Chip Errata for the MPC885 Duet Family

This document details all known silicon errata for the MPC885, MPC880, MPC875, and MPC870 Duet family. The MPC885 has the same functionality as the MPC880, MPC875, and MPC870, and any differences are detailed in the reference manual. Table 1 provides a revision history for this device errata document.

Table 1. Document Revision History

Revision Number	Date	Substantive Change(s)
6	04/2009	Updated CPU15
5	02/2009	Updated CPU15. Added GLL2.
4	12/2007	Updated G14.
3	11/2004	Added G14.
2	08/02/2004	Put errata in new Freescale template. Changed the Mask number.
1	06/01/2004	Added CPM16, removed CPU10.
0	01/13/2004	Initial release of document

freescale™
semiconductor

Table 2 summarizes all known errata and lists the corresponding silicon revision level to which it applies. For the "Errata in Silicon," "Y" entry indicates the erratum applies to a particular revision level, while an "N" entry means it does not apply.

Table 2. MPC885/MPC875 Summary of Silicon Errata and Applicable Revision

No.	Name	Projected Impact Overview	Work Around Exists	Errata in Silicon	
				MPC875 Rev 0 Mask Set 0K51M	MPC885 Rev 0 Mask Set 0K51M
Global Errata					
GLL1	Some registers are not initialized correctly during power-up RESET, HRESET, and SRESET	Some registers may work differently than expected due to incorrect initialization.	N	Y	Y
GLL2	If HRESET assertion is detected at the positive edge of EXTCLK, then EXTCLK and CLKOUT will be out of phase.	EXTCLK and CLKOUT will be out of phase.	N	Y	Y
SIU Errata					
SIU4	Spurious External Bus Transaction following PLPRCR Write	This behavior will only affect systems with bus slaves that implement synchronous state machines that are sensitive to bus protocol violations.	Y	Y	Y
SIU5	Bus Monitor Failure After TEA Assertion	If an access after $\overline{\text{TEA}}$ is to an unmapped address, the access will not be terminated by the MPC885/MPC875. If this is allowed to occur, this cycle could potentially be endless, causing the system to hang.	Y	Y	Y
SIU9	CPU Receives Machine Check After Writing to the PLPRCR	The problem is most often exhibited when entering and exiting low-power modes frequently, such as when using doze mode to conserve power. The probability of encountering this problem is small but finite (approximately 1 in a million).	Y	Y	Y
SIU10	PIT does not Count Properly when the Pre-divider is Four	The periodic interrupt timer (PIT) will not count properly and the generation of interrupts from the PIT may be affected as well.	Y	Y	Y
CPM Errata					
CPM12	RCCR Must Be Written as a Word	RMDS[ERAM4K] (located at address IMMR+0x9C7) will be cleared.	Y	Y	Y
CPM14	Erratic Behavior Can Occur if Extra Clock Pulse is Detected on SCL	The internal counter may not get reset correctly. This could generate partial frames (less than 8 bits) in the next transaction.	Y	Y	Y

Chip Errata for the MPC885 Duet Family

No.	Name	Projected Impact Overview	Work Around Exists	Errata in Silicon	
				MPC875 Rev 0 Mask Set 0K51M	MPC885 Rev 0 Mask Set 0K51M
CPM15	Invalid CRC Generation/Checking in SCCs When Using SOF Generation	Although the data is received correctly, the CRC result will be corrupted.	Y	Y	Y
CPM16	Internal timer connection for SOF generation is not functional	Generation of SOF packets will not work if the internal connection between TOUT1 and DREQ0 is used.	Y	Y	Y
General Errata					
G9	Conflict Between Data Show Cycles and SDMA Burst Writes	The observed phenomenon is that a burst write with four operands will hold the second operand into the third and fourth burst beats.	Y	Y	Y
G11	BSDL—Pin B6 is Output Pin only	None.	N	Y	Y
G14	Possible I/O glitches	This erratum applies to all pins including input-only pins such as PORESET. As such, when the glitch occurs there is a possibility for contention.	Y	Y	Y
CPU Errata					
CPU5	Instruction MMU Bug at Page Boundaries in Show-all Mode	The address driven by the core will be of the previous page and not the current one.	Y	Y	Y
CPU11	Wrong Data Breakpoint Detection using Address Range Function	Erroneous detection can occur.	Y	Y	Y
CPU12	Non-Maskable Interrupts are Unrecoverable	The contents of the MSR is unreliable upon entry into this "second" exception routine.	Y	Y	Y
CPU13	First instruction in Decrementer or External Interrupt handler may be skipped if Instruction fetch show cycles are enabled in ICTRL[IST_SER]	The instructions of the first cache line may be fetched in the wrong order.	Y	Y	Y
CPU14	RCPU: do not execute 'o' form integer instructions before '.' form integer multiply/divide complete	The condition register bit #3 (SO) may be wrongly updated by XER[SO] bit changed by the "o" form instruction.	Y	Y	Y
CPU15	Incorrect code execution after branch on MMU page boundary	An incorrect or invalid instruction can be executed.	Y	Y	Y
ESAR Errata					

Chip Errata for the MPC885 Duet Family

Table 2. MPC885/MPC875 Summary of Silicon Errata and Applicable Revision (continued)

No.	Name	Projected Impact Overview	Work Around Exists	Errata in Silicon	
				MPC875 Rev 0 Mask Set 0K51M	MPC885 Rev 0 Mask Set 0K51M
ESAR6	I2C / SPI Relocation Feature	If the 16-bit locations IMMR+0x3cac or IMMR+0x3dac are overwritten, I2C/SPI will cease to function.	N	Y	Y
ESAR7	(UTOPIA-Serial ATM / PTP switching ucode): Overlapping usage of the parameter BDPOINT in serial ATM by both Rx and Tx may cause errors. (MPC885 Only)	This errata can trash memory content or also freeze the CPM due to bus errors. UTOPIA is not affected.	Y	N	Y
ESAR8	(UTOPIA-Serial PTP switching ucode) Port-To-Port to a serial ATM SCC with a PTP TCT in external memory can lead to trashed BD rings and stopped PTP traffic on that channel. (MPC885 Only)	Port-To-Port to a serial ATM SCC with a PTP TCT in external memory can lead to trashed BD rings and stopped PTP traffic on that channel. UTOPIA is not affected.	Y	N	Y
ATM Errata					
ATM12	APC Overruns cause potential loss of paced Bandwidth. (MPC885 Only)	APCT_PTR is not updated correctly, leading to a potential loss in scheduled bandwidth.	Y	N	Y
ATM14	Statistical counters do not count properly in NIM mode and when MCF filtering SRSTATE[MCF]=1 is used for an RCT. (MPC885 Only)	The statistical counters count each OAM cell twice so the statistical information is not accurate.	Y	N	Y
FEC Errata					
FEC12	CRC Error Reported if MII_RX_ER Asserts While in Internal Loopback Mode	The receive logic currently mistakenly reports a CRC error if MII_RX_ER asserts when a frame is being looped internally.	Y	Y	Y
FEC14	7-wire Interface Compatibility Problem	A glitch may occur on the internal clock line which may cause unpredictable results.	Y	Y	Y

Chip Errata for the MPC885 Duet Family

Errata Number: GLL1

Description:

Some registers are not initialized correctly during Power-Up RESET, $\overline{\text{HRESET}}$, and $\overline{\text{SRESET}}$.The following table is provided to clarify/correct the power-on RESET value of many of the registers and lists whether each register is affected by $\overline{\text{HRESET}}$ and/or $\overline{\text{SRESET}}$. The table applies for the MPC885 and MPC875.

Table 3. Power-On Reset Value of the MPC885 Registers

REGISTER	Value at Power-On RESET [1]	Affected by $\overline{\text{HRESET}}$	Affected by $\overline{\text{SRESET}}$
SIUMCR	01200000	YES	NO
SYPCR	FFFFFF07	YES	NO
SWSR	0	YES	YES
SIPEND	0000xxxx	YES	YES
SIMASK	0000xxxx	YES	YES
SIEL	0000xxxx	YES	NO
SIVEC	(xx11)(11xx)xxxxxx	YES	YES
TESR	XXXX0000	YES	YES
SDCR	0	YES	NO
PBR0	x	NO	NO
POR0	x	NO	NO
PBR1	x	NO	NO
POR1	x	NO	NO
PBR2	x	NO	NO
POR2	x	NO	NO
PBR3	x	NO	NO
POR3	x	NO	NO
PBR4	x	NO	NO
POR4	x	NO	NO
PBR5	x	NO	NO
POR5	x	NO	NO
PBR6	x	NO	NO
POR6	x	NO	NO
PBR7	x	NO	NO
POR7	x	NO	NO
PGCRA	0	YES	NO

Chip Errata for the MPC885 Duet Family

Table 3. Power-On Reset Value of the MPC885 Registers (continued)

REGISTER	Value at Power-On RESET [1]	Affected by HRESET	Affected by SRESET
PGCRB	0	YES	NO
PSCR	x	NO	NO
PIPR	??00??00	YES	YES
PER	0	YES	YES
BR0	XXXXX(??00)0(000?)	YES	NO
OR0	00000FF4	YES	NO
BR1	XXXXXX(xx00)0	YES	NO
OR1	XXXXXXX(xxx0)	YES	NO
BR2	XXXXXX(xx00)0	YES	NO
OR2	XXXXXXX(xxx0)	YES	NO
BR3	XXXXXX(xx00)0	YES	NO
OR3	XXXXXXX(xxx0)	YES	NO
BR4	XXXXXX(xx00)0	YES	NO
OR4	XXXXXXX(xxx0)	YES	NO
BR5	XXXXXX(xx00)0	YES	NO
OR5	XXXXXXX(xxx0)	YES	NO
BR6	XXXXXX(xx00)0	YES	NO
OR6	XXXXXXX(xxx0)	YES	NO
BR7	XXXXXX(xx00)0	YES	NO
OR7	XXXXXXX(xxx0)	YES	NO
MAR	x	NO	NO
MCR	(xx00)0(x000)0(xxx0)X(00xx)X	YES	NO
MAMR	xx001000	YES	NO
MBMR	xx001000	YES	NO
MSTAT	0	YES	NO
MPTPR	0200	YES	NO
MDR	x	NO	NO
TBSCR	0	YES	NO
TBREFA	x	NO	NO
TBREFB	x	NO	NO
RTCSC	00(000x)(000x)	YES	YES
RTC	x	NO	YES
RTSEC	x	NO	YES

Chip Errata for the MPC885 Duet Family

Table 3. Power-On Reset Value of the MPC885 Registers (continued)

REGISTER	Value at Power-On RESET [1]	Affected by $\overline{\text{HRESET}}$	Affected by $\overline{\text{SRESET}}$
RTCAL	x	NO	NO
PISCR	0	YES	NO
PITC	x	NO	NO
PITR	x	N/A	N/A
SCCR	0(000?)(?000)(0??0)0000	YES	NO
PLPRCR	???0(0100)000	YES	YES
RSR	0	YES	YES
TBSCRK	x	YES	YES
TBREFAK	x	YES	YES
TBREFBK	x	YES	YES
TBK	x	YES	YES
RTCSCK	x	YES	YES
RTCK	x	YES	YES
RTSECK	x	YES	YES
RTCALK	x	YES	YES
PISCRK	x	YES	YES
PITCK	x	YES	YES
SCCRK	x	YES	YES
PLPRCRK	x	YES	YES
RSRK	x	YES	YES
I2MOD	0	YES	YES
I2ADD	x	NO	NO
I2BRG	FFFF	YES	NO
I2COM	0	YES	YES
I2CER	0	YES	YES
I2CMR	0	YES	YES
SDAR	x	NO	NO
SDSR	0	YES	YES
SDMR	0	YES	YES
IDSR1	0	YES	YES
IDMR1	0	YES	YES
IDSR2	0	YES	YES
IDMR2	0	YES	YES

Chip Errata for the MPC885 Duet Family

Table 3. Power-On Reset Value of the MPC885 Registers (continued)

REGISTER	Value at Power-On RESET [1]	Affected by $\overline{\text{HRESET}}$	Affected by $\overline{\text{SRESET}}$
CIVR	0	YES	YES
CICR	0	YES	NO
CIPR	0	YES	YES
CIMR	0	YES	YES
CISR	0	YES	YES
PADIR	0	YES	NO
PAPAR	0	YES	NO
PAODR	0	YES	NO
PADAT	x	NO	NO
PCDIR	0	YES	NO
PCPAR	0	YES	NO
PCSO	0	YES	NO
PCDAT	x	NO	NO
PCINT	0	YES	NO
PDDIR	0	YES	NO
PDPAR	0	YES	NO
PDDAT	x	NO	NO
TGCR	0	YES	YES
TMR1	0	YES	YES
TMR2	0	YES	YES
TRR1	FFFF	YES	YES
TRR2	FFFF	YES	YES
TCR1	0	YES	YES
TCR2	0	YES	YES
TCN1	0	YES	YES
TCN2	0	YES	YES
TMR3	0	YES	YES
TMR4	0	YES	YES
TRR3	FFFF	YES	YES
TRR4	FFFF	YES	YES
TCR3	0	YES	YES
TCR4	0	YES	YES
TCN3	0	YES	YES

Chip Errata for the MPC885 Duet Family

Table 3. Power-On Reset Value of the MPC885 Registers (continued)

REGISTER	Value at Power-On RESET [1]	Affected by $\overline{\text{HRESET}}$	Affected by $\overline{\text{SRESET}}$
TCN4	0	YES	YES
TER1	0	YES	YES
TER2	0	YES	YES
TER3	0	YES	YES
TER4	0	YES	YES
CPCR	0	YES	YES
RCCR	0	YES	NO
RCTR1	NA	YES	YES
RCTR2	NA	YES	YES
RCTR3	NA	YES	YES
RCTR4	NA	YES	YES
RTER	0	YES	YES
RTMR	0	YES	YES
BRGC1	0	YES	NO
BRGC2	0	YES	NO
BRGC3	0	YES	NO
BRGC4	0	YES	NO
GSMR_L1	0	YES	YES
GSMR_H1	0	YES	YES
PSMR1	0	YES	YES
TODR1	0	YES	YES
DSR1	7E7E	YES	YES
SCCE1	0	YES	YES
SCCM1	0	YES	YES
SCCS1	0	YES	YES
GSMR_L2	0	YES	YES
GSMR_H2	0	YES	YES
PSMR2	0	YES	YES
TODR2	0	YES	YES
DSR2	7E7E	YES	YES
SCCE2	0	YES	YES
SCCM2	0	YES	YES
SCCS2	0	YES	YES

Chip Errata for the MPC885 Duet Family

Table 3. Power-On Reset Value of the MPC885 Registers (continued)

REGISTER	Value at Power-On RESET [1]	Affected by HRESET	Affected by SRESET
GSMR_L3	0	YES	YES
GSMR_H3	0	YES	YES
PSMR3	0	YES	YES
TODR3	0	YES	YES
DSR3	7E7E	YES	YES
SCCE3	0	YES	YES
SCCM3	0	YES	YES
SCCS3	0	YES	YES
GSMR_L4	0	YES	YES
GSMR_H4	0	YES	YES
PSMR4	0	YES	YES
TODR4	0	YES	YES
DSR4	7E7E	YES	YES
SCCE4	0	YES	YES
SCCM4	0	YES	YES
SCCS4	0	YES	YES
SMCMR1	0	YES	YES
SMCE1	0	YES	YES
SMCM1	0	YES	YES
SMCMR2	0	YES	YES
SMCE2	0	YES	YES
SMCM2	0	YES	YES
SPMODE	0	YES	YES
SPIE	0	YES	YES
SPIM	0	YES	YES
SPCOM	0	YES	YES
PIPC	0	YES	NO
PTPR	0	YES	NO
PBDIR	xxx(xx00)0000	YES	NO
PBPAR	xxx(xx00)0000	YES	NO
PBODR	0	YES	NO
PBDAT	x	YES	YES
SIMODE	0	YES	YES

Chip Errata for the MPC885 Duet Family

Table 3. Power-On Reset Value of the MPC885 Registers (continued)

REGISTER	Value at Power-On RESET [1]	Affected by $\overline{\text{HRESET}}$	Affected by $\overline{\text{SRESET}}$
SIGMR	0	YES	NO
SISTR	0	YES	NO
SICMR	0	YES	YES
SICR	0	YES	NO
SIRP	0	YES	YES

[1]

x or X = "don't care" in either bits, nibbles, or the entire register.

0 = a single zero indicates the entire register is reset to zeros.

( ) = isolates bits of a nibble of the register.

? = a don't care for POR, but if this register is affected by $\overline{\text{HRESET}}$ or $\overline{\text{SRESET}}$, indicates that the value will remain the same as what it was before the reset occurred.

NA = Not Applicable, indicates that this register has no POR value.

Projected Impact:

Some registers may work differently than expected due to incorrect initialization.

Work Around:

None.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: GLL2

Description:

In 1:2:1 mode, when HRESET is detected at the positive edge of EXTCLK, then there will be a loss of phase between EXTCLK and CLKOUT.

Projected Impact:

EXTCLK and CLKOUT will be out of phase.

Work Around:

EXTCLK and CLKOUT will come back in phase if either of the following programming sequences are followed:

- Reprogramming the PLPRCR:
  Program the PLPRCR such that the PLL clock will change, then reprogram the PLPRCR value back to the desired value.

    OR

- Reprogram the SCCR:
    1. Write 1'b00 to SCCR[EBDF].
    2. Write 1'b01 to SCCR[EBDF].
    3. Rewrite the desired value to the PLPRCR register.

Disposition:

Will not be fixed.

Errata Number: SIU4

Description:

Spurious external bus transactions can occur after executing a store to the PLPRCR register which changes the PLL multiplication factor (MF bits). This store causes the PLL to freeze the clocks while another external bus access is already visible on the pins of the chip. This appears externally as a transaction which begins, has its clocks frozen, and then is abruptly aborted without following the bus protocol.

This behavior will only affect systems with bus slaves that implement synchronous state machines that are sensitive to bus protocol violations. Synchronous DRAMs are not affected, and synchronous bus slaves that ignore bus signals when not selected (e.g. Tundra QSPAN) are not affected.

The only cases in which this erratum will cause problems are if:

The device is executing code from a slave which implements a state machine dependent on the PowerPC™ bus protocol, where that state machine might "get lost".

There is an external device which snoops the PowerPC bus and implements a state machine; this state machine might "get lost".

Projected Impact:

This behavior will only affect systems with bus slaves that implement synchronous state machines that are sensitive to bus protocol violations.

Work Around:

The behavior exhibited in this erratum is a secondary symptom of the behavior of SIU9. See the description of SIU9 for the suggested work around. The work around described there will also avoid the above-described spurious external bus transaction.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: SIU5

Description:

The bus monitor does not activate for the access immediately following an access which terminated with a $\overline{\text{TEA}}$ assertion (either internally or externally generated). Therefore, if the following access is to an unmapped address, the access will not be terminated by the MPC885/MPC875. If this is allowed to occur, this cycle could potentially be endless, causing the system to hang.

Projected Impact:

If an access after $\overline{\text{TEA}}$ is to an unmapped address, the access will not be terminated by the MPC885/MPC875. If this is allowed to occur, this cycle could potentially be endless, causing the system to hang.

Work Around:

- Avoid this situation (which should not occur in a normal system).
- Terminate the cycle externally (that is, implement an external bus monitor).
- Rely on the software watchdog timer to reset the system if this error occurs.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: SIU9

Description:

The CPU may receive a machine check after writing to the PLPRCR. This error is caused by an extra clock generated by the clock block after the SIU releases the bus.

The error occurs when the CPU begins a transaction after the SIU releases the internal bus. Due to the PLPRCR write, the CPU's clocks are stopped. However, the extra clock allows the CPU's transaction to begin before the clocks stop. Therefore, the CPU's clocks are stopped in mid-transaction, and its transaction never receives an acknowledgement. When the CPU resumes operation, it receives a machine check.

The failure mechanism is due to an internal logic synchronization issue aggravated by memory refreshes performed by the UPM. The problem is most often exhibited when entering and exiting low-power modes frequently, such as when using doze mode to conserve power. The probability of encountering this problem is small but finite (approximately 1 in a million).

Projected Impact:

The problem is most often exhibited when entering and exiting low-power modes frequently, such as when using doze mode to conserve power. The probability of encountering this problem is small but finite (approximately 1 in a million).

Work Around:

Prevent the CPU from getting the bus during the extra clock following the PLPRCR write. To do so, you must enable the instruction cache and create a software delay. To calculate how long of a delay is necessary, take the longest bus transaction in CPU clocks (including memory refresh or PCMCIA access). The resultant number of clocks must be executed using instructions such as NOP (1 clock), ISYNC (2 clocks) or DIVW (13 clocks). The instructions for the delay must either fit in one cache line, or some other mechanism must be used to make certain that the instructions for the delay are in cache. The instructions of the delay must be executed after the PLPRCR write, which can be enforced by the ISYNC instruction. The length of the software delay must also be greater than or equal to the length of the longest bus transaction (in CPU clocks).

For example, if your longest transaction is 8 bus clocks and the CPU is in 1:2 mode, the delay required is 16 clocks. This can be accomplished with two DIVW instructions. The instruction cache must be enabled to make certain that the code sequence is loaded into the cache as a unit, and the PLPRCR write should be aligned to a 16-byte boundary to make sure it is the first instruction in the cache line containing the sequence.

The following work around software is thus suggested:

.align 16

.global SetPLPRCR

SetPLPRCR:

stw 4, PLPRCR_OFFSET(3) # burst aligned address

isync   # isync

addi 3, 0, 1 # safeguard against divide-by-zero

divw 4, 4, 3

divw 4, 4, 3

blr

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: SIU10

Description:

The periodic interrupt timer (PIT) will not count properly when the predivider in SCCR[PTDIV] is four. This may also affect the generation of interrupts from the PIT.

Projected Impact:

The periodic interrupt timer (PIT) will not count properly and the generation of interrupts from the PIT may be affected as well.

Work Around:

Use a 512 pre-divider setting, SCCR[PTDIV] = 1.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPM12

RCCR Must Be Written as a Word

Description:

If the RCCR is written as a byte or half word (to addresses IMMR+0x9C4, IMMR+0x9C5), then RMDS[ERAM4K] (located at address IMMR+0x9C7) will be cleared.

Projected Impact:

RMDS[ERAM4K] (located at address IMMR+0x9C7) will be cleared.

Work Around:

Change the RCCR bytes at address IMMR+(0x9c4/0x9c5), write the whole word.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPM14

Description:

The I^{2}C controller has an internal counter that counts the number of bits sent. This counter is reset when the I^{2}C controller detects a START condition. When an extra SCL clock pulse is inserted in between transactions (before START and after STOP conditions), the internal counter may not get reset correctly. This could generate partial frames (less than 8 bits) in the next transaction.

Projected Impact:

The internal counter may not get reset correctly. This could generate partial frames (less than 8 bits) in the next transaction.

Work Around:

Do not generate SCL pulses in between transactions. In a noisy environment, the digital filter I2MOD[FLT] and additional filtering capacitors should be used on SCL to eliminate clock spikes that may be misinterpreted as clock pulses.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPM15

Description:

When the USB controller is configured in Host mode, and the SOF generation (SFTE=1 in USMOD register) is being used, there may be false CRC error indication in other SCCs. Although the data is received correctly, the CRC result will be corrupted.

Projected Impact:

Although the data is received correctly, the CRC result will be corrupted.

Work Around:

Use available SOF microcode package to resolve this problem.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPM16

Description:

Setting USBMOD[SFTE] to 1 should connect TOUT1 to DREQ0 internally, however this internal connection does not work properly.

Projected Impact:

Generation of SOF packets will not work if the internal connection between TOUT1 and DREQ0 is used (USBMOD[SFTE] = 1).

Work Around:

Use the external connection (USBMOD[SFTE] = 0) providing a 1 KHz signal to the DREQ0 pin (PC15) and configuring the parallel port I/O accordingly. The 1KHz signal can be generated by a CPM general purpose timer or BRG.

Projected Solution:

This erratum will be fixed in the next revision of silicon.

Errata Number: G9

Description:

IF:

data show cycles are enabled via SIUMCR[DSHW]

AND:

an internal register or dual-port RAM access is made immediately following an SDMA burst write,

THEN:

the SDMA burst write may be corrupted.

The observed phenomenon is that a burst write with four operands will hold the second operand into the third and fourth burst beats. For example, a burst write of A-B-C-D will be observed on the bus as A-B-B-B.

Note: This behavior can also occur when the SDMA burst is to burst-inhibited memory. Setting the memory to burst-inhibited will not solve the problem.

Projected Impact:

The observed phenomenon is that a burst write with four operands will hold the second operand into the third and fourth burst beats.

Work Around:

Do not use data show cycles in a system that performs SDMA bursts. These include systems that use ATM, Fast Ethernet, and memory-to-memory IDMA.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: G11

BSDL—Pin B6 is Output Pin only

Description:

This pin can only be used in output mode during JTAG scan testing.

Projected Impact:

None.

Work Around:

None.

Projected Solution:

This errata is scheduled to be corrected in a future version.

Errata Number: G14

Possible I/O glitches

Description:

If core supply voltage (VDDL) is below recommended operating conditions while I/O supply voltage (VDDH) is high during a power-on reset sequence, I/O pins (including those designated input only) might drive a value instead of being Hi-Z. This could confuse connected devices and, in turn, cause the PowerQUICC device to behave improperly. Affected pins include such interfaces as the communication parallel I/O, reset, address and data bus pins, and JTAG pins.

The value a pin may drive is random. As soon as core voltage has stabilized at its nominal level within recommended operating conditions, all pins will behave normally and the PowerQUICC will continue to function properly.

Work Around:

It is recommended that VDDL/VDDSYN be raised before or simultaneously with VDDH during the power-on reset sequence; that is, while VDDH <= recommended operating condition for VDDL/VDDSYN during ramp of all voltages, it should be ensured that VDDL/VDDSYN >= VDDH at all times.

Refer to Figure 1.
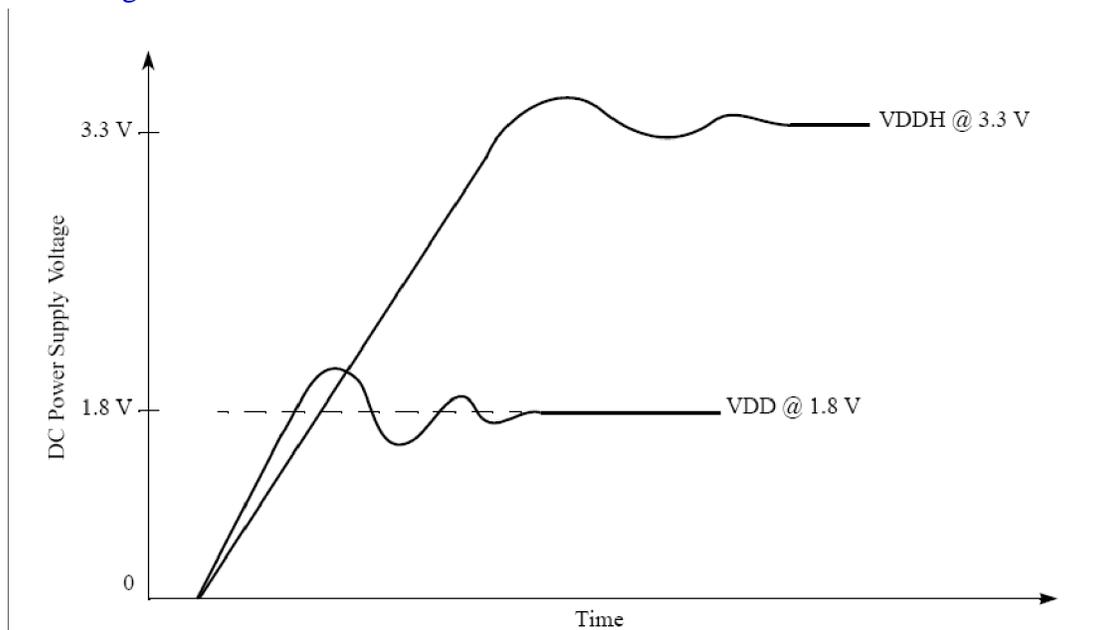
Figure 1. Power Supply Ramping

If the power supply design cannot be altered as recommended, then any logic connected to the PowerQUICC which may react or be affected in an undesirable manner by unexpected low or high signal values during power ramp up should be appropriately secured in the design. Using SDRAM as an example, it is usually sufficient to connect the CKE input to PORESET. The practice of tying CKE high may not be safe and may lead to an unrecoverable system state.

Errata Number: CPU5

Description:

The wrong instruction address is driven by the core when all the following conditions occur:

- MPC885/MPC875 works in 'show all' mode (for example, ISCT_SER bits=000 in ICTRL).
- Sequential instruction crosses IMMU page boundary.
- Instruction cache fails to get ownership of the internal U-bus on the first clock.

In this case the address driven by the core will be of the previous page and not the current one.

Projected Impact:

The address driven by the core will be of the previous page and not the current one.

Work Around:

Possible work arounds include:

- Disable show all mode
- Invalidate the page next to current (by using the tlbie instruction) when performing the TLB reload operation.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPU11

Description:

When the data breakpoint/watchpoint logic is used to trap on load/store accesses to addresses less than or greater than a given address, erroneous detection can occur. This error occurs for byte-size load/stores where the address accessed and the compare address (in comparator E or F) differ only in the two least-significant bits and the two least-significant bits of the address accessed are not 00.

Projected Impact:

Erroneous detection can occur.

Work Around:

Select the address in the compare register (E or F) such that it will differ by more than the two least-significant bits from the address range to be detected. For example, if the address range that you wish to detect are addresses less than or equal to 0xA0000006, then use address 0xA0000008 instead.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPU12

Description:

When an interrupt occurs during an RFI instruction, the RFI partially executes before the exception is taken. The "partial execution" includes copying the contents of SRR1 to the MSR. Therefore, the MSR value seen in the exception handler is the MSR value of the previous machine state, not of the machine state when the RFI occurred. The contents of the MSR is therefore unreliable upon entry into this "second" exception routine.

Projected Impact:

The contents of the MSR is unreliable upon entry into this "second" exception routine.

Work Around:

This erratum only affects exceptions which can occur during an RFI instruction. For a normal application, the only exceptions which can occur during an RFI instruction are non-maskable interrupts or breakpoints. [External interrupts will be masked during the machine-state recovery portion of an exception handler.] Therefore, work arounds for only these two cases are required:

Do not put an instruction breakpoint on an RFI instruction.

Treat all non-maskable interrupts as non-recoverable, since the state of the MSR[RI] bit is not reliable.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPU13

Description:

The MPC885/MPC875 IMMU works in the following way: Each time there is an access from the core, the access reaches the cache using the page from the previous access, on the assumption that the page has not changed. Usually that assumption is correct, so there is no stall. If the page has changed, a retry is done with the correct page. The MMU also signals the cache to disregard the previous access since the page was incorrect and the results are useless. When a TLB miss occurs, there is no retry, but the previous access is normally aborted since it used the wrong page value. Then the MMU shuts itself off until there is an abort signal from the core that specifies a change in flow. Normally, this change of flow is the ITLB miss exception, in which case the instruction pipeline is cleared before taking the exception. However, if the decrementer or external interrupt is asserted, this exception can take precedence over the ITLB miss exception.

Note that the problematic case is when the show cycles are enabled. There is an ITLB miss detected which is followed immediately by a decrementer or external interrupt exception. The first address of the exception routine is then presented by the core. Since the previous page was not valid (ITLB miss), the MMU provides an incorrect address (ff800500). As a result of the internal state of the MMU and the fact that it is a program trace show cycle, this cycle is (mistakenly) not cancelled, although it is re-tried. This causes the bad address to appear on the external bus as a show cycle. Although the show cycle is incorrect, the program flow usually continues correctly.

However, if the internal U-Bus pipeline happens to be full on the cycle discussed above, the retry is not handled correctly (as a result of the same issue mentioned above) and the second address of the routine (504) is used for the fetch instead of the first address (500). This causes the instructions of the first cache line to be fetched in the wrong order.

Projected Impact:

The instructions of the first cache line may be fetched in the wrong order.

Work Around:

1. Because the first instruction of the decrementer and external interrupt routines might not be executed before the second instruction, it should not do anything useful, so it won't matter if it is skipped. Therefore, a **nop** or **isync** instruction should be placed in this location. An **isync** instruction should be placed in the second location of these routines to re-synchronize the core with the IMMU. If the bug occurs, the isync instruction at 504 will be executed as if it is at location 500 and will cause another fetch of itself from address 504. At this point, the program flow is re-synchronized and continues as usual.

2. Disable Instruction fetch show cycle in ICTRL[IST_SER].

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPU14

Description:

When an "o" form integer instruction is started to execute before previously started "." form integer multiply or divide instruction completes, the Condition Register bit #3 (SO) may be wrongly updated by XER[SO] bit changed by "o" form instruction. For example, instruction sequence "divw. Rz, Rx, Ry, subfo Rt, Rv, Ru" may cause this problem. It does not happen if the "o" form instruction is also "." form or register dependencies exist.

Projected Impact:

The condition register bit #3 (SO) may be wrongly updated by XER[SO] bit changed by the "o" form instruction.

Work Around:

1. Keep the "o" form instruction 4 integer or 6 other instructions apart from integer divide "." form instruction in the code or more than 1 instruction in case of integer multiply instruction.

2. Use both instructions with the "." form.

3. Run RCPU in serialized mode.

4. Place the "sync" instruction between multiply/divide and "o" form integer instruction.

5. Do not use the "." form of integer divide or multiply instructions.

Projected Solution:

This errata will not be fixed due to the minimal system impact.

Errata Number: CPU15

Description:

An incorrect or invalid instruction is executed if all of the following are true:

- A conditional branch or branch to LR/CTR is executed in the last word of MMU page <n> (branches with primary opcodes 16 or 19).
- The target address of the branch points to the last cache line of MMU page <n>+1.
- The branch remains unresolved long enough for a prefetch of the first cache line of MMU page <n>+1 to start.
- Code execution continues after the branch beyond the end of MMU page <n>+1 into the first cache line of page <n>+2, which must already be in the cache.
- Both ICACHE and IMMU are enabled.

An incorrect or invalid instruction will be executed instead of the correct one out of the first cache line of MMU page <n>+2.

NOTE

Absolute branches (bx with primary opcode 18) are not the cause of the problem, as they are always resolved implicitly. Every MMU page containing a "risky" branch as described above is called a "risky" page below.

Workarounds:

A variety of workarounds are available:

- Do not put conditional branches or branches to LR/CTR into the last word of an MMU page, that is, generate and load the code image appropriately.

- If running code out of RAM, replace the conditional branch with an absolute branch, to patch a function containing the "true" branch code out of RAM, or replace each "risky" branch with a trap to a patch function if an absolute branch is not possible due to code size.

- Specifically for blr, add enough nops or other statements between mtlr and blr so that blr is definitely resolved during execution according to execution timing. This is only a partial workaround for blr.

- In the ITLB miss exception code, when loading the TLB for an MMU page, also invalidate any TLB referring to the next and previous page using tlbie. This intentionally forces an ITLB miss exception on every execution across sequential MMU page boundaries.

- Turn off ICACHE completely.

Below is a simple example ITLB miss handler showing how a tlbie-based workaround can be implemented. The actual code needed depends on how precisely the MMU is used in a system.

```
asm void os_inst_tlb_miss(void)
{
    mtspr        M_TW,r2      /* save r2 into M_TW */
    mfspr        r2,SRR0      /* SRR0 */
    mtspr        MD_EPN,r2    /* save instruction miss effective address in MD_EPN */
    mfspr        r2,M_TWB     /* load r2 with level one pointer from M_TWB*/
    lwz          r2,0(r2)     /* load level one page entry */
    mtspr        MI_TWC,r2    /* save level one attributes into MI_TWC */
    mtspr        MD_TWC,r2    /* save level two base pointer into MD_TWC */
    mfspr        r2,MD_TWC    /* load r2 with level two pointer while taking */
                        /* into account the page size */
    lwz          r2,0(r2)     /* load level two page entry */
    mtspr        MI_RPN,r2    /* Write TLB entry via MI_RPN */

    // The following code assumes that under all circumstances, instruction pages are
    // consecutive physical pages, i.e., the Physical Addresses (PA) of sequential pages
        // are sequential.
    // A page size of 4KB is also assumed.
    // If this is not the case, modifications are needed to translate the
    // address appropriately
    rlwinm       r2,r2,0,0,31-12 // Eliminate lower twelve bits (4K page!)
                            // extract "true" PA
    ori          r2,r2,0x1000-4  // Point to end of current page (4K page!)
    subi         r2,r2,0x1000    // Move to end of previous page
                        // (4K page! PA translation!)
    mtspr        SPRG0,r3
    mtspr        SPRG1,r4
    mfcr         r4
    lwz          r3,0(r2)        // Read instruction word at end of page using PA
                        //(This should work for LE)
    rlwinm       r3,r3,6,26,31   // Isolate primary opcode in lower 6 bits
    cmpwi        r3,16           // Conditional branch?
```

```
    beq             @tlbieprev      // Invalidate previous page
    cmpwi           r3,19           // Possibly conditional branch to register address?
    beq             @tlbieprev      // Invalidate previous page

    addi            r2,r2,0x1000    // Move back to current page address
                                    // (4K page! PA translation!)
    lwz             r3,0(r2)        // Read instruction word at end of page using PA
                                    //(This should work for LE)
    rlwinm          r3,r3,6,26,31   // Isolate primary opcode in lower 6 bits
    cmpwi           r3,16           // Conditional branch?
    beq             @tlbienext      // Invalidate next page
    cmpwi           r3,19           // Possibly conditional branch to register address?
    beq             @tlbienext      // Invalidate next page
    mtcr            r4
    mfspr           r4,SPRG1
    mfspr           r3,SPRG0

    mfspr           r2,M_TW     /* Restore r2 previously saved in M_TW */
    rfi

@tlbienext:
    addi            r2,r2,0x1000    // Move to next page address (4K page! PA translation!)
@tlbieprev:
    tlbie           r2              // Make sure we refetch the ITLB on the dangerous page
    mtcr            r4
    mfspr           r4,SPRG1
    mfspr           r3,SPRG0

    mfspr           r2,M_TW     /* Restore r2 previously saved in M_TW */
    rfi
} /* os_inst_tlb_miss */
```

A variant that is further optimized can often be generated to avoid memory reloads. However, this specifically depends on MMU usage in the system. The following example handler improves performance by relying on three bits of a L2 TLB part not being used by other software.

```
asm void os_inst_tlb_miss_l2_based(void)
{
    mtspr           SPRG0,r3
    mtspr           SPRG1,r4
    mfcr            r4

    mtspr           M_TW,r2     /* save r2 into M_TW */
    mfspr           r2,SRR0     /* SRR0 */
    mtspr           MD_EPN,r2   /* save instruction miss effective address in MD_EPN */
    mfspr           r2,M_TWB    /* load r2 with level one pointer from M_TWB*/
    lwz             r2,0(r2)    /* load level one page entry */
    mtspr           MI_TWC,r2   /* save level one attributes into MI_TWC */
    mtspr           MD_TWC,r2   /* save level two base pointer into MD_TWC */

    // We abuse the SPS/SH/CI bits as indicator of our state in terms of the erratum.
    // We know that they are normally 0 in most applications, which is why we chose them.
```

Chip Errata for the MPC885 Duet Family

```
    // SPS/SH/CI==0b000: We have never before encountered this code page. We need to decide
    // SPS/SH/CI==0b100: This page does not contain a "bad" branch.
    // SPS/SH/CI==0b001: This page contains a bad branch. tlbie the next page
    // SPS/SH/CI==0b010: The previous page contains a bad branch. tlbie the previous page
    mfspr       r2,MD_TWC   /* load r2 with level two pointer while taking into account
                                    the page size */
    lwz         r2,0(r2)    /* load level two page entry */
    rlwinm      r3,r2,0,31,27 // Clear SPS/SH/CI before updating TLB
    mtspr       MI_RPN,r3   /* Write TLB entry via MI_RPN */

    // Now the magic starts. We automagically learn if pages need to be treated.
    // Fast case, i.e., nothing special comes first.
@recheckstatus:
    rlwinm.     r2,r2,28,0,2    // Extract the SPS/SH/CI field left justified
    bge         @checkipage     // This page has not been checked before or is known
                                // to be "bad"

    mtcr        r4
    mfspr       r4,SPRG1
    mfspr       r3,SPRG0
    mfspr       r2,M_TW     /* Restore r2 previously saved in M_TW */
    rfi


@checkipage:
    beq         @evalpagestatus // This page has not been tested before, so we need to
                                // update its info

    // When we get to this point, we know that we need to tlbie a page.
    // Depending on the SPS/SH/CI bits, we tlbie the next or previous page.
    // The initial assumption is that we have to tlbie the next page.
    // This code works only for 4KB pages.
    mfspr       r3,SRR0         // The address pointing into the current page
    addi        r3,r3,0x1000    // Move to next page (4K page! PA translation!)
    andis.      r2,r2,0x2000    // This maps to 0b001 in the left justified value
    bne         @tlbienext
    subi        r3,r3,0x2000    // Move back to prev page (4K page! PA translation!)
@tlbienext:
    tlbie       r3              // Make sure we refetch the ITLB on the dangerous page

@pagecheckdone:
    mtcr        r4
    mfspr       r4,SPRG1
    mfspr       r3,SPRG0
    mfspr       r2,M_TW     /* Restore r2 previously saved in M_TW */
    rfi

// The code below probably does not need to be locked in the cache. It won't run a lot.
@evalpagestatus:
    // The following code assumes that under all circumstances, instruction
    // pages are mapped to consequtive
    // physical pages, i.e., the Physical Addresses of sequential pages are sequential.
    // A page size of 4KB is also assumed.
```

Chip Errata for the MPC885 Duet Family

```
        // If this is not the case, modifications are needed to translate the address
        // appropriately
        rlwinm      r2,r3,0,0,31-12 // Eliminate lower twelve bits (4K page!) to extract
                                    // "true" PA
        ori         r2,r2,0x1000-4  // Point to end of current page (4K page!)
        lwz         r3,0(r2)        // Read instruction word at end of page using PA
                                    // (This should work for LE)
        rlwinm      r3,r3,6,26,31   // Isolate primary opcode in lower 6 bits
        cmpwi       r3,16           // Conditional branch?
        beq         @riskypage      // This is a risky page
        cmpwi       r3,19           // Possibly conditional branch to register address?
        beq         @riskypage      // This is a risky page

        subi        r2,r2,0x1000    // Move to the previous page (4K page! PA translation!)
        lwz         r3,0(r2)        // Read instruction word at end of page using PA
                                    // (This should work for LE)
        rlwinm      r3,r3,6,26,31   // Isolate primary opcode in lower 6 bits
        cmpwi       r3,16           // Conditional branch?
        beq         @prevrisky      // We are on the page after a risky one
        cmpwi       r3,19           // Possibly conditional branch to register address?
        beq         @prevrisky      // We are on the page after a risky one

        // This page is not risky and not near a risky page. So we mark it as safe!
        mfspr       r3,MD_TWC       /* load r3 with level two pointer while taking into account
                                        the page size */
        lwz         r2,0(r3)        /* load level one page entry */
        ori         r2,r2,0x0008    // Set SPS/SH/CI to 0b100 for an innocent page
        stw         r2,0(r3)        /* store level one page entry */
        b           @pagecheckdone

@prevrisky:
        mfspr       r3,MD_TWC       /* load r3 with level two pointer while taking into account
                                        the page size */
        lwz         r2,0(r3)        /* load level one page entry */
        ori         r2,r2,0x0004    // Set SPS/SH/CI to 0b010 for a page next to a risky one
        stw         r2,0(r3)        /* store level one page entry */
        b           @recheckstatus

@riskypage:
        mfspr       r3,MD_TWC       /* load r3 with level two pointer while taking into account
                                        the page size */
        lwz         r2,0(r3)        /* load level one page entry */
        ori         r2,r2,0x0002    // Set SPS/SH/CI to 0b001 for a risky page
        stw         r2,0(r3)        /* store level one page entry */
        b           @recheckstatus

} /* os_inst_tlb_miss_l2_based */
```

Chip Errata for the MPC885 Duet Family

ESAR6   I2C/SPI relocation feature

Description:

The MPC866/859T/852T supports I2C or SPI parameter relocation without a RAM microcode.

NOTE

Do not touch the 16-bit locations IMMR+0x3cac or IMMR+0x3dac on init. If they are overwritten, I2C/SPI cease to function. Locations 0x3cac/0x3dac contain a 16-bit internal DPRAM offset to the actual I2C/SPI parameter RAM. They must be left alone after a CPM reset, or can be initialized to point to a 32-byte aligned parameter area containing the I2C/SPI parameter areas to allow I2C/SPI parameter relocation in case of a parameter RAM conflict.

Errata Number: ESAR6

Description:

The MPC885 and MPC875 supports I2C or SPI parameter relocation without a RAM microcode.

Note that the user *must not* touch the 16-bit locations IMMR+0x3cac or IMMR+0x3dac on init. If they are overwritten, I2C/SPI will cease to function. Locations 0x3cac/0x3dac contain a 16-bit internal DPRAM offset to the actual I2C/SPI parameter RAM. Either they must be left alone after a CPM reset or can be initialized to point to a 32-byte aligned parameter area containing the I2C/SPI parameter areas to allow I2C/SPI parameter relocation in case of a parameter RAM conflict.

Projected Impact:

If the 16-bit locations IMMR+0x3cac or IMMR+0x3dac are overwritten, I2C/SPI will cease to function.

Work Around:

None.

Errata Number: ESAR7

(UTOPIA-Serial ATM / PTP switching ucode): Overlapping usage of the parameter BDPOINT in serial ATM by both Rx and Tx may cause errors. (MPC885 Only)

Description:

Due to overlapping usage of the parameter BDPOINT in serial ATM (either general serial ATM or PTP mode) by both Rx and Tx, an incorrect SDMA address may be used, leading to invalid memory accesses.

Projected Impact:

This errata can trash memory content or also freeze the CPM due to bus errors. UTOPIA is not affected.

Work Around:

Use a serial ATM SCC only in half duplex, that is, only Rx or only Tx. Do not enable both sides of the serial SCC. Or use a RAM microcode patch (TBD).

Errata Number: ESAR8

(UTOPIA-Serial PTP switching ucode) Port-To-Port to a serial ATM SCC with a PTP TCT in external memory can lead to trashed BD rings and stopped PTP traffic on that channel. (MPC885 Only)

Description:

> Due to a parameter update synchronization issue, Port-To-Port to a serial ATM SCC with a PTP TCT in external memory can lead to trashed BD rings and stopped PTP traffic on that channel.

Projected Impact:

> Port-To-Port to a serial ATM SCC with a PTP TCT in external memory can lead to trashed BD rings and stopped PTP traffic on that channel. UTOPIA is not affected.

Work Around:

> On a serial ATM SCC, use only internal PTP TCTs (channels 0-31). Or use a RAM microcode patch (TBD).

Errata Number: ATM12

APC Overruns cause potential loss of paced Bandwidth. (MPC885 Only)

Description:

When an APC Overrun event occurs and APCST[APCOM]==0, APCT_PTR is not updated correctly, leading to a potential loss in scheduled bandwidth.

Projected Impact:

APCT_PTR is not updated correctly, leading to a potential loss in scheduled bandwidth.

Work Around:

Always set APCOM if an APC Overrun must not affect the maximum available bandwidth to be scheduled.

Errata Number: ATM14

Statistical counters do not count properly in NIM mode and when MCF filtering SRSTATE[MCF]=1 is used for an RCT. (MPC885 Only)

Description:

> When MCF filtering SRSTATE[MCF] = 1 is used for an RCT, and the Non-Intrusive Mode (RCT[NIM]) bit is set, the MCF pass all data and OAM cells to the channel's RxBD. However, it also screens out all OAM cells to the raw cell queue Rx BD's for host monitoring. To implement this mode (NIM), each OAM cell is duplicated using the microcode and also counted by the statistical counters. As a result, the statistical counters count each OAM cell twice and the statistical information is not accurate.

Projected Impact:

> The statistical counters count each OAM cell twice so the statistical information is not accurate.

Work Around:

> To get the accurate results in the statistical counters, a simple statistical counter adjustment can be done. The user should decrease the number of OAM cells that were screened out from the statistical counters.

> For example: 100 OAM cells (out of 200 cells) where received. The TotalRxCell count will be 300 (100 regular cells + 100x2 OAM cells).

> In the Raw cell queue 100 cells screened out due NIM mode and the adjustment will be as follows:

> Adjusted TotalRxCell = TotalRxCell - Num of Raw cell queue cells.

Errata Number: FEC12

CRC Error Reported if MII_RX_ER Asserts While in Internal Loopback Mode

Description:

External inputs on the MII interface should be ignored when in internal loopback mode. The receive logic currently mistakenly reports a CRC error if MII_RX_ER asserts when a frame is being looped internally.

Projected Impact:

The receive logic currently mistakenly reports a CRC error if MII_RX_ER asserts when a frame is being looped internally.

Work Around:

When running an internal loopback test, the MII_RX_ER signal must be held low.

Errata Number: FEC14

7-wire Interface Compatibility Problem

Description:

The FEC uses the receive clock to clock in data and run the FEC while the data is coming in. When the message has ended (signified by RENA being negated), the FEC switches to an internal clock source. It was designed assuming the external clock quits immediately after RENA is negated. Certain framers actually give clock pulses after RENA is negated. If a clock pulse is input at the same time the FEC is switching internal clock sources, a glitch may occur on the internal clock line, which may cause unpredictable results.

Work Around:

An external AND gate can be used to gate the receive clock with RENA

When running an internal loopback test, the MII_RX_ER signal must be held low.

Projected Impact:

A glitch may occur on the internal clock line which may cause unpredictable results.

How to Reach Us:

Home Page:
www.freescale.com

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

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
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

For Literature Requests Only:
Freescale Semiconductor
    Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
    @hibbertgroup.com

Document Number: MPC885CE
Rev. 6
04/2009