Document Number: MPC8245CE
Rev. 10, 08/2005

MPC8245/MPC8241 Integrated Processor Device Errata

This document details all known silicon errata for the MPC8245/MPC8241 and its derivatives. The MPC8245 and MPC8241 are PowerPC™ integrated microprocessors. Table 1 provides a revision history for this chip errata document.

Table 1. Document Revision History

Rev. No.	Significant Changes
0–2.1	Earlier releases of document.
3	Added Error 18.
4	Added Error 19.
5	Added Errors 20 and 21. Updated projected solution status for Errors 18 and 19.
6	Updated errata fixes for Rev. 1.4 silicon. Updated work around for Error 15. Updated detailed description for Error 19.
7	Updated reference to Rev. B in work around for Errors 11 and 14. Added Errors 22 through 26.
8	Added Error 27.
9	Added Errors 28 and 29.
10	Corrected drive strength information for Error 19

freescale™ semiconductor

Table 2 provides a cross-reference to match the revision code in the processor version register to the revision level marked on the part.

Table 2. Revision Level to Part Marking Cross-Reference

MPC8241 Revision	MPC8245 Revision	Part Marking	Processor Version Register	Revision ID Register
—	1.0	Eng	0x80811014	0x10
1.1	1.1	A	0x80811014	0x11
1.2	1.2	B	0x80811014	0x12
1.4	1.4	D	0x80811014	0x14[1]

[1] Note that Rev. 1.3 = Rev. C parts are not production parts and are not referenced in this document.

Table 3 summarizes all known errata and lists the corresponding silicon revision level to which the erratum applies. A 'Y' entry indicates that the erratum applies to a specific revision level, and a dash ('—') entry means that the erratum does not apply.

Table 3. Summary of Silicon Errata and Applicable Revision

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
1	Reduced FastScan test coverage	The COP scan chain contains defects. The COP scan chain must be disabled for FastScan testing. Total test coverage is reduced to 25% coverage.	FastScan test patterns are not dependable for screening the part on the factory tester. Note that the CPU mode of scan test is not affected.	None	Y	—	—	—
2	Extended ROM mode not functional	In the extended ROM mode, the CPU's $\overline{\text{sreset}}$ input receives the logic state of the SDMA12 signal instead of from the internal system logic. The CPU randomly receives a soft reset when extended ROM mode is enabled. Similarly, for the TBEN signal, which becomes SDMA13 in extended ROM mode, the CPU's internal TBEN input signal receives the SDMA13 logic state when extended ROM mode is enabled. The time base counter is randomly enabled and disabled when extended ROM mode is enabled.	Extended ROM mode is not functioning correctly and should not be enabled.	None. Do not use extended ROM mode.	Y	—	—	—
3	Debug mode not functional	In debug address mode, the output enable of $\overline{\text{REQ4}}$ is not functioning. The $\overline{\text{REQ4}}$ signal does not drive the DA4 debug address signal.	Signal DA4 is not functioning correctly in debug address mode.	None. Do not rely on the state of the DA4 signal in debug address mode.	Y	—	—	—
4	PCI read from DUART register returns corrupted data	Corrupted data is returned when an external PCI master reads the DUART run time register. The byte-enable signals occur too late when the DUART logic returns the run time register's data. The DUART is returning the data based on the byte-enable signals from the previously successful transaction (which caused data transfer, either read or write) when the MPC8245 was a target during the previous transaction.	External PCI masters cannot read the DUART run time registers.	Perform a dummy PCI read (for example, a dummy read of the DMA run time register) with the byte-enables desired for the DUART access. Then perform a PCI read from the DUART register with the same byte-enable signals. Note: This will work assuming no other transactions occur between these two PCI reads which results in different byte-enable signals before the PCI read access from the DUART is completed.	Y	—	—	—

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
5	$\overline{\text{MCP}}$ driven during reset state	At reset, $\overline{\text{MCP}}$ should be sampled as an input configuration signal, but is driven instead by the MPC8245 as logic 1. The reset configuration register bit (inverted) that $\overline{\text{MCP}}$ defines is effectively hardcoded to 0b0 in PMCR2[6], which causes the CKE reset configuration signal (sets PMCR2[5]) to determine the effective PCI_HOLD_DEL (PMCR2[6–5]) configuration register setting value as either 0b00 or 0b01.	Reset configuration can only choose one of the two minimum extra output delay settings for PCI signals.	If the system requires extra output delay other than the 0b00 or 0b01 settings, software can change the PMCR2[6–5] setting after the system is reset.	Y	—	—	—
6	Output PCI clock signals not functional	The PCI output clock signals are not exported on the PCI_SYNC_OUT and PCI_CLK[0:4] signals.	Systems which rely on PCI_SYNC_OUT and PCI_CLK[0:4] signals may not function properly.	Systems should not use PCI_SYNC_OUT and PCI_CLK[0:4] signals that the MPC8245 provides. Systems should drive the PCI_SYNC_IN signal from an external clock source.	Y	—	—	—
7	PCI access to local system ROM address space in agent mode not functional	When the MPC8245 is operating in agent mode and an external PCI master attempts to access the local system ROM address space through in-bound ATU translation, the access is misdirected to the EXTENDED ROM space. For read accesses, the device that is mapped to that EXTENDED ROM space returns the data. For write accesses, data is written to the device that is mapped to that EXTENDED ROM space.	PCI accesses to local system ROM space are not functional.	None	Y	—	—	—
8	Performance monitor events 55–58 and 63–66 not functional	The MPC8245/MPC8241 performance monitor events (55–58 and 63–66) do not count correctly.	Cannot measure PCI request and grant delay for devices that utilize the MPC8245/MPC8241 internal PCI arbiter.	None	Y	Y	Y	Y
9	CBR refresh in sleep mode not functional	CBR (CAS before RAS) refresh cycles do not function when enabled in the sleep mode.	The MPC8245 cannot perform CBR refresh cycles when the MPC8245 is in sleep mode.	Use the self-refresh mode that the SDRAM devices provide internally.	Y	—	—	—

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
10	Driver mode control for memory output clock signals incorrect	The driver strength of the memory output clock signals (SDRAM_CLK[0:3] and SDRAM_SYNC_OUT) are controlled by register 0x73 bit 5:4, DRV_MEM_CTRL_1 and DRV_MEM_CTRL_2, instead of being controlled by configuration register 0x73 bit 1:0, DRV_MEM_CLK_1 and DRV_MEM_CLK_2.	The memory output clocks' driver strength control must be the same as the other memory interface signals.	None	Y	Y	—	—
11	Four signals fail COP/JTAG operations	For the HIGHZ, EXTEST, and CLAMP JTAG operations, the output enable signal path of the 4-pin signals TBEN, $\overline{\text{SRESET}}$, TRIG_IN, and $\overline{\text{CHKSTOP\_IN}}$ have an extra inversion. These four signals are not in the correct state for the HIGHZ, EXTEST, and CLAMP JTAG operations. COP/JTAG tools must be aware of this condition if they are using the HIZ COP instruction or manually setting boundary scan bits on the LSRL. Examples: For the JTAG HIGHZ instruction, all the signals should go to high-impedance state. However, the extra inversion causes the MPC8245/MPC8241 to drive the TBEN, $\overline{\text{SRESET}}$, TRIG_IN, and $\overline{\text{CHKSTOP\_IN}}$ signals. For the JTAG EXTEST instruction, if the boundary-scan sequence attempts to put the TBEN, $\overline{\text{SRESET}}$, TRIG_IN, and $\overline{\text{CHKSTOP\_IN}}$ signals to the output mode, due to the extra inversion, these signals are in the input mode instead.	The MPC8245/MPC8241 may experience problems on the TBEN, $\overline{\text{SRESET}}$, TRIG_IN, and $\overline{\text{CHKSTOP\_IN}}$ signals with the HIGHZ, EXTEST, and CLAMP JTAG operations and with the LSRL scan chain and HIZ COP instruction.	For Rev. A devices, the JTAG board level test program must be aware of this errata, and may need modification to work around the problem. (This solution may or may not be workable, depending on the testing system). Otherwise, this situation has no work around.	Y	Y	—	—

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
12	DLL may not lock under all conditions	The memory interface DLL is sensitive to operating variables that consist of voltage, frequency, temperature, and processing. For a given frequency, an associated feedback loop length range does not function reliably because the DLL intermittently loses lock.	Board designs must avoid certain SDRAM_SYNC_OUT to SDRAM_SYNC_IN feedback loop lengths to prevent intermittent loss of DLL lock.	See errata sheet for work around details.	Y	Y	—	—
13	SIN1 and $\overline{\text{CTS1}}$ are being driven at reset if DUART mode is enabled	When the power up enables DUART mode, configuration signals SDMA0, SIN1, and $\overline{\text{CTS1}}$ that used to be PCI_CLK(1) and PCI_CLK(3), become input to the MPC8245/MPC8241. However, after reset, the MPC8245/MPC8241 drives these two signals.	SIN1 and $\overline{\text{CTS1}}$ potentially have bus contention when reset is active.	None	Y	Y	—	—
14	UART2 will report break interrupt when software enables the 4-pin DUART mode	When the power-up configuration signal SDMA0 enables DUART mode, the MPC8245/MPC8241 is operating at 4-pin UART mode. Later, software can switch this mode to become 4-pin DUART mode. In that case, UART2 always has the break-interrupt status bit set. If the break interrupt is enabled, the CPU core receives the interrupt request.	Software will see the spurious break interrupt. The problem with the break interrupt causes a delay in clearing the status bit until valid data is received.	For Rev. A devices, software should use the loopback mode to send a valid character to clear the status bit before it can turn on the interrupt enable bit.	Y	Y	—	—

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
15	SDRAM_SYNC_IN and *sys_logic_clk* phase alignment	The MPC8245/MPC8241 has an internal delay in the feedback path for SDRAM_SYNC_IN with respect to the internal *sys_logic_clk* signal. The SDRAM_CLKs is not phase-aligned with *sys_logic_clk*.	The peripheral logic uses internal *sys_logic_clk* to latch input data and launch output data on the memory interface. The additional internal delay present by *sys_logic_clk* and the DLL causes SDRAM_CLK pins to be offset by the delayed amount. For pre-existing MPC8240 designs, this delay was not present. Those designs must analyze better timing that the MPC8245/MPC8241 provides, with respect to the MPC8240 memory interface, and determine if that is an issue. This delay affects the output and input hold timing. Because the MPC8245/MPC8241 improved output hold timing equal to the maximum delay, it does not become an issue. The MPC8245/MPC8241 input hold timing is programmable, however, and does not default to a value that accounts for this delay.	See errata sheet for work around details.	Y	Y	Y	Y
16	I^2C general broadcast feature is inoperable	The I^2C general broadcast feature on the MPC8245/MPC8241 is inoperable. Broadcast address detections are not passed to the system. In some cases, if the control register's broadcast enable bit is set, the I^2C could hang.	The I^2C broadcast feature is not operable because it detects broadcast addresses one I^2C clock late.	The only hardware work around requires that the logic adjusts from detecting a general broadcast address one I^2C clock too late.	Y	Y	—	—

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
17	Data parity errors on 60x bus single beat writes are not detected	MCP8245 fails to detect the write parity errors and does not invoke a machine check error if all of the following conditions are met: • A single beat 60x write is being performed. • The CPU bus has corrupted data. • RMW parity mode is turned on (MCCR2[RMW_Par] is set). • Either ECC or parity modes are enabled.	This problem occurs when working in ECC (where RMW must be on), or in normal parity modes, and CPU data gets corrupted in a single-beat 60x write transaction. As a result, for ECC transactions, the corrupted data is used to generate an ECC syndrome. Both are written to SDRAM without detection. For normal parity mode, the corrupted data is written into the SDRAM and the parity byte is recalculated for the entire line (with the corrupted data) and also written into SDRAM. For CPU-burst write transactions, the problem does not occur for either ECC or parity modes.	All CPU-to-local memory writes that require error reporting should be burst writes (cacheable, write-back accesses). This work around cannot be implemented for transactions that involve I/O devices that may not have caching capability.	Y	Y	Y	Y
18	5-V tolerance issue on MPC8245/MPC8241	MCP8245/MPC8241 fails for 5-V input PCI signals when LV_{DD} is set to 5 V and an external PCI device drives a 5-V signal to the MPC8245/MPC8241. If the input high voltage (V_{IH}) is greater than 1.2 V over the I/O buffer supply for PCI and standard (OV_{DD}), bus violations can occur.	MPC8245/MPC8241 is not 5-V PCI tolerant.	Set LV_{DD} to 3.3 V instead of 5 V. This work around draws a much higher current from the 5-V PCI device, and potentially can affect the long-term reliability. Do not use this approach for production purposes.	Y	Y	Y	—
19	Driver strength issue for DRV_STD_MEM	The driver strength for standard signals is not controlled by bit 6 of 0x73, but by bit 5 of 0x73. The drive strengths are 6 ohms and 40 ohms.	Using bit 6 to control the standard driver signals capability is not functional; use bit 5 instead. The following signals will have a driver strength of 40 or 6 Ω, depending on the setting of bit 5 of the Output Driver Control Register (0x73): PMAA[0:2], SDA, SCL, CKO, QACK, DA[10:6], MCP, MDH[0:31], MDL[0:31], PAR[0:7], and MAA[0:2].	Use bit 5 to control the driver capability for standard signals.	Y	Y	Y	Y

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
20	Type 2 fast back-to-back transactions result in data corruption	Type 2 fast back-to-back transactions access multiple targets sequentially. If a PCI master issues a type 2 fast back-to-back transaction to the MPC8245, the transaction causes data corruption for read and write transactions. Data that is read is corrupted. Write transactions write to incorrect locations or write bad data to a specified location.	Type 2 fast back-to-back transactions are not supported on the MPC8245.	Software should disable the ability to run fast back-to-back transactions on PCI master devices that can issue fast back-to-back transactions to the MPC8245. Clear bit 9 of the PCI Command Register in the external master device.	Y	Y	Y	—
21	Enabling the detection of PCI $\overline{\text{SERR}}$ does not work	Enabling bit 6 of the Error Enabling Register 2 (0xC4) should report $\overline{\text{SERR}}$ assertions that occur on the PCI bus at any time, regardless of whether the MPC8245 is the initiator, the target, or a non-participating agent. This action does not occur on the MPC8245, and bit 6 of the Error Detection Register 2 (0xC5) does not report any $\overline{\text{SERR}}$ assertions that occur on the PCI bus by an external PCI agent.	$\overline{\text{SERR}}$ assertions that occur by an external PCI agent are not reported by the MPC8245.	None	Y	Y	Y	—

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
22	**stfd** of uninitialized FPR can hang part	The 64-bit FPRs each have additional internal bits with which they are associated that specify the type of floating-point number in the register. These bits are set properly whenever the FPR is loaded. However, the part could power up with the internal bits randomly set. The FPR could be interpreted as containing a denormalized number with a mantissa that contains all zeros. If this random state is stored with an **stfd** instruction before a floating-point load operation corrects the internal bits, the part hangs while searching for a leading 1 in the mantissa. The **stfd** instruction is the only instruction that causes this behavior. Note that this problem was discovered when compiled code stored out FPRs before using them as scratch registers early in the boot sequence.	This error affects all systems that use floating-point operations.	When emerging from reset, initialize all the FPRs that will be used. The initialization value is not important.	Y	Y	Y	Y
23	PCI_SYNC_IN signal is not 5-V tolerant	The PCI_SYNC_IN signal of the MPC8245 is powered by OV_{DD}, which is a 3.3-V power source and does not support a 5-V power supply.	All parts require that OV_{DD} powers PCI_SYNC_IN. The range of the power supply is 3.0- to 3.6-V.	Power sources outside of the 3- to 3.6-V range must be scaled accordingly to be within the 3.0- to 3.6-V range.	Y	Y	Y	Y
24	MPC8245 does not detect assertion of $\overline{\text{PERR}}$ signal for a certain case	The MPC8245 fails to detect the assertion of the $\overline{\text{PERR}}$ signal when the following conditions are met: • The memory to PCI clock ratio is 2:1 or higher (for example, 2:1, 3:1, and 4:1). • The MPC8245 is the initiator of the PCI bus transaction. • A parity error occurs in the last data transfer of the transaction (for either single- or multiple-beat transactions).	The MPC8245 fails to detect the assertion of $\overline{\text{PERR}}$ and does not report the parity error to the core by means of the internal $\overline{mcp}$ signal. Potentially corrupt data may be propagated.	Use external logic to monitor the $\overline{\text{PERR}}$ signal and assert the NMI signal when a data parity error is detected on the last data transfer.	Y	Y	Y	Y

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
25	MPC8245 does not detect assertion of $\overline{\text{MCP}}$ signal for a certain doorbell register case in the messaging unit	The MPC8245 fails to detect the assertion of the $\overline{\text{MCP}}$ signal whether in host or agent mode when bit 31 of the inbound doorbell register (IDBR) is set, even if the requirements for an $\overline{\text{MCP}}$ are met. The requirements for an $\overline{\text{MCP}}$ in this special case include: HID0[EMCP] = 1, PICR1[MCP_EN] = 1, MSR[ME] = 1. Set bit 31 of the IDBR while the IMIMR[DMCM] = 0. Even if the conditions described in the paragraph above are met, an $\overline{\text{MCP}}$ does not occur as asserted long enough to be recognized by the processor.	The internal $\overline{\text{MCP}}$ signal is asserted for only one clock. Because the processor requires the internal $\overline{\text{MCP}}$ signal to be held asserted for at least two clocks, the MPC8245 fails to detect the assertion of $\overline{\text{MCP}}$ in this case. Apart from not getting an $\overline{\text{MCP}}$ for the above description, the MPC8245 does not take a machine check exception even if enabled and bits 8, 7, or 4 of the Inbound Message Interrupt Status Register are set (IMISR[8–7, 4] = 1).	Use interrupts in the messaging unit to generate an $\overline{\text{MCP}}$.	Y	Y	Y	Y
26	Concurrent writes occur to the UART registers when UAFR[0] is set	The UART Alternate Function Register (UAFR) enables software to write concurrently to both UART1 and UART2 registers with the same write operation. However, this action occurs when the UARTs are individually written to, but in addition, concurrent writes to the UART registers occur as a result of writing to other EUMBAR registers, including the error injection registers, when UAFR[0] is set.	Concurrent writes will occur to the UART registers when UAFR(0) is set and EUMBAR registers are written to.	Set UAFR(0) only for concurrent writes to UART registers. Clear this bit immediately afterwards.	Y	Y	Y	Y
27	Posted writes from host to I^2O queue port may result in data corruption with 32-bit SDRAM devices	For 32-bit SDRAM configuration, multiple I^2O queue port writes with store gathering may cause the last I^2O write to get lost. See errata sheet for a detailed description.	This problem is likely to occur during I^2O queue port writes and if configured for 32-bit SDRAM.	See errata sheet for work around details.	Y	Y	Y	Y

Table 3. Summary of Silicon Errata and Applicable Revision (continued)

No.	Problem	Description	Impact	Work Around	Silicon Revision			
					1.0	1.1	1.2	1.4
28	PCI reads from local memory may return stale data	When the MPC8245/MPC8241 issues a speculative read to local memory on behalf of the current PCI/DMA read access, if the speculative read hits the PCI-to-system-memory-write buffer (PCMWB) or the copyback buffer, the MPC8245/MPC8241 will read from the local memory before flushing the write data from the PCMWB or copyback buffer.	The MPC8245/MPC8241 may return stale data to PCI/DMA read access and results in data corruption.	Set 0xA0[0] or 0xE0[0]. See errata sheet for work around details.	Y	Y	Y	Y
29	PCI writes may not invalidate speculative read data	This error may occur when an incoming PCI read has triggered a prefetch for the next cache line.	Potentially stale data may be returned for PCI reads.	Currently there are two work arounds: 1. Insert a dummy PCI read from an unrelated memory location between the PCI write and the desired PCI read sequence. 2. Turn off speculative reading by clearing PICR1[2] and prevent external PCI masters from issuing memory-read-multiple commands.	Y	Y	Y	Y

Error No. 1: Reduced FastScan test coverage

Detailed Description:

The COP scan chain contains defects. The COP scan chain must be disabled for FastScan testing. Total test coverage is reduced to 25% coverage.

Projected Impact:

FastScan test patterns are not dependable for screening the part on the factory tester. Note that the CPU mode of scan test is not affected.

Work Arounds:

None

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 2: Extended ROM mode not functional

Detailed Description:

In the extended ROM mode, the CPU's $\overline{\text{sreset}}$ input receives the logic state of the SDMA12 signal instead of from the internal system logic. The CPU randomly receives a soft reset when extended ROM mode is enabled.

Similarly, for the TBEN signal, which becomes SDMA13 in extended ROM mode, the CPU's internal TBEN input signal receives the SDMA13 logic state when extended ROM mode is enabled. The time base counter is randomly enabled and disabled when extended ROM mode is enabled.

Projected Impact:

Extended ROM mode is not functioning correctly and should not be enabled.

Work Arounds:

None. Do not use extended ROM mode.

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 3: Debug mode not functional

Detailed Description:

In debug address mode, the output enable of $\overline{\text{REQ4}}$ is not functioning. The $\overline{\text{REQ4}}$ signal does not drive the DA4 debug address signal.

Projected Impact:

Signal DA4 is not functioning correctly in debug address mode.

Work Arounds:

None. Do not rely on the state of the DA4 signal in debug address mode.

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 4: PCI read from DUART register returns corrupted data

Detailed Description:

Corrupted data is returned when an external PCI master reads the DUART run time register. The byte-enable signals occur too late when the DUART logic returns the run time register's data. The DUART is returning the data based on the byte-enable signals from the previously successful transaction (which caused data transfer, either read or write) when the MPC8245 was a target during the previous transaction.

Projected Impact:

External PCI masters cannot read the DUART run time registers.

Work Arounds:

Perform a dummy PCI read (for example, a dummy read of the DMA run time register) with the byte-enables for the DUART access. Then perform a PCI read from the DUART register with the same byte-enable signals. Note that this tactic works if no other transactions occur between these two PCI reads that cause different byte-enable signals before the PCI read access from the DUART is completed.

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 5: $\overline{\text{MCP}}$ driven during reset state

Detailed Description:

At reset, $\overline{\text{MCP}}$ should be sampled as an input configuration signal, but is driven instead by the MPC8245 as logic 1. The reset configuration register bit (inverted) that $\overline{\text{MCP}}$ defines is effectively hardcoded to 0b0 in PMCR2[6], which causes the CKE reset configuration signal (sets PMCR2[5]) to determine the effective PCI_HOLD_DEL (PMCR2[6–5]) configuration register setting value as either 0b00 or 0b01.

Projected Impact:

Reset configuration can choose only one of the two minimum extra output delay settings for PCI signals.

Work Arounds:

If the system requires extra output delay other than the 0b00 or 0b01 settings, software can change the PMCR2[6–5] setting after the system is reset.

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 6: Output PCI clock signals not functional

Detailed Description:

The PCI output clock signals are not exported on the PCI_SYNC_OUT and PCI_CLK[0:4] signals.

Projected Impact:

Systems which rely on PCI_SYNC_OUT and PCI_CLK[0:4] signals may not function properly.

Work Arounds:

Systems should not use PCI_SYNC_OUT and PCI_CLK[0:4] signals that the MPC8245 provides. Systems should drive the PCI_SYNC_IN signal from an external clock source.

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 7: PCI access to local system ROM address space in agent mode not functional

Detailed Description:

When the MPC8245 is operating in agent mode and an external PCI master attempts to access the local system ROM address space through in-bound ATU translation, the access is misdirected to the EXTENDED ROM space. For read accesses, the device that is mapped to that EXTENDED ROM space returns the data. For write accesses, data is written to the device that is mapped to that EXTENDED ROM space.

Projected Impact:

PCI accesses to local system ROM space are not functional.

Work Arounds:

None

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 8: Performance monitor events 55–58 and 63–66 not functional

Detailed Description:

The MPC8245/MPC8241 performance monitor events (55–58 and 63–66) do not count correctly.

Projected Impact:

Cannot measure PCI request and grant delay for devices that utilize the MPC8245/MPC8241 internal PCI arbiter.

Work Arounds:

None

Projected Solution:

No plans to fix.

Error No. 9: CBR refresh in sleep mode not functional

Detailed Description:

CBR (CAS before RAS) refresh cycles do not function when enabled in the sleep mode.

Projected Impact:

The MPC8245 cannot perform CBR refresh cycles when the MPC8245 is in sleep mode.

Work Arounds:

Use the self-refresh mode that the SDRAM devices provide internally.

Projected Solution:

Fixed in Rev. 1.1 = Rev. A devices.

Error No. 10: Driver mode control for memory output clock signals incorrect

Detailed Description:

The driver strength of the memory output clock signals (SDRAM_CLK[0:3] and SDRAM_SYNC_OUT) are controlled by register 0x73 bit 5:4, DRV_MEM_CTRL_1 and DRV_MEM_CTRL_2, instead of being controlled by configuration register 0x73 bit 1:0, DRV_MEM_CLK_1, and DRV_MEM_CLK_2.

Projected Impact:

The memory output clocks' driver strength control must be the same as the other memory interface signals.

Work Arounds:

None

Projected Solution:

Fixed in Rev. 1.2 = Rev. B devices.

Error No. 11: Four signals fail COP/JTAG operations

Detailed Description:

For the HIGHZ, EXTEST, and CLAMP JTAG operations, the output enable signal path of the 4-pin signals TBEN, SRESET, TRIG_IN, and CHKSTOP_IN have an extra inversion. These four signals are not in the correct state for the HIGHZ, EXTEST, and CLAMP JTAG operations.

COP/JTAG tools must be aware of this condition if they are using the HIZ COP instruction or manually setting boundary scan bits on the LSRL.

Examples:

For the JTAG HIGHZ instruction, all the signals should go to high-impedance state. However, the extra inversion causes the MPC8245/MPC8241 to drive the TBEN, SRESET, TRIG_IN, and CHKSTOP_IN signals.

For the JTAG EXTEST instruction, if the boundary-scan sequence attempts to put the TBEN, SRESET, TRIG_IN, and CHKSTOP_IN signals to the output mode, due to the extra inversion, these signals are in the input mode instead.

Projected Impact:

The MPC8245/MPC8241 may experience problems on the TBEN, SRESET, TRIG_IN, and CHKSTOP_IN signals with the HIGHZ, EXTEST, and CLAMP JTAG operations and with the LSRL scan chain and HIZ COP instruction.

Work Arounds:

For Rev. A devices, the JTAG board level test program must be aware of this errata, and may need modification to work around the problem.(This solution may or may not be workable, depending on the testing system). Otherwise, this situation has no work around.

Projected Solution:

Fixed in Rev. 1.2 = Rev. B devices.

Error No. 12: DLL may not lock under all conditions

Detailed Description:

The memory interface DLL is sensitive to operating variables that consist of voltage, frequency, temperature, and processing. For a given frequency, an associated feedback loop length range does not function reliably because the DLL intermittently loses lock.

Projected Impact:

Board designs must avoid certain SDRAM_SYNC_OUT to SDRAM_SYNC_IN feedback loop lengths to prevent intermittent loss of DLL lock.

Work Arounds:

Using the charts shown in Figure 1 through Figure 4, find the clock speed of the memory interface for the design in question by finding the associated clock period in nanoseconds (ns) on the Y axis. Then use the black band associated with that frequency to index delay times in ns (X axis) that must be avoided when designing the feedback loop for the design. The grey bands indicate valid operating points for different DLL mode settings and represent where the DLL locks. (Do not use white spaces: the DLL does not lock in any non-grey areas.) The black band represents regions; if the DLL is locked in these regions it will be unstable. The width of the black and grey bands take into account temperature, voltage, and processing variations. For any given part, the actual failure region is more narrow than what is shown in Figure 1 through Figure 4. Consult the *MPC8245 Integrated Processor Hardware Specifications* or the *MPC8241 Integrated Processor Hardware Specifications* for more information on DLL locking and the settings. In addition, see Application Note AN2164, *MPC8245/MPC8241 Memory Clock Design Guidelines*.

Projected Solution:

Fixed in Rev. 1.2.

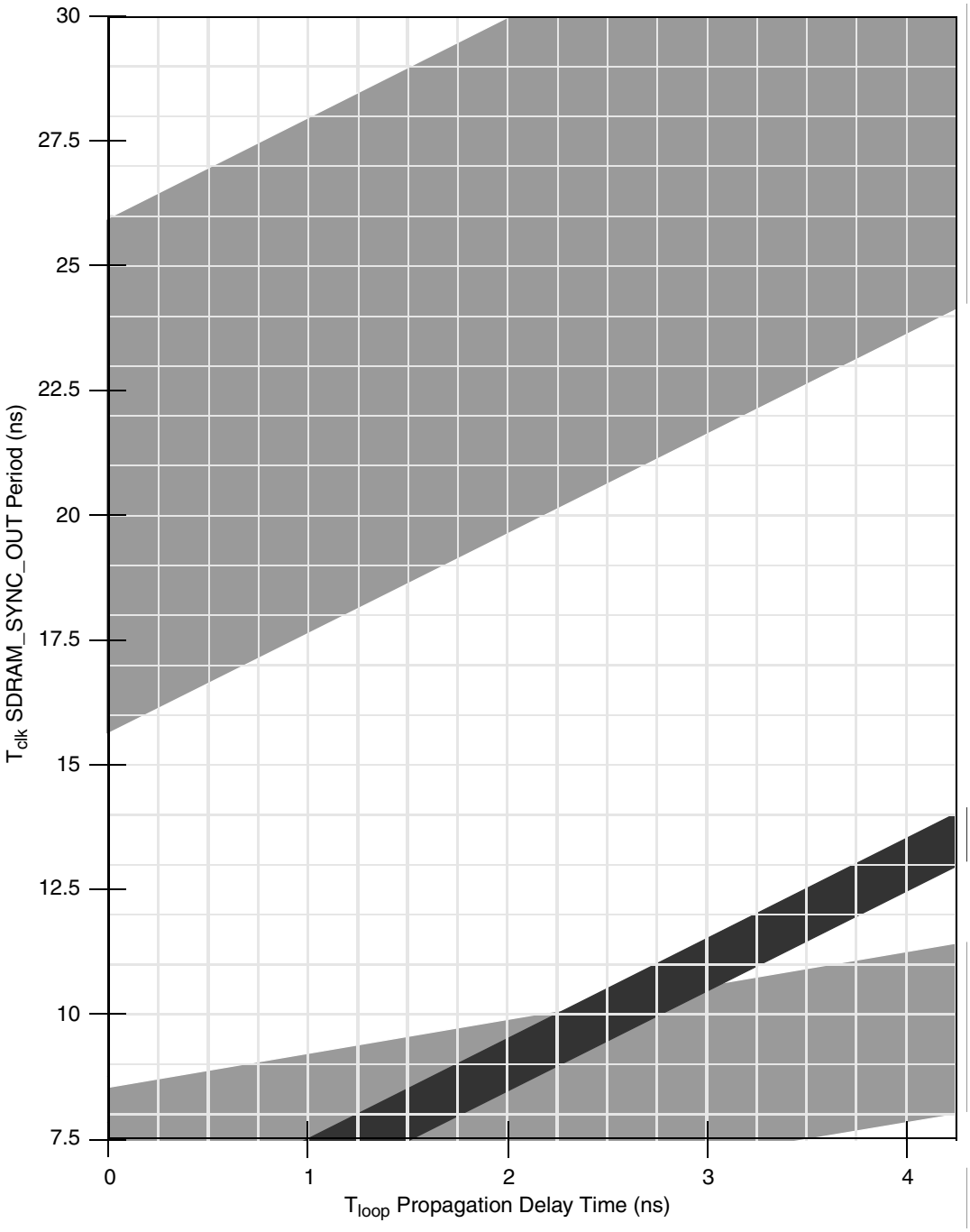

Figure 1. Work Around Chart 1: (DLL_Extend = 1 and Normal Tap Delay) (DLL Locking Range versus Frequency of Operation)

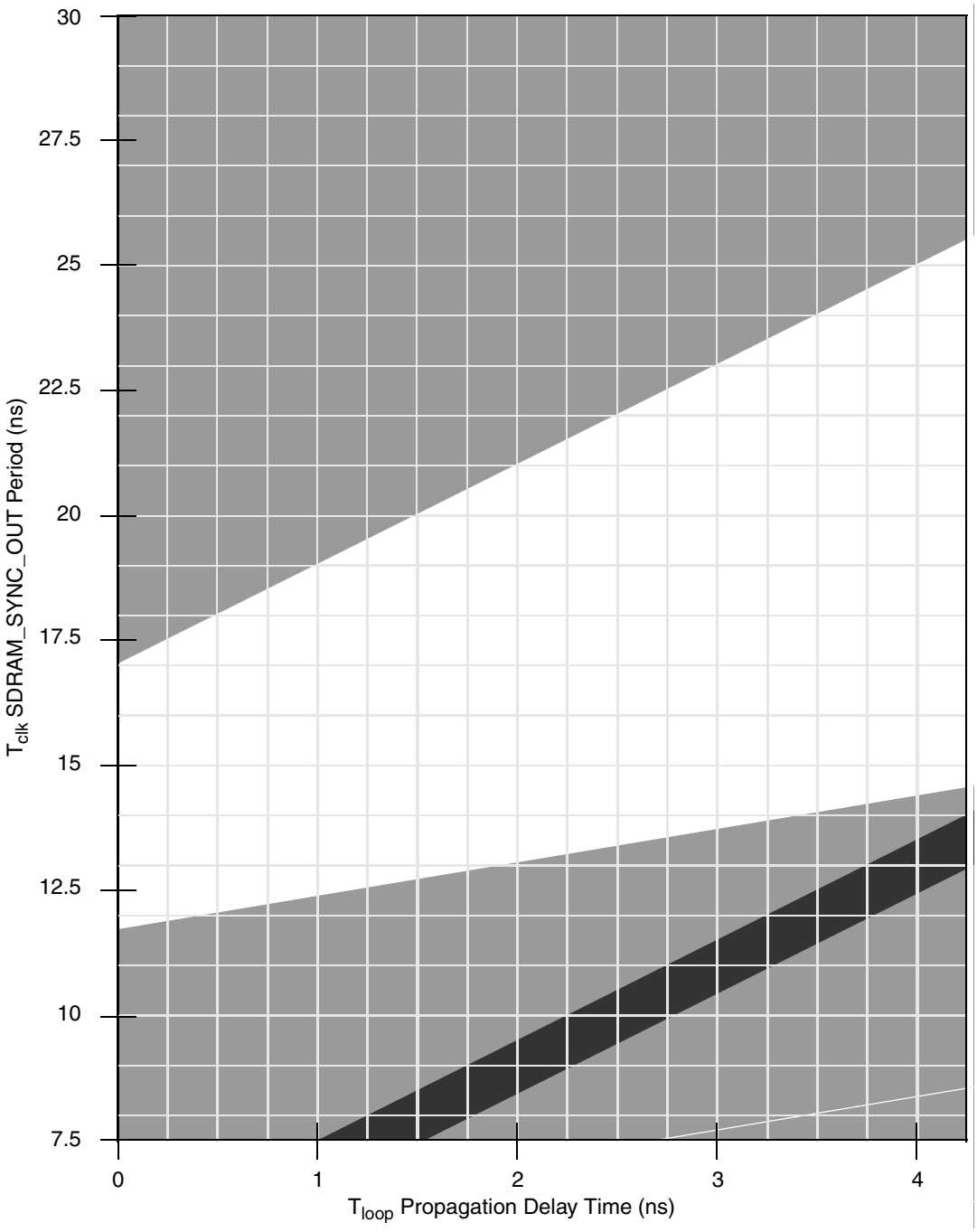

Figure 2. Work Around Chart 2: (DLL_Extend = 1 and Maximum Tap Delay) (DLL Locking Range versus Frequency of Operation)

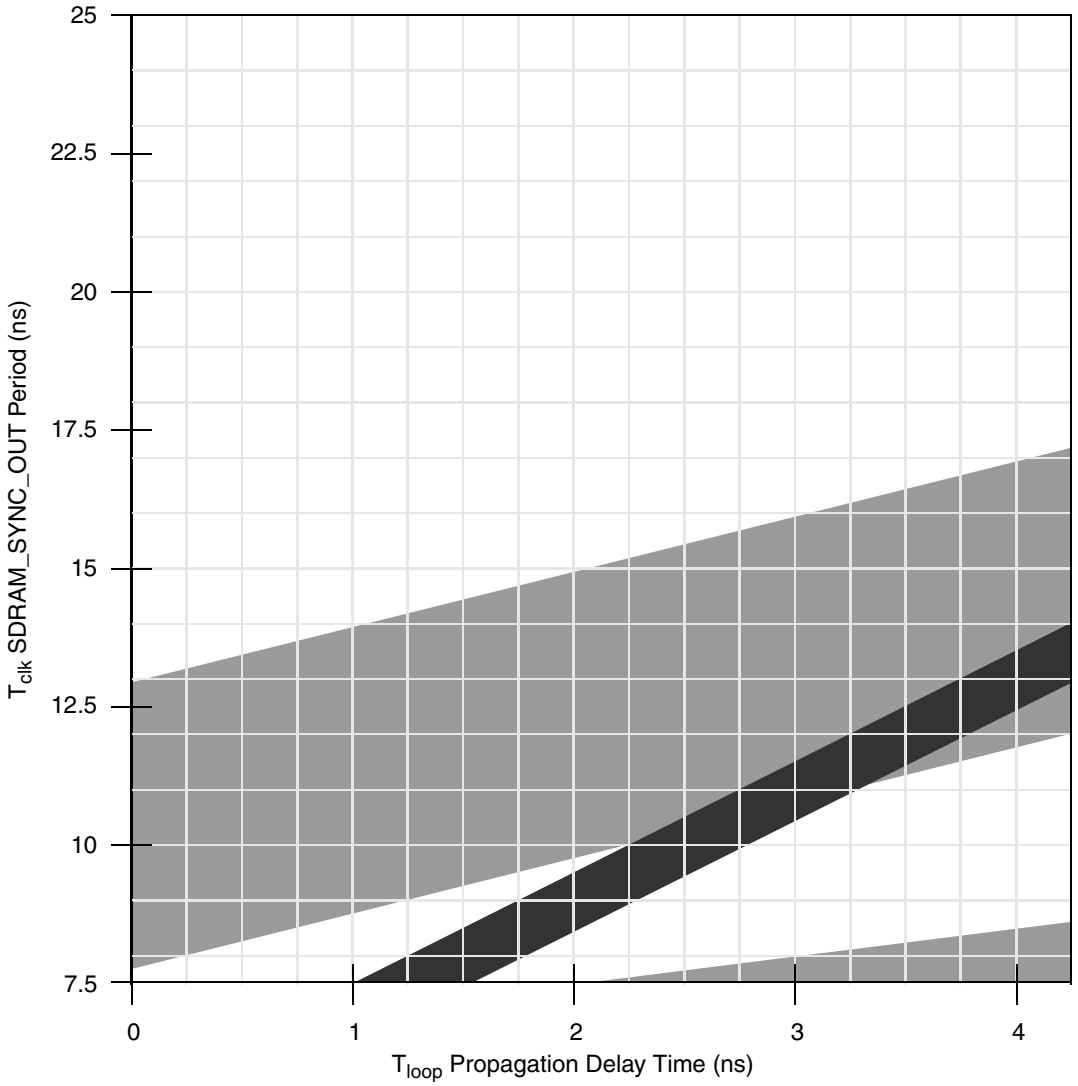

Figure 3. Work Around Chart 3: (DLL_Extend = 0 and Normal Tap Delay) (DLL Locking Range versus Frequency of Operation)

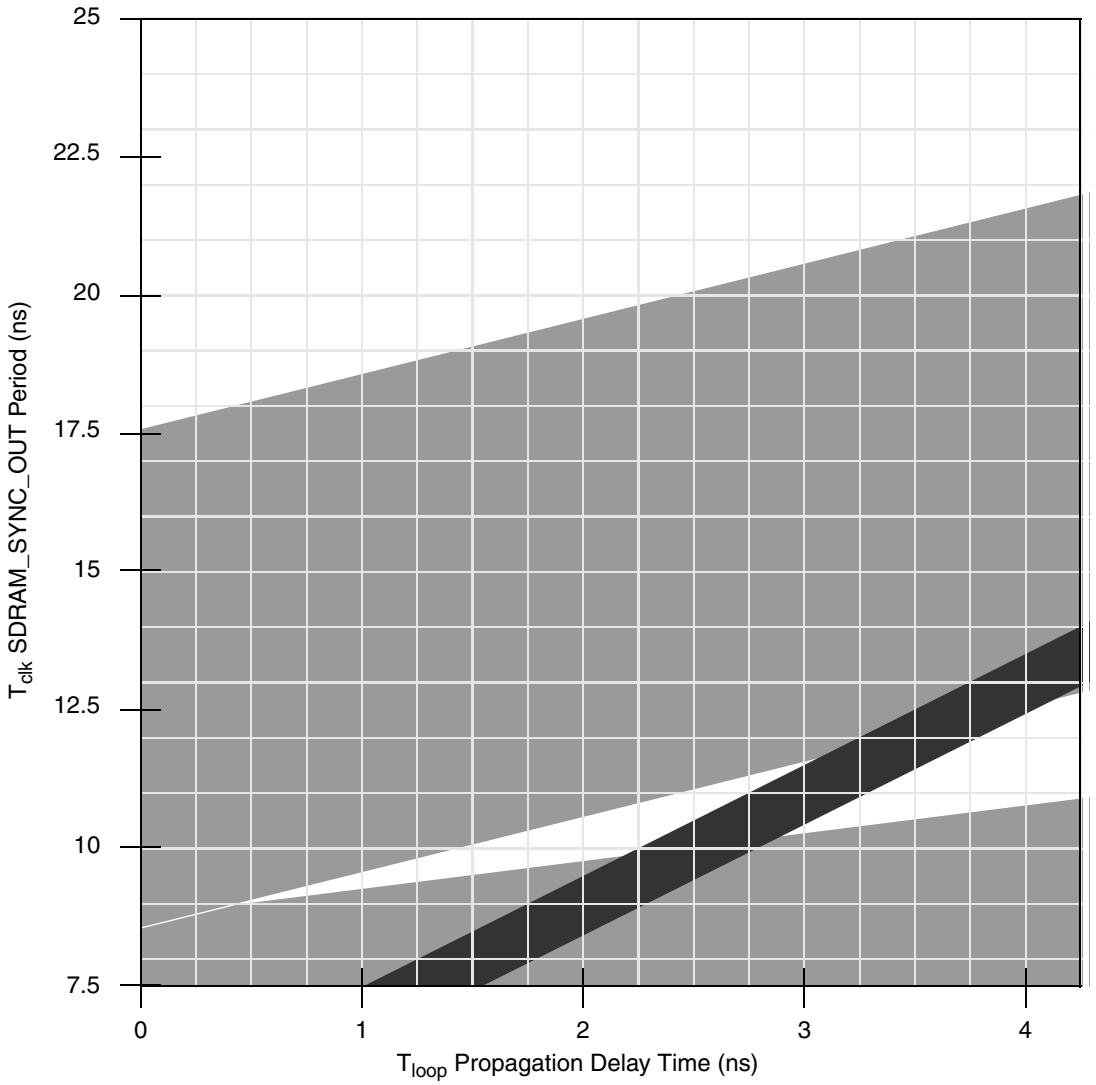

Figure 4. Work Around Chart 4: (DLL_Extend = 0 and Maximum Tap Delay) (DLL Locking Range versus Frequency of Operation)

Error No. 13: SIN1 and $\overline{\text{CTS1}}$ are being driven at reset if DUART mode is enabled

Detailed Description:

When the power up enables DUART mode, configuration signals SDMA0, SIN1, and $\overline{\text{CTS1}}$ that used to be PCI_CLK(1) and PCI_CLK(3), become input to the MPC8245/MPC8241. However, after reset, the MPC8245/MPC8241 drives these two signals.

Projected Impact:

SIN1 and $\overline{\text{CTS1}}$ potentially have bus contention when reset is active.

Work Arounds:

None

Projected Solution:

Fixed in Rev. 1.2 = Rev. B devices.

Error No. 14: UART2 will report break interrupt when software enables the 4-pin DUART mode

Detailed Description:

When the power-up configuration signal SDMA0 enables DUART mode, the MPC8245/MPC8241 is operating at 4-pin UART mode. Later, software can switch this mode to become 4-pin DUART mode. In that case, UART2 always has the break-interrupt status bit set. If the break interrupt is enabled, the CPU core receives the interrupt request.

Projected Impact:

Software will see the spurious break interrupt. The problem with the break interrupt causes a delay in clearing the status bit until valid data is received.

Work Arounds:

For Rev. A devices, software should use the loopback mode to send a valid character to clear the status bit before it can turn on the interrupt enable bit.

Projected Solution:

Fixed in Rev. 1.2 = Rev. B devices.

Error No. 15: SDRAM_SYNC_IN and *sys_logic_clk* phase alignment

Detailed Description:

The MPC8245/MPC8241 has an internal delay in the feedback path for SDRAM_SYNC_IN with respect to the internal *sys_logic_clk* signal. The SDRAM_CLK is not phase-aligned with *sys_logic_clk*.

Projected Impact:

The peripheral logic uses internal *sys_logic_clk* to latch input data and launch output data on the memory interface. The additional internal delay present by *sys_logic_clk* and the DLL causes SDRAM_CLK pins to be offset by the delayed amount.

For pre-existing MPC8240 designs, this delay was not present. Those designs must analyze better timing that the MPC8245/MPC8241 provides, with respect to the MPC8240 memory interface, and determine if that is an issue. This delay affects the output and input hold timing. Because the MPC8245/MPC8241 improved output hold timing equal to the maximum delay, it does not become an issue. The MPC8245/MPC8241 input hold timing is programmable, however, and does not default to a value that accounts for this delay.

Work Arounds:

The SDRAM_SYNC_OUT to SDRAM_SYNC_IN trace is normally laid out to match the SDRAM clock trace length, plus an additional amount for SDRAM output hold.

For new MPC8245/MPC8241 designs, the feedback trace length of SDRAM_SYNC_OUT to SDRAM_SYNC_IN must be shortened by the value T_{os} in comparison to the SDRAM clock trace lengths. T_{os} has a minimum value of 0.65 ns and a maximum value of 1.0 ns. The trace methodology used on the routing of those signals determines the amount of trace length that corresponds to this value.

When upgrading MPC8240 systems to the MPC8245, another solution is to ensure that register 0x77 bits 5:4 are programmed to 0b00 rather than the default value of 0b10.

Projected Solution:

No plan to fix. Implement the work around.

Error No. 16: I^2C general broadcast feature is inoperable

Detailed Description:

The I^2C general broadcast feature on the MPC8245/MPC8241 is inoperable. Broadcast address detections are not passed to the system. In some cases, if the control register's broadcast enable bit is set, the I^2C could hang.

Projected Impact:

The I^2C broadcast feature is not operable because it detects broadcast addresses one I^2C clock late.

Work Arounds:

The only hardware work around requires that the logic adjusts from detecting a general broadcast address one I^2C clock too late.

Projected Solution:

Fixed in Rev. 1.2 = Rev. B devices.

Error No. 17: Data parity errors on 60x bus single beat writes are not detected

Detailed Description:

MCP8245/MPC8241 fails to detect the write-parity errors and does not invoke a machine-check error if all of the following conditions are met:

- A single-beat 60x write is being performed.
- The CPU bus has corrupted data.
- RMW parity mode is turned on (MCCR2[RMW_Par] is set).
- Either ECC or parity modes are enabled.

Projected Impact:

This problem occurs when working in ECC (where RMW must be on), or in normal parity modes, and CPU data gets corrupted in a single-beat 60x write transaction. As a result, for ECC transactions, the corrupted data is used to generate an ECC syndrome. Both are written to SDRAM without detection. For normal parity mode, the corrupted data is written into the SDRAM and the parity byte is recalculated for the entire line (with the corrupted data) and also written into SDRAM.

For CPU-burst write transactions, the problem does not occur for either ECC or parity modes.

Work Arounds:

All CPU-to-local memory writes that require error reporting should be burst writes (cacheable, write-back accesses). This work around cannot be implemented for transactions that involve I/O devices that may not have caching capability.

Projected Solution:

No plans to fix.

Error No. 18: 5-V tolerance issue on MPC8245/MPC8241

Detailed Description:

MCP8245/MPC8241 fails for 5-V input PCI signals when LV_{DD} is set to 5 V and an external PCI device drives a 5-V signal to the MPC8245/MPC8241. If the input high voltage (V_{IH}) is greater than 1.2 V over the I/O buffer supply for PCI and standard (OV_{DD}), bus violations can occur.

Projected Impact:

MPC8245/MPC8241 is not 5-V PCI tolerant.

Work Arounds:

Set LV_{DD} to 3.3 V instead of to 5 V.

This work around draws a much higher current from the 5-V PCI device, and potentially can affect the long-term reliability. Do not use this approach for production purposes.

Projected Solution:

Fixed in Rev. 1.4 = Rev. D devices.

Error No. 19: Driver strength issue for DRV_STD_MEM

Detailed Description:

Bit 6 of the Output Driver Control Register (0x73) does not control the driver capability for standard signals.

The *MPC8245 Integrated Processor User's Manual*, Rev. 1, states that for bit 6 of 0x73 a 40-Ω drive capability is obtained with a bit setting of 1 and a 20-Ω capability when this bit is cleared. However, this bit does not control the driver strength for standard drivers. Instead, bit 5 of 0x73 controls driver strength. The following table shows how driver strength capability for certain memory and I/O signals is determined for this bit:

Setting	Description
1	6-Ω drive capability
0	40-Ω drive capability

Projected Impact:

Using bit 6 to control the standard driver signals capability is not functional; use bit 5 instead. The following signals will have a driver strength of 40 or 6 Ω, depending on the setting of bit 5 of the Output Driver Control Register (0x73):

PMAA[0:2], SDA, SCL, CKO, $\overline{\text{QACK}}$, DA[10:6], $\overline{\text{MCP}}$, MDH[0:31], MDL[0:31], PAR[0:7], and MAA[0:2]

Prior documentation describing the drive strengths for these signals incorrectly stated the impedance as being 20 and 40 ohms. However, it has been verified that the strengths for bit 5 for the above signals are 6 and 40 ohms.

Work Arounds:

Use bit 5 to control the driver capability for standard signals.

Projected Solution:

No plans to fix.

Error No. 20: Type 2 fast back-to-back transactions result in data corruption

Detailed Description:

Type 2 fast back-to-back transactions access multiple targets sequentially. If a PCI master issues a type 2 fast back-to-back transaction to the MPC8245, the transaction causes data corruption for read and write transactions.

Data that is read is corrupted. Write transactions write to incorrect locations or write bad data to a specified location.

Projected Impact:

Type 2 fast back-to-back transactions are not supported on the MPC8245.

Work Arounds:

Software should disable the ability to run fast back-to-back transactions on PCI master devices that can issue fast back-to-back transactions to the MPC8245. Clear bit 9 of the PCI Command Register in the external master device.

Projected Solution:

Bit 7 of the PCI status register (0x06) disabled in Rev. 1.4 = Rev. D. The bit is hardwired to 0, indicating that the MPC8245 (as a target) is not capable of accepting fast back-to-back transactions.

Error No. 21: Enabling the detection of PCI $\overline{\text{SERR}}$ does not work

Detailed Description:

Enabling bit 6 of the Error Enabling Register 2 (0xC4) should report $\overline{\text{SERR}}$ assertions that occur on the PCI bus at any time, regardless of whether the MPC8245 is the initiator, the target, or a non-participating agent. This action does not occur on the MPC8245, and bit 6 of the Error Detection Register 2 (0xC5) does not report any $\overline{\text{SERR}}$ assertions that occur on the PCI bus by an external PCI agent.

Note that the reporting of a $\overline{\text{SERR}}$ assertion when it occurs on the PCI bus two clock cycles after the address phase of transactions where the MPC8245 is the initiator works as expected. Therefore, if bit 7 of the Error Enabling Register 1 (ErrEnR1–0xC0) is set and the case described in the previous sentence occurs, the system error is reported in bit 7 of the Error Detection Register 1 (ErrDR1–0xC1).

Projected Impact:

$\overline{\text{SERR}}$ assertions that occur by an external PCI agent are not reported by the MPC8245.

Work Arounds:

None

Projected Solution:

Fixed in Rev. 1.4 = Rev. D.

Error No. 22: stfd of uninitialized FPR can hang part

Detailed Description:

The 64-bit FPRs each have additional internal bits with which they are associated that specify the type of floating-point number in the register. These bits are set properly whenever the FPR is loaded. However, the part could power up with the internal bits randomly set. The FPR could be interpreted as containing a denormalized number with a mantissa that contains all zeros. If this random state is stored with an **stfd** instruction before a floating-point load operation corrects the internal bits, the part hangs while searching for a leading 1 in the mantissa. The **stfd** instruction is the only instruction that causes this behavior.

Note that this problem was discovered when compiled code stored out FPRs before using them as scratch registers early in the boot sequence.

Projected Impact:

This error affects all systems that use floating-point operations.

Work Arounds:

When emerging from reset, initialize all the FPRs that will be used. The initialization value is not important.

Projected Solution:

Fixed in documentation.

Error No. 23: PCI_SYNC_IN signal is not 5-V tolerant

Detailed Description:

The PCI_SYNC_IN signal of the MPC8245 is powered by $OV_{DD,}$ which is a 3.3-V power source and does not support a 5-V power supply.

Projected Impact:

All parts require that OV_{DD} powers PCI_SYNC_IN. The range of the power supply is 3.0- to 3.6-V.

Work Arounds:

Power sources outside of the 3- to 3.6-V range must be scaled accordingly to be within the 3.0- to 3.6-V range.

Detailed Solution:

No plans to fix.

Error No. 24: MPC8245 does not detect assertion of $\overline{\text{PERR}}$ signal for a certain case

Detailed Description:

The MPC8245 fails to detect the assertion of the $\overline{\text{PERR}}$ signal when all of the following conditions are met:

- The memory-to-PCI clock ratio is 2:1 or higher (for example, 2:1, 3:1, and 4:1).
- The MPC8245 is the initiator of the PCI bus transaction.
- A parity error occurs in the last data transfer of the transaction (for either single- or multiple-beat transactions).

Projected Impact:

The MPC8245 fails to detect the assertion of $\overline{\text{PERR}}$ and does not report the parity error to the core by means of the internal $\overline{mcp}$ signal. Potentially corrupt data may be propagated.

Work Arounds:

Use external logic to monitor the $\overline{\text{PERR}}$ signal and assert the NMI signal when a data parity error is detected on the last data transfer.

Projected Solution:

No plans to fix.

Error No. 25: MPC8245 does not detect assertion of $\overline{\text{MCP}}$ signal for a certain doorbell register case in the messaging unit

Detailed Description:

The MPC8245 fails to detect the assertion of the $\overline{\text{MCP}}$ signal whether in host or agent mode when bit 31 of the Inbound Doorbell Register (IDBR) is set, even if the requirements for an $\overline{\text{MCP}}$ are met.

The requirements for an $\overline{\text{MCP}}$ in this special case include:

HID0[EMCP] = 1, PICR1[MCP_EN] = 1, MSR[ME] = 1

Set bit 31 of the IDBR while the IMIMR[DMCM] = 0.

Even if the conditions described in the paragraph above are met, an $\overline{\text{MCP}}$ does not occur as asserted long enough to be recognized by the processor.

Projected Impact:

The internal $\overline{\text{MCP}}$ signal is asserted for only one clock. Because the processor requires the internal $\overline{\text{MCP}}$ signal to be held asserted for at least two clocks, the MPC8245 fails to detect the assertion of $\overline{\text{MCP}}$ in this case.

Apart from not getting an $\overline{\text{MCP}}$ for the above description, the MPC8245 does not take a machine check exception even if enabled and bits 8, 7, or 4 of the Inbound Message Interrupt Status Register are set (IMISR[8–7, 4] = 1).

Work Arounds:

Use interrupts in the messaging unit to generate an $\overline{\text{MCP}}$.

Projected Solution:

No plans to fix.

Error No. 26: Concurrent writes occur to the UART registers when UAFR[0] is set

Detailed Description:

The UART Alternate Function Register (UAFR) enables software to write concurrently to both UART1 and UART2 registers with the same write operation. However, this action occurs when the UARTs are individually written to, but in addition, concurrent writes to the UART registers occur as a result of writing to other EUMBAR registers, including the error injection registers, when UAFR[0] is set.

Projected Impact:

Concurrent writes occur to the UART registers when UAFR(0) is set and EUMBAR registers are written to.

Work Arounds:

Set UAFR(0) only for concurrent writes to UART registers. Clear this bit immediately afterwards.

Projected Solution:

No plans to fix.

Error No. 27: Posted writes from host to I^2O queue port may result in data corruption with 32-bit SDRAM devices

Detailed Description:

For 32-bit SDRAM configuration, multiple I^2O queue port writes with store gathering may cause the last I^2O write to be lost.

The sequence for processing an inbound message includes the following steps:

1. Remote host writes message #1 data.
2. Remote host writes I^2O inbound queue port for message #1.
3. Remote host writes message #2 data.
4. Remote host writes I^2O inbound queue port for message #2

   The second I^2O queue write (step 4) can be lost because a store merging problem in the PCMWB buffer occurs with the first I^2O queue write (step 2). Store merging happens when the first I^2O queue write in the PCMWB buffer has not been flushed to memory because the memory is busy and cannot respond to the first I^2O queue write yet. The second I^2O queue write occurs and is lost.

The sequence of accepting an outbound message includes the following steps:

1. Remote host receives the interrupt and reads to I^2O outbound queue port.
2. Remote host reads message #1 data.
3. Remote host writes I^2O outbound queue port to move on to next message.
4. Remote host receives the interrupt (optional), and reads the I^2O outbound queue port.
5. Remote host reads message #2 data.
6. Remote host writes I^2O outbound queue port to move on to the next message.

   The store-merging problem for steps 4 and 6 may occur for outbound messages also.

Projected Impact:

This problem is likely to occur during I^2O queue port writes and if configured for 32-bit SDRAM.

Work Arounds:

Motorola recommends that dummy single-beat PCI write cycles are inserted after writing to the I^2O outbound queue port to ensure that the I^2O write is flushed from the PCMWB buffer. Software should reserve a region of memory for the dummy PCI write cycles. Each dummy single-beat PCI write address must be in separate cache lines. Table 4 contains the recommended number of dummy PCI writes according to the number of PCMWB buffers that are used.

The following sections describe recommended procedures for processing outbound and inbound messages.

Outbound Messages

Motorola recommends using the following steps for processing outbound messages:

1. Remote host receives the interrupt, and reads to I^2O outbound queue port.
2. Remote host reads message #1 data.
3. Remote host writes I^2O outbound queue port to move on to next message. Remote host inserts dummy single beat PCI write(s).

4. Remote host receives the interrupt (optional), and reads to I^2O outbound queue port.
5. Remote host reads message #2 data.
6. Remote host writes I^2O outbound queue port to move on to the next message. Remote host inserts dummy single-beat PCI write(s).

Inbound Messages

The modified sequence for processing an inbound message includes the following steps:

1. Remote host writes message #1 data. Remote host may need to insert dummy single beat PCI writes.
2. Remote host writes I^2O inbound queue port for message #1.
3. Remote host writes message #2 data.
4. Remote host writes I^2O inbound queue port for message #2. Remote host may need to insert dummy single beat PCI write(s).

The two main work arounds include either of the following methods:

- Use 64-bit SDRAM. This problem does not appear in the 64-bit environment for either inbound or outbound messages.

or

- Use bits 5-4 of the configuration register at offset 0xE1 (described in Table 5) according to the guidelines in Table 4 and the recommended steps for inbound and outbound messages.

Table 4. Buffers to Frame Size Guidelines

Message Size	Recommended Number of Buffers for Inbound or Outbound Messages	Number of Dummy PCI Writes for Outbound Message	Number of Dummy PCI Writes for Inbound Message
Less than or equal 32 bytes	1 PCMWB	1 single-beat	0 single-beat
	2 PCMWBs	2 single-beats	1 single-beat
	3 PCMWBs	3 single-beats	2 single-beats
	4 PCMWBs	4 single-beats	3 single-beats
33 to 64 bytes	1 PCMWB	1 single-beat	0 single-beat
	2 PCMWBs	2 single-beats	0 single-beat
	3 PCMWBs	3 single-beats	1 single-beat
	4 PCMWBs	4 single-beats	2 single-beats
65 to 96 bytes	1 PCMWB	1 single-beat	0 single-beat
	2 PCMWBs	2 single-beats	0 single-beat
	3 PCMWBs	3 single-beats	0 single-beat
	4 PCMWBs	4 single-beats	1 single-beat
Above 96 bytes	1 PCMWB	1 single-beat	0 single-beat
	2 PCMWBs	2 single-beats	0 single-beat
	3 PCMWBs	3 single-beats	0 single-beat
	4 PCMWBs	4 single-beats	0 single-beat

Table 5. Description of Bits 5-4 of 0xE1

Bits	Description
5-4	00 4 PCMWB available 01 3 PCMWB available 10 2 PCMWB available 11 1 PCMWB available

Projected Solution:

No plans to fix.

Error No. 28: PCI reads from local memory may return stale data

Detailed Description:

When the MPC8245/MPC8241 issues a speculative read to local memory on behalf of the current PCI/DMA read access, if the speculative read hits the PCI-to-system-memory-write buffer (PCMWB) or the copyback buffer, the MPC8245/MPC8241 will read from the local memory before flushing the write data from the PCMWB or copyback buffer. In this case, MPC8245/MPC8241 will return stale data to the PCI master or the DMA controller. Note that speculative reads occur when PCI read or PCI read line occurs with PICR1[2] = 1, or when a PCI read multiple is issued.

Projected Impact:

The MPC8245/MPC8241 may return stale data to PCI/DMA read access and results in data corruption.

Work Arounds:

1. Set bit 0 of the AMBOR configuration register at 0xE0.

Table 6. Bit Description of Bit 0 of 0xE0

Bit 0	Description
0 (default)	E0[0]—PCMWB optimization—1 0 CCU can start the speculative read (or prefetch) of the next cache line (for PCI read streaming purposes), even if any PCMWBs have valid data 1 CCU will not start the speculative read (or prefetch) of the next cache line (for PCI read streaming purposes), until all PCMWBs are invalidated

Setting bit 0 of the AMBOR configuration register (0xE0) forces the read prefetch to wait until all PCMWB buffers are invalid (regardless, the read is hit or not) before it will fetch the read data from memory.

or

2. Set bit 0 of the PICR2 configuration register at 0xAC.

Table 7. Bit Description of Bit 0 of 0xAC

Bit 0	Description
0 (default)	AC[0]—Copyback optimization 0 CCU can start the speculative read (or prefetch) of the next cache line (for PCI read streaming purposes), even if the copyback buffer has valid data 1 CCU will not start the speculative read (or prefetch) of the next cache line (for PCI read streaming purposes), until the copyback buffer is invalidated

Setting bit 0 of PICR2 (0xAC) forces the read prefetch to wait until the copyback buffer is invalid (regardless, the read is hit or not), before it will fetch the read data from memory.

Projected Solution:

No plans to fix.

Error No. 29: PCI writes may not invalidate speculative read data

Detailed Description:

This error may occur when an incoming PCI read has triggered a prefetch for the next cache line. This can occur when either a device issues a PCI read or a PCI memory-read-line command when PICR1[2] = 1 or issues a PCI memory-read-multiple command. Typically, if the current PCI read does not need the data from the speculative prefetch, the speculative read transaction can be terminated while the internal logic is still trying to fetch the data from memory and put it into the internal PCMRB buffer. During this period, an incoming PCI write that hits to the same prefetching cache line will be accepted and, normally, the prefetched data will be tracked and invalidated after the physical read operation from memory is complete. However, due to the existence of this error, if another incoming PCI read hits to this cache line while the prefetch is still in progress, the MPC8245/MPC8241 will return the stale data (from the prefetched read operation, which occurs before the write data is flushed).

Note that usually, the duration of the prefetched read should not take long, when compared with the PCI write followed by another PCI read transaction. Thus, the error will not be observed. However, in certain cases, the prefetched read may be delayed due to the memory bus being busy (an extreme case is a CPU performing a burst read from an 8-bit ROM). This can significantly increase the likelihood of seeing the problem.

Alternatively, the exposure to the error is greater if the PCI write and PCI read accesses are happening very fast and are short accesses, such as in memory testing algorithms where single-beat writes are followed by single-beat reads and then compared.

Projected Impact:

Potentially stale data may be returned for PCI reads.

Work Arounds:

Currently there are two work arounds:

1. Insert a dummy PCI read from an unrelated memory location between the PCI write and the desired PCI read sequence. Using this work around, the prefetched read that is triggered before the PCI write will be performed before the dummy PCI read and it will be discarded immediately. By the time the real PCI read comes in, it will wait until the previous write has been flushed before the latest data will be returned to the PCI bus. Note that the PCI write may be used to qualify the insertion of the dummy read to minimize the need for too many dummy reads.
2. Turn off speculative reading by clearing PICR1[2] and prevent external PCI masters from issuing memory-read-multiple commands.

Projected Solution:

No plans to fix.

How to Reach Us:

Home Page:
www.freescale.com

email:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com

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)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: MPC8245CE
Rev. 10
08/2005

freescale™
semiconductor