PCI 9056BA

Revision 1.5
March 2006

Errata Documentation

A. Affected Silicon Revision

This document details errata in the following silicon:

Product	Part Number	Description	Status
PCI 9056BA	PCI9056-BA66BI	32-bit 66MHz PCI, 32-bit 66MHz Local, PBGA Package	Production Released

B. Silicon Documentation

The following documentation is the baseline functional description of the silicon. Errata are defined as behaviors in the silicon that do not match behaviors detailed in the data book.

Document	Revision	Description	Publication Date
PCI 9056BA Data Book	1.1	Silicon Data Book	October 2003
PCI 9056BA Data Book Addendum	1.0	Data Book Addendum	August 2005

C. Errata Documentation Revision History

Revision	Description
1.0	Baseline. Errata #1 to #11.
1.1	Added Errata #12, 13.
1.2	Updated Errata #8 and #11 workarounds, retracted Errata #12.
1.3	Clarified Errata #10.
1.4	Added Errata #14.
1.5	Added Errata #15.

D. Errata Summary

#	Description	C	J	M
1	BI# Assertion during Scatter/Gather DMA Descriptor Load from 8-/16-bit Local Bus			✓
2	Direct Slave Dummy Single Cycle Read of M-mode 16-bit Local Bus Memory			✓
3	Scatter/Gather DMA Clear Count Mode with Both Channels' Descriptors in PCI Memory	✓	✓	✓
4	MPC860 Processor Local Bus TSIZ[0:1] Values when C/BE# Values Change			✓
5	EOT# Assertion at Start of Local-to-PCI Non-Qword/Lword-Aligned DMA			✓
6	Direct Master Write and Back-to-Back Non-Contiguous Address Direct Master Read	✓		✓
7	EOT# Assertion at End of Local-to-PCI DMA with Unaligned Last Address			✓
8	PME# Pin Isolation	✓	✓	✓
9	Power Management PMDATA Register	✓	✓	✓

#	Description	C	J	M
10	JTAG IDCODE Instruction	✓	✓	✓
11	D3cold PME# Generation	✓	✓	✓
~~12~~	(retracted)			
13	DMA Constant Local Address	✓	✓	✓
14	Direct Master Burst Read with non-contiguous LBE[3:0]# inputs and Deadlock with BREQo output assertion	✓	✓	
15	USERi Input Low at RST# De-assertion Might Allow Initial PCI Configuration Access to the Silicon to Fail	✓	✓	✓

E. Errata Details

1. BI# Assertion during Scatter/Gather DMA Descriptor Load from 8-/16-bit Local Bus

Note. This is only an erratum for devices that use the MPC860 Processor/Local Bus Mode (M Mode). It is not an erratum for generic Processor/Local Bus Mode designs (C/J Mode).

Documented Behavior: The silicon supports the Burst Inhibit (BI#) input signal/pin and feature to break up a burst transaction into multiple single cycle accesses when loading Scatter/Gather descriptors from Local memory, regardless of the Local bus width.

Actual Behavior: If the BI# input signal is asserted while the silicon is attempting to read a Scatter/Gather descriptor from either a 8-bit or 16-bit Local memory location, the silicon will assert incorrect values on the TSIZ[0:1] signals/pins for the transfer following the assertion of BI#.

Workarounds (do one of the following):

1. Store Scatter/Gather descriptors in burst-able memory and keep BI# negated.
2. Store Scatter/Gather descriptors in PCI memory.

2. Direct Slave Dummy Single Cycle Read of M-mode 16-bit Local Bus Memory

Note. This is only an erratum for devices that use the MPC860 Processor/Local Bus Mode (M Mode). It is not an erratum for generic Processor/Local Bus Mode designs (C/J Mode).

Documented Behavior: The silicon supports Dummy Direct Slave Read cycles (PCI reads of the silicon where all of the PCI byte enables are negated) from Local memory regardless of the Local bus width. When the silicon detects a single cycle Dummy Direct Slave Read to a 16-bit device, it will initiate a read transfer on the local bus and drive the TSIZ[0:1] signals/pins to binary 11. Binary 11 to a 16-bit device indicates that the transfer size is undetermined.

Actual Behavior: A Direct Slave Single Cycle Dummy Read of a 16-bit wide local bus device will result in the silicon asserting a TSIZ[0:1] value of binary 00 rather than binary 11. Because a value of binary 00 is not defined, the targeted Local slave device may or may not properly respond to supply read data and/or adhere to the M-mode local bus protocol.

Workarounds (do one of the following):

1. A work around may not be necessary, since the original request was for a dummy read cycle on the PCI bus. No data corruption will occur, because no data will actually be transferred on the PCI bus during a dummy read cycle. Whether or not the system fails would depend upon how the Local Slave Device responds when the silicon attempts to fetch Local bus data using TSIZ[0:1] = binary 00. If the Local Slave Device supplies the data and properly adheres to the local bus protocol, no system failure will occur.
2. Avoid Dummy Read Cycles to 16-bit local Slave devices that will become confused if TSIZ[0:1] = binary 00.

3. Scatter/Gather DMA Clear Count Mode with Both Channels' Descriptors in PCI Memory

Documented Behavior: The silicon supports internal requests for the PCI bus from both DMA channels individually or simultaneously to perform a Clear Count Mode access if DMAMODE0/1[16] = 1 for Scatter/Gather descriptors stored in PCI memory. If EOT# is not used, the value of zero is to be written back into the Transfer Size field of the Scatter/Gather descriptor upon completion of the DMA transfer for that descriptor. If EOT# is used, upon its assertion the effected channel(s) are to write the remaining number of bytes that were to be transferred into the Transfer Size field within their respective Scatter/Gather descriptors.

Actual Behavior: If EOT# is not used and the Scatter/Gather DMA Clear Count Mode feature is enabled (DMAMODE0/1[16] = 1) for both DMA channels and both of the channels Scatter/Gather descriptors reside in PCI memory and both channels are internally arbitrating simultaneously for the PCI bus to perform clear count write-back, the silicon will incorrectly write a non-zero value to the Transfer Size field location in the Scatter/Gather descriptor that won the arbitration. Instead of writing a value of zero into the Transfer Size field in the Scatter/Gather descriptor, the silicon will write the PCI address of where the Transfer Size field within the Scatter/Gather descriptor resides.

If EOT# is used and asserted, the silicon will write the PCI address of where the Transfer Size field is within the Scatter/Gather descriptor, rather than the remaining number of bytes that were to be transferred.

Workarounds (do one of the following):

1. If EOT# is not used, upon reading the Transfer Size field of the Scatter/Gather Descriptor (to determine if that descriptor's transfer has been completed), treat a value that is other than the original byte count as zero. To use this work around, the PCI Address of the location of where the Scatter/Gather Descriptors DMA Transfer field resides must not be equal to the original byte count of that descriptor.

2. Put the descriptors for at least one of the DMA channels in Local Memory.

4. MPC860 Processor Local Bus TSIZ[0:1] Values when C/BE# Values Change

Note 1. This is only an erratum for devices that use the MPC860 Processor/Local Bus Mode (M Mode). It is not an erratum for generic Processor/Local Bus Mode designs (C/J Mode).

Note 2. This erratum is only an issue for Single Cycle Direct Slave Delayed Reads targeted to Local Address Spaces configured to 8-bit or 16-bit Local Bus widths.

Documented Behavior: The silicon supports a PCI Single Cycle Direct Slave Delayed Read (MARBR[24] = 1) followed by a Direct Slave Write with the Direct Slave Read Write Flush Mode enabled (MARBR[26] = 1), followed by another PCI Single Cycle Direct Slave Delayed read which is the same or different than the first read, from Local memory regardless of the Local bus width.

Actual Behavior: If MARBR[26:24] = 101b and a PCI Single Cycle Direct Slave Delayed Read is followed by a Direct Slave Write which is then followed by another PCI Single Cycle Direct Slave Delayed read which has different C/BE[3:0]# signals asserted than for the first PCI Single Cycle Direct Slave Delayed Read, the silicon will assert an incorrect TSIZ[0:1] transfer size value onto the Local bus when reading data for the first Single Cycle Direct Slave Delayed Read if the second read on the PCI interface is coincident with the first read access on the Local Bus. The TSIZ[0:1] value for the second read ends up being used for the first delayed read.

Workarounds (do one of the following):

1. For 32-bit Local Bus devices this is not an issue, since any TSIZ[0:1] value is legal, and the data for the first read will get flushed properly when the Direct Slave Write is detected, if MARBR[26] = 1.
2. For 8-bit or 16-bit Local Bus devices, ensure that C/BE[3:0]# = ‘h0 for each read. This will ensure that a legal TSIZ[0:1] value is asserted for each read (reads might be to different devices). If MARBR[26] = 1, the data read from the local bus for the first read will get flushed properly out of the FIFO when the Direct Slave Write is detected.
3. Enable PCI Read No Write Mode (MARBR[25:24] = 11b) to retry writes while a delayed read is pending completion.

5. EOT# Assertion at Start of Local-to-PCI Non-Qword/Lword-Aligned DMA

Note. This is only an erratum for devices that use the MPC860 Processor/Local Bus Mode (M Mode). It is not an erratum for generic Processor/Local Bus Mode designs (C/J Mode).

Documented Behavior: During Local-to-PCI DMA in true MPC860 mode (DMAMOD0/1[7] = 0, default), and Slow Terminate mode enabled (DMAMODE0/1[15] = 0, default), if EOT# is asserted when a DMA Local bus read is in progress, the silicon will complete a Local bus single cycle read (DMAMODE0/1[8] = 0) or a burst read (DMAMODE0/1[8] = 1) transfer normally regardless of the Local Bus address when EOT# is asserted.

For Local burst DMA, if the Local bus starting address is not Quad-Lword aligned (the starting address is not x0h), the silicon will perform single cycle reads until the address becomes Quad-Lword aligned (x0h), at which point bursting begins, for multiples of 16 bytes remaining to be transferred. If the DMA is programmed to finish on an address that is not Quad-Lword aligned, Local bus bursting terminates at the previous Quad-Lword aligned address, and the remaining bytes are transferred by single cycle accesses.

Actual Behavior: If DMA is configured to start at a non-Quad-Lword aligned address, if EOT# assertion coincides with the address phase (TS# assertion) of the first (single cycle) Local-to-PCI DMA read at address xCh, the silicon will perform the single cycle read and incorrectly assert the BURST# signal/pin.

Workaround:

Design the external logic responsible for EOT# assertion such that EOT# is not asserted if the Local-to-PCI DMA starting address is not Quad-Lword aligned, the first Quad-Lword address boundary has not yet been reached, and the next Local Bus address to be read for the DMA transfer has the four least significant bits of the Local Address (LA[28:31]) equal to Ch.

6. Direct Master Write and Back-to-Back Non-Contiguous Address Direct Master Read

Note. This is only an erratum for devices that use either the MPC860 Processor/Local Bus Mode (M Mode) or the non-multiplexed generic Processor/Local Bus Mode (C Mode). It is not an erratum for multiplexed generic Processor/Local Bus Mode designs (J Mode).

Documented Behavior: The silicon allows Direct Master reads to immediately follow Direct Master Writes (no data-to-address wait states nor turnaround cycle between transfers). If the Direct Master Cache Enable bit is enabled (DMPBAM[2] = 1), upon the first Direct Master Read the silicon will fetch and deliver the initial data requested by the Local Processor. The silicon will then continue to prefetch and store contiguous read data from the PCI bus until the Direct Master Read FIFO becomes full. This data is then available immediately if the Local Processor returns and requests the next contiguous data relative to where the last Direct Master Read disconnected. If a non-contiguous Direct Master read occurs after a Direct Master Read or Write, the silicon is to flush the pre-fetched data out of the Direct Master Read FIFO and fetch the newly requested data.

Actual Behavior: The silicon will corrupt pending Direct Master Write data stored in the Direct Master Write FIFO if the Direct Master Cache Enable bit is set (DMPBAM[2] = 1), the silicon has pre-fetched and stored contiguous read data from the PCI bus into the Direct Master Read FIFO (due to a previous Direct Master read), and a Direct Master Write that nearly fills (one Lword of space left) the Direct Master Write FIFO is immediately followed by a back-to-back (no data-to-address wait states), non-contiguous address Direct Master Read access.

Workarounds (do one of the following):

1. Disable the Direct Master Cache feature by setting DMPBAM[2] = 0.
2. Program the Local CPU for 1 Data-to-Address wait state (program i960 NXDA register to ≥1), (MPC860 program EHTR = 1).

7. EOT# Assertion at End of Local-to-PCI DMA with Unaligned Last Address

Note. This is only an erratum for devices that use the MPC860 Processor/Local Bus Mode (M Mode). It is not an erratum for generic Processor/Local Bus Mode designs (C/J Mode).

Documented Behavior: When executing a Local-to-PCI DMA transfer, the silicon will transfer data until the DMA completes or EOT# is asserted. If the Local Bus address for the last byte of DMA data is not adjacent to the next Lword boundary (which might occur if the Local Bus address (DMALADR[31:0]) and/or Transfer Size (DMASIZ0/1[22:0] is not a multiple of 4), partial data (less than 1 Lword) will be written to the PCI bus by appropriate, selective assertion of the PCI byte enables.

Actual Behavior: If the assertion of EOT# coincides with the end of the DMA transfer as specified by the DMASIZ0/1[22:0] register, and if the last data of the transfer is a partial Lword, the silicon will assert incorrect PCI Byte Enables (C/BE[7:0]#) when attempting to write the last partial data to the PCI bus.

Workarounds (do one of the following):

1. Increase DMA Transfer Size count (DMASIZ0/1[22:0]) to its maximum value and use EOT# to terminate the transfer.
2. Use Lword-aligned Addresses and DMA Transfer Sizes.

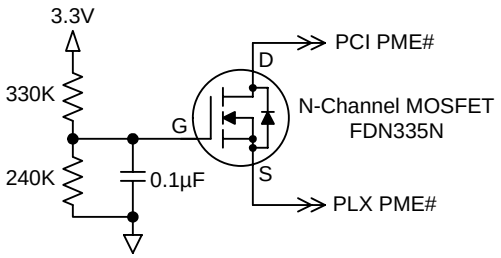
8. PME# Pin Isolation

Documented Behavior: Components that support the PCI Power Management D3cold state and implement PME# must ensure that, whenever main power is removed, the non-powered PME# generating logic and its output do not present a low impedance path to ground. Otherwise, there could be an accidental system wake up event.

Actual Behavior: The silicon does not ensure such protection and does present a low impedance path to ground when in the D3cold state, essentially causing an accidental wake up event every time the add-in card is inserted.

Workaround:

If PME# is to be implemented, an external FET must be inserted between the silicon's PME# signal pin and the PCI Slot. Suggested circuit:



9. Power Management PMDATA Register

Documented Behavior: Reading the PMDATA (Power Management Data) register from the PCI bus returns the Power Consumption and Power Dissipation values for one of the eight Power Management States supported by the silicon. To access a particular value, a PCI master must first write the index of that value to PMCSR[12:9] and then read back the value from PMDATA.

Actual Behavior: Writing an index to PMCSR[12:9] does not cause the specified value to be automatically loaded into the PMDATA register. Instead, a Local Bus master must read the index and write the requested value to PMDATA before the PCI Master reads the value.

Workaround:

If enabled (INTCSR[4] = 1), LINTo# is asserted whenever a PCI Master writes to PMCSR[1:0] to change the current power state. If, when writing PMCSR[1:0], the PCI master also writes the index of the desired PMDATA value to PMCSR[12:9], the Local Bus Power Management LINTo# servicing logic/code can be extended to load the indexed value into PMDATA and set the PME_Status bit (PMCSR[15] = 1). If enabled (PMCSR[8] = 1), this causes PME# assertion on the PCI Bus to indicate that it's ready to change the Power Management State. Upon receiving PME#, the PCI Master can read the indexed Power Management Data value from PMDATA[7:0] and service the PME# interrupt.

10. JTAG IDCODE Instruction

Documented Behavior: When the silicon's JTAG TAP machine logic comes out of power-on reset and enters the Shift_DR state, it will scan out the 32-bit IDCODE value. The JTAG TAP machine also supports being halted in the Pause_DR state during a scan operation and then returned to Shift_DR state.

Actual Behavior: When the silicon is in the Shift DR state after reset, if the TAP controller state is changed from SHIFT_DR to PAUSE_DR, and is then changed back to the SHIFT_DR before all data bits in the entire scan chain have been shifted through the silicon, the remaining bits in the scan chain will be corrupted.

Specifically, the silicon's IDCODE will be written over any data that has been shifted into the silicon from upstream devices in the chain.

Workarounds: (do one of the following)

1. Do not enter the Pause_DR state during the initial free run of the TAP machine clock.
2. Put the silicon at the first position in the scan chain, so that no devices are upstream from the silicon's TDI input. Provided the silicon's TAP controller is not paused before the entire IDCODE of the silicon is shifted out, the IDCODEs of all downstream devices in the chain will be correctly scanned out.

11. D3cold PME# Generation

Documented Behavior: In the D3cold state, the silicon generates a PME# assertion to the PCI bus if it senses a PMEREQ# assertion on the Local Bus.

Actual Behavior: In the D3cold state, due to improper powering of some PME#-related gates via Vdd, the silicon will generate extraneous PME# assertions upon entering a D3cold power state and it will not assert PME# in response to PMEREQ# assertions.

Workaround:

There is no workaround. The D3cold PME# Generation feature is not supported by the silicon.

The PRESENT_DET input (ball D8) should be pulled or tied low to force reads of the PME_Support D3cold bit in the Power Management Capabilities register (PMC, offset 42h bit 15) to return 0 regardless of the actual programmed value.

12. BTERM# Assertion in J-Mode during a Direct Slave Write or PCI-to-Local DMA Transfer causes data to be written twice (retracted)

The previously published erratum has been determined to not be an issue in the PCI 9056, and therefore this erratum is retracted.

13. DMA Constant Local Address

Issue #1

Documented Behavior: In DMA Constant Local Address mode (DMAMODEx[11] = 1) with Continuous Burst mode enabled (Local Burst Enable bit set, with Continuous Burst Enable bit as Don't Care, DMAMODEx[8:7] = 1xb), the silicon supports Local bus bursting to any Lword-aligned address (refer to Data Book v1.1 Sections 2.2.5.2, 2.2.5.3, 4.2.5.2 and 4.2.5.3, and to PCI 9056 Design Notes #1).

Actual Behavior: In DMA Constant Local Address mode with Local Burst enabled (DMAMODEx[8] = 1):

C/J modes: The silicon will always perform single cycle transfers (ADS# asserted for each data) rather than a burst transfer on the Local bus if the Local address is xCh and the Continuous Burst Enable bit is disabled (DMAMODEx[7] = 0).

M-mode: The silicon will always perform single cycle transfers (TS# asserted for each data) rather than a burst transfer on the Local bus if the Local address is x4h or x8h, or if the Local address is xCh and the Continuous Burst Enable bit is disabled (DMAMODEx[7] = 0). If the Local address is xCh and the Continuous Burst Enable bit is enabled (DMAMODEx[7] = 1), the silicon will transfer the first data using a single cycle transfer and will then burst remaining data until it releases the Local bus.

Workaround:

In DMA Constant Local Address mode (DMAMODEx[11] = 1) with Local Burst mode enabled (DMAMODEx[8] = 1), for burst transfers enable the Continuous Burst Enable bit (DMAMODEx[7] = 1). Additionally in M-mode, use quad-Lword aligned Local addresses (x0h).

Issue #2

Note. This is only an erratum for DMA channels configured for a 32-bit or 16-bit Local bus. It is not an erratum DMA channels configured for an 8-bit Local bus.

Documented Behavior: In Local-to-PCI DMA with Constant Local Address mode enabled (DMAMODEx[11] = 1), if the last datum of a block transfer to the Local bus is partial data relative to the Local bus width (in bytes), the silicon will toggle the C/J mode Local Byte Enables (LBE[3:0]#) or M-mode LA[30:31] and TSIZ[0:1] outputs to enable/disable the appropriate byte lanes.

Actual Behavior: In Local-to-PCI DMA with Constant Local Address mode enabled, if the DMA Transfer Size value defines the last datum of a block transfer to be partial data relative to the Local bus width (in bytes), the silicon will write this last datum as a full bus width transfer. In such case, in C/J modes, Local Byte Enables (LBE[3:0]#) all remain asserted, and in M-mode, LA[30:31] and TSIZ[0:1] are all 0's for the last datum written to the Local bus. In DMA Constant Local Address mode, only the first datum (relative to the Lword-aligned constant Local address) can be partial relative to bus width.

Workaround:

Use DMA Transfer Size byte counts that are a multiple of the Local bus width (in bytes). If partial data relative to Local bus width must be transferred, either transfer this data as a separate block, or as the start of the next block to transfer.

14. Direct Master Burst Read with non-contiguous LBE[3:0]# inputs and Deadlock with BREQo output assertion

Note. This is only an erratum for devices that use the generic Processor/Local Bus Mode designs (C/J Mode). It is not an erratum for MPC860 Processor/Local Bus Mode (M Mode).

Documented Behavior: If during a Direct Master Burst Read access a deadlock condition is encountered, causing BREQo output assertion (if enabled in EROMBA[4]), the PCI 9056 latches the command (LW/R#), address and Local Byte Enable (LBE[3:0]#) inputs for the Direct Master cycle on which BREQo output is asserted. After the backoff condition ends, the Local Bus Master restarts the last cycle with ADS# input assertion and the same command, address and byte enable inputs. The PCI 9056 compares these command, address, and byte enable inputs against the values that were latched. The values must be equal for the PCI 9056 to accept the repeated access. If the values are not equal, the PCI 9056 does not accept the access (the PCI 9056 does not request the PCI bus nor assert READY# output), and the PCI 9056 signals non-acceptance of the access by asserting BREQo output for 1 clock.

Actual Behavior: If a deadlock condition is encountered and BREQo is enabled and asserted during a Direct Master Burst Read access, the PCI9056 fails to latch the LBE[3:0]# value for the access at which BREQo is asserted, and the internal latch contains the value of LBE[3:0]# from the initial data phase of the burst read. On the subsequent retry attempt by the local master, if the LBE[3:0]# value does not match that from the initial burst read data phase, the PCI 9056 will not accept the cycle (and signals such by asserting BREQo output for 1 clock).

Workaround:

If a deadlock condition is encountered during a Direct Master Burst Read access with BREQo output assertion, the subsequent access by the Local Bus Master must match the same byte enables values that were used at the first data phase of the Direct Master access prior to deadlock. *For example,* this condition is satisfied when a Direct Master Burst Read initially requests 32-bit data at an Lword-aligned address (LBE[3:0]# = 0h), and the subsequent Direct Master access also requests 32-bit data at the backed-off address.

15. USERi Input Low at RST# De-assertion Might Allow Initial PCI Configuration Access to the Silicon to Fail

Documented Behavior: If the USERi input is sampled low at the rising edge of RST#, then the silicon does not respond to PCI Configuration transactions until the Local Init Status bit is set to one (LMISC1[2] = 1). Once the Local Init Status bit is set, if a PCI Configuration transaction targets the silicon, the silicon is to respond and complete the cycle (by asserting DEVSEL# and TRDY#).

The Local Init Status bit is set by any of the following:

1. EEDI/EEDO pulled high with:
   - LMISC1[2] value programmed (= 1) in the serial EEPROM, or,
   - blank EEPROM, or,
   - Local bus master programs LMISC1[2] = 1.
2. EEDI/EEDO pulled low

Actual Behavior: If the USERi input is sampled low at the rising edge of RST#, when the first PCI Configuration transaction targeting the silicon occurs after the Local Init Status bit becomes set (LMISC1[2] = 1), the silicon will not respond to this first Configuration transaction unless a PCI master has attempted a PCI bus transaction (by asserting FRAME# and IRDY#) in the interval between RST# de-assertion to the silicon and this first Configuration transaction to the silicon.

Subsequent attempts to access the silicon's PCI Configuration registers will complete normally (since the silicon has now detected FRAME# and IRDY# input assertion at least once prior to the Configuration transaction).

Impact: Systems that attempt to configure the silicon first and that make only one attempt to do so will not detect the device.

Workarounds (do one of the following):

1. If there are other slots available in the system, move the board to a different slot such that it will not be the first target of a Configuration transaction.
2. Retry the failed Configuration access a second time.

3. Pull or drive the USERi input high at RST# de-assertion. This will cause the silicon to respond to Configuration accesses with a PCI RETRY until the Local Init Status bit is set (LMISC1[2] = 1).

Note: USERi input should be held in the same state for at least 3 PCI CLK periods after RST# is de-asserted (high) to ensure its value is properly sampled. LCLK must also be running at this time.