MPC8360E Chip Errata

This document details all known silicon errata for the MPC8360. The following table provides a revision history for this document.

Table 1. Document Revision History

Revision	Date	Significant Changes
5	9/2011	• Added CPU-A022, SEC-A001 • Modified PCI20 workaround
4	8/4/2010	• In QE_UART3, changed workaround to "None."
3	8/2010	• Added QE_ENET-A002, QE_ATM-A001, QE_UART-A001, QE_QMC-A001, and QE_General-A003 • Updated General12 • Updated General17 and QE_HDLC2 workaround • In QE_ENET25, moved what had been 3rd bullet in description to the impact section. • In QE_UART1, added the word "simultaneous" to headline and description. • In workaround for QE_UART1, QE_UART2, QE_UART3, QE_UART5 changed "Use microcode patch found in QERAMPKG" To "Use a microcode patch."
2	9/2009	• Added General16, General17, LBC1, LBC-A001, QE_USB5, QE_USB6, QE_HDLC3, QE_ENET22, QE_ENET24, QE_ENET25, QE_L2_SWITCH2, QE_L2_SWITCH3, QE_MCC4, QE_UPC-A001, QE_UART7, QE_SI5, QE_BISYNC3, QE_QMC3, QE_Transparent1, QE_Transparent2 • Updated QE_UART6
1	12/6/2007	Initial release

The following table provides a cross-reference to match the revision code in the processor version register to the revision level marked on the part.

Table 2. MPC8360E Family Devices and Silicon Revisions

Device	Silicon Revision	2.0	2.1
	Mask	1M13E	1M13E
MPC8360E		√	√

Table continues on the next page...

Table 2. MPC8360E Family Devices and Silicon Revisions (continued)

Device	Silicon Revision	2.0	2.1
	Mask	1M13E	1M13E
MPC8358E		√	√

Table 3 summarizes all known errata and lists the corresponding silicon revision level to which they apply. A 'Yes' entry indicates the erratum applies to a particular revision level, and an 'No' entry means it does not apply.

Table 3. Summary of Silicon Errata and Applicable Revision

Errata	Name	Projected Solution	Silicon Rev.	
			2.0	2.1
	CPU			
CPU-A022	The e300 core may hang while using critical interrupt	No plans to fix	Yes	Yes
	DDR			
DDR18	MCK to MDQS (t_{DDKHMH}) timing does not meet the specification	Hardware specification updated for revision 2.1	Yes	No
	RESET			
RESET3	External Soft reset functionality is not functional	No plans to fix	Yes	Yes
	General			
General5	Additional 256K clock cycles of $\overline{\text{HRESET}}$ signal	No plans to fix	Yes	Yes
General9	CPU/QE maximum frequency combination of 667/500 is not supported	Fixed in Rev 2.1	Yes	No
General12	CSB deadlock	No plans to fix	Yes	Yes
General13	$\overline{\text{TGATE4}}$ is not functional if the reset configuration word is taken from any hard-coded option	No plans to fix	No	Yes
General16	Enabling I^2C could cause I^2C bus freeze when other I^2C devices communicate	No plans to fix	Yes	Yes
General17	DUART: Break detection triggered multiple times for a single break assertion	No plans to fix	Yes	Yes
	SEC			
SEC6	AES-CTR mode data size error	No plans to fix	Yes	Yes
SEC7	Single descriptor SRTP error	No plans to fix	Yes	Yes
SEC-A001	Channel Hang with Zero Length Data	No plans to fix	Yes	Yes
	PCI			
PCI15	Assertion of $\overline{\text{STOP}}$ by a target device on the last beat of a PCI memory write transaction can cause a hang	No plans to fix	Yes	Yes
PCI16	PCI does not work when PCI:CSB ratio is 1:16	No plans to fix	Yes	Yes

Table continues on the next page...

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

Errata	Name	Projected Solution	Silicon Rev.	
			2.0	2.1
PCI17	PCI input hold timing (t_{PCIXKH}) does not meet the specification	Fixed in Rev 2.1	Yes	No
PCI19	Dual-address cycle inbound write accesses can cause data corruption	No plans to fix	Yes	Yes
PCI20	PCI controller may hang when returning from "PCI pins low" state	No plans to fix	Yes	Yes
PCI21	PCI clock to output valid (t_{PCKHOV}) does not meet the specification	Fixed in Rev 2.1	Yes	No
LBC				
LBC1	UPM does not have indication of completion of a RUN PATTERN special operation	No plans to fix	Yes	Yes
LBC-A001	The Local Bus Controller, when configured in DLL enabled mode, may not work after Reset is applied	No plans to fix	Yes	Yes
QE				
QE_USB2	USB receive input signal is inverted	Fixed in Rev 2.1	Yes	No
QE_USB5	USB Host might halt in automatic SOF transmission mode	No plans to fix	Yes	Yes
QE_USB6	Reset cannot be driven by the host when USB is using PG11 & PG12 pins	No plans to fix	Yes	Yes
QE_HDLC1	UCC does not support HDLC bus mode through the TDM interface	No plans to fix	Yes	Yes
QE_HDLC2	UCC TxD is always set as open drain in HDLC bus NMSI mode	Fixed in Rev 2.1	Yes	No
QE_HDLC3	"Enter Hunt Mode" host command is not functional for UCC in HDLC mode	No plans to fix	Yes	Yes
QE_ENET1	RMON errors	No plans to fix	Yes	Yes
QE_ENET2	Rx MAC might accept dropped frame	No plans to fix	Yes	Yes
QE_ENET10	RGMII AC values do not meet the specification	Improved in Rev 2.1	Yes	Yes
QE_ENET18	GMII AC values do not meet the specification	Improved in Rev 2.1	Yes	Yes
QE_ENET19	TBI AC values do not meet the specification	Improved in Rev 2.1	Yes	Yes
QE_ENET20	UEC may stop transmitting after late collision	No plans to fix	Yes	Yes
QE_ENET22	Ethernet receiver may become stuck while in a multithreaded configuration	No plans to fix	Yes	Yes
QE_ENET24	Magic Packet sequence embedded in partial sequence not recognized	No plans to fix	Yes	Yes
QE_ENET25	Malformed Magic Packet triggers Magic Packet exit	No plans to fix	Yes	Yes
QE_ENET-A002	False broadcast indication in RGMII mode	No plans to fix	Yes	Yes
QE_L2_SWITCH2	Invalid memory access	No plans to fix	Yes	Yes
QE_L2_SWITCH3	Error may occur for any TCBD	No plans to fix	Yes	Yes

Table continues on the next page...

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

Errata	Name	Projected Solution	Silicon Rev.	
			2.0	2.1
QE_MCC4	MCC INIT host commands may cause GUN event	No plans to fix	Yes	Yes
QE_ATM9	ATM policer is not functional	Fixed in Rev 2.1	Yes	No
QE_ATM10	AAL5 transmitter could stop transmission	Fixed in Rev 2.1	Yes	No
QE_ATM11	IDLE state can halt ATM transmitter operation	Fixed in Rev 2.1	Yes	No
QE_ATM12	Cell duplication can occur when during ATM AAL2 operation on IMA	Fixed in Rev 2.1	Yes	No
QE_ATM-A001	ATM APC flux compensation mode is not functional	No plans to fix	Yes	Yes
QE_UPC2	User-defined cell Rx extra header size limited to 8 bytes	No plans to fix	Yes	Yes
QE_UPC3	UPC AC values do not meet the specification	Package specific	Yes	No
QE_UPC-A001	UPC may hang or receive corrupted data in receive buffer when RVAL is tri-stated	No plans to fix	Yes	Yes
QE_UART1	Simultaneous loopback and echo mode is not functional	No plans to fix	Yes	Yes
QE_UART2	Overrun indication is not reported on the current BD	No plans to fix	Yes	Yes
QE_UART3	DRT mode (UPSMR[DRT] = 1) is not functional	No plans to fix	Yes	Yes
QE_UART5	Rx FIFO data can be lost when ENTER HUNT MODE command is executed	Fixed in Rev 2.1	Yes	No
QE_UART6	UART/AHDLC receive data corruption on QUICC Engine UCC	No plans to fix	Yes	Yes
QE_UART7	UART does not set UCCE[AB] correctly for autobaud	No plans to fix	Yes	Yes
QE_UART-A001	QE UART controller idle status bit may not be reliable	No plans to fix	Yes	Yes
QE_SI5	Disabling the TDM interface on-the-fly may cause an MCC GUN event	No plans to fix	Yes	Yes
QE_BISYNC2	RxBD in continuous mode is not closed after an Rx parity error	No plans to fix	Yes	Yes
QE_BISYNC3	BISYNC controller does not enter HUNT MODE	No plans to fix	Yes	Yes
QE_IEEE1588_1	Out of band detection of PTP frames is not functional on UCC1 and UCC2	Fixed in Rev 2.1	Yes	No
QE_SPI1	SPI output hold timing (t_{NIKHOX}) does not meet the specification	Hardware Specfication updated for revision 2.1	Yes	No
QE_QMC1	Corruption of interrupt queue entries	Fixed in Rev 2.1	Yes	No
QE_QMC2	Data may be shifted if QMC is disabled and enabled during reinitialization or reconfiguration	No plans to fix	Yes	Yes
QE_QMC3	Missing Interrupt after Stop Rx Host command	No plans to fix	Yes	Yes
QE_QMC-A001	QMC GOV event is not reported in the UCC event register	No plans to fix.	Yes	Yes
QE_Transparent1	The CRC error is not checked in transparent mode if $\overline{\text{CD}}$ is de-asserted	No plans to fix	Yes	Yes
QE_Transparent2	QE may erroneously report CRC error if GUMR[REVD] is set to 1 in transparent mode	No plans to fix	Yes	Yes

Table continues on the next page...

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

Errata	Name	Projected Solution	Silicon Rev.	
			2.0	2.1
QE_General4	Potential spikes on BRG output clock when using odd divisions	Fixed in Rev 2.1	Yes	No
QE_General-A003	High Tx Virtual FIFO threshold size can cause UCC to halt	No plans to fix	Yes	Yes
DMA				
DMA2	Data corruption by DMA when destination address hold (DAHE) bit is used	No plans to fix	Yes	Yes

CPU-A022: The e300 core may hang while using critical interrupt

Description: Devices: MPC8360E, MPC8358E

If BOTH critical interrupt AND normal interrupt types are used in a system, the e300 core may hang.

Impact: The processor may stop dispatching instructions until a hardware reset($\overline{\text{HRESET}}$). Debug tools will not be able to read any register correctly except program counter IAR which points to a location in the critical interrupt vector.

Workaround: If both critical interrupt and normal interrupt types are used, then instead of using an **rfi** instruction at the end of every exception handler, replace the **rfi** with the following:

1. Disable critical interrupts by setting MSR[CE] to 0 with a **mtspr** instruction.
2. Copy SRR0 and SRR1 to CSRR0 and CSRR1, respectively.
3. Execute an **rfci** instruction. This enables MSR[CE] and any other bits that the original **rfi** would have set including the MSR[EE].

Sample Code:

```
// Disable MSR[CE]
mfmsr r2
lis r3, 0xffff
ori r3, r3, 0xff7f
and r2, r2, r3
sync
mtmsr r2
isync
// Copy SRR0, SRR1 to CSRR0 and CSRR1
mfspr r2, srr0
mfspr r3, srr1
mtspr csrr0, r2
mtspr csrr1, r3
...restore GPRs
rfci
```

Fix plan: No plans to fix

DDR18: MCK to MDQS (t_{DDKHMH}) timing does not meet the specification

Description: Devices: MPC8360E, MPC8358E

MCK to MDQS timing does not meet the specification. The t_{DDKHMH} minimum value is -0.9 ns instead of -0.6 ns.

Impact: The timing margin between the MDQS to MCK skew and the DDR required minimum setup is reduced.

Workaround: None.

Fix plan: Hardware specification updated for revision 2.1

RESET3: External Soft reset functionality is not functional

Description: Devices: MPC8360E, MPC8358E

External Soft reset is defined such that the DDR controller and the local bus controller will not be reset in the event of an external soft reset. However, the internal bus masters and the internal bus components will be held in reset immediately and as long as external soft reset event is valid and the soft reset sequence is in progress. As a result, if the soft reset event happens in the middle of an outstanding write transaction which is targeted to the DDR main memory or to one of the local bus slaves, the internal bus component will transition to reset state while it needs to supply the data to be written and as a result wrong data could be written to the main memory or to the local bus slave.

External Soft reset is defined such that it has no effect on system configuration registers, including IMMRBAR. As a result, software should behave different in regards to IMMRBAR initialization when recovering from soft reset as opposed to recovering from hard reset (in which the IMMRBAR is guaranteed to be located in its default mapping). Since software cannot tell if it is recovering from soft reset or hard reset without reading the platform's RSR register, and since software has to know the IMMRBAR value in order to read the RSR register, there is no way to recover from soft reset (unless the IMMRBAR is kept in its default value at all time).

Impact: Recovery from a soft reset event is not guaranteed.

Workaround: Do not use soft reset at all. Use hard reset or power on reset. SRESET signal must be used as an output-only signal. Software must not write 1 to the RCR[SWSR] bit.

Fix plan: No plans to fix

General5: Additional 256K clock cycles of $\overline{\text{HRESET}}$ signal

Description: Devices: MPC8360E, MPC8358E

New I^2C module will cause the loading of HRCW to be extended by the additional 256K cycles of PCI_SYNC_IN (in the worst case).

Impact: HRCW loading time will be extended by an additional 256K PCI_SYNC_IN clock cycles.

Workaround: None

Fix plan: No plans to fix

General9: CPU/QE maximum frequency combination of 667/500 is not supported

Description: Devices: MPC8360E

Achieving a 667 MHz CPU and 500 MHz QE frequency across full voltage and full temperature per the hardware specification range will not be supported with rev 2.0 silicon. A relaxation of either maximum junction temperature or minimum operating voltage, or some combination of those two, will be required to achieve either 667 MHz CPU or 500 MHz QE performance.

Impact: A CPU/QE frequency combination of 667/500 MHz is not supported.

Workaround: Relax either maximum junction temperature or minimum operating voltage, or some combination of the two, to achieve either 667 MHz CPU or 500 MHz QE performance.

For example, a frequency combination of 500/333/500 (CPU/DDR/QE) or 667/333/400 (CPU/DDR/QE), both with a T_j relaxation from 105 deg C to 70 deg C and a V_{DD} = 1.3V ± 0.5V, can be used. In order to obtain a 500/333/500 or 667/333/400, contact your Freescale Sales Representative.

A higher T_j is possible by lowering either the CPU (to less than 667 MHz) or the QE frequency (to less than 500 MHz) of the above combinations or increasing the V_{DD} value. For more information on possible combinations and the relationship to temperature and voltage, please contact Freescale.

Fix plan: Fixed in Rev 2.1

General12: CSB deadlock

Description: Devices: MPC8360E, MPC8358E

Case 1: Instruction Caching is disabled for PCI space

If the e300 core initiates an instruction read from the PCI outbound space and, before its completion, starts another high priority data read from the same cache-line, a deadlock on the CSB results.

The gasket between the CSB and I/O sequencer (IOS) will ARTRY the first instruction read due to a ‘delayed read’ from PCI space. The gasket does not accept any further transactions within the same cache line until the previous one completes. When the e300 core performs a high priority data read after initiating an instruction fetch, it does not rerun the instruction read before the data load has been serviced. On the other hand, CSB2IOS does not serve data read from the same cache line until the instruction read completes. Thus, both the e300 core and CSB2IOS respectively wait for the data read and instruction read to complete, hence resulting in a deadlock.

Case 2: Instruction Caching is enabled for PCI space

When Instruction Caching is enabled for PCI space, e300 fetches complete cache-line from instruction space before it starts to execute it. Now if instruction in currently executed cache-line loads data from next cache-line, then it will create same scenario as Case 1 above.

Here core would initiate instruction fetch from next cache-line. On encountering data-load instruction (before instruction fetch completion), it will initiate high priority data read from next cache-line. As a result, the gasket between the CSB and I/O sequencer (IOS) will again have instruction read and data read from same cache-line with high priority data read following instruction read. Hence it will result in deadlock on CSB bus.

Impact: Deadlock on CSB. The gasket between the CSB and IOS checks the 32-bit address, transfer-type, transfer size, and transfer burst fields before deciding on a further course of action. If all four fields are identical to some previous buffered transaction, the gasket will generate ARTRY until read data is available. If any of the fields are different, the gasket will treat that as a different transaction and forward it to PCI.

Workaround: Use one of the following workarounds:

- Instruction space and data space should be kept mutually exclusive of each other.
- When executing a code from the PCI outbound space, data reads should not fall on the same cache line as an ongoing instruction read.

Fix plan: No plans to fix

General13: $\overline{\text{TGATE4}}$ is not functional if the reset configuration word is taken from any hard-coded option

Description: Devices: MPC8360E, MPC8358E

If any of the hard-coded options are used (CFG_RESET_SOURCE[0:2] is 011 to 111) for the reset configuration word, $\overline{\text{TGATE4}}$ input signal is masked and can not be used by GTM4, GTM3–GTM4 pair-cascade, or by GTM1–GTM4 super-cascade.

Impact: GTM4 cannot use the external gate ($\overline{\text{TGATE4}}$) if one of the hard-coded options is used for the reset configuration word. This also applies to cascade of GTM3–GTM4 and to super-cascade of GTM1–GTM4.

Workaround: Use one of the following options:

- Use GTM1, GTM2, or GTM3 for external $\overline{\text{TGATE4}}$
- Use alternative reset configuration source (EEPROM or I^{2}C).

Fix plan: No plans to fix

General16: Enabling I^{2}C could cause I^{2}C bus freeze when other I^{2}C devices communicate

Description: Devices: MPC8360E, MPC8358E

When the I^{2}C controller is enabled by software, if the signal SCL is high, the signal SDA is low, and the I^{2}C address matches the data pattern on the SDA bus right after enabling, an ACK is issued on the bus. The ACK is issued because the I^{2}C controller detects a START condition due to the nature of the SCL and SDA signals at the point of enablement. When this occurs, it may cause the I^{2}C bus to freeze. However, it happens very rarely due to the need for two conditions to occur at the same time.

Impact: Enabling the I^{2}C controller may cause the I^{2}C bus to freeze while other I^{2}C devices communicate on the bus.

Workaround: Use one of the following workarounds:

- Enable the I^{2}C controller before starting any I^{2}C communications on the bus. This is the preferred solution.
- If the I^{2}C controller is configured as a slave, implement the following steps:
  a. Software enables the device by setting I2CnCR[MEN] = 1 and starts a timer.
  b. Delay for 4 I^{2}C bus clocks.
  c. Check Bus Busy bit (I2CnSR[MBB])

```
if MBB == 0
                jump to Step f; (Good condition. Go to Normal
operation)
                else
                Disable Device (I2CnCR[MEN] = 0)
```

  d. Reconfigure all I^{2}C registers if necessary.
  e. Go back to Step a.
  f. Normal operation.

Fix plan: No plans to fix

General17: DUART: Break detection triggered multiple times for a single break assertion

Description: Devices: MPC8360E, MPC8358E

A DUART break signal is defined as a logic zero being present on the UART data pin for a time longer than (START bit + Data bits + Parity bit + Stop bits).The break signal persists until the data signal rises to a logic one.

A received break is detected by reading the ULSRn and checking for BI=1. This read to ULSRn will clear the BI bit. Once the break is detected, the normal handling of the break condition is to read the URBR to clear the ULSRn[DR] bit. The expected behavior is that the ULSRn[BI] and ULSRn[DR] bits will not get set again for the duration of the break signal assertion. However, the ULSRn[BI] and ULSRn[DR] bits will continue to get set each character period after they are cleared. This will continue for the entire duration of the break signal.

At the end of the break signal, a random character may be falsely detected and received in the URBR, with the ULSRn[DR] being set.

Impact: The ULSRn[BI] and ULSRn[DR] bits will get set multiple times, approximately once every character period, for a single break signal. A random character may be mistakenly received at the end of the break.

Workaround: The break is first detected when ULSRn is read and ULSRn[BI]=1. To prevent the problem from occuring, perform the following sequence when a break is detected:

1. Read URBRn, which will return a value of zero, and will clear the ULSRn[DR] bit
2. Delay at least 1 character period
3. Read URBRn again

ULSR[BI] will remain asserted for the duration of the break. The UART block will not trigger any additional interrupts for the duration of the break.

This workaround requires that the break signal be at least 2 character-lengths in duration.

This workaround applies to both polling and interrupt-driven implementations.

Fix plan: No plans to fix

SEC6: AES-CTR mode data size error

Description: Devices: MPC8360E, MPC8358E

The SEC 2.1 supports acceleration of AES Counter mode, an underlying algorithm in Secure Realtime Transport Protocol (SRTP), and optionally, IPSec. The SEC is designed to accelerate AES-CTR alone (using descriptor type 0001_0) or in parallel with an HMAC-SHA-1 using a special SRTP descriptor type 0010_1. SRTP uses AES-CTR with HMAC-SHA-1. Although AES in counter mode (AES-CTR) is meant to act as a stream cipher, the AESU considers any input data size that is not an even multiple of 16 bytes to be an error.

Impact: None

Workaround: Use one of the following options:

- The AESU Data Size error can be disabled via the AESU Interrupt Control Register to prevent a nuisance interrupt.
- The input data length (in the descriptor) can be rounded up to the nearest 16B. Set the data-in length (in the descriptor) to include X bytes of data beyond the payload. Set the data-out length to only output the relevant payload (don't need to output the padding). SEC reads from memory are not destructive, so the extra bytes included in the AES-CTR operation can be whatever bytes are contiguously trailing the payload.

Fix plan: No plans to fix

SEC7: Single descriptor SRTP error

Description: Devices: MPC8360E, MPC8358E

The SRTP protocol specifies the use of AES-CTR with HMAC-SHA-1. The SEC 2.1 is designed to accelerate AES-CTR in parallel with HMAC-SHA-1 using a special SRTP descriptor type 0010_1. Single descriptor SRTP does not work for data sizes that are not even multiples of 16 bytes.

Impact: When operating on input data which is N*16B, the AESU and MDEU can each read in the portion of the datastream relevant to their respective operations. When the input data is not N*16B, the AESU data size workaround described in SEC6 (rounding up to the next 16B) does not work because these excess bytes are ‘snooped’ by the MDEU and corrupt the HMAC-SHA-1 operation.

Workaround: Because the SRTP protocol does not require the payload to be N*16B, most packets will likely be of a data length that hits this erratum. The workaround is to use two descriptors to perform SRTP.

Outbound: The first descriptor is type 0001_0 "common non-snoop" set for an AES-CTR operation using either of the workarounds described in SEC6. The second descriptor is type 0001_0 “common non-snoop” set for an HMAC-SHA-1 of the headers + unpadded encrypted payload + MKI (when present).

Inbound: Same descriptors as outbound, but in reverse order.

To minimize the performance impact of this workaround, these two descriptors should be created simultaneously and launched back-to-back. By configuring the SEC Crypto-channel to perform Done notification on selected descriptors, the first descriptor should be set to not generate a Done interrupt, while the second descriptor (which completes the SRTP operation) should be set to generate the Done interrupt. All the parameters required to build both descriptors are available at the start of the request to the SEC device driver, so there is no reason to wait for the first descriptor to complete before building and launching the second.

Fix plan: No plans to fix

SEC-A001: Channel Hang with Zero Length Data

Description: Devices: MPC8360E, MPC8358E

Many algorithms have a minimum data size or block size on which they must operate. The SEC EUs detect when the input data size is not a legal value and signal this as an error. For most EUs, a zero byte length data input should be considered illegal, however the EUs do not properly notify the crypto-channel using the CHA of a data size error. Instead, the EUs wait forever for a valid data length, leading to an apparent channel hang condition.

Impact: When EUs detect illegal input data size, EUs wait forever for a valid data length, leading to an apparent channel hang condition.

Workaround: Ensure that software does not create SEC descriptors to encrypt or decrypt zero length data.

Fix plan: No plans to fix

PCI15: Assertion of $\overline{\text{STOP}}$ by a target device on the last beat of a PCI memory write transaction can cause a hang

Description: Devices: MPC8360E, MPC8358E

As a master, the PCI IP block can combine a memory write to the last PCI double word (4 bytes) of a cacheline with a 4 byte memory write to the first PCI double word of the subsequent cacheline.

This only occurs if the second memory write arrives to the PCI IP block before the deassertion of $\overline{\text{FRAME}}$ for the first write transaction. If the writes are combined, the PCI IP block masters a single memory-write transaction on the PCI bus. If for this transaction, the PCI target asserts $\overline{\text{STOP}}$ during the last data beat of the transaction ($\overline{\text{FRAME}}$ is deasserted, but $\overline{\text{TRDY}}$ and $\overline{\text{IRDY}}$are asserted), the transaction completes correctly. A subsequent write transaction other than an 8-byte write transaction causes a hang on the bus. Two different hang conditions can occur:

- If the target disconnects with data on the first beat of this last write transaction, the PCI IP block deasserts $\overline{\text{IRDY}}$ on the same cycle as it deasserts $\overline{\text{FRAME}}$ (PCI protocol violation), and no more transactions will be mastered by the PCI IP block.
- If the target does not disconnect with data on the first beat of this last write transaction, $\overline{\text{IRDY}}$ will be deasserted after the first beat is transferred and will not be asserted anymore after that, causing a hang.

Impact: This affects 32-bit PCI target devices that blindly assert $\overline{\text{STOP}}$ on memory-write transactions, without detecting that the data beat being transferred is the last data beat of the transaction. It can cause a hang.

If the PCI transaction is a one data beat transaction and the target asserts $\overline{\text{STOP}}$ during the transfer of that beat, there is no impact.

Workaround: Hardware workaround:

Ensure that the PCI target device does not assert $\overline{\text{STOP}}$ during the last beat of a PCI memory write transaction that is greater than one data beat and crosses a cacheline boundary. It could assert $\overline{\text{STOP}}$ during the last data beat of the 32-byte cacheline or not assert $\overline{\text{STOP}}$ at all.

Software workarounds:

Set bit 10, the master disabling streaming (MDS) bit, of the PCI bus function register (address 0x44) to prevent the combining of discrete outbound PCI writes or the recombining of writes that cross the 32-byte cacheline boundary into a burst.

Set the PCI latency timer register (offset 0x0D) to zero. A value of zero is the reset value for this register, so keeping this register unmodified after reset prevents the PCI IP block from ever combining writes. It is not necessary to set the PCI latency timer register to zero if the MDS bit is set.

Fix plan: No plans to fix

PCI16: PCI does not work when PCI:CSB ratio is 1:16

Description:	Devices: MPC8360E, MPC8358E PCI does not work with a clock ratio of 1:16 (PCI to CSB) when using non-standard PCI frequencies above 25 MHz.
Impact:	The CSB frequency is limited for non-standard PCI frequencies below 25 MHz.
Workaround:	None.
Fix plan:	No plans to fix

PCI17: PCI input hold timing (t_{PCIXKH}) does not meet the specification

Description: Devices: MPC8360E, MPC8358E

PCI input hold timing (data to input clock at receiver) does not meet the specification for 33 MHz and 66 MHz. The t_{PCIXKH} minimum value is 1 ns instead of 0ns.

Impact: PCI interface may encounter sporadic data corruption.

Workaround: The board design should take into account the input hold timing violation.

Fix plan: Fixed in Rev 2.1

PCI19: Dual-address cycle inbound write accesses can cause data corruption

Description: Devices: MPC8360E, MPC8358E

When using a dual-address cycle (DAC) for inbound write accesses and when the IOS is full, (that is, a previous PCI request to the IOS is being retried), the PCI overwrites the address for the IOS with the new address from the bus, despite the transaction being retried on the PCI bus.

Impact: Dual address cycle (DAC) feature cannot be sustained by the device while working as PCI target. As PCI initiator, DAC is not supported.

Workaround: For Revision 2.0:

- When operating in host mode, map inbound windows to 32-bit addressing (below 4G addressing space) where this type of application is allowed.
- When operating in agent mode, the above workaround is not valid. The host is mapping the agents according to the address type in configuration registers, which, in this case, is ‘10’. Thus, it could map anywhere in the 64-bit address space. The only option in agent mode is to make sure that all the hosts that will be connected to the MPC8360E PCI agent port will map inbound windows to 32-bit addressing.

For Revision 2.1:

- For agent mode, set bit 11 of the PCI function configuration register (Offset 0x44) to disable 64-bit address support. Setting this bit forces the GPL base address register 1/2[Type] to 0b00 instead of 0b10 when read from the PCI bus with a configuration access. As a result, the agent will be mapped in the lower 32-bit address space.

Fix plan: No plans to fix

PCI20: PCI controller may hang when returning from "PCI pins low" state

Description: Devices: MPC8360E, MPC8358E

When PCI_GCR[PPL] is set, the output and bidirectional PCI bus signals are forced low to allow other PCI bus clients to switch off their power supply, putting the PCI controller into the “PCI pins low” state. When returning to an operational state (clearing the PCI_GCR[PPL] bit), the PCI controller may remain in a PCI “non-idle” state and not be able to perform any further PCI transactions.

The sequence of the failure is as follows:

1. While the PCI bus is inactive, all external bus requests to the PCI arbiter are blocked (by setting PCI_GCR[BBR]), and PCI pins are pulled-low (by setting PCI_GCR[PPL]) to enter “PCI pins low” state.
2. In “PCI pins low” state, the PCI controller state-machine still remains active. It is actively tracking the bus signals and is expecting them to meet the AC timing specifications.
3. When PCI_GCR[PPL] = 0, the PCI control signals start to rise to “1” logic, pulled by the bus pull-up resistors only.
4. The AC timing specifications of the rising signals at this moment may not always be met, causing a timing violation to occur and, in turn, causing the PCI controller to hang.

Impact: The PCI controller may hang when attempting to return from the “PCI pins low” state.

Workaround: Use one of the following options:

- Before Setting PCI_GCR[BBR] and PCI_GCR[PPL] to enter “PCI pins low” state, Clear PCI command bit1 to disable PCI Memory Space. While coming back from “PCI pins low” state, after clearing PCI_GCR[PPL] = 0, read back the PCI_GCR[PPL] to ensure the operation was completed. Immediately afterward, turn off the clocks to the PCI controller (clearing SCCR[PCICM]) to ensure that the PCI controller will stop during the slow rise time of the PCI signals. After waiting up to 10 microseconds, turn the PCI clocks ON and resume the operations to enable the full functionality of PCI bus. Resume the sequence to enable the PCI bus to return to full functionality. At last, Set PCI Command bit1 to enable PCI memory space.
- Connect a wire between an unused GPIOn[x] pin and the FRAME signal. Make sure that this GPIOn[x] is an input while HRESET is active and after its negation (GPnDIR[x] = 0). During the resume procedure, before clearing PCI_GCR[PPL], set the GPIOn[x] data register to '0' (GPnDAT[x] = 0), and then set GPIOn[x] to be an output signal (GPnDIR[x] = 1). Only now, clear PCI_GCR[PPL]. Wait up to 10 microseconds, allowing the PCI signals (except FRAME) to return to a steady state. Set the GPIOn[x] to '1' (GPnDAT[x] = 1), then set GPIOn[x] as an input (GPnDIR[x] = 0). Resume the sequence to enable the PCI bus to return to full functionality.

Fix plan: No plans to fix

PCI21: PCI clock to output valid (t_{PCKHOV}) does not meet the specification

Description: Devices: MPC8360E, MPC8358E (PBGA package only)

PCI clock to output valid does not meet the specification for 66 MHz. The t_{PCKHOV} maximum value is 6.6 ns instead of 6 ns.

Impact: PCI interface may encounter sporadic data corruption.

Workaround: The board design should take into account the timing violation.

Fix plan: Fixed in Rev 2.1

LBC1: UPM does not have indication of completion of a RUN PATTERN special operation

Description: Devices: MPC8360E, MPC8358E

A UPM special operation is initiated by writing to MxMR[OP] and then triggering the special operation by performing a dummy access to the bank.

The UPM is expected to have an indication of when a special operation is completed. The UPM will see MxMR[MAD] increment when a write to or read from UPM array special operation completes. However, the UPM does not have any status indication of completion of the Run Pattern special operation.

The following scenario could be affected by this erratum:

- A UPM Run Pattern special operation is initiated and a second UPM command is issued before the Run Pattern is completed.

If the above scenario occurs the programmed mode registers could be altered according to the second operation and cause the current/first operation to encounter errors due to mode changes in the middle of the operation. Note that if a second command issued is a Run Pattern operation and it does not change the mode registers, the first operation should not encounter errors.

The behavior of the LBIU is unpredictable if the Run Pattern special operation mode is altered between initiation of the operation and the relevant memory controller completing the operation.

Impact: Because of this erratum, when a UPM Run Pattern special operation is to be followed by any other UPM command for which MxMR needs to be changed, the run pattern operation may not be handled properly. Software does not have any means to confirm when the current Run Pattern special operation has completed so that register programming for the next operation can be done safely.

Workaround: None

Fix plan: No plans to fix

LBC-A001: The Local Bus Controller, when configured in DLL enabled mode, may not work after Reset is applied

Description: Devices: MPC8360E, MPC8358E

Due to internal timing when the chip comes out of reset, the Local Bus Controller DLL may not lock and as a consequence the LBC, when configured in DLL enabled mode, will not function properly.

Impact: The LBC may not be functional when configured in DLL enabled mode. Any read access to the LBC will end up with a data time out error indication, and any write access to the LBC will not reach the final target.

Workaround: The following procedure will properly enable the DLL:

1. Put the device in DLL bypass mode (DLL bypass mode is the default state out of reset).
2. Set DLLCR[REG0]. The DLL Control Register (DLLCR) is located at offset 0x1104 from IMMRBAR. REG0 is the field at bit zero of DLLCR and is set when the DLL circuitry has received the reset negation event.
3. Enable the DLL per the Device Reference Manual guidelines:
   a. Clear LCRR[DBYP] to enable the DLL.
   b. Sync.
   c. Wait 1 µs.
   d. Poll the DLLSR[LOCK] bit until it is set.
   e. Access to the local bus devices can now begin.

Fix plan: No plans to fix

QE_USB2: USB receive input signal is inverted

Description: Devices: MPC8360E, MPC8358E

USB controller is not functional because the RXD input signal is inverted.

Impact: USB controller is not functional without external logic.

Workaround: Add an external inverter between the USB PHY and the USB RXD input signal.

Fix plan: Fixed in Rev 2.1

QE_USB5: USB Host might halt in automatic SOF transmission mode

Description: Devices: MPC8360E, MPC8358E

The application software should guarantee that the USB host completes all pending transactions prior to the 1 ms tick so that the transmit FIFO is empty at this point. The current value of the SOF timer and frame number can be read at any time to synchronize the software with the USB frame. Failure to ensure that the FIFO is empty a sufficient time (at least 5% setup time, 50 microseconds, before the end of the 1 msec USB frame) before the end of the frame period may result in the SOF packet not being transmitted (this situation is reported by event USBER[MSF]) or in erratic behavior of the USB controller.

If the software fails to guarantee or can not guarantee that the transmit FIFO is empty before the 1 ms tick, the USB controller might halt and would not complete pending transactions. The USB will not halt on any MSF, but if the MSF event is repeatedly reported by the USB, it is an indication that the above might happen.

Impact: The QE USB controller in host mode, with automatic SOF transmission enabled (USMOD[SFTE] = 1), may experience erratic behavior or the SOF packet may not be transmitted.

Workaround: Use one of the following options:

- Resume operation by executing a USB reset (disable/enable the USB controller using USMOD[EN] with initialization of the USB parameters). This workaround is not recommended if the USB controller is connected to root Hub.
- Use a microcode patch. The microcode patch is intended for use with customer's proprietary USB driver.

Fix plan: No plans to fix

QE_USB6: Reset cannot be driven by the host when USB is using PG11 & PG12 pins

Description: Devices: MPC8360E, MPC8358E

When configured as a Host, the USB does not by definition support driving reset to a device (Single-ended 0). This is usually done by changing the dedicated transmit pins (OE, TP, TN) to function as General Purpose IO's (GPIO's) and then driving these pins to '0'. PG11 and PG12 cannot be configured as GPIO when using PG11 (USB_OE) and/or PG12 (USB_TP) for USB.

Impact: PG11 and/or PG12 parallel IO pins cannot be used for USB host.

Workaround: Use one of the following options:

- Use alternate pin options for these functions:
  - PB2 for USB_OE
  - PB3 for USB_TP

- Use the following settings (this only applies when UCC2 is NOT configured to ATM, BISYNC, Transparent, HDLC, Ethernet GMII/TBI or QMC modes):
  - If UCC2 is not used, configure GUMR[MODE]=1100 (Ethernet mode). Else, if UCC2 is used as UART, Asynchronous HDLC or Ethernet RGMII/RTBI/MII/RMII make sure the UCC mode is configured accordingly before you continue to the next steps.
  - For USB_OE: change PG11 to SEL=01 (CPPAR1G[SEL11]=01), this pin should now drive a constant '0'.
  - For USB_TP: change PG12 to SEL=10 (CPPAR1G[SEL12]=10), this pin should now drive a constant '0'.
  - For USB_TN: This pin doesn't have an issue. Change the pin to general purpose output and drive the pin to '0' by writing to the CPDATG register.
  - After the reset sequence, the PG11 & PG12 pins should be changed back to function as USB dedicated pins.

Fix plan: No plans to fix

QE_HDLC1: UCC does not support HDLC bus mode through the TDM interface

Description: Devices: MPC8360E, MPC8358E

The connectivity of $\overline{\text{CTS}}$ from the NMSI input to the UCC in TSA mode is not enabled.

Impact: HDLC bus mode is not supported through the TDM interface.

Workaround: Program the UCC to NMSI mode and use external logic to generate the gated clock for the UCC at the assigned time slot, see Figure 1 below.

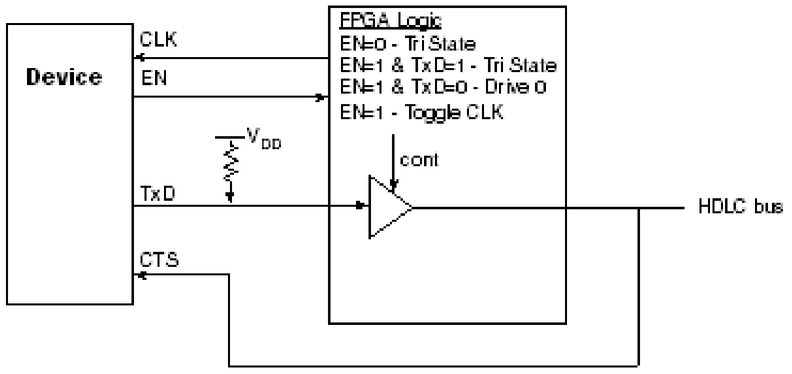

Figure 1. External Logic To Generate Gated Clock for the UCC

Fix plan: No plans to fix

QE_HDLC2: UCC TxD is always set as open drain in HDLC bus NMSI mode

Description: Devices: MPC8360E, MPC8358E

TxD output is always set as open drain in HDLC bus NMSI mode, independent of the parallel I/O configuration, which is the correct setting when there are two or more stations on the HDLC bus. However, it is not required for TxD to be set as open drain when there is only a single station connected to the bus through an external line driver (which implements the open drain capability).

Impact: HDLC bus with external line driver: TxD has to be pulled up even if TxD is not set as open-drain.

Workaround: Place pull-up resistor on TxD signal.

Fix plan: Fixed in Rev 2.1

QE_HDLC3: "Enter Hunt Mode" host command is not functional for UCC in HDLC mode

Description: Devices: MPC8360E, MPC8358E

"Enter Hunt Mode" host command may stall the UCC receiver in the following Fast Protocol modes:

- HDLC
- Transparent

Impact: UCC receiver may halt.

Workaround: Use a microcode patch.

Fix plan: No plans to fix

QE_ENET1: RMON errors

Description: Devices: MPC8360E, MPC8358E

RMON statistics are inaccurate under the following conditions:

- Transmit collision counter status increments after underrun even though no collision has occurred. This bug only shows up when CRC is not appended.
- The MAC (TX module) does not report CRC error on a frame that is transmitted if the previous frame had underrun.
- When BPNB (back pressure no backoff) is enabled the MAC may report a wrong value of collision count (a value that is less than the real collision count).
- If all the following conditions occur, Tx CRC error might not be reported by the Tx MAC:
    - Underrun on last data byte of previous frame
    - No pad or CRC is set on MACCFG2
    - No pad or CRC bit in TxBD is set on current frame (TxBD[TC,PAD/CRC])
    - There is a CRC error on the current frame

Impact: Inaccurate RMON statistics.

Workaround: None

Fix plan: No plans to fix

QE_ENET2: Rx MAC might accept dropped frame

Description: Devices: MPC8360E, MPC8358E

When a dropped frame contains data with 5d pattern, the MAC assumes that it is a new frame and cancels the drop event. The MAC effectively accepts this partial/dropped frame. However, the MAC receive statistic vector (receive previous packet dropped) is asserted to indicate that the received packet is dropped.

Impact: Partial/incomplete frame being received and a false count in the MIBs. However, this partial frame will be rejected by the receiver (due to CRC error).

Workaround: None

Fix plan: No plans to fix

QE_ENET10: RGMII AC values do not meet the specification

Description: Devices: MPC8360E, MPC8358E

RGMII AC values (data to clock output skew at transmitter and data to clock input setup at receiver) do not meet the specification.

For revision 2.0 (TBGA only):

- The $t_{SKRGTKHDX}$ minimum value is -2.3 ns instead of -0.5 ns.
- The $t_{SKRGTKHDV}$ maximum value is as follows (instead of 0.5 ns):
  - 1.0 ns for UCC1
  - 1.2 ns for UCC2 option 1
  - 1.8 ns for UCC2 option 2

For revision 2.1:

- The $t_{SKRGTKHDX}$ minimum value is as follows instead of -0.5 ns:
  - -0.65 ns for UCC2 option 1 (only if 0b1010 is written to bits 24:27 at address IMMRBAR + 0x14AC)
  - -0.9 ns for UCC2 option 2

NOTE

UCC1 meets $t_{SKRGTKHDX}$ minimum value of -0.5 ns in revision 2.1.

- The $t_{SKRGTKHDV}$ maximum value is as follows instead of 0.5 ns:
  - 0.75 ns for UCC1
  - 0.75 ns for UCC2 option 1
  - 0.85 ns for UCC2 option 2

Impact: RGMII interface may not be able to connect to an external PHY.

Workaround: For revision 2.0:

Select RGMII of 10/100 Mbps and delay the transmit clock using the programmable I/O delay as shown in Table 4. The optimal value of the I/O delay depends on the clock delay provided by the PHY and the board trace delay of the transmit clock.

$t_{SKRGTKHDX}$ is not critical for 10/100 Mbps interfaces because the clock frequency is 25 MHz or less.

$t_{SKRGDVKH}$ is not specified for 10/100 Mbps interfaces. Refer to the RGMII standard document.

For revision 2.1:

When using the RGMII interface, delay the transmit data using the programmable I/O delay as shown in Table 4.

For UCC2 option 2, $t_{SKRGTKHDX}$ minimum value can be improved from -0.65 ns to -0.5 ns by adding trace delay of 0.15 ns to the signals: PA19 and PA20. $t_{SKRGTKHDV}$ will not be affected.

Table 4. RGMII Programmable I/O Delay Work Arounds

Silicon Revision	UCC1	UCC2 Option 1	UCC2 Option 2
2.0	Write 0b11 to bits 18:19 at address IMMRBAR + 0x14A8	Write 0b11 to bits 4:5 at address IMMRBAR + 0x14A8	Write 0b11 to bits 16:17 at address IMMRBAR + 0x14AC
2.1	Use default settings	Write 0b1010 to bits 24:27 at address IMMRBAR + 0x14AC	Use default settings

NOTE

The optimal value of the I/O delay depends on the clock delay provided by the PHY and the board trace delay of the transmit clock. The programmable I/O delay provides the following output delay adjustments:

Table 5. Output Delay Adjustments

I/O Delay (programmed value)	Typical Output Delay (ns)
01 (default)	0
00	-0.5
10	+0.5
11	+1.5
NOTE: The values above should be programmed to the following bits at address IMMRBAR + 0x14A8: • (UCC1) bits 18:19 • (UCC2 option 1) bits 4:5 • (UCC2 option 2) bits 16:17	

Fix plan: Improved in Rev 2.1

Hardware specifications updated; the AC timings were improved in Rev 2.1 for all modes (10/100/1000 Kbps).

QE_ENET18: GMII AC values do not meet the specification

Description: Devices: MPC8360E, MPC8358E (TBGA Package Only)

GMII AC values (data to clock output skew at transmitter and data to clock input hold at receiver) do not meet the specification.

For revision 2.0:

- The t_{GTKHDV} maximum value of 5ns and t_{GTKHDX} minimum value of 0.5 ns are not supported when the GTX_CLK is selected.
- The t_{GRDXKH} minimum value is 0.5 ns which is not compliant with the standard.

For revision 2.1:

- The t_{GTKHDV} maximum value of 5ns and t_{GTKHDX} minimum value of 0.5 ns are not supported when the GTX_CLK is selected.
- The t_{GRDXKH} minimum value is 0.2 ns which is not compliant with the standard.

Impact: GMII interface may encounter sporadic data corruption.

Workaround: For revision 2.0:

To resolve the t_{GTKHDV} timing issue, pull in the transmit clock using the programmable I/O delay:

For UCC1: Configure the transmit clock on CE_PC9 to CLK0 instead of GTX_CLK by setting bits 18:19 of CPPAR1C to 0b10 and write 0b10 to bits 18:19 at address IMMRBAR + 0x14A8.

For UCC2: Configure the transmit clock on CE_PC2 to CLK0 instead of GTX_CLK by setting bits 4:5 of CPPAR1C to 0b11 and write 0b10 to bits 4:5 at address IMMRBAR + 0x14A8.

For the t_{GRDXKH} minimum value of 0.5 ns, add a small trace delay (100–200 ps) to the receive signals to guarantee sufficient hold time margin.

For revision 2.1:

To resolve the t_{GTKHDV} timing issue, pull in the transmit clock using the programmable I/O delay:

For UCC1: Configure the transmit clock on CE_PC9 to CLK0 instead of GTX_CLK by setting bits 18:19 of CPPAR1C to 0b10, write 0b11 to bits 18:19 at address IMMRBAR + 0x14A8, and write 0b0000 to bits 20:23 at address IMMRBAR + 0x14AC.

For UCC2: Configure the transmit clock on CE_PC2 to CLK0 instead of GTX_CLK by setting bits 4:5 of CPPAR1C to 0b11, write 0b10 to bits 4:5 at address IMMRBAR + 0x14A8, and write 0b0000 to bits 24:27 at address IMMRBAR + 0x14AC.

For the t_{GRDXKH} minimum value of 0.2 ns, add a small trace delay (100–200 ps) to the receive signals if greater margin is needed.

Fix plan: Improved in Rev 2.1

QE_ENET19: TBI AC values do not meet the specification

Description: Devices: MPC8360E, MPC8358E

TBI AC values (data to clock output skew at transmitter) do not meet the specification.

For revision 2.0:

- The t_{TTKHDX} maximum value is 0.7 ns instead of 1ns (for UCC1 on the TBGA package & UCC1 and UCC2 on the PBGA package).

For revision 2.1:

- The t_{TTKHDX} maximum value of 1ns is not supported when the GTX_CLK is selected.

Impact: TBI interface may encounter sporadic data corruption.

Workaround: For revision 2.0:

To improve the t_{TTKHDX} timing issue, do the following:

For UCC1: Write 0b00 to bits 18:19 at address IMMRBAR + 0x14A8. The tTTKHDX minimum value will be 0.7 ns when either of PB6, PB7, PB9, or PB10 is selected for TxD[4:7]. The minimum value will be 1ns when PC22, PC23, PC24, and PC25 are selected for TxD[4:7].

For UCC2: Write 0b00 to bits 4:5 at address IMMRBAR + 0x14A8. The t_{TTKHDX} minimum value will be 1 ns.

The board design should take into account the output hold timing violations.

For revision 2.1:

To resolve the t_{TTKHDX} timing issue pull in the transmit clock using the programmable I/O delay:

For UCC1: Configure the transmit clock on CE_PC9 to CLK0 instead of GTX_CLK by setting bits 18:19 of CPPAR1C to 0b10, write 0b11 to bits 18:19 at address IMMRBAR + 0x14A8, and write 0b0101 to bits 20:23 at address IMMRBAR + 0x14AC

For UCC2: Configure the transmit clock on CE_PC2 to CLK0 instead of GTX_CLK by setting bits 4:5 of CPPAR1C to 0b11, write 0b11 to bits 4:5 at address IMMRBAR + 0x14A8, and write 0b1010 to bits 24:27 at address IMMRBAR + 0x14AC.

Fix plan: Improved in Rev 2.1

QE_ENET20: UEC may stop transmitting after late collision

Description: Devices: MPC8360E, MPC8358E

When a late collision event coincides with the last byte of a transmitted packet, the UEC's transmitter may halt (TxBD[LC] and UCCE[TXE] will be set). In some cases, if the UEC also reaches the retransmit limit, TxBD[RL] will also be set.

Impact: The UEC (in half-duplex mode) may not self recover from a transmit error.

Workaround: The following steps are required to avoid halting the transmitter during the situation described above:

1. Disable automatic recovery from a transmit error by setting UPSMR[7] (reserved bit). The UEC will stop transmission when an error occurs.
2. Modify InitEnet command parameter (see the " InitEnet Command Parameter" table in the QUICC Engine Block Reference Manual with Protocol Interworking): Initialize CECDR +2 (reserved internal variable) to 0x01.
3. Enable RMON by setting TEMODER[7].
4. Load microcode patch.

Fix plan: No plans to fix

QE_ENET22: Ethernet receiver may become stuck while in a multithreaded configuration

Description: Devices: MPC8360E, MPC8358E

When there is heavy traffic in a multithread configuration, the Ethernet receiver may become stuck. This condition is influenced by both the VFIFO size and the number of threads that are enabled.

Impact: Ethernet receiver may halt.

Workaround: At least one of these measures must be taken to avoid the situation described above:

- Only one Rx thread is enabled.
- The Rx VFIFO size should be equal to 0.5 Kbytes.
- If the Rx VFIFO size is between 0.5 to 1.1 Kbytes, at least 4 threads must be enabled.
- If the Rx VFIFO size is between 1.1 to 1.6 Kbytes, at least 6 threads must be enabled.
- If the Rx VFIFO size is between 1.6 to 2.2 Kbytes, at least 8 threads must be enabled.
- If the Rx VFIFO size is between 2.2 to 4.5 Kbytes, the user must allocate 512 bytes (must be initialized to zero) in the Multiuser RAM. The base address for this area must be 512 bytes aligned. Immediately after the Ethernet Rx INIT command is ended (and before the UCC receiver is enabled), this address (24 bits) must be written to the Rx GPRAM in offsets 0x09–0x0B and 0x11–0x13. The user must write the value "0x3F" to offsets 0x08 and 0x10 . For example, if the base address for this new structure is 0x12_3400, the Rx GPRAM[0x08–0x0b and 0x10–0x13] is equal to 0x3F12_3400. Note: In this case there is no limitation on the number of threads that must be enabled.
- If none of the above measures apply, use a microcode patch.

Fix plan: No plans to fix

QE_ENET24: Magic Packet sequence embedded in partial sequence not recognized

Description: Devices: MPC8360E, MPC8358E

The Ethernet MAC should recognize the Magic Packet sequences as follows: Any Ethernet frame containing a valid Ethernet header (Destination and Source Addresses) and valid FCS (CRC-32) and whose payload includes the specific Magic Packet byte sequence at any offset from the start of data payload. The specific byte sequence comprises an unbroken stream of 102 bytes, the first 6 bytes of which are 0xFFs, followed by 16 copies of the MAC's unique IEEE station address in the normal byte order for Ethernet addresses.

If a complete Magic Packet sequence (including 6 byes of 0xFF) immediately follows a partial Magic Packet sequence, however, the complete sequence will not be recognized and the MAC does not exit Magic Packet mode.

The following are example partial sequences followed by the start of a complete sequence for station address 01_02_03_04_05_06:

- FF_FF_FF_FF_FF_FF_FF_01_02_03_04_05_06_01...

  Seventh byte of 0xFF does not match next expected byte of Magic Packet sequence (01). Pattern search restarts looking for 6 bytes of FF at byte 01.
- FF_FF_FF_FF_FF_FF_01_FF_FF_FF_FF_FF_FF_01_02_03_04_05_06_01...

  First FF byte following 01 does not match Magic Packet sequence. Pattern search restarts looking for 6 bytes of FF at second byte of FF following 01.

The following is an example partial sequence followed by the start of a complete sequence which is erroneously not recognized for station address 01_02_03_04_FF_06:

- FF_FF_FF_FF_FF_FF_01_02_03_04_FF_FF_FF_FF_FF_FF_01_<complete sequence>

  11th byte (0xFF) is seen as the 11 byte of the partial pattern and is not recognized as the start of a complete sequence. Pattern search restarts looking for 6 bytes of 0xFF at 12th byte, but sees only 5.

Impact: The Ethernet controller does not exit Magic Packet mode if the Magic Packet sequence is placed immediately after other frame data that partially matches the Magic Packet sequence.

Workaround: Place 1 byte of data that is not 0xFF and does not match any bytes of Destination Address before the start of the Magic Packet sequence in the frame.

Because the Magic Packet sequence pattern search starts at the third byte after Destination Address, the Magic Packet sequence can be placed at the start of the data payload as long as the second byte of the length/type field follows the above rule.

Fix plan: No plans to fix

QE_ENET25: Malformed Magic Packet triggers Magic Packet exit

Description: Devices: MPC8360E, MPC8358E

The Ethernet MAC should recognize Magic Packet sequences, as follows:

1. Any Ethernet frame containing a valid Ethernet header (Destination and Source Addresses) and valid FCS (CRC-32), and whose payload includes the specific Magic Packet byte sequence at any offset from the start of data payload. The specific byte sequence comprises an unbroken stream of 102 bytes, the first 6 bytes of which are 0xFFs, followed by 16 copies of the MAC's unique IEEE station address in the normal byte order for Ethernet addresses.
2. Once the Ethernet MAC has recognized a valid Destination Address for one frame, it continues searching for valid 102-byte Magic Packet sequences through multiple frames without checking for a valid Destination Address on each frame. The only events that cause the MAC to go back to check for valid Destination Address before checking for a Magic Packet sequence on new frames are:
    - A frame containing a recognized full Magic Packet sequence (with valid or invalid FCS)
    - Software disable of Magic packet mode (MACCFG2[MPE] = 0)
    - Perform soft reset via MACCFG1[Rx_EN] and MACCFG1[Tx_EN].

Impact: The Ethernet controller may exit Magic Packet mode if it receives a frame with Destination Address not matching station address, or invalid unicast or broadcast address in a valid Magic Packet sequence for the device.

Workaround: None

Fix plan: No plans to fix

QE_ENET-A002: False broadcast indication in RGMII mode

Description: **Devices**: MPC8360E, MPC8358E

In RGMII mode, the Ethernet controller may erroneously indicate a broadcast status on a frame after an in-band CRS event.

Impact: Ethernet controller may erroneously indicate broadcast at the previous frame (wrong RxBD[BC] indication).

Workaround: Ignore the broadcast (BC) indication bit in the RxBD and check the broadcast address using software.

Fix plan: No plans to fix

QE_L2_SWITCH2: Invalid memory access

Description:	Devices: MPC8360E, MPC8358E The first 64 bytes of a frame might be stored in memory in little endian byte ordering or might be stored on the wrong bus.
Impact:	Data might be transmitted byte swapped or other erratic behavior might occur.
Workaround:	Use a microcode patch.
Fix plan:	No plans to fix

QE_L2_SWITCH3: Error may occur for any TCBD

Description: Devices: MPC8360E, MPC8358E

Error may occur for any Tx CPU BD (TCBD).

Impact: L2 Ethernet Switch erratic behaviour may occur if TCBD are used.

Workaround: Use a microcode patch.

Fix plan: No plans to fix

QE_MCC4: MCC INIT host commands may cause GUN event

Description: Devices: MPC8360E

MCC INIT host commands (which contain transmit initialization) may not be executed as expected for a particular channel if executed after MCC was enabled and 16-bit superchannels are used. When initializing several channels in one INIT command, this error may affect the first channel that is initialized in the series.

The INIT host commands are expected to pre-load the internal channel's FIFO with data to a sufficient level that will allow transmission to start. If the INIT is not performed, the channel remains with the previous data that may be insufficient for the current configuration. The old data may lead to data corruption and its insufficient amount may lead to a GUN event.

For example, if an 8-bit channel is modified to a 16-bit superchannel, a GUN event will occur.

Impact: This failure can lead to data corruption and GUN event on the first channel being initialized. The failure will cause data corruption for the initialized channel.

Workaround: Use a microcode patch for the INIT command.

Fix plan: No plans to fix

QE_ATM9: ATM policer is not functional

Description: Devices: MPC8360E, MPC8358E

ATM policer is not functional.

Impact: Memory corruption, false PM table updates, and wrong memory accesses by the QUICC Engine block can occur and cause AAL0 traffic to operate incorrectly.

Workaround: Use a microcode patch.

Fix plan: Fixed in Rev 2.1

QE_ATM10: AAL5 transmitter could stop transmission

Description: Devices: MPC8360E, MPC8358E

When working in ATM using hierarchical scheduling, the parent channel (V-TCT) may stop transmission.

Impact: ATM channel may stop transmission.

Workaround: Use a microcode patch.

Fix plan: Fixed in Rev 2.1

QE_ATM11: IDLE state can halt ATM transmitter operation

Description: Devices: MPC8360E, MPC8358E

When the device is in an IDLE state, the ATM transmitter operation can cease.

Impact: The ATM transmitter may cease operation.

Workaround: Use a microcode patch.

Fix plan: Fixed in Rev 2.1

QE_ATM12: Cell duplication can occur when during ATM AAL2 operation on IMA

Description: Devices: MPC8360E, MPC8358E

Cell duplication can occur when using more than four PHYs during ATM AAL2 operation on IMA.

Impact: AAL2 cell duplication on IMA traffic.

Workaround: Use a microcode patch.

Fix plan: Fixed in Rev 2.1

QE_ATM-A001: ATM APC flux compensation mode is not functional

Description: APC flux compensation mode is not functional.

Impact: When APC flux compensation mode is used, half of the time frames are not transmitted.

Workaround: Use regular APC or GCRA schedulers, and disable AFC mode in ATM APC.

Fix plan: No plans to fix

QE_UPC2: User-defined cell Rx extra header size limited to 8 bytes

Description: Devices: MPC8360E, MPC8358E

User-defined cells in Rx slave mode with extra header size of 9–12 bytes may cause data corruption.

Impact: The maximum extra header size UPCDx[REHS] is limited to eight bytes.

Workaround: None

Fix plan: No plans to fix

QE_UPC3: UPC AC values do not meet the specification

Description: Devices: MPC8360E, MPC8358E

UPC AC values (external clock input setup and hold) do not meet the specification:

- The t_{UEIVKH} minimum value is 4.3 ns instead of 4 ns.
- UPC2: The t_{UEIXKH} minimum value is 1.4 ns instead of 1 ns when $\overline{\text{PCI\_RESET\_OUT}}$/CE_PF6 is selected for RxAddr[1]/RADR[1].

Impact: The UPC interface may encounter sporadic data corruption.

Workaround: For t_{UEIVKH} minimum value:

- None.

For t_{UEIXKH} minimum value:

- Select PA8 for UPC2:RxAddr[1]/RADR[1] functionality to guarantee 1 ns timing.

Fix plan: Package specific

TBGA Package: Fixed in Rev 2.1

PBGA Package: For revision 2.1, t_{UEIVKH} is fixed and t_{UEIXKH} was updated in the MPC8358E Hardware Specfication.

QE_UPC-A001: UPC may hang or receive corrupted data in receive buffer when RVAL is tri-stated

Description: Devices: MPC8360E, MPC8358E

According to the PL2 specification, the PHY Layer device (QE POS Slave) shall tri-state REOP, RSOP and RVAL signals when RENB is de-asserted. The QE POS controller (UPC) does not ignore these tri-state signals when RENB is de-asserted as it should. When the RVAL signal is tri-stated it may be sampled as active by the UPC which will result in an error. In this case, the UPC will signal a protocol violation event by setting UCCE[PV].

Impact: Corrupted data may be received in the receive buffer or the UPC peripheral may hang.

Workaround: The customer system should drive RVAL low within one POS interface clock after RENB negation (a high value). This can be achieved, for example, by connecting the inverted, one clock delayed sample of RENB (Master output) to the device RVAL pin (Master input), instead of the POS Slave RVAL output signal as illustrated in the following figure.

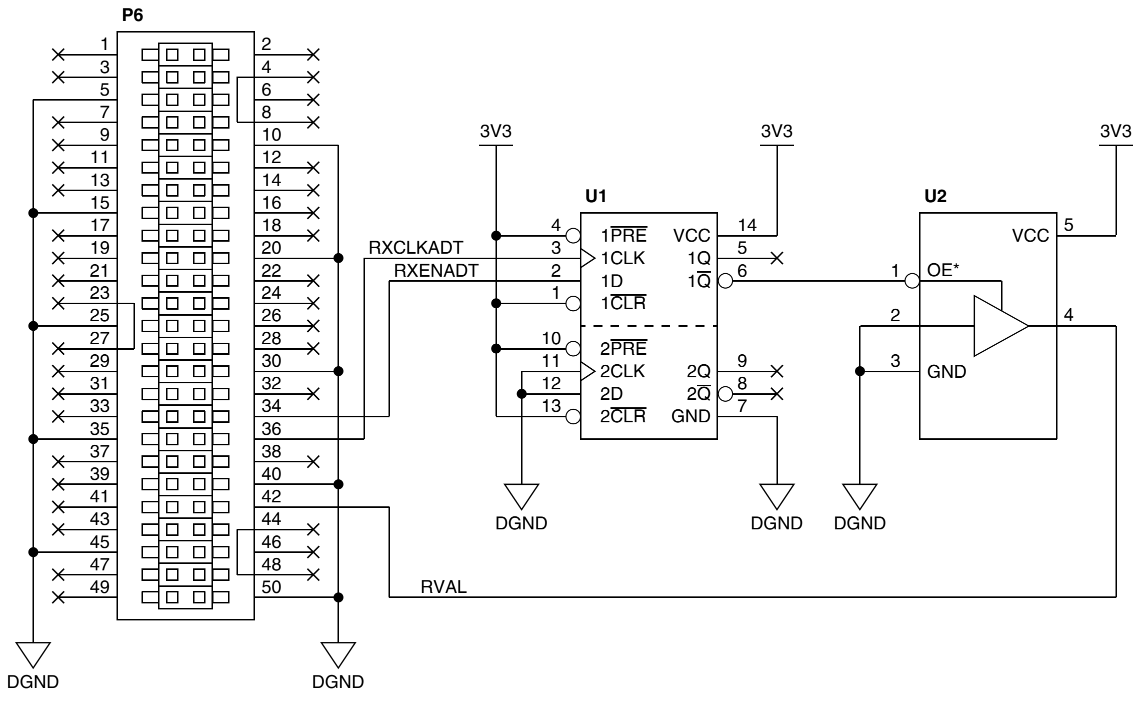

Figure 2. Example Schematic Drawing

Fix plan: No plans to fix

QE_UART1: Simultaneous loopback and echo mode is not functional

Description: Devices: MPC8360E, MPC8358E

Simultaneous loopback and echo diagnostic mode (GUMR_L[DIAG] = 11) is not functional.

Impact: Issue only in diagnostic mode.

Workaround: Use one of the following options:

- Test echo mode and loopback mode separately.
- Use a microcode patch.

Fix plan: No plans to fix

QE_UART2: Overrun indication is not reported on the current BD

Description: Devices: MPC8360E, MPC8358E

The overrun indication is not reported on the current Rx buffer descriptor but on first buffer descriptor related to the received data after the overrun condition ended.

Impact: The user will not have any knowledge about the overrun event until the event ends and data is received. Not backward compatible with the PowerQUICC II family.

Workaround: Use a microcode patch.

Fix plan: No plans to fix

QE_UART3: DRT mode (UPSMR[DRT] = 1) is not functional

Description: Devices: MPC8360E, MPC8358E

Disable receive while in transmitting mode (UPSMR[DRT] = 1) is not functional.

Impact: DRT mode cannot be used.

Workaround: None

Fix plan: No plans to fix

QE_UART5: Rx FIFO data can be lost when ENTER HUNT MODE command is executed

Description: Devices: MPC8360E, MPC8358E

RX ENTER HUNT MODE host command is not functional.

Impact: Data may be lost in the UART receive FIFO.

Workaround: Use a microcode patch.

Fix plan: Fixed in Rev 2.1

QE_UART6: UART/AHDLC receive data corruption on QUICC Engine UCC

Description: Devices: MPC8360E, MPC8358E

The QE UCC hardware may experience synchronization problems when it is configured for UART or Asynchronous HDLC in both normal operation and multidrop operation. Framing errors and/or CRC errors may be reported in the receive buffer descriptors (BDs) when synchronization issues occur. It is also possible for received data to be corrupted.

Impact: Data might be received incorrectly leading to data integrity errors, e.g. framing errors (for UART) or CRC errors (for AHDLC).

Workaround: Use one of the following options:

- Use DUART controller (for UART), if possible
- Use the microcode patch found in QERAMPKG. The microcode patch disables UART/AHDLC mode, and instead collects all the data using Transparent mode. By over-sampling, the microcode processes the data based on either the UART or AHDLC protocol that has been configured. There are some limits due to this software solution. Please see QERAM Microcode Release Note for details.
- Sample the recieve data signal before it reaches the device and connect the sampled version of the receive data signal to the device instead of the original signal. The clock that should be used for this sampling of receive data should be the one that is used by the UCC’s UART hardware (either the external clock or BRGO clock that is driven to an external pin and used for sampling receive data). The negative edge of the clock should be used to sample the receive data.

Fix plan: No plans to fix

QE_UART7: UART does not set UCCE[AB] correctly for autobaud

Description: Devices: MPC8360E, MPC8358E

The QE UCC hardware does not set UCCE[AB] correctly when it is configured to be UART for the autobaud operation. The autobaud control function detects the baud rate on the corresponding RXDx input. The control function then writes the detected divide ratio to BRGCx[CD] and BRGCx[DIV16]. UCCE[AB] should be set at the same time so that the software (interrupt server routine (ISR)) can adjust the ratio to be an exact value. However, UCCE[AB] is not set due to this erratum.

Impact: The UCCE[AB] bit cannot be used for checking the autobaud interrupt.

Workaround: The following steps are required to ensure that the UCC operates correctly during the situation described above:

1. Set UCCM[Rx] and UCCM[AB].
2. Set MRBLR to 1 (BD contains 1 char) on system initialization.
   - Receive data is driven (AT commands) and triggers autobaud mechanism that sets BRGCx[CD] and BRGCx[DIV16], receive data is received to data buffer (one char).
   - UCCE[Rx] is set and an interrupt is issued on the received data character. This bit can be used instead of UCCE[AB], because it does not work due to this erratum.
3. Configure as follows for the interrupt service routine (ISR):
   a. Use the first receive(Rx) ISR to process the autobaud (AB) interrupt request. This can be done by setting BRGCx[CD] and BRGCx[DIV16], if necessary.
   b. Re-configure MRBLR to the required value, so that later, the UCCE[Rx] bit can be used for a regular receive data interrupt.

Fix plan: No plans to fix

QE_UART-A001: QE UART controller idle status bit may not be reliable

Description: Devices: MPC8360E, MPC8358E

The idle status bit in the UCC UART status register, UCCS[ID], may not always reflect the actual status of the line.

Impact: The idle status bit cannot be used by software to monitor the line state.

Workaround: None. Software should not use the UART Idle status bit.

Fix plan: No plans to fix

QE_SI5: Disabling the TDM interface on-the-fly may cause an MCC GUN event

Description: Devices: MPC8360E, MPC8358E

An MCC GUN event may occur if the TDM interface is disabled on-the-fly using the SI's global mode register.

Disabling a TDM interface during a transition between a time slot which is routed to the MCC (an actual channel transmitting data) and an unrouted time slot (unused time slot) or UCC slot, may lead to an MCC GUN event.

Impact: MCC GUN events may occur if TDM interface is disabled on-the-fly.

Workaround: The two proposed workarounds ensure that the TDM frame has ended before being disabled.

- TDM Sync Disable Option

  The TDM sync signal is disconnected from the Time Slot Assigner by programming the corresponding parallel I/O pin. By doing so, the current TDM frame is allowed to end before the TDM interface is disabled. By disabling the TDM interface when the MCC hardware is inactive, the false data request and the subsequent GUN situation is avoided.

  The TDM Sync Disable workaround procedure consists of the following steps:

  a. Disable the TDM sync input from GPIO: The TDM sync disable can be done by programming the CPPARx[SELn] to 0b00 and CPDIRx[DIRn] to 0b01. For example, if using PA16 as TxSYNC for TDMA, programming CPPAR2A[SEL16] = 0b00 and CPDIR2A[DIR16] = 0b01 will disable the TxSYNC for TDMA.
  b. Wait the equivalent time of 2 TDM frame length or more per activated TDM.
  c. Disable the TDM by clearing the SIGLMRx enable bit corresponding to the disabled TDM (SIGLMRx[ENx] = 0b0). For example, disabling TDMA is done by clearing SIGLMRH[ENA]. Before the TDM is enabled, the GPIO/CE_MUX should be configured to enable TDM sync input.

- Neutralized MCC Option

  The SIRAM for the corresponding TDM interface is programmed to a short, unrouted frame. The short frame will have no transition points between routed and unrouted time slots and will allow for the disabling of the TDM without the GUN event.

  The Neutralize MCC workaround procedure is comprised of the following steps:

  a. Check that the command register is cleared for this TDM (SICMDRH/L[CSRTx]) so there is no pending switch command.
  b. Write to the first two lines of the current transmit SIRAM unrouted entries. The current SIRAM entry being used can be identified by reading SISTRH/L[CROTx], and the initial address is the first address of the SI RAM that is being used. For example, on the first line 0x0002_0002 and on the second line 0x0003_XXXX.
  c. Wait the equivalent time of 2 TDM frame length or more per activated TDM.
  d. Read SI RAM counter (SITxRC) until it is equal to the "INITIAL ADDRESS +1".
  e. Read SI RAM counter (SITxRC) until it is equal to the "INITIAL ADDRESS +0".
  f. Read SI RAM counter (SITxRC) until it is equal to the "INITIAL ADDRESS +1".
  g. Disable the TDM by clearing the SIGLMRx enable bit corresponding to the disabled TDM (SIGLMRx[ENx] = 0).

The SI hardware has counters that indicate the position of the transmitter in the transmitted frame. Depending on the workaround used, there are certain values that indicate the frame has ended and it is safe to issue DISABLE. Due to pipeline mechanisms in the hardware, waiting for those counters to indicate "First Line" is not always enough.

In this case, if the last entry is routed and first entry is unrouted, DISABLE may be asserted at exactly the moment of transition, which leads to a GUN event. The “Neutralize MCC” procedure contains the proper counter checks needed to guarantee that the TDM can be disabled safely.

Fix plan: No plans to fix

QE_BISYNC2: RxBD in continuous mode is not closed after an Rx parity error

Description: Devices: MPC8360E, MPC8358E

RxBD in continuous mode (CM is set in the RxBD) is not closed after an Rx parity error. After receiving a parity error, the controller sets the PR status bit in the RxBD, but does not clear the Empty control bit as expected.

Impact: The receiver does not work as expected in continuous mode when an Rx parity error occurs.

Workaround: When using CM in the receiver, the software should poll PR status during the routine handling of RxBD.

Fix plan: No plans to fix

QE_BISYNC3: BISYNC controller does not enter HUNT MODE

Description: Devices: MPC8360E, MPC8358E

The BISYNC controller should revert to hunt mode (which means the current opened Receive BD is closed, no additional bytes will be written to the current data buffer, and the BISYNC controller will search for the SYN1/SYN2 sequence, after which reception continues using the next BD) when it receives an ENTER HUNT MODE command, upon an error condition, or after reception of a pre-defined control character (RCCM entry H bit is set). Instead, if the BISYNC controller was in a state of receiving a message, it continues the reception of characters.

Impact: The BISYNC controller keeps transferring data to the receive buffers and does not enter hunt mode.

Workaround: Use a microcode patch.

Fix plan: No plans to fix

QE_IEEE1588_1: Out of band detection of PTP frames is not functional on UCC1 and UCC2

Description: Devices: MPC8360E, MPC8358E

When UCC1 or UCC2 are set for PTP and a PTP frame is received, the RxBD[PTP] bit in the BD and the PTP event are not set.

Impact: Software can not detect PTP frames by polling the RxBD[PT] bit in the BD when UCC1 or UCC2 are used for PTP.

Workaround: Use in-band PTP frame detection or use UCC3–8.

Fix plan: Fixed in Rev 2.1

QE_SPI1: SPI output hold timing (t_{NIKHOX}) does not meet the specification

Description: Devices: MPC8360E, MPC8358E

SPI output hold timing (data to output clock at transmitter) does not meet the specification. The tNIKHOX minimum value is -0.55 ns instead of 0.5 ns.

Impact: SPI interface may encounter sporadic data corruption.

Workaround: In order to improve tNIKHOX to 0ns, write 0b00 to bits 12:13 at address IMMRBAR + 0x14AC. The board design should take into account the output hold timing violation.

Fix plan: Hardware Specfication updated for revision 2.1

QE_QMC1: Corruption of interrupt queue entries

Description: Devices: MPC8360E, MPC8358E

QMC interrupt queue entries may be corrupted when two RISCs are operating concurrently, one working on UCC Rx and the other working on UCC Tx, and both want to generate an interrupt. Both RISCs could change the same internal registers used for preparing the interrupt event entry.

Impact: QMC interrupt queue entries may be corrupted.

Workaround: Use a microcode patch.

Fix plan: Fixed in Rev 2.1

QE_QMC2: Data may be shifted if QMC is disabled and enabled during reinitialization or reconfiguration

Description: Devices: MPC8360E, MPC8358E

If the QMC transmitter is disabled and enabled through GUMR_L[ENT] during a reconfiguration or reinitialization procedure, data may be transmitted in the wrong timeslot.

Impact: Data may be shifted up to three timeslots if the QMC is disabled and enabled during a reconfiguration or reinitialization procedure.

Workaround: Before enabling the QMC transmit controller (before GUMR_L[ENT] is set), always initialize the UCC transmit startup ROM address to 0x80 with the PushSched host command and the UCC receive startup ROM address to 0x82 with the PushSched host command.

See the “Serial Number (SNUM)” and “QUICC Engine Commands” section of the *QUICC Engine Block Reference Manual with Protocol Interworking*.

Fix plan: No plans to fix

QE_QMC3: Missing Interrupt after Stop Rx Host command

Description: Devices: MPC8360E, MPC8358E

When a Stop Rx Host Command is issued, channel specific RXB and RXF interrupts may not be generated for data written to buffer prior to the command. The receive BD status indications are not affected.

Impact: RXB and RXF interrupts may be missing after Stop Rx Host Command.

Workaround: After issuing Stop Rx Host Command to a specific channel, RxBD[E] should be polled for the indication of data that was written to Rx buffer.

Fix plan: No plans to fix

QE_QMC-A001: QMC GOV event is not reported in the UCC event register

Description: Devices: MPC8360E, MPC8358E

GOV events are not reported in the UCC event register.

Impact: UCCE[GOV] is not set and an interrupt is not issued on QMC GOV event.

Workaround: Use a microcode patch.

Fix plan: No plans to fix.

QE_Transparent1: The CRC error is not checked in transparent mode if $\overline{\text{CD}}$ is de-asserted

Description: Devices: MPC8360E, MPC8358E

When operating the UCC transparent protocol in the following modes, the Carrier Detect ($\overline{\text{CD}}$) lost bit is always set when the $\overline{\text{CD}}$ signal is de-asserted at the end of the packet:

- $\overline{\text{CD}}$ envelope mode: GUMR[CDP] = 0
- External sync mode: GUMR[SYNL] = 00

However, when the QUICC Engine detects and reports the $\overline{\text{CD}}$ loss, it no longer checks for CRC error.

Impact: The QUICC Engine might receive a bad packet with wrong CRC, but RxBD[CR] will not be set by the QUICC Engine to indicate the CRC error.

Workaround: Use a microcode patch.

Fix plan: No plans to fix

QE_Transparent2: QE may erroneously report CRC error if GUMR[REVD] is set to 1 in transparent mode

Description: Devices: MPC8360E, MPC8358E

In UCC transparent mode, if GUMR[REVD] is set to 1, the receiver may calculate a wrong CRC for a packet.

Impact: Due to wrong CRC calculation, the QUICC Engine block may report a CRC error for a correct packet.

Workaround: Use a microcode patch.

Fix plan: No plans to fix

QE_General4: Potential spikes on BRG output clock when using odd divisions

Description: Devices: MPC8360E, MPC8358E

When the BRG is dividing the clock using an odd division factor greater than 3, there are potentially spikes on the BRG output clock.

- Clock Division Factor = BRGCx[CD] + 1

Impact: BRG division factor cannot be odd (greater than 3).

Workaround: Use one of the following options:

- Use an even division factor, i.e. (BRGCx[CD] + 1) is divisible by 2
- Use an odd division factor only in DIV16 mode

Fix plan: Fixed in Rev 2.1

QE_General-A003: High Tx Virtual FIFO threshold size can cause UCC to halt

Description: Devices: MPC8360E, MPC8358E

Due to the structure of Tx Virtual FIFO, it is possible for the Tx Virtual FIFO to contain part of a frame when there is no room for the remainder of the frame (i.e. when the frame size is on the scale of magnitude of the Tx Virtual FIFO size).

If the Tx Virtual FIFO contains a partial frame, the transmission of the frame may start even if fewer than UTFTT (UCC Tx Virtual FIFO Threshold) bytes of data reside in the Tx Virtual FIFO. The only reason the frame will not start transmission is if it meets a specific sequence in which the UCC transmit-start condition is not met for the frame. This is rare but possible.

The erroneous behavior is a result of incorrect recovery after the pausing data-retrieve phase from the BD buffer to the Tx Virtual FIFO in a certain internal stage.

Reaching the UTFTT watermark in the Tx Virtual FIFO for each transmit frame prevents this sequence. UTFTT has an upper limit value, which is less than the UTFS (UCC Tx Virtual FIFO Size). For each UTFS value there is a maximum value of UTFTT that corresponds to it. Setting UTFTT higher than the suggested value can result in a UCC halt, and Ethernet transmission stops.

NOTE

1. As defined in the reference manual, UTFTT refers to the payload of a single Ethernet frame.
2. Transmission may start before surpassing the threshold if there is no space for additional data in Tx Virtual FIFO. This is the correct behavior.

Impact: The UCC may stop transmitting when a large frame is partially stored in the Tx Virtual FIFO before the transmission of the frame starts.

Note that the Tx BD ring buffer configuration (data allocation to BD buffers) may adversely affect Tx Virtual FIFO utilization, thus preventing the Tx Virtual FIFO from reaching the threshold for transmission.

Workaround: Use one of the following options:

- To recover from a possible failure, the application code should configure UTFTT to 0x40 and then set it back to the original value. This sequence triggers transmission from the point it stopped. Detection of a UCC halt can be done by monitoring Tx BD ring vacancy. A full Tx BD ring may indicate the UCC has halted.
- To prevent the failure, configure UTFS and UTFTT so that it is possible to store UTFTT data (payload) bytes in Tx VFIFO.
  a. For the usage of a single buffer per frame, here are the maximum UTFTT values that can be used:
     - Ethernet Controller: Set UTFTT < [(0.9375 x UTFS) – 128]
     - HDLC Controller: Set UTFTT < [(0.7500 x UTFS) – 32]
     - For example, for an Ethernet controller with UTFS = 1024, set UTFTT < 832.

  b. For the usage of a multiple buffer per frame, here are the maximum UTFTT values that can be used: (M is the smallest buffer size used)
     - Ethernet Controller: Set UTFTT < [(UTFS x (M – 8) ÷ M) – 128]

- HDLC Controller: Set UTFTT < [(UTFS x (M – 8) ÷ M) – 32]
- For example, for an Ethernet Controller with UTFS = 1024 and M = 64, set UTFTT < 768.

Fix plan: No plans to fix

DMA2: Data corruption by DMA when destination address hold (DAHE) bit is used

Description: Devices: MPC8360E, MPC8358E

There can be corruption of the DMA data under the following conditions:

- DMAMR[DAHE] = 1 (destination address hold)
- DMAMR[DAHTS] = 10 (4 bytes) or 11 (8 bytes)
- DMA source address is not aligned to the transaction size specified by DAHTS
- The source port width is smaller than the destination transaction size or the source port returns valid read data only in the valid byte lanes

Examples of error condition are as follows:

- DAHTS is 8 bytes and the source port is a 32-bit PCI bus
- The source memory space is on the PCI bus and is not prefetchable

Impact: Corrupted data written to the destination peripheral or memory.

Workaround: Use one of the following options:

- Use a source address aligned to the destination transaction size
- Do not access any DMA registers while this type of DMA transfer is active

Fix plan: No plans to fix

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
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: MPC8360ECE
Rev. 5
09/2011