PCI 9080-3

Errata

August 4, 1998

A. Product Status

The scope of this document encompasses the PCI 9080-3

Product status	Description	Samples	Production	Collateral Documentation
PCI 9080-3	Fixes selected errata	31 October 1997	January 1998	Technical update rev 08
				Data sheet v1.04

B. PCI 9080-3 Errata

1) EOT pin usage

The use of EOT to terminate ongoing DMA transfers may result in corrupted data being transferred across the chip. This is true for aligned and unaligned transfers, and for local to PCI and PCI to local transfers.

Solution/Workaround 1:

Use the DMA engines in demand mode. Use the DREQ# pin to pause ongoing DMA transactions. EOT or DMA abort can then be used to terminate the pending transfers once the DMA channel has paused.

Solution/Workaround 2:

Write to DMA control registers to pause or terminate ongoing DMA transfers.

Solution/Workaround 3:

Use of DMA when transferring to and from 32 bit buses under the following condition: (1) The starting addresses on both the PCI and local sides must be 32 bit aligned; (2) The transfer count must also be in 32 bit multiples; and (3) The transfers must not be in Write and Invalidate or clear count modes.

2) Local Lock output assertions with Direct Slave Reads

After the first Direct Slave Read transaction, subsequent Direct Slave Reads will cause the LLOCKO# signal to assert with the LHOLD signal. The 9080 will act as if the two lines are shorted together. This has the consequence of locking the local processor every time.

Solution/Workaround 1:

Execute a Direct Slave Write cycle before any Direct Slave Reads. A Direct Slave Write cycle sets the internal state machine such that any number of subsequent Direct Slave Reads will not activate the LLOCKO# bug.

Solution/Workaround 2:

Disconnect use of the LLOCKO# pin.

3) Demand Mode DMA in Stop Transfer Mode

This update only applies to Demand Mode DMA transfers when the Stop Transfer bit DMAMODE0[15]) is active. If DREQ# is deasserted before all data transfers are complete (pausing the DMA), subsequent assertions of DREQ# will fail to transfer the last long word of the intended transfers. Furthermore, if chaining DMA is active, the 9080 will fail to read the descriptor of the following link in the chain.

Solution/Workaround 1

Always specify one extra long word to be transferred when attempting to pause demand mode transfers using DREQ#.

Solution/Workaround 2

Do not deassert DREQ# until all intended transfers are complete.

Solution/Workaround 3

Pause the DMA transfer by writing to Channel enable bit rather than using demand mode.

Solution/Workaround 4

Pause the DMA without the stop transfer bit enabled.

4) Unaligned DMA In Big Endian Mode.

Unaligned DMA cycles in Big Endian Mode may result in corrupted data being sent across the device

Solution/Workaround 1:

(Convention: Local device refers to device which uses big endian byte ordering.)

To interface PCI 9080 to a Big Endian device on the local bus, disable the Big Endian feature in the PCI 9080 and connect the data bits [0:31] of the local device directly to the data bit [0:31] of the local bus and PCI 9080. Therefore, long word transfers are compatible between the two domains. If the local device writes a byte to address 0, the data appears on bits [24:31] and LBE3# to the PCI 9080 is asserted. This corresponds to byte address 3 on the PCI bus. If byte or word transfers to PCI 9080 are performed, then the local device software must map the address so that the data appears on the correct byte lane. Also, if byte or word data structures are shared by the local device and other PCI Little Endian devices, then address mapping in the software must be included.

Solution/Workaround 2:

Swap the byte lanes when connecting the local device to the PCI 9080 local bus. In this case byte addressing is compatible, but word and long word addressing is not.

5) Direct Slave Non- 2.1, Cache, Write flush mode

If the PCI9080's Local/DMA arbitration register is set for non 2.1 mode (bit 24 = 0) , cache mode enabled (bit 28 = 1, no Flush mode), write flushes read mode off (bit 26 = 0), a write will cause the local bus to stop reading data. The local bus never resumes reading data, causing all read attempts to be retried on the PCI bus.

Solution/Workaround 1:

Set the PCI9080 to operate in PCI 2.1 mode. (Set Local/DMA arbitration register bit 24 = 1)

Solution/Workaround 2:

Do not use Cache mode (Set Local/DMA arbitration register bit 28 = 0, Write Flushes cache mode).

Solution/Workaround 3:

Set the PCI 9080 to operate in PCI Read with Write Flush Mode on. (Set Local/DMA arbitration register bit 26=1)

6) Direct Master Burst Read past a 64K boundary

This errata prevents the 9080 from prefetching read data past a 64K boundary (low 16 bits of the PCI address). This implies that a direct master read must not request burst data past this boundary. If it does, the 9080 will hang the local bus at the boundary, refusing to gather any further data.

This 64K boundary is not a hard wall. It can be crossed in 2 separate cycles. It only cannot be crossed in a single burst cycle.

Solution/Workaround 1:

Break all direct master burst reads at 64K boundaries. Separating bursts across the boundary into 2 separate bursts will work just fine.

Solution/Workaround 2:

Break all direct master burst reads at 4, 16, or 32K boundaries. While this places even more restrictions on the requesting device, it may be easier to implement than the 64K boundary. (The 9080 provides an option bit which prevents prefetching past 4K boundaries)

Solution/Workaround 3:

The 64K boundary can be crossed if the local master only reads in single cycles.

7) Simultaneous PCI DMA Descriptor read with local access to internal registers.

If the PCI 9080 encounters a PCI target retry when attempting to read a DMA descriptor from the PCI bus, the PCI 9080 will retry the read after a few clocks. A local read of the PCI 9080's internal configuration registers coincident with this retried descriptor read will result in the local initiator receiving data from the DMA descriptor register for which the descriptor is being read. A local write to configuration registers synchronous to this descriptor read will receive a READYo# from the 9080 even though the write never gets through. (The write data never actually goes anywhere.)

Solution/Workaround 1:

On local reads, if the value returned is the same as the DMA configuration register, reread the internal register.

Solution/Workaround 2:

Read or write to the internal registers twice and compare the values.

8) Aborting chained DMA with descriptors on both PCI and local buses

The PCI 9080 has 2 independent DMA controllers that support chaining DMA with their descriptors located on either the PCI or local buses. This makes it possible for one DMA channel to have descriptors on the PCI bus while the other channel has descriptors on the local bus. If a DMA channel is aborted while doing local to PCI transfers and the local bus is running faster than the PCI bus, a subsequent DMA local to PCI transfer will result in incorrect data being transferred across the PCI 9080 (when both channels are active with descriptors on both sides of the bus).

Solution/Workaround:

Follow the DMA abort with a DMA PCI to local transfer (on the aborted channel) of zero bytes. This will reset the internal state machines for subsequent local to PCI transfers.

9) Configuration Write with WAITi#

WAITi# can be used by the Local Bus master to insert wait state when accessing the PCI9080 or PCI bus. During a configuration write, WAITi# must be asserted at least 2 clocks before READYo# is asserted for the 9080 to sense it. If not, PCI9080 asserts READYo# for 1 clock and terminates the cycle instead of waits until WAITi# is negated. This problem does not apply to configuration read, memory read, or memory write cycle.

Solution/Workaround:

During a configuration write, assert WAITi# at least 2 clocks before READYo# is asserted for the PCI 9080 to sense it.