PCI Hot-Plug Specification

Revision 1.0

October 6, 1997

Revision History

Revision	Issue Date	Comments
1.0	October 6, 1997	Original issue.

Contents

Figures

Tables

Chapter 1 Introduction

1.1 Objectives of the PCI Hot-Plug Specification

The primary objective of this specification is to enable higher availability of file and application servers by standardizing key aspects of the process of removing and installing PCI adapter cards while the system is running. Although these same principles can be applied to desktop and portable systems using PCI buses, the operations described here target server platforms.

To expedite market acceptance of hot-plug capability, the burden of hardware changes has been placed on the platform vendor, not the adapter vendor. This specification describes a new class of platforms, *not* a new class of adapter cards. Electrical requirements for the adapter card, beyond those already stated in the *PCI Local Bus Specification,* have been kept to an absolute minimum to guarantee, as far as possible, that existing adapter cards will be hot-pluggable. It is expected that the vast majority of PCI adapter cards designed prior to the publication of the *PCI Hot-Plug Specification* will meet the requirements of this specification.

It is the expressed intent of this document to do the following:

1. Specify what an adapter-card vendor must do to enable an adapter card to be hot-pluggable.
2. Specify required features of the hot-plug platform, so operating-system vendors can be assured of a minimum feature set. The list of required platform features has been kept short to enable the implementation of cost-sensitive PCI hot-plug server platforms.
3. Establish a framework of terminology and architecture for hot-plug system hardware and software, and assign responsibility for each component and operation either to the platform vendor, the operating-system vendor, or the adapter-card vendor.

It is expressly *not* the intent of this document to do the following:

1. Specify the detailed implementation of the PCI platform. To the greatest extent possible, the design of the PCI platform has been left flexible to allow for product differentiation.

2. Specify the implementation or structure of hot-plug software routines. Concepts and features are presented, but implementation is strictly determined by each operating-system vendor.

2. Specify higher-level operating-system behavior, such as what should happen to a file system while its controller card is being replaced. Such functions may vary from one operating system to another, and are controlled by each operating-system vendor.

3. Specify higher-level hot-plug operations such as like-for-like replacement of cards, versus adding a new card. The hardware and low-level software components described here are required by all hot-plug systems, and can result in various implementations at the discretion of the operating-system vendor.

1.2 Areas of Standardization

Figure 1-1 illustrates some of the hardware and software components of a hot-plug system. In the figure the *Conventional System Software* box represents applications, system management functions, the operating system, and device drivers for peripherals, including PCI adapter cards, which are present in conventional, non-hot-plug systems. The *Conventional Platform Hardware* box includes the CPU(s) and peripheral devices. The cloud in the center represents software and hardware required to control power and bus connections on PCI slots that accept standard PCI adapter cards.

This specification allows the software and hardware *within* the cloud to change, and standardizes the two *interfaces* of the cloud shown in Figure 1-1. The first standard interface is the hardware interface between the platform and the adapter card. In most respects this interface is a standard PCI interface. This document will describe additional aspects of powering up and down a card plugged into a PCI connector that are beyond the scope of the *PCI Local Bus Specification*.

The second standard interface shown in Figure 1-1 is the software interface between the higher-level system software and the platform-specific hot-plug software. The information *content* of the requests and responses that cross this interface are specified in this document. However, since this is a device-driver interface, the information *format* may vary from one operating system to another and is controlled by each operating-system vendor.

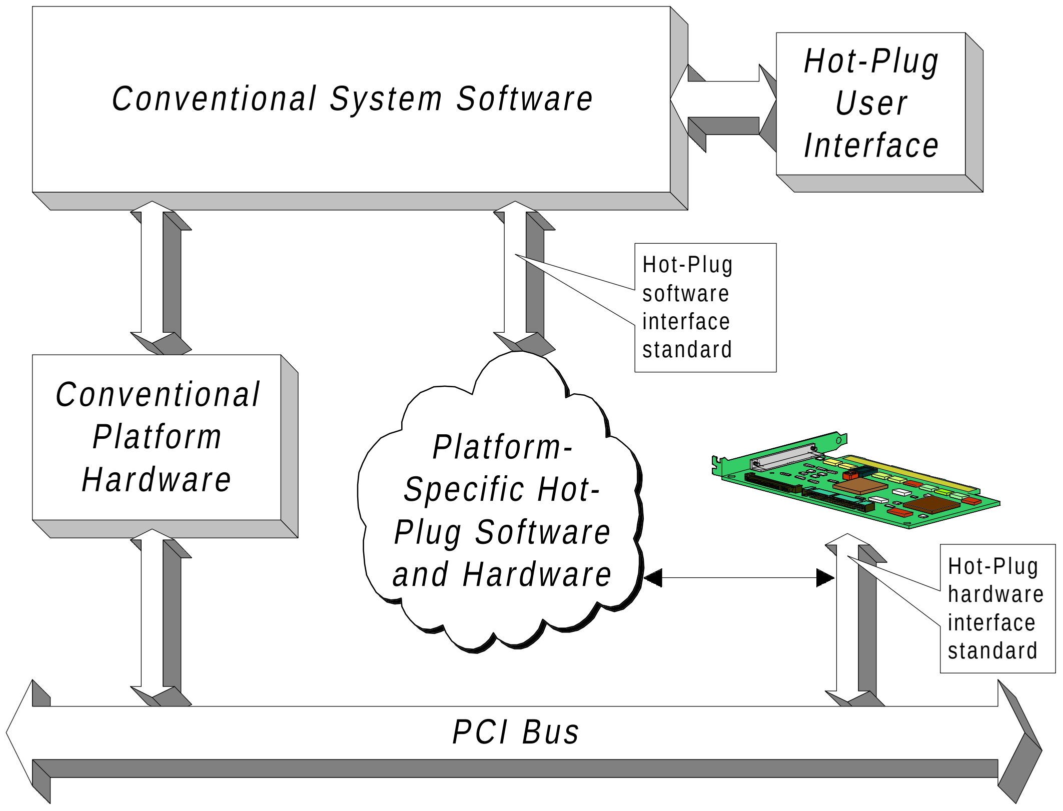

Conventional System Software
Hot-Plug User Interface
Hot-Plug software interface standard
Conventional Platform Hardware
Platform-Specific Hot-Plug Software and Hardware
Hot-Plug hardware interface standard
PCI Bus

Figure 1-1: Standardized Hot-Plug Software and Hardware Interfaces

1.3 Assumptions

1.3.1 PCI Adapter Card Failure

The PCI bus inherently is not a fault-tolerant bus. A failure of a PCI adapter card can have a wide range of effects on system behavior. For example, the failure might be as harsh as preventing further PCI bus transactions or corrupting the contents of main memory. Or the failure might be as localized as losing a single network connection. If a failure of a device compromises the integrity of the PCI bus or the software system, then the only recourse is to restart the system. It is assumed throughout the remainder of this specification that device failures that compromise the integrity of the system are beyond the scope of this specification. Only failures that do not compromise the integrity of the system are relevant to this specification.

1.3.2 Orderly Removal and Insertion

The operating systems that are widely used throughout the PCI industry are not generally designed to handle the unexpected removal of devices. Therefore, the operating-system vendor and platform vendor define the sequence of user actions and system management facilities that will inform the operating system that removal of a card is desired. The actual removal cannot occur until the software system acknowledges that it is ready.

Furthermore, existing PCI adapter cards are not generally designed to be connected to a slot that has power applied. Therefore, the operating-system vendor and platform vendor define a sequence of user actions and system behavior that will guarantee that power is always removed from a slot before a card is inserted.

Inserting or removing an adapter card without following the proper sequence may lead to unpredictable results, including data corruption, abnormal termination of the operating system, or damage to card or platform hardware. Unless otherwise stated, it is assumed throughout the remainder of this specification that the user always follows the proper removal and insertion sequence.

1.3.3 Programmatic Access to the Hot-Plug Service

This specification describes the required function of high-level software, but not its form or structure. The operating-system vendor exclusively controls the form and structure. However, it is expected that the operating-system vendor will provide programmatic interfaces to hot-plug functionality. These interfaces would enable additional software to add other function such as:

- Remote power down of intermittent or bad PCI cards
- Diagnostic packages
- Remote/Local tracking/display of system resources and status
- Remote/Local notification of configuration changes
- Remote/Local control of hot-plug operations

It is assumed throughout the remainder of this specification that whenever a user interface is described, that the operating system will provide an equivalent programmatic interface. See Section 4.1.7, for more details on the user interface.

1.4 Notational Conventions

Throughout this specification, references to 33-MHz operation refer to systems that operate with clock speed from 0 to 33 1/3 MHz, and references to 66-MHz operation refer to systems that operate with clock speed greater than 33 1/3 MHz up to 66 2/3 MHz.

Signal names are indicated with this **bold** font.

Chapter 2
Overview of PCI Hot Plug

2.1 Definitions of Terms

Definitions of the terms used in this specification are listed below. Most terms are capitalized throughout the remainder of this specification to call attention to their special definitions.

Adapter Card	A card designed in accordance with the *PCI Local Bus Specification, Revision 2.0*[1] or *Revision 2.1*[2] to be plugged into a PCI connector. This includes cards that are 32- or 64-bits wide, operate at 33 MHz or 66 MHz, and use 3.3 V or 5 V signaling. The card could contain a single PCI device or multiple devices behind a PCI-to-PCI bridge.
Adapter Driver	The software driver which controls the Adapter Card. It is generally supplied by the Adapter Card vendor.
Attention Indicator	A physical indicator, located to draw the attention of the user to a particular slot. The Platform is required to provide one Attention Indicator per hot-plug slot. The state of the Attention Indicator is determined by the Hot-Plug Service.

[1] *PCI Local Bus Specification, Revision 2.0,* PCI Special Interest Group, 1993.

[2] *PCI Local Bus Specification, Revision 2.1,* PCI Special Interest Group, 1995.

Hot-Plug Controller	Hardware supplied by the Platform vendor that controls the electrical aspects of powering up and down a PCI slot. A single Hot-Plug Controller will typically control more than one slot. A hot-plug Platform may contain more than one Hot-Plug Controller.
Hot-Plug Primitives	Specific requests issued by the Hot-Plug Service to the Hot-Plug System Driver to determine the status of, or to initiate changes to, a hot-plug slot in the Platform.
Hot-Plug Service	High-level software that has overall control of hot-plug operations. It includes a user interface, and can issue requests to the operating system to quiesce adapter activity, and issue Hot-Plug Primitives to the Hot-Plug System Driver to turn slots on and off. The Hot-Plug Service is unique to each particular operating system and is generally supplied by the operating-system vendor.
Hot-Plug System Driver	Software driver that controls and monitors the Hot-Plug Controller hardware. If there is more than one Hot-Plug Controller in a Platform, then more than one Hot-Plug System Driver may be required. The Hot-Plug System Driver is supplied by the Platform vendor.
Logical Slot Identifier	A parameter of a Hot-Plug Primitive that uniquely identifies a particular slot. The operating-system vendor specifies the encoding of this parameter. For example, some may use PCI bus and device number, while others may use physical slot numbers.
Physical Slot Identifier	A designation assigned by the Platform vendor that uniquely identifies a physical slot. For example, in a single-chassis system it could simply be a slot number. In a multiple-chassis system it could be a combination of chassis and slot number.
Platform	The collection of hardware in which the PCI bus resides. Generally includes the power supply, one or more CPUs, a host-bus-to-PCI bridge, and various peripherals such as disk drives, a keyboard, etc.

Platform Configuration Routine	Software responsible for initializing the PCI Configuration Space header for a newly installed Adapter Card. The operating-system vendor specifies whether this routine is run from the Hot-Plug Service, or from the Hot-Plug System Driver.
quiesce	Before an Adapter Card in a slot can be removed, adapter activity must be quiesced. When adapter activity is quiesced, the Adapter Driver will not send any PCI operations to the Adapter Card, and the Adapter Card will not initiate any interrupts or bus master activity. See Section 4.1.2 for more details.
slot	A location designed to accept an Adapter Card. This is the basic unit of hot-pluggability. Individual slots must be isolated from the rest of the Platform for reliable insertion and removal of an Adapter Card.

2.2 System Components

Figure 2-1 illustrates the hardware and software components of a typical hot-plug system and how they are connected.

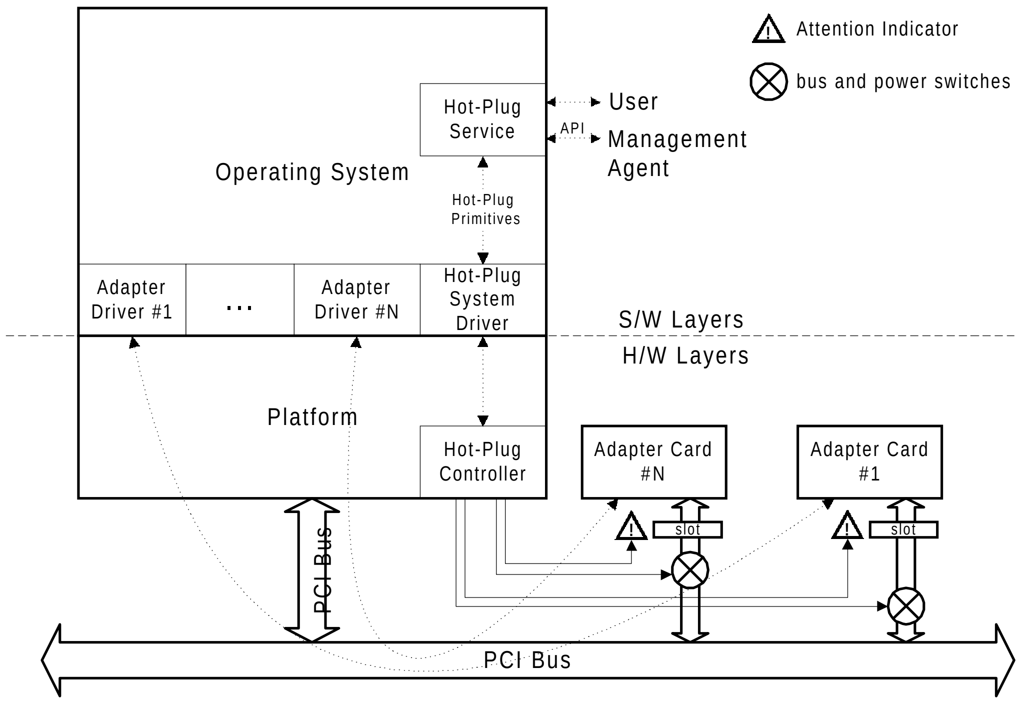


Figure 2-1: System Block Diagram

The electrical capabilities that the Platform vendor must provide are defined in Chapter 3. They are summarized as follows:

- The ability to isolate an individual slot from its associated PCI bus.
- Individual control of the power and control signals for a slot such that the slot is isolated and powered down during Adapter Card insertion or removal.
- Power-up and power-down sequences for hot-plug slots that meet the electrical requirements of the *PCI Local Bus Specification*.
- An Attention Indicator to draw the user's attention to a particular slot.

The Platform and system software (operating system and drivers) must define the sequence of user actions that defines a proper hot-insertion or removal sequence. This proper sequence will define the actions that a user is required to perform to inform the operating system that an insertion or removal of an Adapter Card is desired. The physical insertion or removal will not occur until the system software has:

- Quiesced any operating system services or drivers using the Adapter Card.
- Isolated and powered down the slot.
- Indicated to the user that it is ready.

If an Adapter Card is inserted or removed without following the proper sequence, this is considered an improper operation and error conditions and other unexpected events may occur, including data corruption and hardware damage.

2.3 Illustration of a Hot-Plug Sequence

The following hot-plug scenarios for removing and installing an Adapter Card are given for example and overview purposes, and do not represent requirements upon either Platforms or operating systems. For any given system the Hot-Plug Service, the Hot-Plug System Driver, and the Hot-Plug Controller each control different portions of the actual order.

2.3.1 Hot Removal

The following general sequence of steps is necessary to remove an Adapter Card from a slot that is powered up:

1. The user determines that an Adapter Card must be removed or replaced, and notifies the Hot-Plug Service of the desire to remove the card from its slot. Examples of notification methods include issuing a console command or activating a switch designed for this purpose.
2. The Hot-Plug Service uses operating system functions to quiesce the appropriate Adapter Driver instance(s) and the Adapter Card.
3. The Hot-Plug Service issues a Hot-Plug Primitive to the Hot-Plug System Driver to turn off the appropriate slot.

4. The Hot-Plug System Driver uses the Hot-Plug Controller to do the following:
    a) Assert **RST#** to the slot and isolate the slot from the rest of the bus, in either order.
    b) Power down the slot.
    c) Change the optional slot-state indicator to show that the slot is off.
5. The Hot-Plug Service reports to the user that the slot is off.
6. The user removes the Adapter Card.

2.3.2 Hot Insertion

A slot must be powered down and isolated from the bus before an Adapter Card can be inserted. The process of making a slot ready for insertion can vary from one Platform and operating system to another. The following general sequence of steps is necessary to insert an Adapter Card into a slot after it is powered down and ready for insertion:

1. The user inserts the new Adapter Card.
2. The user notifies the Hot-Plug Service to turn on the slot containing the new Adapter Card.
3. The Hot-Plug Service issues a Hot-Plug Primitive to the Hot-Plug System Driver to turn on the appropriate slot.
4. The Hot-Plug System Driver uses the Hot-Plug Controller to do the following:
    a) Power up the slot.
    b) Deassert **RST#** on the slot and connect the slot to the rest of the bus, in either order.
    c) Change the optional slot-state indicator to show that the slot is on.
5. The Hot-Plug Service notifies the operating system that the new Adapter Card is installed, so the operating system can initialize the adapter and prepare to use it.
6. The Hot-Plug Service notifies the user that the card is ready.

Chapter 3
Electrical Requirements

3.1 Platform Requirements

3.1.1 Supplementary Hardware Requirements

A hot-plug Platform must provide the following features beyond what is specified in the *PCI Local Bus Specification*:

1. At least one Hot-Plug Controller.
2. Slot-specific power switches. The Platform must provide a means for the software to remove all power from a slot while the rest of the Platform is running.
3. Slot-specific bus isolation devices. The Platform must provide a means for the software to isolate all the signals on a slot from the bus, while the rest of the Platform is running. The Platform is required to meet all the AC timing specifications of the *PCI Local Bus Specification* at the slot when that slot's isolation devices are in the conducting mode.
4. Slot-specific **RST#**. The **RST#** pin for each slot must be independently controlled.
5. Slot-specific Attention Indicator. Each hot-plug slot must have an Attention Indicator that is located to draw the attention of the user to that slot; e.g., an LED located near the slot. The state of the Attention Indicator is controlled by the software.
6. A means for software to determine the state of the **PRSNT[1:2]#** pins for each hot-plug slot regardless of whether the slot is on or off and connected or isolated.
7. A means for software to determine whether the bus is currently operating at 66 MHz.

8. A means for software to determine the state of the **M66EN** pin for each 66-MHz hot-plug slot, while the slot is isolated from the bus. A Platform is not required to implement this means if either of the following is true:
    a) The Platform hardware guarantees that a 33-MHz card is not connected to the bus if the bus is operating at 66 MHz.
    b) The bus includes no devices other than the source bridge and a single slot, and the Platform hardware or the Hot-Plug System Driver guarantees that the bus is operating at 33 MHz before connecting a 33-MHz card.
9. A Physical Slot Identifier associated with each hot-plug slot.
10. Information accessible by the software that the operating-system vendor can use to define translation mechanisms between Physical Slot Identifier and the PCI bus and device number associated with that slot (see Section 4.1.8).

A hot-plug Platform may provide the following optional features beyond what is specified in the *PCI Local Bus Specification*:

1. Slot-specific slot state indicator. Each hot-plug slot may have an indicator that shows whether the slot is on or off. If implemented, this indicator must reflect the current slot state as controlled by the Setting Slot Status Hot-Plug Primitive. See Section 4.2.2.2.

    An implementation that combines the slot state indicator with the Attention Indictor into a single device is permitted. In such implementations, the attention state is required to take precedence over the slot state, if both states cannot be shown simultaneously. For example, a single LED might be implemented that is on if the slot if on, off if the slot is off, and blinking in the attention state whether the slot is on or off.
2. Slot-specific power fault detector. The Platform may include the capability of detecting when an Adapter Card exceeds the power limits of a slot (see Section 3.3), and automatically turning off the slot.
3. System power budget. The Platform vendor may implement a means of tracking power usage and determining whether there is sufficient power available to turn on an additional Adapter Card.

The Platform vendor is permitted to implement other features, but is responsible for providing all necessary software support for them.

3.1.2 Platform Initialization

If no hot-plug software (other than Platform firmware) is loaded onto a hot-plug Platform, and no hot-insertion or removal operations are performed, the Platform must behave as if it were not a hot-plug Platform. After power is initially applied to a hot-plug system, at the time when the operating system begins loading, Platform firmware must ensure that all Attention Indicators are off, and the hot-plug slots are in a state that would be appropriate for loading non-hot-plug system software.

3.1.3 Turning On a Slot

When a slot is available for an Adapter Card to be inserted while the rest of the Platform is running, the power is removed from the connector, and the connector is isolated from the PCI bus. The Platform vendor must guarantee that, when turning on a slot, the slot will behave in full accordance with the *PCI Local Bus Specification*. Selected sections from the *PCI Local Bus Specification* related to power-up and power-down timing have been reproduced in Appendix A to assist the reader.

The first step in turning on a slot is turning on the power. The steady-state supply load and amount of decoupling capacitance on the Adapter Card is specified in Section 3.3. The Platform is required to turn on power to a slot containing an Adapter Card that meets these load requirements, in a way that does not interfere with the operation of the rest of the Platform, and that meets all the requirements of the *PCI Local Bus Specification*. Some voltage ramp characteristics are specified in Section 3.3. The Platform vendor determines all other voltage and current ramp characteristics.

After the supply voltages are stable, the Hot-Plug Controller must electrically connect the signals on the slot to the bus and deassert **RST#**. The Platform may connect the slot to the bus either before or after **RST#** is deasserted, but, in either case, must meet all the setup- and hold-time requirements of the *PCI Local Bus Specification* with respect to the rising edge of **RST#**.

Transactions between other devices may occur before and after the slot is connected and **RST#** is deasserted. The Platform may control the PCI bus at the time the bus is connected and the time **RST#** is deasserted to prevent interference with other transactions.

3.1.4 Turning Off a Slot

When a slot is turned off, the Platform is required to assert **RST#** to the slot and electrically isolate the slot from the bus, in either order, and then to remove power from the slot. Power-down timing must comply with the *PCI Local Bus Specification*. Refer to Appendix A for a copy of some relevant sections.

3.2 Adapter-Card Requirements

3.2.1 Noteworthy Requirements of the PCI Local Bus Specification

The following requirements of the *PCI Local Bus Specification* are of particular interest during hot-plug operations with an Adapter Card, since violations could potentially go unnoticed in a non-hot-plug system.

1. **PRSNT[1:2]# connection.**[3] One or both of these pins must be grounded by Adapter Cards to indicate that a card is present in the slot and how much power the card requires. **PRSNT[1:2]#** pins not grounded must be left unconnected. Hot-plug Platforms read these pins to determine which slots are occupied and how much power the card requires.
2. **All outputs float when RST# asserted.**[4] All bus outputs from the Adapter Card must be asynchronously tri-stated, or floated if open drain, when **RST#** is asserted. In hot-plug Platforms, the bus may be running transactions between other active devices while a device being hot-plugged has its **RST#** pin asserted.
3. **PCI interface state machines' response to RST#.**[5] All PCI interface state machines must be held in their reset state as long as **RST#** is asserted. In a hot-plug Platform, a slot with **RST#** asserted may be connected to an active bus carrying transactions between other devices. These transactions addressing other devices should have no effect on the PCI interface of the device being reset.

   Once **RST#** is deasserted, the PCI interface state machines should remain in the IDLE state until an address phase containing a valid address for the device is encountered.

Implementation Note: Slow Recognition of Deassertion of RST#

Some PCI devices may require an extended period of time to initialize the PCI interface after **RST#** is deasserted. For example, the device may need to wait for a local PLL to lock, or for configuration registers to initialize from a slow external device. If the initialization period takes more than a few clocks, such a device could come out of reset in the middle of a transaction between two other devices. (This would be true when the whole system was powered up as well as when a card was hot-plugged.) To avoid misinterpreting a transaction that is already in progress, such a device should not leave the reset state unless the PCI bus is idle.

[3] *PCI Local Bus Specification, Revision 2.1*, pp. 15, 142, 149.

[4] *PCI Local Bus Specification, Revision 2.1*, pp. 9, 139, 140.

[5] *PCI Local Bus Specification, Revision 2.1*, pp. 9, 236.

4. **Behavior of CLK prior to the rising edge of RST#.**[6] The *PCI Local Bus Specification* requires **CLK** to be stable a certain length of time before the rising edge of **RST#**. There are no specifications for **CLK** prior to this time. The Adapter Card must not depend on **CLK** to have any particular timing characteristics or valid logic levels prior to its setup to the rising edge of **RST#**.

Implementation Note: Examples of CLK Behavior Prior to Rising Edge of RST#

Some examples of how **CLK** might behave prior to its stable setup to **RST#** are: **CLK** could remain at a logic low level continuously from the time power is first applied until the bus signals are connected, then change instantaneously to the proper frequency. While **CLK** is being connected, pulse widths might be of any value, and logic levels may not be valid. Or **CLK** could begin toggling between ground and the ramping supply voltage, while the supply voltage is ramping. **CLK** could also oscillate at various frequencies for various periods of time, or continuously vary in frequency. Other examples are also possible.

5. **Preference for registers to be mapped into Memory Space rather than I/O Space.**[7] The *PCI Local Bus Specification* highly recommends the use of Memory Space over I/O Space, because I/O Space is limited and fragmented in PC Platforms. The limitation and fragmentation of I/O Space will be even more severe in hot-plug Platforms after the system has booted.

6. **Efficient resource requests.**[8] Although the *PCI Local Bus Specification* permits a device to consume more address space than it needs, it suggests decoding down to a 4 KB space for devices that require less than that amount of memory. Devices requiring I/O Space are required to use no more than 256 bytes per Base Address Register, and are required to provide memory mapping for the registers as well.

   Increasing the amount of resources a device requests will decrease the likelihood that the Platform Configuration Routine can configure new Adapter Cards added after the system boots. Decoding to the smallest address space possible (even smaller than 4 KB of memory and 256 bytes of I/O space) improves the likelihood that more devices can be added without rebooting the system. Wasting resources (requesting more resources than are actually required) makes it more difficult for other Adapter Cards to be configured. Devices should only request resources that will actually be utilized.

7. **Switching bus frequency between 33 MHz and 66 MHz.**[9] The *PCI Local Bus Specification* requires switching between the two ranges of bus frequency to occur in conjunction with a PCI reset. The Adapter Card cannot assume that such a bus frequency change will also be in conjunction with a power cycle.

---

[6] *PCI Local Bus Specification, Revision 2.1,* pp. 134, 140.

[7] *PCI Local Bus Specification, Revision 2.1,* p. 26.

[8] *PCI Local Bus Specification, Revision 2.1,* p. 197.

[9] *PCI Local Bus Specification, Revision 2.1,* p. 224, Table 7-3, Note 1.

3.2.2 Remote Power Sources

An Adapter Card may have a connection to another device that receives power from a source other than the Platform slot. The Adapter Card vendor must guarantee that the card's PCI interface is properly reset when **RST#** is asserted, whether the remote device is powered or not.

The Adapter Card vendor must further guarantee that the user is not exposed to any hazard, and that neither the Adapter Card nor the remote device will sustain any damage, if the Adapter Card or the remote device is powered for an indefinite length of time while the other is not.

3.2.3 Multiple-Card Sets

A multiple-card set is any group of Adapter Cards that is normally installed and removed together. A multiple-card set may be interconnected with sideband cables, and may appear in PCI Configuration Space as multiple PCI devices, or as a single PCI device (using the additional slots only for physical space or power-consumption reasons).

Multiple-card sets that do not require that power be applied and removed to the whole set simultaneously are accommodated by all hot-plug Platforms by using multiple single-card operations. Therefore, it is recommended that all multiple-card sets be designed such that they are not adversely affected if only a portion of the set is powered for an indefinite period of time. Furthermore, each card in the set must properly connect the **PRSNT[1:2]#** pins to guarantee that power will be applied to each of the slots (see Section 3.2.1).

This specification neither specifically allows nor disallows multiple-card sets that require simultaneous application and removal of power. Nor does this specification include a means for the software to uniquely identify which cards are in a multiple-card set. Although various places in the specification (e.g., the Hot-Plug Primitives) assume only single-slot operation, Platform vendors are permitted to accommodate such multiple-card sets and develop all the necessary hardware and software to support them.

3.3 Slot Power Requirements

Table 3-1 describes power requirements to ensure that an Adapter Card can be inserted into a running system.

The first requirement is the maximum operating current drawn by an Adapter Card, defined as current drawn by all loads other than the decoupling capacitors. These values are specified in the *PCI Local Bus Specification*, and are repeated in Table 3-1 for clarity. Adapter Cards that have a range of operating currents, or that have a dynamic or switching load must guarantee that the peak operating loads never exceed the values shown in Table 3-1, even while the Adapter Card's power is turning on. For example, a switching voltage regulator must maintain the peak operating current below the specified maximum even when initially charging the secondary decoupling capacitors.

The second requirement is the maximum decoupling capacitance on an Adapter Card. Adapter Card decoupling capacitance must not exceed the values shown in Table 3-1.

The third and fourth requirements involve power supply voltage characteristics when a slot is being turned on. The Platform must ensure that the supply voltage slew rate during power up is between the minimum and maximum values in Table 3-1. The Adapter Card must tolerate supply voltage slew rates between this minimum and maximum without suffering damage.

Table 3-1: Slot Power Requirements

Supply Voltage	Maximum Operating Current (Note 1, 2)	Maximum Adapter Card Decoupling Capacitance	Minimum Supply Voltage Slew Rate	Maximum Supply Voltage Slew Rate
+5 V	5 A	3000 µF	25 V/s	3300 V/s
+3.3 V	7.6 A	3000 µF	16.5 V/s	3300 V/s
+12 V	500 mA	300 µF	60 V/s	33000 V/s
-12 V	100 mA	150 µF	60 V/s	66000 V/s

NOTES:

1. This parameter is specified by the *PCI Local Bus Specification,*[10] and is included here for reference purposes only.
2. Combined maximum power drawn by all supply voltages in any one slot must not exceed 25 W.[11]

3.4 66-MHz Bus Considerations

The *PCI Local Bus Specification* requires that 66-MHz Adapter Cards operate when installed in a 33-MHz bus, and 66-MHz buses operate at 33 MHz when a 33-MHz Adapter Card is installed.[12] The Hot-Plug System Driver is required to guarantee that a slot containing a 33-MHz Adapter Card is never connected to a bus that is operating at 66 MHz. See Section 4.2.2.2 for more information.

The Platform vendor is solely responsible for determining when and how the bus operating frequency is switched between 33 MHz and 66 MHz. However, the *PCI Local Bus Specification* requires that switching between these frequencies occurs only in conjunction with a PCI reset.[13] Therefore, the Platform must not switch the bus operating frequency between 33 MHz and 66 MHz, and the Platform must not change the **M66EN** pin for a particular slot, if the following conditions are all true:

- An Adapter Card is present in the slot.
- The slot's power is on.
- The slot's **RST#** pin is deasserted.

[10] *PCI Local Bus Specification, Revision 2.1,* p. 142.

[11] *PCI Local Bus Specification, Revision 2.1,* p. 153.

[12] *PCI Local Bus Specification, Revision 2.1,* p. 221.

[13] *PCI Local Bus Specification, Revision 2.1,* p. 224, Table 7-3, Note 1.

Chapter 4
Software Requirements

This chapter presents requirements for two categories of software that support inserting and removing Adapter Cards while the system is running.

The first category is system software, which consists of the operating system, Adapter Drivers, and the Hot-Plug Service. With a few exceptions, it is not the purpose of this specification to divide system software functionality between these three sections. Rather this specification discusses requirements that the operating system must guarantee, either by itself or by assigning duties to the Hot-Plug Service or Adapter Driver. The operating-system vendor determines the interfaces between these three sections.

The second category is platform-specific software, which consists of the Hot-Plug System Driver and its programming interface, the Hot-Plug Primitives.

4.1 System Software

4.1.1 Items Specified by the Operating-System Vendor

The following is a summary of architectural issues or alternatives that must be specified by an operating-system vendor to enable the development of hot-plug Adapter Drivers and Hot-Plug System Drivers.

Adapter Driver Specification:

- How the Adapter Driver will be notified to quiesce adapter activity.
- Whether the operating system supports “pausing” an Adapter Driver in expectation that a similar Adapter Card will be reinstalled. If so, how the operation is initiated and how additional error conditions are handled.
- What responsibility the Adapter Driver has for replacing the functionality of Adapter Card option ROMs.

- How the Adapter Driver will be notified to start using a new Adapter Card or resume using an old one.

Hot-Plug System Driver Specification:

- Format and structure of the Hot-Plug Primitives and their parameters, including the Logical Slot Identifier parameter.
- Whether the Hot-Plug System Driver is responsible for executing the Platform Configuration Routine after a hot-insertion. If so, whether partial initialization of the Configuration Space header is acceptable, and the state of the slot after a failure during Configuration Space header initialization.

The operating-system vendor also specifies the Hot-Plug Service, and its user interface and/or programming interface for remote management. See Section 4.1.7 for more details.

4.1.2 Quiescing Adapter Activity

Before an Adapter Card can be removed from a Platform, the system must stop accessing the card, and the card must stop accessing the system. The sequence of steps that the system uses to make this guarantee is called quiescing adapter activity. The operating-system vendor must define the method of quiescing and restarting adapter activity. Although the details of this operation are strictly operating-system specific, the sequence must include the following elements, or their equivalent:

1. The system stops issuing new requests to the Adapter Driver, or notifies the Adapter Driver to stop accepting new requests.
2. The Adapter Driver completes or terminates all outstanding requests.
3. The Adapter Driver places the Adapter Card in a state in which it will not initiate any interrupts or bus master activity on the bus.

The operating system's implementation of quiescing adapter activity must not depend on the selected Adapter Card ever coming back. In addition to quiescing adapter activity, an operating-system vendor may optionally implement a less drastic "pause" capability, in anticipation of the same or a similar Adapter Card being reinserted. However, if the card is not reinserted, the old Adapter Driver eventually must be quiesced.

Implementation Note: Quiescing an Adapter Driver

When the Adapter Driver has been quiesced, it will issue no bus transactions to the Adapter Card, even if another device sharing the same interrupt input as this driver instance generates an interrupt to its driver. In some operating systems, the Adapter Driver may actually be unloaded when it is quiesced.

An Adapter Driver that controls multiple Adapter Cards must quiesce only the binding for the selected Adapter Card.

If an Adapter Driver is being quiesced because the Adapter Card has failed, it may not be possible to complete some outstanding requests normally. Adapter Drivers should detect conditions when the Adapter Card does not respond properly, attempt to reset the Adapter Card, and terminate any outstanding requests.

4.1.3 Initializing the Configuration Space Header

Each operating-system vendor must specify the policy for his environment for initializing Adapter Card Configuration Space headers. The policy includes the following:

1. Whether initialization of the Configuration Space headers is the responsibility of Platform-specific software or operating-system-specific software.
2. If such initialization is the responsibility of Platform-specific software, then:
   a) Whether the operating system requires complete initialization of all resources, or whether some resources are optional (such as doubly mapped I/O Space ranges).
   b) In what state a slot must be left if an error occurs during the initialization process.

The module responsible for initializing the Configuration Space header of a newly installed Adapter Card is defined to be the Platform Configuration Routine. It is responsible for assigning system resources, including I/O space, memory space, and PCI bus numbers, to all devices and functions on the new Adapter Card.

The operating-system vendor may specify that configuration is an operating system function. In that case, the operating system is responsible for executing the Platform Configuration Routine after a slot is turned on.

Alternatively, the operating-system vendor may specify that the Platform Configuration Routine must be handled by Platform-specific software such as the Hot-Plug System Driver. In that case, the operating system requires that the Hot-Plug System Driver execute the Platform Configuration Routine whenever it is given a request to turn a slot on.

In either case the operating system must guarantee that the Configuration Space header is appropriately initialized before the Adapter Driver is allowed to access the Adapter Card.

Implementation Note: Assigning Responsibility for the Platform Configuration Routine

The operating system determines which routine executes the Platform Configuration Routine based on how it tracks available configuration resources. For example, if the operating system depends upon the Platform BIOS to initialize the Configuration Space header at boot time, then the Hot-Plug System Driver (supplied by the Platform vendor) might handle configuration when an Adapter Card is hot-plugged. However, if the operating system places the responsibility for configuring the header at boot time in higher-level routines, then the Hot-Plug Service (supplied by the operating-system vendor) might handle configuration after a hot-plug event.

If the operating-system vendor requires the Platform Configuration Routine to be run by Platform-specific software, then the operating-system vendor must also specify whether all resources must be assigned, or whether the operating system can accept a partial configuration, such as assigning memory resources but not I/O resources to an Adapter Card that requests both. The operating-system vendor must also specify whether the card is to be left in the on or off state if configuration fails.

An Adapter Card that has no Configuration Space header should be treated as if all resource requests have been satisfied.

4.1.4 Adapter Card Option ROMs

Each operating-system vendor must specify the policy for his environment for the treatment of Adapter Card option ROM code after an Adapter Card is hot-inserted. Possible alternatives include, but are not limited to, the following:

1. Execution of an Open Firmware code image.
2. Require that option ROM functionality be provided by some other means, e.g., duplicating the function in the Adapter Driver.

Implementation Note: Replacing Functionality of Option ROMs on Adapter Cards

Adapter Cards may include option ROMs that establish operating parameters of the card. If these ROMs contain an image of Code Type 0 (Intel x86, PC-AT compatible), then the code stored in this image is designed to execute during initial power-on, before the operating system is loaded, and, in general, cannot be executed during a hot-insertion operation.

If the operating-system vendor specifies that option ROMs will not be executed after a hot-insertion, then the Adapter Card vendor must use the capabilities and functions available through the operating system at run-time to replace the functionality provided by these option ROMs at boot-time. For example, the adapter vendor may choose to implement this function by adding it to the Adapter Driver itself, or by creating a separate adapter-specific configuration application that communicates with the Adapter Driver.

4.1.5 Adapter Driver Requirements

An Adapter Driver that supports removing and installing an Adapter Card while the operating system is running has several requirements beyond those of a conventional Adapter Driver.

First, it must be possible to quiesce and start the driver while the system is running. The operating-system vendor is responsible for specifying how the Adapter Driver is to be quiesced, and how it will be notified to start using a new Adapter Card, or to resume using an old one.

Second, the Adapter Driver must guarantee that the Adapter Card has completed its internal initialization sequence before the driver uses the card. The Adapter Driver may become active much sooner after a hot-insertion event than it would after power is initially applied to the system.

Third, one alternative for the treatment of PCI options ROMs is for the operating-system vendor to require that the Adapter Driver replace any necessary functionality of option ROMs after a hot-insertion event (see Section 4.1.4).

4.1.6 Attention Indicator

The Platform is required to provide an Attention Indicator with each hot-plug slot (see Section 3.1.1). The Hot-Plug Service controls the state of this indicator by issuing Hot-Plug Primitives to the Hot-Plug System Driver.

This specification does not define the conditions under which the Hot-Plug Service must activate the Attention Indicator. The Hot-Plug Service may activate the Attention Indicator at any time to call the user's attention to a particular slot. For example, the Hot-Plug Service might activate the Attention Indicator when the system detects problem conditions that require user intervention at the Adapter Card. Or the Hot-Plug Service might allow the user to specify functions such as locating all the 66-MHz slots or locating a particular slot so an associated cable can be changed.

4.1.7 Hot-Plug Service

The Hot-Plug Service is a broad collection of software routines supplied by the operating-system vendor that monitor and control hot-plug operations, including the user interface and hot-plug sequence control. The Hot-Plug Service is responsible for issuing requests to the operating system to quiesce adapter activity and to the Hot-Plug System Driver to turn slots on and off. It determines when it is appropriate for the system to stop or resume using an Adapter Card or to start using a new one.

The Hot-Plug Service must provide an interface to the user by which the user can turn slots on and off. This interface is required to use the Physical Slot Identifiers provided by the Platform to designate slots. See Section 4.1.8 for more information.

The design and implementation of the Hot-Plug Service is unique to each particular operating system, and all other aspects of the Hot-Plug Service and its operation are controlled by the operating-system vendor.

4.1.8 Slot Identification

Several forms of slot identification are required throughout a hot-plug system. The first is the Physical Slot Identifier. The user interface in the Hot-Plug Service is required to identify slots by Physical Slot Identifiers provided by the Platform (see Section 4.1.7).

The second form of slot identification is PCI bus and device number. The Hot-Plug System Driver is required to run PCI configuration cycles to read the card's 66 MHz Capable bit (see Section 4.2.2.3) and, in some systems, to run the Platform Configuration Routine (see Section 4.1.3). PCI configuration cycles require the Adapter Card in the slot to be identified by its PCI bus and device number.

The third form of slot identification is the Logical Slot Identifier. The operating-system vendor must specify the encoding of a parameter used by the Hot-Plug Primitives for uniquely identifying each slot. This parameter is called a Logical Slot Identifier. See Section 4.2.2 for a discussion of how the Hot-Plug Primitives use the Logical Slot Identifier.

The operating-system vendor must specify translation mechanisms between these three forms of slot identification, using information supplied by the Platform vendor (see Section 3.1.1).

Implementation Note: Examples of Translating Between Forms of Slot Identification

The Platform vendor may supply information about the way the Platform slots are wired in the BIOS ROM, the Hot-Plug Controller, Platform-specific code in the Hot-Plug System Driver, or any other suitable place. The operating-system vendor, in cooperation with the Platform vendor, will define the Logical Slot Identifiers and the translation mechanisms to use this information.

For example, in a single-chassis X86-class system that uses slot number for the Physical Slot Identifier, the BIOS ROM includes the translation between slot number and bus and device number in the IRQ Routing Table. The operating-system vendor might define the Logical Slot Identifier to be the physical slot number. In that case, the Hot-Plug System Driver would perform the remaining translation between Logical Slot Identifier and PCI bus and device number by reading the IRQ Routing Table.

Alternatively, the operating-system vendor might define the Logical Slot Identifier to be the PCI bus and device number. In that case, no translation is required by the Hot-Plug System Driver, but the Hot-Plug Service would access the IRQ Routing Tables to translate to physical slot numbers for the user interface.

In all cases, the translation mechanisms must comprehend the effect that adding and removing PCI-to-PCI bridges will have on the PCI bus numbers of other devices in the system.

4.2 Platform-Specific Software

4.2.1 Hot-Plug System Driver

The Hot-Plug System Driver is the device driver for the Hot-Plug Controller, and is normally supplied by the Platform vendor. It is responsible for executing the Hot-Plug Primitive requests and providing results as described in Section 4.2.2.

A hot-plug system must include at least one Hot-Plug System Driver. Any given hot-plug slot is controlled by exactly one Hot-Plug System Driver.

4.2.2 Hot-Plug Primitives

The Hot-Plug Primitives presented in this section define what information is passed between the Hot-Plug Service and the Hot-Plug System Driver. Although this specification presents the Primitives in the form of requests and parameters, the actual programming interface is operating-system dependent and *not* controlled by this specification. Furthermore, the operating-system vendor is permitted to combine or split primitives into any number of specific operations.

Only requests, parameters, and error conditions specifically related to hot-plug operations are specified here. It is assumed that each operating system will include other procedural constraints and operating-system-specific error conditions, such as invalid-parameter errors and additional requests for diagnostic and system-management functions.

While there may be more than one Hot-Plug System Driver in a system, this section assumes the Hot-Plug Service addresses the particular Hot-Plug System Driver that controls the slot of interest. In an actual implementation, the operating system may instead choose a broadcast structure for the Primitives, and require each Hot-Plug System Driver to determine which Primitives to execute.

Unless otherwise specified, the parameters shown for each Hot-Plug Primitive assume that the minimum hot-plug Platform hardware shown in Section 3.1.1 is implemented. If the Platform implements additional features, then additional parameters may be needed for some or all Hot-Plug Primitives.

Implementation Note: Implementation Options for Hot-Plug Primitives

Although it is possible to implement the hot-plug primitives as single function calls or messages, this may not yield the most usable implementation. This technique yields an awkward interface where the Hot-Plug Service needs to pass down the current value of the parameter it doesn't intend to set when using the Setting Slot Status primitive. It also implies that the Hot-Plug System Driver needs to handle set-slot-on requests while on, and set-slot-off while off.

Similarly, if gathering up all the information in Querying Slot Status is undesirable in some way (because it is slow, causes power cycling of the card, etc.), then it would be appropriate to implement Querying Slot Status as multiple requests so that the Hot-Plug Service can ask for only the items of interest.

An operating system may wish to split Setting Slot Status into a Set Attention-Indicator function and a Set Slot State function to relieve the Hot-Plug Service from providing the correct current value of the parameter it doesn't wish to modify. Alternatively, an operating system may choose to have three values for each parameter: on (attention), off (normal), or no-change, and retain a single call or message per primitive. Similar considerations apply to the design of the calls or messages to implement the Querying Slot Status primitive, if the operating system designer believes there is some advantage in gathering only the parameters in which the Hot-Plug Service is interested.

In the following definitions of individual Hot-Plug Primitives, braces {} enclose a list of mutually exclusive alternative values for the parameters in question.

4.2.2.1 Querying the Hot-Plug System Driver

Parameters passed:

- none

Parameters returned:

- set of Logical Slot Identifiers for slots controlled by this Hot-Plug System Driver

The Hot-Plug System Driver must report the set of Logical Slot Identifiers for the slots it supports when requested by the Hot-Plug Service.

4.2.2.2 Setting Slot Status

Parameters passed:

- Logical Slot Identifier
- new slot state: {off, on}
- new Attention-Indicator state: {normal, attention}

Parameters returned:

- request completion status: {status change successful, fault—wrong frequency, fault—not enough power available, fault—insufficient configuration resources,[14] fault—power failure, fault—general failure}

This request controls the state of the hot-plug slot and the state of the Attention Indicator for the slot.

The slot's state is either off or on. In the off state the slot is powered down and isolated from the bus and the Adapter Card can safely be removed or installed. In the on state, the slot is powered and, if an Adapter Card is present, it is ready for the Configuration Space header to be configured (or has already been configured, see Section 4.1.3). Although the hardware may pass through multiple intermediate states between off and on, the Hot-Plug Service is only concerned about these two.

See Section 4.1.6 for a complete description of the Attention Indicator.

The Hot-Plug System Driver must enforce the requirement of T_{rhfa} specified in the *PCI Local Bus Specification* as modified by the Engineering Change Request entitled, **RST#** Timing. The Hot-Plug System Driver must wait at least T_{rhfa} after **RST#** has been deasserted to the slot before it accesses the card or completes a request to turn a slot on.

This request may complete in a number of ways. If the slot is successfully turned on or off, then the request will return the "successful" completion status. Unless otherwise specified, if a fault occurs while attempting to turn on a slot, then the Hot-Plug System Driver must leave the slot in the off state, and the completion status will indicate one of the following types of errors occurred. Some of these error conditions require optional Platform support (see Section 3.1.1). The Hot-Plug Service is required to support all of these error conditions. The Hot-Plug System Driver may optionally support a subset of the failure codes and report the rest as "fault—general failure." However, it is recommended that the Hot-Plug System Driver implement codes for each failure type that the Platform can detect.

- If the Adapter Card is a 33-MHz card, and the bus is already operating at 66 MHz, then the completion status will be "fault—wrong frequency."
- If the Hot-Plug System Driver can determine by reading the **PRSNT[1:2]#** pins on the Adapter Card that the Platform does not have enough power available to turn on the slot, the completion status will be "fault—not enough power available."

[14] Applies only if the Platform Configuration Routine is executed from the Hot-Plug System Driver.

- If the Hot-Plug System Driver is responsible for running the Platform Configuration Routine, but the configuration failed because there were insufficient resources available, then the completion status will be "fault—insufficient configuration resources." The operating-system vendor must specify whether the slot should be left on or off after this error. If the operating system can accept a partial configuration of an Adapter Card, the operating-system vendor must also specify how the Hot-Plug System Driver must indicate this condition. See Section 4.1.3 for more details.
- If a slot power fault is detected while turning on the slot, then the completion status will be "fault—power failure."
- If the Hot-Plug System Driver was not successful in turning on the slot (and configuring the card, if appropriate) for any other reason, then the completion status will be "fault—general failure."

4.2.2.3 Querying Slot Status

Parameters passed:

- Logical Slot Identifier

Parameters returned:

- Slot state {off, on}
- Adapter Card power requirement {not present, low, medium, high}
- Adapter Card frequency capability {33 MHz, 66 MHz, insufficient power}
- Slot frequency {33 MHz, 66 MHz}

This request returns the state of the hot-plug slot and any Adapter Card that might be present.

The Adapter Card power requirement parameter returns the information encoded by the Adapter Card on its **PRSNT[1:2]#** pins, independent of whether the slot is on or off. The exact encodings are specified in the *PCI Local Bus Specification*.[15]

The Adapter Card frequency capability indicates whether the card is capable of operating at 66 MHz. If it is not, then 33 MHz will be indicated. The Hot-Plug System Driver must access the **M66EN** pin and/or the 66 MHz Capable bit in the Adapter Card's Configuration Space header to determine whether the Adapter Card is capable of 66-MHz operation. This determination must be made independent of whether the bus is currently operating at 33 MHz or 66 MHz, and independent of whether the Hot-Plug Service has turned the slot on or off. If the Hot-Plug System Driver must read the 66 MHz Capable bit, but it cannot turn the slot on because the card power requirement exceeds the power budget, then the Hot-Plug System Driver must indicate that there is insufficient power available to determine the Adapter Card's frequency capability. Adapter Card frequency capability is meaningless if no Adapter Card is installed in the slot.

[15] *PCI Local Bus Specification, Revision 2.1,* p. 149.

The Slot Frequency parameter indicates at what frequency the bus for this slot is currently operating, regardless of whether there is an Adapter Card in the slot or not, and whether the slot is on or off.

4.2.2.4 Asynchronous Notification of Slot Status Change

Parameters passed:

- Logical Slot Identifier

When the Hot-Plug System Driver detects an unsolicited change in the status of a slot (e.g., a run-time power fault in a slot, or a new card installed in a previously empty slot), it will notify the Hot-Plug Service using this Hot-Plug Primitive.

Asynchronous notifications are not required for standard hot-removal and hot-insertion operations, because these operations must follow orderly procedures defined by the operating-system and Platform vendors. The primary use of asynchronous notifications is to keep the Hot-Plug Service informed of changes to the state of the system outside of such operations.

It is recommended that the Hot-Plug Service not use asynchronous events to automatically change slot state, because an end-user may insert an Adapter Card and immediately remove it, or repeatedly remove and insert it until it is properly seated.

Appendix A
Power Up and Down Reference Information

This appendix contains selected information relating to **RST#** *timing when a device is powered up or down, and is copied from the PCI Local Bus Specification, Revision 2.1, pages 134, 139, and 140. This information is duplicated here for reference purposes only, and in case of a discrepancy, the original specification takes precedence.*

4.2.3.2. Timing Parameters

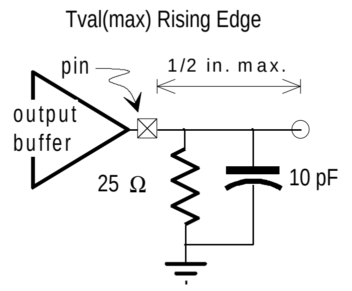
Table 4-6 provides the timing parameters for 5V and 3.3V signaling environments.

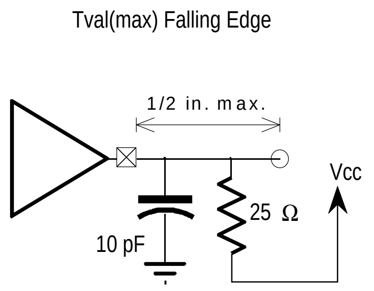
Table 4-6: 5V and 3.3V Timing Parameters

Symbol	Parameter	Min	Max	Units	Notes
t_{val}	**CLK** to Signal Valid Delay - bused signals	2	11	ns	1, 2, 3
t_{val}(ptp)	**CLK** to Signal Valid Delay - point to point	2	12	ns	1, 2, 3
t_{on}	Float to Active Delay	2		ns	1, 7
t_{off}	Active to Float Delay		28	ns	1, 7
t_{su}	Input Set up Time to **CLK** - bused signals	7		ns	3, 4
t_{su}(ptp)	Input Set up Time to **CLK** - point to point	10, 12		ns	3, 4
t_h	Input Hold Time from **CLK**	0		ns	4
t_{rst}	Reset Active Time After Power Stable	1		ms	5
$t_{rst\text{-}clk}$	Reset Active Time After **CLK** Stable	100		µs	5
$t_{rst\text{-}off}$	Reset Active to Output Float delay		40	ns	5, 6,7
t_{rrsu}	**REQ64#** to **RST#** setup time	10*T_{cyc}		ns	
T_{rrh}	**RST#** to **REQ64#** hold time	0	50	ns	

NOTES:

1. See the timing measurement conditions in Figure 4-8.
2. For parts compliant to the 5V signaling environment:
   Minimum times are evaluated with 0 pF equivalent load; maximum times are evaluated with 50 pF equivalent load. Actual test capacitance may vary, but results should be correlated to these specifications. Note that faster buffers may exhibit some ring back when attached to a 50 pF lump load, which should be of no consequence as long as the output buffers are in full compliance with slew rate and V/I curve specifications.
   For parts compliant to the 3.3V signaling environment:
   Minimum times are evaluated with same load used for slew rate measurement (as shown in Table 4-4, note 3); maximum times are evaluated with the following load circuits, for high-going and low-going edges respectively.





3. **REQ#** and **GNT#** are point-to-point signals, and have different output valid delay and input setup times than do bused signals. **GNT#** has a setup of 10; **REQ#** has a setup of 12. All other signals are bused.
4. See the timing measurement conditions in Figure 4-9.
5. **RST#** is asserted and deasserted asynchronously with respect to **CLK**. Refer to Section 4.3.2 for more information.
6. All output drivers must be asynchronously floated when **RST#** is active.
7. For purposes of Active/Float timing measurements, the Hi-Z or "off" state is defined to be when the total current delivered through the component pin is less than or equal to the leakage current specification.

4.3.2 Reset

The assertion and deassertion of the PCI reset signal (**RST#**) is asynchronous with respect to **CLK**. The rising (deassertion) edge of the **RST#** signal must be monotonic (bounce free) through the input switching range and must meet the minimum slew rate specified in Table 4-5. The PCI specification does not preclude the implementation of a synchronous **RST#**, if desired. The timing parameters for reset are contained in Table 4-6, with the exception of the T_{fail} parameter. This parameter provides for system reaction to one or both of the power rails going out of spec. If this occurs, parasitic diode paths could short circuit active output buffers. Therefore, **RST#** is asserted upon power failure in order to float the output buffers.

The value of T_{fail} is the minimum of:

- 500 ns (maximum) from either power rail going out of specification (exceeding specified tolerances by more than 500 mV)
- 100 ns (maximum) from the 5V rail falling below the 3.3V rail by more than 300 mV

The system must assert **RST#** during power up or in the event of a power failure. In order to minimize possible voltage contention between 5V and 3.3V parts, **RST#** should be asserted as soon as possible during the power up sequence. Figure 4-12 shows a worst case assertion of **RST#** asynchronously following the "power good" signal.[34] After **RST#** is asserted, PCI components must asynchronously disable (float) their outputs, but are not considered reset until both T_{rst} and $T_{rst\text{-}clk}$ parameters have been met. Figure 4-12 shows **RST#** signal timing.

[34] Component vendors should note that a fixed timing relationship between **RST#** and power sequencing cannot be guaranteed in all cases.

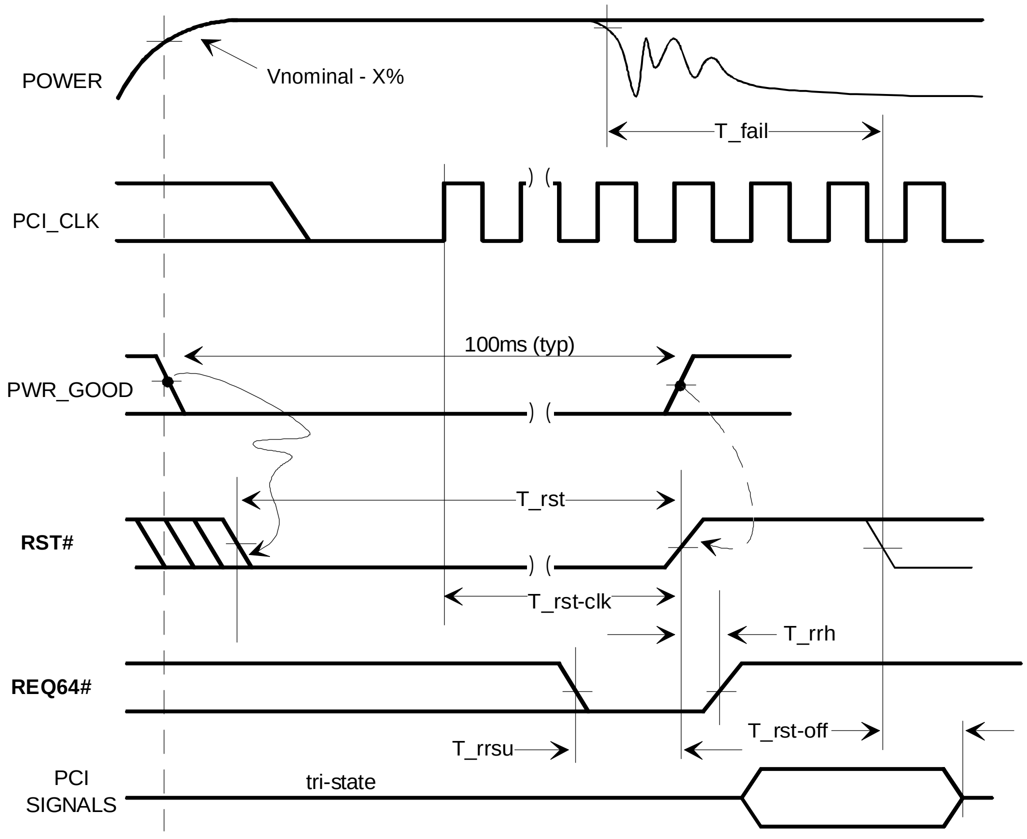


Figure 4-12: Reset Timing[35]

The **REQ64#** signal is used during reset to distinguish between parts that are connected to a 64-bit data path, and those that are not. The **REQ64#** signal is bused to all devices on the motherboard (including PCI connector slots) that support a 64-bit data path. This signal has a single pull-up resistor on the motherboard. On PCI expansion slots that do not support the 64-bit data path, the **REQ64#** signal is NOT bused or connected, but has its own, separate pull-up resistor. The central resource must drive **REQ64#** low (asserted) during the time that **RST#** is asserted, according to the timing specification. Devices that see **REQ64#** asserted during reset are connected to the 64-bit data path, while those that do not see the **REQ64#** assertion are not connected. This information may be used by the component to stabilize floating inputs during runtime, as described in Sections 4.2.1.1. and 4.2.2.1.

During reset, **REQ64#** has setup and hold time requirements with respect to the deasserting (high going) edge[36] of **RST#**. **REQ64#** is asynchronous with respect to the clock during reset.

[35] This reset timing figure optionally shows the "PWR_GOOD" signal as a pulse which is used to time the **RST#** pulse. In many systems "PWR_GOOD" may be a level, in which case the **RST#** pulse must be timed in another way.

[36] This allows **REQ64#** to be sampled on the deassertion edge of **RST#**.