# THE DATASHEET OF

# PCI1510ZGU

www.win-source.net 0086-755-83957316

TEXAS INSTRUMENTS

# PCI1510 GGU/GVF/PGE/ZGU/ZVF

## PC Card Controllers

# Data Manual

December 2004

Connectivity Solutions

SCPS071E

## IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

| **Products** | | **Applications** | |
|---|---|---|---|
| Amplifiers | amplifier.ti.com | Audio | www.ti.com/audio |
| Data Converters | dataconverter.ti.com | Automotive | www.ti.com/automotive |
| DSP | dsp.ti.com | Broadband | www.ti.com/broadband |
| Interface | interface.ti.com | Digital Control | www.ti.com/digitalcontrol |
| Logic | logic.ti.com | Military | www.ti.com/military |
| Power Mgmt | power.ti.com | Optical Networking | www.ti.com/opticalnetwork |
| Microcontrollers | microcontroller.ti.com | Security | www.ti.com/security |
| | | Telephony | www.ti.com/telephony |
| | | Video & Imaging | www.ti.com/video |
| | | Wireless | www.ti.com/wireless |

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

Copyright © 2004, Texas Instruments Incorporated

# Contents


| Section | Title | Page |
|---|---|---|
| **1** | **Introduction** | **1–1** |
| 1.1 | Description | 1–1 |
| 1.2 | Features | 1–1 |
| 1.3 | Related Documents | 1–2 |
| 1.4 | Trademarks | 1–2 |
| 1.5 | Document Conventions | 1–2 |
| 1.6 | Ordering Information | 1–3 |
| 1.7 | PCI1510 Data Manual Document History | 1–3 |
| **2** | **Terminal Descriptions** | **2–1** |
| 2.1 | PCI1510 Terminal Assignments | 2–3 |
| 2.2 | Terminal Descriptions | 2–15 |
| **3** | **Feature/Protocol Descriptions** | **3–1** |
| 3.1 | Power Supply Sequencing | 3–1 |
| 3.2 | I/O Characteristics | 3–2 |
| 3.3 | Clamping Voltages | 3–2 |
| 3.4 | Peripheral Component Interconnect (PCI) Interface | 3–2 |
| 3.4.1 | PCI $\overline{\text{GRST}}$ Signal | 3–2 |
| 3.4.2 | PCI Bus Lock ($\overline{\text{LOCK}}$) | 3–3 |
| 3.4.3 | Loading Subsystem Identification | 3–3 |
| 3.5 | PC Card Applications | 3–4 |
| 3.5.1 | PC Card Insertion/Removal and Recognition | 3–4 |
| 3.5.2 | Parallel Power-Switch Interface (TPS2211A) | 3–4 |
| 3.5.3 | Zoomed Video Support | 3–5 |
| 3.5.4 | Standardized Zoomed-Video Register Model | 3–6 |
| 3.5.4.1 | Zoomed-Video Card Insertion and Configuration Procedure | 3–6 |
| 3.5.5 | Internal Ring Oscillator | 3–7 |
| 3.5.6 | Integrated Pullup Resistors | 3–7 |
| 3.5.7 | SPKROUT and CAUDPWM Usage | 3–8 |
| 3.5.8 | LED Socket Activity Indicators | 3–8 |
| 3.5.9 | CardBus Socket Registers | 3–9 |
| 3.6 | Serial-Bus Interface | 3–9 |
| 3.6.1 | Serial-Bus Interface Implementation | 3–9 |
| 3.6.2 | Serial-Bus Interface Protocol | 3–10 |
| 3.6.3 | Serial-Bus EEPROM Application | 3–12 |
| 3.6.4 | Accessing Serial-Bus Devices Through Software | 3–13 |

| Section | | Title | Page |
|---|---|---|---|
| 3.7 | | Programmable Interrupt Subsystem | 3−13 |
| | 3.7.1 | PC Card Functional and Card Status Change Interrupts | 3−13 |
| | 3.7.2 | Interrupt Masks and Flags | 3−15 |
| | 3.7.3 | Using Parallel IRQ Interrupts | 3−15 |
| | 3.7.4 | Using Parallel PCI Interrupts | 3−16 |
| | 3.7.5 | Using Serialized IRQSER Interrupts | 3−16 |
| | 3.7.6 | SMI Support in the PCI1510 Controller | 3−16 |
| 3.8 | | Power Management Overview | 3−17 |
| | 3.8.1 | Integrated Low-Dropout Voltage Regulator (LDO-VR) | 3−17 |
| | 3.8.2 | Clock Run Protocol | 3−17 |
| | 3.8.3 | CardBus PC Card Power Management | 3−17 |
| | 3.8.4 | 16-Bit PC Card Power Management | 3−17 |
| | 3.8.5 | Suspend Mode | 3−18 |
| | 3.8.6 | Requirements for Suspend Mode | 3−18 |
| | 3.8.7 | Ring Indicate | 3−19 |
| | 3.8.8 | PCI Power Management | 3−19 |
| | 3.8.9 | CardBus Bridge Power Management | 3−20 |
| | 3.8.10 | ACPI Support | 3−21 |
| | 3.8.11 | Master List of PME Context Bits and Global Reset-Only Bits | 3−22 |
| **4** | | **PC Card Controller Programming Model** | **4−1** |
| 4.1 | | PCI Configuration Registers | 4−1 |
| 4.2 | | Vendor ID Register | 4−2 |
| 4.3 | | Device ID Register | 4−2 |
| 4.4 | | Command Register | 4−3 |
| 4.5 | | Status Register | 4−4 |
| 4.6 | | Revision ID Register | 4−4 |
| 4.7 | | PCI Class Code Register | 4−5 |
| 4.8 | | Cache Line Size Register | 4−5 |
| 4.9 | | Latency Timer Register | 4−5 |
| 4.10 | | Header Type Register | 4−5 |
| 4.11 | | BIST Register | 4−6 |
| 4.12 | | CardBus Socket/ExCA Base-Address Register | 4−6 |
| 4.13 | | Capability Pointer Register | 4−6 |
| 4.14 | | Secondary Status Register | 4−7 |
| 4.15 | | PCI Bus Number Register | 4−8 |
| 4.16 | | CardBus Bus Number Register | 4−8 |
| 4.17 | | Subordinate Bus Number Register | 4−8 |
| 4.18 | | CardBus Latency Timer Register | 4−8 |
| 4.19 | | Memory Base Registers 0, 1 | 4−9 |
| 4.20 | | Memory Limit Registers 0, 1 | 4−9 |
| 4.21 | | I/O Base Registers 0, 1 | 4−10 |

| Section | Title | Page |
|---|---|---|
| 4.22 | I/O Limit Registers 0, 1 | 4−10 |
| 4.23 | Interrupt Line Register | 4−11 |
| 4.24 | Interrupt Pin Register | 4−11 |
| 4.25 | Bridge Control Register | 4−12 |
| 4.26 | Subsystem Vendor ID Register | 4−13 |
| 4.27 | Subsystem ID Register | 4−13 |
| 4.28 | PC Card 16-Bit I/F Legacy-Mode Base Address Register | 4−13 |
| 4.29 | System Control Register | 4−14 |
| 4.30 | Multifunction Routing Register | 4−16 |
| 4.31 | Retry Status Register | 4−17 |
| 4.32 | Card Control Register | 4−18 |
| 4.33 | Device Control Register | 4−19 |
| 4.34 | Diagnostic Register | 4−20 |
| 4.35 | Capability ID Register | 4−20 |
| 4.36 | Next-Item Pointer Register | 4−20 |
| 4.37 | Power-Management Capabilities Register | 4−21 |
| 4.38 | Power-Management Control/Status Register | 4−22 |
| 4.39 | Power-Management Control/Status Register Bridge Support Extensions | 4−23 |
| 4.40 | Power-Management Data Register | 4−23 |
| 4.41 | General-Purpose Event Status Register | 4−24 |
| 4.42 | General-Purpose Event Enable Register | 4−25 |
| 4.43 | General-Purpose Input Register | 4−26 |
| 4.44 | General-Purpose Output Register | 4−26 |
| 4.45 | Serial-Bus Data Register | 4−27 |
| 4.46 | Serial-Bus Index Register | 4−27 |
| 4.47 | Serial-Bus Slave Address Register | 4−28 |
| 4.48 | Serial-Bus Control and Status Register | 4−29 |
| **5** | **ExCA Compatibility Registers** | **5−1** |
| 5.1 | ExCA Identification and Revision Register | 5−4 |
| 5.2 | ExCA Interface Status Register | 5−5 |
| 5.3 | ExCA Power Control Register | 5−6 |
| 5.4 | ExCA Interrupt and General Control Register | 5−8 |
| 5.5 | ExCA Card Status-Change Register | 5−9 |
| 5.6 | ExCA Card Status-Change Interrupt Configuration Register | 5−10 |
| 5.7 | ExCA Address Window Enable Register | 5−11 |
| 5.8 | ExCA I/O Window Control Register | 5−12 |
| 5.9 | ExCA I/O Windows 0 and 1 Start-Address Low-Byte Registers | 5−13 |
| 5.10 | ExCA I/O Windows 0 and 1 Start-Address High-Byte Registers | 5−13 |
| 5.11 | ExCA I/O Windows 0 and 1 End-Address Low-Byte Registers | 5−13 |
| 5.12 | ExCA I/O Windows 0 and 1 End-Address High-Byte Registers | 5−13 |
| 5.13 | ExCA Memory Windows 0−4 Start-Address Low-Byte Registers | 5−14 |

| Section | Title | Page |
|---|---|---|
| 5.14 | ExCA Memory Windows 0−4 Start-Address High-Byte Registers | 5−14 |
| 5.15 | ExCA Memory Windows 0−4 End-Address Low-Byte Registers | 5−15 |
| 5.16 | ExCA Memory Windows 0−4 End-Address High-Byte Registers | 5−15 |
| 5.17 | ExCA Memory Windows 0−4 Offset-Address Low-Byte Registers | 5−16 |
| 5.18 | ExCA Memory Windows 0−4 Offset-Address High-Byte Registers | 5−16 |
| 5.19 | ExCA Card Detect and General Control Register | 5−17 |
| 5.20 | ExCA Global Control Register | 5−18 |
| 5.21 | ExCA I/O Windows 0 and 1 Offset-Address Low-Byte Registers | 5−19 |
| 5.22 | ExCA I/O Windows 0 and 1 Offset-Address High-Byte Registers | 5−19 |
| 5.23 | ExCA Memory Windows 0−4 Page Registers | 5−19 |
| **6** | **CardBus Socket Registers** | **6−1** |
| 6.1 | Socket Event Register | 6−2 |
| 6.2 | Socket Mask Register | 6−3 |
| 6.3 | Socket Present-State Register | 6−4 |
| 6.4 | Socket Force Event Register | 6−5 |
| 6.5 | Socket Control Register | 6−7 |
| 6.6 | Socket Power-Management Register | 6−8 |
| **7** | **Electrical Characteristics** | **7−1** |
| 7.1 | Absolute Maximum Ratings Over Operating Temperature Ranges | 7−1 |
| 7.2 | Recommended Operating Conditions | 7−2 |
| 7.3 | Electrical Characteristics Over Recommended Operating Conditions | 7−3 |
| 7.4 | PCI Clock/Reset Timing Requirements Over Recommended Ranges of Supply Voltage and Operating Free-Air Temperature | 7−3 |
| 7.5 | PCI Timing Requirements Over Recommended Ranges of Supply Voltage and Operating Free-Air Temperature | 7−4 |
| **8** | **Mechanical Information** | **8−1** |

# List of Illustrations


| *Figure* | *Title* | *Page* |
|---|---|---|
| 2−1 | PCI1510 GGU-Package Terminal Diagram | 2−1 |
| 2−2 | PCI1510 GVF-Package Terminal Diagram | 2−2 |
| 2−3 | PCI1510 PGE-Package Terminal Diagram | 2−3 |
| 3−1 | PCI1510 Simplified Block Diagram | 3−1 |
| 3−2 | 3-State Bidirectional Buffer | 3−2 |
| 3−3 | TPS2211A Typical Application | 3−5 |
| 3−4 | Zoomed Video Implementation Using the PCI1510 Controller | 3−5 |
| 3−5 | Zoomed Video Switching Application | 3−6 |
| 3−6 | Sample Application of SPKROUT and CAUDPWM | 3−8 |
| 3−7 | Two Sample LED Circuits | 3−9 |
| 3−8 | Serial EEPROM Application | 3−10 |
| 3−9 | Serial-Bus Start/Stop Conditions and Bit Transfers | 3−10 |
| 3−10 | Serial-Bus Protocol Acknowledge | 3−11 |
| 3−11 | Serial-Bus Protocol − Byte Write | 3−11 |
| 3−12 | Serial-Bus Protocol − Byte Read | 3−11 |
| 3−13 | EEPROM Interface Doubleword Data Collection | 3−12 |
| 3−14 | IRQ Implementation | 3−16 |
| 3−15 | Signal Diagram of Suspend Function | 3−18 |
| 3−16 | $\overline{\text{RI_OUT}}$ Functional Diagram | 3−19 |
| 3−17 | Block Diagram of a Status/Enable Cell | 3−21 |
| 5−1 | ExCA Register Access Through I/O | 5−1 |
| 5−2 | ExCA Register Access Through Memory | 5−1 |
| 6−1 | Accessing CardBus Socket Registers Through PCI Memory | 6−1 |

# List of Tables


| *Table* | *Title* | *Page* |
|---|---|---|
| 2−1 | Signal Names Sorted by PGE Terminal Number | 2−4 |
| 2−2 | Signal Names Sorted by GGU Terminal Number | 2−6 |
| 2−3 | Signal Names Sorted by GVF Terminal Number | 2−8 |
| 2−4 | CardBus PC Card Signal Names Sorted Alphabetically to Device Terminals | 2−11 |
| 2−5 | 16-Bit PC Card Signal Names Sorted Alphabetically to Device Terminals | 2−13 |
| 2−6 | Power Supply Terminals | 2−15 |
| 2−7 | PC Card Power Switch Terminals | 2−15 |
| 2−8 | PCI System Terminals | 2−16 |
| 2−9 | PCI Address and Data Terminals | 2−17 |
| 2−10 | PCI Interface Control Terminals | 2−18 |
| 2−11 | Multifunction and Miscellaneous Terminals | 2−19 |
| 2−12 | 16-Bit PC Card Address and Data Terminals | 2−20 |
| 2−13 | 16-Bit PC Card Interface Control Terminals | 2−21 |
| 2−14 | CardBus PC Card Interface System Terminals | 2−22 |
| 2−15 | CardBus PC Card Address and Data Terminals | 2−23 |
| 2−16 | CardBus PC Card Interface Control Terminals | 2−24 |
| 3−1 | PC Card Card-Detect and Voltage-Sense Connections | 3−4 |
| 3−2 | Zoomed-Video Card Interrogation | 3−7 |
| 3−3 | Integrated Pullup Resistors | 3−7 |
| 3−4 | CardBus Socket Registers | 3−9 |
| 3−5 | Register- and Bit-Loading Map | 3−12 |
| 3−6 | PCI1510 Registers Used to Program Serial-Bus Devices | 3−13 |
| 3−7 | Interrupt Mask and Flag Registers | 3−14 |
| 3−8 | PC Card Interrupt Events and Description | 3−14 |
| 3−9 | SMI Control | 3−16 |
| 3−10 | Requirements for Internal/External 2.5-V Core Power Supply | 3−17 |
| 3−11 | Power-Management Registers | 3−20 |
| 4−1 | PCI Configuration Registers | 4−1 |
| 4−2 | Bit Field Access Tag Descriptions | 4−2 |
| 4−3 | Command Register Description | 4−3 |
| 4−4 | Status Register Description | 4−4 |
| 4−5 | Secondary Status Register Description | 4−7 |
| 4−6 | Bridge Control Register Description | 4−12 |
| 4−7 | System Control Register Description | 4−14 |
| 4−8 | Multifunction Routing Register Description | 4−16 |
| 4−9 | Retry Status Register Description | 4−17 |

| Table | Title | Page |
|---|---|---|
| 4−10 | Card Control Register Description | 4−18 |
| 4−11 | Device Control Register Description | 4−19 |
| 4−12 | Diagnostic Register Description | 4−20 |
| 4−13 | Power-Management Capabilities Register Description | 4−21 |
| 4−14 | Power-Management Control/Status Register Description | 4−22 |
| 4−15 | Power-Management Control/Status Register Bridge Support Extensions Description | 4−23 |
| 4−16 | General-Purpose Event Status Register Description | 4−24 |
| 4−17 | General-Purpose Event Enable Register Description | 4−25 |
| 4−18 | General-Purpose Input Register Description | 4−26 |
| 4−19 | General-Purpose Output Register Description | 4−26 |
| 4−20 | Serial-Bus Data Register Description | 4−27 |
| 4−21 | Serial-Bus Index Register Description | 4−27 |
| 4−22 | Serial-Bus Slave Address Register Description | 4−28 |
| 4−23 | Serial-Bus Control and Status Register Description | 4−29 |
| 5−1 | ExCA Registers and Offsets | 5−2 |
| 5−2 | ExCA Identification and Revision Register Description | 5−4 |
| 5−3 | ExCA Interface Status Register Description | 5−5 |
| 5−4 | ExCA Power Control Register Description—82365SL Support | 5−6 |
| 5−5 | ExCA Power Control Register Description—82365SL-DF Support | 5−7 |
| 5−6 | ExCA Interrupt and General Control Register Description | 5−8 |
| 5−7 | ExCA Card Status-Change Register Description | 5−9 |
| 5−8 | ExCA Card Status-Change Interrupt Configuration Register Description | 5−10 |
| 5−9 | ExCA Address Window Enable Register Description | 5−11 |
| 5−10 | ExCA I/O Window Control Register Description | 5−12 |
| 5−11 | ExCA Memory Windows 0−4 Start-Address High-Byte Registers Description | 5−14 |
| 5−12 | ExCA Memory Windows 0−4 End-Address High-Byte Registers Description | 5−15 |
| 5−13 | ExCA Memory Windows 0−4 Offset-Address High-Byte Registers Description | 5−16 |
| 5−14 | ExCA Card Detect and General Control Register Description | 5−17 |
| 5−15 | ExCA Global Control Register Description | 5−18 |
| 6−1 | CardBus Socket Registers | 6−1 |
| 6−2 | Socket Event Register Description | 6−2 |
| 6−3 | Socket Mask Register Description | 6−3 |
| 6−4 | Socket Present-State Register Description | 6−4 |
| 6−5 | Socket Force Event Register Description | 6−6 |
| 6−6 | Socket Control Register Description | 6−7 |
| 6−7 | Socket Power-Management Register Description | 6−8 |

# 1 Introduction

## 1.1 Description

The Texas Instruments PCI1510 device, a 144-terminal or a 209-terminal single-slot CardBus controller designed to meet the *PCI Bus Power Management Interface Specification for PCI to CardBus Bridges*, is an ultralow-power high-performance PCI-to-CardBus controller that supports a single PC card socket compliant with the *PC Card Standard* (rev. 7.2). The controller provides features that make it the best choice for bridging between PCI and PC Cards in both notebook and desktop computers. The *PC Card Standard* retains the 16-bit PC Card specification defined in the *PCI Local Bus Specification* and defines the 32-bit PC Card, CardBus, capable of full 32-bit data transfers at 33 MHz. The controller supports both 16-bit and CardBus PC Cards, powered at 5 V or 3.3 V, as required.

The controller is compliant with the *PCI Local Bus Specification*, and its PCI interface can act as either a PCI master device or a PCI slave device. The PCI bus mastering is initiated during CardBus PC Card bridging transactions. The controller is also compliant with *PCI Bus Power Management Interface Specification* (rev. 1.1).

All card signals are internally buffered to allow hot insertion and removal without external buffering. The controller is register-compatible with the Intel 82365SL-DF and 82365SL ExCA controllers. The controller internal data path logic allows the host to access 8-, 16-, and 32-bit cards using full 32-bit PCI cycles for maximum performance. Independent buffering and a pipeline architecture provide an unsurpassed performance level with sustained bursting. The controller can also be programmed to accept fast posted writes to improve system-bus utilization.

Multiple system-interrupt signaling options are provided, including parallel PCI, parallel ISA, serialized ISA, and serialized PCI. Furthermore, general-purpose inputs and outputs are provided for the board designer to implement sideband functions. Many other features designed into the PCI1510 controller, such as a socket activity light-emitting diode (LED) outputs, are discussed in detail throughout this document.

An advanced complementary metal-oxide semiconductor (CMOS) process achieves low system power consumption while operating at PCI clock rates up to 33 MHz. Several low-power modes enable the host power management system to further reduce power consumption.

## 1.2 Features

The controller supports the following features:

- A 144-terminal low-profile QFP (PGE), 144-terminal MicroStar BGA™ ball-grid array (GGU/ZGU) package, or a 209-terminal PBGA (GVF/ZVF) package
- 2.5-V core logic and 3.3-V I/O with universal PCI interfaces compatible with 3.3-V and 5-V PCI signaling environments
- Integrated low-dropout voltage regulator (LDO-VR) eliminates the need for an external 2.5-V power supply
- Mix-and-match 5-V/3.3-V 16-bit PC Cards and 3.3-V CardBus Cards
- A single PC Card or CardBus slot with hot insertion and removal
- Parallel interface to TI TPS2211A single-slot PC Card power switch
- Burst transfers to maximize data throughput with CardBus Cards
- Interrupt configurations: parallel PCI, serialized PCI, parallel ISA, and serialized ISA
- Serial EEPROM interface for loading subsystem ID, subsystem vendor ID, and other configuration registers
- Pipelined architecture for greater than 130-Mbps throughput from CardBus-to-PCI and from PCI-to-CardBus

- Up to five general-purpose I/Os
- Programmable output select for $\overline{\text{CLKRUN}}$
- Five PCI memory windows and two I/O windows available for the 16-bit interface
- Two I/O windows and two memory windows available to the CardBus socket
- Exchangeable-card-architecture- (ExCA-) compatible registers are mapped in memory and I/O space
- Intel™ 82365SL-DF and 82365SL register compatible
- Ring indicate, $\overline{\text{SUSPEND}}$, PCI $\overline{\text{CLKRUN}}$, and CardBus $\overline{\text{CCLKRUN}}$
- Socket activity LED terminal
- PCI bus lock ($\overline{\text{LOCK}}$)
- Internal ring oscillator

## 1.3 Related Documents

- *Advanced Configuration and Power Interface (ACPI) Specification (revision 1.1)*
- *PCI Bus Power Management Interface Specification (revision 1.1)*
- *PCI Bus Power Management Interface Specification for PCI to CardBus Bridges (revision 0.6)*
- *PCI to PCMCIA CardBus Bridge Register Description (Yenta) (revision 2.1)*
- *PCI Local Bus Specification (revision 2.2)*
- *PCI Mobile Design Guide (revision 1.0)*
- *PC Card Standard (revision 7.2)*
- *Serialized IRQ Support for PCI Systems (revision 6)*

## 1.4 Trademarks

Intel is a trademark of Intel Corporation.

MicroStar BGA is a trademark of Texas Instruments.

Other trademarks are the property of their respective owners.

## 1.5 Document Conventions

Throughout this data manual, several conventions are used to convey information. These conventions are listed below:

1. To identify a binary number or field, a lower case b follows the numbers. For example: 000b is a 3-bit binary field.
2. To identify a hexadecimal number or field, a lower case h follows the numbers. For example: 8AFh is a 12-bit hexadecimal field.
3. All other numbers that appear in this document that do not have either a b or h following the number are assumed to be decimal format.
4. If the signal or terminal name has a bar above the name (for example, $\overline{\text{GRST}}$), then this indicates the logical NOT function. When asserted, this signal is a logic low, 0, or 0b.
5. RSVD indicates that the referenced item is reserved.
6. In Sections 4 through 6, the configuration space for the controller is defined. For each register bit, the software access method is identified in an access column. The legend for this access column includes the following entries:

   r – read-only access

ru – read-only access with updates by the controller internal hardware

rw – read and write access

rcu – read access with the option to clear an asserted bit with a write-back of 1b including updates by the controller internal hardware.

## 1.6 Ordering Information

| ORDERING NUMBER | NAME | VOLTAGE | PACKAGE |
|---|---|---|---|
| PCI1510 | PC Card controller | 3.3 V, 5-V tolerant I/Os | 144-terminal LQFP<br>144-ball PBGA (GGU or ZGU)<br>209-ball PBGA (GVF or ZVF) |

## 1.7 PCI1510 Data Manual Document History

| DATE | PAGE NUMBER | REVISION |
|---|---|---|
| 01/2003 | 2−23 | Modified terminal number of CAD30 from 143 to 142 for PGE package |
| 01/2003 | 3−2 | Added new subsection 3.4.1 to describe $\overline{\text{GRST}}$ during power up |
| 01/2003 | 3−11 | Modified byte-read diagram (Figure 3−12) to better reflect a read transaction to the EEPROM |
| 01/2003 | 3−20 | Modified the description of the power management capabilities register. This register is not a static read-only register. |
| 08/2003 | 1−3 | Added lead-free (Pb, atomic number 82) MicroStar BGA™ package (ZGU) to ordering information |
| 08/2003 | 2−1 | Added description for ZGU package |
| 08/2003 | 8−4 | Added ZGU mechanical drawing |
| 10/2003 | 1−1 | Added GVF package to features |
| 10/2003 | 1−3 | Added GVF package to ordering information |
| 10/2003 | 2−8 | Added GVF terminal descriptions, Table 2−3 |
| 10/2003 | 8−2 | Added GVF mechanical drawing. |
| 07/2004 | Chapters 1, 2, 8 | Added RGVF, RZVF, and ZVF packages and pinout. |
| 12/2004 | Chapters 1, 2, 8 | Removed RGVF and RZVF packages and pinout.<br>Added Section 1.5, Document Conventions |

# 2 Terminal Descriptions

The PCI1510 controller is available in five packages, a 144-terminal quad flatpack (PGE), two 144-terminal MicroStar BGA™ packages (GGU/ZGU), and two 209-terminal PBGA packages (GVF/ZVF). The GGU and ZGU packages are mechanically and electrically identical, but the ZGU is a lead-free (Pb, atomic number 82) design. Throughout the remainder of this manual, only the GGU package designator is used for either the GGU or ZGU package. The terminal layout for the GGU package is shown in Figure 2−1. The GVF and ZVF packages are mechanically and electrically identical, but the ZVF is a lead-free (Pb, atomic number 82) design. Throughout the remainder of this manual, only the GVF package designator is used for either the GVF or ZVF package. The terminal layout for the GVF package is shown in Figure 2−2. The terminal layout with signal names for the PGE package is shown in Figure 2−3.

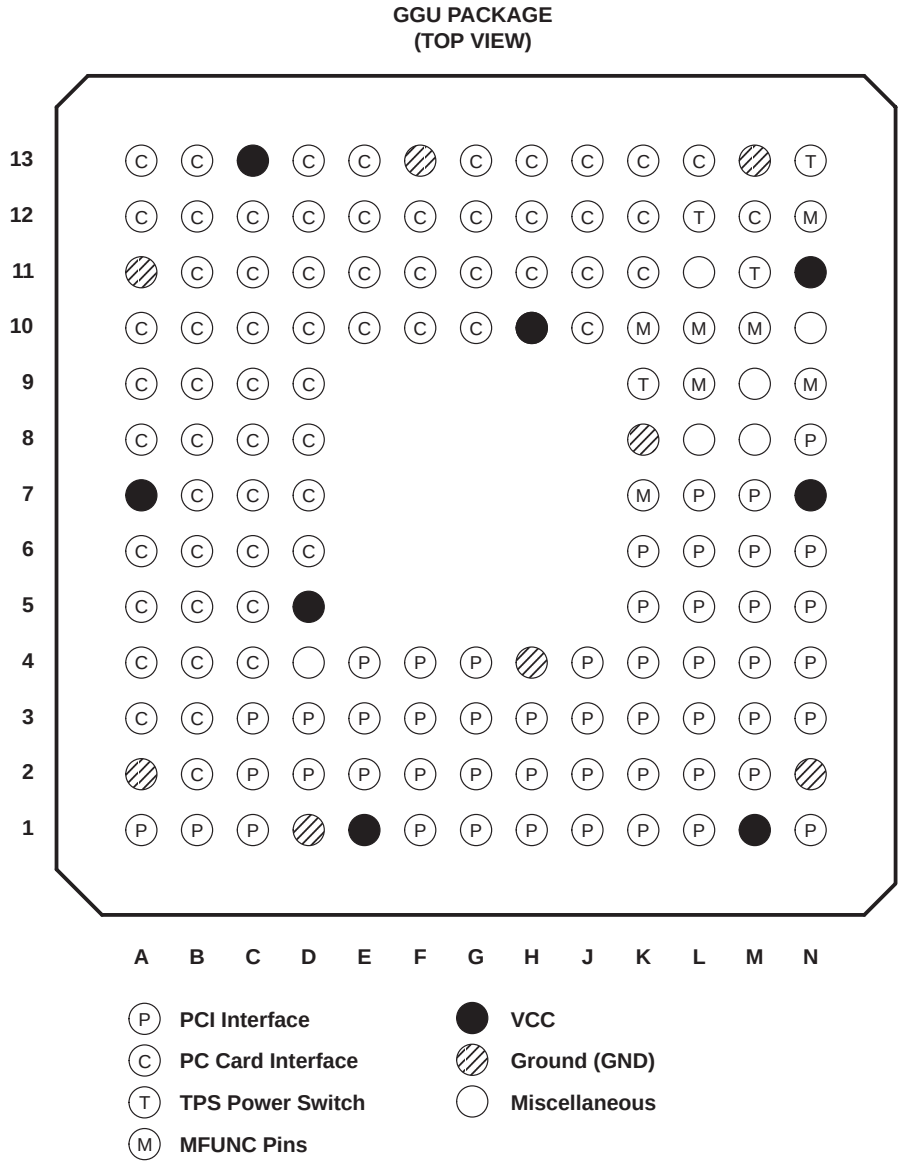
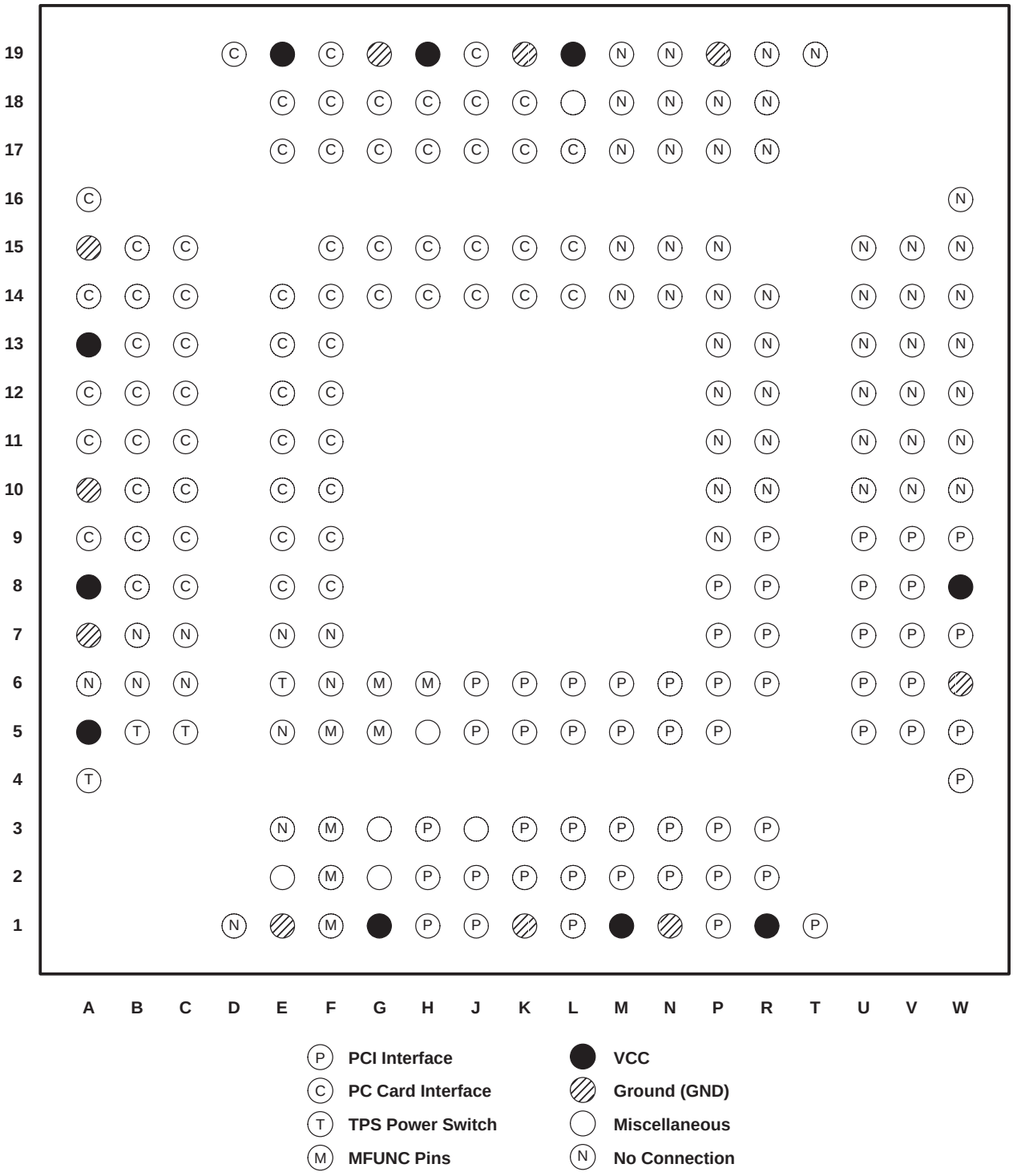



**Figure 2−1. PCI1510 GGU-Package Terminal Diagram**

**GVF PACKAGE**
**(TOP VIEW)**


19 C C C C C C N N N N N
18 C C C C C C C N N N N
17 C C C C C C C C N N N N
16 C N
15 C C C C C C C C C N N N N N N N
14 C C C C C C C C C C C N N N N N N N N
13 C C C C C N N N N N N N
12 C C C C C N N N N N N N
11 C C C C C N N N N N N N
10 C C C C C N N N N N N N
9 C C C C C N P P P P
8 C C C C C P P P P
7 N N N N N P P P P P
6 N N N T N M M P P P P P

## 2.1 PCI1510 Terminal Assignments

Figure 2−3 and Table 2−1 show the terminal assignments for the PGE package. Table 2−2 and Table 2−3 list the terminal assignments for the GGU and GVF packages, respectively. The signal names for the PC Card slot are given in a CardBus // 16-bit signal format. All tables are arranged in order by increasing terminal designator, which is numeric for the PGE package and alphanumeric for the other packages. Table 2−4 and Table 2−5 list the CardBus and 16-bit signal names, respectively, in alphanumerical order with the corresponding terminal numbers for each package.

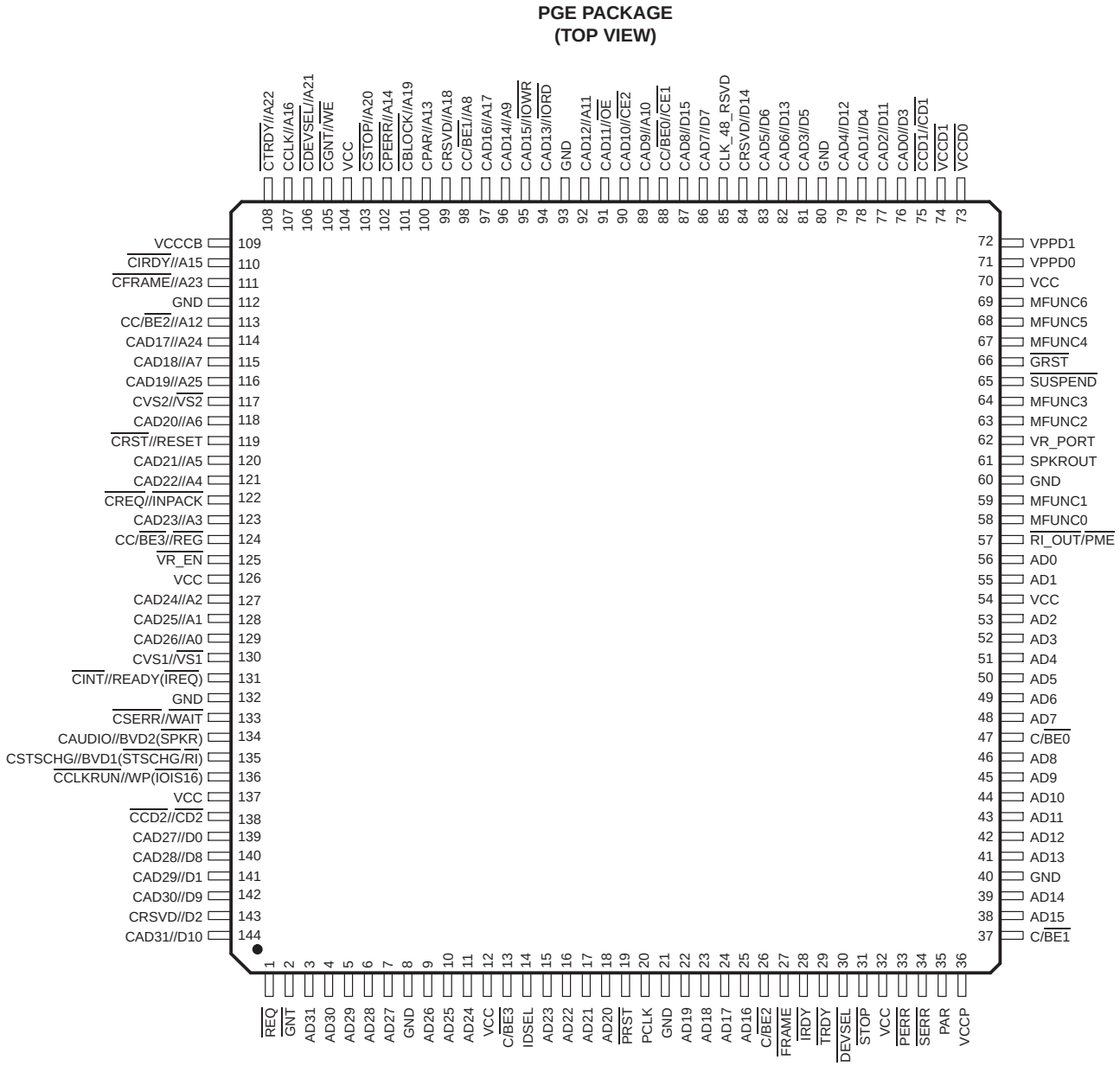



**Figure 2−3. PCI1510 PGE-Package Terminal Diagram**

## Table 2−1. Signal Names Sorted by PGE Terminal Number

| TERMINAL | SIGNAL NAME | | TERMINAL | SIGNAL NAME | |
|---|---|---|---|---|---|
| | CARDBUS | 16-BIT | | CARDBUS | 16-BIT |
| 1 | $\overline{\text{REQ}}$ | $\overline{\text{REQ}}$ | 43 | AD11 | AD11 |
| 2 | $\overline{\text{GNT}}$ | $\overline{\text{GNT}}$ | 44 | AD10 | AD10 |
| 3 | AD31 | AD31 | 45 | AD9 | AD9 |
| 4 | AD30 | AD30 | 46 | AD8 | AD8 |
| 5 | AD29 | AD29 | 47 | C/$\overline{\text{BE0}}$ | C/$\overline{\text{BE0}}$ |
| 6 | AD28 | AD28 | 48 | AD7 | AD7 |
| 7 | AD27 | AD27 | 49 | AD6 | AD6 |
| 8 | GND | GND | 50 | AD5 | AD5 |
| 9 | AD26 | AD26 | 51 | AD4 | AD4 |
| 10 | AD25 | AD25 | 52 | AD3 | AD3 |
| 11 | AD24 | AD24 | 53 | AD2 | AD2 |
| 12 | $V_{CC}$ | $V_{CC}$ | 54 | $V_{CC}$ | $V_{CC}$ |
| 13 | C/$\overline{\text{BE3}}$ | C/$\overline{\text{BE3}}$ | 55 | AD1 | AD1 |
| 14 | IDSEL | IDSEL | 56 | AD0 | AD0 |
| 15 | AD23 | AD23 | 57 | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ |
| 16 | AD22 | AD22 | 58 | MFUNC0 | MFUNC0 |
| 17 | AD21 | AD21 | 59 | MFUNC1 | MFUNC1 |
| 18 | AD20 | AD20 | 60 | GND | GND |
| 19 | $\overline{\text{PRST}}$ | $\overline{\text{PRST}}$ | 61 | SPKROUT | SPKROUT |
| 20 | PCLK | PCLK | 62 | VR_PORT | VR_PORT |
| 21 | GND | GND | 63 | MFUNC2 | MFUNC2 |
| 22 | AD19 | AD19 | 64 | MFUNC3 | MFUNC3 |
| 23 | AD18 | AD18 | 65 | $\overline{\text{SUSPEND}}$ | $\overline{\text{SUSPEND}}$ |
| 24 | AD17 | AD17 | 66 | $\overline{\text{GRST}}$ | $\overline{\text{GRST}}$ |
| 25 | AD16 | AD16 | 67 | MFUNC4 | MFUNC4 |
| 26 | C/$\overline{\text{BE2}}$ | C/$\overline{\text{BE2}}$ | 68 | MFUNC5 | MFUNC5 |
| 27 | $\overline{\text{FRAME}}$ | $\overline{\text{FRAME}}$ | 69 | MFUNC6 | MFUNC6 |
| 28 | $\overline{\text{IRDY}}$ | $\overline{\text{IRDY}}$ | 70 | $V_{CC}$ | $V_{CC}$ |
| 29 | $\overline{\text{TRDY}}$ | $\overline{\text{TRDY}}$ | 71 | VPPD0 | VPPD0 |
| 30 | $\overline{\text{DEVSEL}}$ | $\overline{\text{DEVSEL}}$ | 72 | VPPD1 | VPPD1 |
| 31 | $\overline{\text{STOP}}$ | $\overline{\text{STOP}}$ | 73 | $\overline{\text{VCCD0}}$ | $\overline{\text{VCCD0}}$ |
| 32 | $V_{CC}$ | $V_{CC}$ | 74 | $\overline{\text{VCCD1}}$ | $\overline{\text{VCCD1}}$ |
| 33 | $\overline{\text{PERR}}$ | $\overline{\text{PERR}}$ | 75 | $\overline{\text{CCD1}}$ | $\overline{\text{CD1}}$ |
| 34 | $\overline{\text{SERR}}$ | $\overline{\text{SERR}}$ | 76 | CAD0 | D3 |
| 35 | PAR | PAR | 77 | CAD2 | D11 |
| 36 | $V_{CCP}$ | $V_{CCP}$ | 78 | CAD1 | D4 |
| 37 | $\overline{\text{C/BE1}}$ | $\overline{\text{C/BE1}}$ | 79 | CAD4 | D12 |
| 38 | AD15 | AD15 | 80 | GND | GND |
| 39 | AD14 | AD14 | 81 | CAD3 | D5 |
| 40 | GND | GND | 82 | CAD6 | D13 |
| 41 | AD13 | AD13 | 83 | CAD5 | D6 |
| 42 | AD12 | AD12 | 84 | CRSVD | D14 |

## Table 2−1. Signal Names Sorted by PGE Terminal Number (Continued)

| TERMINAL | SIGNAL NAME | | TERMINAL | SIGNAL NAME | |
|---|---|---|---|---|---|
| | CARDBUS | 16-BIT | | CARDBUS | 16-BIT |
| 85† | CLK_48_RSVD | CLK_48_RSVD | 115 | CAD18 | A7 |
| 86 | CAD7 | D7 | 116 | CAD19 | A25 |
| 87 | CAD8 | D15 | 117 | CVS2 | $\overline{\text{VS2}}$ |
| 88 | CC/$\overline{\text{BE0}}$ | $\overline{\text{CE1}}$ | 118 | CAD20 | A6 |
| 89 | CAD9 | A10 | 119 | $\overline{\text{CRST}}$ | RESET |
| 90 | CAD10 | $\overline{\text{CE2}}$ | 120 | CAD21 | A5 |
| 91 | CAD11 | $\overline{\text{OE}}$ | 121 | CAD22 | A4 |
| 92 | CAD12 | A11 | 122 | $\overline{\text{CREQ}}$ | $\overline{\text{INPACK}}$ |
| 93 | GND | GND | 123 | CAD23 | A3 |
| 94 | CAD13 | $\overline{\text{IORD}}$ | 124 | CC/$\overline{\text{BE3}}$ | $\overline{\text{REG}}$ |
| 95 | CAD15 | $\overline{\text{IOWR}}$ | 125 | $\overline{\text{VR_EN}}$ | $\overline{\text{VR_EN}}$ |
| 96 | CAD14 | A9 | 126 | $V_{CC}$ | $V_{CC}$ |
| 97 | CAD16 | A17 | 127 | CAD24 | A2 |
| 98 | CC/$\overline{\text{BE1}}$ | A8 | 128 | CAD25 | A1 |
| 99 | CRSVD | A18 | 129 | CAD26 | A0 |
| 100 | CPAR | A13 | 130 | CVS1 | $\overline{\text{VS1}}$ |
| 101 | $\overline{\text{CBLOCK}}$ | A19 | 131 | $\overline{\text{CINT}}$ | READY($\overline{\text{IREQ}}$) |
| 102 | $\overline{\text{CPERR}}$ | A14 | 132 | GND | GND |
| 103 | $\overline{\text{CSTOP}}$ | A20 | 133 | $\overline{\text{CSERR}}$ | $\overline{\text{WAIT}}$ |
| 104 | $V_{CC}$ | $V_{CC}$ | 134 | CAUDIO | BVD2($\overline{\text{SPKR}}$) |
| 105 | $\overline{\text{CGNT}}$ | $\overline{\text{WE}}$ | 135 | CSTSCHG | BVD1($\overline{\text{STSCHG}}$/$\overline{\text{RI}}$) |
| 106 | $\overline{\text{CDEVSEL}}$ | A21 | 136 | $\overline{\text{CCLKRUN}}$ | WP($\overline{\text{IOIS16}}$) |
| 107 | CCLK | A16 | 137 | $V_{CC}$ | $V_{CC}$ |
| 108 | $\overline{\text{CTRDY}}$ | A22 | 138 | $\overline{\text{CCD2}}$ | $\overline{\text{CD2}}$ |
| 109 | $V_{CCCB}$ | $V_{CCCB}$ | 139 | CAD27 | D0 |
| 110 | $\overline{\text{CIRDY}}$ | A15 | 140 | CAD28 | D8 |
| 111 | $\overline{\text{CFRAME}}$ | A23 | 141 | CAD29 | D1 |
| 112 | GND | GND | 142 | CAD30 | D9 |
| 113 | CC/$\overline{\text{BE2}}$ | A12 | 143 | CRSVD | D2 |
| 114 | CAD17 | A24 | 144 | CAD31 | D10 |

† Terminal 85 is an NC on the PCI1510 to allow for terminal compatibility with the next generation of devices.

## Table 2−2. Signal Names Sorted by GGU Terminal Number

| TERMINAL | SIGNAL NAME | | TERMINAL | SIGNAL NAME | |
|---|---|---|---|---|---|
| | CARDBUS | 16-BIT | | CARDBUS | 16-BIT |
| A01 | C/$\overline{BE3}$ | C/$\overline{BE3}$ | D05 | $V_{CC}$ | $V_{CC}$ |
| A02 | GND | GND | D06 | CAUDIO | BVD2($\overline{SPKR}$) |
| A03 | CRSVD | D2 | D07 | CAD25 | A1 |
| A04 | CAD27 | D0 | D08 | $\overline{CRST}$ | RESET |
| A05 | $\overline{CCLKRUN}$ | WP($\overline{IOIS16}$) | D09 | CC/$\overline{BE2}$ | A12 |
| A06 | $\overline{CINT}$ | READY($\overline{IREQ}$) | D10 | CAD23 | A3 |
| A07 | $V_{CC}$ | $V_{CC}$ | D11 | $\overline{CDEVSEL}$ | A21 |
| A08 | CC/$\overline{BE3}$ | $\overline{REG}$ | D12 | $\overline{CPERR}$ | A14 |
| A09 | CVS2 | $\overline{VS2}$ | D13 | $\overline{CGNT}$ | $\overline{WE}$ |
| A10 | $\overline{CFRAME}$ | A23 | E01 | $V_{CC}$ | $V_{CC}$ |
| A11 | GND | GND | E02 | AD25 | AD25 |
| A12 | CAD18 | A7 | E03 | AD31 | AD31 |
| A13 | $\overline{CBLOCK}$ | A19 | E04 | AD29 | AD29 |
| B01 | AD27 | AD27 | E10 | $\overline{CSTOP}$ | A20 |
| B02 | CVS1 | $\overline{VS1}$ | E11 | CC/$\overline{BE1}$ | A8 |
| B03 | CAD31 | D10 | E12 | CPAR | A13 |
| B04 | CAD30 | D9 | E13 | CRSVD | A18 |
| B05 | $\overline{CCD2}$ | $\overline{CD2}$ | F01 | AD22 | AD22 |
| B06 | $\overline{CSERR}$ | $\overline{WAIT}$ | F02 | IDSEL | IDSEL |
| B07 | CAD24 | A2 | F03 | AD24 | AD24 |
| B08 | $\overline{CREQ}$ | $\overline{INPACK}$ | F04 | AD26 | AD26 |
| B09 | CAD19 | A25 | F10 | CAD16 | A17 |
| B10 | CAD17 | A24 | F11 | CAD14 | A9 |
| B11 | $V_{CCCB}$ | $V_{CCCB}$ | F12 | CAD13 | $\overline{IORD}$ |
| B12 | CAD22 | A4 | F13 | GND | GND |
| B13 | CCLK | A16 | G01 | PCLK | PCLK |
| C01 | $\overline{GNT}$ | $\overline{GNT}$ | G02 | AD20 | AD20 |
| C02 | $\overline{REQ}$ | $\overline{REQ}$ | G03 | $\overline{PRST}$ | $\overline{PRST}$ |
| C03 | AD23 | AD23 | G04 | AD21 | AD21 |
| C04 | CAD29 | D1 | G10 | CAD11 | $\overline{OE}$ |
| C05 | CAD28 | D8 | G11 | CAD9 | A10 |
| C06 | CSTSCHG | BVD1($\overline{STSCHG}$/$\overline{RI}$) | G12 | CAD12 | A11 |
| C07 | CAD26 | A0 | G13 | CAD10 | $\overline{CE2}$ |
| C08 | CAD21 | A5 | H01 | AD17 | AD17 |
| C09 | CAD20 | A6 | H02 | AD19 | AD19 |
| C10 | $\overline{CIRDY}$ | A15 | H03 | AD18 | AD18 |
| C11 | CAD15 | $\overline{IOWR}$ | H04 | GND | GND |
| C12 | $\overline{CTRDY}$ | A22 | H10† | CLK_48_RSVD | CLK_48_RSVD |
| C13 | $V_{CC}$ | $V_{CC}$ | H11 | CAD8 | D15 |
| D01 | GND | GND | H12 | CAD7 | D7 |
| D02 | AD28 | AD28 | H13 | CC/$\overline{BE0}$ | $\overline{CE1}$ |
| D03 | AD30 | AD30 | J01 | $\overline{FRAME}$ | $\overline{FRAME}$ |
| D04 | $\overline{VR_EN}$ | $\overline{VR_EN}$ | J02 | C/$\overline{BE2}$ | C/$\overline{BE2}$ |

† Terminal H10 is not bonded out in the packaged parts in order to have pin compatibility with future devices.

## Table 2−2. Signal Names Sorted by GGU Terminal Number (Continued)

| TERMINAL | SIGNAL NAME | | TERMINAL | SIGNAL NAME | |
|---|---|---|---|---|---|
| | CARDBUS | 16-BIT | | CARDBUS | 16-BIT |
| J03 | $\overline{\text{TRDY}}$ | $\overline{\text{TRDY}}$ | L11 | $\overline{\text{GRST}}$ | $\overline{\text{GRST}}$ |
| J04 | AD16 | AD16 | L12 | $\overline{\text{VCCD1}}$ | $\overline{\text{VCCD1}}$ |
| J10 | CAD5 | D6 | L13 | $\overline{\text{CCD1}}$ | $\overline{\text{CD1}}$ |
| J11 | CAD4 | D12 | M01 | $V_{CC}$ | $V_{CC}$ |
| J12 | CRSVD | D14 | M02 | AD9 | AD9 |
| J13 | CAD3 | D5 | M03 | C/$\overline{\text{BE1}}$ | C/$\overline{\text{BE1}}$ |
| K01 | $\overline{\text{IRDY}}$ | $\overline{\text{IRDY}}$ | M04 | AD15 | AD15 |
| K02 | $\overline{\text{DEVSEL}}$ | $\overline{\text{DEVSEL}}$ | M05 | AD10 | AD10 |
| K03 | $\overline{\text{PERR}}$ | $\overline{\text{PERR}}$ | M06 | AD5 | AD5 |
| K04 | AD4 | AD4 | M07 | AD1 | AD1 |
| K05 | AD13 | AD13 | M08 | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ |
| K06 | C/$\overline{\text{BE0}}$ | C/$\overline{\text{BE0}}$ | M09 | SPKROUT | SPKROUT |
| K07 | MFUNC0 | MFUNC0 | M10 | MFUNC4 | MFUNC4 |
| K08 | GND | GND | M11 | VPPD1 | VPPD1 |
| K09 | VPPD0 | VPPD0 | M12 | CAD2 | D11 |
| K10 | MFUNC3 | MFUNC3 | M13 | GND | GND |
| K11 | CAD0 | D3 | N01 | PAR | PAR |
| K12 | CAD1 | D4 | N02 | GND | GND |
| K13 | CAD6 | D13 | N03 | AD12 | AD12 |
| L01 | $\overline{\text{STOP}}$ | $\overline{\text{STOP}}$ | N04 | AD8 | AD8 |
| L02 | $\overline{\text{SERR}}$ | $\overline{\text{SERR}}$ | N05 | AD7 | AD7 |
| L03 | $V_{CCP}$ | $V_{CCP}$ | N06 | AD3 | AD3 |
| L04 | AD11 | AD11 | N07 | $V_{CC}$ | $V_{CC}$ |
| L05 | AD14 | AD14 | N08 | AD0 | AD0 |
| L06 | AD6 | AD6 | N09 | MFUNC1 | MFUNC1 |
| L07 | AD2 | AD2 | N10 | $\overline{\text{SUSPEND}}$ | $\overline{\text{SUSPEND}}$ |
| L08 | VR_PORT | VR_PORT | N11 | $V_{CC}$ | $V_{CC}$ |
| L09 | MFUNC2 | MFUNC2 | N12 | MFUNC5 | MFUNC5 |
| L10 | MFUNC6 | MFUNC6 | N13 | $\overline{\text{VCCD0}}$ | $\overline{\text{VCCD0}}$ |

## Table 2−3. Signal Names Sorted by GVF Terminal Number

| TERMINAL | SIGNAL NAME | | TERMINAL | SIGNAL NAME | |
|---|---|---|---|---|---|
| | CARDBUS | 16-BIT | | CARDBUS | 16-BIT |
| A04 | VPPD0 | VPPD0 | E07 | NC | NC |
| A05 | $V_{CC}$ | $V_{CC}$ | E08 | CAD31 | D10 |
| A06 | NC | NC | E09 | CAD28 | D8 |
| A07 | GND | GND | E10 | $\overline{\text{CSERR}}$ | $\overline{\text{WAIT}}$ |
| A08 | $V_{CC}$ | $V_{CC}$ | E11 | CAD25 | A1 |
| A09 | CSTSCHG | BVD1($\overline{\text{STSCHG}}$/$\overline{\text{RI}}$) | E12 | CAD21 | A5 |
| A10 | GND | GND | E13 | CAD18 | A7 |
| A11 | $V_{CCCB}$ | $V_{CCCB}$ | E14 | $\overline{\text{CTRDY}}$ | A22 |
| A12 | CAD23 | A3 | E17 | $\overline{\text{CSTOP}}$ | A20 |
| A13 | $V_{CC}$ | $V_{CC}$ | E18 | $\overline{\text{CBLOCK}}$ | A19 |
| A14 | CAD19 | A25 | E19 | $V_{CC}$ | $V_{CC}$ |
| A15 | GND | GND | F01 | MFUNC5 | MFUNC5 |
| A16 | $\overline{\text{CDEVSEL}}$ | A21 | F02 | MFUNC3 | MFUNC3 |
| B05 | $\overline{\text{VCCD1}}$ | $\overline{\text{VCCD1}}$ | F03 | MFUNC2 | MFUNC2 |
| B06 | NC | NC | F05 | MFUNC0 | MFUNC0 |
| B07 | NC | NC | F06 | NC | NC |
| B08 | CAD29 | D1 | F07 | NC | NC |
| B09 | $\overline{\text{CCLKRUN}}$ | WP($\overline{\text{IOIS16}}$) | F08 | CRSVD | D2 |
| B10 | CVS1 | $\overline{\text{VS1}}$ | F09 | CAD27 | D0 |
| B11 | CC/$\overline{\text{BE3}}$ | $\overline{\text{REG}}$ | F10 | CAUDIO | BVD2($\overline{\text{SPKR}}$) |
| B12 | $\overline{\text{CREQ}}$ | $\overline{\text{INPACK}}$ | F11 | CAD26 | A0 |
| B13 | $\overline{\text{CRST}}$ | RESET | F12 | CVS2 | $\overline{\text{VS2}}$ |
| B14 | CAD17 | A24 | F13 | $\overline{\text{CIRDY}}$ | A15 |
| B15 | $\overline{\text{CFRAME}}$ | A23 | F14 | CPAR | A13 |
| C05 | VPPD1 | VPPD1 | F15 | $\overline{\text{CPERR}}$ | A14 |
| C06 | NC | NC | F17 | CRSVD | A18 |
| C07 | NC | NC | F18 | CAD16 | A17 |
| C08 | CAD30 | D9 | F19 | CAD14 | A9 |
| C09 | $\overline{\text{CCD2}}$ | $\overline{\text{CD2}}$ | G01 | $V_{CC}$ | $V_{CC}$ |
| C10 | $\overline{\text{CINT}}$ | READY($\overline{\text{IREQ}}$) | G02 | VR_PORT | VR_PORT |
| C11 | CAD24 | A2 | G03 | $\overline{\text{SUSPEND}}$ | $\overline{\text{SUSPEND}}$ |
| C12 | CAD22 | A4 | G05 | MFUNC4 | MFUNC4 |
| C13 | CAD20 | A6 | G06 | MFUNC1 | MFUNC1 |
| C14 | CC/$\overline{\text{BE2}}$ | A12 | G14 | $V_{CCCB}$ | $V_{CCCB}$ |
| C15 | CCLK | A16 | G15 | CC/$\overline{\text{BE1}}$ | A8 |
| D01 | NC | NC | G17 | CAD15 | $\overline{\text{IOWR}}$ |
| D19 | $\overline{\text{CGNT}}$ | $\overline{\text{WE}}$ | G18 | CAD13 | $\overline{\text{IORD}}$ |
| E01 | GND | GND | G19 | GND | GND |
| E02 | SPKROUT | SPKROUT | H01 | PCLK | PCLK |
| E03 | NC | NC | H02 | $\overline{\text{GRST}}$ | $\overline{\text{GRST}}$ |
| E05 | NC | NC | H03 | $\overline{\text{PRST}}$ | $\overline{\text{PRST}}$ |
| E06 | $\overline{\text{VCCD0}}$ | $\overline{\text{VCCD0}}$ | H05 | $\overline{\text{VR_EN}}$ | $\overline{\text{VR_EN}}$ |

† Terminal F06 is not bonded out in the packaged parts in order to have pin compatibility with future devices.

## Table 2−3. Signal Names Sorted by GVF Terminal Number (Continued)

| TERMINAL | SIGNAL NAME | | TERMINAL | SIGNAL NAME | |
|---|---|---|---|---|---|
| | CARDBUS | 16-BIT | | CARDBUS | 16-BIT |
| H06 | MFUNC6 | MFUNC6 | M17 | NC | NC |
| H14 | CAD11 | $\overline{OE}$ | M18 | NC | NC |
| H15 | CAD12 | A11 | M19 | NC | NC |
| H17 | CAD10 | $\overline{CE2}$ | N01 | GND | GND |
| H18 | CAD9 | A10 | N02 | AD19 | AD19 |
| H19 | $V_{CC}$ | $V_{CC}$ | N03 | AD18 | AD18 |
| J01 | $\overline{GNT}$ | $\overline{GNT}$ | N05 | $\overline{FRAME}$ | $\overline{FRAME}$ |
| J02 | $\overline{REQ}$ | $\overline{REQ}$ | N06 | AD17 | AD17 |
| J03 | $\overline{RI_OUT}/\overline{PME}$ | $\overline{RI_OUT}/\overline{PME}$ | N14 | NC | NC |
| J05 | AD31 | AD31 | N15 | NC | NC |
| J06 | AD30 | AD30 | N17 | NC | NC |
| J14 | CAD8 | D15 | N18 | NC | NC |
| J15 | CC/$\overline{BE0}$ | $\overline{CE1}$ | N19 | NC | NC |
| J17 | CAD7 | D7 | P01 | AD16 | AD16 |
| J18 | CRSVD | D14 | P02 | C/$\overline{BE2}$ | C/$\overline{BE2}$ |
| J19 | CAD5 | D6 | P03 | $\overline{IRDY}$ | $\overline{IRDY}$ |
| K01 | GND | GND | P05 | $\overline{STOP}$ | $\overline{STOP}$ |
| K02 | AD29 | AD29 | P06 | $\overline{TRDY}$ | $\overline{TRDY}$ |
| K03 | AD28 | AD28 | P07 | AD14 | AD14 |
| K05 | AD27 | AD27 | P08 | AD9 | AD9 |
| K06 | AD26 | AD26 | P09 | NC | NC |
| K14 | CAD6 | D13 | P10 | NC | NC |
| K15 | CAD3 | D5 | P11 | NC | NC |
| K17 | CAD4 | D12 | P12 | NC | NC |
| K18 | CAD1 | D4 | P13 | NC | NC |
| K19 | GND | GND | P14 | NC | NC |
| L01 | $V_{CCP}$ | $V_{CCP}$ | P15 | NC | NC |
| L02 | AD25 | AD25 | P17 | NC | NC |
| L03 | AD24 | AD24 | P18 | NC | NC |
| L05 | IDSEL | IDSEL | P19 | GND | GND |
| L06 | C/$\overline{BE3}$ | C/$\overline{BE3}$ | R01 | $V_{CC}$ | $V_{CC}$ |
| L14 | CAD2 | D11 | R02 | $\overline{DEVSEL}$ | $\overline{DEVSEL}$ |
| L15 | CAD0 | D3 | R03 | $\overline{PERR}$ | $\overline{PERR}$ |
| L17 | $\overline{CCD1}$ | $\overline{CD1}$ | R06 | AD15 | AD15 |
| L18 | VR_PORT | VR_PORT | R07 | AD10 | AD10 |
| L19 | $V_{CC}$ | $V_{CC}$ | R08 | AD6 | AD6 |
| M01 | $V_{CC}$ | $V_{CC}$ | R09 | AD0 | AD0 |
| M02 | AD23 | AD23 | R10 | NC | NC |
| M03 | AD22 | AD22 | R11 | NC | NC |
| M05 | AD20 | AD20 | R12 | NC | NC |
| M06 | AD21 | AD21 | R13 | NC | NC |
| M14 | NC | NC | R14 | NC | NC |
| M15 | NC | NC | R17 | NC | NC |

## Table 2−3. Signal Names Sorted by GVF Terminal Number (Continued)

| TERMINAL | SIGNAL NAME | | TERMINAL | SIGNAL NAME | |
|---|---|---|---|---|---|
| | CARDBUS | 16-BIT | | CARDBUS | 16-BIT |
| R18 | NC | NC | V10 | NC | NC |
| R19 | NC | NC | V11 | NC | NC |
| T01 | $\overline{\text{SERR}}$ | $\overline{\text{SERR}}$ | V12 | NC | NC |
| T19 | NC | NC | V13 | NC | NC |
| U05 | C/$\overline{\text{BE1}}$ | C/$\overline{\text{BE1}}$ | V14 | NC | NC |
| U06 | AD12 | AD12 | V15 | NC | NC |
| U07 | AD8 | AD8 | W04 | PAR | PAR |
| U08 | AD5 | AD5 | W05 | $V_{CCP}$ | $V_{CCP}$ |
| U09 | AD1 | AD1 | W06 | GND | GND |
| U10 | NC | NC | W07 | AD7 | AD7 |
| U11 | NC | NC | W08 | $V_{CC}$ | $V_{CC}$ |
| U12 | NC | NC | W09 | AD3 | AD3 |
| U13 | NC | NC | W10 | NC | NC |
| U14 | NC | NC | W11 | NC | NC |
| U15 | NC | NC | W12 | NC | NC |
| V05 | AD13 | AD13 | W13 | NC | NC |
| V06 | AD11 | AD11 | W14 | NC | NC |
| V07 | C/$\overline{\text{BE0}}$ | C/$\overline{\text{BE0}}$ | W15 | NC | NC |
| V08 | AD4 | AD4 | W16 | NC | NC |
| V09 | AD2 | AD2 | | | |

## Table 2−4. CardBus PC Card Signal Names Sorted Alphabetically to Device Terminals

| SIGNAL NAME | TERMINAL | | | SIGNAL NAME | TERMINAL | | |
|---|---|---|---|---|---|---|---|
| | PGE | GGU | GVF | | PGE | GGU | GVF |
| AD0 | 56 | N08 | R09 | CAD11 | 91 | G10 | H14 |
| AD1 | 55 | M07 | U09 | CAD12 | 92 | G12 | H15 |
| AD2 | 53 | L07 | V09 | CAD13 | 94 | F12 | G18 |
| AD3 | 52 | N06 | W09 | CAD14 | 96 | F11 | F19 |
| AD4 | 51 | K04 | V08 | CAD15 | 95 | C11 | G17 |
| AD5 | 50 | M06 | U08 | CAD16 | 97 | F10 | F18 |
| AD6 | 49 | L06 | R08 | CAD17 | 114 | B10 | B14 |
| AD7 | 48 | N05 | W07 | CAD18 | 115 | A12 | E13 |
| AD8 | 46 | N04 | U07 | CAD19 | 116 | B09 | A14 |
| AD9 | 45 | M02 | P08 | CAD20 | 118 | C09 | C13 |
| AD10 | 44 | M05 | R07 | CAD21 | 120 | C08 | E12 |
| AD11 | 43 | L04 | V06 | CAD22 | 121 | B12 | C12 |
| AD12 | 42 | N03 | U06 | CAD23 | 123 | D10 | A12 |
| AD13 | 41 | K05 | V05 | CAD24 | 127 | B07 | C11 |
| AD14 | 39 | L05 | P07 | CAD25 | 128 | D07 | E11 |
| AD15 | 38 | M04 | R06 | CAD26 | 129 | C07 | F11 |
| AD16 | 25 | J04 | P01 | CAD27 | 139 | A04 | F09 |
| AD17 | 24 | H01 | N06 | CAD28 | 140 | C05 | E09 |
| AD18 | 23 | H03 | N03 | CAD29 | 141 | C04 | B08 |
| AD19 | 22 | H02 | N02 | CAD30 | 142 | B04 | C08 |
| AD20 | 18 | G02 | M05 | CAD31 | 144 | B03 | E08 |
| AD21 | 17 | G04 | M06 | CAUDIO | 134 | D06 | F10 |
| AD22 | 16 | F01 | M03 | C/$\overline{\text{BE0}}$ | 47 | K06 | V07 |
| AD23 | 15 | C03 | M02 | C/$\overline{\text{BE1}}$ | 37 | M03 | U05 |
| AD24 | 11 | F03 | L03 | C/$\overline{\text{BE2}}$ | 26 | J02 | P02 |
| AD25 | 10 | E02 | L02 | C/$\overline{\text{BE3}}$ | 13 | A01 | L06 |
| AD26 | 9 | F04 | K06 | $\overline{\text{CBLOCK}}$ | 101 | A13 | E18 |
| AD27 | 7 | B01 | K05 | CC/$\overline{\text{BE0}}$ | 88 | H13 | J15 |
| AD28 | 6 | D02 | K03 | CC/$\overline{\text{BE1}}$ | 98 | E11 | G15 |
| AD29 | 5 | E04 | K02 | CC/$\overline{\text{BE2}}$ | 113 | D09 | C14 |
| AD30 | 4 | D03 | J06 | CC/$\overline{\text{BE3}}$ | 124 | A08 | B11 |
| AD31 | 3 | E03 | J05 | $\overline{\text{CCD1}}$ | 75 | L13 | L17 |
| CAD0 | 76 | K11 | L15 | $\overline{\text{CCD2}}$ | 138 | B05 | C09 |
| CAD1 | 78 | K12 | K18 | CCLK | 107 | B13 | C15 |
| CAD2 | 77 | M12 | L14 | $\overline{\text{CCLKRUN}}$ | 136 | A05 | B09 |
| CAD3 | 81 | J13 | K15 | $\overline{\text{CDEVSEL}}$ | 106 | D11 | A16 |
| CAD4 | 79 | J11 | K17 | $\overline{\text{CFRAME}}$ | 111 | A10 | B15 |
| CAD5 | 83 | J10 | J19 | $\overline{\text{CGNT}}$ | 105 | D13 | D19 |
| CAD6 | 82 | K13 | K14 | $\overline{\text{CINT}}$ | 131 | A06 | C10 |
| CAD7 | 86 | H12 | J17 | $\overline{\text{CIRDY}}$ | 110 | C10 | F13 |
| CAD8 | 87 | H11 | J14 | CLK_48_RSVD | 85 | H10 | — |
| CAD9 | 89 | G11 | H18 | CPAR | 100 | E12 | F14 |
| CAD10 | 90 | G13 | H17 | $\overline{\text{CPERR}}$ | 102 | D12 | F15 |

**Table 2−4. CardBus PC Card Signal Names Sorted Alphabetically to Device Terminals (Continued)**

| SIGNAL NAME | TERMINAL | | | SIGNAL NAME | TERMINAL | | |
|---|---|---|---|---|---|---|---|
| | PGE | GGU | GVF | | PGE | GGU | GVF |
| $\overline{\text{CREQ}}$ | 122 | B08 | B12 | MFUNC4 | 67 | M10 | G05 |
| $\overline{\text{CRST}}$ | 119 | D08 | B13 | MFUNC5 | 68 | N12 | F01 |
| CRSVD | 84 | A03 | F08 | MFUNC6 | 69 | L10 | H06 |
| CRSVD | 99 | E13 | F17 | PAR | 35 | N01 | W04 |
| CRSVD | 143 | J12 | J18 | PCLK | 20 | G01 | H01 |
| $\overline{\text{CSERR}}$ | 133 | B06 | E10 | $\overline{\text{PERR}}$ | 33 | K03 | R03 |
| $\overline{\text{CSTOP}}$ | 103 | E10 | E17 | $\overline{\text{PRST}}$ | 19 | G03 | H03 |
| CSTSCHG | 135 | C06 | A09 | $\overline{\text{REQ}}$ | 1 | C02 | J02 |
| $\overline{\text{CTRDY}}$ | 108 | C12 | E14 | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ | 57 | M08 | J03 |
| CVS1 | 130 | B02 | B10 | $\overline{\text{SERR}}$ | 34 | L02 | T01 |
| CVS2 | 117 | A09 | F12 | SPKROUT | 61 | M09 | E02 |
| $\overline{\text{DEVSEL}}$ | 30 | K02 | R02 | $\overline{\text{STOP}}$ | 31 | L01 | P05 |
| $\overline{\text{FRAME}}$ | 27 | J01 | N05 | $\overline{\text{SUSPEND}}$ | 65 | N10 | G03 |
| $\overline{\text{GNT}}$ | 2 | C01 | J01 | $\overline{\text{TRDY}}$ | 29 | J03 | P06 |
| $\overline{\text{GRST}}$ | 66 | L11 | H02 | $\overline{\text{VCCD0}}$ | 73 | N13 | E06 |
| IDSEL | 14 | F02 | L05 | $\overline{\text{VCCD1}}$ | 74 | L12 | B05 |
| $\overline{\text{IRDY}}$ | 28 | K01 | P03 | VPPD0 | 71 | K09 | A04 |
| MFUNC0 | 58 | K07 | F05 | VPPD1 | 72 | M11 | C05 |
| MFUNC1 | 59 | N09 | G06 | $\overline{\text{VR_EN}}$ | 125 | D04 | H05 |
| MFUNC2 | 63 | L09 | F03 | VR_PORT | 62 | L08 | G02, L18 |
| MFUNC3 | 64 | K10 | F02 | | | | |

## Table 2−5. 16-Bit PC Card Signal Names Sorted Alphabetically to Device Terminals

| SIGNAL NAME | TERMINAL | | | SIGNAL NAME | TERMINAL | | |
|---|---|---|---|---|---|---|---|
| | PGE | GGU | GVF | | PGE | GGU | GVF |
| AD0 | 56 | N08 | R09 | A11 | 92 | G12 | H15 |
| AD1 | 55 | M07 | U09 | A12 | 113 | D09 | C14 |
| AD2 | 53 | L07 | V09 | A13 | 100 | E12 | F14 |
| AD3 | 52 | N06 | W09 | A14 | 102 | D12 | F15 |
| AD4 | 51 | K04 | V08 | A15 | 110 | C10 | F13 |
| AD5 | 50 | M06 | U08 | A16 | 107 | B13 | C15 |
| AD6 | 49 | L06 | R08 | A17 | 97 | F10 | F18 |
| AD7 | 48 | N05 | W07 | A18 | 99 | E13 | F17 |
| AD8 | 46 | N04 | U07 | A19 | 101 | A13 | E18 |
| AD9 | 45 | M02 | P08 | A20 | 103 | E10 | E17 |
| AD10 | 44 | M05 | R07 | A21 | 106 | D11 | A16 |
| AD11 | 43 | L04 | V06 | A22 | 108 | C12 | E14 |
| AD12 | 42 | N03 | U06 | A23 | 111 | A10 | B15 |
| AD13 | 41 | K05 | V05 | A24 | 114 | B10 | B14 |
| AD14 | 39 | L05 | P07 | A25 | 116 | B09 | A14 |
| AD15 | 38 | M04 | R06 | BVD1($\overline{\text{STSCHG}}$/$\overline{\text{RI}}$) | 135 | C06 | A09 |
| AD16 | 25 | J04 | P01 | BVD2($\overline{\text{SPKR}}$) | 134 | D06 | F10 |
| AD17 | 24 | H01 | N06 | C/$\overline{\text{BE0}}$ | 47 | K06 | V07 |
| AD18 | 23 | H03 | N03 | $\overline{\text{C/BE1}}$ | 37 | M03 | U05 |
| AD19 | 22 | H02 | N02 | C/$\overline{\text{BE2}}$ | 26 | J02 | P02 |
| AD20 | 18 | G02 | M05 | C/$\overline{\text{BE3}}$ | 13 | A01 | L06 |
| AD21 | 17 | G04 | M06 | $\overline{\text{CD1}}$ | 75 | L13 | L17 |
| AD22 | 16 | F01 | M03 | $\overline{\text{CD2}}$ | 138 | B05 | C09 |
| AD23 | 15 | C03 | M02 | $\overline{\text{CE1}}$ | 88 | H13 | J15 |
| AD24 | 11 | F03 | L03 | $\overline{\text{CE2}}$ | 90 | G13 | H17 |
| AD25 | 10 | E02 | L02 | CLK_48_RSVD | 85 | H10 | — |
| AD26 | 9 | F04 | K06 | $\overline{\text{DEVSEL}}$ | 30 | K02 | R02 |
| AD27 | 7 | B01 | K05 | D0 | 139 | A04 | F09 |
| AD28 | 6 | D02 | K03 | D1 | 141 | C04 | B08 |
| AD29 | 5 | E04 | K02 | D2 | 143 | A03 | F08 |
| AD30 | 4 | D03 | J06 | D3 | 76 | K11 | L15 |
| AD31 | 3 | E03 | J05 | D4 | 78 | K12 | K18 |
| A0 | 129 | C07 | F11 | D5 | 81 | J13 | K15 |
| A1 | 128 | D07 | E11 | D6 | 83 | J10 | J19 |
| A2 | 127 | B07 | C11 | D7 | 86 | H12 | J17 |
| A3 | 123 | D10 | A12 | D8 | 140 | C05 | E09 |
| A4 | 121 | B12 | C12 | D9 | 142 | B04 | C08 |
| A5 | 120 | C08 | E12 | D10 | 144 | B03 | E08 |
| A6 | 118 | C09 | C13 | D11 | 77 | M12 | L14 |
| A7 | 115 | A12 | E13 | D12 | 79 | J11 | K17 |
| A8 | 98 | E11 | G15 | D13 | 82 | K13 | K14 |
| A9 | 96 | F11 | F19 | D14 | 84 | J12 | J18 |
| A10 | 89 | G11 | H18 | D15 | 87 | H11 | J14 |

**Table 2−5. 16-Bit PC Card Signal Names Sorted Alphabetically to Device Terminals (Continued)**

| SIGNAL NAME | TERMINAL | | | SIGNAL NAME | TERMINAL | | |
|---|---|---|---|---|---|---|---|
| | PGE | GGU | GVF | | PGE | GGU | GVF |
| $\overline{\text{FRAME}}$ | 27 | J01 | N05 | $\overline{\text{REG}}$ | 124 | A08 | B11 |
| $\overline{\text{GNT}}$ | 2 | C01 | J01 | $\overline{\text{REQ}}$ | 1 | C02 | J02 |
| $\overline{\text{GRST}}$ | 66 | L11 | H02 | RESET | 119 | D08 | B13 |
| IDSEL | 14 | F02 | L05 | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ | 57 | M08 | J03 |
| $\overline{\text{INPACK}}$ | 122 | B08 | B12 | $\overline{\text{SERR}}$ | 34 | L02 | T01 |
| $\overline{\text{IORD}}$ | 94 | F12 | G18 | SPKROUT | 61 | M09 | E02 |
| $\overline{\text{IOWR}}$ | 95 | C11 | G17 | $\overline{\text{STOP}}$ | 31 | L01 | P05 |
| $\overline{\text{IRDY}}$ | 28 | K01 | P03 | $\overline{\text{SUSPEND}}$ | 65 | N10 | G03 |
| MFUNC0 | 58 | K07 | F05 | $\overline{\text{TRDY}}$ | 29 | J03 | P06 |
| MFUNC1 | 59 | N09 | G06 | $\overline{\text{VCCD0}}$ | 73 | N13 | E06 |
| MFUNC2 | 63 | L09 | F03 | $\overline{\text{VCCD1}}$ | 74 | L12 | B05 |
| MFUNC3 | 64 | K10 | F02 | VPPD0 | 71 | K09 | A04 |
| MFUNC4 | 67 | M10 | G05 | VPPD1 | 72 | M11 | C05 |
| MFUNC5 | 68 | N12 | F01 | $\overline{\text{VR_EN}}$ | 125 | D04 | H05 |
| MFUNC6 | 69 | L10 | H06 | VR_PORT | 62 | L08 | G02, L18 |
| $\overline{\text{OE}}$ | 91 | G10 | H14 | $\overline{\text{VS1}}$ | 130 | B02 | B10 |
| PAR | 35 | N01 | W04 | $\overline{\text{VS2}}$ | 117 | A09 | F12 |
| PCLK | 20 | G01 | H01 | $\overline{\text{WAIT}}$ | 133 | B06 | E10 |
| $\overline{\text{PERR}}$ | 33 | K03 | R03 | $\overline{\text{WE}}$ | 105 | D13 | D19 |
| $\overline{\text{PRST}}$ | 19 | G03 | H03 | WP($\overline{\text{IOIS16}}$) | 136 | A05 | B09 |
| READY($\overline{\text{IREQ}}$) | 131 | A06 | C10 | | | | |

## 2.2 Terminal Descriptions

The terminals are grouped in tables by functionality, such as PCI system function, power-supply function, etc. The terminal numbers are also listed for convenient reference.

**Table 2−6. Power Supply Terminals**

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| GND | 8, 21, 40, 60, 80, 93, 112, 132 | A02, A11, D01, F13, H04, K08, M13, N02 | A07, A10, A15, E01, G19, K01, K19, N01, P19, W06 | | Device ground terminals |
| $V_{CC}$ | 12, 32, 54, 70, 104, 126, 137 | A07, C13, D05, E01, M01, N07, N11 | A05, A08, A13, E19, G01, H19, L19, M01, R01, W08 | | Power supply terminals for I/O and internal voltage regulator |
| $V_{CCCB}$ | 109 | B11 | A11, G14 | | Clamp voltage for PC Card interface. Matches card signaling environment, 5 V or 3.3 V |
| $V_{CCP}$ | 36 | L03 | L01, W05 | | Clamp voltage for PCI and miscellaneous I/O, 5 V or 3.3 V |
| $\overline{VR_EN}$ | 125 | D04 | H05 | I | Internal voltage regulator enable. Active-low |
| VR_PORT | 62 | L08 | G02, L18 | | Internal voltage regulator input/output. When $\overline{VR_EN}$ is low, the regulator is enabled and this terminal is an output. An external bypass capacitor is required on this terminal. When $\overline{VR_EN}$ is high, the regulator is disabled and this terminal is an input for an external 2.5-V core power source. |

**Table 2−7. PC Card Power Switch Terminals**

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| $\overline{VCCD0}$<br>$\overline{VCCD1}$ | 73<br>74 | N13<br>L12 | E06<br>B05 | O | Logic controls to the TPS2211A PC Card power interface switch to control AVCC |
| VPPD0<br>VPPD1 | 71<br>72 | K09<br>M11 | A04<br>C05 | O | Logic controls to the TPS2211A PC Card power interface switch to control AVPP |

**Table 2−8. PCI System Terminals**

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| $\overline{\text{GRST}}$ | 66 | L11 | H02 | I | Global reset. When the global reset is asserted, the $\overline{\text{GRST}}$ signal causes the controller to place all output buffers in a high-impedance state and reset all internal registers. When $\overline{\text{GRST}}$ is asserted, the device is completely in its default state. For systems that require wake-up from D3, $\overline{\text{GRST}}$ normally is asserted only during initial boot. $\overline{\text{PRST}}$ must be asserted following initial boot so that PME context is retained during the transition from D3 to D0.<br>When the $\overline{\text{SUSPEND}}$ mode is enabled, the device is protected from $\overline{\text{GRST}}$, and the internal registers are preserved. All outputs are placed in a high-impedance state. |
| PCLK | 20 | G01 | H01 | I | PCI bus clock. PCLK provides timing for all transactions on the PCI bus. All PCI signals are sampled at the rising edge of PCLK. |
| $\overline{\text{PRST}}$ | 19 | G03 | H03 | I | PCI bus reset. When the PCI bus reset is asserted, $\overline{\text{PRST}}$ causes the controller to place all output buffers in a high-impedance state and reset internal registers. When $\overline{\text{PRST}}$ is asserted, the device can generate the $\overline{\text{PME}}$ signal only if it is enabled. After $\overline{\text{PRST}}$ is deasserted, the controller is in a default state.<br>When the $\overline{\text{SUSPEND}}$ mode is enabled, the device is protected from $\overline{\text{PRST}}$, and the internal registers are preserved. All outputs are placed in a high-impedance state. |

## Table 2−9. PCI Address and Data Terminals

| TERMINAL NAME | TERMINAL NUMBER PGE | TERMINAL NUMBER GGU | TERMINAL NUMBER GVF | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| AD31 | 3 | E03 | J05 | I/O | PCI address/data bus. These signals make up the multiplexed PCI address and data bus on the primary interface. During the address phase of a primary-bus PCI cycle, AD31–AD0 contain a 32-bit address or other destination information. During the data phase, AD31–AD0 contain data. |
| AD30 | 4 | D03 | J06 | | |
| AD29 | 5 | E04 | K02 | | |
| AD28 | 6 | D02 | K03 | | |
| AD27 | 7 | B01 | K05 | | |
| AD26 | 9 | F04 | K06 | | |
| AD25 | 10 | E02 | L02 | | |
| AD24 | 11 | F03 | L03 | | |
| AD23 | 15 | C03 | M02 | | |
| AD22 | 16 | F01 | M03 | | |
| AD21 | 17 | G04 | M06 | | |
| AD20 | 18 | G02 | M05 | | |
| AD19 | 22 | H02 | N02 | | |
| AD18 | 23 | H03 | N03 | | |
| AD17 | 24 | H01 | N06 | | |
| AD16 | 25 | J04 | P01 | | |
| AD15 | 38 | M04 | R06 | | |
| AD14 | 39 | L05 | P07 | | |
| AD13 | 41 | K05 | V05 | | |
| AD12 | 42 | N03 | U06 | | |
| AD11 | 43 | L04 | V06 | | |
| AD10 | 44 | M05 | R04 | | |
| AD9 | 45 | M02 | P08 | | |
| AD8 | 46 | N04 | U07 | | |
| AD7 | 48 | N05 | W07 | | |
| AD6 | 49 | L06 | R08 | | |
| AD5 | 50 | M06 | U08 | | |
| AD4 | 51 | K04 | V08 | | |
| AD3 | 52 | N06 | W09 | | |
| AD2 | 53 | L07 | V09 | | |
| AD1 | 55 | M07 | U09 | | |
| AD0 | 56 | N08 | R09 | | |
| C/$\overline{BE3}$ | 13 | A01 | L06 | I/O | PCI-bus commands and byte enables. These signals are multiplexed on the same PCI terminals. During the address phase of a primary-bus PCI cycle, C/$\overline{BE3}$–C/$\overline{BE0}$ define the bus command. During the data phase, this 4-bit bus is used as a byte enable. The byte enable determines which byte paths of the full 32-bit data bus carry meaningful data. C/$\overline{BE0}$ applies to byte 0 (AD7–AD0), C/$\overline{BE1}$ applies to byte 1 (AD15–AD8), C/$\overline{BE2}$ applies to byte 2 (AD23–AD16), and C/$\overline{BE3}$ applies to byte 3 (AD31–AD24). |
| C/$\overline{BE2}$ | 26 | J02 | P02 | | |
| C/$\overline{BE1}$ | 37 | M03 | U05 | | |
| C/$\overline{BE0}$ | 47 | K06 | V07 | | |
| PAR | 35 | N01 | W04 | I/O | PCI-bus parity. In all PCI-bus read and write cycles, the controller calculates even parity across the AD31–AD0 and C/$\overline{BE3}$–C/$\overline{BE0}$ buses. As an initiator during PCI cycles, the controller outputs this parity indicator with a one-PCLK delay. As a target during PCI cycles, the controller compares its calculated parity to the parity indicator of the initiator. A compare error results in the assertion of a parity error ($\overline{PERR}$). |

**Table 2−10. PCI Interface Control Terminals**

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| $\overline{\text{DEVSEL}}$ | 30 | K02 | R02 | I/O | PCI device select. The controller asserts $\overline{\text{DEVSEL}}$ to claim a PCI cycle as the target device. As a PCI initiator on the bus, the controller monitors $\overline{\text{DEVSEL}}$ until a target responds. If no target responds before timeout occurs, then the controller terminates the cycle with an initiator abort. |
| $\overline{\text{FRAME}}$ | 27 | J01 | N05 | I/O | PCI cycle frame. $\overline{\text{FRAME}}$ is driven by the initiator of a bus cycle. $\overline{\text{FRAME}}$ is asserted to indicate that a bus transaction is beginning, and data transfers continue while this signal is asserted. When $\overline{\text{FRAME}}$ is deasserted, the PCI bus transaction is in the final data phase. |
| $\overline{\text{GNT}}$ | 2 | C01 | J01 | I | PCI bus grant. $\overline{\text{GNT}}$ is driven by the PCI bus arbiter to grant the controller access to the PCI bus after the current data transaction has completed. $\overline{\text{GNT}}$ may or may not follow a PCI bus request, depending on the PCI bus parking algorithm. |
| IDSEL | 14 | F02 | L05 | I | Initialization device select. IDSEL selects the controller during configuration space accesses. IDSEL can be connected to one of the upper 24 PCI address lines on the PCI bus. |
| $\overline{\text{IRDY}}$ | 28 | K01 | P03 | I/O | PCI initiator ready. $\overline{\text{IRDY}}$ indicates the ability of the PCI bus initiator to complete the current data phase of the transaction. A data phase is completed on a rising edge of PCLK where both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are asserted. Until $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are both sampled asserted, wait states are inserted. |
| $\overline{\text{PERR}}$ | 33 | K03 | R03 | I/O | PCI parity error indicator. $\overline{\text{PERR}}$ is driven by a PCI device to indicate that calculated parity does not match PAR when $\overline{\text{PERR}}$ is enabled through bit 6 of the command register (PCI offset 04h, see Section 4.4). |
| $\overline{\text{REQ}}$ | 1 | C02 | J02 | O | PCI bus request. $\overline{\text{REQ}}$ is asserted by the controller to request access to the PCI bus as an initiator. |
| $\overline{\text{SERR}}$ | 34 | L02 | T01 | O | PCI system error. $\overline{\text{SERR}}$ is an output that is pulsed from the controller when enabled through bit 8 of the command register (PCI offset 04h, see Section 4.4) indicating a system error has occurred. The controller need not be the target of the PCI cycle to assert this signal. When $\overline{\text{SERR}}$ is enabled in the command register, this signal also pulses, indicating that an address parity error has occurred on a CardBus interface. |
| $\overline{\text{STOP}}$ | 31 | L01 | P05 | I/O | PCI cycle stop signal. $\overline{\text{STOP}}$ is driven by a PCI target to request the initiator to stop the current PCI bus transaction. $\overline{\text{STOP}}$ is used for target disconnects and is commonly asserted by target devices that do not support burst data transfers. |
| $\overline{\text{TRDY}}$ | 29 | J03 | P06 | I/O | PCI target ready. $\overline{\text{TRDY}}$ indicates the ability of the primary bus target to complete the current data phase of the transaction. A data phase is completed on a rising edge of PCLK when both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are asserted. Until both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are asserted, wait states are inserted. |

## Table 2−11. Multifunction and Miscellaneous Terminals

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| CLK_48_RSVD | 85 | H10 | — | | No connect. These terminals have no connection anywhere within the package. Terminals H10 on the GGU package and 85 on the PGE package will be used as a 48-MHz clock input on future-generation devices. |
| MFUNC0 | 58 | K07 | F05 | I/O | Multifunction terminal 0. MFUNC0 can be configured as parallel PCI interrupt $\overline{\text{INTA}}$, GPI0, GPO0, socket activity LED output, ZV switching output, CardBus audio PWM, $\overline{\text{GPE}}$, or a parallel IRQ. See Section 4.30, *Multifunction Routing Register*, for configuration details. |
| MFUNC1 | 59 | N09 | G06 | I/O | Multifunction terminal 1. MFUNC1 can be configured as GPI1, GPO1, socket activity LED output, $\overline{\text{D3_STAT}}$, ZV switching output, CardBus audio PWM, $\overline{\text{GPE}}$, or a parallel IRQ. See Section 4.30, *Multifunction Routing Register*, for configuration details.<br>Serial data (SDA). When $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$ are detected high after a global reset, the MFUNC1 terminal provides the SDA signaling for the serial bus interface. The two-terminal serial interface loads the subsystem identification and other register defaults from an EEPROM after a global reset. See Section 3.6.1, *Serial Bus Interface Implementation*, for details on other serial bus applications. |
| MFUNC2 | 63 | L09 | F03 | I/O | Multifunction terminal 2. MFUNC2 can be configured as GPI2, GPO2, socket activity LED output, ZV switching output, CardBus audio PWM, $\overline{\text{GPE}}$, $\overline{\text{RI_OUT}}$, $\overline{\text{D3_STAT}}$, or a parallel IRQ. See Section 4.30, *Multifunction Routing Register*, for configuration details. |
| MFUNC3/ IRQSER | 64 | K10 | F02 | I/O | Multifunction terminal 3. MFUNC3 can be configured as a parallel IRQ or the serialized interrupt signal IRQSER. This terminal is IRQSER by default. See Section 4.30, *Multifunction Routing Register*, for configuration details. |
| MFUNC4 | 67 | M10 | G05 | I/O | Multifunction terminal 4. MFUNC4 can be configured as PCI $\overline{\text{LOCK}}$, GPI3, GPO3, socket activity LED output, ZV switching output, CardBus audio PWM, $\overline{\text{GPE}}$, $\overline{\text{D3_STAT}}$, $\overline{\text{RI_OUT}}$, or a parallel IRQ. See Section 4.30, *Multifunction Routing Register*, for configuration details.<br>Serial clock (SCL). When $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$ are detected high after a global reset, the MFUNC4 terminal provides the SCL signaling for the serial bus interface. The two-terminal serial interface loads the subsystem identification and other register defaults from an EEPROM after a global reset. See Section 3.6.1, *Serial Bus Interface Implementation*, for details on other serial bus applications. |
| MFUNC5 | 68 | N12 | F01 | I/O | Multifunction terminal 5. MFUNC5 can be configured as GPI4, GPO4, socket activity LED output, ZV switching output, CardBus audio PWM, $\overline{\text{D3_STAT}}$, $\overline{\text{GPE}}$, or a parallel IRQ. See Section 4.30, *Multifunction Routing Register*, for configuration details. |
| $\overline{\text{MFUNC6}}$/ $\overline{\text{CLKRUN}}$ | 69 | L10 | H06 | I/O | Multifunction terminal 6. MFUNC6 can be configured as a PCI $\overline{\text{CLKRUN}}$ or a parallel IRQ. See Section 4.30, *Multifunction Routing Register*, for configuration details. |
| $\overline{\text{RI_OUT}}$ / $\overline{\text{PME}}$ | 57 | M08 | J03 | O | Ring indicate out and power management event output. Terminal provides an output for ring- indicate or $\overline{\text{PME}}$ signals. |
| SPKROUT | 61 | M09 | E02 | O | Speaker output. SPKROUT is the output to the host system that can carry $\overline{\text{SPKR}}$ or CAUDIO through the controller from the PC Card interface. SPKROUT is driven as the exclusive-OR combination of card $\overline{\text{SPKR}}$//CAUDIO inputs. |
| $\overline{\text{SUSPEND}}$ | 65 | N10 | G03 | I | Suspend. $\overline{\text{SUSPEND}}$ protects the internal registers from clearing when the $\overline{\text{GRST}}$ or $\overline{\text{PRST}}$ signal is asserted. See Section 3.8.5, *Suspend Mode*, for details. |

**Table 2−12. 16-Bit PC Card Address and Data Terminals**

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| A25 | 116 | B09 | A14 | O | PC Card address. 16-bit PC Card address lines. A25 is the most significant bit. |
| A24 | 114 | B10 | B14 | | |
| A23 | 111 | A10 | B15 | | |
| A22 | 108 | C12 | E14 | | |
| A21 | 106 | D11 | A16 | | |
| A20 | 103 | E10 | E17 | | |
| A19 | 101 | A13 | E18 | | |
| A18 | 99 | E13 | F17 | | |
| A17 | 97 | F10 | F18 | | |
| A16 | 107 | B13 | C15 | | |
| A15 | 110 | C10 | F13 | | |
| A14 | 102 | D12 | F15 | | |
| A13 | 100 | E12 | F14 | | |
| A12 | 113 | D09 | C14 | | |
| A11 | 92 | G12 | H15 | | |
| A10 | 89 | G11 | H18 | | |
| A9 | 96 | F11 | F19 | | |
| A8 | 98 | E11 | G15 | | |
| A7 | 115 | A12 | E13 | | |
| A6 | 118 | C09 | C13 | | |
| A5 | 120 | C08 | E12 | | |
| A4 | 121 | B12 | C12 | | |
| A3 | 123 | D10 | A12 | | |
| A2 | 127 | B07 | C11 | | |
| A1 | 128 | D07 | E11 | | |
| A0 | 129 | C07 | F11 | | |
| D15 | 87 | H11 | J14 | I/O | PC Card data. 16-bit PC Card data lines. D15 is the most significant bit. |
| D14 | 84 | J12 | J18 | | |
| D13 | 82 | K13 | K14 | | |
| D12 | 79 | J11 | K17 | | |
| D11 | 77 | M12 | L14 | | |
| D10 | 144 | B03 | E08 | | |
| D9 | 142 | B04 | C08 | | |
| D8 | 140 | C05 | E09 | | |
| D7 | 86 | H12 | J17 | | |
| D6 | 83 | J10 | J19 | | |
| D5 | 81 | J13 | K15 | | |
| D4 | 78 | K12 | K18 | | |
| D3 | 76 | K11 | L15 | | |
| D2 | 143 | A03 | F08 | | |
| D1 | 141 | C04 | B08 | | |
| D0 | 139 | A04 | F09 | | |

## Table 2−13. 16-Bit PC Card Interface Control Terminals

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| BVD1 ($\overline{\text{STSCHG}}$/$\overline{\text{RI}}$) | 135 | C06 | A09 | I | Battery voltage detect 1. BVD1 is generated by 16-bit memory PC Cards that include batteries. BVD1 is used with BVD2 as an indication of the condition of the batteries on a memory PC Card. Both BVD1 and BVD2 are high when the battery is good. When BVD2 is low and BVD1 is high, the battery is weak and should be replaced. When BVD1 is low, the battery is no longer serviceable and the data in the memory PC Card is lost. See Section 5.6, *ExCA Card Status-Change Interrupt Configuration Register*, for enable bits. See Section 5.5, *ExCA Card Status-Change Register*, and Section 5.2, *ExCA Interface Status Register*, for the status bits for this signal.<br>Status change. $\overline{\text{STSCHG}}$ is used to alert the system to a change in the READY, write protect, or battery voltage dead condition of a 16-bit I/O PC Card.<br>Ring indicate. $\overline{\text{RI}}$ is used by 16-bit modem cards to indicate a ring detection. |
| BVD2($\overline{\text{SPKR}}$) | 134 | D06 | F10 | I | Battery voltage detect 2. BVD2 is generated by 16-bit memory PC Cards that include batteries. BVD2 is used with BVD1 as an indication of the condition of the batteries on a memory PC Card. Both BVD1 and BVD2 are high when the battery is good. When BVD2 is low and BVD1 is high, the battery is weak and should be replaced. When BVD1 is low, the battery is no longer serviceable and the data in the memory PC Card is lost. See Section 5.6, *ExCA Card Status-Change Interrupt Configuration Register*, for enable bits. See Section 5.5, *ExCA Card Status-Change Register*, and Section 5.2, *ExCA Interface Status Register*, for the status bits for this signal.<br>Speaker. $\overline{\text{SPKR}}$ is an optional binary audio signal available only when the card and socket have been configured for the 16-bit I/O interface. |
| $\overline{\text{CD1}}$<br>$\overline{\text{CD2}}$ | 75<br>138 | L13<br>B05 | L17<br>C09 | I | Card detect 1 and card detect 2. $\overline{\text{CD1}}$ and $\overline{\text{CD2}}$ are internally connected to ground on the PC Card. When a PC Card is inserted into a socket, $\overline{\text{CD1}}$ and $\overline{\text{CD2}}$ are pulled low. For signal status, see Section 5.2, *ExCA Interface Status Register*. |
| $\overline{\text{CE1}}$<br>$\overline{\text{CE2}}$ | 88<br>90 | H13<br>G13 | J15<br>H17 | O | Card enable 1 and card enable 2. $\overline{\text{CE1}}$ and $\overline{\text{CE2}}$ enable even- and odd-numbered address bytes. $\overline{\text{CE1}}$ enables even-numbered address bytes, and $\overline{\text{CE2}}$ enables odd-numbered address bytes. |
| $\overline{\text{INPACK}}$ | 122 | B08 | B12 | I | Input acknowledge. $\overline{\text{INPACK}}$ is asserted by the PC Card when it can respond to an I/O read cycle at the current address. |
| $\overline{\text{IORD}}$ | 94 | F12 | G18 | O | I/O read. $\overline{\text{IORD}}$ is asserted by the controller to enable 16-bit I/O PC Card data output during host I/O read cycles. |
| $\overline{\text{IOWR}}$ | 95 | C11 | G17 | O | I/O write. $\overline{\text{IOWR}}$ is driven low by the controller to strobe write data into 16-bit I/O PC Cards during host I/O write cycles. |
| $\overline{\text{OE}}$ | 91 | G10 | H14 | O | Output enable. $\overline{\text{OE}}$ is driven low by the controller to enable 16-bit memory PC Card data output during host memory read cycles. |
| READY ($\overline{\text{IREQ}}$) | 131 | A06 | C10 | I | Ready. The ready function is provided by READY when the 16-bit PC Card and the host socket are configured for the memory-only interface. READY is driven low by the 16-bit memory PC Cards to indicate that the memory card circuits are busy processing a previous write command. READY is driven high when the 16-bit memory PC Card is ready to accept a new data transfer command.<br>Interrupt request. $\overline{\text{IREQ}}$ is asserted by a 16-bit I/O PC Card to indicate to the host that a device on the 16-bit I/O PC Card requires service by the host software. $\overline{\text{IREQ}}$ is high (deasserted) when no interrupt is requested. |
| $\overline{\text{REG}}$ | 124 | A08 | B11 | O | Attribute memory select. $\overline{\text{REG}}$ remains high for all common memory accesses. When $\overline{\text{REG}}$ is asserted, access is limited to attribute memory ($\overline{\text{OE}}$ or $\overline{\text{WE}}$ active) and to the I/O space ($\overline{\text{IORD}}$ or $\overline{\text{IOWR}}$ active). Attribute memory is a separately accessed section of card memory and is generally used to record card capacity and other configuration and attribute information. |
| RESET | 119 | D08 | B13 | O | PC Card reset. RESET forces a hard reset to a 16-bit PC Card. |

**Table 2−13. 16-Bit PC Card Interface Control Terminals (Continued)**

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| $\overline{VS1}$<br>$\overline{VS2}$ | 130<br>117 | B02<br>A09 | B10<br>F12 | I/O | Voltage sense 1 and voltage sense 2. $\overline{VS1}$ and $\overline{VS2}$, when used in conjunction with each other, determine the operating voltage of the PC Card. |
| $\overline{WAIT}$ | 133 | B06 | E10 | I | Bus cycle wait. $\overline{WAIT}$ is driven by a 16-bit PC Card to extend the completion of the memory or I/O in progress. |
| $\overline{WE}$ | 105 | D13 | D19 | O | Write enable. $\overline{WE}$ is used to strobe memory write data into 16-bit memory PC Cards. $\overline{WE}$ is also used for memory PC Cards that employ programmable memory technologies. |
| WP<br>($\overline{IOIS16}$) | 136 | A05 | B09 | I | Write protect. WP applies to 16-bit memory PC Cards. WP reflects the status of the write-protect switch on 16-bit memory PC Cards. For 16-bit I/O cards, WP is used for the 16-bit port ($\overline{IOIS16}$) function.<br>I/O is 16 bits. $\overline{IOIS16}$ applies to 16-bit I/O PC Cards. $\overline{IOIS16}$ is asserted by the 16-bit PC Card when the address on the bus corresponds to an address to which the 16-bit PC Card responds, and the I/O port that is addressed is capable of 16-bit accesses. |

**Table 2−14. CardBus PC Card Interface System Terminals**

| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| CCLK | 107 | B13 | C15 | O | CardBus clock. CCLK provides synchronous timing for all transactions on the CardBus interface. All signals except $\overline{CRST}$, $\overline{CCLKRUN}$, $\overline{CINT}$, CSTSCHG, CAUDIO, $\overline{CCD2}$, $\overline{CCD1}$, CVS2, and CVS1 are sampled on the rising edge of CCLK, and all timing parameters are defined with the rising edge of this signal. CCLK operates at the PCI bus clock frequency, but it can be stopped in the low state or slowed down for power savings. |
| $\overline{CCLKRUN}$ | 136 | A05 | B09 | I/O | CardBus clock run. $\overline{CCLKRUN}$ is used by a CardBus PC Card to request an increase in the CCLK frequency, and by the controller to indicate that the CCLK frequency is going to be decreased. |
| $\overline{CRST}$ | 119 | D08 | B13 | O | CardBus reset. $\overline{CRST}$ brings CardBus PC Card-specific registers, sequencers, and signals to a known state. When $\overline{CRST}$ is asserted, all CardBus PC Card signals are placed in a high-impedance state, and the controller drives these signals to a valid logic level. Assertion can be asynchronous to CCLK, but deassertion must be synchronous to CCLK. |

**Table 2−15. CardBus PC Card Address and Data Terminals**

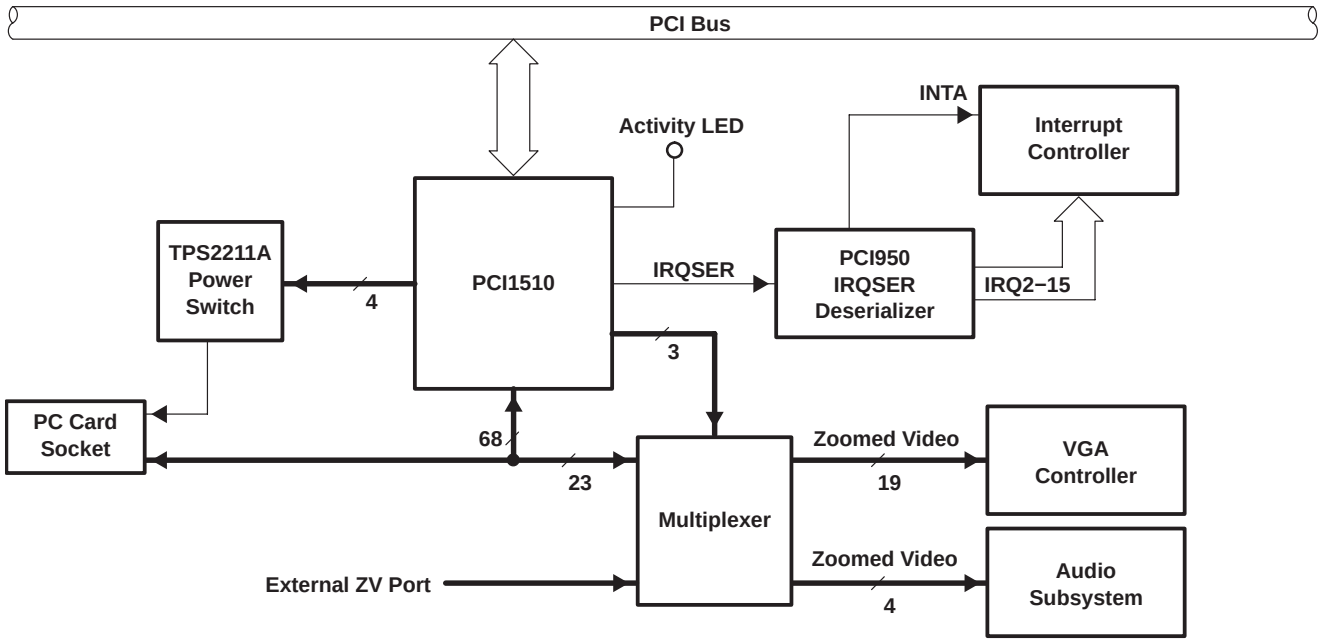
| TERMINAL | | | | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| NAME | NUMBER | | | | |
| | PGE | GGU | GVF | | |
| CAD31 | 144 | B03 | E08 | I/O | CardBus address and data. These signals make up the multiplexed CardBus address and data bus on the CardBus interface. During the address phase of a CardBus cycle, CAD31–CAD0 contain a 32-bit address. During the data phase of a CardBus cycle, CAD31–CAD0 contain data. CAD31 is the most significant bit. |
| CAD30 | 142 | B04 | C08 | | |
| CAD29 | 141 | C04 | B08 | | |
| CAD28 | 140 | C05 | E09 | | |
| CAD27 | 139 | A04 | F09 | | |
| CAD26 | 129 | C07 | F11 | | |
| CAD25 | 128 | D07 | E11 | | |
| CAD24 | 127 | B07 | C11 | | |
| CAD23 | 123 | D10 | A12 | | |
| CAD22 | 121 | B12 | C12 | | |
| CAD21 | 120 | C08 | E12 | | |
| CAD20 | 118 | C09 | C13 | | |
| CAD19 | 116 | B09 | A14 | | |
| CAD18 | 115 | A12 | E13 | | |
| CAD17 | 114 | B10 | B14 | | |
| CAD16 | 97 | F10 | F18 | | |
| CAD15 | 95 | C11 | G17 | | |
| CAD14 | 96 | F11 | F19 | | |
| CAD13 | 94 | F12 | G18 | | |
| CAD12 | 92 | G12 | H15 | | |
| CAD11 | 91 | G10 | H14 | | |
| CAD10 | 90 | G13 | H17 | | |
| CAD9 | 89 | G11 | H18 | | |
| CAD8 | 87 | H11 | J14 | | |
| CAD7 | 86 | H12 | J17 | | |
| CAD6 | 82 | K13 | K14 | | |
| CAD5 | 83 | J10 | J19 | | |
| CAD4 | 79 | J11 | K17 | | |
| CAD3 | 81 | J13 | K15 | | |
| CAD2 | 77 | M12 | L14 | | |
| CAD1 | 78 | K12 | K18 | | |
| CAD0 | 76 | K11 | L15 | | |
| CC/$\overline{\text{BE3}}$ | 124 | A08 | B11 | I/O | CardBus bus commands and byte enables. CC/$\overline{\text{BE3}}$–CC/$\overline{\text{BE0}}$ are multiplexed on the same CardBus terminals. During the address phase of a CardBus cycle, CC/$\overline{\text{BE3}}$–CC/$\overline{\text{BE0}}$ define the bus command. During the data phase, this 4-bit bus is used as a byte enable. The byte enable determines which byte paths of the full 32-bit data bus carry meaningful data. CC/$\overline{\text{BE0}}$ applies to byte 0 (CAD7–CAD0), CC/$\overline{\text{BE1}}$ applies to byte 1 (CAD15–CAD8), CC/$\overline{\text{BE2}}$ applies to byte 2 (CAD23–CAD16), and CC/$\overline{\text{BE3}}$ applies to byte (CAD31–CAD24). |
| CC/$\overline{\text{BE2}}$ | 113 | D09 | C14 | | |
| CC/$\overline{\text{BE1}}$ | 98 | E11 | G15 | | |
| CC/$\overline{\text{BE0}}$ | 88 | H13 | J15 | | |
| CPAR | 100 | E12 | F14 | I/O | CardBus parity. In all CardBus read and write cycles, the controller calculates even parity across the CAD and CC/$\overline{\text{BE}}$ buses. As an initiator during CardBus cycles, the controller outputs CPAR with a one-CCLK delay. As a target during CardBus cycles, the controller compares its calculated parity to the parity indicator of the initiator; a compare error results in a parity error assertion. |

## Table 2−16. CardBus PC Card Interface Control Terminals

| TERMINAL NAME | TERMINAL NUMBER PGE | TERMINAL NUMBER GGU | TERMINAL NUMBER GVF | I/O | DESCRIPTION |
|---|---|---|---|---|---|
| CAUDIO | 134 | D06 | F10 | I | CardBus audio. CAUDIO is a digital input signal from a PC Card to the system speaker. The controller supports the binary audio mode and outputs a binary signal from the card to SPKROUT. |
| $\overline{\text{CBLOCK}}$ | 101 | A13 | E18 | I/O | CardBus lock. $\overline{\text{CBLOCK}}$ is used to gain exclusive access to a target. |
| $\overline{\text{CCD1}}$<br>$\overline{\text{CCD2}}$ | 75<br>138 | L13<br>B05 | L17<br>C09 | I | CardBus detect 1 and CardBus detect 2. $\overline{\text{CCD1}}$ and $\overline{\text{CCD2}}$ are used in conjunction with CVS1 and CVS2 to identify card insertion and interrogate cards to determine the operating voltage and card type. |
| $\overline{\text{CDEVSEL}}$ | 106 | D11 | A16 | I/O | CardBus device select. The controller asserts $\overline{\text{CDEVSEL}}$ to claim a CardBus cycle as the target device. As a CardBus initiator on the bus, the controller monitors $\overline{\text{CDEVSEL}}$ until a target responds. If no target responds before timeout occurs, then the controller terminates the cycle with an initiator abort. |
| $\overline{\text{CFRAME}}$ | 111 | A10 | B15 | I/O | CardBus cycle frame. $\overline{\text{CFRAME}}$ is driven by the initiator of a CardBus bus cycle. $\overline{\text{CFRAME}}$ is asserted to indicate that a bus transaction is beginning, and data transfers continue while this signal is asserted.<br>When $\overline{\text{CFRAME}}$ is deasserted, the CardBus bus transaction is in the final data phase. |
| $\overline{\text{CGNT}}$ | 105 | D13 | D19 | O | CardBus bus grant. $\overline{\text{CGNT}}$ is driven by the controller to grant a CardBus PC Card access to the CardBus bus after the current data transaction has been completed. |
| $\overline{\text{CINT}}$ | 131 | A06 | C10 | I | CardBus interrupt. $\overline{\text{CINT}}$ is asserted low by a CardBus PC Card to request interrupt servicing from the host. |
| $\overline{\text{CIRDY}}$ | 110 | C10 | F13 | I/O | CardBus initiator ready. $\overline{\text{CIRDY}}$ indicates the ability of the CardBus initiator to complete the current data phase of the transaction. A data phase is completed on a rising edge of CCLK when both $\overline{\text{CIRDY}}$ and $\overline{\text{CTRDY}}$ are asserted. Until $\overline{\text{CIRDY}}$ and $\overline{\text{CTRDY}}$ are both sampled asserted, wait states are inserted. |
| $\overline{\text{CPERR}}$ | 102 | D12 | F15 | I/O | CardBus parity error. $\overline{\text{CPERR}}$ reports parity errors during CardBus transactions, except during special cycles. It is driven low by a target two clocks following the data cycle during which a parity error is detected. |
| $\overline{\text{CREQ}}$ | 122 | B08 | B12 | I | CardBus request. $\overline{\text{CREQ}}$ indicates to the arbiter that the CardBus PC Card desires use of the CardBus bus as an initiator. |
| $\overline{\text{CSERR}}$ | 133 | B06 | E10 | I | CardBus system error. $\overline{\text{CSERR}}$ reports address parity errors and other system errors that could lead to catastrophic results. $\overline{\text{CSERR}}$ is driven by the card synchronous to CCLK, but deasserted by a weak pullup; deassertion may take several CCLK periods. The controller can report $\overline{\text{CSERR}}$ to the system by assertion of $\overline{\text{SERR}}$ on the PCI interface. |
| $\overline{\text{CSTOP}}$ | 103 | E10 | E17 | I/O | CardBus stop. $\overline{\text{CSTOP}}$ is driven by a CardBus target to request the initiator to stop the current CardBus transaction. $\overline{\text{CSTOP}}$ is used for target disconnects, and is commonly asserted by target devices that do not support burst data transfers. |
| CSTSCHG | 135 | C06 | A09 | I | CardBus status change. CSTSCHG alerts the system to a change in the card status, and is used as a wake-up mechanism. |
| $\overline{\text{CTRDY}}$ | 108 | C12 | E14 | I/O | CardBus target ready. $\overline{\text{CTRDY}}$ indicates the ability of the CardBus target to complete the current data phase of the transaction. A data phase is completed on a rising edge of CCLK, when both $\overline{\text{CIRDY}}$ and $\overline{\text{CTRDY}}$ are asserted; until this time, wait states are inserted. |
| CVS1<br>CVS2 | 130<br>117 | B02<br>A09 | B10<br>F12 | I/O | CardBus voltage sense 1 and CardBus voltage sense 2. CVS1 and CVS2 are used in conjunction with $\overline{\text{CCD1}}$ and $\overline{\text{CCD2}}$ to identify card insertion and interrogate cards to determine the operating voltage and card type. |

# 3 Feature/Protocol Descriptions

The following sections give an overview of the PCI1510 controller. Figure 3−1 shows a simplified block diagram of the controller. The PCI interface includes all address/data and control signals for PCI protocol. The interrupt interface includes terminals for parallel PCI, parallel ISA, and serialized PCI and ISA signaling. Miscellaneous system interface terminals include multifunction terminals: $\overline{\text{SUSPEND}}$, $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ (power-management control signal), and SPKROUT.




NOTE: The PC Card interface is 68 terminals for CardBus and 16-bit PC Cards. In ZV mode, 23 terminals are used for routing the ZV signals to the VGA controller and audio subsystem.

**Figure 3−1. PCI1510 Simplified Block Diagram**

## 3.1 Power Supply Sequencing

The controller contains 3.3-V I/O buffers with 5-V tolerance requiring an I/O power supply and an LDO-VR power supply for core logic. The core power supply, which is always 2.5 V, can be supplied through the VR_PORT terminal (when $\overline{\text{VR_EN}}$ is high) or from the integrated LDO-VR. The LDO-VR needs a 3.3-V power supply via the $V_{CC}$ terminals. The clamping voltages ($V_{CCCB}$ and $V_{CCP}$) can be either 3.3 V or 5 V, depending on the interface. The following power-up and power-down sequences are recommended.

The power-up sequence is:

1. Assert $\overline{\text{GRST}}$ to the device to disable the outputs during power up. Output drivers must be powered up in the high-impedance state to prevent high current levels through the clamp diodes to the 5-V clamping rails ($V_{CCCB}$ and $V_{CCP}$).
2. Apply 3.3-V power to $V_{CC}$.
3. Apply the clamp voltage.

The power-down sequence is:

1. Assert $\overline{\text{GRST}}$ to the device to disable the outputs during power down. Output drivers must be powered down in the high-impedance state to prevent high current levels through the clamp diodes to the 5-V clamping rails ($V_{CCCB}$ and $V_{CCP}$).

2. Remove the clamp voltage.
3. Remove the 3.3-V power from $V_{CC}$.

**NOTE:** The clamp voltage can be ramped up or ramped down along with the 3.3-V power. The voltage difference between $V_{CC}$ and the clamp voltage must remain within 3.6 V.

## 3.2 I/O Characteristics

Figure 3−2 shows a 3-state bidirectional buffer. Section 7.2, *Recommended Operating Conditions*, provides the electrical characteristics of the inputs and outputs.

**NOTE:** The controller meets the ac specifications of the *PC Card Standard* and *PCI Local Bus Specification.*

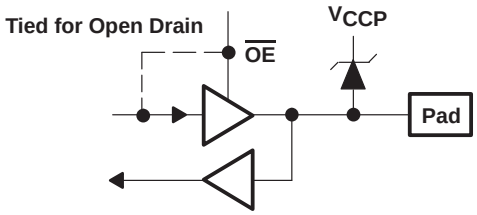



**Figure 3−2. 3-State Bidirectional Buffer**

**NOTE:** Unused terminals (input or I/O) must be held high or low to prevent them from floating.

## 3.3 Clamping Voltages

The clamping voltages are set to match whatever external environment the controller is interfaced with, 3.3 V or 5 V. The I/O sites can be pulled through a clamping diode to a voltage rail that protects the core from external signals. The core power supply is always 3.3 V and is independent of the clamping voltages. For example, PCI signaling can be either 3.3 V or 5 V, and the controller must reliably accommodate both voltage levels. This is accomplished by using a 3.3-V I/O buffer that is 5-V tolerant, with the applicable clamping voltage applied. If a system designer desires a 5-V PCI bus, then $V_{CCP}$ can be connected to a 5-V power supply.

The controller requires three separate clamping voltages because it supports a wide range of features. The three voltages are listed and defined in Section 7.2, *Recommended Operating Conditions*. $\overline{\text{GRST}}$, $\overline{\text{SUSPEND}}$, $\overline{\text{PME}}$, and CSTSCHG are not clamped to any of them.

## 3.4 Peripheral Component Interconnect (PCI) Interface

The controller is fully compliant with the *PCI Local Bus Specification*. The controller provides all required signals for PCI master or slave operation, and may operate in either a 5-V or 3.3-V signaling environment by connecting the $V_{CCP}$ terminal to the desired voltage level. In addition to the mandatory PCI signals, the controller provides the optional interrupt signal $\overline{\text{INTA}}$.

### 3.4.1 PCI $\overline{\text{GRST}}$ Signal

During the power-up sequence, $\overline{\text{GRST}}$ and $\overline{\text{PRST}}$ must be asserted. $\overline{\text{GRST}}$ can only be deasserted 100 µs after PCLK is stable. $\overline{\text{PRST}}$ can be deasserted at the same time as $\overline{\text{GRST}}$ or any time thereafter.

### 3.4.2 PCI Bus Lock ($\overline{\text{LOCK}}$)

The bus-locking protocol defined in the *PCI Local Bus Specification* is not highly recommended, but is provided on the controller as an additional compatibility feature. The PCI $\overline{\text{LOCK}}$ signal can be routed to the MFUNC4 terminal by setting the appropriate values in bits 19−16 of the multifunction routing register. See Section 4.30, *Multifunction Routing Register*, for details. Note that the use of $\overline{\text{LOCK}}$ is only supported by PCI-to-CardBus bridges in the downstream direction (away from the processor).

PCI $\overline{\text{LOCK}}$ indicates an atomic operation that may require multiple transactions to complete. When $\overline{\text{LOCK}}$ is asserted, nonexclusive transactions can proceed to an address that is not currently locked. A grant to start a transaction on the PCI bus does not guarantee control of $\overline{\text{LOCK}}$; control of $\overline{\text{LOCK}}$ is obtained under its own protocol. It is possible for different initiators to use the PCI bus while a single master retains ownership of $\overline{\text{LOCK}}$. Note that the CardBus signal for this protocol is $\overline{\text{CBLOCK}}$ to avoid confusion with the bus clock.

An agent may need to do an exclusive operation because a critical access to memory might be broken into several transactions, but the master wants exclusive rights to a region of memory. The granularity of the lock is defined by PCI to be 16 bytes, aligned. The $\overline{\text{LOCK}}$ protocol defined by the *PCI Local Bus Specification* allows a resource lock without interfering with nonexclusive real-time data transfer, such as video.

The PCI bus arbiter may be designed to support only complete bus locks using the $\overline{\text{LOCK}}$ protocol. In this scenario, the arbiter does not grant the bus to any other agent (other than the $\overline{\text{LOCK}}$ master) while $\overline{\text{LOCK}}$ is asserted. A complete bus lock may have a significant impact on the performance of the video. The arbiter that supports complete bus lock must grant the bus to the cache to perform a writeback due to a snoop to a modified line when a locked operation is in progress.

The controller supports all $\overline{\text{LOCK}}$ protocol associated with PCI-to-PCI bridges, as also defined for PCI-to-CardBus bridges. This includes disabling write posting while a locked operation is in progress, which can solve a potential deadlock when using devices such as PCI-to-PCI bridges. The potential deadlock can occur if a CardBus target supports delayed transactions and blocks access to the target until it completes a delayed read. This target characteristic is prohibited by the *PCI Local Bus Specification*, and the issue is resolved by the PCI master using $\overline{\text{LOCK}}$.

### 3.4.3 Loading Subsystem Identification

The subsystem vendor ID register (PCI offset 40h, see Section 4.26) and subsystem ID register (PCI offset 42h, see Section 4.27) make up a doubleword of PCI configuration space for function 0. This doubleword register is used for system and option card (mobile dock) identification purposes and is required by some operating systems. Implementation of this unique identifier register is a *PC 99/PC 2001* requirement.

The controller offers two mechanisms to load a read-only value into the subsystem registers. The first mechanism relies upon the system BIOS providing the subsystem ID value. The default access mode to the subsystem registers is read-only, but can be made read/write by clearing bit 5 (SUBSYSRW) in the system control register (PCI offset 80h, see Section 4.29). Once this bit is cleared, the BIOS can write a subsystem identification value into the registers at PCI offset 40h. The BIOS must set the SUBSYSRW bit such that the subsystem vendor ID register and subsystem ID register is limited to read-only access. This approach saves the added cost of implementing the serial electrically erasable programmable ROM (EEPROM).

In some conditions, such as in a docking environment, the subsystem vendor ID register and subsystem ID register must be loaded with a unique identifier via a serial EEPROM. The controller loads the data from the serial EEPROM after a reset of the primary bus. Note that the $\overline{\text{SUSPEND}}$ input gates the PCI reset from the entire core, including the serial-bus state machine (see Section 3.8.5, *Suspend Mode*, for details on using $\overline{\text{SUSPEND}}$).

The controller provides a two-line serial-bus host controller that can interface to a serial EEPROM. See Section 3.6, *Serial-Bus Interface*, for details on the two-wire serial-bus controller and applications.

## 3.5 PC Card Applications

This section describes the PC Card interfaces of the controller.

- Card insertion/removal and recognition
- Zoomed video support
- Speaker and audio applications
- LED socket activity indicators
- CardBus socket registers

### 3.5.1 PC Card Insertion/Removal and Recognition

The *PC Card Standard* (release 7.2) addresses the card-detection and recognition process through an interrogation procedure that the socket must initiate on card insertion into a cold, nonpowered socket. Through this interrogation, card voltage requirements and interface (16-bit versus CardBus) are determined.

The scheme uses the card-detect and voltage-sense signals. The configuration of these four terminals identifies the card type and voltage requirements of the PC Card interface. The encoding scheme is defined in the *PC Card Standard* (release 7.2) and in Table 3−1.

**Table 3−1. PC Card Card-Detect and Voltage-Sense Connections**

| $\overline{\text{CD2}}$//$\overline{\text{CCD2}}$ | $\overline{\text{CD1}}$//$\overline{\text{CCD1}}$ | $\overline{\text{VS2}}$//CVS2 | $\overline{\text{VS1}}$//CVS1 | KEY | INTERFACE | VOLTAGE |
|---|---|---|---|---|---|---|
| Ground | Ground | Open | Open | 5 V | 16-bit PC Card | 5 V |
| Ground | Ground | Open | Ground | 5 V | 16-bit PC Card | 5 V and 3.3 V |
| Ground | Ground | Ground | Ground | 5 V | 16-bit PC Card | 5 V, 3.3 V, and X.X V |
| Ground | Ground | Open | Ground | LV | 16-bit PC Card | 3.3 V |
| Ground | Connect to CVS1 | Open | Connect to $\overline{\text{CCD1}}$ | LV | CardBus PC Card | 3.3 V |
| Ground | Ground | Ground | Ground | LV | 16-bit PC Card | 3.3 V and X.X V |
| Connect to CVS2 | Ground | Connect to $\overline{\text{CCD2}}$ | Ground | LV | CardBus PC Card | 3.3 V and X.X V |
| Connect to CVS1 | Ground | Ground | Connect to $\overline{\text{CCD2}}$ | LV | CardBus PC Card | 3.3 V, X.X V, and Y.Y V |
| Ground | Ground | Ground | Open | LV | 16-bit PC Card | X.X V |
| Connect to CVS2 | Ground | Connect to $\overline{\text{CCD2}}$ | Open | LV | CardBus PC Card | X.X V |
| Ground | Connect to CVS2 | Connect to $\overline{\text{CCD1}}$ | Open | LV | CardBus PC Card | X.X V and Y.Y V |
| Connect to CVS1 | Ground | Open | Connect to $\overline{\text{CCD2}}$ | LV | CardBus PC Card | Y.Y V |
| Ground | Connect to CVS1 | Ground | Connect to $\overline{\text{CCD1}}$ | Reserved | | |
| Ground | Connect to CVS2 | Connect to $\overline{\text{CCD1}}$ | Ground | Reserved | | |

### 3.5.2 Parallel Power-Switch Interface (TPS2211A)

The controller provides a parallel interface for control of the PC Card power switch. The $\overline{\text{VCCD}}$ and VPPD terminals are used with the TI TPS2211A single-slot PC Card power-switch interface to provide power-switch support. Figure 3−3 illustrates a typical application, where the controller represents the PC Card controller.

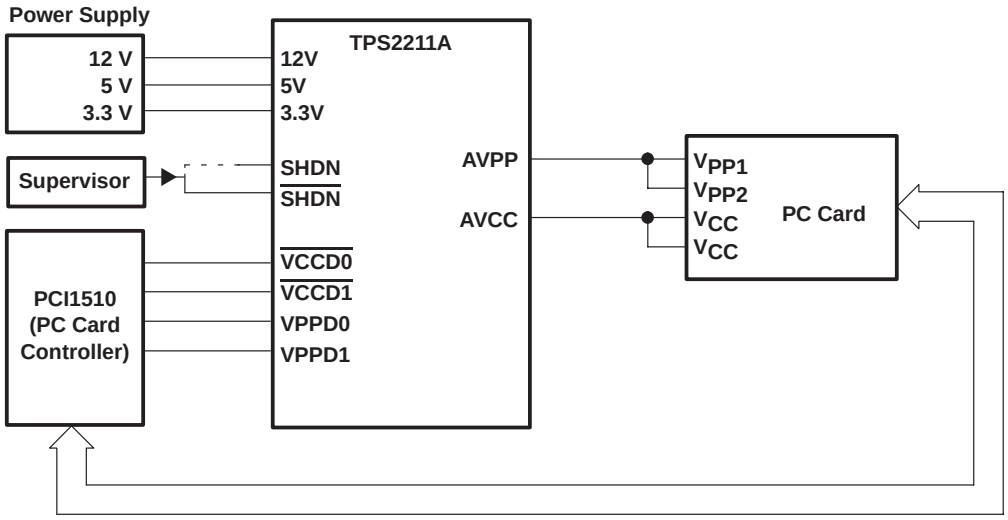



**Figure 3−3. TPS2211A Typical Application**

### 3.5.3 Zoomed Video Support

The controller allows for the implementation of zoomed video (ZV) for PC Cards. Zoomed video is supported by setting bit 6 (ZVENABLE) in the card control register (PCI offset 91h, see Section 4.32). Setting this bit puts 16-bit PC Card address lines A25−A4 of the PC Card interface in the high-impedance state. These lines can then transfer video and audio data directly to the appropriate controller. Card address lines A3−A0 can still access PC Card CIS registers for PC Card configuration. Figure 3−4 illustrates a ZV implementation.

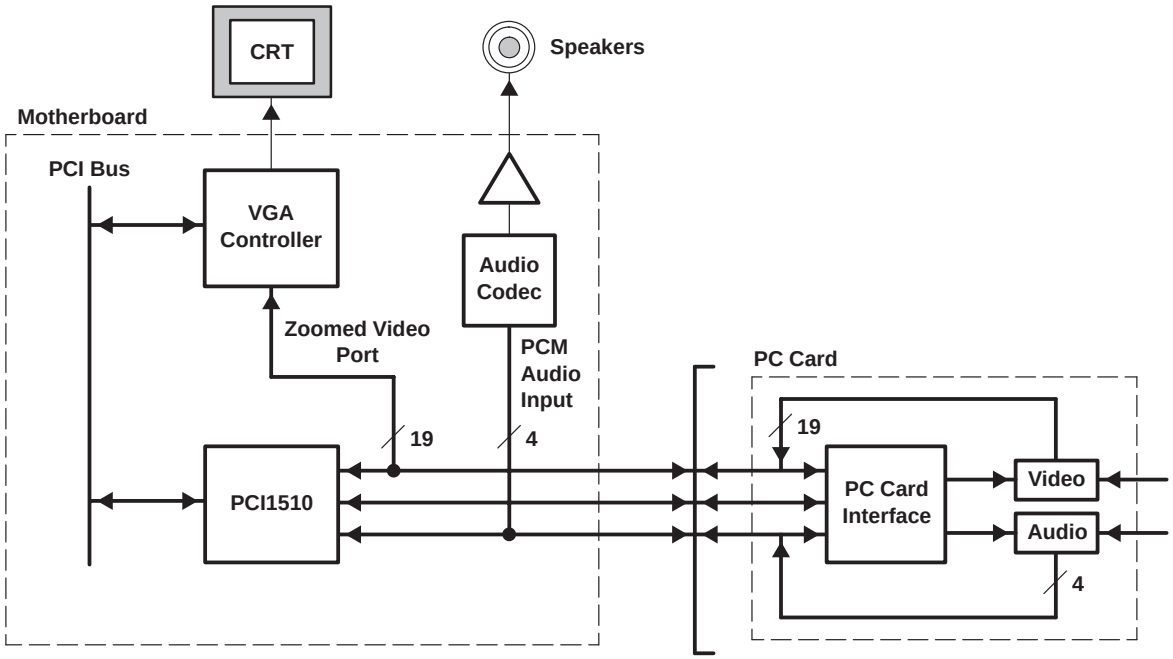



**Figure 3−4. Zoomed Video Implementation Using the PCI1510 Controller**

Not shown in Figure 3−4 is the multiplexing scheme used to route a socket ZV source to the graphics controller. The controller provides ZVSTAT and $\overline{\text{ZVSEL0}}$ signals on the multifunction terminals to switch external bus drivers. Figure 3−5 shows an implementation for switching between two ZV streams using external logic.

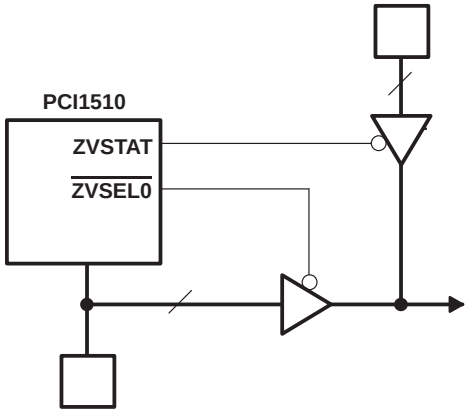



**Figure 3−5. Zoomed Video Switching Application**

Figure 3−5 illustrates an implementation using standard three-state bus drivers with active-low output enables. $\overline{\text{ZVSEL0}}$ is an active-low output indicating that the socket ZV mode is enabled.

### 3.5.4 Standardized Zoomed-Video Register Model

The standardized zoomed-video register model is defined for the purpose of standardizing the ZV port control for PC Card controllers across the industry. The following list summarizes the standardized zoomed-video register model changes to the existing PC Card register set.

- Socket present state register (CardBus socket address + 08h, see Section 6.3)
  Bit 27 (ZVSUPPORT) has been added. The platform BIOS can set this bit via the socket force event register (CardBus socket address + 0Ch, see Section 6.4) to define whether zoomed video is supported on the socket by the platform.
- Socket force event register (CardBus socket address + 0Ch, see Section 6.4)
  Bit 27 (FZVSUPPORT) has been added. The platform BIOS can use this bit to set the ZVSUPPORT bit in the socket present state register (CardBus socket address + 08h, see Section 6.3) to define whether zoomed video is supported on the socket by the platform.
- Socket control register (CardBus socket address +10h, see Section 6.5)
  Bit 11 (ZV_ACTIVITY) has been added. This bit is set when zoomed video is enabled for the PC Card socket.

  Bit 10 (STDZVREG) has been added. This bit defines whether the PC Card controller supports the standardized zoomed-video register model.

  Bit 9 (ZVEN) is provided for software to enable or disable zoomed video.

If the STDZVEN bit (bit 0) in the diagnostic register (PCI offset 93h, see Section 4.34) is 1b, then the standardized zoomed video register model is disabled. For backward compatibility, even if the STDZVEN bit is 0b (enabled), the controller allows software to access zoomed video through the legacy address in the card control register (PCI offset 91h, see Section 4.32), or through the new register model in the socket control register (CardBus socket address + 10h, see Section 6.5).

#### 3.5.4.1 Zoomed-Video Card Insertion and Configuration Procedure

1. A zoomed-video PC Card is inserted into an empty slot.
2. The card is detected and interrogated appropriately.

There are two types of PC Card controllers to consider.

- Legacy controller not using the standardized ZV register model
  Software reads bit 10 (STDZVREG) of the socket control register (CardBus socket address + 10h) to determine if the standardized zoomed-video register model is supported. If the bit returns 0b, then software must use legacy code to enable zoomed video.
- Newer controller that uses the standardized ZV register model
  Software reads bit 10 (STDZVREG) of the socket control register (CardBus socket address + 10h) to determine if the standardized zoomed-video register model is supported. If the bit returns 1b, then software can use the process/register model detailed in Table 3−2 to enable zoomed video.

**Table 3−2. Zoomed-Video Card Interrogation**

| ZVSUPPORT | ZV_ACTIVITY | ACTION |
|---|---|---|
| 1 | 0 | Set ZVEN to enable zoomed video. |
| 1 | 1 | Display a user message such as, *The zoomed video protocol required by this PC Card application is already in use by another card.* |
| 0 | X | Display a user message such as, *This platform does not support the zoomed-video protocol required by this PC Card application.* |

### 3.5.5 Internal Ring Oscillator

The internal ring oscillator provides an internal clock source for the controller so that neither the PCI clock nor an external clock is required in order for the controller to power down a socket or interrogate a PC Card. This internal oscillator can be enabled by setting bit 27 (P2CCLK) of the system control register (PCI offset 80h, see Section 4.29) to 1b. This function is enabled by default.

### 3.5.6 Integrated Pullup Resistors

The *PC Card Standard* (release 7.2) requires pullup resistors on various terminals to support both CardBus and 16-bit card configurations. Unlike the PCI12XX, PCI1450, and PCI4450 controllers which required external pullup resistors, the PCI1510 controller has integrated all of these pullup resistors. The I/O buffer on the BVD1($\overline{\text{STSCHG}}$)//CSTSCHG terminal has the capability to switch either pullup or pulldown. The pullup resistor is turned on when a 16-bit PC Card is inserted, and the pulldown resistor is turned on when a CardBus PC Card is inserted. This prevents unexpected CSTSCHG signal assertion. The integrated pullup resistors are listed in Table 3−3.

**Table 3−3. Integrated Pullup Resistors**

| SIGNAL NAME | TERMINAL NUMBER | | | SIGNAL NAME | TERMINAL NUMBER | | |
|---|---|---|---|---|---|---|---|
| | PGE | GGU | GVF | | PGE | GGU | GVF |
| A14/$\overline{\text{CPERR}}$ | 102 | D12 | F15 | $\overline{\text{CD2}}$/$\overline{\text{CCD2}}$ | 138 | B05 | C09 |
| A15/$\overline{\text{CIRDY}}$ | 110 | C10 | F13 | $\overline{\text{INPACK}}$/$\overline{\text{CREQ}}$ | 122 | B08 | B12 |
| A19/$\overline{\text{CBLOCK}}$ | 101 | A13 | E18 | READY/$\overline{\text{CINT}}$ | 131 | A06 | C10 |
| A20/$\overline{\text{CSTOP}}$ | 103 | E10 | E17 | RESET/$\overline{\text{CRST}}$ | 119 | D08 | B13 |
| A21/$\overline{\text{CDEVSEL}}$ | 106 | D11 | A16 | $\overline{\text{VS1}}$/CVS1 | 130 | B02 | B10 |
| A22/$\overline{\text{CTRDY}}$ | 108 | C12 | E14 | $\overline{\text{VS2}}$/CVS2 | 117 | A09 | F12 |
| BVD1($\overline{\text{STSCHG}}$)/CSTSCHG | 135 | C06 | A09 | $\overline{\text{WAIT}}$/$\overline{\text{CSERR}}$ | 133 | B06 | E10 |
| BVD2($\overline{\text{SPKR}}$)/CAUDIO | 134 | D06 | F10 | WP($\overline{\text{IOIS16}}$)/$\overline{\text{CCLKRUN}}$ | 136 | A05 | B09 |
| $\overline{\text{CD1}}$/$\overline{\text{CCD1}}$ | 75 | L13 | L17 | | | | |

### 3.5.7 SPKROUT and CAUDPWM Usage

SPKROUT carries the digital audio signal from the PC Card to the system. When a 16-bit PC Card is configured for I/O mode, the BVD2 terminal becomes $\overline{\text{SPKR}}$. This terminal is also used in CardBus binary audio applications, and is referred to as CAUDIO. $\overline{\text{SPKR}}$ passes a TTL-level digital audio signal to the controller. The CardBus CAUDIO signal also can pass a single-amplitude binary waveform. The binary audio signal from the PC Card socket is used in the controller to produce SPKROUT. This output is enabled by bit 1 (SPKROUTEN) in the card control register (PCI offset 91h, see Section 4.32).

Older controllers support CAUDIO in binary or PWM mode but use the same terminal (SPKROUT). Some audio chips may not support both modes on one terminal and may have a separate terminal for binary and PWM. The implementation includes a signal for PWM, CAUDPWM, which can be routed to an MFUNC terminal. Bit 2 (AUD2MUX), located in the card control register, is programmed to route a CardBus CAUDIO PWM terminal to CAUDPWM. See Section 4.30, *Multifunction Routing Register*, for details on configuring the MFUNC terminals.

Figure 3−6 provides an illustration of a sample application using SPKROUT and CAUDPWM.

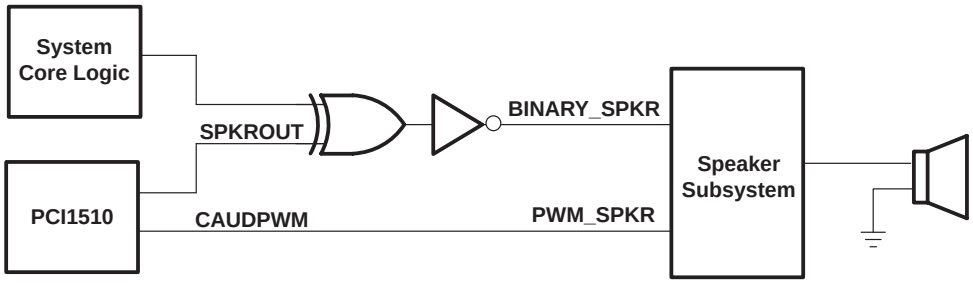



**Figure 3−6. Sample Application of SPKROUT and CAUDPWM**

### 3.5.8 LED Socket Activity Indicators

The socket activity LED is provided to indicate when a PC Card is being accessed. The LED_SKT signal can be routed to the multifunction terminals. When configured for LED output, this terminal outputs an active high signal to indicate socket activity. See Section 4.30, *Multifunction Routing Register*, for details on configuring the multifunction terminals.

The active-high LED signal is driven for 64-ms. When the LED is not being driven high, it is driven to a low state. Either of the two circuits shown in Figure 3−7 can be implemented to provide LED signaling, and the board designer must implement the circuit that best fits the application.

The LED activity signal is valid when a card is inserted, powered, and not in reset. For PC Card-16, the LED activity signal is pulsed when READY/$\overline{\text{IREQ}}$ is low. For CardBus cards, the LED activity signal is pulsed if $\overline{\text{CFRAME}}$, $\overline{\text{IRDY}}$, or $\overline{\text{CREQ}}$ are active.

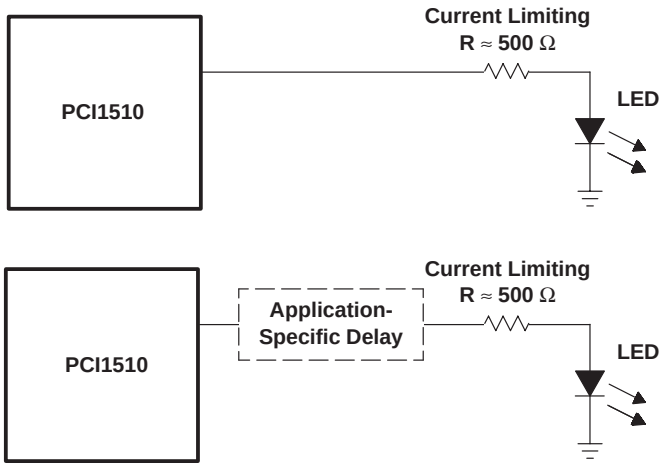



**Figure 3−7. Two Sample LED Circuits**

As indicated, the LED signal is driven for a period of 64 ms by a counter circuit. To avoid the possibility of the LED appearing to be stuck when the PCI clock is stopped, the LED signaling is cut off when the $\overline{\text{SUSPEND}}$ signal is asserted, when the PCI clock is to be stopped during the clock run protocol, or when in the D2 or D1 power state.

If any additional socket activity occurs during this counter cycle, then the counter is reset and the LED signal remains driven. If socket activity is frequent (at least once every 64 ms), then the LED signals remain driven.

### 3.5.9 CardBus Socket Registers

The controller contains all registers for compatibility with the *PC Card Standard*. These registers exist as the CardBus socket registers and are listed in Table 3−4.

**Table 3−4. CardBus Socket Registers**

| REGISTER NAME | OFFSET |
|---|---|
| Socket event | 00h |
| Socket mask | 04h |
| Socket present state | 08h |
| Socket force event | 0Ch |
| Socket control | 10h |
| Reserved | 14h−1Ch |
| Socket power management | 20h |

## 3.6 Serial-Bus Interface

The controller provides a serial-bus interface to load subsystem identification information and selected register defaults from a serial EEPROM, and to provide a PC Card power-switch interface alternative. The serial-bus interface is compatible with various $I^2C$ and SMBus components.

### 3.6.1 Serial-Bus Interface Implementation

To enable the serial interface, a pullup resistor must be implemented on the $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$ terminals and the appropriate pullup resistors must be implemented on the SDA and SCL signals, that is, the MFUNC1 and MFUNC4 terminals. When the interface is detected, bit 3 (SBDETECT) in the serial bus control and status register (PCI offset B3h, see Section 4.48) is set. The SBDETECT bit is cleared by a writeback of 1b.

The controller implements a two-pin serial interface with one clock signal (SCL) and one data signal (SDA). When pullup resistors are provided on the $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$ terminals, the SCL signal is mapped to the MFUNC4 terminal and the SDA signal is mapped to the MFUNC1 terminal. The controller drives SCL at nearly 100 kHz during data

transfers, which is the maximum specified frequency for standard-mode $I^2C$. The serial EEPROM must be located at address A0h. Figure 3−8 illustrates an example application implementing the two-wire serial bus.

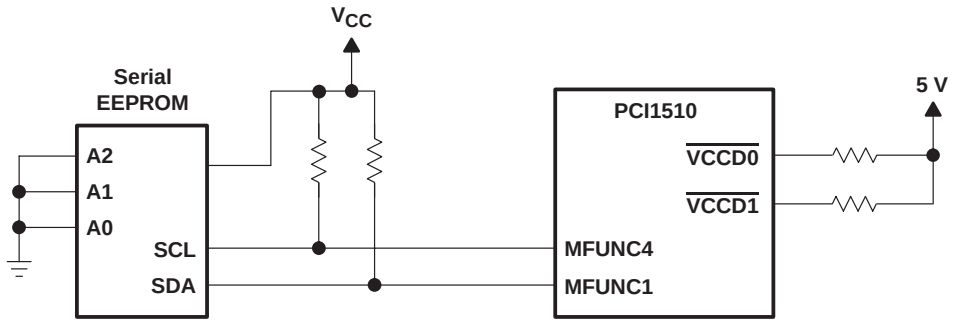



**Figure 3−8. Serial EEPROM Application**

Some serial device applications may include PC Card power switches, ZV source switches, card ejectors, or other devices that may enhance the user's PC Card experience. The serial EEPROM device and PC Card power switches are discussed in the sections that follow.

### 3.6.2 Serial-Bus Interface Protocol

The SCL and SDA signals are bidirectional, open-drain signals and require pullup resistors as shown in Figure 3−8. The controller, which supports up to 100-Kb/s data-transfer rate, is compatible with standard mode $I^2C$ using 7-bit addressing.

All data transfers are initiated by the serial bus master. The beginning of a data transfer is indicated by a start condition, which is signaled when the SDA line transitions to low state while SCL is in the high state, as illustrated in Figure 3−9. The end of a requested data transfer is indicated by a stop condition, which is signaled by a low-to-high transition of SDA while SCL is in the high state, as shown in Figure 3−9. Data on SDA must remain stable during the high state of the SCL signal, as changes on the SDA signal during the high state of SCL are interpreted as control signals, that is, a start or a stop condition.

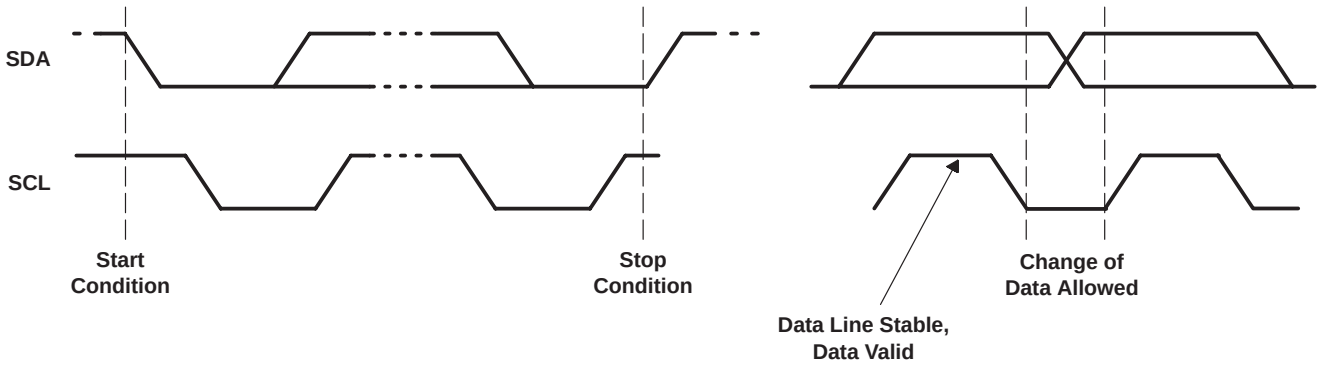



**Figure 3−9. Serial-Bus Start/Stop Conditions and Bit Transfers**

Data is transferred serially in 8-bit bytes. The number of bytes that may be transmitted during a data transfer is unlimited; however, each byte must be completed with an acknowledge bit. An acknowledge (ACK) is indicated by the receiver pulling the SDA signal low, so that it remains low during the high state of the SCL signal. Figure 3−10 illustrates the acknowledge protocol.

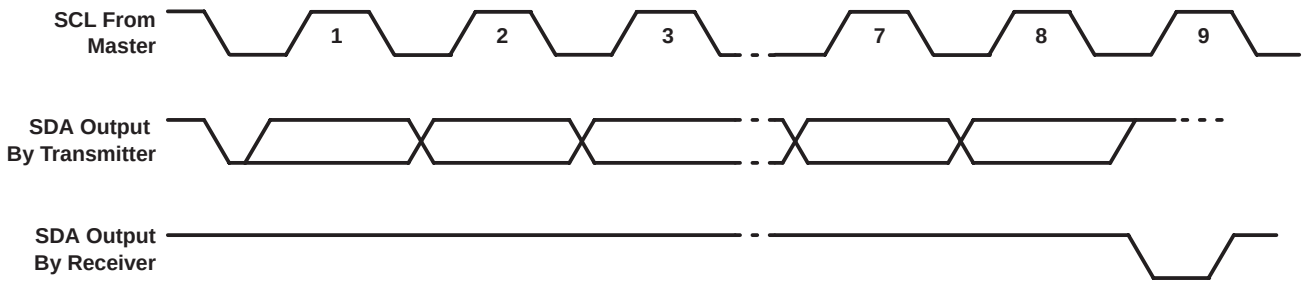



**Figure 3−10. Serial-Bus Protocol Acknowledge**

The controller is a serial bus master; all other devices connected to the serial bus external to the controller are slave devices. As the bus master, the controller drives the SCL clock at nearly 100 kHz during bus cycles and places SCL in a high-impedance state (zero frequency) during idle states.

Typically, the controller masters byte reads and byte writes under software control. Doubleword reads are performed by the serial EEPROM initialization circuitry upon a PCI reset and may not be generated under software control. See Section 3.6.3, *Serial-Bus EEPROM Application*, for details on how the controller automatically loads the subsystem identification and other register defaults through a serial-bus EEPROM.

Figure 3−11 illustrates a byte write. The controller issues a start condition and sends the 7-bit slave device address and the command bit zero. A 0b in the R/$\overline{W}$ command bit indicates that the data transfer is a write. The slave device acknowledges if it recognizes the address. If no acknowledgment is received by the controller, then an appropriate status bit is set in the serial-bus control and status register (PCI offset B3h, see Section 4.48). The word address byte is then sent by the controller, and another slave acknowledgment is expected. Then the controller delivers the data byte MSB first and expects a final acknowledgment before issuing the stop condition.

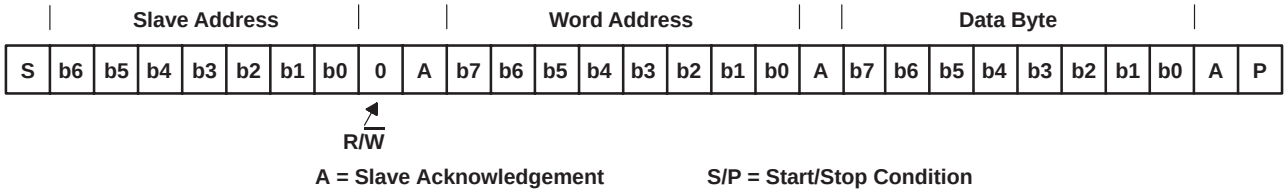



**Figure 3−11. Serial-Bus Protocol − Byte Write**

Figure 3−12 illustrates a byte read. The read protocol is very similar to the write protocol, except the R/$\overline{W}$ command bit must be set to 1b to indicate a read-data transfer. In addition, the master must acknowledge reception of the read bytes from the slave transmitter. The slave transmitter drives the SDA signal during read data transfers. The SCL signal remains driven by the master.

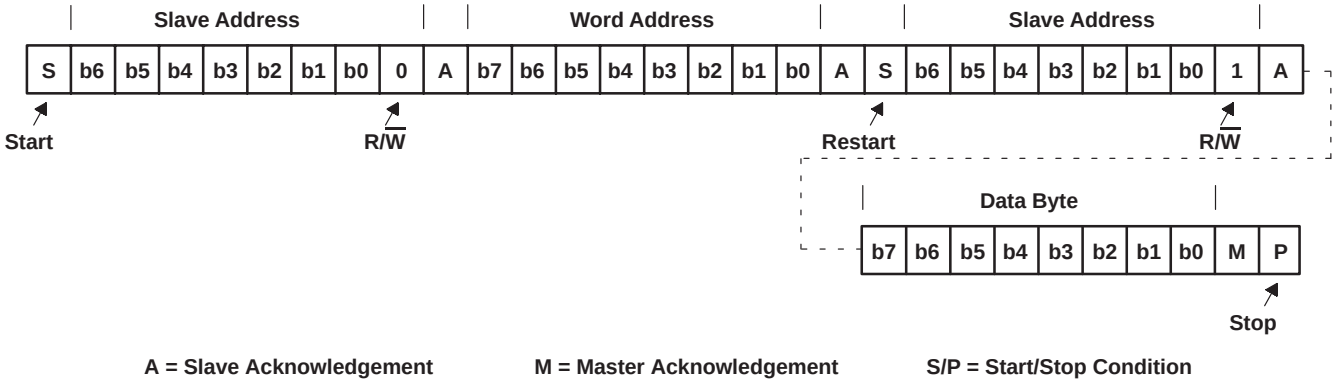



**Figure 3−12. Serial-Bus Protocol − Byte Read**

Figure 3−13 illustrates EEPROM interface doubleword data collection protocol.

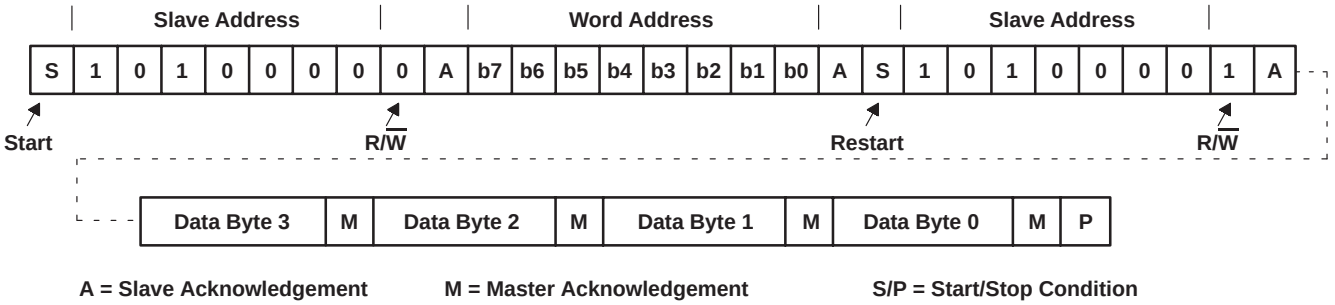



**Figure 3−13. EEPROM Interface Doubleword Data Collection**

### 3.6.3 Serial-Bus EEPROM Application

When the PCI bus is reset and the serial-bus interface is detected, the controller attempts to read the subsystem identification and other register defaults from a serial EEPROM. The registers and corresponding bits that can be loaded with defaults through the EEPROM are provided in Table 3−5.

**Table 3−5. Register- and Bit-Loading Map**

| EEPROM OFFSET | REGISTER OFFSET | REGISTER BITS LOADED FROM EEPROM |
|---|---|---|
| 00h | Flag | 01h: Load / FFh: do not load |
| 01h | PCI 04h | Command register, bit 8, 6−5, 2−0<br>Note: bits loaded per following:<br>bit 8 ← bit 7<br>bit 6 ← bit 6<br>bit 5 ← bit 5<br>bit 2 ← bit 2<br>bit 1 ← bit 1<br>bit 0 ← bit 0 |
| 02h | PCI 40h | Subsystem vendor ID bits 7−0 ← bits 7−0 |
| 03h | PCI 40h | Subsystem vendor ID bits 15−8 ← bits 7−0 |
| 04h | PCI 42h | Subsystem ID bits 7−0 ← bits 7−0 |
| 05h | PCI 42h | Subsystem ID bits 15−8 ← bits 7−0 |
| 06h | PCI 44h | PC Card 16-bit I/F LBAR bits 7−1 ← bits 7−1 |
| 07h | PCI 44h | PC Card 16-bit I/F LBAR bits 15−8 ← bits 7−0 |
| 08h | PCI 44h | PC Card 16-bit I/F LBAR bits 23−16 ← bits 7−0 |
| 09h | PCI 44h | PC Card 16-bit I/F LBAR bits 31−24 ← bits 7−0 |
| 0Ah | PCI 80h | System control bits 7−0 ← bits 7−0 |
| 0Bh | PCI 80h | System control bits 15−8 ← bits 7−0 |
| 0Ch | PCI 80h | System control bits 23−16 ← bits 7−0 |
| 0Dh | PCI 80h | System control bits 31−24 ← bits 7−0 |
| 0Eh | PCI 8Ch | Multifunction routing bits 7−0 ← bits 7−0 |
| 0Fh | PCI 8Ch | Multifunction routing bits 15−8 ← bits 7−0 |
| 10h | PCI 8Ch | Multifunction routing bits 23−16 ← bits 7−0 |
| 11h | PCI 8Ch | Multifunction routing bits 27−24 ← bits 3−0 |
| 12h | PCI 90h | Retry status bits 7, 6 ← bits 7, 6 |
| 13h | PCI 91h | Card control bit 7 ← bit 7 |
| 14h | PCI 92h | Device control bits 6, 3−0 ← bits 6, 3−0 |
| 15h | PCI 93h | Diagnostic bits 7, 4–0 ← bits 7, 4−0 |
| 16h | PCI A2h | Power management capabilities bit 15 ← bit 7 |
| 17h | ExCA 00h | ExCA identification and revision bits 7–0 ← bits 7−0 |
| 18h | CB Socket + 0Ch | Socket force event, bit 27 ← bit 3 |

This format must be followed for the controller to load initializations from a serial EEPROM. All bit fields must be considered when programming the EEPROM.

The serial EEPROM is addressed at slave address 1010 000b by the controller. All hardware address bits for the EEPROM should be tied to the appropriate level to achieve this address. The serial EEPROM chip in the sample application circuit (Figure 3−8) assumes the 1010b high-address nibble. The lower three address bits are terminal inputs to the chip, and the sample application shows these terminal inputs tied to GND.

### 3.6.4 Accessing Serial-Bus Devices Through Software

The controller provides a programming mechanism to control serial bus devices through software. The programming is accomplished through a doubleword of PCI configuration space at offset B0h. Table 3−6 lists the registers used to program a serial-bus device through software.

**Table 3−6. PCI1510 Registers Used to Program Serial-Bus Devices**

| PCI OFFSET | REGISTER NAME | DESCRIPTION |
|---|---|---|
| B0h | Serial-bus data | Contains the data byte to send on write commands or the received data byte on read commands. |
| B1h | Serial-bus index | The content of this register is sent as the word address on byte writes or reads. This register is not used in the quick command protocol. |
| B2h | Serial-bus slave address | Write transactions to this register initiate a serial-bus transaction. The slave device address and the R/$\overline{W}$ command selector are programmed through this register. |
| B3h | Serial-bus control and status | Read data valid, general busy, and general error status are communicated through this register. In addition, the protocol-select bit is programmed through this register. |

## 3.7 Programmable Interrupt Subsystem

Interrupts provide a way for I/O devices to let the microprocessor know that they require servicing. The dynamic nature of PC Cards and the abundance of PC Card I/O applications require substantial interrupt support from the controller. The controller provides several interrupt signaling schemes to accommodate the needs of a variety of platforms. The different mechanisms for dealing with interrupts in this device are based on various specifications and industry standards. The ExCA register set provides interrupt control for some 16-bit PC Card functions, and the CardBus socket register set provides interrupt control for the CardBus PC Card functions. The controller is, therefore, backward compatible with existing interrupt control register definitions, and new registers have been defined where required.

The controller detects PC Card interrupts and events at the PC Card interface and notifies the host controller using one of several interrupt signaling protocols. To simplify the discussion of interrupts in the controller, PC Card interrupts are classified either as card status change (CSC) or as functional interrupts.

The method by which any type of interrupt is communicated to the host interrupt controller varies from system to system. The controller offers system designers the choice of using parallel PCI interrupt signaling, parallel ISA-type IRQ interrupt signaling, or the IRQSER serialized ISA and/or PCI interrupt protocol. It is possible to use the parallel PCI interrupts in combination with either parallel IRQs or serialized IRQs, as detailed in the sections that follow. All interrupt signaling is provided through the seven multifunction terminals, MFUNC0−MFUNC6.

### 3.7.1 PC Card Functional and Card Status Change Interrupts

PC Card functional interrupts are defined as requests from a PC Card application for interrupt service and are indicated by asserting specially-defined signals on the PC Card interface. Functional interrupts are generated by 16-bit I/O PC Cards and by CardBus PC Cards.

Card status change (CSC)-type interrupts are defined as events at the PC Card interface that are detected by the controller and may warrant notification of host card and socket services software for service. CSC events include both card insertion and removal from the PC Card socket, as well as transitions of certain PC Card signals.

Table 3−7 summarizes the sources of PC Card interrupts and the type of card associated with them. CSC and functional interrupt sources are dependent on the type of card inserted in the PC Card socket. The three types of cards that can be inserted into any PC Card socket are:

- 16-bit memory card
- 16-bit I/O card
- CardBus cards

**Table 3−7. Interrupt Mask and Flag Registers**

| CARD TYPE | EVENT | MASK | FLAG |
|---|---|---|---|
| 16-bit memory | Battery conditions (BVD1, BVD2) | ExCA offset 05h/45h/805h bits 1 and 0 | ExCA offset 04h/44h/804h bits 1 and 0 |
| | Wait states (READY) | ExCA offset 05h/45h/805h bit 2 | ExCA offset 04h/44h/804h bit 2 |
| 16-bit I/O | Change in card status ($\overline{\text{STSCHG}}$) | ExCA offset 05h/45h/805h bit 0 | ExCA offset 04h/44h/804h bit 0 |
| | Interrupt request ($\overline{\text{IREQ}}$) | Always enabled | PCI configuration offset 91h bit 0 |
| All 16-bit PC Cards | Power cycle complete | ExCA offset 05h/45h/805h bit 3 | ExCA offset 04h/44h/804h bit 3 |
| CardBus | Change in card status (CSTSCHG) | Socket mask bit 0 | Socket event bit 0 |
| | Interrupt request ($\overline{\text{CINT}}$) | Always enabled | PCI configuration offset 91h bit 0 |
| | Power cycle complete | Socket mask bit 3 | Socket event bit 3 |
| | Card insertion or removal | Socket mask bits 2 and 1 | Socket event bits 2 and 1 |

Functional interrupt events are valid only for 16-bit I/O and CardBus cards; that is, the functional interrupts are not valid for 16-bit memory cards. Furthermore, card insertion and removal-type CSC interrupts are independent of the card type.

**Table 3−8. PC Card Interrupt Events and Description**

| CARD TYPE | EVENT | TYPE | SIGNAL | DESCRIPTION |
|---|---|---|---|---|
| 16-bit memory | Battery conditions (BVD1, BVD2) | CSC | BVD1($\overline{\text{STSCHG}}$)//CSTSCHG | A transition on BVD1 indicates a change in the PC Card battery conditions. |
| | | | BVD2($\overline{\text{SPKR}}$)//CAUDIO | A transition on BVD2 indicates a change in the PC Card battery conditions. |
| | Wait states (READY) | CSC | READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$ | A transition on READY indicates a change in the ability of the memory PC Card to accept or provide data. |
| 16-bit I/O | Change in card status ($\overline{\text{STSCHG}}$) | CSC | BVD1($\overline{\text{STSCHG}}$)//CSTSCHG | The assertion of $\overline{\text{STSCHG}}$ indicates a status change on the PC Card. |
| | Interrupt request ($\overline{\text{IREQ}}$) | Functional | READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$ | The assertion of $\overline{\text{IREQ}}$ indicates an interrupt request from the PC Card. |
| CardBus | Change in card status (CSTSCHG) | CSC | BVD1($\overline{\text{STSCHG}}$)//CSTSCHG | The assertion of CSTSCHG indicates a status change on the PC Card. |
| | Interrupt request ($\overline{\text{CINT}}$) | Functional | READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$ | The assertion of $\overline{\text{CINT}}$ indicates an interrupt request from the PC Card. |
| All PC Cards | Card insertion or removal | CSC | $\overline{\text{CD1}}$//$\overline{\text{CCD1}}$, $\overline{\text{CD2}}$//$\overline{\text{CCD2}}$ | A transition on either $\overline{\text{CD1}}$//$\overline{\text{CCD1}}$ or $\overline{\text{CD2}}$//$\overline{\text{CCD2}}$ indicates an insertion or removal of a 16-bit or CardBus PC Card. |
| | Power cycle complete | CSC | N/A | An interrupt is generated when a PC Card power-up cycle has completed. |

The naming convention for PC Card signals describes the function for 16-bit memory, I/O cards, and CardBus. For example, READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$ includes READY for 16-bit memory cards, $\overline{\text{IREQ}}$ for 16-bit I/O cards, and $\overline{\text{CINT}}$ for CardBus cards. The 16-bit memory card signal name is first, with the I/O card signal name second, enclosed in parentheses. The CardBus signal name follows after a double slash (//).

The *PC Card Standard* describes the power-up sequence that must be followed by the controller when an insertion event occurs and the host requests that the socket $V_{CC}$ and $V_{PP}$ be powered. Upon completion of this power-up sequence, the interrupt scheme can be used to notify the host system (see Table 3−8), denoted by the power cycle complete event. This interrupt source is considered an internal event, because it depends on the completion of applying power to the socket rather than on a signal change at the PC Card interface.

### 3.7.2 Interrupt Masks and Flags

Host software may individually mask (or disable) most of the potential interrupt sources listed in Table 3−8 by setting the appropriate bits in the controller. By individually masking the interrupt sources listed, software can control those events that cause an interrupt. Host software has some control over the system interrupt the controller asserts by programming the appropriate routing registers. The controller allows host software to route PC Card CSC and PC Card functional interrupts to separate system interrupts. Interrupt routing somewhat specific to the interrupt signaling method used is discussed in more detail in the following sections.

When an interrupt is signaled by the controller, the interrupt service routine must determine which of the events listed in Table 3−7 caused the interrupt. Internal registers in the controller provide flags that report the source of an interrupt. By reading these status bits, the interrupt service routine can determine the action to be taken.

Table 3−7 details the registers and bits associated with masking and reporting potential interrupts. All interrupts can be masked except the functional PC Card interrupts, and an interrupt status flag is available for all types of interrupts.

Notice that there is not a mask bit to stop the controller from passing PC Card functional interrupts through to the appropriate interrupt scheme. These interrupts are not valid until the card is properly powered, and there should never be a card interrupt that does not require service after proper initialization.

Table 3−7 lists the various methods of clearing the interrupt flag bits. The flag bits in the ExCA registers (16-bit PC Card-related interrupt flags) can be cleared using two different methods. One method is an explicit write of 1b to the flag bit to clear and the other is by reading the flag bit register. The selection of flag bit clearing methods is made by bit 2 (IFCMODE) in the ExCA global control register (ExCA offset 1Eh/5Eh/81Eh, see Section 5.20), and defaults to the flag-cleared-on-read method.

The CardBus-related interrupt flags can be cleared by an explicit write of 1b to the interrupt flag in the socket event register (see Section 6.1). Although some of the functionality is shared between the CardBus registers and the ExCA registers, software should not program the chip through both register sets when a CardBus card is functioning.

### 3.7.3 Using Parallel IRQ Interrupts

The seven multifunction terminals, MFUNC6−MFUNC0, implemented in the controller can be routed to obtain a subset of the ISA IRQs. The IRQ choices provide ultimate flexibility in PC Card host interruptions. To use the parallel ISA-type IRQ interrupt signaling, software must program the device control register (PCI offset 92h, see Section 4.33), to select the parallel IRQ signaling scheme. See Section 4.30, *Multifunction Routing Register*, for details on configuring the multifunction terminals.

A system using parallel IRQs requires (at a minimum) one PCI terminal, $\overline{\text{INTA}}$, to signal CSC events. This requirement is dictated by certain card and socket-services software. The $\overline{\text{INTA}}$ requirement calls for routing the MFUNC0 terminal for $\overline{\text{INTA}}$ signaling. This leaves (at a maximum) six different IRQs to support legacy 16-bit PC Card functions.

As an example, suppose the six IRQs used by legacy PC Card applications are IRQ3, IRQ4, IRQ5, IRQ10, IRQ11, and IRQ15. The multifunction routing register must be programmed to a value of 0FBA 5432h. This value routes the MFUNC0 terminal to $\overline{\text{INTA}}$ signaling and routes the remaining terminals as illustrated in Figure 3−14. Not shown is that $\overline{\text{INTA}}$ must also be routed to the programmable interrupt controller (PIC), or to some circuitry that provides parallel PCI interrupts to the host.

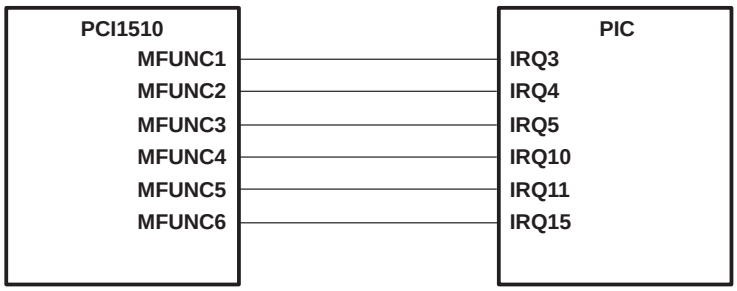



**Figure 3−14. IRQ Implementation**

Power-on software is responsible for programming the multifunction routing register to reflect the IRQ configuration of a system implementing the controller. See Section 4.30, *Multifunction Routing Register*, for details on configuring the multifunction terminals.

The parallel ISA-type IRQ signaling from the MFUNC6−MFUNC0 terminals is compatible with the input signal requirements of the 8259 PIC. The parallel IRQ option is provided for system designs that require legacy ISA IRQs. Design constraints may demand more MFUNC6−MFUNC0 IRQ terminals than the controller makes available.

### 3.7.4 Using Parallel PCI Interrupts

Parallel PCI interrupts are available when exclusively in parallel PCI interrupt/parallel ISA IRQ signaling mode, and when only IRQs are serialized with the IRQSER protocol. Socket functional interrupts can be routed to $\overline{\text{INTA}}$.

### 3.7.5 Using Serialized IRQSER Interrupts

The serialized interrupt protocol implemented in the controller uses a single terminal to communicate all interrupt status information to the host controller. The protocol defines a serial packet consisting of a start cycle, multiple interrupt indication cycles, and a stop cycle. All data in the packet is synchronous with the PCI clock. The packet data describes 16 parallel ISA IRQ signals and the optional 4 PCI interrupts $\overline{\text{INTA}}$, $\overline{\text{INTB}}$, $\overline{\text{INTC}}$, and $\overline{\text{INTD}}$. For details on the IRQSER protocol, refer to the document *Serialized IRQ Support for PCI Systems*.

### 3.7.6 SMI Support in the PCI1510 Controller

The controller provides a mechanism for interrupting the system when power changes have been made to the PC Card socket interfaces. The interrupt mechanism is designed to fit into a system maintenance interrupt (SMI) scheme. SMI interrupts are generated by the controller, when enabled, after a write cycle to either the socket control register (CB offset 10h, see Section 6.5) of the CardBus register set, or the ExCA power control register (ExCA offset 02h/42h/802h, see Section 5.3) causes a power cycle change sequence to be sent on the power switch interface.

The SMI control is programmed through three bits in the system control register (PCI offset 80h, see Section 4.29). These bits are SMIROUTE (bit 26), SMISTATUS (bit 25), and SMIENB (bit 24). Table 3−9 describes the SMI control bits function.

**Table 3−9. SMI Control**

| BIT NAME | FUNCTION |
|---|---|
| SMIROUTE | This shared bit controls whether the SMI interrupts are sent as a CSC interrupt or as IRQ2. |
| SMISTAT | This socket dependent bit is set when an SMI interrupt is pending. This status flag is cleared by writing back 1b. |
| SMIENB | When set, SMI interrupt generation is enabled. |

If CSC SMI interrupts are selected, then the SMI interrupt is sent as the CSC on a per-socket basis. The CSC interrupt can be either level or edge mode, depending upon the CSCMODE bit in the ExCA global control register (ExCA offset 1Eh/5Eh/81Eh, see Section 5.20).

If IRQ2 is selected by SMIROUTE, then the IRQSER signaling protocol supports SMI signaling in the IRQ2 IRQ/Data slot. In a parallel ISA IRQ system, the support for an active low IRQ2 is provided only if IRQ2 is routed to either MFUNC3 or MFUNC6 through the multifunction routing register (PCI offset 8Ch, see Section 4.30).

## 3.8 Power Management Overview

In addition to the low-power CMOS technology process used for the controller, various features are designed into the device to allow implementation of popular power-saving techniques. These features and techniques are discussed in this section.

### 3.8.1 Integrated Low-Dropout Voltage Regulator (LDO-VR)

The controller requires 2.5-V core voltage. The core power can be supplied by the controller itself using the internal LDO-VR. The core power can alternatively be supplied by an external power supply through the VR_PORT terminal. Table 3−10 lists the requirements for both the internal core power supply and the external core power supply.

**Table 3−10. Requirements for Internal/External 2.5-V Core Power Supply**

| SUPPLY | $V_{CC}$ | $\overline{VR_EN}$ | VR_PORT | NOTE |
|---|---|---|---|---|
| Internal | 3.3 V | GND | 2.5-V output | Internal 2.5-V LDO-VR is enabled. A 1.0-µF bypass capacitor is required on the VR_PORT terminal for decoupling. This output is not for external use. |
| External | 3.3 V | $V_{CC}$ | 2.5-V input | Internal 2.5-V LDO-VR is disabled. An external 2.5-V power supply, of minimum 50-mA capacity, is required. A 0.1-µF bypass capacitor on the VR_PORT terminal is required. |

### 3.8.2 Clock Run Protocol

The PCI $\overline{CLKRUN}$ feature is the primary method of power management on the PCI interface of the controller. $\overline{CLKRUN}$ signaling is provided through the MFUNC6 terminal. Since some chip sets do not implement $\overline{CLKRUN}$, this is not always available to the system designer, and alternate power-saving features are provided. For details on the $\overline{CLKRUN}$ protocol see the *PCI Mobile Design Guide*.

The controller does not permit the central resource to stop the PCI clock under any of the following conditions:

- Bit 1 (KEEPCLK) in the system control register (PCI offset 80h, see Section 4.29) is set.
- The 16-bit PC Card- resource manager is busy.
- The CardBus master state machine is busy. A cycle may be in progress on CardBus.
- The master is busy. There may be posted data from CardBus to PCI in the controller.
- Interrupts are pending.
- The CardBus CCLK for either socket has not been stopped by the $\overline{CCLKRUN}$ manager.

The controller restarts the PCI clock using the $\overline{CLKRUN}$ protocol under any of the following conditions:

- A 16-bit PC Card $\overline{IREQ}$ or a CardBus $\overline{CINT}$ has been asserted by either card.
- A CardBus CBWAKE (CSTSCHG) or 16-bit PC Card $\overline{STSCHG}$/$\overline{RI}$ event occurs in either socket.
- A CardBus attempts to start the CCLK using $\overline{CCLKRUN}$.
- A CardBus card arbitrates for the CardBus bus using $\overline{CREQ}$.

### 3.8.3 CardBus PC Card Power Management

The controller implements its own card power-management engine that can turn off the CCLK to a socket when there is no activity to the CardBus PC Card. The PCI clock-run protocol is followed on the CardBus $\overline{CCLKRUN}$ interface to control this clock management.

### 3.8.4 16-Bit PC Card Power Management

The COE bit (bit 7) of the ExCA power control register (ExCA offset 02h/42h/802h, see Section 5.3) and PWRDWN bit (bit 0) of the ExCA global control register (ExCA offset 1Eh/5Eh/81Eh, see Section 5.20) bits are provided for 16-bit PC Card power management. The COE bit places the card interface in a high-impedance state to save power. The power savings when using this feature are minimal. The COE bit resets the PC Card when used, and the PWRDWN bit does not. Furthermore, the PWRDWN bit is an automatic COE, that is, the PWRDWN performs the COE function when there is no card activity.

> **NOTE:** The 16-bit PC Card must implement the proper pullup resistors for the COE and PWRDWN modes.

### 3.8.5 Suspend Mode

The $\overline{\text{SUSPEND}}$ signal, provided for backward compatibility, gates the $\overline{\text{PRST}}$ (PCI reset) signal and the $\overline{\text{GRST}}$ (global reset) signal from the controller. Besides gating $\overline{\text{PRST}}$ and $\overline{\text{GRST}}$, $\overline{\text{SUSPEND}}$ also gates PCLK inside the controller in order to minimize power consumption.

It should also be noted that asynchronous signals, such as card status change interrupts and $\overline{\text{RI_OUT}}$, can be passed to the host system without a PCI clock. However, if card status change interrupts are routed over the serial interrupt stream, then the PCI clock must be restarted in order to pass the interrupt, because neither the internal oscillator nor an external clock is routed to the serial-interrupt state machine. Figure 3−15 is a signal diagram of the suspend function.

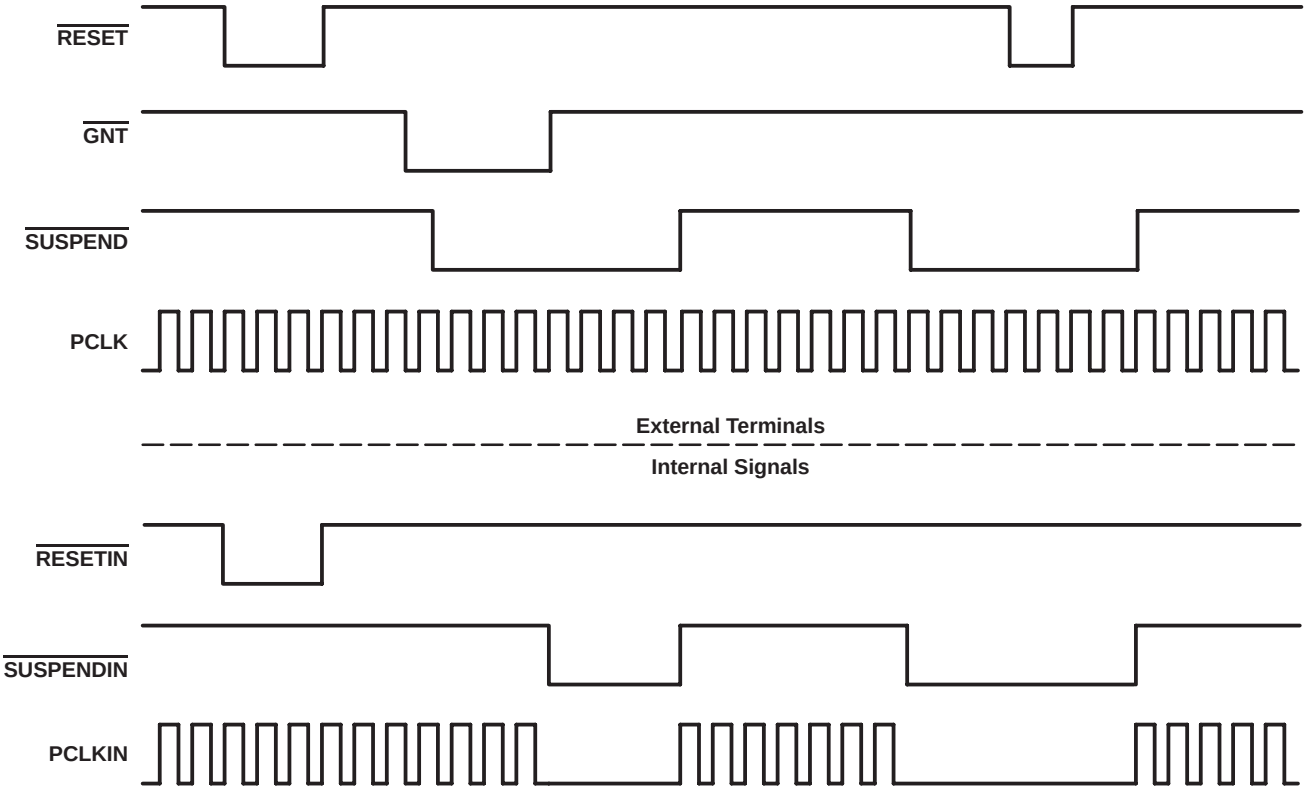



**Figure 3−15. Signal Diagram of Suspend Function**

### 3.8.6 Requirements for Suspend Mode

The suspend mode prevents the clearing of all register contents on the assertion of reset ($\overline{\text{PRST}}$ or $\overline{\text{GRST}}$) which would require the reconfiguration of the controller by software. Asserting the $\overline{\text{SUSPEND}}$ signal places the PCI outputs of the controller in a high-impedance state and gates the PCLK signal internally to the controller unless a PCI transaction is currently in process ($\overline{\text{GNT}}$ is asserted). It is important that the PCI bus not be parked on the controller when $\overline{\text{SUSPEND}}$ is asserted, because the outputs are in a high-impedance state.

The GPIOs, MFUNC signals, and $\overline{\text{RI_OUT}}$ signal are all active during $\overline{\text{SUSPEND}}$, unless they are disabled in the appropriate registers.

### 3.8.7 Ring Indicate

The RI_OUT output is an important feature in power management, allowing a system to go into a suspended mode and wake up on modem rings and other card events. TI-designed flexibility permits this signal to fit wide platform requirements. RI_OUT on the controller can be asserted under any of the following conditions:

- A 16-bit PC Card modem in a powered socket asserts RI to indicate to the system the presence of an incoming call.
- A powered down CardBus card asserts CSTSCHG (CBWAKE) requesting system and interface wake up.
- A powered CardBus card asserts CSTSCHG from the insertion/removal of cards or change in battery voltage levels.

Figure 3−16 shows various enable bits for the RI_OUT function; however, it does not show the masking of CSC events. See Table 3−7 for a detailed description of CSC interrupt masks and flags.

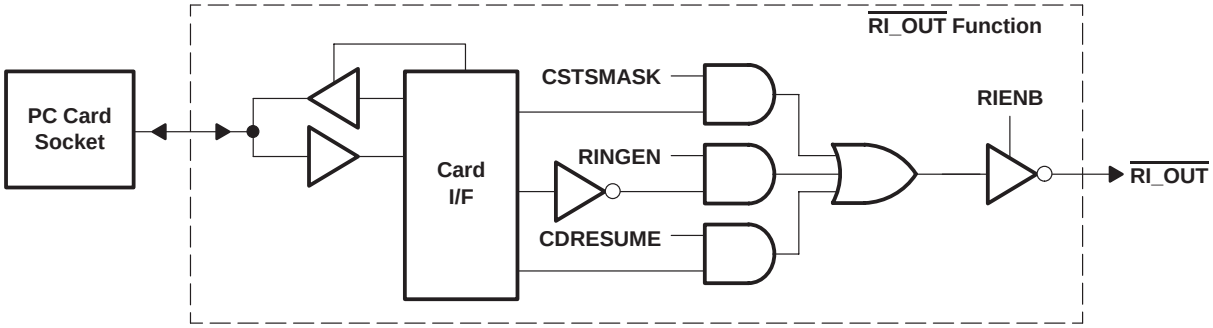



**Figure 3−16. RI_OUT Functional Diagram**

RI from the 16-bit PC Card interface is masked by bit 7 (RINGEN) in the ExCA interrupt and general control register (ExCA offset 03h/43h/803h, see Section 5.4). This is only applicable when a 16-bit card is powered in the socket.

The CBWAKE signaling to RI_OUT is enabled through the same mask as the CSC event for CSTSCHG. The mask bit (bit 0, CSTSMASK) is programmed through the socket mask register (CB offset 04h, see Section 6.2) in the CardBus socket registers.

RI_OUT can be routed through any of three different pins, RI_OUT/PME, MFUNC2, or MFUNC4. The RI_OUT function is enabled by setting RIENB in the card control register (PCI offset 91h, see Section 4.32). The PME function is enabled by setting PMEEN in the power management control/status register (PCI offset A4h, see Section 4.38). When RIMUX in the system control register (PCI offset 80h, see Section 4.29) is set to 0b, both the RI_OUT function and the PME function are routed to the RI_OUT/PME terminal. If both functions are enabled and RIMUX is set to 0b, the RI_OUT/PME terminal becomes RI_OUT only and PME assertions will never be seen. Therefore, in a system using both the RI_OUT function and the PME function, RIMUX must be set to 1b and RI_OUT must be routed to either MFUNC2 or MFUNC4.

### 3.8.8 PCI Power Management

The *PCI Bus Power Management Interface Specification for PCI to CardBus Bridges* establishes the infrastructure required to let the operating system control the power of PCI functions. This is done by defining a standard PCI interface and operations to manage the power of PCI functions on the bus. The PCI bus and the PCI functions can be assigned one of seven power-management states, resulting in varying levels of power savings.

The seven power-management states of PCI functions are:

- D0-uninitialized − Before device configuration, device not fully functional
- D0-active − Fully functional state
- D1 − Low-power state
- D2 − Low-power state

- $D3_{hot}$ − Low-power state. Transition state before $D3_{cold}$
- $D3_{cold}$ − $\overline{PME}$ signal-generation capable. Main power is removed and VAUX is available.
- $D3_{off}$ − No power and completely non-functional

**NOTE:**

In the D0-uninitialized state, the controller does not generate $\overline{PME}$ and/or interrupts. When the IO_EN and MEM_EN bits (bits 0 and 1) of the command register (PCI offset 04h, see Section 4.4) are both set, the controller switches the state to D0-active. Transition from $D3_{cold}$ to the D0-uninitialized state happens at the deassertion of $\overline{PRST}$. The assertion of $\overline{GRST}$ forces the controller to the D0-uninitialized state immediately.

The PWR_STATE bits (bits 0−1) of the power-management control/status register (PCI offset A4h, see Section 4.38) only code for four power states, D0, D1, D2, and $D3_{hot}$. The differences between the three D3 states is invisible to the software because the controller is not accessible in the $D3_{cold}$ or $D3_{off}$ state.

Similarly, bus power states of the PCI bus are B0−B3. The bus power states B0−B3 are derived from the device power state of the originating bridge device.

For the operating system (OS) to manage the device power states on the PCI bus, the PCI function should support four power-management operations. These operations are:

- Capabilities reporting
- Power status reporting
- Setting the power state
- System wake up

The OS identifies the capabilities of the PCI function by traversing the new capabilities list. The presence of capabilities in addition to the standard PCI capabilities is indicated by a 1b in bit 4 (CAPLIST) of the status register (PCI offset 06h, see Section 4.5).

The capabilities pointer provides access to the first item in the linked list of capabilities. For the controller, a CardBus bridge with PCI configuration space header type 2, the capabilities pointer is mapped to an offset of 14h. The first byte of each capability register block is required to be a unique ID of that capability. PCI power management has been assigned an ID of 01h. The next byte is a pointer to the next pointer item in the list of capabilities. If there are no more items in the list, then the next item pointer must be set to 0b. The registers following the next item pointer are specific to the capability of the function. The PCI power-management capability implements the register block outlined in Table 3−11.

**Table 3−11. Power-Management Registers**

| REGISTER NAME | | | | OFFSET |
|---|---|---|---|---|
| Power-management capabilities | | Next-item pointer | Capability ID | A0h |
| Power-management data | Power-management control/status bridge support extensions | Power-management control/status | | A4h |

The power management capabilities register (PCI offset A2h, see Section 4.37) provides information on the capabilities of the function related to power management. The power-management control/status register (PCI offset A4h, see Section 4.38) enables control of power-management states and enables/monitors power-management events. The data register is an optional register that can provide dynamic data.

For more information on PCI power management, see the *PCI Bus Power Management Interface Specification for PCI to CardBus Bridges*.

### 3.8.9 CardBus Bridge Power Management

The *PCI Bus Power Management Interface Specification for PCI to CardBus Bridges* was approved by PCMCIA in December of 1997. This specification follows the device and bus state definitions provided in the *PCI Bus Power Management Interface Specification* published by the PCI Special Interest Group (SIG). The main issue addressed in the *PCI Bus Power Management Interface Specification for PCI to CardBus Bridges* is wake-up from $D3_{hot}$ or $D3_{cold}$ without losing wake-up context (also called $\overline{PME}$ context).

The specific issues addressed by the *PCI Bus Power Management Interface Specification for PCI to CardBus Bridges* for D3 wake up are as follows:

- Preservation of device context. The specification states that a reset must occur during the transition from D3 to D0. Some method to preserve wake-up context must be implemented so that the reset does not clear the $\overline{\text{PME}}$ context registers.
- Power source in $D3_{cold}$ if wake-up support is required from this state.

The PCI1510 controller addresses these D3 wake-up issues in the following manner:

- Two resets are provided to handle preservation of $\overline{\text{PME}}$ context bits:
  - Global reset ($\overline{\text{GRST}}$) is used only on the initial boot up of the system after power up. It places the controller in its default state and requires BIOS to configure the device before becoming fully functional.
  - PCI reset ($\overline{\text{PRST}}$) has dual functionality based on whether $\overline{\text{PME}}$ is enabled or not. If $\overline{\text{PME}}$ is enabled, then $\overline{\text{PME}}$ context is preserved. If $\overline{\text{PME}}$ is not enabled, then $\overline{\text{PRST}}$ acts the same as a normal PCI reset. Please see the master list of $\overline{\text{PME}}$ context bits in Section 3.8.11.
- Power source in $D3_{cold}$ if wake-up support is required from this state. Since $V_{CC}$ is removed in $D3_{cold}$, an auxiliary power source must be supplied to the $V_{CC}$ terminals. Consult the *PCI14xx Implementation Guide for D3 Wake-Up* or the *PCI Power Management Interface Specification for PCI to CardBus Bridges* for further information.

### 3.8.10 ACPI Support

The *Advanced Configuration and Power Interface (ACPI) Specification* provides a mechanism that allows unique pieces of hardware to be described to the ACPI driver. The controller offers a generic interface that is compliant with ACPI design rules.

Two doublewords of general-purpose ACPI programming bits reside in PCI configuration space at offset A8h. The programming model is broken into status and control functions. In compliance with ACPI, the top level event status and enable bits reside in the general-purpose event status register (PCI offset A8h, see Section 4.41) and general-purpose event enable register (PCI offset AAh, see Section 4.42). The status and enable bits are implemented as defined by ACPI and illustrated in Figure 3−17.

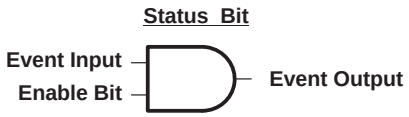



**Figure 3−17. Block Diagram of a Status/Enable Cell**

The status and enable bits generate an event that allows the ACPI driver to call a control method associated with the pending status bit. The control method can then control the hardware by manipulating the hardware control bits or by investigating child status bits and calling their respective control methods. A hierarchical implementation would be somewhat limiting, however, as upstream devices would have to remain in some level of power state to report events.

For more information of ACPI, see the *Advanced Configuration and Power Interface (ACPI) Specification.*

### 3.8.11 Master List of $\overline{\text{PME}}$ Context Bits and Global Reset-Only Bits

If the $\overline{\text{PME}}$ enable bit (bit 8) of the power-management control/status register (PCI offset A4h, see section 4.38) is asserted, then the assertion of $\overline{\text{PRST}}$ will not clear the following $\overline{\text{PME}}$ context bits. If the $\overline{\text{PME}}$ enable bit is not asserted, then the $\overline{\text{PME}}$ context bits are cleared with $\overline{\text{PRST}}$. The $\overline{\text{PME}}$ context bits are:

- Bridge control register (PCI offset 3Eh): bit 6
- System control register (PCI offset 80h): bits 10, 9, 8
- Power-management control/status register (PCI offset A4h): bits 15, 8
- ExCA power control register (ExCA offset 802h): bits 7, 5†, 4−3, 1−0 († 82365SL mode only)
- ExCA interrupt and general control register (ExCA offset 803h): bits 6−5
- ExCA card status change register (ExCA offset 804h): bits 11−8, 3−0
- ExCA card status-change-interrupt configuration register (ExCA offset 805h): bits 3−0
- CardBus socket event register (CardBus offset 00h): bits 3−0
- CardBus socket mask register (CardBus offset 04h): bits 3−0
- CardBus socket present state register (CardBus offset 08h): bits 13−7, 5−1
- CardBus socket control register (CardBus offset 10h): bits 6−4, 2−0

Global reset places all registers in their default state regardless of the state of the $\overline{\text{PME}}$ enable bit. The $\overline{\text{GRST}}$ signal is gated only by the $\overline{\text{SUSPEND}}$ signal. This means that assertion of $\overline{\text{SUSPEND}}$ blocks the $\overline{\text{GRST}}$ signal internally, thus preserving all register contents. The registers cleared only by $\overline{\text{GRST}}$ are:

- Status register (PCI offset 06h): bits 15−11, 8
- Secondary status register (PCI offset 16h): bits 15−11, 8
- Interrupt pin register (PCI offset 3Dh): bits 1,0
- Subsystem vendor ID register (PCI offset 40h): bits 15–0
- Subsystem ID register (PCI offset 42h): bits 15–0
- PC Card 16-bit legacy mode base address register (PCI offset 44h): bits 31–1
- System control register (PCI offset 80h): bits 31–29, 27–13, 11, 6−0
- Multifunction routing register (PCI offset 8Ch): bits 27−0
- Retry status register (PCI offset 90h): bits 7−5, 3, 1
- Card control register (PCI offset 91h): bits 7−5, 2−0
- Device control register (PCI offset 92h): bits 7−5, 3−0
- Diagnostic register (PCI offset 93h): bits 7−0
- Power management capabilities register (PCI offset A2h): bit 15
- General-purpose event status register (PCI offset A8h): bits 15−14
- General-purpose event enable register (PCI offset AAh): bits 15−14, 11, 8, 4−0
- General-purpose output (PCI offset AEh): bits 4−0
- Serial bus data (PCI offset B0h): bits 7−0
- Serial bus index (PCI offset B1h): bits 7−0
- Serial bus slave address register (PCI offset B2h): bits 7−0
- Serial bus control and status register (PCI offset B3h): bits 7, 5−0
- ExCA identification and revision register (ExCA offset 00h): bits 7−0
- ExCA global control register (ExCA offset 1Eh): bits 2−0
- Socket present state register (CardBus offset 08h): bit 29
- Socket power management register (CardBus offset 20h): bits 25−24

# 4 PC Card Controller Programming Model

This chapter describes the PCI1510 PCI configuration registers that make up the 256-byte PCI configuration header.

## 4.1 PCI Configuration Registers

The configuration header is compliant with the *PCI Local Bus Specification* as a CardBus bridge header and is *PC 99* compliant as well. Table 4−1 shows the PCI configuration header, which includes both the predefined portion of the configuration space and the user-definable registers.

**Table 4−1. PCI Configuration Registers**

| REGISTER NAME | | | | OFFSET |
|---|---|---|---|---|
| Device ID | | Vendor ID | | 00h |
| Status | | Command | | 04h |
| Class code | | | Revision ID | 08h |
| BIST | Header type | Latency timer | Cache line size | 0Ch |
| CardBus socket/ExCA base address | | | | 10h |
| Secondary status | | Reserved | Capability pointer | 14h |
| CardBus latency timer | Subordinate bus number | CardBus bus number | PCI bus number | 18h |
| CardBus Memory base register 0 | | | | 1Ch |
| CardBus Memory limit register 0 | | | | 20h |
| CardBus Memory base register 1 | | | | 24h |
| CardBus Memory limit register 1 | | | | 28h |
| CardBus I/O base register 0 | | | | 2Ch |
| CardBus I/O limit register 0 | | | | 30h |
| CardBus I/O base register 1 | | | | 34h |
| CardBus I/O limit register 1 | | | | 38h |
| Bridge control | | Interrupt pin | Interrupt line | 3Ch |
| Subsystem ID | | Subsystem vendor ID | | 40h |
| PC Card 16-bit I/F legacy-mode base address | | | | 44h |
| Reserved | | | | 48h−7Ch |
| System control | | | | 80h |
| Reserved | | | | 84h−88h |
| Multifunction routing | | | | 8Ch |
| Diagnostic | Device control | Card control | Retry status | 90h |
| Reserved | | | | 94h−9Ch |
| Power-management capabilities | | Next-item pointer | Capability ID | A0h |
| Power-management data | Power-management control/status bridge support extensions | Power-management control/status | | A4h |
| General-purpose event enable | | General-purpose event status | | A8h |
| General-purpose output | | General-purpose input | | ACh |
| Serial bus control/status | Serial bus slave address | Serial bus index | Serial bus data | B0h |
| Reserved | | | | B4h−FCh |

A bit description table, typically included when a register contains bits of more than one type or purpose, indicates bit field names, which appear in the signal column; a detailed field description, which appears in the function column; and field access tags, which appear in the type column of the bit description table. Table 4−2 describes the field access tags.

**Table 4−2. Bit Field Access Tag Descriptions**

| ACCESS TAG | NAME | MEANING |
|---|---|---|
| R | Read | Field may be read by software. |
| W | Write | Field may be written by software to any value. |
| S | Set | Field may be set by a write of 1b. Writes of 0b have no effect. |
| C | Clear | Field may be cleared by a write of 1b. Writes of 0b have no effect. |
| U | Update | Field may be autonomously updated by the controller. |

## 4.2 Vendor ID Register

This 16-bit register contains a value allocated by the PCI Special Interest Group (SIG) and identifies the manufacturer of the PCI device. The vendor ID assigned to TI is 104Ch.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

Register: **Vendor ID**
Offset: 00h
Type: Read-only
Default: 104Ch

## 4.3 Device ID Register

This 16-bit register contains a value assigned to the controller by TI. The device identification for the controller is AC56h.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |

Register: **Device ID**
Offset: 02h
Type: Read-only
Default: AC56h

## 4.4 Command Register

The command register provides control over the controller interface to the PCI bus. All bit functions adhere to the definitions in *PCI Local Bus Specification*. See Table 4−3 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Command**
Offset: 04h
Type: Read-only, Read/Write
Default: 0000h

**Table 4−3. Command Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15−10 | RSVD | R | Reserved. Bits 15−10 return 00 0000b when read. |
| 9 | FBB_EN | R | Fast back-to-back enable. The controller does not generate fast back-to-back transactions; therefore, bit 9 returns 0b when read. |
| 8 | SERR_EN | RW | System error ($\overline{\text{SERR}}$) enable. Bit 8 controls the enable for the $\overline{\text{SERR}}$ driver on the PCI interface. $\overline{\text{SERR}}$ can be asserted after detecting an address parity error on the PCI bus. Both bits 8 and 6 must be set for the controller to report address parity errors.<br>0 = Disable $\overline{\text{SERR}}$ output driver (default)<br>1 = Enable $\overline{\text{SERR}}$ output driver |
| 7 | STEP_EN | R | Address/data stepping control. The controller does not support address/data stepping; therefore, bit 7 is hardwired to 0b. |
| 6 | PERR_EN | RW | Parity error response enable. Bit 6 controls the controller response to parity errors through $\overline{\text{PERR}}$. Data parity errors are indicated by asserting $\overline{\text{PERR}}$, whereas address parity errors are indicated by asserting $\overline{\text{SERR}}$.<br>0 = The controller ignores detected parity error (default)<br>1 = The controller responds to detected parity errors |
| 5 | VGA_EN | RW | VGA palette snoop. Bit 5 controls how PCI devices handle accesses to video graphics array (VGA) palette registers. |
| 4 | MWI_EN | R | Memory write-and-invalidate enable. Bit 4 controls whether a PCI initiator device can generate memory write-and-Invalidate commands. The controller does not support memory write-and-invalidate commands, but uses memory write commands instead; therefore, this bit is hardwired to 0b. |
| 3 | SPECIAL | R | Special cycles. Bit 3 controls whether or not a PCI device ignores PCI special cycles. The controller does not respond to special cycle operations; therefore, this bit is hardwired to 0b. |
| 2 | MAST_EN | RW | Bus master control. Bit 2 controls whether or not the controller can act as a PCI bus initiator (master). The controller can take control of the PCI bus only when this bit is set.<br>0 = Disables the controller from generating PCI bus accesses (default)<br>1 = Enables the controller to generate PCI bus accesses |
| 1 | MEM_EN | RW | Memory space enable. Bit 1 controls whether or not the controller can claim cycles in PCI memory space.<br>0 = Disables the controller from responding to memory space accesses (default)<br>1 = Enables the controller to respond to memory space accesses |
| 0 | IO_EN | RW | I/O space control. Bit 0 controls whether or not the controller can claim cycles in PCI I/O space.<br>0 = Disables the controller from responding to I/O space accesses (default)<br>1 = Enables the controller to respond to I/O space accesses |

## 4.5 Status Register

The status register provides device information to the host system. Bits in this register can be read normally. A bit in the status register is reset when a 1b is written to that bit location; a 0b written to a bit location has no effect. All bit functions adhere to the definitions in the *PCI Local Bus Specification*. See Table 4−4 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Register: **Status**
Offset: 06h
Type: Read-only, Read/Clear
Default: 0210h

**Table 4−4. Status Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | PAR_ERR | RC | Detected parity error. Bit 15 is set when a parity error is detected (either address or data). |
| 14 | SYS_ERR | RC | Signaled system error. Bit 14 is set when $\overline{\text{SERR}}$ is enabled and the controller signals a system error to the host. |
| 13 | MABORT | RC | Received master abort. Bit 13 is set when a cycle initiated by the controller on the PCI bus is terminated by a master abort. |
| 12 | TABT_REC | RC | Received target abort. Bit 12 is set when a cycle initiated by the controller on the PCI bus is terminated by a target abort. |
| 11 | TABT_SIG | RC | Signaled target abort. Bit 11 is set by the controller when it terminates a transaction on the PCI bus with a target abort. |
| 10−9 | PCI_SPEED | R | $\overline{\text{DEVSEL}}$ timing. These bits encode the timing of $\overline{\text{DEVSEL}}$ and are hardwired 01b, indicating that the controller asserts PCI_SPEED at a medium speed on nonconfiguration cycle accesses. |
| 8 | DATAPAR | RC | Data parity error detected.<br>0 = The conditions for setting bit 8 have not been met.<br>1 = A data parity error occurred, and the following conditions were met:<br>a. $\overline{\text{PERR}}$ was asserted by any PCI device including the controller.<br>b. The controller was the bus master during the data parity error.<br>c. The parity error response bit is set in the command register (PCI offset 04h, see Section 4.4). |
| 7 | FBB_CAP | R | Fast back-to-back capable. The controller cannot accept fast back-to-back transactions; therefore, bit 7 is hardwired to 0b. |
| 6 | UDF | R | User-definable feature support. The controller does not support the user-definable features; therefore, bit 6 is hardwired to 0b. |
| 5 | 66MHZ | R | 66-MHz capable. The controller operates at a maximum PCLK frequency of 33 MHz; therefore, bit 5 is hardwired to 0b. |
| 4 | CAPLIST | R | Capabilities list. Bit 4 returns 1b when read. This bit indicates that capabilities in addition to standard PCI capabilities are implemented. The linked list of PCI power-management capabilities is implemented in this function. |
| 3−0 | RSVD | R | Reserved. Bits 3−0 return 0h when read. |

## 4.6 Revision ID Register

The revision ID register indicates the silicon revision of the controller.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Revision ID**
Offset: 08h
Type: Read-only
Default: 00h

## 4.7 PCI Class Code Register

The class code register recognizes the controller as a bridge device (06h) and a CardBus bridge device (07h), with a 00h programming interface.

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Base class | | | | | | | | Subclass | | | | | | | | Programming interface | | | | | | | |
| Default | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **PCI class code**
Offset: 09h
Type: Read-only
Default: 06 0700h

## 4.8 Cache Line Size Register

The cache line size register is programmed by host software to indicate the system cache line size.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Cache line size**
Offset: 0Ch
Type: Read/Write
Default: 00h

## 4.9 Latency Timer Register

The latency timer register specifies the latency time for the controller in units of PCI clock cycles. When the controller is a PCI bus initiator and asserts $\overline{\text{FRAME}}$, the latency timer begins counting from zero. If the latency timer expires before the transaction has terminated, then the controller terminates the transaction when its $\overline{\text{GNT}}$ is deasserted.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Latency timer**
Offset: 0Dh
Type: Read/Write
Default: 00h

## 4.10 Header Type Register

This register returns 02h when read, indicating that the configuration space adheres to the CardBus bridge PCI header. The CardBus bridge PCI header ranges from PCI register 00h to 7Fh, and 80h to FFh is user-definable extension registers.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

Register: **Header type**
Offset: 0Eh
Type: Read-only
Default: 02h

## 4.11 BIST Register

Because the controller does not support a built-in self-test (BIST), this register returns the value of 00h when read.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **BIST**
Offset: 0Fh
Type: Read-only
Default: 00h

## 4.12 CardBus Socket/ExCA Base-Address Register

The CardBus socket/ExCA base-address register is programmed with a base address referencing the CardBus socket registers and the memory-mapped ExCA register set. Bits 31−12 are read/write and allow the base address to be located anywhere in the 32-bit PCI memory address space on a 4-Kbyte boundary. Bits 11−0 are read-only, returning 000h when read. When software writes FFFF FFFFh to this register, the value read back is FFFF F000h, indicating that at least 4 Kbytes of memory address space are required. The CardBus registers start at offset 000h, and the memory-mapped ExCA registers begin at offset 800h.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **CardBus socket/ExCA base-address**
Offset: 10h
Type: Read-only, Read/Write
Default: 0000 0000h

## 4.13 Capability Pointer Register

The capability pointer register provides a pointer into the PCI configuration header where the PCI power-management register block resides. PCI header doublewords at A0h and A4h provide the power-management (PM) registers. This register returns A0h when read.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

Register: **Capability pointer**
Offset: 14h
Type: Read-only
Default: A0h

## 4.14 Secondary Status Register

The secondary status register is compatible with the PCI-to-PCI bridge secondary status register and indicates CardBus-related device information to the host system. This register is very similar to the status register (offset 06h, see Section 4.5); status bits are cleared by writing a 1b. See Table 4−5 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Secondary status**
Offset: 16h
Type: Read-only, Read/Clear
Default: 0200h

**Table 4−5. Secondary Status Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | CBPARITY | RC | Detected parity error. Bit 15 is set when a CardBus parity error is detected (either address or data). |
| 14 | CBSERR | RC | Signaled system error. Bit 14 is set when $\overline{\text{CSERR}}$ is signaled by a CardBus card. The controller does not assert $\overline{\text{CSERR}}$. |
| 13 | CBMABORT | RC | Received master abort. Bit 13 is set when a cycle initiated by the controller on the CardBus bus has been terminated by a master abort. |
| 12 | REC_CBTA | RC | Received target abort. Bit 12 is set when a cycle initiated by the controller on the CardBus bus is terminated by a target abort. |
| 11 | SIG_CBTA | RC | Signaled target abort. Bit 11 is set by the controller when it terminates a transaction on the CardBus bus with a target abort. |
| 10−9 | CB_SPEED | R | CDEVSEL timing. These bits encode the timing of $\overline{\text{CDEVSEL}}$ and are hardwired 01b, indicating that the controller asserts CB_SPEED at a medium speed. |
| 8 | CB_DPAR | RC | CardBus data parity error detected.<br>0 = The conditions for setting bit 8 have not been met.<br>1 = A data parity error occurred and the following conditions were met:<br>a. $\overline{\text{CPERR}}$ was asserted on the CardBus interface.<br>b. The controller was the bus master during the data parity error.<br>c. The parity error response bit is set in the bridge control. |
| 7 | CBFBB_CAP | R | Fast back-to-back capable. The controller cannot accept fast back-to-back transactions; therefore, bit 7 is hardwired to 0b. |
| 6 | CB_UDF | R | User-definable feature support. The controller does not support user-definable features; therefore, bit 6 is hardwired to 0b. |
| 5 | CB66MHZ | R | 66-MHz capable. The CardBus interface operates at a maximum CCLK frequency of 33 MHz; therefore, bit 5 is hardwired to 0b. |
| 4−0 | RSVD | R | Reserved. Bits 4−0 return 00000b when read. |

## 4.15 PCI Bus Number Register

This register is programmed by the host system to indicate the bus number of the PCI bus to which the controller is connected. The controller uses this register in conjunction with the CardBus bus number and subordinate bus number registers to determine when to forward PCI configuration cycles to its secondary buses.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **PCI bus number**
Offset: 18h
Type: Read/Write
Default: 00h

## 4.16 CardBus Bus Number Register

This register is programmed by the host system to indicate the bus number of the CardBus bus to which the controller is connected. The controller uses this register in conjunction with the PCI bus number and subordinate bus number registers to determine when to forward PCI configuration cycles to its secondary buses.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **CardBus bus number**
Offset: 19h
Type: Read/Write
Default: 00h

## 4.17 Subordinate Bus Number Register

This register is programmed by the host system to indicate the highest-numbered bus below the CardBus bus. The controller uses this register in conjunction with the PCI bus number and CardBus bus number registers to determine when to forward PCI configuration cycles to its secondary buses.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Subordinate bus number**
Offset: 1Ah
Type: Read/Write
Default: 00h

## 4.18 CardBus Latency Timer Register

This register is programmed by the host system to specify the latency timer for the CardBus interface in units of CCLK cycles. When the controller is a CardBus initiator and asserts $\overline{\text{CFRAME}}$, the CardBus latency timer begins counting. If the latency timer expires before the transaction has terminated, then the controller terminates the transaction at the end of the next data phase. A recommended minimum value for this register is 40h, which allows most transactions to be completed.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **CardBus latency timer**
Offset: 1Bh
Type: Read/Write
Default: 00h

## 4.19 Memory Base Registers 0, 1

The memory base registers indicate the lower address of a PCI memory address range. These registers are used by the controller to determine when to forward a memory transaction to the CardBus bus and when to forward a CardBus cycle to PCI. Bits 31−12 of these registers are read/write and allow the memory base to be located anywhere in the 32-bit PCI memory space on 4-Kbyte boundaries. Bits 11−0 are read-only and always return 000h. Write transactions to these bits have no effect. Bits 8 and 9 of the bridge control register specify whether memory windows 0 and 1 are prefetchable or nonprefetchable. The memory base register or the memory limit register must be nonzero for the controller to claim any memory transactions through CardBus memory windows (that is, these windows are not enabled by default to pass the first 4 Kbytes of memory to CardBus).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Memory base registers 0, 1**
Offset: 1Ch, 24h
Type: Read-only, Read/Write
Default: 0000 0000h

## 4.20 Memory Limit Registers 0, 1

The memory limit registers indicate the upper address of a PCI memory address range. These registers are used by the controller to determine when to forward a memory transaction to the CardBus bus and when to forward a CardBus cycle to PCI. Bits 31−12 of these registers are read/write and allow the memory base to be located anywhere in the 32-bit PCI memory space on 4-Kbyte boundaries. Bits 11−0 are read-only and always return 000h. Write transactions to these bits have no effect. Bits 8 and 9 of the bridge control register specify whether memory windows 0 and 1 are prefetchable or nonprefetchable. The memory base register or the memory limit register must be nonzero for the controller to claim any memory transactions through CardBus memory windows; that is, these windows are not enabled by default to pass the first 4 Kbytes of memory to CardBus.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Memory limit registers 0, 1**
Offset: 20h, 28h
Type: Read-only, Read/Write
Default: 0000 0000h

## 4.21 I/O Base Registers 0, 1

The I/O base registers indicate the lower address of a PCI I/O address range. These registers are used by the controller to determine when to forward an I/O transaction to the CardBus bus and when to forward a CardBus cycle to the PCI bus. The lower 16 bits of this register locate the bottom of the I/O window within a 64-Kbyte page, and the upper 16 bits (31−16) are a page register which locates this 64-Kbyte page in 32-bit PCI I/O address space. Bits 31−2 are read/write. Bits 1 and 0 are read-only and always return 00b, forcing I/O windows to be aligned on a natural doubleword boundary.

**NOTE:** Either the I/O base register or the I/O limit register must be nonzero to enable any I/O transactions.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **I/O base registers 0, 1**
Offset: 2Ch, 34h
Type: Read-only, Read/Write
Default: 0000 0000h

## 4.22 I/O Limit Registers 0, 1

The I/O limit registers indicate the upper address of a PCI I/O address range. These registers are used by the controller to determine when to forward an I/O transaction to the CardBus bus and when to forward a CardBus cycle to PCI. The lower 16 bits of this register locate the top of the I/O window within a 64-Kbyte page, and the upper 16 bits are a page register that locates this 64-Kbyte page in 32-bit PCI I/O address space. Bits 15−2 are read/write and allow the I/O limit address to be located anywhere in the 64-Kbyte page (indicated by bits 31−16 of the appropriate I/O base) on doubleword boundaries.

Bits 31−16 are read-only and always return 0000h when read. The page is set in the I/O base register. Bits 1 and 0 are read-only and always return 00b, forcing I/O windows to be aligned on a natural doubleword boundary. Write transactions to read-only bits have no effect. The controller assumes that the lower 2 bits of the limit address are 11b.

**NOTE:** The I/O base or the I/O limit register must be nonzero to enable an I/O transaction.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **I/O limit registers 0, 1**
Offset: 30h, 38h
Type: Read-only, Read/Write
Default: 0000 0000h

## 4.23 Interrupt Line Register

The interrupt line register communicates interrupt line routing information.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Register: **Interrupt line**
Offset: 3Ch
Type: Read/Write
Default: FFh

## 4.24 Interrupt Pin Register

The value read from the interrupt pin register is function dependent and depends on the interrupt signaling mode, selected through bits 2−1 (INTMODE field) of the device control register (PCI offset 92h, see Section 4.33).

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Register: **Interrupt pin**
Offset: 3Dh
Type: Read-only
Default: 01h

## 4.25 Bridge Control Register

The bridge control register provides control over various bridging functions. See Table 4−6 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Bridge control**
Offset: 3Eh
Type: Read-only, Read/Write
Default: 0340h

**Table 4−6. Bridge Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15−11 | RSVD | R | Reserved. Bits 15−11 return 00 0000b when read. |
| 10 | POSTEN | RW | Write posting enable. Enables write posting to and from the CardBus socket. Write posting enables posting of write data on burst cycles. Operating with write posting disabled inhibits performance on burst cycles. Note that burst write data can be posted, but various write transactions may not. |
| 9 | PREFETCH1 | RW | Memory window 1 type. Bit 9 specifies whether or not memory window 1 is prefetchable. Bit 9 is encoded as:<br>0 = Memory window 1 is nonprefetchable<br>1 = Memory window 1 is prefetchable (default) |
| 8 | PREFETCH0 | RW | Memory window 0 type. Bit 8 specifies whether or not memory window 0 is prefetchable. This bit is encoded as:<br>0 = Memory window 0 is nonprefetchable<br>1 = Memory window 0 is prefetchable (default) |
| 7 | INTR | RW | PCI interrupt − IREQ routing enable. Bit 7 selects whether PC Card functional interrupts are routed to PCI interrupts or the IRQ specified in the ExCA registers.<br>0 = Functional interrupts routed to PCI interrupts (default)<br>1 = Functional interrupts routed by ExCAs |
| 6 | CRST | RW | CardBus reset. When bit 6 is set, $\overline{\text{CRST}}$ is asserted on the CardBus interface. $\overline{\text{CRST}}$ can also be asserted by passing a $\overline{\text{PRST}}$ assertion to CardBus.<br>0 = $\overline{\text{CRST}}$ deasserted<br>1 = $\overline{\text{CRST}}$ asserted (default) |
| 5 | MABTMODE | RW | Master abort mode. Bit 5 controls how the controller responds to a master abort when the controller is an initiator on the CardBus interface. This bit is common between each socket.<br>0 = Master aborts not reported (default)<br>1 = Signal target abort on PCI and $\overline{\text{SERR}}$ (if enabled) |
| 4 | RSVD | R | Reserved. Bit 4 returns 0b when read. |
| 3 | VGAEN | RW | VGA enable. Bit 3 affects how the controller responds to VGA addresses. When this bit is set, accesses to VGA addresses are forwarded. |
| 2 | ISAEN | RW | ISA mode enable. Bit 2 affects how the controller passes I/O cycles within the 64-Kbyte ISA range. When this bit is set, the controller does not forward the last 768 bytes of each 1K I/O range to CardBus. |
| 1 | CSERREN | RW | $\overline{\text{CSERR}}$ enable. Bit 1 controls the response of the controller to $\overline{\text{CSERR}}$ signals on the CardBus bus.<br>0 = $\overline{\text{CSERR}}$ is not forwarded to PCI $\overline{\text{SERR}}$<br>1 = $\overline{\text{CSERR}}$ is forwarded to PCI $\overline{\text{SERR}}$ |
| 0 | CPERREN | RW | CardBus parity error response enable. Bit 0 controls the response of the controller to CardBus parity errors.<br>0 = CardBus parity errors are ignored<br>1 = CardBus parity errors are reported using $\overline{\text{CPERR}}$ |

## 4.26 Subsystem Vendor ID Register

The subsystem vendor ID register is used for system and option-card identification purposes and may be required for certain operating systems. This register is read-only or read/write, depending on the setting of bit 5 (SUBSYSRW) in the system control register (PCI offset 80h, see Section 4.29).

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Subsystem vendor ID**
Offset: 40h
Type: Read-only (Read/Write if enabled by SUBSYSRW)
Default: 0000h

## 4.27 Subsystem ID Register

The subsystem ID register is used for system and option-card identification purposes and may be required for certain operating systems. This register is read-only or read/write, depending on the setting of bit 5 (SUBSYSRW) in the system control register (PCI offset 80h, see Section 4.29).

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Subsystem ID**
Offset: 42h
Type: Read-only (Read/Write if enabled by SUBSYSRW)
Default: 0000h

## 4.28 PC Card 16-Bit I/F Legacy-Mode Base Address Register

The controller supports the index/data scheme of accessing the ExCA registers, which are mapped by this register. An address written to this register is the address for the index register and the address + 1 is the data address. Using this access method, applications requiring index/data ExCA access can be supported. The base address can be mapped anywhere in 32-bit I/O space on a word boundary; hence, bit 0 is read-only, returning 1b when read. See Section 5, *ExCA Compatibility Registers*, for register offsets.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Register: **PC Card 16-bit I/F legacy-mode base address**
Offset: 44h
Type: Read-only, Read/Write
Default: 0000 0001h

## 4.29 System Control Register

System-level initializations are performed by programming this doubleword register. See Table 4−7 for a complete description of the register contents.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Register: **System control**
Offset: 80h
Type: Read-only, Read/Write, Read/Clear
Default: 0844 9060h

**Table 4−7. System Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31−30 | SER_STEP | RW | Serialized PCI interrupt routing step. Bits 31 and 30 configure the serialized PCI interrupt stream signaling and accomplish an even distribution of interrupts signaled on the four PCI interrupt slots.<br>00 = $\overline{\text{INTA}}$ signal in $\overline{\text{INTA}}$ IRQSER slots<br>01 = $\overline{\text{INTA}}$ signal in $\overline{\text{INTB}}$ IRQSER slots<br>10 = $\overline{\text{INTA}}$ signal in $\overline{\text{INTC}}$ IRQSER slots<br>11 = $\overline{\text{INTA}}$ signal in $\overline{\text{INTD}}$ IRQSER slots |
| 29−28 | RSVD | R | Reserved. Bit 28 returns 0b when read. |
| 27 | OSEN | R/W | Internal oscillator enable.<br>0 = Internal oscillator is disabled<br>1 = Internal oscillator is enabled (default) |
| 26 | SMIROUTE | RW | SMI interrupt routing. Bit 26 selects whether IRQ2 or CSC is signaled when a write occurs to power a PC Card socket.<br>0 = PC Card power change interrupts routed to IRQ2 (default)<br>1 = A CSC interrupt is generated on PC Card power changes |
| 25 | SMISTATUS | RC | SMI interrupt status. This bit is set when bit 24 (SMIENB) is set and a write occurs to set the socket power. Writing a 1b to bit 25 clears the status.<br>0 = SMI interrupt signaled (default)<br>1 = SMI interrupt not signaled |
| 24 | SMIENB | RW | SMI interrupt mode enable. When bit 24 is set and a write to the socket power control occurs, the SMI interrupt signaling is enabled and generates an interrupt. This bit defaults to 0b (disabled). |
| 23 | RSVD | R | Reserved. Bit 23 returns 0b when read. |
| 22 | CBRSVD | RW | CardBus reserved terminals signaling. When a CardBus card is inserted and bit 22 is set, the RSVD CardBus terminals are driven low. When this bit is 0b, these terminals are placed in a high-impedance state.<br>0 = Place CardBus RSVD terminals in a high-impedance state<br>1 = Drive Cardbus RSVD terminals low (default) |
| 21 | VCCPROT | RW | $V_{CC}$ protection enable.<br>0 = $V_{CC}$ protection enabled for 16-bit cards (default)<br>1 = $V_{CC}$ protection disabled for 16-bit cards |
| 20 | REDUCEZV | RW | Reduced zoomed video enable. When this bit is enabled, terminals A25−A22 of the card interface for PC Card-16 cards are placed in the high-impedance state. This bit should not be set for normal ZV operation. This bit is encoded as:<br>0 = Reduced zoomed video disabled (default)<br>1 = Reduced zoomed video enabled |
| 19−16 | RSVD | RW | Reserved. Do not change the default value. |

**Table 4−7. System Control Register Description (Continued)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | MRBURSTDN | RW | Memory read burst enable downstream. When bit 15 is set, memory read transactions are allowed to burst downstream.<br>0 = Downstream memory read burst is disabled<br>1 = Downstream memory read burst is enabled (default) |
| 14 | MRBURSTUP | RW | Memory read burst enable upstream. When bit 14 is set, the controller allows memory read transactions to burst upstream.<br>0 = Upstream memory read burst is disabled (default)<br>1 = Upstream memory read burst is enabled |
| 13 | SOCACTIVE | R | Socket activity status. When set, bit 13 indicates access has been performed to or from a PC card and is cleared upon read of this status bit.<br>0 = No socket activity (default)<br>1 = Socket activity |
| 12 | RSVD | R | Reserved. Bit 12 returns 1b when read. |
| 11 | PWRSTREAM | R | Power stream in progress status bit. When set, bit 11 indicates that a power stream to the power switch is in progress and a powering change has been requested. This bit is cleared when the power stream is complete.<br>0 = Power stream is complete and delay has expired<br>1 = Power stream is in progress |
| 10 | DELAYUP | R | Power-up delay in progress status. When set, bit 10 indicates that a power-up stream has been sent to the power switch and proper power may not yet be stable. This bit is cleared when the power-up delay has expired. |
| 9 | DELAYDOWN | R | Power-down delay in progress status. When set, bit 9 indicates that a power-down stream has been sent to the power switch and proper power may not yet be stable. This bit is cleared when the power-down delay has expired. |
| 8 | INTERROGATE | R | Interrogation in progress. When set, bit 8 indicates an interrogation is in progress and clears when interrogation completes.<br>0 = Interrogation not in progress (default)<br>1 = Interrogation in progress |
| 7 | AUTOPWRSWEN | R/W | Auto power-switch enable<br>0 = Bit 5 (AUTOPWRSWEN) in ExCA power control register (ExCA offset 02h, see Section 5.3) is disabled (default)<br>1 = Bit 5 (AUTOPWRSWEN) in ExCA power control register (ExCA offset 02h, see Section 5.3) is enabled |
| 6 | PWRSAVINGS | RW | Power savings mode enable. When this bit is set, if a CB card is inserted, idle, and without a CB clock, then the applicable CB state machine will not be clocked. |
| 5 | SUBSYSRW | RW | Subsystem ID (PCI offset 42h, see Section 4.27), subsystem vendor ID (PCI offset 40H, see Section 4.26), ExCA identification and revision (ExCA offset 00h/40h/800h, see Section 5.1) registers read/write enable.<br>0 = Subsystem ID, subsystem vendor ID, ExCA identification and revision registers are read/write<br>1 = Subsystem ID, subsystem vendor ID, ExCA identification and revision registers are read-only (default) |
| 4 | CB_DPAR | RW | CardBus data parity $\overline{\text{SERR}}$ signaling enable<br>0 = CardBus data parity not signaled on PCI $\overline{\text{SERR}}$<br>1 = CardBus data parity signaled on PCI $\overline{\text{SERR}}$ |
| 3 | RSVD | RW | Reserved. Do not change the default value. |
| 2 | EXCAPOWER | RW | ExCA power-control bit.<br>0 = Enables 3.3 V<br>1 = Enables 5 V |
| 1 | KEEPCLK | RW | Keep clock. This bit works with PCI and CB $\overline{\text{CLKRUN}}$ protocols.<br>0 = Allows normal functioning of both $\overline{\text{CLKRUN}}$ protocols (default)<br>1 = Does not allow CB clock or PCI clock to be stopped using the $\overline{\text{CLKRUN}}$ protocols |
| 0 | RIMUX | RW | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ multiplex enable.<br>0 = $\overline{\text{RI_OUT}}$ and $\overline{\text{PME}}$ are both routed to the $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ terminal. If both functions are are enabled at the same time, the terminal becomes $\overline{\text{RI_OUT}}$ only and $\overline{\text{PME}}$ assertions are not seen.<br>1 = Only $\overline{\text{PME}}$ is routed to the $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ terminal. |

## 4.30 Multifunction Routing Register

The multifunction routing register is used to configure the MFUNC0−MFUNC6 terminals. These terminals may be configured for various functions. All multifunction terminals default to the general-purpose input configuration. This register is intended to be programmed once at power-on initialization. The default value for this register can also be loaded through a serial bus EEPROM. See Table 4−8 for a complete description of the register contents.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Multifunction routing**
Offset: 8Ch
Type: Read-only, Read/Write
Default: 0000 1000h

**Table 4−8. Multifunction Routing Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31−28 | RSVD | R | Bits 31−28 return 0h when read. |
| 27−24 | MFUNC6 | RW | Multifunction terminal 6 configuration. These bits control the internal signal mapped to the MFUNC6 terminal as follows:<br>0000 = RSVD† 0100 = IRQ4 1000 = IRQ8 1100 = IRQ12<br>0001 = $\overline{\text{CLKRUN}}$ 0101 = IRQ5 1001 = IRQ9 1101 = IRQ13<br>0010 = IRQ2 0110 = IRQ6 1010 = IRQ10 1110 = IRQ14<br>0011 = IRQ3 0111 = IRQ7 1011 = IRQ11 1111 = IRQ15 |
| 23−20 | MFUNC5 | RW | Multifunction terminal 5 configuration. These bits control the internal signal mapped to the MFUNC5 terminal as follows:<br>0000 = GPI4† 0100 = $\overline{\text{IRQ4}}$ 1000 = $\overline{\text{CAUDPWM}}$ 1100 = LED_SKT<br>0001 = $\overline{\text{GPO4}}$ 0101 = $\overline{\text{D3_STAT}}$ 1001 = $\overline{\text{D3_STAT}}$ 1101 = $\overline{\text{LED_SKT}}$<br>0010 = $\overline{\text{PCGNT}}$ 0110 = $\overline{\text{ZVSTAT}}$ 1010 = IRQ10 1110 = $\overline{\text{GPE}}$<br>0011 = IRQ3 0111 = $\overline{\text{ZVSEL0}}$ 1011 = IRQ11 1111 = IRQ15 |
| 19−16 | MFUNC4 | RW | Multifunction terminal 4 configuration. These bits control the internal signal mapped to the MFUNC4 terminal as follows:<br>NOTE: When the serial bus mode is implemented by pulling up the $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$ terminals, the MFUNC4 terminal provides the SCL signaling.<br>0000 = GPI3† 0100 = IRQ4 1000 = CAUDPWM 1100 = $\overline{\text{RI_OUT}}$<br>0001 = $\overline{\text{GPO3}}$ 0101 = IRQ5 1001 = IRQ9 1101 = $\overline{\text{LED_SKT}}$<br>0010 = $\overline{\text{LOCK}}$ PCI 0110 = $\overline{\text{ZVSTAT}}$ 1010 = IRQ10 1110 = $\overline{\text{GPE}}$<br>0011 = IRQ3 0111 = $\overline{\text{ZVSEL0}}$ 1011 = IRQ11 1111 = $\overline{\text{D3_STAT}}$ |
| 15−12 | MFUNC3 | RW | Multifunction terminal 3 configuration. These bits control the internal signal mapped to the MFUNC3 terminal as follows:<br>0000 = RSVD 0100 = IRQ4 1000 = IRQ8 1100 = IRQ12<br>0001 = IRQSER† 0101 = IRQ5 1001 = IRQ9 1101 = IRQ13<br>0010 = IRQ2 0110 = IRQ6 1010 = IRQ10 1110 = IRQ14<br>0011 = IRQ3 0111 = IRQ7 1011 = IRQ11 1111 = IRQ15 |
| 11−8 | MFUNC2 | RW | Multifunction terminal 2 configuration. These bits control the internal signal mapped to the MFUNC2 terminal as follows:<br>0000 = GPI2† 0100 = IRQ4 1000 = CAUDPWM 1100 = $\overline{\text{RI_OUT}}$<br>0001 = $\overline{\text{GPO2}}$ 0101 = IRQ5 1001 = IRQ9 1101 = $\overline{\text{D3_STAT}}$<br>0010 = $\overline{\text{PCREQ}}$ 0110 = $\overline{\text{ZVSTAT}}$ 1010 = $\overline{\text{IRQ10}}$ 1110 = $\overline{\text{GPE}}$<br>0011 = IRQ3 0111 = $\overline{\text{ZVSEL0}}$ 1011 = $\overline{\text{IRQ11}}$ 1111 = IRQ7 |

**Table 4−8. Multifunction Routing Register Description (Continued)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−4 | MFUNC1 | RW | Multifunction terminal 1 configuration. These bits control the internal signal mapped to the MFUNC1 terminal as follows:<br>NOTE: When the serial bus mode is implemented by pulling up the $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$ terminals, the MFUNC1 terminal provides the SDA signaling.<br>0000 = GPI1† 0100 = IRQ4 1000 = CAUDPWM 1100 = LED_SKT<br>0001 = $\underline{\text{GPO1}}$ 0101 = IRQ5 1001 = IRQ9 1101 = $\underline{\text{IRQ13}}$<br>0010 = D3_STAT 0110 = $\underline{\text{ZVSTAT}}$ 1010 = IRQ10 1110 = $\overline{\text{GPE}}$<br>0011 = IRQ3 0111 = ZVSEL0 1011 = IRQ11 1111 = IRQ15 |
| 3−0 | MFUNC0 | RW | Multifunction terminal 0 configuration. These bits control the internal signal mapped to the MFUNC0 terminal as follows:<br>0000 = GPI0† 0100 = IRQ4 1000 = CAUDPWM 1100 = LED_SKT<br>0001 = $\underline{\text{GPO0}}$ 0101 = IRQ5 1001 = IRQ9 1101 = $\underline{\text{IRQ13}}$<br>0010 = $\overline{\text{INTA}}$ 0110 = $\underline{\text{ZVSTAT}}$ 1010 = IRQ10 1110 = $\overline{\text{GPE}}$<br>0011 = IRQ3 0111 = ZVSEL0 1011 = IRQ11 1111 = IRQ15 |

† Default value

## 4.31 Retry Status Register

The retry status register enables the retry timeout counters and displays the retry expiration status. The flags are set when the controller retries a PCI or CardBus master request and the master does not return within $2^{15}$ PCI clock cycles. The flags are cleared by writing a 1b to the bit. These bits are expected to be incorporated into the PCI command, PCI status, and bridge control registers by the PCI SIG. See Table 4−9 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Retry status**
Offset: 90h
Type: Read-only, Read/Write, Read/Clear
Default: C0h

**Table 4−9. Retry Status Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | PCIRETRY | RW | PCI retry timeout counter enable. Bit 7 is encoded:<br>0 = PCI retry counter disabled<br>1 = PCI retry counter enabled (default) |
| 6 | CBRETRY | RW | CardBus retry timeout counter enable. Bit 6 is encoded:<br>0 = CardBus retry counter disabled<br>1 = CardBus retry counter enabled (default) |
| 5 | TEXP_CBB | RC | CardBus target B retry expired. Write a 1b to clear bit 5.<br>0 = Inactive (default)<br>1 = Retry has expired |
| 4 | RSVD | R | Reserved. Bit 4 returns 0b when read. |
| 3 | TEXP_CBA | RC | CardBus target A retry expired. Write a 1b to clear bit 3.<br>0 = Inactive (default)<br>1 = Retry has expired |
| 2 | RSVD | R | Reserved. Bit 2 returns 0b when read. |
| 1 | TEXP_PCI | RC | PCI target retry expired. Write a 1b to clear bit 1.<br>0 = Inactive (default)<br>1 = Retry has expired |
| 0 | RSVD | R | Reserved. Bit 0 returns 0b when read. |

## 4.32 Card Control Register

The card control register is provided for PCI1130 compatibility. $\overline{\text{RI_OUT}}$ is enabled through this register. See Table 4−10 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Card control**
Offset: 91h
Type: Read-only, Read/Write, Read/Clear
Default: 00h

**Table 4−10. Card Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | RIENB | RW | Ring indicate output enable.<br>0 = Disables any routing of $\overline{\text{RI_OUT}}$ signal (default)<br>1 = Enables $\overline{\text{RI_OUT}}$ signal for routing to the $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ terminal, when RIMUX is set to 0b, and for routing to MFUNC2 or MFUNC4 |
| 6 | ZVENABLE | RW | Compatibility ZV mode enable. When set, the corresponding PC Card socket interface ZV terminals enter a high-impedance state. This bit defaults to 0b. |
| 5 | RSVD | RW | Reserved. Do not change default value. |
| 4−3 | RSVD | R | Reserved. Bits 4 and 3 return 00b when read. |
| 2 | AUD2MUX | RW | CardBus audio-to-IRQMUX. When set, the CAUDIO CardBus signal is routed to the corresponding multifunction terminal which may be configured for CAUDPWM. |
| 1 | SPKROUTEN | RW | Speaker out enable. When bit 1 is set, $\overline{\text{SPKR}}$ on the PC Card is enabled and is routed to SPKROUT. The SPKROUT terminal drives data only when the SPKROUTEN bit is set. This bit is encoded as:<br>0 = $\overline{\text{SPKR}}$ to SPKROUT not enabled<br>1 = $\overline{\text{SPKR}}$ to SPKROUT enabled |
| 0 | IFG | RC | Interrupt flag. Bit 0 is the interrupt flag for 16-bit I/O PC Cards and for CardBus cards. Bit 0 is set when a functional interrupt is signaled from a PC Card interface. Write back a 1b to clear this bit.<br>0 = No PC Card functional interrupt detected (default).<br>1 = PC Card functional interrupt detected. |

## 4.33 Device Control Register

The device control register is provided for PCI1130 compatibility. See Table 4−11 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |

Register: **Device control**
Offset: 92h
Type: Read-only, Read/Write
Default: 66h

**Table 4−11. Device Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | SKTPWR_LOCK | RW | Socket power lock bit. When this bit is set to 1b, software cannot power down the PC Card socket while in D3. This may be necessary to support wake on LAN or RING if the operating system is programmed to power down a socket when the CardBus controller is placed in the D3 state. |
| 6 | 3VCAPABLE | RW | 3-V socket capable force<br>0 = Not 3-V capable<br>1 = 3-V capable (default) |
| 5 | IO16V2 | RW | Diagnostic bit. This bit defaults to 1b. |
| 4 | RSVD | R | Reserved. Bit 4 returns 0b when read. |
| 3 | TEST | RW | TI test. Only a 0b should be written to bit 3. |
| 2−1 | INTMODE | RW | Interrupt signaling mode. Bits 2 and 1 select the interrupt signaling mode. The interrupt signaling mode bits are encoded:<br>00 = Parallel PCI interrupts only<br>01 = Parallel IRQ and parallel PCI interrupts<br>10 = IRQ serialized interrupts and parallel PCI interrupt<br>11 = IRQ and PCI serialized interrupts (default) |
| 0 | RSVD | RW | Reserved. Bit 0 is reserved for test purposes. Only 0b should be written to this bit. |

## 4.34 Diagnostic Register

The diagnostic register is provided for internal TI test purposes. It is a read/write register, but only 00h should be written to it. See Table 4−12 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Register: **Diagnostic**
Offset: 93h
Type: Read/Write
Default: 60h

**Table 4−12. Diagnostic Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | TRUE_VAL | RW | This bit defaults to 0b. This bit is encoded as:<br>0 = Reads true values in PCI vendor ID and PCI device ID registers (default)<br>1 = Reads all 1s in reads from the PCI vendor ID and PCI device ID registers |
| 6 | RSVD | R | Reserved. Bit 6 returns 1b when read. |
| 5 | CSC | RW | CSC interrupt routing control<br>0 = CSC interrupts routed to PCI if ExCA 803 bit 4 = 1b<br>1 = CSC interrupts routed to PCI if ExCA 805 bits 7−4 = 0000b (default)<br>In this case, the setting of ExCA 803 bit 4 is a don't care. |
| 4 | DIAG4 | RW | Diagnostic RETRY_DIS. Delayed transaction disable. |
| 3 | DIAG3 | RW | Diagnostic RETRY_EXT. Extends the latency from 16 to 64. |
| 2 | DIAG2 | RW | Diagnostic DISCARD_TIM_SEL_CB. Set = $2^{10}$, reset = $2^{15}$. |
| 1 | DIAG1 | RW | Diagnostic DISCARD_TIM_SEL_PCI. Set = $2^{10}$, reset = $2^{15}$. |
| 0 | STDZVEN | RW | Standardized zoomed video register model enable.<br>0 = Enable the standardized zoomed video register model (default)<br>1 = Disable the standardized zoomed video register model |

## 4.35 Capability ID Register

The capability ID register identifies the linked list item as the register for PCI power management. The register returns 01h when read, which is the unique ID assigned by the PCI SIG for the PCI location of the capabilities pointer and the value.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Register: **Capability ID**
Offset: A0h
Type: Read-only
Default: 01h

## 4.36 Next-Item Pointer Register

The next-item pointer register indicates the next item in the linked list of the PCI power-management capabilities. Because the controller function includes only one capabilities item, this register returns 00h when read.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Next-item pointer**
Offset: A1h
Type: Read-only
Default: 00h

## 4.37 Power-Management Capabilities Register

This register contains information on the capabilities of the PC Card function related to power management. The CardBus bridge supports the D0, D1, D2, and D3 power states. See Table 4−13 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

Register: **Power-management capabilities**
Offset: A2h
Type: Read/Write, Read-only
Default: FE12h

**Table 4−13. Power-Management Capabilities Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | PME_Support | RW | $\overline{\text{PME}}$ support. This 5-bit field indicates the power states from which the controller function may assert $\overline{\text{PME}}$. A 0b for any bit indicates that the function cannot assert the $\overline{\text{PME}}$ signal while in that power state. These five bits return 11111b when read. Each of these bits is described below:<br>Bit 15 defaults to 1b indicating the $\overline{\text{PME}}$ signal can be asserted from the $D3_{cold}$ state. This bit is R/W because wake-up support from $D3_{cold}$ is contingent on the system providing an auxiliary power source to the $V_{CC}$ terminals. If the system designer chooses not to provide an auxiliary power source to the $V_{CC}$ terminals for $D3_{cold}$ wake-up support, then BIOS should write a 0b to this bit. |
| 14−11 | PME_Support | R | Bit 14 contains the value 1b, indicating that the $\overline{\text{PME}}$ signal can be asserted from $D3_{hot}$ state.<br>Bit 13 contains the value 1b, indicating that the $\overline{\text{PME}}$ signal can be asserted from D2 state.<br>Bit 12 contains the value 1b, indicating that the $\overline{\text{PME}}$ signal can be asserted from D1 state.<br>Bit 11 contains the value 1b, indicating that the $\overline{\text{PME}}$ signal can be asserted from the D0 state. |
| 10 | D2_Support | R | D2 support. Bit 10 returns a 1b when read, indicating that the CardBus function supports the D2 device power state. |
| 9 | D1_Support | R | D1 support. Bit 9 returns a 1b when read, indicating that the CardBus function supports the D1 device power state. |
| 8−6 | RSVD | R | Reserved. Bits 8−6 return 000b when read. |
| 5 | DSI | R | Device-specific initialization. Bit 5 returns 1b when read, indicating that the CardBus controller function requires special initialization (beyond the standard PCI configuration header) before the generic-class device driver is able to use it. |
| 4 | AUX_PWR | R | Auxiliary power source. Bit 4 is tied to bit 15. When bit 4 is set, it indicates that support for $\overline{\text{PME}}$ in $D3_{cold}$ requires auxiliary power supplied by the system by way of a proprietary delivery vehicle. When bit 4 is 0b, it indicates that the function supplies its own auxiliary power source. |
| 3 | PMECLK | R | $\overline{\text{PME}}$ clock. Bit 3 returns 0b when read, indicating that no host bus clock is required for the controller to generate $\overline{\text{PME}}$. |
| 2−0 | VERSION | R | Version. Bits 2−0 return 010b when read, indicating that the power-management registers (PCI offsets A4h−A7h, see Sections 4.38−4.40) are defined in the *PCI Bus Power Management Interface Specification* version 1.1. |

## 4.38 Power-Management Control/Status Register

The power-management control/status register determines and changes the current power state of the controller CardBus function. The contents of this register are not affected by the internally-generated reset caused by the transition from $D3_{hot}$ to D0 state. All PCI, ExCA, and CardBus registers are reset as a result of a $D3_{hot}$ to D0 state transition. TI-specific registers, PCI power-management registers, and the PC Card 16-bit legacy-mode base address register (PCI offset 44h, see Section 4.28) are not reset. See Table 4−14 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Power-management control/status**
Offset: A4h
Type: Read-only, Read/Write, Read/Clear
Default: 0000h

**Table 4−14. Power-Management Control/Status Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | PMESTAT | RC | $\overline{\text{PME}}$ status. Bit 15 is set when the CardBus function would normally assert $\overline{\text{PME}}$, independent of the state of bit 8 (PME_EN). Bit 15 is cleared by a writeback of 1b, and this also clears the $\overline{\text{PME}}$ signal if $\overline{\text{PME}}$ was asserted by this function. Writing a 0b to this bit has no effect. |
| 14−13 | DATASCALE | R | Data scale. This 2-bit field returns 00b when read. The CardBus function does not return any dynamic data. |
| 12−9 | DATASEL | R | Data select. This 4-bit field returns 0h when read. The CardBus function does not return any dynamic data. |
| 8 | PME_EN | RW | $\overline{\text{PME}}$ enable. Bit 8 enables the function to assert $\overline{\text{PME}}$. If this bit is cleared, then assertion of $\overline{\text{PME}}$ is disabled. |
| 7−2 | RSVD | R | Reserved. Bits 7−2 return 000000b when read. |
| 1−0 | PWR_STATE | RW | Power state. This 2-bit field is used both to determine the current power state of a function and to set the function into a new power state. This field is encoded as:<br>00 = D0<br>01 = D1<br>10 = D2<br>11 = $D3_{hot}$ |

## 4.39 Power-Management Control/Status Register Bridge Support Extensions

The power-management control/status register bridge support extensions support PCI bridge specific functionality. See Table 4−15 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Power-management control/status register bridge support extensions**
Offset: A6h
Type: Read-only
Default: C0h

**Table 4−15. Power-Management Control/Status Register Bridge Support Extensions Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | BPCC_EN | R | BPCC_Enable. Bus power/clock control enable. This bit returns 1b when read.<br>This bit is encoded as:<br>0 = Bus power/clock control is disabled<br>1 = Bus power/clock control is enabled (default)<br>A 0b indicates that the bus power/clock control policies defined in the *PCI Bus Power Management Interface Specification* are disabled. When the bus power/clock control enable mechanism is disabled, the bridge power-management control/status register power state field (see Section 4.38, bits 1−0) cannot be used by the system software to control the power or the clock of the bridge secondary bus. A 1b indicates that the bus power/clock control mechanism is enabled. |
| 6 | B2_$\overline{B3}$ | R | B2/B3 support for $D3_{hot}$. The state of this bit determines the action that is to occur as a direct result of programming the function to $D3_{hot}$. This bit is only meaningful if bit 7 (BPCC_EN) is a 1b. This bit is encoded as:<br>0 = When the bridge is programmed to $D3_{hot}$, its secondary bus has its power removed (B3)<br>1 = When the bridge function is programmed to $D3_{hot}$, its secondary bus PCI clock is stopped (B2) (default) |
| 5−0 | RSVD | R | Reserved. Bits 5−0 return 000000b when read. |

## 4.40 Power-Management Data Register

The power-management data register returns 00h when read, because the CardBus function does not report dynamic data.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Power-management data**
Offset: A7h
Type: Read-only
Default: 00h

## 4.41 General-Purpose Event Status Register

The general-purpose event status register contains status bits that are set when events occur that are controlled by the general-purpose control register. The bits in this register and the corresponding $\overline{\text{GPE}}$ are cleared by writing a 1b to the corresponding bit location. The status bits in this register do not depend upon the states of corresponding bits in the general-purpose enable register. See Table 4−16 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **General-purpose event status**
Offset: A8h
Type: Read-only, Read/Clear
Default: 0000h

**Table 4−16. General-Purpose Event Status Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | ZV_STS | RC | PC Card socket 0 ZV status. Bit 15 is set on a change in status of bit 6 (ZVENABLE) in the function 0 card control register (PCI offset 91h, see Section 4.32). |
| 14−12 | RSVD | R | Reserved. Bits 14−12 return 000b when read. |
| 11 | PWR_STS | RC | Power-change status. Bit 11 is set when software has changed the power state of the socket. A change in either $V_{CC}$ or $V_{PP}$ causes this bit to be set. |
| 10−9 | RSVD | R | Reserved. Bits 10 and 9 return 00b when read. |
| 8 | VPP12_STS | RC | 12-V $V_{PP}$ request status. Bit 8 is set when software has changed the requested $V_{PP}$ level to or from 12 V for the PC Card socket. |
| 7−5 | RSVD | R | Reserved. Bits 7−5 return 000b when read. |
| 4 | GP4_STS | RC | GPI4 Status. Bit 4 is set on a change in status of the MFUNC5 terminal input level. |
| 3 | GP3_STS | RC | GPI3 Status. Bit 3 is set on a change in status of the MFUNC4 terminal input level. |
| 2 | GP2_STS | RC | GPI2 Status. Bit 2 is set on a change in status of the MFUNC2 terminal input level. |
| 1 | GP1_STS | RC | GPI1 Status. Bit 1 is set on a change in status of the MFUNC1 terminal input level. |
| 0 | GP0_STS | RC | GPI0 Status. Bit 0 is set on a change in status of the MFUNC0 terminal input level. |

## 4.42 General-Purpose Event Enable Register

The general-purpose event enable register contains bits that are set to enable a $\overline{\text{GPE}}$ signal. The $\overline{\text{GPE}}$ signal is driven until the corresponding status bit is cleared and the event is serviced. The $\overline{\text{GPE}}$ can only be signaled if one of the multifunction terminals, MFUNC6−MFUNC0, is configured for $\overline{\text{GPE}}$ signaling. See Table 4−17 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **General-purpose event enable**
Offset: AAh
Type: Read-only, Read/Write
Default: 0000h

**Table 4−17. General-Purpose Event Enable Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | ZV0_EN | RW | PC Card ZV enable. When bit 15 is set, a $\overline{\text{GPE}}$ is signaled on a change in status of bit 6 (ZVENABLE) in the card control register (PCI offset 91h, see Section 4.32). |
| 14−12 | RSVD | R | Reserved. Bits 14−12 return 000b when read. |
| 11 | PWR_EN | RW | Power change enable. When bit 11 is set, a $\overline{\text{GPE}}$ is signaled on when software has changed the power state. |
| 10−9 | RSVD | R | Reserved. Bits 10 and 9 return 00b when read. |
| 8 | VPP12_EN | RW | 12-V $V_{PP}$ request enable. When bit 8 is set, a $\overline{\text{GPE}}$ is signaled when software has changed the requested $V_{PP}$ level to or from 12 V. |
| 7−5 | RSVD | R | Reserved. Bits 7−5 return 000b when read. |
| 4 | GP4_EN | RW | GPI4 enable. When bit 4 is set, a $\overline{\text{GPE}}$ is signaled when there has been a change in status of the MFUNC5 terminal input level if configured as GPI4. |
| 3 | GP3_EN | RW | GPI3 enable. When bit 3 is set, a $\overline{\text{GPE}}$ is signaled when there has been a change in status of the MFUNC4 terminal input level if configured as GPI3. |
| 2 | GP2_EN | RW | GPI2 enable. When bit 2 is set, a $\overline{\text{GPE}}$ is signaled when there has been a change in status of the MFUNC2 terminal input if configured as GPI2. |
| 1 | GP1_EN | RW | GPI1 enable. When bit 1 is set, a $\overline{\text{GPE}}$ is signaled when there has been a change in status of the MFUNC1 terminal input if configured as GPI1. |
| 0 | GP0_EN | RW | GPI0 enable. When bit 0 is set, a $\overline{\text{GPE}}$ is signaled when there has been a change in status of the MFUNC0 terminal input if configured as GPI0. |

## 4.43 General-Purpose Input Register

The general-purpose input register provides the logical value of the data input from the GPI terminals, MFUNC5, MFUNC4, and MFUNC2−MFUNC0. See Table 4−18 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | X | X | X | X | X |

Register: **General-purpose input**
Offset: ACh
Type: Read-only
Default: 00XXh

**Table 4−18. General-Purpose Input Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15−5 | RSVD | R | Reserved. Bits 15−5 return 0s when read. |
| 4 | GPI4_DATA | R | GPI4 data bit. The value read from bit 4 represents the logical value of the data input from the MFUNC5 terminal. |
| 3 | GPI3_DATA | R | GPI3 data bit. The value read from bit 3 represents the logical value of the data input from the MFUNC4 terminal. |
| 2 | GPI2_DATA | R | GPI2 data bit. The value read from bit 2 represents the logical value of the data input from the MFUNC2 terminal. |
| 1 | GPI1_DATA | R | GPI1 data bit. The value read from bit 1 represents the logical value of the data input from the MFUNC1 terminal. |
| 0 | GPI0_DATA | R | GPI0 data bit. The value read from bit 0 represents the logical value of the data input from the MFUNC0 terminal. |

## 4.44 General-Purpose Output Register

The general-purpose output register is used for control of the general-purpose outputs. See Table 4−19 for a complete description of the register contents.

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **General-purpose output**
Offset: AEh
Type: Read-only, Read/Write
Default: 0000h

**Table 4−19. General-Purpose Output Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15−5 | RSVD | R | Reserved. Bits 15−5 return 0s when read. |
| 4 | GPO4_DATA | RW | GPO4 data bit. The value written to bit 4 represents the logical value of the data driven to the MFUNC5 terminal if configured as GPO4. Read transactions return the last data value written. |
| 3 | GPO3_DATA | RW | GPIO3 data bit. The value written to bit 3 represents the logical value of the data driven to the MFUNC4 terminal if configured as GPO3. Read transactions return the last data value written. |
| 2 | GPO2_DATA | RW | GPO2 data bit. The value written to bit 2 represents the logical value of the data driven to the MFUNC2 terminal if configured as GPO2. Read transactions return the last data value written. |
| 1 | GPO1_DATA | RW | GPO1 data bit. The value written to bit 1 represents the logical value of the data driven to the MFUNC1 terminal if configured as GPO1. Read transactions return the last data value written. |
| 0 | GPO0_DATA | RW | GPO0 data bit. The value written to bit 0 represents the logical value of the data driven to the MFUNC0 terminal if configured as GPO0. Read transactions return the last data value written. |

## 4.45 Serial-Bus Data Register

The serial-bus data register is for programmable serial-bus byte reads and writes. This register represents the data when generating cycles on the serial bus interface. To write a byte, this register must be programmed with the data, the serial bus index register must be programmed with the byte address, the serial-bus slave address must be programmed with the 7-bit slave address, and the read/write indicator bit must be reset.

On byte reads, the byte address is programmed into the serial-bus index register, the serial bus slave address register must be programmed with both the 7-bit slave address and the read/write indicator bit, and bit 5 (REQBUSY) in the serial bus control and status register (PCI offset B3h, see Section 4.48) must be polled until clear. Then the contents of this register are valid read data from the serial bus interface. See Table 4−20 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial-bus data**
Offset: B0h
Type: Read/Write
Default: 00h

**Table 4−20. Serial-Bus Data Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−0 | SBDATA | RW | Serial-bus data. This bit field represents the data byte in a read or write transaction on the serial interface. On reads, the REQBUSY bit must be polled to verify that the contents of this register are valid. |

## 4.46 Serial-Bus Index Register

The serial-bus index register is for programmable serial-bus byte reads and writes. This register represents the byte address when generating cycles on the serial-bus interface. To write a byte, the serial-bus data register must be programmed with the data, this register must be programmed with the byte address, and the serial-bus slave address register must be programmed with both the 7-bit slave address and the read/write indicator bit.

On byte reads, the word address is programmed into this register, the serial-bus slave address must be programmed with both the 7-bit slave address and the read/write indicator bit, and bit 5 (REQBUSY) in the serial-bus control and status register (see Section 4.48) must be polled until clear. Then the contents of the serial-bus data register are valid read data from the serial-bus interface. See Table 4−21 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial-bus index**
Offset: B1h
Type: Read/Write
Default: 00h

**Table 4−21. Serial-Bus Index Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−0 | SBINDEX | RW | Serial-bus index. This bit field represents the byte address in a read or write transaction on the serial interface. |

## 4.47 Serial-Bus Slave Address Register

The serial-bus slave address register is for programmable serial-bus byte read and write transactions. To write a byte, the serial-bus data register must be programmed with the data, the serial-bus index register must be programmed with the byte address, and this register must be programmed with both the 7-bit slave address and the read/write indicator bit.

On byte reads, the byte address is programmed into the serial bus index register, this register must be programmed with both the 7-bit slave address and the read/write indicator bit, and bit 5 (REQBUSY) in the serial-bus control and status register (PCI offset B3h, see Section 4.48) must be polled until clear. Then the contents of the serial-bus data register are valid read data from the serial-bus interface. See Table 4−22 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial-bus slave address**
Offset: B2h
Type: Read/Write
Default: 00h

**Table 4−22. Serial-Bus Slave Address Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−1 | SLAVADDR | RW | Serial-bus slave address. This bit field represents the slave address of a read or write transaction on the serial interface. |
| 0 | RWCMD | RW | Read/write command. Bit 0 indicates the read/write command bit presented to the serial bus on byte read and write accesses.<br>0 = A byte write access is requested to the serial bus interface<br>1 = A byte read access is requested to the serial bus interface |

## 4.48 Serial-Bus Control and Status Register

The serial-bus control and status register communicates serial-bus status information and selects the quick command protocol. Bit 5 (REQBUSY) in this register must be polled during serial-bus byte reads to indicate when data is valid in the serial-bus data register. See Table 4−23 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial-bus control and status**
Offset: B3h (function 0)
Type: Read-only, Read/Write, Read/Clear
Default: 00h

**Table 4−23. Serial-Bus Control and Status Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | PROT_SEL | RW | Protocol select. When bit 7 is set, the send-byte protocol is used on write requests and the receive-byte protocol is used on read commands. The word-address byte in the serial-bus index register (PCI offset B1h, see Section 4.46) is not output by the controller when bit 7 is set. |
| 6 | RSVD | R | Reserved. Bit 6 returns 0b when read. |
| 5 | REQBUSY | R | Requested serial-bus access busy. Bit 5 indicates that a requested serial-bus access (byte read or write) is in progress. A request is made, and bit 5 is set, by writing to the serial-bus slave address register (PCI offset B2h, see Section 4.47). Bit 5 must be polled on reads from the serial interface. After the byte read access has been requested, the read data is valid in the serial-bus data register. |
| 4 | ROMBUSY | R | Serial EEPROM busy status. Bit 4 indicates the status of the serial EEPROM circuitry. Bit 4 is set during the loading of the subsystem ID and other default values from the serial-bus EEPROM.<br>0 = Serial EEPROM circuitry is not busy<br>1 = Serial EEPROM circuitry is busy |
| 3 | SBDETECT | RC | Serial-bus detect. When bit 3 is set, it indicates that the serial-bus interface is detected through pullup resistors on the $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$ terminals after reset. If bit 3 is reset, then the MFUNC4 and MFUNC1 terminals can be used for alternate functions such as general-purpose inputs and outputs.<br>0 = Serial-bus interface not detected<br>1 = Serial-bus interface detected |
| 2 | SBTEST | RW | Serial-bus test. When bit 2 is set, the serial-bus clock frequency is increased for test purposes.<br>0 = Serial-bus clock at normal operating frequency, ≈ 100 kHz (default)<br>1 = Serial-bus clock frequency increased for test purposes |
| 1 | REQ_ERR | RC | Requested serial-bus access error. Bit 1 indicates when a data error occurs on the serial interface during a requested cycle, and can be set due to a missing acknowledge. Bit 1 is cleared by a writeback of 1b.<br>0 = No error detected during user-requested byte read or write cycle<br>1 = Data error detected during user-requested byte read or write cycle |
| 0 | ROM_ERR | RC | EEPROM data-error status. Bit 0 indicates when a data error occurs on the serial interface during the auto-load from the serial-bus EEPROM, and can be set due to a missing acknowledge. Bit 0 is also set on invalid EEPROM data formats. See Section 3.6.1, *Serial Bus Interface Implementation*, for details on EEPROM data format. Bit 0 is cleared by a writeback of 1b.<br>0 = No error detected during auto-load from serial-bus EEPROM<br>1 = Data error detected during auto-load from serial-bus EEPROM |

# 5 ExCA Compatibility Registers

The ExCA registers implemented in the PCI1510 controller are register-compatible with the Intel 82365SL−DF PCMCIA controller. ExCA registers are identified by an offset value that is compatible with the legacy I/O index/data scheme used on the Intel 82365 ISA controller. The ExCA registers are accessed through this scheme by writing the register offset value into the index register (I/O base) and reading or writing the data register (I/O base + 1). The I/O base address used in the index/data scheme is programmed in the PC Card 16-bit I/F legacy-mode base address register (PCI offset 44h, see Section 4.28). The offsets from this base address run contiguously from 00h to 3Fh. See Figure 5−1 for an ExCA I/O mapping illustration.

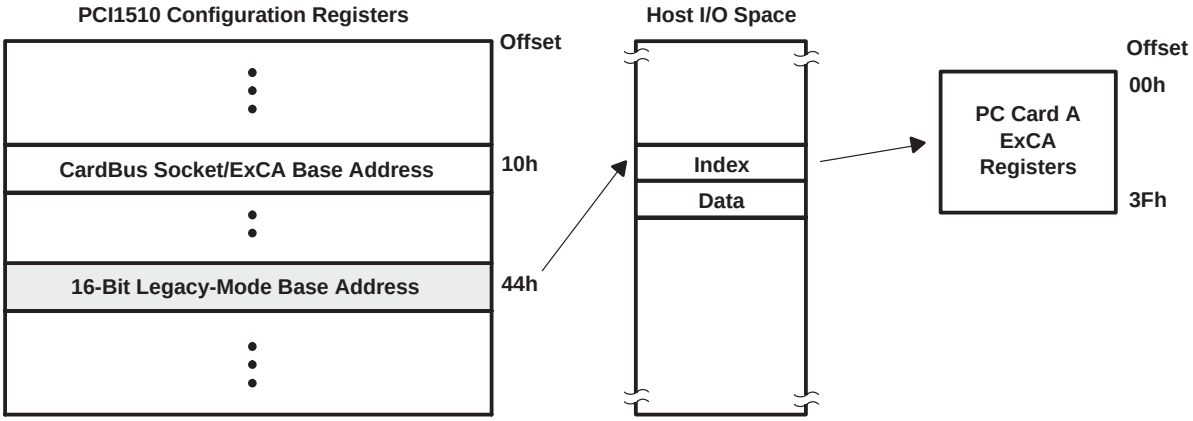



**Figure 5−1. ExCA Register Access Through I/O**

The controller also provides a memory-mapped alias of the ExCA registers by directly mapping them into PCI memory space. They are located through the CardBus socket/ExCA base-address register (PCI offset 10h, see Section 4.12) at memory offset 800h. See Figure 5−2 for an ExCA memory mapping illustration. This illustration also identifies the CardBus socket register mapping, which is mapped into the same 4-K window at memory offset 00h.

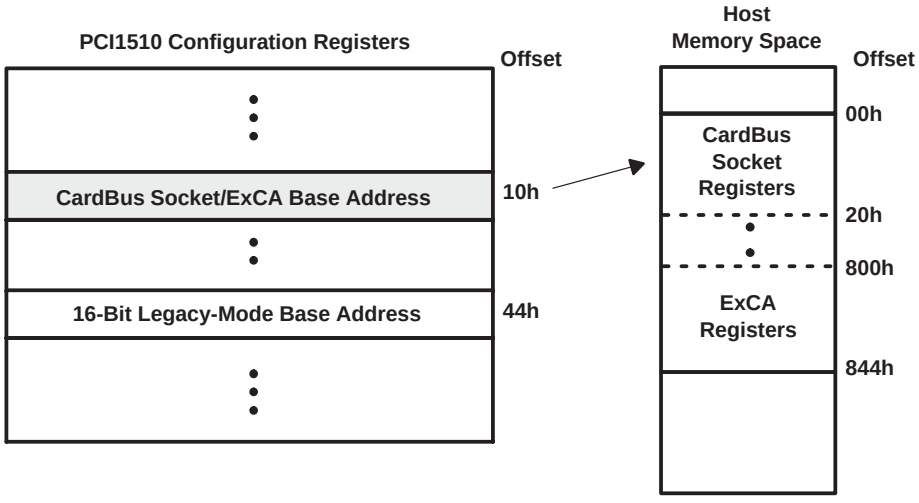



**Figure 5−2. ExCA Register Access Through Memory**

The interrupt registers in the ExCA register set, as defined by the 82365SL−DL specification, control such card functions as reset, type, interrupt routing, and interrupt enables. Special attention must be paid to the interrupt routing registers and the host interrupt signaling method selected for the controller to ensure that all possible interrupts can potentially be routed to the programmable interrupt controller. The ExCA registers that are critical to the interrupt

signaling are the ExCA interrupt and general control register (ExCA offset 03h/43h/803h, see Section 5.4) and the ExCA card status-change interrupt configuration register (05h/45h/805h, see Section 5.6).

Access to I/O mapped 16-bit PC cards is available to the host system via two ExCA I/O windows. These are regions of host I/O address space into which the card I/O space is mapped. These windows are defined by start, end, and offset addresses programmed in the ExCA registers described in this section. I/O windows have byte granularity.

Access to memory mapped 16-bit PC Cards is available to the host system via five ExCA memory windows. These are regions of host memory space into which the card memory space is mapped. These windows are defined by start, end, and offset addresses programmed in the ExCA registers described in this section. Table 5−1 identifies each ExCA register and its respective ExCA offset. Memory windows have 4-Kbyte granularity.

**Table 5−1. ExCA Registers and Offsets**

| ExCA REGISTER NAME | PCI MEMORY ADDRESS OFFSET (HEX) | ExCA OFFSET (HEX) |
|---|---|---|
| Identification and revision | 800 | 00 |
| Interface status | 801 | 01 |
| Power control | 802 | 02 |
| Interrupt and general control | 803 | 03 |
| Card status change | 804 | 04 |
| Card status-change interrupt configuration | 805 | 05 |
| Address window enable | 806 | 06 |
| I / O window control | 807 | 07 |
| I / O window 0 start-address low byte | 808 | 08 |
| I / O window 0 start-address high byte | 809 | 09 |
| I / O window 0 end-address low byte | 80A | 0A |
| I / O window 0 end-address high byte | 80B | 0B |
| I / O window 1 start-address low byte | 80C | 0C |
| I / O window 1 start-address high byte | 80D | 0D |
| I / O window 1 end-address low byte | 80E | 0E |
| I / O window 1 end-address high byte | 80F | 0F |
| Memory window 0 start-address low byte | 810 | 10 |
| Memory window 0 start-address high byte | 811 | 11 |
| Memory window 0 end-address low byte | 812 | 12 |
| Memory window 0 end-address high byte | 813 | 13 |
| Memory window 0 offset-address low byte | 814 | 14 |
| Memory window 0 offset-address high byte | 815 | 15 |
| Card detect and general control | 816 | 16 |
| Reserved | 817 | 17 |
| Memory window 1 start-address low byte | 818 | 18 |
| Memory window 1 start-address high byte | 819 | 19 |
| Memory window 1 end-address low byte | 81A | 1A |
| Memory window 1 end-address high byte | 81B | 1B |
| Memory window 1 offset-address low byte | 81C | 1C |
| Memory window 1 offset-address high byte | 81D | 1D |
| Global control | 81E | 1E |
| Reserved | 81F | 1F |
| Memory window 2 start-address low byte | 820 | 20 |
| Memory window 2 start-address high byte | 821 | 21 |
| Memory window 2 end-address low byte | 822 | 22 |

**Table 5−1. ExCA Registers and Offsets (Continued)**

| ExCA REGISTER NAME | PCI MEMORY ADDRESS OFFSET (HEX) | ExCA OFFSET (HEX) |
|---|---|---|
| Memory window 2 end-address high byte | 823 | 23 |
| Memory window 2 offset-address low byte | 824 | 24 |
| Memory window 2 offset-address high byte | 825 | 25 |
| Reserved | 826 | 26 |
| Reserved | 827 | 27 |
| Memory window 3 start-address low byte | 828 | 28 |
| Memory window 3 start-address high byte | 829 | 29 |
| Memory window 3 end-address low byte | 82A | 2A |
| Memory window 3 end-address high byte | 82B | 2B |
| Memory window 3 offset-address low byte | 82C | 2C |
| Memory window 3 offset-address high byte | 82D | 2D |
| Reserved | 82E | 2E |
| Reserved | 82F | 2F |
| Memory window 4 start-address low byte | 830 | 30 |
| Memory window 4 start-address high byte | 831 | 31 |
| Memory window 4 end-address low byte | 832 | 32 |
| Memory window 4 end-address high byte | 833 | 33 |
| Memory window 4 offset-address low byte | 834 | 34 |
| Memory window 4 offset-address high byte | 835 | 35 |
| I/O window 0 offset-address low byte | 836 | 36 |
| I/O window 0 offset-address high byte | 837 | 37 |
| I/O window 1 offset-address low byte | 838 | 38 |
| I/O window 1 offset-address high byte | 839 | 39 |
| Reserved | 83A | 3A |
| Reserved | 83B | 3B |
| Reserved | 83C | 3C |
| Reserved | 83D | 3D |
| Reserved | 83E | 3E |
| Reserved | 83F | 3F |
| Memory window page 0 | 840 | − |
| Memory window page 1 | 841 | − |
| Memory window page 2 | 842 | − |
| Memory window page 3 | 843 | − |
| Memory window page 4 | 844 | − |

A bit description table, typically included when a register contains bits of more than one type or purpose, indicates bit field names, which appear in the signal column; a detailed field description, which appears in the function column; and field access tags, which appear in the type column of the bit description table. Table 4−2 describes the field access tags.

## 5.1 ExCA Identification and Revision Register

The ExCA identification and revision register provides host software with information on 16-bit PC Card support and Intel 82365SL-DF compatibility. This register is read-only or read/write, depending on the setting of bit 5 (SUBSYSRW) in the system control register (PCI offset 80h, see Section 4.29). See Table 5−2 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

Register: **ExCA identification and revision**
Offset: CardBus socket address + 800h; ExCA offset 00h
Type: Read-only, Read/Write
Default: 84h

**Table 5−2. ExCA Identification and Revision Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−6 | IFTYPE | R | Interface type. These bits, which are hardwired as 10b, identify the 16-bit PC Card support provided by the controller. The controller supports both I/O and memory 16-bit PC cards. |
| 5−4 | RSVD | RW | Reserved. Bits 5 and 4 can be used for Intel 82365SL-DF emulation. |
| 3−0 | 365REV | RW | Intel 82365SL-DF revision. This field stores the Intel 82365SL-DF revision supported by the controller. Host software can read this field to determine compatibility to the Intel 82365SL-DF register set. Writing 0010b to this field puts the controller in 82365SL mode. |

## 5.2 ExCA Interface Status Register

The ExCA interface status register provides information on the current status of the PC Card interface. An X in the default bit value indicates that the value of the bit after reset depends on the state of the PC Card interface. See Table 5−3 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | X | X | X | X | X | X |

Register: **ExCA interface status**
Offset: CardBus socket address + 801h; ExCA offset 01h
Type: Read-only
Default: 00XX XXXXb

**Table 5−3. ExCA Interface Status Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | RSVD | R | Reserved. Bit 7 returns 0b when read. |
| 6 | CARDPWR | R | Card Power. Bit 6 indicates the current power status of the PC Card socket. This bit reflects how the ExCA power control register (ExCA offset 02h/42h/802h, see Section 5.3) is programmed. Bit 6 is encoded as:<br>0 = $V_{CC}$ and $V_{PP}$ to the socket turned off (default)<br>1 = $V_{CC}$ and $V_{PP}$ to the socket turned on |
| 5 | READY | R | Ready. Bit 5 indicates the current status of the READY signal at the PC Card interface.<br>0 = PC Card not ready for data transfer<br>1 = PC Card ready for data transfer |
| 4 | CARDWP | R | Card write protect (WP). Bit 4 indicates the current status of WP at the PC Card interface. This signal reports to the controller whether or not the memory card is write protected. Furthermore, write protection for an entire 16-bit memory window is available by setting the appropriate bit in the memory window offset-address high-byte register.<br>0 = WP is 0b. PC Card is read/write.<br>1 = WP is 1b. PC Card is read-only. |
| 3 | CDETECT2 | R | Card detect 2. Bit 3 indicates the status of $\overline{CD2}$ at the PC Card interface. Software may use this and bit 2 (CDETECT1) to determine if a PC Card is fully seated in the socket.<br>0 = $\overline{CD2}$ is 1b. No PC Card is inserted.<br>1 = $\overline{CD2}$ is 0b. PC Card is at least partially inserted. |
| 2 | CDETECT1 | R | Card detect 1. Bit 2 indicates the status of $\overline{CD1}$ at the PC Card interface. Software may use this and bit 3 (CDETECT2) to determine if a PC Card is fully seated in the socket.<br>0 = $\overline{CD1}$ is 1b. No PC Card is inserted.<br>1 = $\overline{CD1}$ is 0b. PC Card is at least partially inserted. |
| 1−0 | BVDSTAT | R | Battery voltage detect. When a 16-bit memory card is inserted, the field indicates the status of the battery voltage detect signals (BVD1, BVD2) at the PC Card interface, where bit 1 reflects the BVD2 status and bit 0 reflects BVD1.<br>00 = Battery dead<br>01 = Battery dead<br>10 = Battery low; warning<br>11 = Battery good<br>When a 16-bit I/O card is inserted, this field indicates the status of $\overline{SPKR}$ (bit 1) and $\overline{STSCHG}$ (bit 0) at the PC Card interface. In this case, the two bits in this field directly reflect the current state of these card outputs. |

## 5.3 ExCA Power Control Register

The ExCA power control register provides PC Card power control. Bit 7 (COE) of this register controls the 16-bit output enables on the socket interface, and can be used for power management in 16-bit PC Card applications. See Table 5−4 and Table 5−5 for a complete description of the register contents.

The controller supports both the 82365SL and 82365SL-DF register models. Bits 3−0 (365REV) of the ExCA identification and revision register (ExCA offset 00h, see Section 5.1) control which register model is supported.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA power control—82365SL support**
Offset: CardBus socket address + 802h; ExCA offset 02h
Type: Read-only, Read/Write
Default: 00h

**Table 5−4. ExCA Power Control Register Description—82365SL Support**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | COE | RW | Card output enable. Bit 7 controls the state of all of the 16-bit outputs on the controller. This bit is encoded as:<br>0 = 16-bit PC Card outputs disabled (default)<br>1 = 16-bit PC Card outputs enabled |
| 6 | RSVD | R | Reserved. Bit 6 returns 0b when read. |
| 5 | AUTOPWRSWEN | RW | Auto power switch enable.<br>0 = Automatic socket power switching based on card detects is disabled<br>1 = Automatic socket power switching based on card detects is enabled |
| 4 | CAPWREN | RW | PC Card power enable.<br>0 = $V_{CC}$ = No connection<br>1 = $V_{CC}$ is enabled and controlled by bit 2 (EXCAPOWER) of the system control register (PCI offset 80h, see Section 4.29) |
| 3−2 | RSVD | R | Reserved. Bits 3 and 2 return 00b when read. |
| 1−0 | EXCAVPP | RW | PC Card $V_{PP}$ power control. Bits 1 and 0 request changes to card $V_{PP}$. The controller ignores this field unless $V_{CC}$ to the socket is enabled. This field is encoded as:<br>00 = No connection (default)<br>01 = $V_{CC}$<br>10 = 12 V<br>11 = Reserved |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA power control—82365SL-DF support**
Offset: CardBus socket address + 802h; ExCA offset 02h
Type: Read-only, Read/Write
Default: 00h

**Table 5−5. ExCA Power Control Register Description—82365SL-DF Support**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | COE | RW | Card output enable. Bit 7 controls the state of all of the 16-bit outputs on the controller. This bit is encoded as:<br>0 = 16-bit PC Card outputs disabled (default)<br>1 = 16-bit PC Card outputs enabled |
| 6−5 | RSVD | R | Reserved. Bits 6 and 5 return 00b when read. |
| 4−3 | EXCAVCC | RW | $V_{CC}$. Bits 4 and 3 request changes to card $V_{CC}$. This field is encoded as:<br>00 = 0 V (default)<br>01 = 0 V reserved<br>10 = 5 V<br>11 = 3.3 V |
| 2 | RSVD | R | Reserved. Bit 2 returns 0b when read. |
| 1−0 | EXCAVPP | RW | $V_{PP}$. Bits 1 and 0 request changes to card $V_{PP}$. The controller ignores this field unless $V_{CC}$ to the socket is enabled. This field is encoded as:<br>00 = No connection (default)<br>01 = $V_{CC}$<br>10 = 12 V<br>11 = Reserved |

## 5.4 ExCA Interrupt and General Control Register

The ExCA interrupt and general control register controls interrupt routing for I/O interrupts, as well as other critical 16-bit PC Card functions. See Table 5−6 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA interrupt and general control**
Offset: CardBus socket address + 803h; ExCA offset 03h
Type: Read/Write
Default: 00h

**Table 5−6. ExCA Interrupt and General Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | RINGEN | RW | Card ring indicate enable. Bit 7 enables the ring indicate function of BVD1/$\overline{\text{RI}}$. This bit is encoded as:<br>0 = Ring indicate disabled (default)<br>1 = Ring indicate enabled |
| 6 | RESET | RW | Card reset. Bit 6 controls the 16-bit PC Card RESET, and allows host software to force a card reset. Bit 6 affects 16-bit cards only. This bit is encoded as:<br>0 = RESET signal asserted (default)<br>1 = RESET signal deasserted |
| 5 | CARDTYPE | RW | Card type. Bit 5 indicates the PC card type. This bit is encoded as:<br>0 = Memory PC Card installed (default)<br>1 = I/O PC Card installed |
| 4 | CSCROUTE | RW | PCI interrupt CSC routing enable bit. When bit 4 is set (high), the card status change interrupts are routed to PCI interrupts. When low, the card status change interrupts are routed using bits 7−4 (CSCSELECT field) in the ExCA card status-change interrupt configuration register (ExCA offset 05h/45h/805h, see Section 5.6). This bit is encoded as:<br>0 = CSC interrupts are routed by ExCA registers (default)<br>1 = CSC interrupts are routed to PCI interrupts |
| 3−0 | INTSELECT | RW | Card interrupt select for I/O PC Card functional interrupts. Bits 3−0 select the interrupt routing for I/O PC Card functional interrupts. This field is encoded as:<br>0000 = No interrupt routing (default). CSC interrupts are routed to PCI interrupts. This bit setting is ORed with bit 4 (CSCROUTE) for backward compatibility.<br>0001 = IRQ1 enabled<br>0010 = SMI enabled<br>0011 = IRQ3 enabled<br>0100 = IRQ4 enabled<br>0101 = IRQ5 enabled<br>0100 = IRQ6 enabled<br>0111 = IRQ7 enabled<br>1000 = IRQ8 enabled<br>1001 = IRQ9 enabled<br>1010 = IRQ10 enabled<br>1011 = IRQ11 enabled<br>1100 = IRQ12 enabled<br>1101 = IRQ13 enabled<br>1110 = IRQ14 enabled<br>1111 = IRQ15 enabled |

## 5.5 ExCA Card Status-Change Register

The ExCA card status-change register controls interrupt routing for I/O interrupts as well as other critical 16-bit PC Card functions. The register enables these interrupt sources to generate an interrupt to the host. When the interrupt source is disabled, the corresponding bit in this register always reads 0b. When an interrupt source is enabled, the corresponding bit in this register is set to indicate that the interrupt source is active. After generating the interrupt to the host, the interrupt service routine must read this register to determine the source of the interrupt. The interrupt service routine is responsible for resetting the bits in this register as well. Resetting a bit is accomplished by one of two methods: a read of this register or an explicit write back of 1b to the status bit. The choice of these two methods is based on bit 2 (interrupt flag clear mode select) in the ExCA global control register (ExCA offset 1E/5E/81E, see Section 5.20). See Table 5−7 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA card status-change**
Offset: CardBus socket address + 804h; ExCA offset 04h
Type: Read-only
Default: 00h

**Table 5−7. ExCA Card Status-Change Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−4 | RSVD | R | Reserved. Bits 7−4 return 0h when read. |
| 3 | CDCHANGE | R | Card detect change. Bit 3 indicates whether a change on $\overline{CD1}$ or $\overline{CD2}$ occurred at the PC Card interface. This bit is encoded as:<br>0 = No change detected on either $\overline{CD1}$ or $\overline{CD2}$<br>1 = Change detected on either $\overline{CD1}$ or $\overline{CD2}$ |
| 2 | READYCHANGE | R | Ready change. When a 16-bit memory is installed in the socket, bit 2 includes whether the source of an interrupt was due to a change on READY at the PC Card interface, indicating that the PC Card is now ready to accept new data. This bit is encoded as:<br>0 = No low-to-high transition detected on READY (default)<br>1 = Detected low-to-high transition on READY<br>When a 16-bit I/O card is installed, bit 2 is always 0b. |
| 1 | BATWARN | R | Battery warning change. When a 16-bit memory card is installed in the socket, bit 1 indicates whether the source of an interrupt was due to a battery-low warning condition. This bit is encoded as:<br>0 = No battery warning condition (default)<br>1 = Detected battery warning condition<br>When a 16-bit I/O card is installed, bit 1 is always 0b. |
| 0 | BATDEAD | R | Battery dead or status change. When a 16-bit memory card is installed in the socket, bit 0 indicates whether the source of an interrupt was due to a battery dead condition. This bit is encoded as:<br>0 = $\overline{STSCHG}$ deasserted (default)<br>1 = $\overline{STSCHG}$ asserted<br>Ring indicate. When the controller is configured for ring indicate operation, bit 0 indicates the status of $\overline{RI}$. |

## 5.6 ExCA Card Status-Change Interrupt Configuration Register

The ExCA card status-change interrupt configuration register controls interrupt routing for card status-change interrupts, as well as masking CSC interrupt sources. See Table 5−8 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA card status-change interrupt configuration**
Offset: CardBus socket address + 805h; ExCA offset 05h
Type: Read/Write
Default: 00h

**Table 5−8. ExCA Card Status-Change Interrupt Configuration Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−4 | CSCSELECT | RW | Interrupt select for card status change. Bits 7−4 select the interrupt routing for card status-change interrupts.<br>0000 = CSC interrupts routed to PCI interrupts if bit 5 (CSC) of the diagnostic register (PCI offset 93h, see Section 4.34) is set to 1b. In this case bit 4 (CSCROUTE) of the ExCA interrupt and general control register (ExCA offset 03h/43h/803h, see Section 5.4) is a don't care. This is the default setting.<br>0000 = No ISA interrupt routing if bit 5 (CSC) of the diagnostic register is set to 0b (see Section 4.34). In this case, CSC interrupts are routed to PCI interrupts by setting bit 4 (CSCROUTE) of the ExCA interrupt and general control register (ExCA offset 03h/43h/803h, see Section 5.4) to 1b.<br>This field is encoded as:<br>0000 = No interrupt routing (default) 1000 = IRQ8 enabled<br>0001 = IRQ1 enabled 1001 = IRQ9 enabled<br>0010 = SMI enabled 1010 = IRQ10 enabled<br>0011 = IRQ3 enabled 1011 = IRQ11 enabled<br>0100 = IRQ4 enabled 1100 = IRQ12 enabled<br>0101 = IRQ5 enabled 1101 = IRQ13 enabled<br>0110 = IRQ6 enabled 1110 = IRQ14 enabled<br>0111 = IRQ7 enabled 1111 = IRQ15 enabled |
| 3 | CDEN | RW | Card detect enable. Bit 3 enables interrupts on $\overline{CD1}$ or $\overline{CD2}$ changes. This bit is encoded as:<br>0 = Disables interrupts on $\overline{CD1}$ or $\overline{CD2}$ line changes (default)<br>1 = Enables interrupts on $\overline{CD1}$ or $\overline{CD2}$ line changes |
| 2 | READYEN | RW | Ready enable. Bit 2 enables/disables a low-to-high transition on PC Card READY to generate a host interrupt. This interrupt source is considered a card status change. This bit is encoded as:<br>0 = Disables host interrupt generation (default)<br>1 = Enables host interrupt generation |
| 1 | BATWARNEN | RW | Battery warning enable. Bit 1 enables/disables a battery warning condition to generate a CSC interrupt. This bit is encoded as:<br>0 = Disables host interrupt generation (default)<br>1 = Enables host interrupt generation |
| 0 | BATDEADEN | RW | Battery dead enable. Bit 0 enables/disables a battery dead condition on a memory PC Card or assertion of the $\overline{STSCHG}$ I/O PC Card signal to generate a CSC interrupt.<br>0 = Disables host interrupt generation (default)<br>1 = Enables host interrupt generation |

## 5.7 ExCA Address Window Enable Register

The ExCA address window enable register enables/disables the memory and I/O windows to the 16-bit PC Card. By default, all windows to the card are disabled. The controller does not acknowledge PCI memory or I/O cycles to the card if the corresponding enable bit in this register is 0b, regardless of the programming of the memory or I/O window start/end/offset address registers. See Table 5−9 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA address window enable**
Offset: CardBus socket address + 806h; ExCA offset 06h
Type: Read-only, Read/Write
Default: 00h

**Table 5−9. ExCA Address Window Enable Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | IOWIN1EN | RW | I/O window 1 enable. Bit 7 enables/disables I/O window 1 for the PC Card. This bit is encoded as:<br>0 = I/O window 1 disabled (default)<br>1 = I/O window 1 enabled |
| 6 | IOWIN0EN | RW | I/O window 0 enable. Bit 6 enables/disables I/O window 0 for the PC Card. This bit is encoded as:<br>0 = I/O window 0 disabled (default)<br>1 = I/O window 0 enabled |
| 5 | RSVD | R | Reserved. Bit 5 returns 0b when read. |
| 4 | MEMWIN4EN | RW | Memory window 4 enable. Bit 4 enables/disables memory window 4 for the PC Card. This bit is encoded as:<br>0 = Memory window 4 disabled (default)<br>1 = Memory window 4 enabled |
| 3 | MEMWIN3EN | RW | Memory window 3 enable. Bit 3 enables/disables memory window 3 for the PC Card. This bit is encoded as:<br>0 = Memory window 3 disabled (default)<br>1 = Memory window 3 enabled |
| 2 | MEMWIN2EN | RW | Memory window 2 enable. Bit 2 enables/disables memory window 2 for the PC Card. This bit is encoded as:<br>0 = Memory window 2 disabled (default)<br>1 = Memory window 2 enabled |
| 1 | MEMWIN1EN | RW | Memory window 1 enable. Bit 1 enables/disables memory window 1 for the PC Card. This bit is encoded as:<br>0 = Memory window 1 disabled (default)<br>1 = Memory window 1 enabled |
| 0 | MEMWIN0EN | RW | Memory window 0 enable. Bit 0 enables/disables memory window 0 for the PC Card. This bit is encoded as:<br>0 = Memory window 0 disabled (default)<br>1 = Memory window 0 enabled |

## 5.8 ExCA I/O Window Control Register

The ExCA I/O window control register contains parameters related to I/O window sizing and cycle timing. See Table 5−10 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window control**
Offset: CardBus socket address + 807h; ExCA offset 07h
Type: Read/Write
Default: 00h

**Table 5−10. ExCA I/O Window Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | WAITSTATE1 | RW | I/O window 1 wait state. Bit 7 controls the I/O window 1 wait state for 16-bit I/O accesses. Bit 7 has no effect on 8-bit accesses. This wait-state timing emulates the ISA wait state used by the Intel 82365SL-DF. This bit is encoded as:<br>0 = 16-bit cycles have standard length (default)<br>1 = 16-bit cycles are extended by one equivalent ISA wait state |
| 6 | ZEROWS1 | RW | I/O window 1 zero wait state. Bit 6 controls the I/O window 1 wait state for 8-bit I/O accesses. Bit 6 has no effect on 16-bit accesses. This wait-state timing emulates the ISA wait state used by the Intel 82365SL-DF. This bit is encoded as:<br>0 = 8-bit cycles have standard length (default)<br>1 = 8-bit cycles are reduced to equivalent of three ISA cycles |
| 5 | IOIS16W1 | RW | I/O window 1 $\overline{\text{IOIS16}}$ source. Bit 5 controls the I/O window 1 automatic data sizing feature that uses $\overline{\text{IOIS16}}$ from the PC Card to determine the data width of the I/O data transfer. This bit is encoded as:<br>0 = Window data width determined by $\overline{\text{DATASIZE1}}$, bit 4 (default)<br>1 = Window data width determined by $\overline{\text{IOIS16}}$ |
| 4 | DATASIZE1 | RW | I/O window 1 data size. Bit 4 controls the I/O window 1 data size. Bit 4 is ignored if bit 5 (IOSIS16W1) is set. This bit is encoded as:<br>0 = Window data width is 8 bits (default)<br>1 = Window data width is 16 bits |
| 3 | WAITSTATE0 | RW | I/O window 0 wait state. Bit 3 controls the I/O window 0 wait state for 16-bit I/O accesses. Bit 3 has no effect on 8-bit accesses. This wait-state timing emulates the ISA wait state used by the Intel 82365SL-DF. This bit is encoded as:<br>0 = 16-bit cycles have standard length (default)<br>1 = 16-bit cycles are extended by one equivalent ISA wait state |
| 2 | ZEROWS0 | RW | I/O window 0 zero wait state. Bit 2 controls the I/O window 0 wait state for 8-bit I/O accesses. Bit 2 has no effect on 16-bit accesses. This wait-state timing emulates the ISA wait state used by the Intel 82365SL-DF. This bit is encoded as:<br>0 = 8-bit cycles have standard length (default)<br>1 = 8-bit cycles are reduced to equivalent of three ISA cycles |
| 1 | IOIS16W0 | RW | I/O window 0 $\overline{\text{IOIS16}}$ source. Bit 1 controls the I/O window 0 automatic data sizing feature that uses $\overline{\text{IOIS16}}$ from the PC Card to determine the data width of the I/O data transfer. This bit is encoded as:<br>0 = Window data width is determined by $\overline{\text{DATASIZE0}}$, bit 0 (default)<br>1 = Window data width is determined by $\overline{\text{IOIS16}}$ |
| 0 | DATASIZE0 | RW | I/O window 0 data size. Bit 0 controls the I/O window 0 data size. Bit 0 is ignored if bit 1 (IOSIS16W0) is set. This bit is encoded as:<br>0 = Window data width is 8 bits (default)<br>1 = Window data width is 16 bits |

## 5.9 ExCA I/O Windows 0 and 1 Start-Address Low-Byte Registers

These registers contain the low byte of the 16-bit I/O window start address for I/O windows 0 and 1. The 8 bits of these registers correspond to the lower 8 bits of the start address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window 0 start-address low-byte**
Offset: CardBus socket address + 808h; ExCA offset 08h
Register: **ExCA I/O window 1 start-address low-byte**
Offset: CardBus socket address + 80Ch; ExCA offset 0Ch
Type: Read/Write
Default: 00h

## 5.10 ExCA I/O Windows 0 and 1 Start-Address High-Byte Registers

These registers contain the high byte of the 16-bit I/O window start address for I/O windows 0 and 1. The 8 bits of these registers correspond to the upper 8 bits of the end address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window 0 start-address high-byte**
Offset: CardBus socket address + 809h; ExCA offset 09h
Register: **ExCA I/O window 1 start-address high-byte**
Offset: CardBus socket address + 80Dh; ExCA offset 0Dh
Type: Read/write
Default: 00h

## 5.11 ExCA I/O Windows 0 and 1 End-Address Low-Byte Registers

These registers contain the low byte of the 16-bit I/O window end address for I/O windows 0 and 1. The 8 bits of these registers correspond to the lower 8 bits of the end address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window 0 end-address low-byte**
Offset: CardBus socket address + 80Ah; ExCA offset 0Ah
Register: **ExCA I/O window 1 end-address low-byte**
Offset: CardBus socket address + 80Eh; ExCA offset 0Eh
Type: Read/Write
Default: 00h

## 5.12 ExCA I/O Windows 0 and 1 End-Address High-Byte Registers

These registers contain the high byte of the 16-bit I/O window end address for I/O windows 0 and 1. The 8 bits of these registers correspond to the upper 8 bits of the end address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window 0 end-address high-byte**
Offset: CardBus socket address + 80Bh; ExCA offset 0Bh
Register: **ExCA I/O window 1 end-address high-byte**
Offset: CardBus socket address + 80Fh; ExCA offset 0Fh
Type: Read/write
Default: 00h

## 5.13 ExCA Memory Windows 0−4 Start-Address Low-Byte Registers

These registers contain the low byte of the 16-bit memory window start address for memory windows 0, 1, 2, 3, and 4. The 8 bits of these registers correspond to bits A19−A12 of the start address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory window 0 start-address low-byte**
Offset: CardBus socket address + 810h; ExCA offset 10h
Register: **ExCA memory window 1 start-address low-byte**
Offset: CardBus socket address + 818h; ExCA offset 18h
Register: **ExCA memory window 2 start-address low-byte**
Offset: CardBus socket address + 820h; ExCA offset 20h
Register: **ExCA memory window 3 start-address low-byte**
Offset: CardBus socket address + 828h; ExCA offset 28h
Register: **ExCA memory window 4 start-address low-byte**
Offset: CardBus socket address + 830h; ExCA offset 30h
Type: Read/Write
Default: 00h

## 5.14 ExCA Memory Windows 0−4 Start-Address High-Byte Registers

These registers contain the high nibble of the 16-bit memory window start address for memory windows 0, 1, 2, 3, and 4. The lower 4 bits of these registers correspond to bits A23−A20 of the start address. In addition, the memory window data width and wait states are set in this register. See Table 5−11 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory window 0 start-address high-byte**
Offset: CardBus socket address + 811h; ExCA offset 11h
Register: **ExCA memory window 1 start-address high-byte**
Offset: CardBus socket address + 819h; ExCA offset 19h
Register: **ExCA memory window 2 start-address high-byte**
Offset: CardBus socket address + 821h; ExCA offset 21h
Register: **ExCA memory window 3 start-address high-byte**
Offset: CardBus socket address + 829h; ExCA offset 29h
Register: **ExCA memory window 4 start-address high-byte**
Offset: CardBus socket address + 831h; ExCA offset 31h
Type: Read/Write
Default: 00h

**Table 5−11. ExCA Memory Windows 0−4 Start-Address High-Byte Registers Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | DATASIZE | RW | Data size. Bit 7 controls the memory window data width. This bit is encoded as:<br>0 = Window data width is 8 bits (default)<br>1 = Window data width is 16 bits |
| 6 | ZEROWAIT | RW | Zero wait state. Bit 6 controls the memory window wait state for 8- and 16-bit accesses. This wait-state timing emulates the ISA wait state used by the Intel 82365SL-DF. This bit is encoded as:<br>0 = 8- and 16-bit cycles have standard length (default)<br>1 = 8-bit cycles are reduced to equivalent of three ISA cycles.<br>16-bit cycles are reduced to equivalent of two ISA cycles. |
| 5−4 | SCRATCH | RW | Scratch pad bits. Bits 5 and 4 have no effect on memory window operation. |
| 3−0 | STAHN | RW | Start-address high nibble. Bits 3−0 represent the upper address bits A23−A20 of the memory window start address. |

## 5.15 ExCA Memory Windows 0−4 End-Address Low-Byte Registers

These registers contain the low byte of the 16-bit memory window end address for memory windows 0, 1, 2, 3, and 4. The 8 bits of these registers correspond to bits A19−A12 of the end address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory window 0 end-address low-byte**
Offset: CardBus socket address + 812h; ExCA offset 12h
Register: **ExCA memory window 1 end-address low-byte**
Offset: CardBus socket address + 81Ah; ExCA offset 1Ah
Register: **ExCA memory window 2 end-address low-byte**
Offset: CardBus socket address + 822h; ExCA offset 22h
Register: **ExCA memory window 3 end-address low-byte**
Offset: CardBus socket address + 82Ah; ExCA offset 2Ah
Register: **ExCA memory window 4 end-address low-byte**
Offset: CardBus socket address + 832h; ExCA offset 32h
Type: Read/Write
Default: 00h

## 5.16 ExCA Memory Windows 0−4 End-Address High-Byte Registers

These registers contain the high nibble of the 16-bit memory window end address for memory windows 0, 1, 2, 3, and 4. The lower 4 bits of these registers correspond to bits A23−A20 of the end address. In addition, the memory window wait states are set in this register. See Table 5−12 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory window 0 end-address high-byte**
Offset: CardBus socket address + 813h; ExCA offset 13h
Register: **ExCA memory window 1 end-address high-byte**
Offset: CardBus socket address + 81Bh; ExCA offset 1Bh
Register: **ExCA memory window 2 end-address high-byte**
Offset: CardBus socket address + 823h; ExCA offset 23h
Register: **ExCA memory window 3 end-address high-byte**
Offset: CardBus socket address + 82Bh; ExCA offset 2Bh
Register: **ExCA memory window 4 end-address high-byte**
Offset: CardBus socket address + 833h; ExCA offset 33h
Type: Read-only, Read/Write
Default: 00h

**Table 5−12. ExCA Memory Windows 0−4 End-Address High-Byte Registers Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−6 | MEMWS | RW | Wait state. Bits 7 and 6 specify the number of equivalent ISA wait states to be added to 16-bit memory accesses. The number of wait states added is equal to the binary value of these two bits. |
| 5−4 | RSVD | R | Reserved. Bits 5 and 4 return 00b when read. |
| 3−0 | ENDHN | RW | End-address high nibble. Bits 3−0 represent the upper address bits A23−A20 of the memory window end address. |

## 5.17 ExCA Memory Windows 0−4 Offset-Address Low-Byte Registers

These registers contain the low byte of the 16-bit memory window offset address for memory windows 0, 1, 2, 3, and 4. The 8 bits of these registers correspond to bits A19−A12 of the offset address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory window 0 offset-address low-byte**
Offset: CardBus socket address + 814h; ExCA offset 14h
Register: **ExCA memory window 1 offset-address low-byte**
Offset: CardBus socket address + 81Ch; ExCA offset 1Ch
Register: **ExCA memory window 2 offset-address low-byte**
Offset: CardBus socket address + 824h; ExCA offset 24h
Register: **ExCA memory window 3 offset-address low-byte**
Offset: CardBus socket address + 82Ch; ExCA offset 2Ch
Register: **ExCA memory window 4 offset-address low-byte**
Offset: CardBus socket address + 834h; ExCA offset 34h
Type: Read/Write
Default: 00h

## 5.18 ExCA Memory Windows 0−4 Offset-Address High-Byte Registers

These registers contain the high 6 bits of the 16-bit memory window offset address for memory windows 0, 1, 2, 3, and 4. The lower 6 bits of these registers correspond to bits A25−A20 of the offset address. In addition, the write protection and common/attribute memory configurations are set in this register. See Table 5−13 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory window 0 offset-address high-byte**
Offset: CardBus socket address + 815h; ExCA offset 15h
Register: **ExCA memory window 1 offset-address high-byte**
Offset: CardBus socket address + 81Dh; A ExCA offset 1Dh
Register: **ExCA memory window 2 offset-address high-byte**
Offset: CardBus socket address + 825h; ExCA offset 25h
Register: **ExCA memory window 3 offset-address high-byte**
Offset: CardBus socket address + 82Dh; ExCA offset 2Dh
Register: **ExCA memory window 4 offset-address high-byte**
Offset: CardBus socket address + 835h; ExCA offset 35h
Type: Read/Write
Default: 00h

**Table 5−13. ExCA Memory Windows 0−4 Offset-Address High-Byte Registers Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | WINWP | RW | Write protect. Bit 7 specifies whether write operations to this memory window are enabled. This bit is encoded as:<br>0 = Write operations are allowed (default)<br>1 = Write operations are not allowed |
| 6 | REG | RW | Bit 6 specifies whether this memory window is mapped to card attribute or common memory. This bit is encoded as:<br>0 = Memory window is mapped to common memory (default)<br>1 = Memory window is mapped to attribute memory |
| 5−0 | OFFHB | RW | Offset-address high byte. Bits 5−0 represent the upper address bits A25−A20 of the memory window offset address. |

## 5.19 ExCA Card Detect and General Control Register

The ExCA card detect and general control register controls how the ExCA registers for the socket respond to card removal, as well as reports the status of $\overline{VS1}$ and $\overline{VS2}$ at the PC Card interface. See Table 5−14 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | X | X | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA card detect and general control**
Offset: CardBus socket address + 816h; ExCA offset 16h
Type: Read-only, Read/Write
Default: XX00 0000b

**Table 5−14. ExCA Card Detect and General Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | VS2STAT | R | $\overline{VS2}$ state. Bit 7 reports the current state of $\overline{VS2}$ at the PC Card interface and, therefore, does not have a default value.<br>0 = $\overline{VS2}$ low<br>1 = $\overline{VS2}$ high |
| 6 | VS1STAT | R | $\overline{VS1}$ state. Bit 6 reports the current state of $\overline{VS1}$ at the PC Card interface and, therefore, does not have a default value.<br>0 = $\overline{VS1}$ low<br>1 = $\overline{VS1}$ high |
| 5 | SWCSC | RW | Software card detect interrupt. If bit 3 (CDEN) in the ExCA card status-change interrupt configuration register (ExCA offset 05h/45h/805, see Section 5.6) is set, then writing a 1b to bit 5 causes a card-detect card-status change interrupt for the associated card socket. If bit 3 (CDEN) in the ExCA card status-change-interrupt configuration register (ExCA offset 05h/45h/805, see Section 5.6) is cleared to 0b, then writing a 1b to bit 5 has no effect. A read operation of this bit always returns 0b. |
| 4 | CDRESUME | RW | Card detect resume enable. If bit 4 is set to 1b, then once a card detect change has been detected on $\overline{CD1}$ and $\overline{CD2}$ inputs, $\overline{RI_OUT}$ goes from high to low. $\overline{RI_OUT}$ remains low until bit 0 (card status change) in the ExCA card status-change register is cleared (see Section 5.5). If this bit is a 0b, then the card detect resume functionality is disabled.<br>0 = Card detect resume disabled (default)<br>1 = Card detect resume enabled |
| 3−2 | RSVD | R | Reserved. Bits 3 and 2 return 00b when read. |
| 1 | REGCONFIG | RW | Register configuration on card removal. Bit 1 controls how the ExCA registers for the socket react to a card removal event. This bit is encoded as:<br>0 = No change to ExCA registers on card removal (default)<br>1 = Reset ExCA registers on card removal |
| 0 | RSVD | R | Reserved. Bit 0 returns 0b when read. |

## 5.20 ExCA Global Control Register

The host interrupt mode bits in this register are retained for Intel 82365SL-DF compatibility. See Table 5−15 for a complete description of the register contents.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA global control**
Offset: CardBus socket address + 81Eh; ExCA offset 1Eh
Type: Read-only, Read/Write
Default: 00h

**Table 5−15. ExCA Global Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7−5 | RSVD | R | Reserved. Bits 7−5 return 000b when read. |
| 4 | INTMODEB | RW | Level/edge interrupt mode select − card B. Bit 4 selects the signaling mode for the controller host interrupt for card B interrupts. This bit is encoded as:<br>0 = Host interrupt is edge mode (default)<br>1 = Host interrupt is level mode |
| 3 | INTMODEA | RW | Level/edge interrupt mode select − card A. Bit 3 selects the signaling mode for the controller host interrupt for card A interrupts. This bit is encoded as:<br>0 = Host interrupt is edge mode (default)<br>1 = Host interrupt is level mode |
| 2 | IFCMODE | RW | Interrupt flag clear mode select. Bit 2 selects the interrupt flag clear mechanism for the flags in the ExCA card status change register (ExCA offset 04h/44h/804h, see Section 5.5). This bit is encoded as:<br>0 = Interrupt flags are cleared by read of CSC register (default)<br>1 = Interrupt flags are cleared by explicit writeback of 1b |
| 1 | CSCMODE | RW | Card status change level/edge mode select. Bit 1 selects the signaling mode for the controller host interrupt for card status changes. This bit is encoded as:<br>0 = Host interrupt is edge mode (default)<br>1 = Host interrupt is level mode |
| 0 | PWRDWN | RW | Power-down mode select. When bit 0 is set to 1b, the controller is in power-down mode. In power-down mode, the controller card outputs are high-impedance until an active cycle is executed on the card interface. Following an active cycle, the outputs are again high-impedance. The controller still receives functional interrupts and/or card status-change interrupts; however, an actual card access is required to wake up the interface. This bit is encoded as:<br>0 = Power-down mode is disabled (default)<br>1 = Power-down mode is enabled |

## 5.21 ExCA I/O Windows 0 and 1 Offset-Address Low-Byte Registers

These registers contain the low byte of the 16-bit I/O window offset address for I/O windows 0 and 1. The 8 bits of these registers correspond to the lower 8 bits of the offset address, and bit 0 is always 0b.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window 0 offset-address low-byte**
Offset: CardBus socket address + 836h; ExCA offset 36h
Register: **ExCA I/O window 1 offset-address low-byte**
Offset: CardBus socket address + 838h; ExCA offset 38h
Type: Read-only, Read/Write
Default: 00h

## 5.22 ExCA I/O Windows 0 and 1 Offset-Address High-Byte Registers

These registers contain the high byte of the 16-bit I/O window offset address for I/O windows 0 and 1. The 8 bits of these registers correspond to the upper 8 bits of the offset address.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window 0 offset-address high-byte**
Offset: CardBus socket address + 837h; ExCA offset 37h
Register: **ExCA I/O window 1 offset-address high-byte**
Offset: CardBus socket address + 839h; ExCA offset 39h
Type: Read/Write
Default: 00h

## 5.23 ExCA Memory Windows 0−4 Page Registers

The upper 8 bits of a 4-byte PCI memory address are compared to the contents of this register when decoding addresses for 16-bit memory windows. Each window has its own page register, all of which default to 00h. By programming this register to a nonzero value, host software can locate 16-bit memory windows in any 1 of 256 16-Mbyte regions in the 4-Gbyte PCI address space. These registers are only accessible when the ExCA registers are memory-mapped; that is, these registers cannot be accessed using the index/data I/O scheme.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory windows 0−4 page**
Offset: CardBus socket address + 840h, 841h, 842h, 843h, 844h
Type: Read/Write
Default: 00h

# 6 CardBus Socket Registers

The *PC Card Standard* requires a CardBus socket controller to provide five 32-bit registers that report and control socket-specific functions. The PCI1510 controller provides the CardBus socket/ExCA base-address register (PCI offset 10h, see Section 4.12) to locate these CardBus socket registers in PCI memory address space. Table 6−1 gives the location of the socket registers in relation to the CardBus socket/ExCA base address.

The controller implements an additional register at offset 20h that provides power management control for the socket.

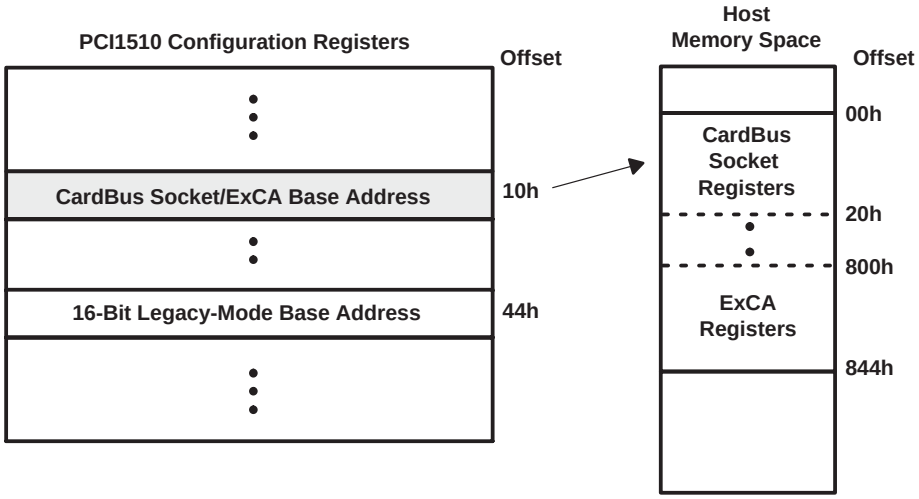



**Figure 6−1. Accessing CardBus Socket Registers Through PCI Memory**

**Table 6−1. CardBus Socket Registers**

| REGISTER NAME | OFFSET |
|---|---|
| Socket event | 00h |
| Socket mask | 04h |
| Socket present-state | 08h |
| Socket force event | 0Ch |
| Socket control | 10h |
| Reserved | 14h−1Ch |
| Socket power-management | 20h |

A bit description table, typically included when a register contains bits of more than one type or purpose, indicates bit field names, which appear in the signal column; a detailed field description, which appears in the function column; and field access tags, which appear in the type column of the bit description table. Table 4−2 describes the field access tags.

## 6.1 Socket Event Register

The socket event register indicates a change in socket status has occurred. These bits do not indicate what the change is, only that one has occurred. Software must read the socket present-state register (CB offset 08h, see Section 6.3) for current status. Each bit in this register can be cleared by writing a 1b to that bit. The bits in this register can be set to a 1b by software by writing a 1b to the corresponding bit in the socket force event register (CB offset 0Ch, see Section 6.4). All bits in this register are cleared by PCI reset. They can be immediately set again, if, when coming out of PC Card reset, the bridge finds the status unchanged (that is, $\overline{\text{CSTSCHG}}$ reasserted or card detect is still true). Software must clear this register before enabling interrupts. If it is not cleared when interrupts are enabled, then an interrupt is generated (but not masked) based on any bit set. See Table 6−2 for a complete description of the register contents.

| **Bit** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Default** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Default** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket event**
Offset: CardBus socket address + 00h
Type: Read-only, Read/Write, Read/Clear
Default: 0000 0000h

**Table 6−2. Socket Event Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31−4 | RSVD | R | Reserved. Bits 31−4 return 000 0000h when read. |
| 3 | PWREVENT | R/C | Power cycle. Bit 3 is set when the controller detects that bit 3 (PWRCYCLE) in the socket present-state register (CB offset 08h, see Section 6.3) has changed state. This bit is cleared by writing a 1b. |
| 2 | CD2EVENT | R/C | $\overline{\text{CCD2}}$. Bit 2 is set when the controller detects that bit 2 (CDETECT2) in the socket present-state register (CB offset 08h, see Section 6.3) has changed state. This bit is cleared by writing a 1b. |
| 1 | CD1EVENT | R/C | $\overline{\text{CCD1}}$. Bit 1 is set when the controller detects that bit 1 (CDETECT1) in the socket present-state register (CB offset 08h, see Section 6.3) has changed state. This bit is cleared by writing a 1b. |
| 0 | CSTSEVENT | R/C | $\overline{\text{CSTSCHG}}$. Bit 0 is set when bit 0 (CARDSTS) in the socket present-state register (CB offset 08h, see Section 6.3) has changed state. For CardBus cards, bit 0 is set on the rising edge of $\overline{\text{CSTSCHG}}$. For 16-bit PC Cards, bit 0 is set on both transitions of $\overline{\text{CSTSCHG}}$. This bit is reset by writing a 1b. |

## 6.2 Socket Mask Register

The socket mask register allows software to control the CardBus card events that generate a status change interrupt. The state of these mask bits does not prevent the corresponding bits from reacting in the socket event register (CB offset 00h, see Section 6.1). See Table 6−3 for a complete description of the register contents.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket mask**
Offset: CardBus socket address + 04h
Type: Read-only, Read/Write
Default: 0000 0000h

**Table 6−3. Socket Mask Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31−4 | RSVD | R | Reserved. Bits 31−4 return 000 0000b when read. |
| 3 | PWRMASK | RW | Power cycle. Bit 3 masks bit 3 (PWRCYCLE) in the socket present-state register (CB offset 08h, see Section 6.3) from causing a status change interrupt.<br>0 = PWRCYCLE event does not cause CSC interrupt (default)<br>1 = PWRCYCLE event causes CSC interrupt |
| 2−1 | CDMASK | RW | Card detect mask. Bits 2 and 1 mask bits 1 and 2 (CDETECT1 and CDETECT2) in the socket present-state register (CB offset 08h, see Section 6.3) from causing a CSC interrupt.<br>00 = Insertion/removal does not cause CSC interrupt (default)<br>01 = Reserved (undefined)<br>10 = Reserved (undefined)<br>11 = Insertion/removal causes CSC interrupt |
| 0 | CSTSMASK | RW | $\overline{\text{CSTSCHG}}$ mask. Bit 0 masks bit 0 (CARDSTS) in the socket present-state register (CB offset 08h, see Section 6.3) from causing a CSC interrupt.<br>0 = CARDSTS event does not cause CSC interrupt (default)<br>1 = CARDSTS event causes CSC interrupt |

## 6.3 Socket Present-State Register

The socket present-state register reports information about the socket interface. Write transactions to the socket force event register (CB offset 0Ch, see Section 6.4) are reflected here, as well as general socket interface status. Information about PC Card $V_{CC}$ support and card type is only updated at each insertion. Also note that the controller uses $\overline{CCD1}$ and $\overline{CCD2}$ during card identification, and changes on these signals during this operation are not reflected in this register. See Table 6−4 for a complete description of the register contents.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | X | 0 | 0 | 0 | X | X | X |

Register: **Socket present-state**
Offset: CardBus socket address + 08h
Type: Read-only
Default: 3000 00XXh

**Table 6−4. Socket Present-State Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31 | YVSOCKET | R | YV socket. Bit 31 indicates whether or not the socket can supply $V_{CC}$ = Y.Y V to PC Cards. The controller does not support Y.Y-V $V_{CC}$; therefore, this bit is always reset unless overridden by the socket force event register (CB offset 0Ch, see Section 6.4). This bit is hardwired to 0b. |
| 30 | XVSOCKET | R | XV socket. Bit 30 indicates whether or not the socket can supply $V_{CC}$ = X.X V to PC Cards. The controller does not support X.X-V $V_{CC}$; therefore, this bit is always reset unless overridden by the socket force event register (CB offset 0Ch, see Section 6.4). This bit is hardwired to 0b. |
| 29 | 3VSOCKET | R | 3-V socket. Bit 29 indicates whether or not the socket can supply $V_{CC}$ = 3.3 V to PC Cards. The controller does support 3.3-V $V_{CC}$; therefore, this bit is always set unless overridden by the socket force event register (CB offset 0Ch, see Section 6.4). |
| 28 | 5VSOCKET | R | 5-V socket. Bit 28 indicates whether or not the socket can supply $V_{CC}$ = 5 V to PC Cards. The controller does support 5-V $V_{CC}$; therefore, this bit is always set unless overridden by the socket force event register (CB offset 0Ch, see Section 6.4). |
| 27 | ZVSUPPORT | R | Zoomed-video support. This bit indicates whether or not the socket has support for zoomed video.<br>0 = Zoomed video is not supported<br>1 = Zoomed video is supported |
| 26−14 | RSVD | R | Reserved. Bits 26−14 return 0s when read. |
| 13 | YVCARD | R | YV card. Bit 13 indicates whether or not the PC Card inserted in the socket supports $V_{CC}$ = Y.Y V.<br>0 = Y.Y-V $V_{CC}$ is not supported<br>1 = Y.Y-V $V_{CC}$ is supported |
| 12 | XVCARD | R | XV card. Bit 12 indicates whether or not the PC Card inserted in the socket supports $V_{CC}$ = X.X V.<br>0 = X.X-V $V_{CC}$ is not supported<br>1 = X.X-V $V_{CC}$ is supported |
| 11 | 3VCARD | R | 3-V card. Bit 11 indicates whether or not the PC Card inserted in the socket supports $V_{CC}$ = 3.3 V.<br>0 = 3.3-V $V_{CC}$ is not supported<br>1 = 3.3-V $V_{CC}$ is supported |

**Table 6−4. Socket Present-State Register (Continued)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 10 | 5VCARD | R | 5-V card. Bit 10 indicates whether or not the PC Card inserted in the socket supports $V_{CC}$ = 5 V.<br>0 = 5-V $V_{CC}$ is not supported.<br>1 = 5-V $V_{CC}$ is supported. |
| 9 | BADVCCREQ | R | Bad $V_{CC}$ request. Bit 9 indicates that the host software has requested that the socket be powered at an invalid voltage.<br>0 = Normal operation (default)<br>1 = Invalid $V_{CC}$ request by host software |
| 8 | DATALOST | R | Data lost. Bit 8 indicates that a PC Card removal event may have caused lost data because the cycle did not terminate properly or because write data still resides in the controller.<br>0 = Normal operation (default)<br>1 = Potential data loss due to card removal |
| 7 | NOTACARD | R | Not a card. Bit 7 indicates that an unrecognizable PC Card has been inserted in the socket. This bit is not updated until a valid PC Card is inserted into the socket.<br>0 = Normal operation (default)<br>1 = Unrecognizable PC Card detected |
| 6 | IREQCINT | R | READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$. Bit 6 indicates the current status of READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$ at the PC Card interface.<br>0 = READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$ low<br>1 = READY($\overline{\text{IREQ}}$)//$\overline{\text{CINT}}$ high |
| 5 | CBCARD | R | CardBus card detected. Bit 5 indicates that a CardBus PC Card is inserted in the socket. This bit is not updated until another card interrogation sequence occurs (card insertion). |
| 4 | 16BITCARD | R | 16-bit card detected. Bit 4 indicates that a 16-bit PC Card is inserted in the socket. This bit is not updated until another card interrogation sequence occurs (card insertion). |
| 3 | PWRCYCLE | R | Power cycle. Bit 3 indicates the status of each card powering request. This bit is encoded as:<br>0 = Socket powered down (default)<br>1 = Socket powered up |
| 2 | CDETECT2 | R | $\overline{\text{CCD2}}$. Bit 2 reflects the current status of $\overline{\text{CCD2}}$ at the PC Card interface. Changes to this signal during card interrogation are not reflected here.<br>0 = $\overline{\text{CCD2}}$ low (PC Card may be present)<br>1 = $\overline{\text{CCD2}}$ high (PC Card not present) |
| 1 | CDETECT1 | R | $\overline{\text{CCD1}}$. Bit 1 reflects the current status of $\overline{\text{CCD1}}$ at the PC Card interface. Changes to this signal during card interrogation are not reflected here.<br>0 = $\overline{\text{CCD1}}$ low (PC Card may be present)<br>1 = $\overline{\text{CCD1}}$ high (PC Card not present) |
| 0 | CARDSTS | R | $\overline{\text{CSTSCHG}}$. Bit 0 reflects the current status of $\overline{\text{CSTSCHG}}$ at the PC Card interface.<br>0 = $\overline{\text{CSTSCHG}}$ low<br>1 = $\overline{\text{CSTSCHG}}$ high |

## 6.4 Socket Force Event Register

The socket force event register is used to force changes to the socket event register (CB offset 00h, see Section 6.1) and the socket present-state register (see Section 6.3). Bit 14 (CVSTEST) in this register must be written when forcing changes that require card interrogation. See Table 6−5 for a complete description of the register contents.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket force event**
Offset: CardBus socket address + 0Ch
Type: Read-only, Write-only
Default: 0000 0000h

**Table 6−5. Socket Force Event Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31−28 | RSVD | R | Reserved. Bits 31−28 return 0h when read. |
| 27 | FZVSUPPORT | W | Zoomed-video support. This bit indicates whether or not the socket has support for zoomed video. |
| 26−15 | RSVD | R | Reserved. Bits 26−15 return 000h when read. |
| 14 | CVSTEST | W | Card VS test. When bit 14 is set, the controller re-interrogates the PC Card, updates the socket present-state register (CB offset 08h, see Section 6.3), and enables the socket control register (CB offset 10h, see Section 6.5). |
| 13 | FYVCARD | W | Force YV card. Write transactions to bit 13 cause bit 13 (YVCARD) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. When set, this bit disables the socket control register (CB offset 10h, see Section 6.5). |
| 12 | FXVCARD | W | Force XV card. Write transactions to bit 12 cause bit 12 (XVCARD) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. When set, this bit disables the socket control register (CB offset 10h, see Section 6.5). |
| 11 | F3VCARD | W | Force 3-V card. Write transactions to bit 11 cause bit 11 (3VCARD) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. When set, this bit disables the socket control register (CB offset 10h, see Section 6.5). |
| 10 | F5VCARD | W | Force 5-V card. Write transactions to bit 10 cause bit 10 (5VCARD) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. When set, this bit disables the socket control register (CB offset 10h, see Section 6.5). |
| 9 | FBADVCCREQ | W | Force bad $V_{CC}$ request. Changes to bit 9 (BADVCCREQ) in the socket present-state register (CB offset 08h, see Section 6.3) can be made by writing to bit 9. |
| 8 | FDATALOST | W | Force data lost. Write transactions to bit 8 cause bit 8 (DATALOST) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. |
| 7 | FNOTACARD | W | Force not-a-card. Write transactions to bit 7 cause bit 7 (NOTACARD) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. |
| 6 | RSVD | R | Reserved. Bit 6 returns 0b when read. |
| 5 | FCBCARD | W | Force CardBus card. Write transactions to bit 5 cause bit 5 (CBCARD) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. |
| 4 | F16BITCARD | W | Force 16-bit card. Write transactions to bit 4 cause bit 4 (16BITCARD) in the socket present-state register (CB offset 08h, see Section 6.3) to be written. |
| 3 | FPWRCYCLE | W | Force power cycle. Write transactions to bit 3 cause bit 3 (PWREVENT) in the socket event register (CB offset 00h, see Section 6.1) to be written, and bit 3 (PWRCYCLE) in the socket present-state register (CB offset 08h, see Section 6.3) is unaffected. |
| 2 | FCDETECT2 | W | Force $\overline{\text{CCD2}}$. Write transactions to bit 2 cause bit 2 (CD2EVENT) in the socket event register (CB offset 00h, see Section 6.1) to be written, and bit 2 (CDETECT2) in the socket present-state register (CB offset 08h, see Section 6.3) is unaffected. |
| 1 | FCDETECT1 | W | Force $\overline{\text{CCD1}}$. Write transactions to bit 1 cause bit 1 (CD1EVENT) in the socket event register (CB offset 00h, see Section 6.1) to be written, and bit 1 (CDETECT1) in the socket present-state register (CB offset 08h, see Section 6.3) is unaffected. |
| 0 | FCARDSTS | W | Force CSTSCHG. Write transactions to bit 0 cause bit 0 (CSTSEVENT) in the socket event register (CB offset 00h, see Section 6.1) to be written, and bit 0 (CARDSTS) in the socket present-state register (CB offset 08h, see Section 6.3) is unaffected. |

## 6.5 Socket Control Register

The socket control register provides control of the voltages applied to the socket and instructions for the CB $\overline{\text{CLKRUN}}$ protocol. The controller ensures that the socket is powered up only at acceptable voltages when a CardBus card is inserted. See Table 6−6 for a complete description of the register contents.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket control**
Offset: CardBus socket address + 10h
Type: Read-only, Read/Write
Default: 0000 0400h

**Table 6−6. Socket Control Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31−12 | RSVD | R | Reserved. These bits return 00000h when read. A write to these bits has no effect. |
| 11 | ZV_ACTIVITY | R | Zoomed video activity. This bit returns 0b when the ZVEN bit is 0b (disabled). If the ZVEN bit is set to 1b, the ZV_ACTIVITY bit returns 1b. |
| 10 | STDZVREG | R | Standardized zoomed video register model support. This bit returns 1b when the STDZVEN bit (bit 0) in the diagnostic register is cleared (PCI offset 93h, see Section 4.34). |
| 9 | ZVEN | RW | Zoomed video enable. This bit enables zoomed video for this socket. |
| 8 | RSVD | R | Reserved. ThIs bit returns 0b when read. A write to thIs bit has no effect. |
| 7 | STOPCLK | RW | CB $\overline{\text{CLKRUN}}$ protocol instructions.<br>0 = CB $\overline{\text{CLKRUN}}$ protocol can only attempt to stop/slow the CB clock if the socket is idle and the PCI $\overline{\text{CLKRUN}}$ protocol is preparing to stop/slow the PCI bus clock<br>1 = CB $\overline{\text{CLKRUN}}$ protocol can attempt to stop/slow the CB clock if the socket is idle |
| 6−4 | VCCCTRL | RW | $V_{CC}$ control. Bits 6−4 request card $V_{CC}$ changes.<br>000 = Request power off (default)<br>001 = Reserved<br>010 = Request $V_{CC}$ = 5 V<br>011 = Request $V_{CC}$ = 3.3 V<br>100 = Request $V_{CC}$ = X.X V<br>101 = Request $V_{CC}$ = Y.Y V<br>110 = Reserved<br>111 = Reserved |
| 3 | RSVD | R | Reserved. Bit 3 returns 0b when read. |
| 2−0 | VPPCTRL | RW | $V_{PP}$ control. Bits 2−0 request card $V_{PP}$ changes.<br>000 = Request power off (default)<br>001 = Request $V_{PP}$ = 12 V<br>010 = Request $V_{PP}$ = 5 V<br>011 = Request $V_{PP}$ = 3.3 V<br>100 = Request $V_{PP}$ = X.X V<br>101 = Request $V_{PP}$ = Y.Y V<br>110 = Reserved<br>111 = Reserved |

## 6.6 Socket Power-Management Register

This register provides power management control over the socket through a mechanism for slowing or stopping the clock on the card interface when the card is idle. See Table 6−7 for a complete description of the register contents.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket power-management**
Type: Read-only, Read/Write
Offset: CardBus socket address + 20h
Default: 0000 0000h

**Table 6−7. Socket Power-Management Register Description**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31−26 | RSVD | R | Reserved. Bits 31−26 return 00 0000b when read. |
| 25 | SKTACCES | R | Socket access status. This bit provides information on when a socket access has occurred. This bit is cleared by a read access.<br>0 = A PC card access has not occurred (default)<br>1 = A PC card access has occurred |
| 24 | SKTMODE | R | Socket mode status. This bit provides clock mode information.<br>0 = Clock is operating normally<br>1 = Clock frequency has changed |
| 23−17 | RSVD | R | Reserved. Bits 23−17 return 000 0000b when read. |
| 16 | CLKCTRLEN | RW | CardBus clock control enable. When bit 16 is set, bit 0 (CLKCTRL) is enabled.<br>0 = Clock control is disabled (default)<br>1 = Clock control is enabled |
| 15−1 | RSVD | R | Reserved. Bits 15−1 return 0s when read. |
| 0 | CLKCTRL | RW | CardBus clock control. This bit determines whether the CB $\overline{\text{CLKRUN}}$ protocol stops or slows the CB clock during idle states. Bit 16 (CLKCTRLEN) enables this bit.<br>0 = Allows CB $\overline{\text{CLKRUN}}$ protocol to stop the CB clock (default)<br>1 = Allows CB $\overline{\text{CLKRUN}}$ protocol to slow the CB clock by a factor of 16 |

# 7 Electrical Characteristics

## 7.1 Absolute Maximum Ratings Over Operating Temperature Ranges†

Supply voltage range, $V_{CC}$ . . . . . −0.5 V to 4.6 V
Clamping voltage range, $V_{CCP}$, $V_{CCCB}$ . . . . . −0.5 V to 6 V
Input voltage range, $V_I$: PCI, miscellaneous . . . . . −0.5 V to $V_{CCP}$ + 0.5 V
PC Card . . . . . −0.5 to $V_{CCCB}$ + 0.5 V
Fail safe . . . . . −0.5 V to $V_{CC}$ + 0.5 V
Output voltage range, $V_O$: PCI, miscellaneous . . . . . −0.5 V to $V_{CCP}$ + 0.5 V
PC Card . . . . . −0.5 to $V_{CCCB}$ + 0.5 V
Fail safe . . . . . −0.5 V to $V_{CC}$ + 0.5 V
Input clamp current, $I_{IK}$ ($V_I$ < 0 or $V_I$ > $V_{CC}$) (see Note 1) . . . . . ±20 mA
Output clamp current, $I_{OK}$ ($V_O$ < 0 or $V_O$ > $V_{CC}$) (see Note 2) . . . . . ±20 mA
Storage temperature range, $T_{stg}$ . . . . . −65°C to 150°C
Virtual junction temperature, $T_J$ . . . . . 150°C

† Stresses beyond those listed under absolute maximum ratings may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions beyond those indicated under recommended operating conditions is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES: 1. Applies for external input and bidirectional buffers. $V_I$ > $V_{CC}$ does not apply to fail-safe terminals. PCI terminals and miscellaneous terminals are measured with respect to $V_{CCP}$ instead of $V_{CC}$. PC Card terminals are measured with respect to $V_{CCCB}$. The limit specified applies for a dc condition.

2. Applies for external output and bidirectional buffers. $V_O$ > $V_{CC}$ does not apply to fail-safe terminals. PCI terminals and miscellaneous terminals are measured with respect to $V_{CCP}$ instead of $V_{CC}$. PC Card terminals are measured with respect to $V_{CCCB}$. The limit specified applies for a dc condition.

## 7.2 Recommended Operating Conditions (see Note 3)

| | | | OPERATION | MIN | NOM | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Core voltage | Commercial | 3.3 V | 3 | 3.3 | 3.6 | V |
| $V_{CCP}$ | PCI and miscellaneous I/O clamp voltage | Commercial | 3.3 V | 3 | 3.3 | 3.6 | V |
| | | | 5 V | 4.75 | 5 | 5.25 | |
| $V_{CCCB}$ | PC Card I/O clamp voltage | Commercial | 3.3 V | 3 | 3.3 | 3.6 | V |
| | | | 5 V | 4.75 | 5 | 5.25 | |
| $V_{IH}$† | High-level input voltage | PCI | 3.3 V | 0.5 $V_{CCP}$ | | $V_{CCP}$ | V |
| | | | 5 V | 2 | | $V_{CCP}$ | |
| | | PC Card | 3.3 V | 0.475 $V_{CCCB}$ | | $V_{CCCB}$ | |
| | | | 5 V | 2.4 | | $V_{CCCB}$ | |
| | | Miscellaneous‡ | | 2 | | $V_{CC}$ | |
| $V_{IL}$† | Low-level input voltage | PCI | 3.3 V | 0 | | 0.3 $V_{CCP}$ | V |
| | | | 5 V | 0 | | 0.8 | |
| | | PC Card | 3.3 V | 0 | | 0.325 $V_{CCCB}$ | |
| | | | 5 V | 0 | | 0.8 | |
| | | Miscellaneous‡ | | 0 | | 0.8 | |
| $V_I$ | Input voltage | PCI | | 0 | | $V_{CCP}$ | V |
| | | PC Card | | 0 | | $V_{CCCB}$ | |
| | | Miscellaneous‡ | | 0 | | $V_{CC}$ | |
| $V_O$§ | Output voltage | PCI | | 0 | | $V_{CC}$ | V |
| | | PC Card | | 0 | | $V_{CC}$ | |
| | | Miscellaneous‡ | | 0 | | $V_{CC}$ | |
| $t_t$ | Input transition time ($t_r$ and $t_f$) | PCI and PC Card | | 1 | | 4 | ns |
| | | Miscellaneous‡ | | 0 | | 6 | |
| $T_A$ | Operating ambient temperature range | | | 0 | 25 | 70 | °C |
| $T_J$¶ | Virtual junction temperature | | | 0 | 25 | 115 | °C |

† Applies to external inputs and bidirectional buffers without hysteresis

‡ Miscellaneous terminals are 65, 66, 75, 117, 130, and 138 for the PGE-packaged device; A09, B02, B05, L11, L13, and N10 for the GGU-packaged device; and B10, C09, F12, G03, H02, and L17 for the GVF-packaged device ($\overline{SUSPEND}$, $\overline{GRST}$, $\overline{CDx}$, and $\overline{VSx}$ terminals).

§ Applies to external output buffers

¶ These junction temperatures reflect simulation conditions. The customer is responsible for verifying junction temperature.

NOTE 3: Unused terminals (input or I/O) must be held high or low to prevent them from floating.

## 7.3 Electrical Characteristics Over Recommended Operating Conditions (unless otherwise noted)

| PARAMETER | | TERMINALS | OPERATION | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | High-level output voltage | PCI | 3.3 V | $I_{OH}$ = −0.5 mA | 0.9 $V_{CC}$ | | V |
| | | | 5 V | $I_{OH}$ = −2 mA | 2.4 | | |
| | | PC Card | 3.3 V | $I_{OH}$ = −0.15 mA | 0.9 $V_{CC}$ | | V |
| | | | 5 V | $I_{OH}$ = −0.15 mA | 2.4 | | |
| | | Miscellaneous | | $I_{OH}$ = −4 mA | $V_{CC}$−0.6 | | |
| $V_{OL}$ | Low-level output voltage | PCI | 3.3 V | $I_{OL}$ = 1.5 mA | | 0.1 $V_{CC}$ | V |
| | | | 5 V | $I_{OL}$ = 6 mA | | 0.55 | |
| | | PC Card | 3.3 V | $I_{OL}$ = 0.7 mA | | 0.1 $V_{CC}$ | |
| | | | 5 V | $I_{OL}$ = 0.7 mA | | 0.55 | |
| | | Miscellaneous | | $I_{OL}$ = 4 mA | | 0.5 | |
| $I_{OZL}$ | High-impedance, low-level output current | Output terminals | 3.6 V | $V_I$ = $V_{CC}$ | | −1 | µA |
| | | | 5.25 V | $V_I$ = $V_{CC}$ | | −1 | |
| $I_{OZH}$ | High-impedance, high-level output current | Output terminals | 3.6 V | $V_I$ = $V_{CC}$† | | 10 | µA |
| | | | 5.25 V | $V_I$ = $V_{CC}$† | | 25 | |
| $I_{IL}$ | Low-level input current | Input terminals | | $V_I$ = GND | | −1 | µA |
| | | I/O terminals | | $V_I$ = GND | | −10 | |
| | | Pullup terminals | | $V_I$ = GND | | −330 | |
| $I_{IH}$ | High-level input current | Input terminals | 3.6 V | $V_I$ = $V_{CC}$‡ | | 10 | µA |
| | | | 5.25 V | $V_I$ = $V_{CC}$‡ | | 20 | |
| | | I/O terminals | 3.6 V | $V_I$ = $V_{CC}$‡ | | 10 | |
| | | | 5.25 V | $V_I$ = $V_{CC}$‡ | | 25 | |
| | | Pulldown | 5.25 V | $V_I$ = $V_{CC}$‡ | | 30 | |

† For PCI and miscellaneous terminals, $V_I$ = $V_{CCP}$. For PC Card terminals, $V_I$ = $V_{CCCB}$.
‡ For I/O terminals, input leakage ($I_{IL}$ and $I_{IH}$) includes $I_{OZ}$ leakage of the disabled output.

## 7.4 PCI Clock/Reset Timing Requirements Over Recommended Ranges of Supply Voltage and Operating Free-Air Temperature

| PARAMETER | | ALTERNATE SYMBOL | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $t_c$ | Cycle time, PCLK | $t_{cyc}$ | | 30 | | ns |
| $t_{w(H)}$ | Pulse duration (width), PCLK high | $t_{high}$ | | 11 | | ns |
| $t_{w(L)}$ | Pulse duration (width), PCLK low | $t_{low}$ | | 11 | | ns |
| $t_r$, $t_f$ | Slew rate, PCLK | $\Delta v/\Delta t$ | | 1 | 4 | V/ns |
| $t_w$ | Pulse duration (width), $\overline{\text{PRST}}$ | $t_{rst}$ | | 1 | | ms |
| $t_{su}$ | Setup time, PCLK active at end of PRST | $t_{rst\text{-}clk}$ | | 100 | | µs |

## 7.5 PCI Timing Requirements Over Recommended Ranges of Supply Voltage and Operating Free-Air Temperature

This data manual uses the following conventions to describe time ( t ) intervals. The format is $t_A$, where *subscript A* indicates the type of dynamic parameter being represented. One of the following is used: $t_{pd}$ = propagation delay time, $t_d$ ($t_{en}$, $t_{dis}$) = delay time, $t_{su}$ = setup time, and $t_h$ = hold time.

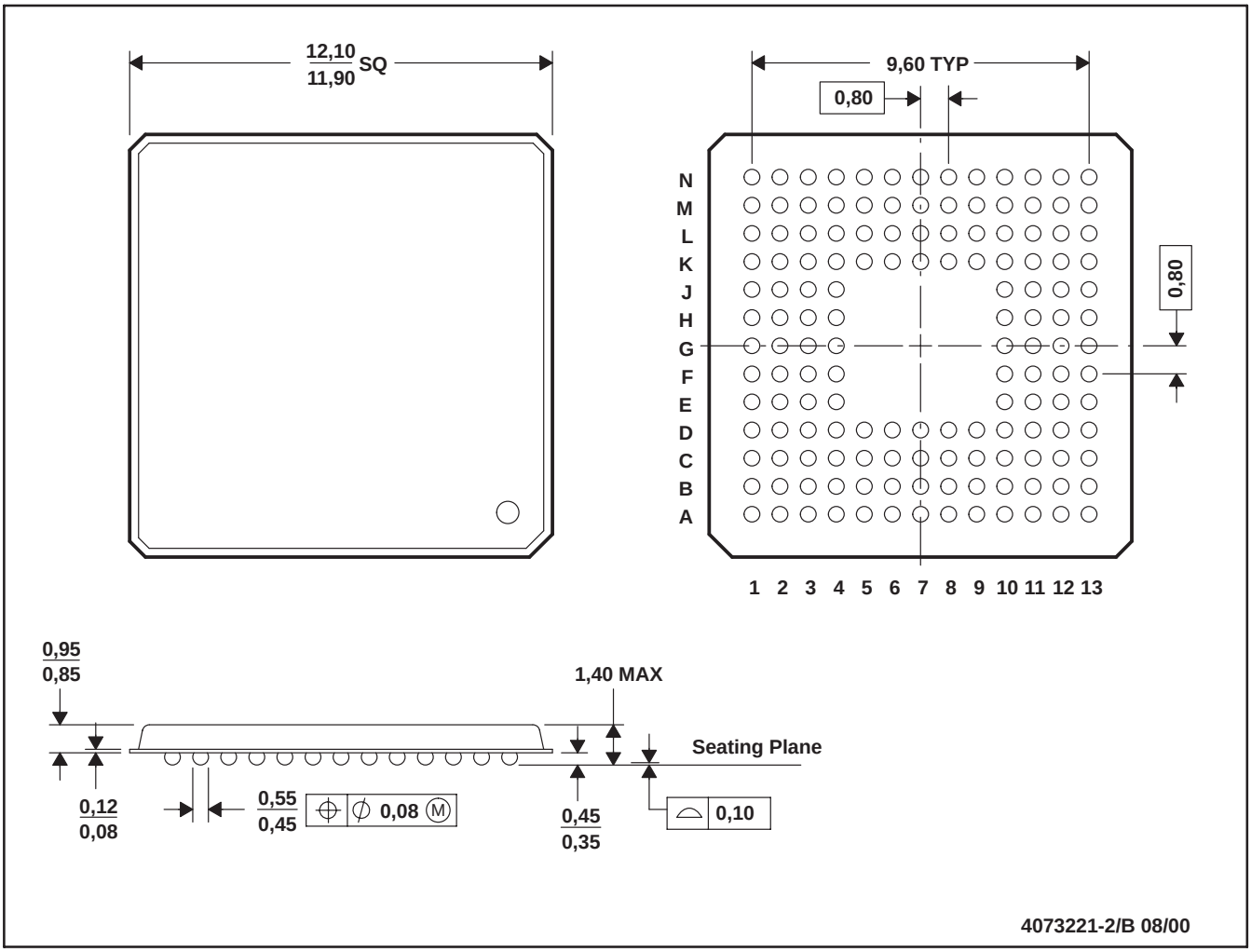
| PARAMETER | | | ALTERNATE SYMBOL | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{pd}$ | Propagation delay time, See Note 4 | PCLK-to-shared signal valid delay time | $t_{val}$ | $C_L$ = 50 pF, See Note 4 | | 11 | ns |
| | | PCLK-to-shared signal invalid delay time | $t_{inv}$ | | 2 | | |
| $t_{en}$ | Enable time, high impedance-to-active delay time from PCLK | | $t_{on}$ | | 2 | | ns |
| $t_{dis}$ | Disable time, active-to-high impedance delay time from PCLK | | $t_{off}$ | | | 28 | ns |
| $t_{su}$ | Setup time before PCLK valid | | $t_{su}$ | | 7 | | ns |
| $t_h$ | Hold time after PCLK high | | $t_h$ | | 0 | | ns |

NOTE 4: PCI shared signals are AD31−AD0, C/$\overline{BE3}$−C/$\overline{BE0}$, $\overline{FRAME}$, $\overline{TRDY}$, $\overline{IRDY}$, $\overline{STOP}$, IDSEL, $\overline{DEVSEL}$, and PAR.

# 8 Mechanical Information

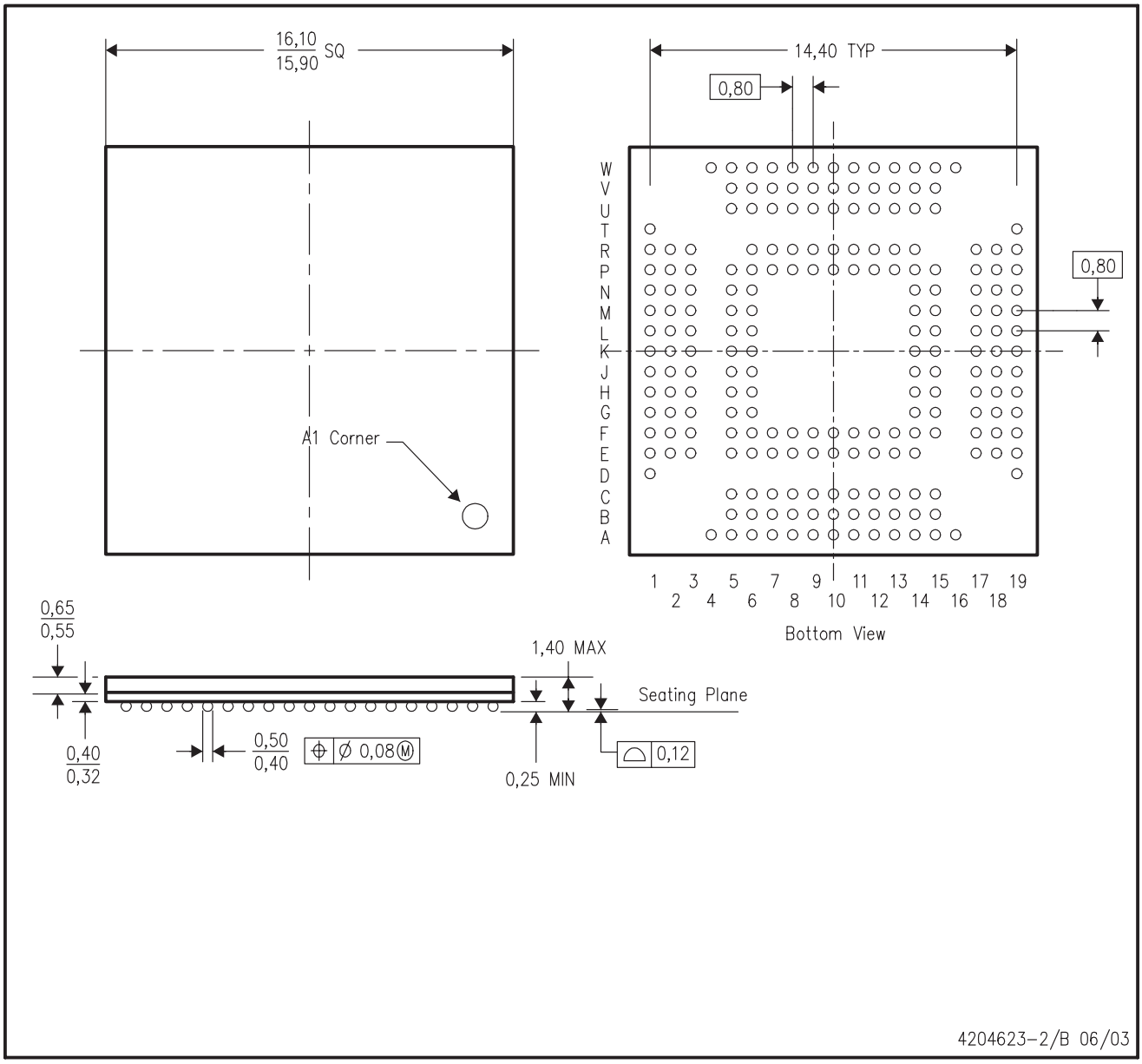
The PCI1510 is packaged in either a 144-ball GGU or ZGU BGA, 209-ball GVF or ZVF BGA, or a 144-pin PGE package. The following shows the mechanical dimensions for the GGU, GVF, PGE, ZGU, and ZVF packages.

**GGU (S-PBGA-N144)** **PLASTIC BALL GRID ARRAY**



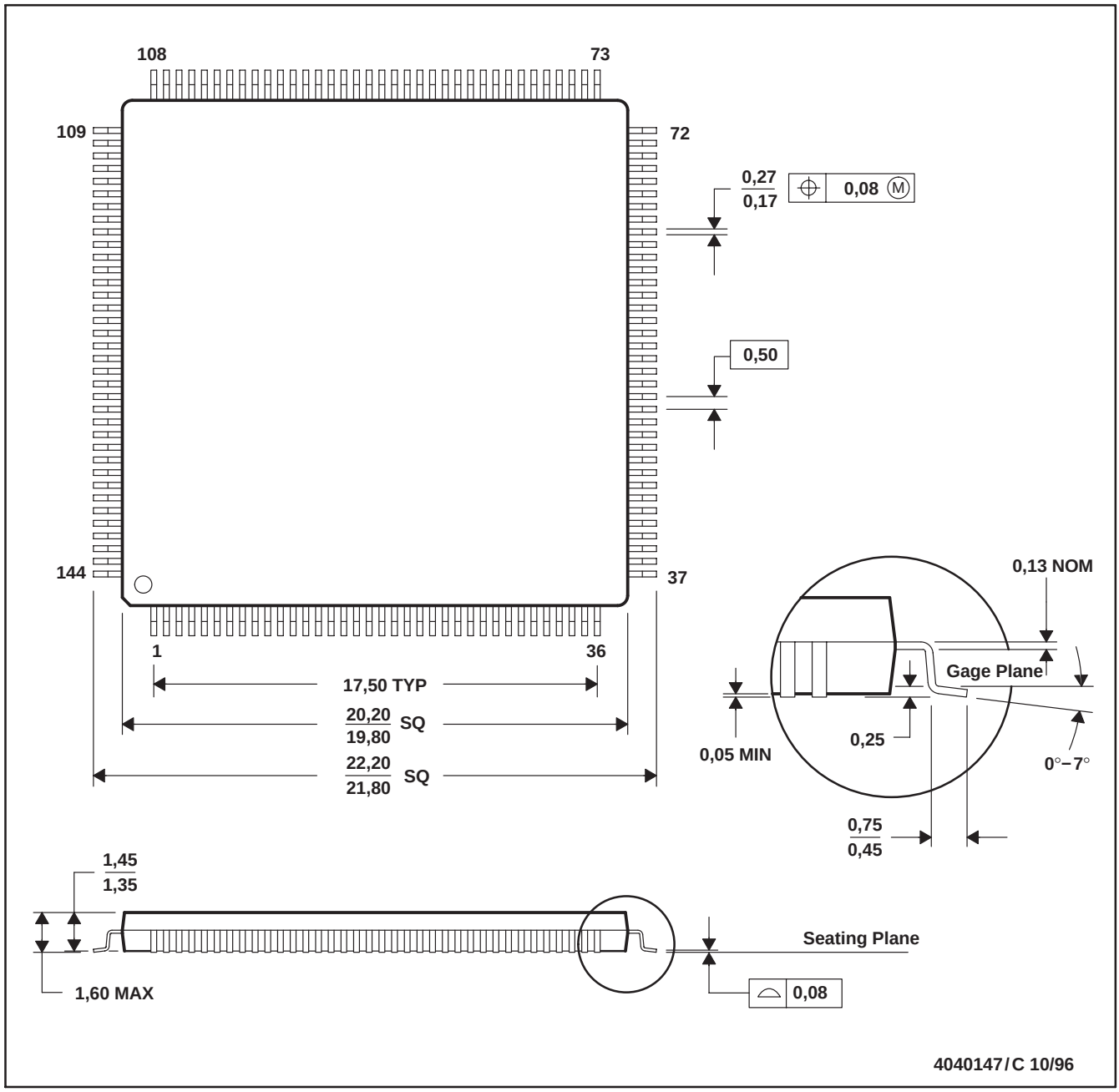

NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. MicroStar BGA™ configuration

MicroStar BGA is a trademark of Texas Instruments.

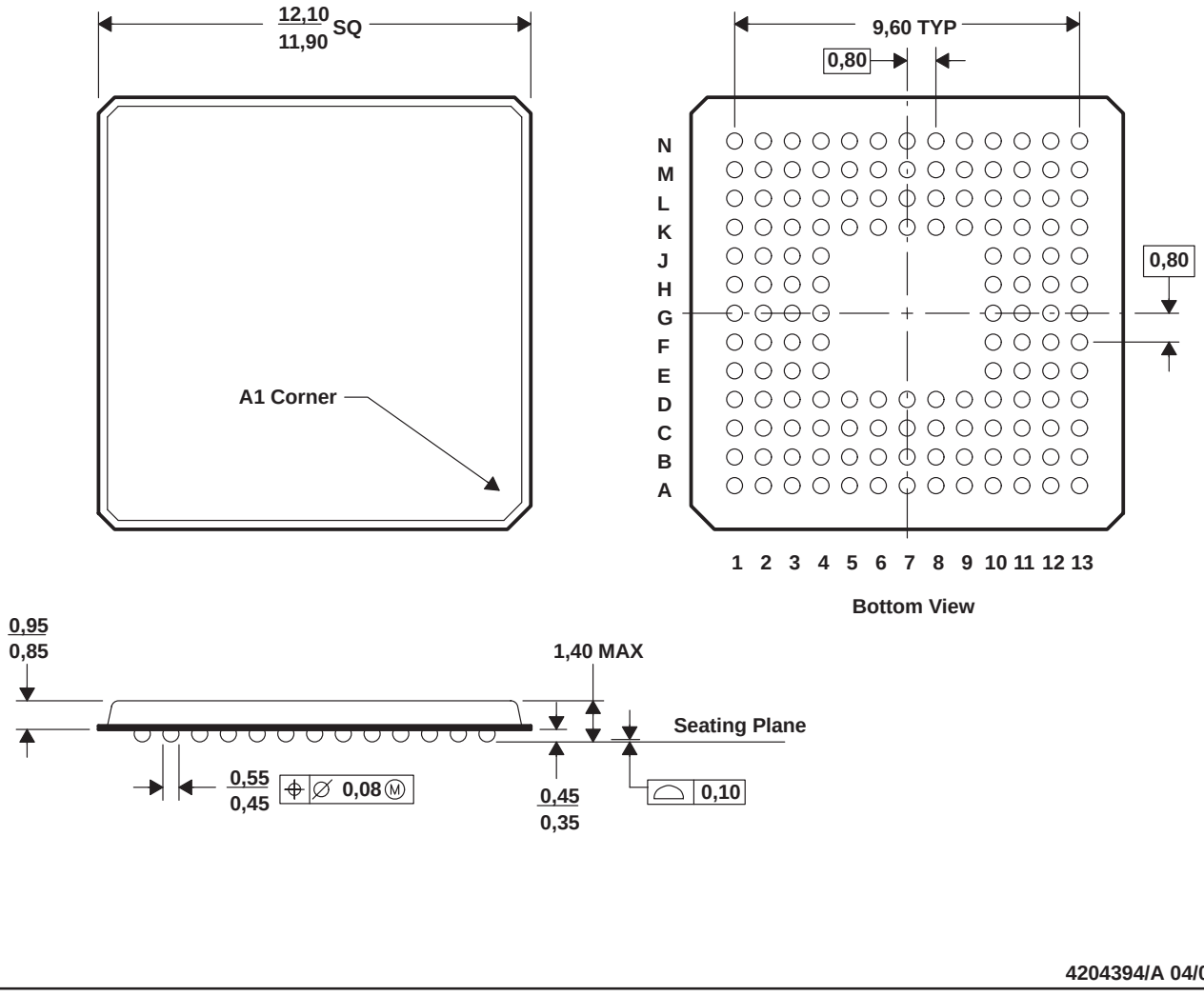



NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.

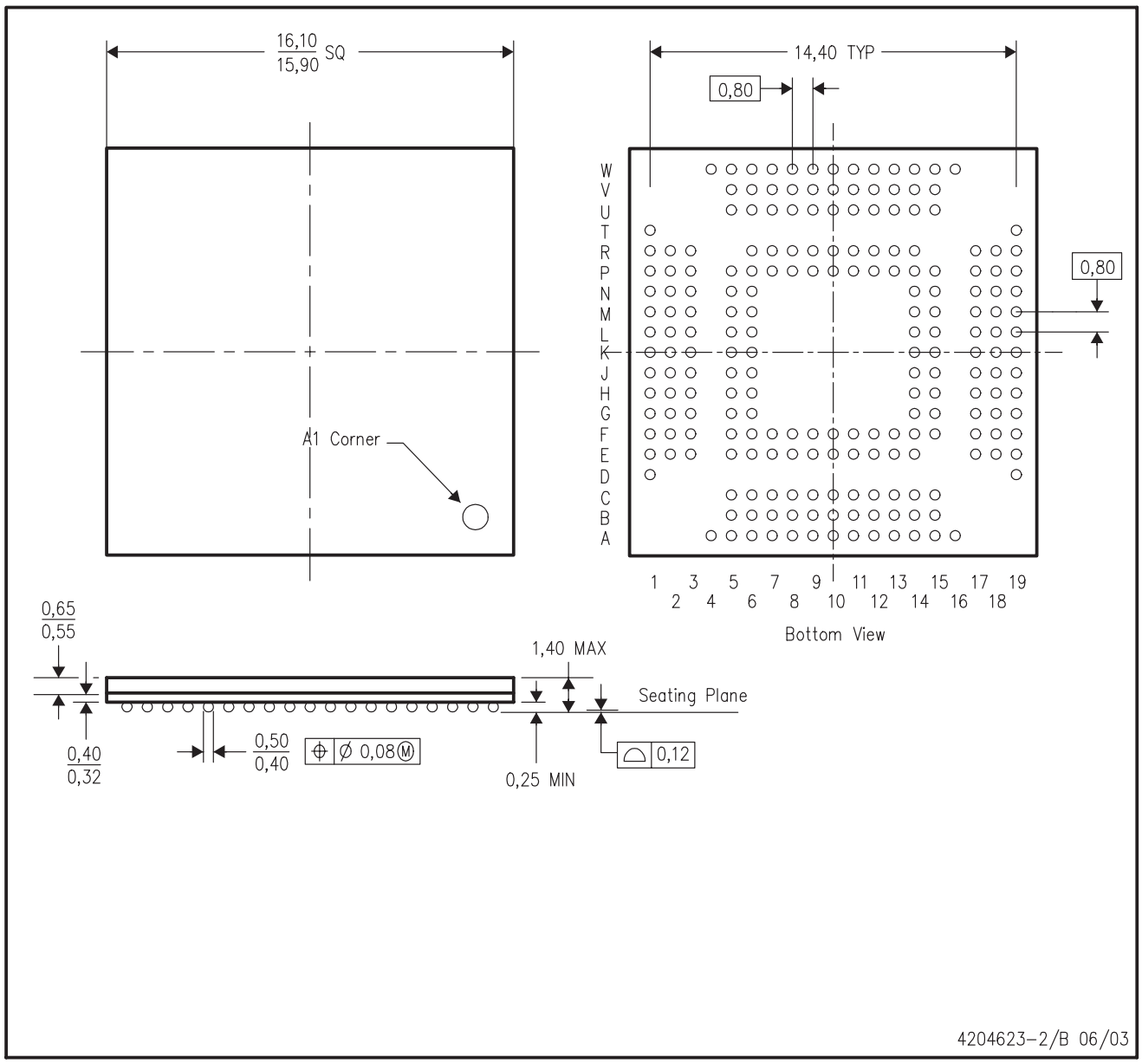
**PGE (S-PQFP-G144)** **PLASTIC QUAD FLATPACK**




NOTES: C. All linear dimensions are in millimeters.
D. This drawing is subject to change without notice.
E. Falls within JEDEC MS-026




NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. MicroStar BGA™ configuration
D. This package is lead-free.




NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. This package is lead-free.

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View PCI1510ZGU on WIN SOURCE
- Texas Instruments Information

## Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management