CHAPTER 8

PCI

This chapter presents the requirements and recommendations for PCI host controllers and peripherals under the Microsoft Windows family of operating systems.

See also information about PCI-to-PC Card bridges in the “PC Cards” chapter in Part 3 of this guide.

Version 1.1:
Includes changes to items 2, 10, 11, 15, and References and Resources for PCI, as previously published in the PC 97 FAQ on http://www.microsoft.com/hwdev/pc97.htm and and the PC 97 OnNow Requirements on http://www.microsoft.com/hwdev/desguid/onnowpc97.htm

Contents

Overview of PCI Design Issues

Microsoft supports the PCI effort within the PC industry. The effort to support PCI has uncovered related issues during operating system development and testing. Microsoft has developed some solutions for these in Windows 95 and Windows NT. Microsoft is also working on solutions for future versions of Windows and Windows NT. This chapter describes some of these issues and solutions, especially with regard to hardware design requirements.

The PCI architecture has become the most common method used to extend the PC for add-in adapters. Windows 95 and Windows NT use the basic PCI infrastructure to gain information about devices attached to the PCI bus. The ability of PCI to supply such information makes it an integral part of the Plug and Play architecture in Windows.

PCI Basic Requirements

This section summarizes the basic design requirements for PCI.

1. PCI Local Bus Specification 2.1 (or higher)
Required
All cards, bridges, and devices that use PCI must be designed to meet the requirements defined in PCI specification version 2.1 or higher.

2. No ghost cards in system
Required
A computer must not include any “ghost cards,” which are cards that do not correctly decode transaction Type 1 cycles and so appear on all buses because the ghost card’s PCI Configuration Space appears in more than one bus/device/function coordinate. This also includes, for example, devices that ignore some Type 0 transaction bits and therefore appear at multiple device/function addresses.

A PCI card should be visible through hardware configuration access at only one bus/device/function coordinate.

Version 1.1 Clarification:
A computer must not include any ghost cards, which are cards that do not decode the type 1/ type 0 indicator. Such a card will appear on bus 0, as do all the buses behind it that use the same IDSEL. Notice that it is acceptable, as defined in PCI Local Bus Specification, Revision 2.1 or higher (PCI 2.1), for a single-function card to decode the IDSEL and AD[1::0] pins and not decode AD[10::8] if the card does not have bit 7 set in the header type.

This requirement also excludes, for example, devices that ignore some type 0 transaction bits and therefore appear at multiple device/function

addresses. A PCI card should be visible through hardware configuration access at only one bus/device/function coordinate. (Change date: September 12, 1997)

3. Standard method used to close BAR windows on nonsubtractive decode PCI bridges

Required

Setting the Base Address Register (BAR) to its maximum value and the limit register to zeroes should effectively close the I/O or memory window referenced in that bridge BAR.

4. PCI docking through a bridge connector
Recommended

Microsoft recommends docking through a bridge connector, with the actual bridge on the docking station (not on the mobile unit). The bridge can be positive or subtractive decoding. The bridge should create a new bus number so that devices behind the bridge are not on the same bus number as other devices in the system.

After a warm dock, the BIOS should not configure the bridge nor any other devices in the docking station. That is the responsibility of the operating system.

Microsoft recommends putting the PCI-to-ISA bridge on the docking station, and having no PCI-to-ISA bridge on the mobile unit. Mobile PCs typically have no ISA expansion slots, and the ISA devices on the mobile PC can be controlled by the Plug and Play interface. For more information on requirements for docking station systems, see the “Basic PC 97” chapter in Part 2 of this guide.

Notice that implementing delayed transactions for PCI-to-PCI and PCI-to-ISA docking bridges are required in the PCI 2.1 specification only when certain timing conditions are not met. For PC 97 design requirements, the PCI 2.1 specification is interpreted to mean that delayed transactions are required only when “targets cannot complete the initial data phase within the requirements of this specification,” as stated in the PCI 2.1 specification. Delayed transactions are a hardware-related timing issue (and will provide a performance advantage) but are not related to operating system requirements.

PCI Controller Requirements

This section summarizes PCI standards for controllers.

5. System board bus complies with PCI 2.1 (or higher)
Required

The system board bus hardware should comply with the written specifications. The bus design must fully implement all bus requirements on every expansion card connector.

6. Allow bus master privileges for all connectors
Required

To ensure full Plug and Play functionality on a PCI bus with expansion cards, all PCI connectors on the system board must be able to allow any PCI expansion card to have bus master privileges.

7. ISA Write Data Port address propagated to ISA bus at power-up
Required

If the system uses an ISA bus in conjunction with a PCI bridge, the Plug and Play ISA Write Data Port address must be propagated at all times through the bridge to all ISA buses that might contain external ISA Plug and Play cards. The address must be propagated at power-up and system reset. This will ensure that the system can identify, isolate, and configure external Plug and Play ISA cards plugged into the ISA bus during the boot process.

8. Writable PCI Configuration Space bits not shared by CardBus controllers
Required

CardBus controllers are 32-bit multifunction PCI devices, and Windows treats each function of a multifunction PCI device as an independent device. As such, there can be no sharing between functions of writable PCI Configuration Space bits (such as the Command register).

Notice that the PC Card 16-bit Interface Legacy Mode Base Address Register (offset 44h in the Type 2 PCI header) is the only exception to this requirement. This register must be shared between the two functions, as they must share the same registers for compatibility with the Exchangeable Card Architecture defined by Intel (ExCA programming model, as defined in the "Yenta" specification).

For more information about design requirements for CardBus controllers, see the "PC Card" chapter in Part 3 of this guide.

Plug and Play for PCI Controllers and Peripherals

This section summarizes the Plug and Play requirements for PCI devices.

Note The recommendation for PC 95, "PCI-to-PCI bridge configured but not enabled by the BIOS," is replaced in PC 97 by the requirement in this section for BIOS support for IRQ routing.

9. PCI 2.1 Configuration Space for Plug and Play device identifier
Required

The PCI specification describes the configuration register space used by the system to identify and configure each device attached to the bus. This configuration space is made up of a 256-byte field for each device and contains sufficient information for the system to identify the capabilities of the device. Configuration of the device is also controlled from this register space.

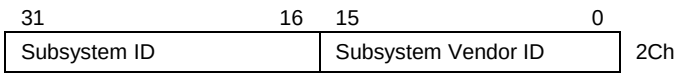
The configuration register space is made up of a header region and a device-dependent region. Each configuration space must have a 64-byte header at offset 0. All of the device registers that the device circuit uses for initialization, configuration, and catastrophic error handling must fit within the space between byte 64 and byte 255. All other registers that the device uses during normal operation must be located in normal I/O or memory space. Unimplemented registers or reads to reserved registers must complete normally and return zero (0). Writes to reserved registers must complete normally, and the data must be discarded.

10. PCI Specification 2.1 Subsystem IDs

Required

The following diagram shows the two registers added to the Configuration Space Header for PCI Specification 2.1. Although these registers are only recommended in the PCI specification, they are mandatory for PC 97. Support of these registers requires non-zero values to be populated for both the Subsystem ID and Subsystem Vendor ID. These values must be populated prior to the system BIOS or system software accessing the PCI configuration space, as stated in the PCI 2.1 specification. The recommended method for populating these registers is by using serial ROMs.

New registers in configuration space header for PCI version 2.1



These fields are important for the correct enumeration of a device. When the Subsystem fields are populated correctly for your adapter, Windows 95 can differentiate between adapters based on the same PCI chip.

The Subsystem IDs also allow Windows to load system miniports for system board devices. Thus, Subsystem IDs are also required on system board devices, but not for core chip sets.

The exceptions to this requirement are PCI-to-PCI bridges and core chip sets.

Version 1.1 Clarification:
PCI 2.1 specifies that Subsystem Vendor IDs are read-only. As such, the current HCT will fail any card whose Subsystem Vendor ID is writeable. If a device vendor wants to be able to program the Subsystem Vendor ID using BIOS, there are two designs that are logo compliant:

- Make a copy of the Subsystem Vendor ID in PCI user defined space. Writes to this location will change both the copy and the Subsytem Vendor ID field. Writes to the Subsystem Vendor ID are discarded.
- Make a write-enable bit in the PCI user-defined space. The BIOS can turn this bit on, change the Subsystem Vendor ID, and then turn it off.

Either of these methods guarantee that third-party software or drivers conforming to PCI 2.1 will function properly.

Notice for display adapters: Multimonitor support allows Display class devices to be intitialized independently of the system intialization process. For this reason, system-board and add-on display devices cannot use the VGA BIOS POST routine to populate the Subsystem Vendor ID because the device's POST code might not be executed until later in the process, after device enumeration occurs. For system board devices, the system BIOS should populate the Subsystem Vendor ID at power on. Add-on display adapters should provide a method for populating the Subsystem Vendor ID at the point when power is applied and the device is intialized to the state that it is ready for POST.

11. Configuration space populated correctly

Required

Windows 95 places extra constraints on a few configuration registers and has uncovered some problem usage of other registers. Microsoft provides a program named Pci.exe to help you debug the use of your configuration space. This program is available on the Microsoft FTP server, as described in the “Resources and References for PCI” section at the end of this chapter.

The following items are specific requirements for the configuration space:

- Populate the Class Code Register (09h) for all devices.

  Follow the Base Class, Sub-Class, and Programming Interface values outlined in the PCI 2.1 specification.

- Devices must not fill Base Address Registers with random values.

  Devices must not attempt to fill in random values to these registers. See the PCI 2.1 specification for the correct usage of these registers. Notice that Base Address Registers (10, 14, 18, 1C, 20, and 24h) should return zero (0) if they are not used, indicating that no memory or I/O space is needed.

Also, for performance reasons, it is recommended that runtime registers for PCI devices should not be placed in the configuration space.

Version 1.1 Clarification:

The configuration registers must contain a correct base class and subclass as defined in Appendix D of PCI 2.1. If you have a new device type not listed in Appendix D, you must request a new encoding from the PCI Special Interest Group (SIG) rather than using an existing base class or subclass that does not directly apply for your device. For information about obtaining a new encoding for a device type, see the PCI SIG web site at http://www.pcisig.com. (Change date: August 6, 1997)

12. BIOS support for IRQ routing, for x86-based systems

Required

The BIOS must support IRQ routing using the method defined in the PCI 2.1 IRQ Routing engineering change request (ECR). The addition of IRQ routing support in PCI 2.1 allows the operating system better control over PCI interrupt routing, allowing the operating system to configure bridges and devices beyond bridges.

Note The BIOS should support a "BIOS Flag" that the OEM sets, selecting by inference the level of PCI support within the operating system, depending on which operating system is installed in the factory or selected by the end user. The BIOS should allow the OEM to select between MS-DOS/Windows 3.11, Windows 95, and future Windows versions, thereby informing the BIOS about how much of the PCI configuration and enabling work should be done.

13. BIOS does not configure I/O systems to share PCI interrupts

Recommended

This applies to boot devices configured by the BIOS on x86-based systems. All other devices should be configured by Windows. Design the BIOS so that it does not configure the I/O systems in the PC to share PCI interrupts for boot devices.

Windows 95 does not support sharing an IRQ between real-mode and protected-mode code within the I/O subsystem—an example could be, an NDIS 2 driver

(real mode) and a SCSI miniport driver (protected mode) for two PCI devices that share the same IRQ. The problem is the IRQ needs to be reflected to real mode for the NDIS 2 driver to work. However, if the IRQ is reflected to real mode, the real-mode SCSI driver (which usually is not called because Windows 95 takes over in protected mode) might touch the hardware, which would cause the SCSI miniport to be confused. Windows 95 solves this problem by switching everything either to protected mode or by falling back to real mode.

14. BIOS configures device IRQ and writes to the Interrupt Line Register
Required

This applies to boot devices configured by the BIOS on x86-based systems. All other devices should be configured by Windows, because after an IRQ is assigned by the system BIOS, Windows cannot change the IRQ if necessary. If the BIOS assigns the interrupt and Windows needs it for another device, an IRQ sharing problem occurs.

The BIOS must configure the device IRQ and write the IRQ into the Interrupt Line Register 3Ch, even if the BIOS does not enable the device. This way, Windows can still enable the device with the known IRQ, if possible, at configuration time. Otherwise, if the IRQ is not known, there is no way that Windows can enable the device, even if all other resources are available.

Power Management for PCI Controllers and Peripherals

This section summarizes the specific power management requirements for PCI.

15. Compliance with PCI Bus Power Management Interface Specification
Required

The PCI bus, any bridges on the bus, and all devices on the PCI bus must comply with the PCI Bus Power Management Interface Specification. This specification provides definitions of the OnNow device power states (D0−D3) for PCI devices. It also covers bus functionality expected in each power state and the mechanism for signaling wakeup events over the bus. For PC 97, PCI bus implementations must support all four device and bus power states, as well as the standard wakeup mechanism.

At a minimum, integrated PCI devices must support the D0 and D3 power states. See the device class sections for additional power state and wakeup requirements.

This specification will allow Windows to power manage PCI devices individually to conserve power and to allow PCI peripherals to wake up the PC system when the need arises.

Version 1.1 Clarification:

Logo compliance testing of systems and devices for PCI power management, based on *PCI Bus Power Management Interface Specification, Revision 1.0,* will begin **April 1, 1998.**

The requirement for PCI power management means that the system must support PCI add-on adapters that implement the PCI power management specification, including support for the B0 bus state and the wake-up signal (#PME) on all PCI slots.

It is an acceptable alternative for embedded PCI devices (on the system board) to use ACPI for power-state and wake-up control instead of the new PCI specification. Notice that the B2 and B3 bus states are not required as defined in the *PCI Bus Power Management Interface Specification.* Systems are not required to implement independent switching of PCI bus power.

For PCI add-on adapters, including AGP devices, compliance with the *PCI Bus Power Management Interface Specification* is required beginning **April 1, 1998,** including requirements for the Configuration Space registers and the device state (D*x*) definitions.

Design Features for PCI Devices

Specific requirements related to devices that use PCI are defined in the following chapters:

- Requirements for dual IDE adapters that use PCI are defined in "ATA and ATAPI" in Part 3 of this guide.
- Requirements for graphics devices that use PCI are defined in "Graphics Adapters" in Part 4 of this guide.
- Requirements for audio implementations that use PCI are defined in "Audio Components" in Part 4 of this guide.

References and Resources for PCI

The following lists some of the services and tools available to help build hardware that works with Windows operating systems.

Microsoft testing tools, specifications, and information
http://www.microsoft.com/hwtest/
http://www.microsoft.com/hwdev/busbios/pcidsgn.htm
ftp://ftp.microsoft.com/developr/drg/plug-and-play/pci/pci.exe
E-mail: pciinfo@microsoft.com

PCI Special Interest Group
For information about current revisions of all PCI specifications and about joining the PCI Special Interest Group, contact the PCI-SIG:
Phone: (800) 433-5177
http://www.pcisig.com/

See also the papers available from the PCI-SIG, including the Compliance Checklist Rev. 2.1, at http://www.teleport.com/~pc2/reldocs.htm

PCI Bus Power Management Interface Specification
http://www.microsoft.com/hwdev/onnow.htm

"Yenta" specification: PCI to PCMCIA CardBus Bridge Register Description
Published by Intel Corporation, available to all "Yenta" members.
Phone: (800) 879-4683

Version 1.1 References Update:
Advanced Configuration and Power Interface Specification, Revision 1.0
http://www.teleport.com/~acpi/

"Efficient Use of PCI," Platform Architecture Labs, Intel Corporation
http://support.intel.com/oem_developer/chipsets/pci/general/pci001.htm

"IDs and Serial Numbers for Plug and Play" and other related articles
http://www.microsoft.com/hwdev/busbios/

Implementing Legacy Audio Devices on the PCI Bus
http://www.intel.com/pc-supp/platform/ac97/wp/leg_pci.htm

Microsoft testing tools, specifications, and information
E-mail: pciinfo@microsoft.com
http://www.microsoft.com/hwtest/
http://www.microsoft.com/hwdev/busbios/
ftp://ftp.microsoft.com/developr/drg/plug-and-play/pci/pci.exe

PCI Bus Power Management Interface Specification, Revision 1.0
PCI Local Bus Specification, Revision 2.1 (PCI 2.1)
PCI to PCI Bridge Specification, Revision1.0.
PCI SIG
Phone: (800) 433-5177
http://www.pcisig.com/

PCI to PCMCIA CardBus Bridge Register Description (Yenta specification)
PCMCIA
2635 North First Street, Suite 209
San Jose, CA 95134 USA
Phone: (408) 433-2273
Fax: (408) 433-9558
E-mail: office@pcmcia.org
http://www.pc-card.com/

Checklist for PCI

PCI Basic Requirements

1. PCI Local Bus Specification 2.1 (or higher)
Required

2. No ghost cards in system
Required

3. Standard method used to close BAR windows on nonsubtractive decode PCI bridges
Required

4. PCI docking through a bridge connector
Recommended

PCI Controller Requirements

5. System board bus complies with PCI 2.1 (or higher)
Required

6. Allow bus master privileges for all connectors
Required

7. ISA Write Data Port address propagated to ISA bus at power-up
Required

8. Writable PCI Configuration Space bits not shared by CardBus controllers
Required

Plug and Play for PCI Controllers and Peripherals

9. PCI 2.1 Configuration Space for Plug and Play device identifier
Required

10. PCI Specification 2.1 Subsystem IDs
Required

11. Configuration space populated correctly
Required

12. BIOS support for IRQ routing, for x86-based systems
Required

13. BIOS does not configure I/O systems to share PCI interrupts
Recommended

14. BIOS configures device IRQ and writes to the Interrupt Line Register
Required

Power Management for PCI Controllers and Peripherals

15. Compliance with PCI Bus Power Management Interface Specification
Required