PCI 9030
Errata Rev. 1.6
December 2001

Errata Documentation

A. Affected Silicon Revision

This document details errata in the following silicon:

Product	Part Number	Description	Status
PCI 9030	PCI9030-AA60PI	176-pin PQFP Package	Production Released Silicon
PCI 9030	PCI9030-AA60BI	180-pin μBGA Package	Production Released Silicon

B. Documentation Revision

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

Document	Revision	Description	Publication Date
PCI 9030 Data Book	1.2	Data Book	December 2001

C. Errata Summary

#	Description
1	Vital Product Data (VPD) Transfer Status Flag update
2	GPIO (General Purpose Input/Output) pins configured as outputs are driven only when PCI 9030 owns the local bus
3	In PCI 9030 Data Book version 1.0, pin assignments for pins 52 and 53 of the PQFP package are switched
4	LA[27:24] / GPIO[4:7] multiplexed I/O pins, and Non-multiplexed Mode LD[31:0] data or Multiplexed Mode LAD[31:0] address/data I/O pins, are driven low during PCI reset
5	JTAG Test Reset input (TRST#) is not optional
6	JTAG Bypass Mode
7	Power Management Interface Specification version support (revised from 1.5)
8	TDI is incorrectly shifted into the PCI 9030 JTAG Data Register while in BYPASS Mode when the Tap Controller is in the Shift-DR State
9	New Capability Pointer (CAP_PTR; PCI:34h) register value must be 40h (default) for the PCI 9030 to function correctly

D. Errata Details

1. Vital Product Data (VPD) Transfer Status Flag update

Problem: To perform a VPD write, data is written to the PVPDATA register, followed by a write of the VPD address with bit 31, the VPD Flag (PVPDAD[15]) set, after which the bit clears when the write completes. For a VPD read, the address is written with bit 31, the VPD Flag (PVPDAD[15]) clear, after which the Flag is set when the read completes. However, during VPD Write/Read transfers the status completion flag (PVPDAD [15]) intermittently fails to update. The system may hang if the flag is not set to the appropriate value after the transaction. This problem does not affect reads of the EEPROM during PCI 9030 initialization.

Solutions/Workarounds: (either/both)

1. If the flag is not set within 500 μsec time frame, the executed transaction needs to be restarted, to guarantee successful completion of the VPD transaction.

2. Use CNTRL[27:24] bits rather than VPD to read/write from/to the Serial EEPROM.

2. GPIO (General Purpose Input/Output) pins configured as outputs are driven only when PCI 9030 owns the local bus

Problem: GPIO (General Purpose Input/Output) pins configured as outputs are driven only when the PCI 9030 owns the local bus. When another local master owns the bus, GPIO pins configured as outputs are floated.

Solution/Workaround: If PCI 9030 GPIO pins are used and if another local master will be granted the local bus, configure and use the pins as input (GPI) only.

3. In PCI 9030 Data Book version 1.0, pin assignments for pins 52 and 53 of the PQFP package are switched

Problem: In The PCI 9030 Data Book version 1.0, pin assignments for pins 52 and 53 of the PQFP package are switched. The correct values are, LEDon# for pin 52 and $V_{I/O}$ for pin 53. The values specified in Table 11-2 (Power and Ground Pins (176-Pin PQFP)), Table 11-7 (Local Bus Mode Independent Interface Pins), and Figure 13-3 (Physical Specification) in version 1.0 of the data book are not correct and should instead indicate LEDon# for pin 52 and $V_{I/O}$ for pin 53.

Note that during PCI reset (RST# asserted), LEDon# is asserted. If the LEDon# and $V_{I/O}$ signals are switched, LEDon# will sink short circuit current from the $V_{I/O}$ source during PCI reset.

Solution: The PCI 9030 Data Book has been updated from version 1.0 to version 1.1. This error has been corrected on pages 11-2, 11-7, and 13-3 of the PCI 9030 Data Book, version 1.1 (January 2001). The updated PCI 9030 Data Book, available on the www.plxtech.com web site, reflects these changes.

4. LA[27:24] / GPIO[4:7] multiplexed I/O pins, and Non-multiplexed Mode LD[31:0] data or Multiplexed Mode LAD[31:0] address/data I/O pins, are driven low during PCI reset

Problem: The PCI 9030 drives the LA[27:24] address pins (which can be configured as GPIO[4:7] pins after PCI reset completes), and Non-multiplexed mode LD[31:0] data or Multiplexed mode LAD[31:0] address/data I/O pins, low during PCI reset rather than to high impedance state. This may result in bus contention if other devices use these signals during PCI reset. The remaining address bus signals, LA[23:2], as well as the LBE[3:0]# byte enables, are floated during PCI reset.

Solutions/Workarounds: (any)

1. If the LA[27:24] pins must be floated during PCI reset, isolate these address outputs with a three-state bus switch such as a Pericom PI3C3126A (3.3V, 5V-tolerant 4-bit active high bus switch), using LRESETo# to drive the bus switch Enable inputs.

2. If the Non-multiplexed mode LD[31:0] data bus, or Multiplexed mode LAD[31:0] multiplexed address/data bus must be floated during PCI reset, isolate the bus with a three-state bus switch such as a Pericom PI3C34X245 (3.3V, 5V-tolerant 32-bit bus switch), using LRESETo# to drive an inverter that has its output connected to the active low bus switch Enable inputs.

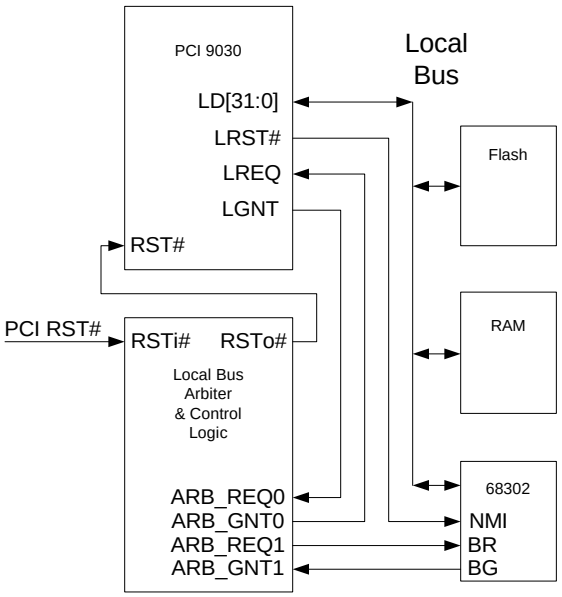
3. If both LA[27:24] and LD/LAD[31:0] buses must be floated during PCI reset, isolate these buses with a three-state bus switch such as a Pericom PI3C34X461 (3.3V, 5V-tolerant 40-bit bus switch), using LRESETo# to drive an inverter that has its output connected to the active low bus switch Enable inputs.

4. Create external logic that controls the local bus arbitration and captures PCI RST# to provide a conditioned RST# signal to the PCI 9030. The basic concept is to always request the local bus from the PCI 9030 and only grant the bus to PCI 9030 when it informs the system that it needs the bus by negating its LGNT pin. Below is a detailed description of this workaround.

Block Diagram, Workaround concept

Below is a block diagram of one possible implementation, showing only the relevant signals. The necessity of this workaround is required only if the user does not want PCI reset to generate a system/board reset. Because the PCI 9030 drives the LA[27:24], LD/LAD[31:0], and LRST# pins low as long its RST# is asserted, bus contention will occur on the local bus if other local bus devices are using the bus at that time. The workaround concept is as follows:

- Capture the PCI RST#,
- "give" the local bus to the PCI 9030;
- assert a reset pulse on the PCI 9030 RST# pin for a minimum of 2 rising PCI clock edges;
- wait for the PCI 9030 to process its RST#;
- "take" local bus control back from the PCI 9030, resume normal local bus activity, prepare for the next PCI RST# after it negates



Block Diagram of one possible work around implementation

Using the PCI 9030 with other Local Masters

Because the PCI 9030 contains only Direct Slave functionality, the Local Bus Arbiter was designed to assume it owns the local bus at all times except when it gives it up (by asserting LGNT). Therefore, (under this assumption) it is acceptable to drive the local bus anytime, including during PCI reset, except when Local bus ownership is granted to another Local bus device during normal operation. If LGNT and LREQ are negated, bus contention will occur if another Local Master tries to drive control and data signals on the Local bus.

To allow another device or devices to be a master on the local bus, an external arbiter needs to be designed such that it takes the local bus from the PCI 9030 and only gives it back to the PCI 9030 when it needs it. This can be done by setting register CNTRL[7] = 0 and asserting LREQ (as early as power up). Then the only time the PCI 9030 gets the local bus back is when it negates LGNT (preempt condition, indicating it has a pending command - only if CNTRL[7] = 0), and the external arbiter grants the PCI 9030 the Local bus by negating LREQ. The arbiter can then reassert LREQ a minimum of one clock after it was negated and wait for LGNT to be asserted before taking the Local bus back over. Depending upon the system, the user may want to customize their solution to allow windows of access depending upon expected activity.

Workaround to avoid Local Bus contention during PCI reset assertion

The workaround requires the PCI 9030 RST# signal to be asserted for a minimum of two PCI rising clock edges. If you only have access to the Local clock then you would need to determine the minimum frequency of PCI operation. A close approximation is - Minimum PCI frequency = 2.01 x 1/period, where period is the amount of time you assert RST#. If the PCI clock is available for use, the design can be constructed to ensure RST# is asserted for two rising edges, and using the PCI clock will allow the design to work down to 0.0 MHz. LREQ may be asserted before, during and/or after RST# and LRST# assertion.

5. JTAG Test Reset input (TRST#) is not optional

Problem: If TRST# is not asserted during PCI RST# assertion, the PCI 9030 could initialize into an undefined state, precluding normal chip logic operation.

Solutions/Workarounds for a Non-JTAG system:

Keep TRST# asserted by pulling it low using a 1K to 10K resistor to ground. This will keep the JTAG logic in a reset state and will enable normal chip logic operation.

Solutions/Workarounds for a system with JTAG or Hot Swap:

Buffer PCI RST# and use the output to drive both the TRST# and RST# pins on the PCI 9030 simultaneously.

Notes about JTAG implementations:

1. Please refer to both the PCI specification and the JTAG specification (IEEE 1149.1) for JTAG pin requirements.

2. The JTAG specification requires pull-ups on TDI, TMS and TRST#. To stay compliant with the PCI Specification, no internal pull-ups are provided on the JTAG pins in the PCI 9030. Therefore the user must add these pull-ups externally.

6. JTAG Bypass Mode

Problem: According to Rule 9.1.1 (b) of the JTAG specification "the shift-register stage shall be set to a logic zero on the rising edge of TCK following entry into the *Capture-DR* controller state." If the PCI 9030 TDI pin is set to a logic 1 one TCK before entry into the *Shift-DR* state, the first value clocked out of the PCI 9030 TDO pin will be a 1 instead of the required 0.

Solutions/Workarounds: (any)

1. Put the PCI 9030 at the end of the boundary scan chain and, when using the Bypass command, either mask out the first value of TDO or expect the first value out of TDO to be a 1.

2. If the PCI 9030 is not at the end of the chain, use software to modify the scan pattern such that parts further down the chain will not be affected if the first value out of the Bypass register is a 1.

7. Power Management Interface Specification version support

The previously published erratum is retracted and the issue is republished as PCI 9030 Design Notes #1.

8. TDI is incorrectly shifted into the PCI 9030 JTAG Boundary Scan Register while in BYPASS Mode when the Tap Controller is in the Shift-DR State

Description: While in Bypass mode, bypassed data will be incorrectly shifted into the PCI 9030 JTAG Boundary Scan Register. The JTAG Boundary Scan Register shifts each Boundary Scan Register location and TDI into its first Register location on each rising edge of TCK while in Shift-DR state. The new (unwanted) Boundary Scan Register values will be applied to the PCI 9030 pins when the tap controller passes through the Update-DR State. TDI is correctly shifted to TDO during this time in Bypass mode.

Solutions/Workaround(s) #1: Do not put the PCI 9030 into Bypass Mode. Always shift in the Data Register values needed into the PCI 9030.

Solutions/Workaround(s) #2: Disconnect the PCI 9030 Tap Controller and create separate control logic to Bypass the PCI 9030 JTAG logic when needed.

9. New Capability Pointer (CAP_PTR; PCI:34h) register value must be 40h (default) for the PCI 9030 to function correctly

Description: The PCI 9030 contains a bit in the PCI Status Register called the New Capability Functions Support bit (PCISR[4]). A value of zero indicates that no New Capabilities linked list is available. A value of one indicates that the value read from the CAP_PTR register at Configuration offset 34h is a pointer in Configuration Space to a linked list of New Capabilities. The default values are PCISR[4] = 1 and CAP_PTR = 40h. PCISR[4] and CAP_PTR are writable only by serial EEPROM.

Problem: Regardless of the setting of PCISR[4], the value in CAP_PTR is used to point to internal registers inside the PCI 9030. Depending upon the value of CAP_PTR, PCI Configuration Write accesses to the PCI 9030 will incorrectly write over the Capabilities registers in addition to writing the intended configuration register. A CAP_PTR value of 40h is the only value that allows the silicon to function properly.

Solutions/Workarounds:

1. If any of the Capabilities List functions (Power Management, Hot Swap, and/or VPD) are to be enabled, the Power Management capability must be enabled. To enable any of the Capabilities List functions, set PCISR[4] = 1 and CAP_PTR = 40h (default values). PMNEXT (PCI:41h) can be set to 48h (default pointing to Hot Swap), or if the Hot Swap capability is to be disabled then to 4Ch (VPD), or if both Hot Swap and VPD are to be disabled then to 0. All values must be programmed in EEPROM. If no EEPROM is present then default register values enable all three capabilities.
2. If it is desirable to disable the Hot Swap capability, set HS_CNTL (PCI:48h) and HS_NEXT (PCI:49h) to 0 in EEPROM. PMNEXT (PCI:41h) must be set to 4Ch if VPD is enabled, otherwise to 0 (not to the default value 48h).
3. If it is desirable to disable the VPD capability (see Errata #1), if Hot Swap capability is enabled then set HS_NEXT (PCI:49h) to 0 in EEPROM, or if Hot Swap capability is disabled then set PMNEXT (PCI:41h) to 0 in EEPROM.
4. Always set PVPD_NEXT (PCI:4Dh) to 0 in EEPROM.
5. If it is desirable to disable all Capabilities List functions, set PCISR[4] = 0 and CAP_PTR = 40h, in EEPROM.