem on the PSC modules is much lower than on the FEC.

Workaround: Using communications protocols with CRCs/checksums and retry mechanisms can be used to mitigate this errata. Specifically for FEC communications, using TCP/IP with CRC checks at layer 3 and higher can workaround this problem. Protocols that do not have an appropriate checksum and automatic retry, such as UDP, are susceptible to this problem without an additional protocol check.

Workaround: When not using frame mode (RFCR[FRM_EN] = 0), program the FIFO control register granularity bits to avoid or eliminate situations where there are less than 16 bytes of data in the FIFO. If the FIFO needs to be read when there might be less than 16 bytes of data in the FIFO (to read the trailing end of an incoming packet), then disable the communications interface before reading the data from the FIFO using either the DMA or core. The port can be re-enabled after the data is read from the FIFO.

- If using the PSC, disable the receiver and read data from the FIFO. Use hardware flow control to prevent an external device from sending to the PSC while the receiver is disabled (PSCnRTS is automatically negated when the receiver is disabled).

Workaround: The PCI module is not affected when used as a bus master or slave module through the XLBus interface. The XLBus mode is the recommended mode of operation for the PCI controller.

Fix plan: Will not be fixed.

SECF219: SIU Software Watchdog Not Reliable in 1:4 Clock Mode

Release date: 3/12/2013

Errata type:	Silicon
Affects:	System Integration Unit (SIU) Software Watchdog
Description:	When using a clock ratio of 1:4 (FB AD[12:8] = 01111), the software watchdog reset may not assert and will cause undefined behavior in the system.
Workaround:	Use an external watchdog reset device or external latch to drive the reset pin, or use 1:2 clock mode when software watchdog reset is required.
Fix plan:	Currently there are no plans to fix this.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

Document Number: MCF5475DE
Rev. 7, 03/2013