AP-753

APPLICATION NOTE

PCI Local Bus Specification Revision 2.1 vs. Revision 2.0

Intel Corporation
5000 W. Chandler Blvd.
Chandler, Arizona 85226

March 1997

Order Number: 273011-001

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation
P.O. Box 7641
Mt. Prospect IL 60056-7641
or call 1-800-548-4725

1.0 Introduction

This application note discusses the technical differences between the 2.0 and 2.1 revisions of the *PCI Local Bus Specification* as they might impact the developer of PCI devices. It is not intended to detail the literal differences between these revisions or to provide a full explanation of the current (2.1) specification. Many clarifications and implementation notes have been added to the 2.1 revision of the PCI specification to aid the developer's understanding of the bus architecture. Please consult the official specification available from the PCI Special Interest Group (PCI SIG) for these additions.

This document assumes that the reader is already familiar with the implementation of the PCI Local Bus Specification, Revision 2.0.

2.0 Document Organization

This document is intended to be a starting point for developers who wish to know what effects the transition from the 2.0 revision to the 2.1 revision of the PCI bus specification may have on their products. To help the reader find topics discussed in this application note, the section numbers and headings that follow correspond to the appropriate headings in the 2.1 revision of the specification.

Throughout this document the *PCI Local Bus Specification, Revision 2.0, Production Version* is referred to as "Revision 2.0." Likewise, the *PCI Local Bus Specification, Revision 2.1, Production Version* is referred to as "Revision 2.1." Refer to Section 4.0 of this document for information on contacting the PCI Special Interest Group.

3.0 Comparison of Revisions 2.0 and 2.1 by Chapter

Chapter 2, Signal Definitions

2.2.8 Additional Signals

Two new signals have been defined in Revision 2.1:

- CLKRUN#

  The *Clock Running signal* is an optional signal that allows the central resource to request permission to stop or slow the PCI bus clock (**CLK**). This feature is useful in the PCI mobile environment for power conservation. **CLKRUN#** is not a defined pin on the PCI connector and therefore cannot be used on add-in boards. The implementation of this signal is discussed further in the *PCI Mobile Design Guide* available from the PCI SIG.

- M66EN

  M66EN is an optional input signal to a PCI device that indicates to the device whether the bus segment is operating at 66 MHz or 33 MHz. By doubling the operating PCI clock to 66 MHz, the bandwidth of the PCI bus is also doubled. Refer to Chapter 7 of the 2.1 specification for more information about the 66 MHz bus speed option.

Chapter 3, Bus Operation

3.1.1 Command Definition

The *Memory Write & Invalidate* command can now use only linear incrementing burst mode, which is enabled by clearing bits *AD[1:0]* during a memory command. The mode of this memory command was not specified in Revision 2.0.

The *Memory Read Line* command now indicates that the master intends to fetch a complete system cacheline. In Revision 2.0, this command indicated only that the master intended to complete more than two 32-bit PCI data phases.

3.2.2 Addressing

The meaning of the *AD[1:0]* signals during Memory commands has changed:

Memory Command Modes

AD1	AD0	Revision 2.0 Burst Order	Revision 2.1 Burst Order
0	0	Linear Incrementing	Linear Incrementing
0	1	Cacheline Toggle Mode	Reserved
1	0	Reserved	Cacheline Wrap Mode
1	1	Reserved	Reserved

The toggle mode addressing used in Revision 2.0 is similar to Intel486™ and Pentium® processor toggle mode addressing. Revision 2.1 employs Cacheline Wrap Mode. The data access proceeds by incrementing linearly to the end of the cacheline. It then wraps around and continues transferring data from the beginning of the line to fill the entire cacheline.

3.2.5 Transaction Ordering

Transaction ordering rules have been introduced to Revision 2.1 to accomplish three goals:

1. To satisfy the write-results ordering requirements of the Producer-Consumer Model detailed in Appendix E of Revision 2.1.

2. To allow some transactions to be posted for improved performance.

3. To prevent bus deadlock conditions when the posting buffers must be flushed to meet the write-results ordering requirement.

Refer to section 3.2.5 of Revision 2.1 for a complete discussion of transaction ordering.

3.3.3.2 Target Initiated Termination

This section of the PCI specification was expanded to clarify the various cases of target initiated termination. The target initiated termination without data that was defined as Disconnect-C in Revision 2.0 is now superseded by two uniquely identified cases: Disconnect-1 and Disconnect-2. These are distinguished by the state of **IRDY**# when the **STOP**# signal is asserted. When **IRDY**# is *deasserted* (high) as the target asserts **STOP**#, the target termination is referred to as Disconnect-1. If **IRDY**# is *asserted* (low) when **STOP**# is asserted, the termination is referred to as Disconnect-2.

3.3.3.2.2 Requirements on a Master because of Target Termination

A master that is retried by a target must unconditionally repeat the same transfer until it is completed with a termination of Normal, Disconnect, Target-abort, or Master-abort. This was implied by the authors in Revision 2.0, but it may have been misunderstood by many readers. Therefore, it is explicitly stated in Revision 2.1.

3.3.3.3 Delayed Transactions

Not all PCI devices will be able to meet the new latency requirements (described below) defined in Revision 2.1. Therefore, the authors of the PCI specification have introduced the concept of *Delayed Transactions* for use by those targets that cannot complete the initial data phase within the required clock cycles. Through the use of delayed transactions, the PCI bus is not held in wait states while a master waits for a slow device.

To implement a delayed transaction, the target must terminate the transaction with the *Retry* signal once the request information has been latched. The master is unaware of the reason that the target terminated the transaction and is required to initiate the transaction again. In the meantime, another master may be allowed to use the bus for an independent transaction.

For a complete discussion of the details and implementation of delayed transactions, refer to Revision 2.1, Section 3.3.3.3.

3.4 Arbitration

Both revisions of the PCI specification allow any bus arbitration algorithm to be implemented by the central PCI bus arbiter as long as only a single GNT# line is asserted on any clock. Revision 2.1 now requires that this algorithm be *fair*. A fairness algorithm allows low-priority masters access to the bus even when higher-priority masters are continually requesting it.

3.5 Latency

The new latency rules introduced in Revision 2.1 may have the greatest impact on the developer of PCI devices, especially those who have Revision 2.0-compliant devices. Revision 2.0 *recommended* that targets be able to complete the initial data phase within 16 clock cycles of the initiation of the transaction. It was also *recommended* that the master be able to assert IRDY# within 8 clock cycles of all data phases within a transaction, including the initial data phase. In Revision 2.1, both of these recommendations are *requirements* of all PCI-compliant devices.

In Revision 2.1, three possibilities have been identified with respect to meeting these new latency rules.

1. The first possibility is that a devices can meet the initial latency requirement of 16 clock cycles. The authors of the specification expect that most I/O controllers developed before Revision 2.1 will be able to meet this requirement and therefore will not require modifications.

2. The second possibility is that a device usually can meet the initial latency requirement, but sometimes cannot. These devices should use the *Retry* target initiated termination so that the request can be repeated.

3. The third possibility is that a device frequently cannot meet the initial latency requirement. These devices are required to implement *Delayed Transaction* as discussed above.

All PCI devices are granted two exceptions to the latency requirements. Both of these exceptions occur during initialization and do not have an upper bound on initial latency:

- POST code accessing the device's configuration registers.
- POST code copying the expansion ROM image to memory.

The final exception to the initial latency rule applies only to host bus bridges. These types of bridges are allowed an additional 16 clocks, for a total of 32, to complete the initial data phase if the access hits a modified cacheline.

It is expected that all masters will be able to meet the new 8 tick rule by using their data buffers.

The section on the new latency requirements in Revision 2.1 is extensive and beyond the scope of this document. Due to the possible impact that these new requirements may have on a system, we recommend that you consult Revision 2.1 for further information.

3.6 Exclusive Access

The PCI specification provides an exclusive access mechanism that allows non-exclusive access to take place in the face of exclusive accesses. This mechanism is referred to as a *resource lock* and is implemented via the **LOCK#** signal. Although Revision 2.1 still supports resource locking, the **LOCK#** signal is no longer *required* on devices providing system memory as it was in Revision 2.0; in Revision 2.1 it is simply *recommended*.

3.7.3 Address/Data Stepping

All agents are now required by Revision 2.1 to handle address and data stepping, whereas generating it is optional. *Stepping* a signal allows devices with "weak" output buffers to assert a set of signals over several clock cycles to reduce the buffer's generated ground current load. Similarly, a device with "strong" output buffers can assert a subset of these over several cycles to reduce the number of signals that must be switched simultaneously.

Chapter 4, Electrical Specification

4.2.1.2 AC Specifications

In Revision 2.0, Note 4 to the *AC Specifications for 5V Signaling* table required that the minimum slew rate be met, but the maximum rate was given only as a guideline. In Revision 2.1, both the maximum and minimum slew rates are required for all PCI devices. Motherboard designers must remember that some legacy devices may have faster signal edges and should account for this in their signal integrity modeling.

Chapter 6, Configuration Space

6.1 Configuration Space Organization

Two locations of the Configuration Space Header that had previously been reserved have now been defined.

- The 32-bit doubleword at location 28h is now the CardBus CIS Pointer.
- The 32-bit doubleword at location 2Ch contains the Subsystem ID information. This consists of a 16-bit word at location 2Ch, which is the Subsystem Vendor ID, and a 16-bit word at 2Eh, which is the Subsystem ID. In revision 2.1, these new items are discussed in section 6.2.3 Miscellaneous Functions.

6.2.1 Device Identification

New Sub-Classes to Existing Base Classes

Base Class	New Sub-Class	Interface	Meaning
01h	04h	00h	RAID Controller
02h	03h	00h	ATM Controller
03h	00h	01h	8514-compatible controller – 2E8h and its aliases, 2EAh-2EFh.
06h	06h	00h	NuBus* bridge
	07h	00h	CardBus* bridge

New Base Classes and Corresponding Sub-Classes (Sheet 1 of 2)

Base Class	New Sub-Class	Interface	Meaning
07h	00h	00h	Generic XT-compatible serial controller
		01h	16450-compatible serial controller
		02h	16550-compatible serial controller
	01h 80h	00h	Parallel port
		01h	Bidirectional parallel port
		02h	ECP 1.X compliant parallel port
		00h	Other communications device
08h	00h	00h	Generic 8259 PIC
		01h	ISA PIC
		02h	EISA PIC
	01h	00h	Generic 8237 DMA controller
		01h	ISA DMA controller
		02h	EISA DMA controller
	02h	00h	Generic 8254 system timer
		01h	ISA system timer
		02h	EISA system timers (two timers)
	03h	00h	Generic RTC controller
		01h	ISA RTC controller
	80h	00h	Other system peripheral

New Base Classes and Corresponding Sub-Classes (Sheet 2 of 2)

Base Class	New Sub-Class	Interface	Meaning
09h	00h	00h	Keyboard controller
	01h	00h	Digitizer (pen)
	02h	00h	Mouse controller
	80h	00h	Other input controller
0Ah	00h	00h	Generic docking station
	80h	00h	Other type of docking station
0Bh	00h	00h	386
	01h	00h	486
	02h	00h	Pentium®
	10h	00h	Alpha*
	20h	00h	PowerPC*
	40h	00h	Co-processor
0Ch	00h	00h	FireWire* (IEEE 1394)
	01h	00h	ACCESS* bus
	02h	00h	SSA*
	03h	00h	Universal Serial Bus (USB)
	04h	00h	Fibre Channel*

6.2.4 Miscellaneous functions

- CardBus CIS Pointer

  To help bridge the PCI bus to the CardBus peripheral interface, a CardBus CIS pointer has been added to the PCI configuration space. This is an optional register to be used for devices that share silicon between CardBus and the PCI bus. The location of this pointer is 28h of the type 00h Configuration Space Header, a previously reserved location. This field is used to point to the Card Information Structure (CIS) of the CardBus card. For more information on the CIS, refer to the *PCMCIA v2.10 Specification* under the heading *Card Metaformat*.

- *Subsystem Vendor ID* and *Subsystem ID*

  Revision 2.1 has extended the range of PCI device identification by defining IDs for the PCI device's subsystem. In this way, separate add-in board manufacturers can uniquely identify their devices even though they may use the same PCI controller. The PCI controllers can have identical Vendor and Device IDs but unique Subsystem IDs. Subsystem Vendor and Subsystem IDs must be obtained through the PCI SIG.

6.2.5.1 Address Maps

Both revisions of the *PCI Local Bus Specification* use the base address registers of the type 00h PCI Configuration Space Header to request address space for PCI devices. However, Revision 2.1 has placed a restriction on those functions that map control functions into I/O space. These devices cannot consume more than 256 bytes per I/O base address register.

6.2.5.2 Expansion ROM Base Address Register

As defined by Revision 2.0, a PCI device can request address space by implementing the expansion ROM base address register. Revision 2.1 has sustained this feature; however, it has imposed a 16-Mbyte maximum on the amount of memory a device may request. This implies that the 24th bit position in the expansion ROM base address register is the most significant bit that can be hard-wired to ground.

6.2.3 Device Status

Two new bits of the PCI Status register that were previously reserved have been defined in Revision 2.1:

Bit position #6, UDF_Supported, is an optional read-only bit that indicates whether the PCI device supports user-definable features as described in section 6.7 of Revision 2.1. If the device supports UDFs, this bit must return *1* when read. Otherwise, it must return *0*.

Bit position #5, 66MHZ_CAPABLE, is also an optional read-only bit that indicates whether or not this device is capable of operating on a 66 MHz PCI bus segment, as defined in Chapter 7 of Revision 2.1. If the device is capable of operating at 66 MHz, this bit must return *1* when read. Otherwise, it must return *0*.

6.3.1.2 PCI Data Structure Format

Code Type 1 of the PCI expansion ROM data structure now conforms to the Open Firmware standard for PCI[1]. Previously, Code Type 1 indicated that a particular section of ROM contained code in the OPENBOOT standard for PCI devices[2]. Open firmware and open boot are a processor-independent and system architecture-independent standard for dealing with device-specific option ROM code.

6.4 Vital Product Data (optional)

Vital Product Data (VPD) is information that uniquely identifies the aspects of a system's hardware, software, and microcode elements. It also provides a mechanism for the system to monitor certain aspects of a device for performance and failure rate measurements. This information is useful for the transition to a true Plug-and-Play computing environment.

The pointer to the VPD was defined in section 6.3.1.2 of Revision 2.0, but the content of the VDP data structure was not defined until Revision 2.1. This pointer is located in the PCI Data Structure, which is within the first 64 Kbytes of the Expansion ROM image. Because the VPD data structure is an optional feature of a PCI device and the location of the pointer has been defined in the previous specification, there should not be any impact on devices that do not implement this feature. Refer to section 6.4 of Revision 2.1 for more information on VPD.

6.7 User-Definable Configuration Items (optional)

Support for User-Definable Features (UDFs) has been added to Revision 2.1. This device-specific option used by PCI devices requires information about the environment into which the devices are installed. The example given in the specification is that of a token ring network adapter that is dependent on the specific token ring network that the adaptor is installed in to configure its speed setting. A text-based PCI Configuration File (PCF) must be supplied with a device that supports UDF so that the system must can present configurable options to the user for selection. The format of the PCF is defined in section 6.7.2 of Revision 2.1.

A new read-only bit has been defined in the PCI Status register to determine if a device supports UDFs. This is bit #6, which was previously reserved in Revision 2.0. When read, this bit must return 0 if the device does not support UDFs, and it must return *1* if UDFs are supported.

---

1. Documentation for Open Firmware is available from IEEE in the *1275-1994 Standard for Boot (Initialization, Configuration) Firmware Core Requirements and Practices* document. Also refer to *PCI Bus Binding to IEEE 1275-1994* which specifies the application of Open Firmware to the PCI bus architecture and *PCI Bus Supplement: 2.0 Aug. 6, 1996* available in postscript format from the Open Firmware Working Group at http://playground.sun.com/1275/bindings/pci/pci2_0.ps
2. Documentation on the OPENBOOT standard is no longer available from IEEE. This standard was previously defined in the IEEE document *Draft Std P1275/D7* dated January 4, 1993.

Chapter 7, 66 MHz PCI Specification

7.0 66 MHz Bus Operation (optional)

As the speed of processors continues to increase and devices such as video adapters require more bandwidth, bus speeds must also increase to avoid data bottlenecks. Revision 2.1 has introduced support for a 66 MHz bus operation speed, doubling the throughput of the current standard, 33 MHz. This is an optional feature and is backward-compatible with 33 MHz PCI devices and buses. If a 66 MHz capable PCI device is installed in a 33 MHz PCI bus, the device must operate at 33 MHz. Likewise, if any 33 MHz PCI devices are installed into a 66 MHz PCI bus, the PCI bus must operate at 33 MHz. All 66 MHz devices and motherboards must be capable of dropping their performance to 33 MHz.

So that the system can determine if a device can operate at 66MHz, bit #5 of the PCI Status register has been defined in Revision 2.1 as the 66MHZ_CAPABLE flag. This is a read-only bit which will return *1* when read if the device is capable of operating at 66 MHz, and *0* when it is not.

An existing ground pin (pin 49, side B) has been defined as the 66MHZ_ENABLE (M66EN) pin. This is defined only on 3.3 V PCI devices. It remains a ground pin on the 5.0 V devices because 66 MHz bus operation can only be implemented in the 3.3 V PCI environment.

4.0 Related Documents

- The *PCI Local Bus Specification*, Revisions 2.0 and 2.1 can be obtained from the PCI SIG at the following address:

  PCI Special Interest Group
  2575 NE Kathryn St. #17
  Hillsboro, OR 97124

  FAX: 503-693-8344

  PCI SIG Web Page: http://www.pcisig.com
  Electronic Mail Reflector:
  pci-sig-request@znyx.com “subscribe”

- The following documents can be obtained from the IEEE at http://www.ieee.org:

  1275-1994 Standard for Boot (Initialization, Configuration) Firmware Core Requirements and Practices

  PCI Bus Binding to IEEE 1275-1994

  Draft Std P1275/D7 (no longer available)

- The *PCI Bus Supplement: 2.0 Aug. 6, 1996* is available from the Open Firmware Working Group at http://playground.sun.com/1275/bindings/pci/pci2_0.ps