- **PC 98/99 Compliant**
- **PCI Bus Power Management Interface Specification 1.0 Compliant**
- **Advanced Configuration and Power Interface (ACPI) 1.0 Compliant**
- **Fully Compatible With the Intel™ 430TX (Mobile Triton II) Chipset**
- **PCI Local Bus Specification Revision 2.2 Compliant**
- **1997 PC Card™ Standard Compliant**
- **3.3-V Core Logic With Universal PCI Interfaces Compatible With 3.3-V and 5-V PCI Signaling Environments**
- **Mix-and-Match 5-V/3.3-V PC Card16 Cards and 3.3-V CardBus Cards**
- **Supports a Single PC Card or CardBus Slot With Hot Insertion and Removal**
- **Provides Interface to Parallel Single-Slot PC Card Power-Interface Switches like the TI™ TPS2211**
- **Supports Burst Transfers to Maximize Data Throughput on the PCI Bus and the CardBus Bus**
- **Supports Parallel PCI Interrupts, Parallel ISA IRQ and Parallel PCI Interrupts, Serial ISA IRQ With Parallel PCI Interrupts, and Serial ISA IRQ and PCI Interrupts**
- **Pin-to-Pin Compatible with PCI1210**
- **Serial EEPROM Interface for Loading Subsystem ID and Subsystem Vendor ID**
- **Pipelined Architecture Allows Greater Than 130M-Bytes-Per-Second Throughput From CardBus to PCI and From PCI to CardBus**
- **Supports Up to Five General-Purpose I/Os**
- **Five PCI Memory Windows and Two I/O Windows Available to the PC Card16 Socket**
- **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 Register Compatible**
- **Supports Distributed DMA (DDMA) and PC/PCI DMA**
- **Supports 16-Bit DMA on the PC Card Socket**
- **Supports Ring Indicate, $\overline{\text{SUSPEND}}$, PCI $\overline{\text{CLKRUN}}$, and CardBus $\overline{\text{CCLKRUN}}$**
- **Supports PCI Bus Lock ($\overline{\text{LOCK}}$)**
- **LED Activity Pin**
- **Advanced Submicron, Low-Power CMOS Technology**
- **Choice of Surface-Mount Packaging:**
  - **PGE Low-Profile Plastic Quad Flat Package (LQFP)**
  - **GGU High Density Ball Grid Array (BGA)**

**Table of Contents**


Description . . . . . 2
System Block Diagram . . . . . 3
Terminal Assignments . . . . . 4
Signal Name/Terminal Assignments . . . . . 6
Terminal Functions . . . . . 11
Power Supply Sequencing . . . . . 21
I/O Characteristics . . . . . 21
Clamping Voltages . . . . . 21
Peripheral Component Interconnect (PCI) Interface . . . . . 22
PC Card Applications . . . . . 23
Serial Bus Interface . . . . . 30
Programmable Interrupt Subsystem . . . . . 34
Power Management Overview . . . . . 38
PC Card Controller Programming Model . . . . . 43
PCI Configuration Registers . . . . . 43
ExCA Compatibility Registers . . . . . 80
CardBus Socket Registers . . . . . 104
Distributed DMA (DDMA) Registers . . . . . 112
Absolute Maximum Ratings . . . . . 118
Recommended Operating Conditions . . . . . 119
Electrical Characteristics . . . . . 120
PCI Clock/Reset Timing Requirements . . . . . 121
PCI Timing Requirements . . . . . 121
Parameter Measurement Information . . . . . 122
PCI Bus Parameter Measurement Information . . . . . 123
Mechanical Data . . . . . 128


Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

Intel is a trademark of Intel Corporation.
PC Card is a trademark of Personal Computer Memory Card International Association (PCMCIA).
TI is a trademark of Texas Instruments Incorporated.

**PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.**

Copyright © 1998, Texas Instruments Incorporated

## description

The Texas Instruments PCI1211 is a high-performance PCI-to-PC Card controller that supports a single PC Card socket compliant with the 1995 PC Card Standard. The PCI1211 provides a rich feature set that makes it the best choice for bridging between PCI and PC Cards in both notebook and desktop computers. The 1997 PC Card Standard retains the 16-bit PC Card specification defined in PCMCIA Release 2.2, and defines the new 32-bit PC Card, CardBus, capable of full 32-bit data transfers at 33 MHz. The PCI1211 supports both 16-bit and CardBus PC Cards, powered at 5 V or 3.3 V, as required.

The PCI1211 is compliant with the *PCI Local Bus Specification, Revision 2.2*, and its PCI interface can act as either a PCI master device or a PCI slave device. The PCI bus mastering is initiated during 16-bit PC Card direct memory access (DMA) transfers or CardBus PC Card bridging transactions. The PCI1211 is also compliant with the latest *PCI Bus Power Management Interface Specification, Revision 1.0*.

All card signals are internally buffered to allow hot insertion and removal without external buffering. The PCI1211 is register compatible with the Intel 82365SL-DF ExCA controller. The PCI1211 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 PCI1211 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 are designed into the PCI1211, such as socket activity light-emitting diode (LED) output, that are discussed in detail throughout the design specification.

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.

Unused PCI1211 inputs must be pulled up using a 43 kΩ resistor.

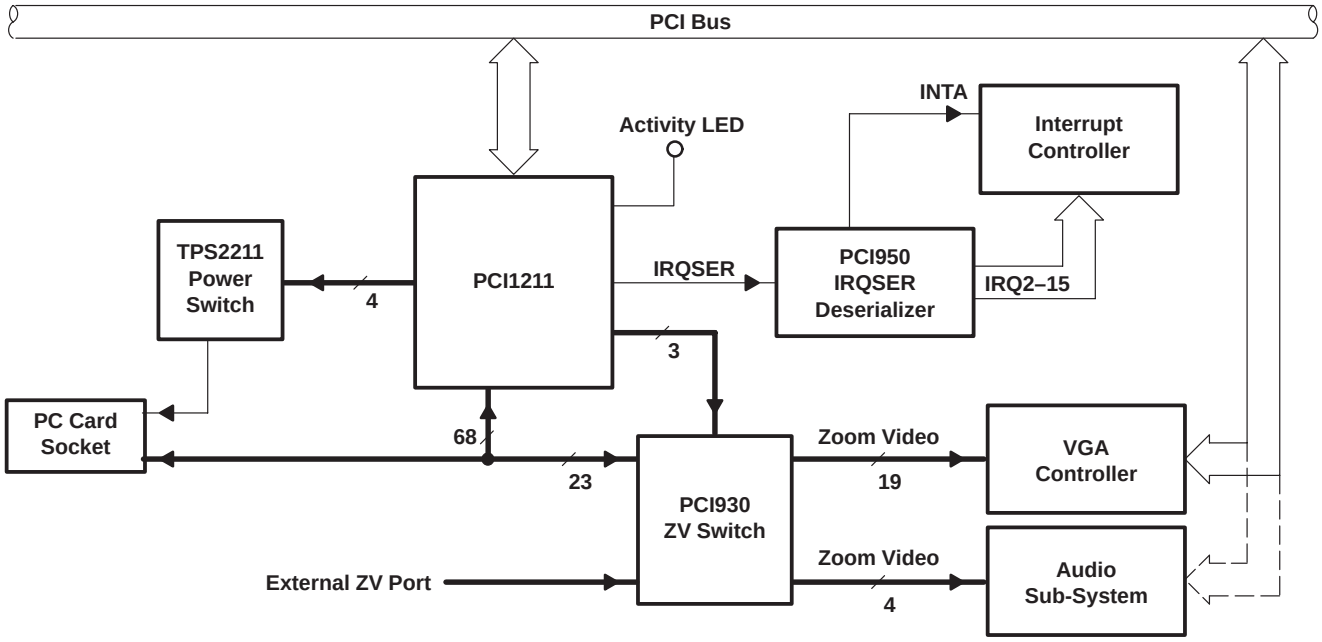
## system block diagram

A simplified system block diagram using the PCI1211 is provided below. The PCI950 IRQ deserializer and the PCI930 zoomed video (ZV) switch are optional functions that can be used when the system requires that capability.

The PCI interface includes all address/data and control signals for PCI protocol. The 68-pin PC Card interface includes all address/data and control signals for CardBus and 16-bit (R2) protocols. When ZV is enabled (in 16-bit PC Card mode) 23 of the 68 signals are redefined to support the ZV protocol.

The interrupt interface includes terminals for parallel PCI, parallel ISA, and serialized PCI and ISA signaling. Other miscellaneous system interface terminals are available on the PCI1211 that include:

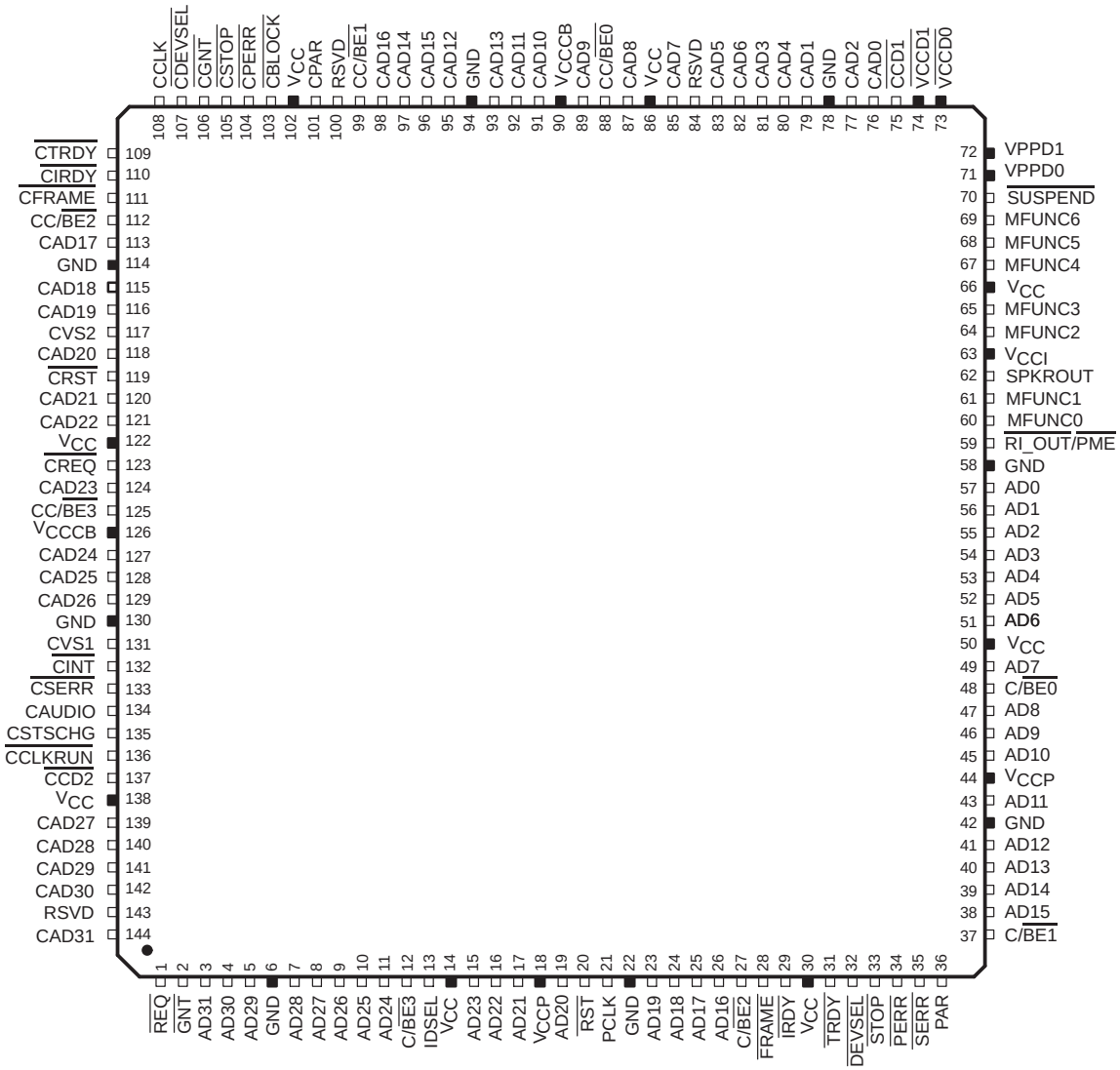
- Programmable multifunction terminals
- $\overline{\text{SUSPEND}}$, $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ (power management control signal)
- SPKROUT




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

## terminal assignments

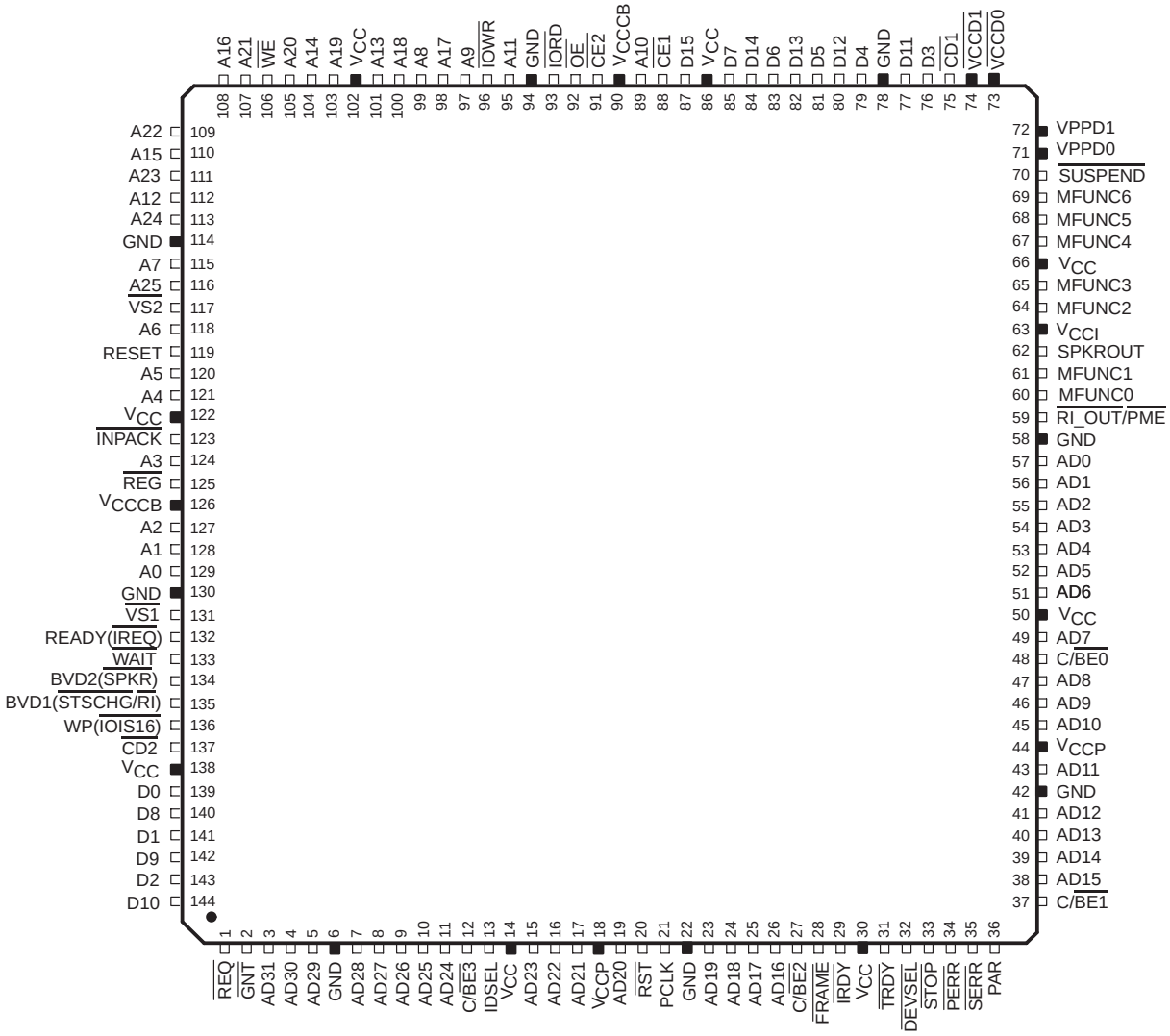
**PGE LOW-PROFILE QUAD FLAT PACKAGE**
**(BOTTOM VIEW)**




**PCI-to-CardBus Pin Diagram**

## terminal assignments (continued)

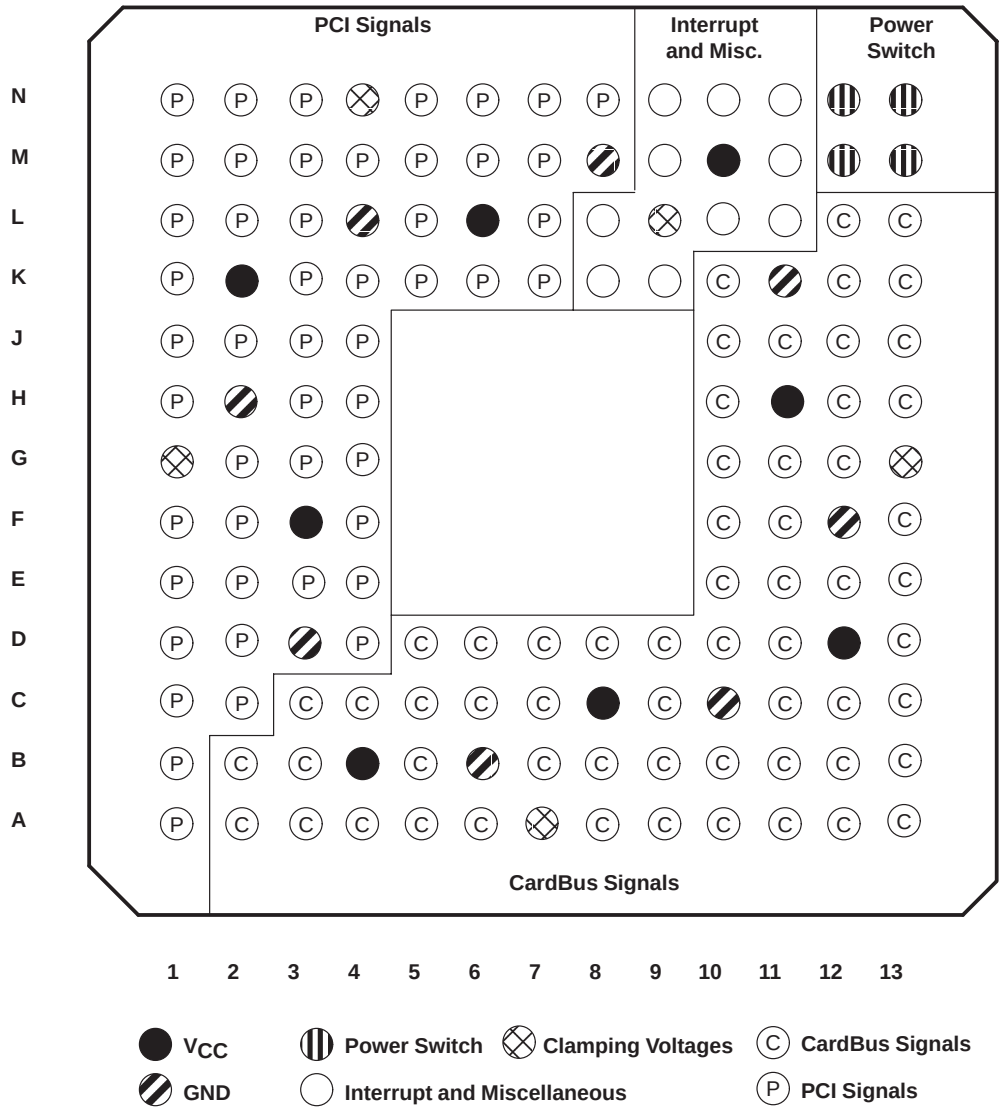
**PGE LOW-PROFILE QUAD FLAT PACKAGE**
**(BOTTOM VIEW)**




**PCI-to-PC Card (16-Bit) Diagram**

## terminal assignments (continued)

**GGU BALL GRID ARRAY PACKAGE**
**(BOTTOM VIEW)**




**PCI-to-CardBus and PCI-to-PC Card (16-Bit) Diagram**

## signal names and terminal assignments

Signal names and their terminal assignments are shown in Table 1 through Table 4. Table 1 and Table 2 show the terminal assignments for the CardBus PC Card, and Table 3 and Table 4 show the terminal assignments for the 16-bit PC Card. Table 2 and Table 4 show the CardBus PC Card and the 16-bit PC Card terminals sorted alphanumerically by the signal name and its associated terminal number.

**Table 1. CardBus PC Card Signal Names – Sorted by BGA Terminal Number†**

| PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME |
|---|---|---|---|---|---|---|---|---|---|---|---|
| GGU | PGE | | GGU | PGE | | GGU | PGE | | GGU | PGE | |
| A1 | 1 | $\overline{\text{REQ}}$ | C11 | 106 | $\overline{\text{CGNT}}$ | G10 | 92 | CAD11 | L4 | 42 | GND |
| A2 | 143 | RSVD | C12 | 105 | $\overline{\text{CSTOP}}$ | G11 | 91 | CAD10 | L5 | 46 | AD9 |
| A3 | 140 | CAD28 | C13 | 104 | $\overline{\text{CPERR}}$ | G12 | 89 | CAD9 | L6 | 50 | $V_{CC}$ |
| A4 | 137 | $\overline{\text{CCD2}}$ | D1 | 8 | AD27 | G13 | 90 | $V_{CCCB}$ | L7 | 55 | AD2 |
| A5 | 133 | $\overline{\text{CSERR}}$ | D2 | 7 | AD28 | H1 | 21 | PCLK | L8 | 59 | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ |
| A6 | 129 | CAD26 | D3 | 6 | GND | H2 | 22 | GND | L9 | 63 | $V_{CCI}$ |
| A7 | 126 | $V_{CCCB}$ | D4 | 5 | AD29 | H3 | 23 | AD19 | L10 | 67 | MFUNC4 |
| A8 | 124 | CAD23 | D5 | 136 | $\overline{\text{CCLKRUN}}$ | H4 | 24 | AD18 | L11 | 70 | $\overline{\text{SUSPEND}}$ |
| A9 | 120 | CAD21 | D6 | 132 | $\overline{\text{CINT}}$ | H10 | 85 | CAD7 | L12 | 75 | $\overline{\text{CCD1}}$ |
| A10 | 116 | CAD19 | D7 | 128 | CAD25 | H11 | 86 | $V_{CC}$ | L13 | 76 | CAD0 |
| A11 | 112 | CC/$\overline{\text{BE2}}$ | D8 | 121 | CAD22 | H12 | 87 | CAD8 | M1 | 35 | $\overline{\text{SERR}}$ |
| A12 | 110 | $\overline{\text{CIRDY}}$ | D9 | 117 | CVS2 | H13 | 88 | CC/$\overline{\text{BE0}}$ | M2 | 36 | PAR |
| A13 | 109 | $\overline{\text{CTRDY}}$ | D10 | 113 | CAD17 | J1 | 25 | AD17 | M3 | 39 | AD14 |
| B1 | 2 | $\overline{\text{GNT}}$ | D11 | 103 | $\overline{\text{CBLOCK}}$ | J2 | 26 | AD16 | M4 | 43 | AD11 |
| B2 | 144 | CAD31 | D12 | 102 | $V_{CC}$ | J3 | 27 | C/$\overline{\text{BE2}}$ | M5 | 47 | AD8 |
| B3 | 141 | CAD29 | D13 | 101 | CPAR | J4 | 28 | $\overline{\text{FRAME}}$ | M6 | 51 | AD6 |
| B4 | 138 | $V_{CC}$ | E1 | 12 | C/$\overline{\text{BE3}}$ | J10 | 81 | CAD3 | M7 | 53 | AD4 |
| B5 | 134 | CAUDIO | E2 | 11 | AD24 | J11 | 82 | CAD6 | M8 | 58 | GND |
| B6 | 130 | GND | E3 | 10 | AD25 | J12 | 83 | CAD5 | M9 | 62 | SPKROUT |
| B7 | 125 | CC/$\overline{\text{BE3}}$ | E4 | 9 | AD26 | J13 | 84 | RSVD | M10 | 66 | $V_{CC}$ |
| B8 | 123 | $\overline{\text{CREQ}}$ | E10 | 100 | RSVD | K1 | 29 | $\overline{\text{IRDY}}$ | M11 | 69 | MFUNC6 |
| B9 | 119 | $\overline{\text{CRST}}$ | E11 | 99 | CC/$\overline{\text{BE1}}$ | K2 | 30 | $V_{CC}$ | M12 | 72 | VPPD1 |
| B10 | 115 | CAD18 | E12 | 98 | CAD16 | K3 | 31 | $\overline{\text{TRDY}}$ | M13 | 74 | $\overline{\text{VCCD1}}$ |
| B11 | 111 | $\overline{\text{CFRAME}}$ | E13 | 97 | CAD14 | K4 | 41 | AD12 | N1 | 37 | C/$\overline{\text{BE1}}$ |
| B12 | 108 | CCLK | F1 | 16 | AD22 | K5 | 45 | AD10 | N2 | 38 | AD15 |
| B13 | 107 | $\overline{\text{CDEVSEL}}$ | F2 | 15 | AD23 | K6 | 49 | AD7 | N3 | 40 | AD13 |
| C1 | 4 | AD30 | F3 | 14 | $V_{CC}$ | K7 | 56 | AD1 | N4 | 44 | $V_{CCP}$ |
| C2 | 3 | AD31 | F4 | 13 | IDSEL | K8 | 60 | MFUNC0 | N5 | 48 | C/$\overline{\text{BE0}}$ |
| C3 | 142 | CAD30 | F10 | 96 | CAD15 | K9 | 64 | MFUNC2 | N6 | 52 | AD5 |
| C4 | 139 | CAD27 | F11 | 95 | CAD12 | K10 | 77 | CAD2 | N7 | 54 | AD3 |
| C5 | 135 | CSTSCHG | F12 | 94 | GND | K11 | 78 | GND | N8 | 57 | AD0 |
| C6 | 131 | CVS1 | F13 | 93 | CAD13 | K12 | 79 | CAD1 | N9 | 61 | MFUNC1 |
| C7 | 127 | CAD24 | G1 | 18 | $V_{CCP}$ | K13 | 80 | CAD4 | N10 | 65 | MFUNC3 |
| C8 | 122 | $V_{CC}$ | G2 | 17 | AD21 | L1 | 32 | $\overline{\text{DEVSEL}}$ | N11 | 68 | MFUNC5 |
| C9 | 118 | CAD20 | G3 | 19 | AD20 | L2 | 33 | $\overline{\text{STOP}}$ | N12 | 71 | VPPD0 |
| C10 | 114 | GND | G4 | 20 | $\overline{\text{RST}}$ | L3 | 34 | $\overline{\text{PERR}}$ | N13 | 73 | $\overline{\text{VCCD0}}$ |

† The PGE (LQFP) pin numbers are shown also.

## Table 2. CardBus PC Card Signal Names – Sorted Alphabetically

| SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | PGE | GGU | | PGE | GGU | | PGE | GGU | | PGE | GGU |
| AD0 | 57 | N8 | CAD0 | 76 | L13 | CC/$\overline{BE2}$ | 112 | A11 | MFUNC2 | 64 | K9 |
| AD1 | 56 | K7 | CAD1 | 79 | K12 | CC/$\overline{BE3}$ | 125 | B7 | MFUNC3 | 65 | N10 |
| AD2 | 55 | L7 | CAD2 | 77 | K10 | CCLK | 108 | B12 | MFUNC4 | 67 | L10 |
| AD3 | 54 | N7 | CAD3 | 81 | J10 | $\overline{CCD1}$ | 75 | L12 | MFUNC5 | 68 | N11 |
| AD4 | 53 | M7 | CAD4 | 80 | K13 | $\overline{CCD2}$ | 137 | A4 | MFUNC6 | 69 | M11 |
| AD5 | 52 | N6 | CAD5 | 83 | J12 | $\overline{CCLKRUN}$ | 136 | D5 | PAR | 36 | M2 |
| AD6 | 51 | M6 | CAD6 | 82 | J11 | $\overline{CDEVSEL}$ | 107 | B13 | PCLK | 21 | H1 |
| AD7 | 49 | K6 | CAD7 | 85 | H10 | $\overline{CFRAME}$ | 111 | B11 | $\overline{PERR}$ | 34 | L3 |
| AD8 | 47 | M5 | CAD8 | 87 | H12 | $\overline{CGNT}$ | 106 | C11 | $\overline{REQ}$ | 1 | A1 |
| AD9 | 46 | L5 | CAD9 | 89 | G12 | $\overline{CINT}$ | 132 | D6 | $\overline{RI_OUT}$/$\overline{PME}$ | 59 | L8 |
| AD10 | 45 | K5 | CAD10 | 91 | G11 | $\overline{CIRDY}$ | 110 | A12 | $\overline{RST}$ | 20 | G4 |
| AD11 | 43 | M4 | CAD11 | 92 | G10 | CPAR | 101 | D13 | $\overline{SERR}$ | 35 | M1 |
| AD12 | 41 | K4 | CAD12 | 95 | F11 | $\overline{CPERR}$ | 104 | C13 | RSVD | 84 | E10 |
| AD13 | 40 | N3 | CAD13 | 93 | F13 | $\overline{CREQ}$ | 123 | B8 | RSVD | 100 | J13 |
| AD14 | 39 | M3 | CAD14 | 97 | E13 | $\overline{CRST}$ | 119 | B9 | RSVD | 143 | A2 |
| AD15 | 38 | N2 | CAD15 | 96 | F10 | $\overline{CSERR}$ | 133 | A5 | SPKROUT | 62 | M9 |
| AD16 | 26 | J2 | CAD16 | 98 | E12 | $\overline{CSTOP}$ | 105 | C12 | $\overline{STOP}$ | 33 | L2 |
| AD17 | 25 | J1 | CAD17 | 113 | D10 | CSTSCHG | 135 | C5 | $\overline{SUSPEND}$ | 70 | L11 |
| AD18 | 24 | H4 | CAD18 | 115 | B10 | $\overline{CTRDY}$ | 109 | A13 | $\overline{TRDY}$ | 31 | K3 |
| AD19 | 23 | H3 | CAD19 | 116 | A10 | CVS1 | 131 | C6 | $V_{CC}$ | 14 | F3 |
| AD20 | 19 | G3 | CAD20 | 118 | C9 | CVS2 | 117 | D9 | $V_{CC}$ | 30 | K2 |
| AD21 | 17 | G2 | CAD21 | 120 | A9 | $\overline{DEVSEL}$ | 32 | L1 | $V_{CC}$ | 50 | L6 |
| AD22 | 16 | F1 | CAD22 | 121 | D8 | $\overline{FRAME}$ | 28 | J4 | $V_{CC}$ | 66 | M10 |
| AD23 | 15 | F2 | CAD23 | 124 | A8 | GND | 6 | D3 | $V_{CC}$ | 86 | H11 |
| AD24 | 11 | E2 | CAD24 | 127 | C7 | GND | 22 | H2 | $V_{CC}$ | 102 | D12 |
| AD25 | 10 | E3 | CAD25 | 128 | D7 | GND | 42 | L4 | $V_{CC}$ | 122 | C8 |
| AD26 | 9 | E4 | CAD26 | 129 | A6 | GND | 58 | M8 | $V_{CC}$ | 138 | B4 |
| AD27 | 8 | D1 | CAD27 | 139 | C4 | GND | 78 | K11 | $V_{CCCB}$ | 90 | G13 |
| AD28 | 7 | D2 | CAD28 | 140 | A3 | GND | 94 | F12 | $V_{CCCB}$ | 126 | A7 |
| AD29 | 5 | D4 | CAD29 | 141 | B3 | GND | 114 | C10 | $\overline{VCCD0}$ | 73 | N13 |
| AD30 | 4 | C1 | CAD30 | 142 | C3 | GND | 130 | B6 | $\overline{VCCD1}$ | 74 | M13 |
| AD31 | 3 | C2 | CAD31 | 144 | B2 | $\overline{GNT}$ | 2 | B1 | $V_{CCI}$ | 63 | L9 |
| C/$\overline{BE0}$ | 48 | N5 | CAUDIO | 134 | B5 | IDSEL | 13 | F4 | $V_{CCP}$ | 18 | G1 |
| C/$\overline{BE1}$ | 37 | N1 | $\overline{CBLOCK}$ | 103 | D11 | $\overline{IRDY}$ | 29 | K1 | $V_{CCP}$ | 44 | N4 |
| C/$\overline{BE2}$ | 27 | J3 | CC/$\overline{BE0}$ | 88 | H13 | MFUNC0 | 60 | K8 | VPPD0 | 71 | N12 |
| C/$\overline{BE3}$ | 12 | E1 | CC/$\overline{BE1}$ | 99 | E11 | MFUNC1 | 61 | N9 | VPPD1 | 72 | M12 |

**Table 3. 16-Bit PC Card Signal Names – Sorted by BGA Terminal Number†**

| PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME |
|---|---|---|---|---|---|---|---|---|---|---|---|
| GGU | PGE | | GGU | PGE | | GGU | PGE | | GGU | PGE | |
| A1 | 1 | $\overline{\text{REQ}}$ | C11 | 106 | $\overline{\text{WE}}$ | G10 | 92 | $\overline{\text{OE}}$ | L4 | 42 | GND |
| A2 | 143 | D2 | C12 | 105 | A20 | G11 | 91 | $\overline{\text{CE2}}$ | L5 | 46 | AD9 |
| A3 | 140 | D8 | C13 | 104 | A14 | G12 | 89 | A10 | L6 | 50 | $V_{CC}$ |
| A4 | 137 | $\overline{\text{CD2}}$ | D1 | 8 | AD27 | G13 | 90 | $V_{CCCB}$ | L7 | 55 | AD2 |
| A5 | 133 | $\overline{\text{WAIT}}$ | D2 | 7 | AD28 | H1 | 21 | PCLK | L8 | 59 | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ |
| A6 | 129 | A0 | D3 | 6 | GND | H2 | 22 | GND | L9 | 63 | $V_{CCI}$ |
| A7 | 126 | $V_{CCCB}$ | D4 | 5 | AD29 | H3 | 23 | AD19 | L10 | 67 | MFUNC4 |
| A8 | 124 | A3 | D5 | 136 | WP($\overline{\text{IOIS16}}$) | H4 | 24 | AD18 | L11 | 70 | $\overline{\text{SUSPEND}}$ |
| A9 | 120 | A5 | D6 | 132 | READY($\overline{\text{IREQ}}$) | H10 | 85 | D7 | L12 | 75 | $\overline{\text{CD1}}$ |
| A10 | 116 | A25 | D7 | 128 | A1 | H11 | 86 | $V_{CC}$ | L13 | 76 | D3 |
| A11 | 112 | A12 | D8 | 121 | A4 | H12 | 87 | D15 | M1 | 35 | $\overline{\text{SERR}}$ |
| A12 | 110 | A15 | D9 | 117 | $\overline{\text{VS2}}$ | H13 | 88 | $\overline{\text{CE1}}$ | M2 | 36 | PAR |
| A13 | 109 | A22 | D10 | 113 | A24 | J1 | 25 | AD17 | M3 | 39 | AD14 |
| B1 | 2 | $\overline{\text{GNT}}$ | D11 | 103 | A19 | J2 | 26 | AD16 | M4 | 43 | AD11 |
| B2 | 144 | D10 | D12 | 102 | $V_{CC}$ | J3 | 27 | C/$\overline{\text{BE2}}$ | M5 | 47 | AD8 |
| B3 | 141 | D1 | D13 | 101 | A13 | J4 | 28 | $\overline{\text{FRAME}}$ | M6 | 51 | AD6 |
| B4 | 138 | $V_{CC}$ | E1 | 12 | C/$\overline{\text{BE3}}$ | J10 | 81 | D5 | M7 | 53 | AD4 |
| B5 | 134 | BVD2($\overline{\text{SPKR}}$) | E2 | 11 | AD24 | J11 | 82 | D13 | M8 | 58 | GND |
| B6 | 130 | GND | E3 | 10 | AD25 | J12 | 83 | D6 | M9 | 62 | SPKROUT |
| B7 | 125 | $\overline{\text{REG}}$ | E4 | 9 | AD26 | J13 | 84 | D14 | M10 | 66 | $V_{CC}$ |
| B8 | 123 | $\overline{\text{INPACK}}$ | E10 | 100 | A18 | K1 | 29 | $\overline{\text{IRDY}}$ | M11 | 69 | MFUNC6 |
| B9 | 119 | RESET | E11 | 99 | A8 | K2 | 30 | $V_{CC}$ | M12 | 72 | VPPD1 |
| B10 | 115 | A7 | E12 | 98 | A17 | K3 | 31 | $\overline{\text{TRDY}}$ | M13 | 74 | $\overline{\text{VCCD1}}$ |
| B11 | 111 | A23 | E13 | 97 | A9 | K4 | 41 | AD12 | N1 | 37 | C/$\overline{\text{BE1}}$ |
| B12 | 108 | A16 | F1 | 16 | AD22 | K5 | 45 | AD10 | N2 | 38 | AD15 |
| B13 | 107 | A21 | F2 | 15 | AD23 | K6 | 49 | AD7 | N3 | 40 | AD13 |
| C1 | 4 | AD30 | F3 | 14 | $V_{CC}$ | K7 | 56 | AD1 | N4 | 44 | $V_{CCP}$ |
| C2 | 3 | AD31 | F4 | 13 | IDSEL | K8 | 60 | MFUNC0 | N5 | 48 | C/$\overline{\text{BE0}}$ |
| C3 | 142 | D9 | F10 | 96 | $\overline{\text{IOWR}}$ | K9 | 64 | MFUNC2 | N6 | 52 | AD5 |
| C4 | 139 | D0 | F11 | 95 | A11 | K10 | 77 | D11 | N7 | 54 | AD3 |
| C5 | 135 | BVD1($\overline{\text{STSCHG}}$/$\overline{\text{RI}}$) | F12 | 94 | GND | K11 | 78 | GND | N8 | 57 | AD0 |
| C6 | 131 | $\overline{\text{VS1}}$ | F13 | 93 | $\overline{\text{IORD}}$ | K12 | 79 | D4 | N9 | 61 | MFUNC1 |
| C7 | 127 | A2 | G1 | 18 | $V_{CCP}$ | K13 | 80 | D12 | N10 | 65 | MFUNC3 |
| C8 | 122 | $V_{CC}$ | G2 | 17 | AD21 | L1 | 32 | $\overline{\text{DEVSEL}}$ | N11 | 68 | MFUNC5 |
| C9 | 118 | A6 | G3 | 19 | AD20 | L2 | 33 | $\overline{\text{STOP}}$ | N12 | 71 | VPPD0 |
| C10 | 114 | GND | G4 | 20 | $\overline{\text{RST}}$ | L3 | 34 | $\overline{\text{PERR}}$ | N13 | 73 | $\overline{\text{VCCD0}}$ |

† The PGE (LQFP) pin numbers are shown also.

**Table 4. 16-Bit PC Card Signal Names – Sorted Alphabetically**

| SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | | SIGNAL NAME | PIN NO. | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | PGE | GGU | | PGE | GGU | | PGE | GGU | | PGE | GGU |
| A0 | 129 | A6 | AD10 | 45 | K5 | D4 | 79 | K12 | PAR | 36 | M2 |
| A1 | 128 | D7 | AD11 | 43 | M4 | D5 | 81 | J10 | PCLK | 21 | H1 |
| A2 | 127 | C7 | AD12 | 41 | K4 | D6 | 83 | J12 | $\overline{\text{PERR}}$ | 34 | L3 |
| A3 | 124 | A8 | AD13 | 40 | N3 | D7 | 85 | H10 | $\overline{\text{REQ}}$ | 1 | A1 |
| A4 | 121 | D8 | AD14 | 39 | M3 | D8 | 140 | A3 | READY($\overline{\text{IREQ}}$) | 132 | D6 |
| A5 | 120 | A9 | AD15 | 38 | N2 | D9 | 142 | C3 | $\overline{\text{REG}}$ | 125 | B7 |
| A6 | 118 | C9 | AD16 | 26 | J2 | D10 | 144 | B2 | RESET | 119 | B9 |
| A7 | 115 | B10 | AD17 | 25 | J1 | D11 | 77 | K10 | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ | 59 | L8 |
| A8 | 99 | E11 | AD18 | 24 | H4 | D12 | 80 | K13 | $\overline{\text{RST}}$ | 20 | G4 |
| A9 | 97 | E13 | AD19 | 23 | H3 | D13 | 82 | J11 | $\overline{\text{SERR}}$ | 35 | M1 |
| A10 | 89 | G12 | AD20 | 19 | G3 | D14 | 84 | J13 | SPKROUT | 62 | M9 |
| A11 | 95 | F11 | AD21 | 17 | G2 | D15 | 87 | H12 | $\overline{\text{STOP}}$ | 33 | L2 |
| A12 | 112 | A11 | AD22 | 16 | F1 | $\overline{\text{DEVSEL}}$ | 32 | L1 | $\overline{\text{SUSPEND}}$ | 70 | L11 |
| A13 | 101 | D13 | AD23 | 15 | F2 | $\overline{\text{FRAME}}$ | 28 | J4 | $\overline{\text{TRDY}}$ | 31 | K3 |
| A14 | 104 | C13 | AD24 | 11 | E2 | GND | 6 | D3 | $V_{CC}$ | 14 | F3 |
| A15 | 110 | A12 | AD25 | 10 | E3 | GND | 22 | H2 | $V_{CC}$ | 30 | K2 |
| A16 | 108 | B12 | AD26 | 9 | E4 | GND | 42 | L4 | $V_{CC}$ | 50 | L6 |
| A17 | 98 | E12 | AD27 | 8 | D1 | GND | 58 | M8 | $V_{CC}$ | 66 | M10 |
| A18 | 100 | E10 | AD28 | 7 | D2 | GND | 78 | K11 | $V_{CC}$ | 86 | H11 |
| A19 | 103 | D11 | AD29 | 5 | D4 | GND | 94 | F12 | $V_{CC}$ | 102 | D12 |
| A20 | 105 | C12 | AD30 | 4 | C1 | GND | 114 | C10 | $V_{CC}$ | 122 | C8 |
| A21 | 107 | B13 | AD31 | 3 | C2 | GND | 130 | B6 | $V_{CC}$ | 138 | B4 |
| A22 | 109 | A13 | BVD1($\overline{\text{STSCHG}}$/$\overline{\text{RI}}$) | 135 | C5 | $\overline{\text{GNT}}$ | 2 | B1 | $V_{CCCB}$ | 90 | G13 |
| A23 | 111 | B11 | BVD2($\overline{\text{SPKR}}$) | 134 | B5 | IDSEL | 13 | F4 | $V_{CCCB}$ | 126 | A7 |
| A24 | 113 | D10 | C/$\overline{\text{BE0}}$ | 48 | N5 | $\overline{\text{INPACK}}$ | 123 | B8 | $\overline{\text{VCCD0}}$ | 73 | N13 |
| A25 | 116 | A10 | C/$\overline{\text{BE1}}$ | 37 | N1 | $\overline{\text{IORD}}$ | 93 | F13 | $\overline{\text{VCCD1}}$ | 74 | M13 |
| AD0 | 57 | N8 | C/$\overline{\text{BE2}}$ | 27 | J3 | $\overline{\text{IOWR}}$ | 96 | F10 | $V_{CCI}$ | 63 | L9 |
| AD1 | 56 | K7 | C/$\overline{\text{BE3}}$ | 12 | E1 | $\overline{\text{IRDY}}$ | 29 | K1 | $V_{CCP}$ | 18 | G1 |
| AD2 | 55 | L7 | $\overline{\text{CD1}}$ | 75 | L12 | MFUNC0 | 60 | K8 | $V_{CCP}$ | 44 | N4 |
| AD3 | 54 | N7 | $\overline{\text{CD2}}$ | 137 | A4 | MFUNC1 | 61 | N9 | VPPD0 | 71 | N12 |
| AD4 | 53 | M7 | $\overline{\text{CE1}}$ | 88 | H13 | MFUNC2 | 64 | K9 | VPPD1 | 72 | M12 |
| AD5 | 52 | N6 | $\overline{\text{CE2}}$ | 91 | G11 | MFUNC3 | 65 | N10 | $\overline{\text{VS1}}$ | 131 | C6 |
| AD6 | 51 | M6 | D0 | 139 | C4 | MFUNC4 | 67 | L10 | $\overline{\text{VS2}}$ | 117 | D9 |
| AD7 | 49 | K6 | D1 | 141 | B3 | MFUNC5 | 68 | N11 | $\overline{\text{WAIT}}$ | 133 | A5 |
| AD8 | 47 | M5 | D2 | 143 | A2 | MFUNC6 | 69 | M11 | $\overline{\text{WE}}$ | 106 | C11 |
| AD9 | 46 | L5 | D3 | 76 | L13 | $\overline{\text{OE}}$ | 92 | G10 | WP($\overline{\text{IOIS16}}$) | 136 | D5 |

## Terminal Functions

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. Terminal numbers are shown for both the PGE LQF package and the GGU ball grid array package.

### power supply

| TERMINAL | | | FUNCTION |
|---|---|---|---|
| NAME | PGE NUMBER | GGU NUMBER | |
| GND | 6, 22, 42, 58, 78, 94, 114, 130 | B6, C10, D3, F12, H2, K11, L4, M8 | Device ground terminals |
| $V_{CC}$ | 14, 30, 50, 66, 86, 102, 122, 138 | B4, C8, D12, F3, H11, K2, L6, M10 | Power supply terminal for core logic (3.3 V) |
| $V_{CCCB}$ | 90, 126 | A7, G13 | Clamping voltage for PC Card interface. Indicates card signaling environment of 5 V or 3.3 V. |
| $V_{CCI}$ | 63 | L9 | Clamping voltage for multifunction terminals (5 V or 3.3 V) |
| $V_{CCP}$ | 18, 44 | G1, N4 | Clamping voltage for PCI signaling (5 V or 3.3 V) |

### PC Card power switch

| TERMINAL | | | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| NAME | PIN NUMBER | | | |
| | PGE | GGU | | |
| $\overline{VCCD0}$ | 73 | N13 | O | Logic controls to the TPS2211 PC Card power interface switch to control AVCC. |
| $\overline{VCCD1}$ | 74 | M13 | | |
| VPPD0 | 71 | N12 | | Logic controls to the TPS2211 PC Card power interface switch to control AVPP. |
| VPPD1 | 72 | M12 | | |

### PCI system

| TERMINAL | | | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| NAME | PIN NUMBER | | | |
| | PGE | GGU | | |
| PCLK | 21 | H1 | 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. |
| $\overline{RST}$ | 20 | G4 | I | PCI reset. When the PCI bus reset is asserted, $\overline{RST}$ causes the PCI1211 to place all output buffers in a high-impedance state and reset all internal registers. When $\overline{RST}$ is asserted, the device is completely nonfunctional. After $\overline{RST}$ is deasserted, the PCI1211 is in its default state.<br><br>When $\overline{SUSPEND}$ and $\overline{RST}$ are asserted, the device is protected from $\overline{RST}$ clearing the internal registers. All outputs are placed in a high-impedance state, but the contents of the registers are preserved. |

## Terminal Functions (Continued)

### PCI address and data

| TERMINAL NAME | TERMINAL PIN NUMBER PGE | TERMINAL PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| AD31 | 3 | C2 | 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 | C1 | | |
| AD29 | 5 | D4 | | |
| AD28 | 7 | D2 | | |
| AD27 | 8 | D1 | | |
| AD26 | 9 | E4 | | |
| AD25 | 10 | E3 | | |
| AD24 | 11 | E2 | | |
| AD23 | 15 | F2 | | |
| AD22 | 16 | F1 | | |
| AD21 | 17 | G2 | | |
| AD20 | 19 | G3 | | |
| AD19 | 23 | H3 | | |
| AD18 | 24 | H4 | | |
| AD17 | 25 | J1 | | |
| AD16 | 26 | J2 | | |
| AD15 | 38 | N2 | | |
| AD14 | 39 | M3 | | |
| AD13 | 40 | N3 | | |
| AD12 | 41 | K4 | | |
| AD11 | 43 | M4 | | |
| AD10 | 45 | K5 | | |
| AD9 | 46 | L5 | | |
| AD8 | 47 | M5 | | |
| AD7 | 49 | K6 | | |
| AD6 | 51 | M6 | | |
| AD5 | 52 | N6 | | |
| AD4 | 53 | M7 | | |
| AD3 | 54 | N7 | | |
| AD2 | 55 | L7 | | |
| AD1 | 56 | K7 | | |
| AD0 | 57 | N8 | | |
| C/$\overline{BE3}$ | 12 | E1 | 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 byte enables. The byte enables determine 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/$\overline{BE2}$ | 27 | J3 | | |
| C/$\overline{BE1}$ | 37 | N1 | | |
| C/$\overline{BE0}$ | 48 | N5 | | |
| PAR | 36 | M2 | I/O | PCI bus parity. In all PCI bus read and write cycles, the PCI1211 calculates even parity across the AD31–AD0 and C/$\overline{BE3}$–C/$\overline{BE0}$ buses. As an initiator during PCI cycles, the PCI1211 outputs this parity indicator with a one-PCLK delay. As a target during PCI cycles, the calculated parity is compared to the initiator's parity indicator. A compare error results in the assertion of a parity error ($\overline{PERR}$). |

## Terminal Functions (Continued)

### PCI interface control

| TERMINAL NAME | PIN NUMBER PGE | PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| $\overline{\text{DEVSEL}}$ | 32 | L1 | I/O | PCI device select. The PCI1211 asserts $\overline{\text{DEVSEL}}$ to claim a PCI cycle as the target device. As a PCI initiator on the bus, the PCI1211 monitors $\overline{\text{DEVSEL}}$ until a target responds. If no target responds before timeout occurs, the PCI1211 terminates the cycle with an initiator abort. |
| $\overline{\text{FRAME}}$ | 28 | J4 | 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 | B1 | I | PCI bus grant. $\overline{\text{GNT}}$ is driven by the PCI bus arbiter to grant the PCI1211 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 | 13 | F4 | I | Initialization device select. IDSEL selects the PCI1211 during configuration space accesses. IDSEL can be connected to one of the upper 24 PCI address lines on the PCI bus. |
| $\overline{\text{IRDY}}$ | 29 | K1 | I/O | PCI initiator ready. $\overline{\text{IRDY}}$ indicates the PCI bus initiator's ability 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. |
| $\overline{\text{PERR}}$ | 34 | L3 | 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. |
| $\overline{\text{REQ}}$ | 1 | A1 | O | PCI bus request. $\overline{\text{REQ}}$ is asserted by the PCI1211 to request access to the PCI bus as an initiator. |
| $\overline{\text{SERR}}$ | 35 | M1 | O | PCI system error. $\overline{\text{SERR}}$ is an output that is pulsed from the PCI1211 when enabled through the command register indicating a system error has occurred. The PCI1211 need not be the target of the PCI cycle to assert this signal. When $\overline{\text{SERR}}$ is enabled in the control register, this signal also pulses, indicating that an address parity error has occurred on a CardBus interface. |
| $\overline{\text{STOP}}$ | 33 | L2 | 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. |
| $\overline{\text{TRDY}}$ | 31 | K3 | I/O | PCI target ready. $\overline{\text{TRDY}}$ indicates the primary bus target's ability 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. |

## Terminal Functions (Continued)

### multifunction and miscellaneous pins

| TERMINAL NAME | PIN NUMBER PGE | PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| MFUNC0 | 60 | K8 | I/O | Multifunction Terminal 0. MFUNC0 can be configured as parallel PCI interrupt $\overline{\text{INTA}}$, GPI0, GPO0, $\overline{\text{GPE}}$, socket activity LED output, ZV output select, CardBus audio PWM, or a parallel IRQ. Refer to the *multifunction routing register* description on page 61 for configuration details. |
| MFUNC1 | 61 | N9 | I/O | Multifunction Terminal 1. MFUNC1 can be configured as GPI1, GPO1, $\overline{\text{GPE}}$, socket activity LED output, ZV output select, CardBus audio PWM, or a parallel IRQ. Refer to the *multifunction routing register* description on page 61 for configuration details.<br><br>Serial Data (SDA). When the serial bus mode is implemented by pulling up the SCA and SCL terminals, the MFUNC1 terminal provides the SDA signaling. The two-pin serial interface is used to load the subsystem identification and other register defaults from an EEPROM after a PCI reset. Refer to the *serial bus interface protocol* description on page 30 for details on other serial bus applications. |
| MFUNC2 | 64 | K9 | I/O | Multifunction Terminal 2. MFUNC2 can be configured as PC/PCI DMA Request, GPI2, GPO2, socket activity LED output, ZV output select, CardBus audio PWM, $\overline{\text{GPE}}$, $\overline{\text{RI_OUT}}$, or a parallel IRQ. Refer to the *multifunction routing register* description on page 61 for configuration details. |
| MFUNC3 | 65 | N10 | I/O | Multifunction Terminal 3. MFUNC3 can be configured as a parallel IRQ or the serialized interrupt signal IRQSER. Refer to the *multifunction routing register* description on page 61 for configuration details. |
| MFUNC4 | 67 | L10 | I/O | Multifunction Terminal 4. MFUNC4 can be configured as PCI $\overline{\text{LOCK}}$, GPI3, GPO3, socket activity LED, $\overline{\text{RI_OUT}}$ output, ZV output select, CardBus audio PWM, $\overline{\text{GPE}}$, or a parallel IRQ. Refer to the *multifunction routing register* description on page 61 for configuration details.<br><br>Serial Clock (SCL). When the serial bus mode is implemented by pulling the SDA and SCL terminals, the MFUNC4 terminal provides the SCL signaling. The two-pin serial interface is used to load the subsystem identification and other register defaults from an EEPROM after a PCI reset. Refer to the *serial bus interface protocol* description on page 30 for details on other serial bus applications. |
| MFUNC5 | 68 | N11 | I/O | Multifunction Terminal 5. MFUNC5 can be configured as PC/PCI DMA Grant, GPI4, GPO4, socket activity LED output, ZV output select, CardBus audio PWM, $\overline{\text{GPE}}$, or a parallel IRQ. Refer to the *multifunction routing register* description on page 61 for configuration details. |
| MFUNC6 | 69 | M11 | I/O | Multifunction Terminal 6. MFUNC6 can be configured as a PCI $\overline{\text{CLKRUN}}$ or a parallel IRQ. Refer to the *multifunction routing register* description on page 61 for configuration details. |
| $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ | 59 | L8 | O | Ring Indicate Out and Power Management Event Output. Provides output for either $\overline{\text{RI_OUT}}$ or $\overline{\text{PME}}$ signals. |
| $\overline{\text{SUSPEND}}$ | 70 | L11 | I | Suspend. $\overline{\text{SUSPEND}}$ is used to protect the internal registers from clearing when the $\overline{\text{RST}}$ signal is asserted. See *suspend mode* on page 39 for details. |
| SPKROUT | 62 | M9 | O | Speaker output. SPKROUT is the output to the host system that can carry $\overline{\text{SPKR}}$ or CAUDIO through the PCI1211 from the PC Card interface. SPKROUT is driven as the exclusive-OR combination of card $\overline{\text{SPKR}}$//CAUDIO inputs. |

## Terminal Functions (Continued)

The address and data and interface control terminals for the 16-bit PC Card are shown in the following two tables.

### 16-bit PC Card address and data

| TERMINAL NAME | TERMINAL PIN NUMBER PGE | TERMINAL PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| A25 | 116 | A10 | O | PC Card address. 16-bit PC Card address lines. A25 is the most-significant bit. |
| A24 | 113 | D10 | | |
| A23 | 111 | B11 | | |
| A22 | 109 | A13 | | |
| A21 | 107 | B13 | | |
| A20 | 105 | C12 | | |
| A19 | 103 | D11 | | |
| A18 | 100 | E10 | | |
| A17 | 98 | E12 | | |
| A16 | 108 | B12 | | |
| A15 | 110 | A12 | | |
| A14 | 104 | C13 | | |
| A13 | 101 | D13 | | |
| A12 | 112 | A11 | | |
| A11 | 95 | F11 | | |
| A10 | 89 | G12 | | |
| A9 | 97 | E13 | | |
| A8 | 99 | E11 | | |
| A7 | 115 | B10 | | |
| A6 | 118 | C9 | | |
| A5 | 120 | A9 | | |
| A4 | 121 | D8 | | |
| A3 | 124 | A8 | | |
| A2 | 127 | C7 | | |
| A1 | 128 | D7 | | |
| A0 | 129 | A6 | | |
| D15 | 87 | H12 | I/O | PC Card data. 16-bit PC Card data lines. D15 is the most-significant bit. |
| D14 | 84 | J13 | | |
| D13 | 82 | J11 | | |
| D12 | 80 | K13 | | |
| D11 | 77 | K10 | | |
| D10 | 144 | B2 | | |
| D9 | 142 | C3 | | |
| D8 | 140 | A3 | | |
| D7 | 85 | H10 | | |
| D6 | 83 | J12 | | |
| D5 | 81 | J10 | | |
| D4 | 79 | K12 | | |
| D3 | 76 | L13 | | |
| D2 | 143 | A2 | | |
| D1 | 141 | B3 | | |
| D0 | 139 | C4 | | |

## Terminal Functions (Continued)

### 16-bit PC Card interface control

| TERMINAL NAME | PIN NUMBER PGE | PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| BVD1 (STSCHG/RI) | 135 | C5 | 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 kept 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 *ExCA card status-change interrupt configuration register* on page 89 for enable bits. See *ExCA card status-change register* on page 88 and the *ExCA interface status register* on page 85 for the status bits for this signal.<br>Status change. 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. RI is used by 16-bit modem cards to indicate a ring detection. |
| BVD2 (SPKR) | 134 | B5 | 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 *ExCA card status-change interrupt configuration register* on page 89 for enable bits. See *ExCA card status-change register* on page 88 and the *interface status register* on page 85 for the status bits for this signal.<br>Speaker. SPKR is an optional binary audio signal available only when the card and socket have been configured for the 16-bit I/O interface. The audio signals from cards A and B are combined by the PCI1211 and are output on SPKROUT.<br>DMA request. BVD2 can be used as the DMA request signal during DMA operations to a 16-bit PC Card that supports DMA. The PC Card asserts BVD2 to indicate a request for a DMA operation. |
| CD1<br>CD2 | 75<br>137 | L12<br>A4 | I | PC Card detect 1 and PC Card detect 2. CD1 and CD2 are internally connected to ground on the PC Card. When a PC Card is inserted into a socket, CD1 and CD2 are pulled low. For signal status, see *interface status register.* |
| CE1<br>CE2 | 88<br>91 | H13<br>G11 | O | Card enable 1 and card enable 2. CE1 and CE2 enable even- and odd-numbered address bytes. CE1 enables even-numbered address bytes, and CE2 enables odd-numbered address bytes. |
| INPACK | 123 | B8 | I | Input acknowledge. INPACK is asserted by the PC Card when it can respond to an I/O read cycle at the current address.<br>DMA request. INPACK can be used as the DMA request signal during DMA operations from a 16-bit PC Card that supports DMA. If used as a strobe, the PC Card asserts this signal to indicate a request for a DMA operation. |
| IORD | 93 | F13 | O | I/O read. IORD is asserted by the PCI1211 to enable 16-bit I/O PC Card data output during host I/O read cycles.<br>DMA write. IORD is used as the DMA write strobe during DMA operations from a 16-bit PC Card that supports DMA. The PCI1211 asserts IORD during DMA transfers from the PC Card to host memory. |
| IOWR | 96 | F10 | O | I/O write. IOWR is driven low by the PCI1211 to strobe write data into 16-bit I/O PC Cards during host I/O write cycles.<br>DMA read. IOWR is used as the DMA write strobe during DMA operations from a 16-bit PC Card that supports DMA. The PCI1211 asserts IOWR during transfers from host memory to the PC Card. |
| OE | 92 | G10 | O | Output enable. OE is driven low by the PCI1211 to enable 16-bit memory PC Card data output during host memory read cycles.<br>DMA terminal count. OE is used as terminal count (TC) during DMA operations to a 16-bit PC Card that supports DMA. The PCI1211 asserts OE to indicate TC for a DMA write operation. |

## Terminal Functions (Continued)

### 16-bit PC Card interface control (continued)

| TERMINAL NAME | PIN NUMBER PGE | PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| READY<br>($\overline{\text{IREQ}}$) | 132 | D6 | 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. |
| $\overline{\text{REG}}$ | 125 | B7 | 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.<br>DMA acknowledge. $\overline{\text{REG}}$ is used as a DMA acknowledge ($\overline{\text{DACK}}$) during DMA operations to a 16-bit PC Card that supports DMA. The PCI1211 asserts $\overline{\text{REG}}$ to indicate a DMA operation. $\overline{\text{REG}}$ is used in conjunction with the DMA read ($\overline{\text{IOWR}}$) or DMA write ($\overline{\text{IORD}}$) strobes to transfer data. |
| RESET | 119 | B9 | O | PC Card reset. RESET forces a hard reset to a 16-bit PC Card. |
| $\overline{\text{WAIT}}$ | 133 | A5 | I | Bus cycle wait. $\overline{\text{WAIT}}$ is driven by a 16-bit PC Card to delay the completion of (i.e., extend) the memory or I/O cycle in progress. |
| $\overline{\text{WE}}$ | 106 | C11 | O | Write enable. $\overline{\text{WE}}$ is used to strobe memory write data into 16-bit memory PC Cards. $\overline{\text{WE}}$ is also used for memory PC Cards that employ programmable memory technologies.<br>DMA terminal count. $\overline{\text{WE}}$ is used as $\overline{\text{TC}}$ during DMA operations to a 16-bit PC Card that supports DMA. The PC1211 asserts $\overline{\text{WE}}$ to indicate $\overline{\text{TC}}$ for a DMA read operation. |
| WP<br>($\overline{\text{IOIS16}}$) | 136 | D5 | 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 ($\overline{\text{IOIS16}}$) function.<br>I/O is 16 bits. $\overline{\text{IOIS16}}$ applies to 16-bit I/O PC Cards. $\overline{\text{IOIS16}}$ 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.<br>DMA request. WP can be used as the DMA request signal during DMA operations to a 16-bit PC Card that supports DMA. If used, the PC Card asserts WP to indicate a request for a DMA operation. |
| $\overline{\text{VS1}}$<br>$\overline{\text{VS2}}$ | 131<br>117 | C6<br>D9 | I/O | Voltage sense 1 and voltage sense 2. $\overline{\text{VS1}}$ and $\overline{\text{VS2}}$, when used in conjunction with each other, determine the operating voltage of the 16-bit PC Card. |

## Terminal Functions (Continued)

The interface system, address and data, and interface control terminals for the CardBus PC Card system are shown in the following three tables.

### CardBus PC Card interface system

| TERMINAL NAME | TERMINAL PIN NUMBER PGE | TERMINAL PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| CCLK | 108 | B12 | O | CardBus PC Card clock. CCLK provides synchronous timing for all transactions on the CardBus interface. All signals except $\overline{\text{CRST}}$, $\overline{\text{CLKRUN}}$, $\overline{\text{CINT}}$, CSTSCHG, CAUDIO, $\overline{\text{CCD1}}$, $\overline{\text{CCD2}}$, and CVS2–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. |
| $\overline{\text{CCLKRUN}}$ | 136 | D5 | O | CardBus PC Card clock run. $\overline{\text{CCLKRUN}}$ is used by a CardBus PC Card to request an increase in the CCLK frequency, and by the PCI1211 to indicate that the CCLK frequency is going to be decreased. |
| $\overline{\text{CRST}}$ | 119 | B9 | I/O | CardBus PC Card reset. $\overline{\text{CRST}}$ is used to bring CardBus PC Card-specific registers, sequencers, and signals to a known state. When $\overline{\text{CRST}}$ is asserted, all CardBus PC Card signals must be 3-stated, and the PCI1211 drives these signals to a valid logic level. Assertion can be asynchronous to CCLK, but deassertion must be synchronous to CCLK. |

## Terminal Functions (Continued)

### CardBus PC Card address and data

| TERMINAL NAME | PIN NUMBER PGE | PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| CAD31 | 144 | B2 | I/O | PC Card 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 | C3 | | |
| CAD29 | 141 | B3 | | |
| CAD28 | 140 | A3 | | |
| CAD27 | 139 | C4 | | |
| CAD26 | 129 | A6 | | |
| CAD25 | 128 | D7 | | |
| CAD24 | 127 | C7 | | |
| CAD23 | 124 | A8 | | |
| CAD22 | 121 | D8 | | |
| CAD21 | 120 | A9 | | |
| CAD20 | 118 | C9 | | |
| CAD19 | 116 | A10 | | |
| CAD18 | 115 | B10 | | |
| CAD17 | 113 | D10 | | |
| CAD16 | 98 | E12 | | |
| CAD15 | 96 | F10 | | |
| CAD14 | 97 | E13 | | |
| CAD13 | 93 | F13 | | |
| CAD12 | 95 | F11 | | |
| CAD11 | 92 | G10 | | |
| CAD10 | 91 | G11 | | |
| CAD9 | 89 | G12 | | |
| CAD8 | 87 | H12 | | |
| CAD7 | 85 | H10 | | |
| CAD6 | 82 | J11 | | |
| CAD5 | 83 | J12 | | |
| CAD4 | 80 | K13 | | |
| CAD3 | 81 | J10 | | |
| CAD2 | 77 | K10 | | |
| CAD1 | 79 | K12 | | |
| CAD0 | 76 | L13 | | |
| CC/$\overline{\text{BE3}}$ | 12 | E1 | 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/$\overline{\text{BE0}}$ defines the bus command. During the data phase, this 4-bit bus is used as byte enables. The byte enables determine 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 3 (CAD31–CAD24). |
| CC/$\overline{\text{BE2}}$ | 27 | J3 | | |
| CC/$\overline{\text{BE1}}$ | 37 | N1 | | |
| CC/$\overline{\text{BE0}}$ | 48 | N5 | | |
| CPAR | 101 | D13 | I/O | CardBus parity. In all CardBus read and write cycles, the PCI1211 calculates even parity across the CAD and CC/$\overline{\text{BE}}$ buses. As an initiator during CardBus cycles, the PCI1211 outputs CPAR with a one-CCLK delay. As a target during CardBus cycles, the calculated parity is compared to the initiator's parity indicator; a compare error results in a parity error assertion. |

## Terminal Functions (Continued)

### CardBus PC Card interface control

| TERMINAL NAME | PIN NUMBER PGE | PIN NUMBER GGU | I/O TYPE | FUNCTION |
|---|---|---|---|---|
| CAUDIO | 134 | B5 | I | CardBus audio. CAUDIO is a digital input signal from a PC Card to the system speaker. The PCI1211 supports the binary audio mode and outputs a binary signal from the card to SPKROUT. |
| $\overline{\text{CBLOCK}}$ | 103 | D11 | 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>137 | L12<br>A4 | 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. |
| $\overline{\text{CDEVSEL}}$ | 107 | B13 | I/O | CardBus device select. The PCI1211 asserts $\overline{\text{CDEVSEL}}$ to claim a CardBus cycle as the target device. As a CardBus initiator on the bus, the PCI1211 monitors $\overline{\text{CDEVSEL}}$ until a target responds. If no target responds before timeout occurs, the PCI1211 terminates the cycle with an initiator abort. |
| $\overline{\text{CFRAME}}$ | 111 | B11 | 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. When $\overline{\text{CFRAME}}$ is deasserted, the CardBus bus transaction is in the final data phase. |
| $\overline{\text{CGNT}}$ | 106 | C11 | I | CardBus bus grant. $\overline{\text{CGNT}}$ is driven by the PCI1211 to grant a CardBus PC Card access to the CardBus bus after the current data transaction has been completed. |
| $\overline{\text{CINT}}$ | 132 | D6 | 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 | A12 | I/O | CardBus initiator ready. $\overline{\text{CIRDY}}$ indicates the CardBus initiator's ability 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. |
| $\overline{\text{CPERR}}$ | 104 | C13 | I/O | CardBus parity error. $\overline{\text{CPERR}}$ is used to report parity errors during CardBus transactions, except during special cycles. It is driven low by a target two clocks following that data when a parity error is detected. |
| $\overline{\text{CREQ}}$ | 123 | B8 | 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 | A5 | 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, and may take several CCLK periods. The PCI1211 can report $\overline{\text{CSERR}}$ to the system by assertion of $\overline{\text{SERR}}$ on the PCI interface. |
| $\overline{\text{CSTOP}}$ | 105 | C12 | 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 | C5 | I | CardBus status change. CSTSCHG is used to alert the system to a change in the card's status, and is used as a wake-up mechanism. |
| $\overline{\text{CTRDY}}$ | 109 | A13 | I/O | CardBus target ready. $\overline{\text{CTRDY}}$ indicates the CardBus target's ability 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 | 131<br>117 | C6<br>D9 | 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. |

## power supply sequencing

The PCI1211 contains 3.3-V I/O buffers with 5-V tolerance requiring a core power supply and clamp voltage. The core power supply is always 3.3 V. The clamp voltage 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. Apply 3.3-V power to the core.
2. Assert $\overline{\text{PRST}}$ 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 supply.
3. Apply the clamp voltage.

The power-down sequence is:

1. Use $\overline{\text{PRST}}$ to switch outputs to a high-impedance state.
2. Remove the clamp voltage.
3. Remove the 3.3-V power from the core.

## I/O characteristics

Figure 1 shows a 3-state bidirectional buffer. The *recommended operating conditions* table, on page 119, provides the electrical characteristics of the inputs and outputs.

**NOTE:**
The PCI1211 meets the ac specifications of the *1997 PC Card Standard* and *PCI Local Bus Specification Revision 2.2.*

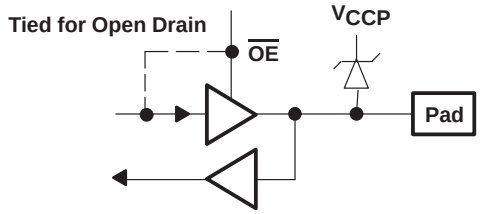



**Figure 1. 3-State Bidirectional Buffer**

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

## clamping voltages

The clamping voltages are set to match whatever external environment the PCI1211 will be working with: 3.3 V or 5 V. The I/O sites can be pulled through a clamping diode to a voltage 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 PCI1211 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, $V_{CCP}$ can be connected to a 5-V power supply.

The PCI1211 requires three separate clamping voltages because it supports a wide range of features. The three voltages are listed and defined in the *recommended operating conditions*, on page 119.

## peripheral component interconnect (PCI) interface

The PCI1211 is fully compliant with the *PCI Local Bus Specification, Revision 2.2*. The PCI1211 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 $V_{CCP}$ terminals to the desired voltage level. In addition to the mandatory PCI signals the PCI1211 provides the optional interrupt signal $\overline{\text{INTA}}$.

### PCI bus lock (LOCK)

The bus-locking protocol defined in the PCI specification is not highly recommended, but is provided on the PCI1211 as an additional compatibility feature. The PCI $\overline{\text{LOCK}}$ signal can be routed to the MFUNC4 terminal via the multifunction routing register, see the *multifunction routing register description* on page 61 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 lock protocol defined by PCI 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 will 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 PCI1211 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*, *Revision 2.2*, and the issue is resolved by the PCI master using $\overline{\text{LOCK}}$.

### loading subsystem identification

The subsystem vendor ID register and subsystem ID register make up a doubleword of PCI configuration space located at offset 40h. 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 '97 requirement.

The PCI1211 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 setting the SUBSYSRW bit in the system control register (bit 5, at PCI offset 80h). Once this bit is set, the BIOS can write a subsystem identification value into the registers at offset 40h. The BIOS must clear 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).

### loading subsystem identification (continued)

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 PCI1211 loads the data from the serial EEPROM after a reset of the primary bus. The $\overline{\text{SUSPEND}}$ input gates the PCI reset from the entire PCI1211 core, including the serial bus state machine (see *suspend mode*, on page 39, for details on using $\overline{\text{SUSPEND}}$).

The PCI1211 provides a two-line serial bus host controller that can be used to interface to a serial EEPROM. Refer to *serial bus interface* on page 30 for details on the two-wire serial bus controller and applications.

## PC Card applications

This section describes the PC Card interfaces of the PCI1211. Discussions are provided for:

- Card insertion/removal and recognition
- $P^2C$ power-switch interface
- Zoom video support
- Speaker and audio applications
- LED socket activity indicator
- PC Card 16-distributed DMA support
- PC Card controller programming model
- CardBus socket registers

### PC Card insertion/removal and recognition

The 1997 PC Card Standard addresses the card-detection and recognition process through an interrogation procedure that the socket must initiate on card insertion into a cold, unpowered socket. Through this interrogation, card voltage requirements and interface (16-bit versus CardBus) are determined.

The scheme uses the $\overline{\text{CD1}}$, $\overline{\text{CD2}}$, $\overline{\text{VS1}}$, and $\overline{\text{VS2}}$ signals ($\overline{\text{CCD1}}$, $\overline{\text{CCD2}}$, CVS1, and CVS2 for CardBus). 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 1995 PC Card Standard and is shown in Table 5.

**Table 5. 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 | Y.Y V |
| Connect to CVS2 | Ground | Connect to $\overline{\text{CCD2}}$ | Open | LV | CardBus PC Card | Y.Y 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 | | |

### P$^2$C power-switch interface (TPS2211)

The PCI1211 provides a P$^2$C (PCMCIA peripheral control) interface for control of the PC Card power switch. The $\overline{\text{VCCD}}$ and VPPD terminals are used with the TI TPS2211 single slot PC Card power interface switch to provide power switch support. Figure 2 shows the terminal assignments for the TPS2211. Figure 3 illustrates a typical application, where the PCI1211 represents the PC Card controller.

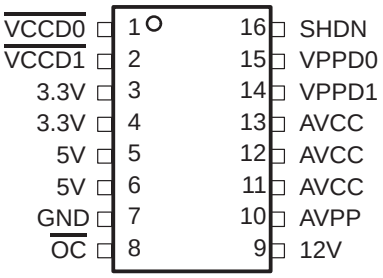



**Figure 2. TPS2211 Terminal Assignments**

The PCI1211 also includes support for the Maxim 1602 single-channel CardBus and PCMCIA power-switching network. Application of this power switch would be similar to the TPS2211.

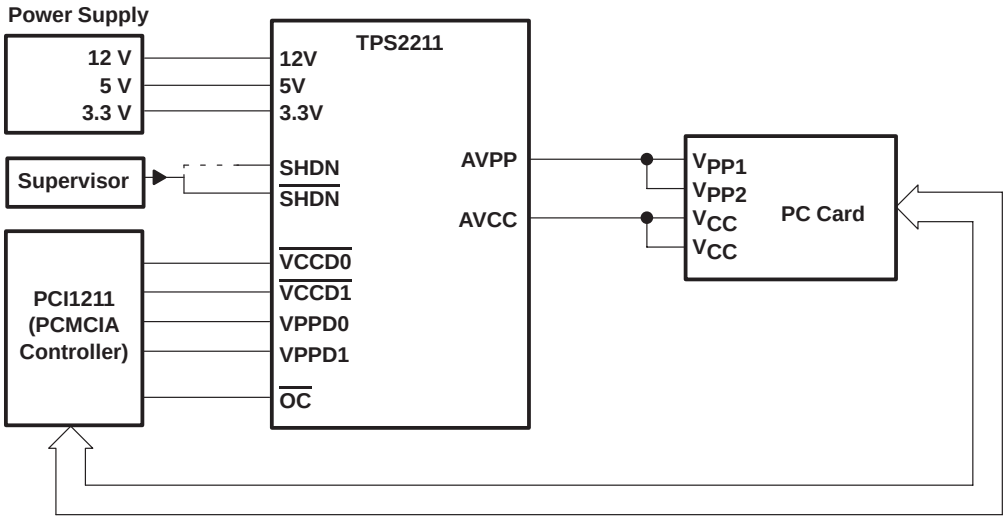



**Figure 3. TPS2211 Typical Application**

### zoom video support

The PCI1211 allows for the implementation of zoom video for PC Cards. Zoom video is supported by setting the ZVENABLE bit in the card control register. Setting this bit puts PC Card-16 address lines A25–A4 of the PC Card interface in the high-impedance state. These lines can then be used to transfer video and audio data directly to the appropriate controller. Card address lines A3–A0 can still be used to access PC Card CIS registers for PC Card configuration. Figure 4 illustrates a PCI1211 ZV implementation.

**zoom video support (continued)**

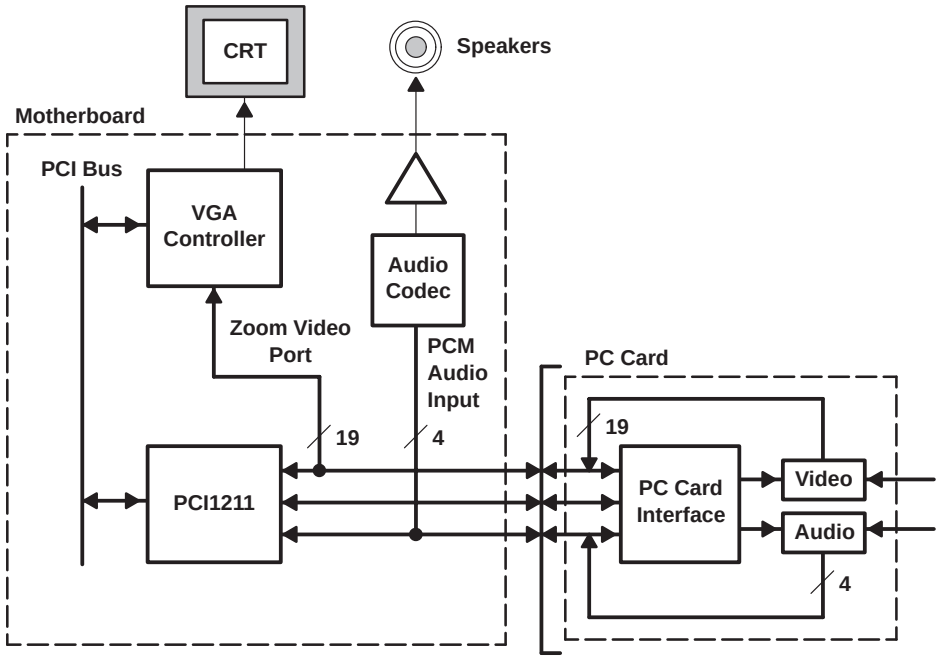



**Figure 4. Zoom Video Implementation Using PCI1211**

Not shown in Figure 4 is the multiplexing scheme used to route either a socket ZV source or an external ZV source to the graphics controller. A typical external source might be provided from a high-speed serial bus like IEEE1394. The PCI1211 provides ZVSTAT, $\overline{\text{ZVSEL0}}$ signals on the multifunction terminals to switch external bus drivers. Figure 5 shows an implementation for switching between two ZV streams using external logic.

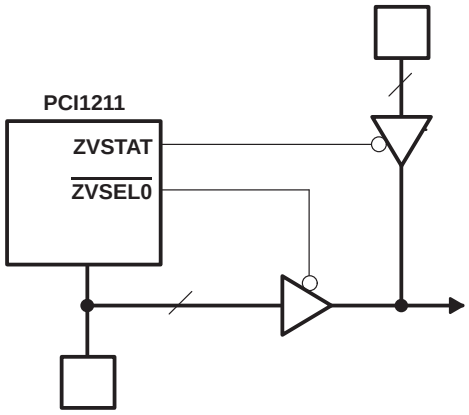



**Figure 5. Zoom Video Switching Application**

The example shown in Figure 5 illustrates an implementation using standard 3-state bus drivers with active-low output enables. $\overline{\text{ZVSEL0}}$ is an active-low output indicating that the Socket ZV mode is enabled. ZVSTAT is an active-high output indicating the PCI1211 socket is enabled for ZV mode. The implementation shown in Figure 5 can be used if PC Card ZV is prioritized over other sources.

### 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 pin 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 PCI1211. The CardBus CAUDIO signal also can pass a single-amplitude binary waveform. The binary audio signals from the PC Card socket is used in the PCI1211 to produce SPKROUT. This output is enabled by the SPKROUTEN bit in the card control register.

Older controllers support CAUDIO in binary or PWM mode but use the same pin (SPKROUT). Some audio chips may not support both modes on one pin and may have a separate pin for binary and PWM. The PCI1211 implementation includes a signal for PWM, CAUDPWM, which can be routed to a MFUNC terminal. The AUD2MUX bit located in the card control register is programmed to route a CardBus CAUDIO PWM terminal to CAUDPWM. Refer to the *multifunction routing register* description on page 61 for details on configuring the MFUNC terminals.

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

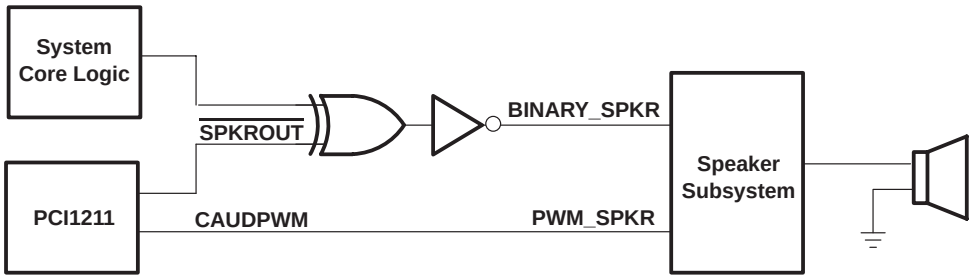



**Figure 6. Sample Application of SPKROUT and CAUDPWM**

### LED socket activity indicators

A socket activity LED indicates 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. Refer to the *multifunction routing register* description on page 61 for details on configuring the multifunction terminals.

The LED signal is active high and is driven for 64-ms durations. When the LED is not being driven high, it is driven to a low state. Either of the two circuits shown in Figure 7 can be implemented to provide LED signaling, and it is left for the board designer to 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 $\overline{\text{CFRAME}}$, $\overline{\text{CIRDY}}$, or $\overline{\text{CREQ}}$ is active.

**LED socket activity indicators (continued)**

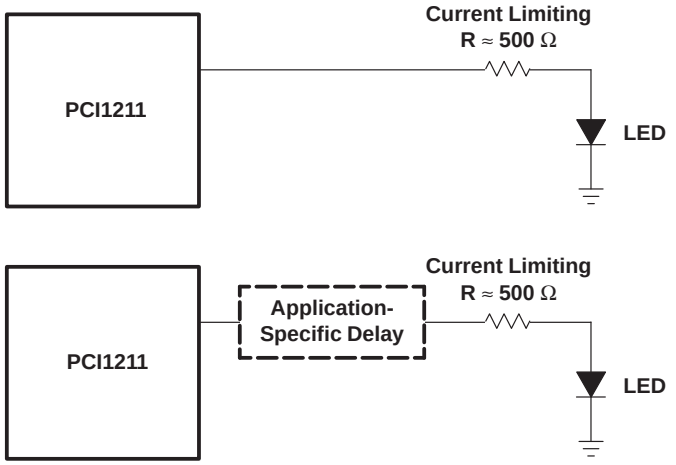



**Figure 7. Two Sample LED Circuits**

As indicated, the LED signal is driven for 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 $\overline{\text{CLKRUN}}$ protocol, or when in the D2 or D1 power state.

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

**PC Card16 Distributed DMA support**

The PCI1211 supports a distributed DMA slave engine for 16-bit PC Card DMA support. The distributed DMA (DDMA) slave register set provides the programmability necessary for the slave DDMA engine. Table 6 shows the DDMA register configuration.

Two critical PCI configuration header registers for DDMA are the socket DMA register 0 and the socket DMA register 1. Distributed DMA is enabled through socket DMA register 0 and the contents of this register configure the PC Card-16 terminal ($\overline{\text{SPKR}}$, $\overline{\text{IOIS16}}$, or $\overline{\text{INPACK}}$) which is used for the DMA request signal, $\overline{\text{DREQ}}$. The base address of the DDMA slave registers and the transfer size (bytes or words) are programmed through the socket DMA register 1. Refer to the *PC Card controller programming model* on page 43 and the accompanying register descriptions for details.

**Table 6. Distributed DMA Registers**

| TYPE | REGISTER NAME | | | | DMA BASE ADDRESS OFFSET (HEX) |
|---|---|---|---|---|---|
| R | Reserved | Page | Current address | | 00 |
| W | | | Base address | | |
| R | Reserved | Reserved | Current count | | 04 |
| W | | | Base count | | |
| R | N/A | Reserved | N/A | Status | 08 |
| W | Mode | | Request | Command | |
| R | Multichannel | Reserved | N/A | Reserved | 0C |
| W | Mask | | Master clear | | |

**PC Card16 Distributed DMA support (continued)**

The DDMA registers contain control and status information consistent with the 8237 DMA controller; however, the register locations are reordered and expanded in some cases. While the DDMA register definitions are identical to those in the 8237 DMA controller of the same name, some register bits defined in the 8237 DMA controller do not apply to distributed DMA in a PCI environment. In such cases, the PCI1211 implements these obsolete register bits as read-only, nonfunctional bits. The reserved registers shown in Table 6 are implemented as read-only and return zeros when read. Writes to reserved registers have no effect.

The DDMA transfer is prefaced by several configuration steps that are specific to the PC Card and must be completed after the PC Card is inserted and interrogated. These steps include setting the proper $\overline{\text{DREQ}}$ signal assignment, setting the data transfer width, and mapping and enabling the DDMA register set. As discussed above, this is done through socket DMA register 0 and socket DMA register 1. The DMA register set is then programmed similarly to an 8237 controller, and the PCI1211 awaits a $\overline{\text{DREQ}}$ assertion from the PC Card requesting a DMA transfer.

DMA writes transfer data from the PC Card to PCI memory addresses. The PCI1211 accepts data 8 or 16 bits at a time, depending on the programmed data width, and then requests access to the PCI bus by asserting its $\overline{\text{REQ}}$ signal. Once granted, the PCI bus and the bus returns to an idle state. The PCI1211 initiates a PCI memory write command to the current memory address and transfers the data in a single data phase. After terminating the PCI cycle, the PCI1211 accepts the next byte(s) from the PC Card until the transfer count expires.

DMA reads transfer data from PCI memory addresses to the PC Card application. Upon the assertion of $\overline{\text{DREQ}}$, the PCI1211 asserts $\overline{\text{REQ}}$ to acquire the PCI bus. Once granted the bus and the bus is idle, the PCI1211 initiates a PCI memory read operation to the current memory address and accepts 8 or 16 bits of data, depending on the programmed data width. After terminating the PCI cycle, the data is passed on to the PC Card. After terminating the PC Card cycle, the PCI1211 requests access to the PCI bus again until the transfer count has expired.

The PCI1211 target interface acts normally during this procedure, and accepts I/O reads and writes to the DDMA registers. While a DDMA transfer is in progress and the host resets the DMA channel, the PCI1211 asserts TC and ends the PC Card cycle(s). TC is indicated in the DDMA status register. At the PC Card interface, the PCI1211 supports demand mode transfers. The PCI1211 asserts DACK during the transfer unless $\overline{\text{DREQ}}$ is deasserted before TC. TC is mapped to the $\overline{\text{OE}}$ PC Card terminal for DMA write operations, and is mapped to $\overline{\text{WE}}$ PC Card terminal for DMA read operations. The DACK signal is mapped to the PC Card $\overline{\text{REG}}$ signal in all transfers, and the $\overline{\text{DREQ}}$ terminal is routed to one of three options which is programmed through socket DMA register 0.

**PC Card-16 PC/PCI DMA**

Some chipsets provide a way for legacy I/O devices to do DMA transfers on the PCI bus. In the PC/PCI DMA protocol, the PCI1211 acts as a PCI target device to certain DMA related I/O addresses. The PCI1211 $\overline{\text{PCREQ}}$ and $\overline{\text{PCGNT}}$ signals are provided as a point-to-point connection to a chipset supporting PC/PCI DMA. The $\overline{\text{PCREQ}}$ and $\overline{\text{PCGNT}}$ signals may be routed to the MFUNC2 and MFUNC5 terminals, respectively. Refer to the *multifunction routing register* description on page 61 for details on configuring the multifunction terminals.

Under the PC/PCI protocol, a PCI DMA slave device (such as the PCI1211) requests a DMA transfer on a particular channel using a serialized protocol on $\overline{\text{PCREQ}}$. The I/O DMA bus master arbitrates for the PCI bus, and grants the channel through a serialized protocol on $\overline{\text{PCGNT}}$ when it is ready for the transfer. The I/O cycle and memory cycles are then presented on the PCI bus which perform the DMA transfers similarly to legacy DMA master devices.

PC/PCI DMA is enabled for the PC Card-16 slot by setting bit 19 in the respective system control register. On power-up this bit is reset and the card PC/PCI DMA is disabled. Bit 3 of the system control register is a global enable for PC/PCI DMA, and is set at power-up and never cleared if the PC/PCI DMA mechanism is implemented. The desired DMA channel for the PC Card-16 slot must be configured through bits 18–16 in the system control register. The channels are configured as indicated in Table 7.

**Table 7. PC/PCI Channel Assignments**

| SYSTEM CONTROL REGISTER | | | DMA CHANNEL | CHANNEL TRANSFER DATA WIDTH |
|---|---|---|---|---|
| BIT 18 | BIT 17 | BIT16 | | |
| 0 | 0 | 0 | Channel 0 | 8-bit DMA transfers |
| 0 | 0 | 1 | Channel 1 | 8-bit DMA transfers |
| 0 | 1 | 0 | Channel 2 | 8-bit DMA transfers |
| 0 | 1 | 1 | Channel 3 | 8-bit DMA transfers |
| 1 | 0 | 0 | Channel 4 | Not used |
| 1 | 0 | 1 | Channel 5 | 16-bit DMA transfers |
| 1 | 1 | 0 | Channel 6 | 16-bit DMA transfers |
| 1 | 1 | 1 | Channel 7 | 16-bit DMA transfers |

As in distributed DMA, the PC Card terminal mapped to $\overline{\text{DREQ}}$ must be configured through socket DMA register 0. The data transfer width is a function of channel number, and the DDMA slave registers are not used. When a $\overline{\text{DREQ}}$ is received from a PC Card, and the channel has been granted, the PCI1211 decodes the I/O addresses listed in Table 8 and performs actions dependent upon the address.

**Table 8. I/O Addresses Used for PC/PCI DMA**

| DMA I/O ADDRESS | DMA CYCLE TYPE | TERMINAL COUNT | PCI CYCLE TYPE |
|---|---|---|---|
| 00h | Normal | 0 | I/O read/write |
| 04h | Normal TC | 1 | I/O read/write |
| C0h | Verify | 0 | I/O read |
| C4h | Verify TC | 1 | I/O read |

The PC/PCI DMA as a PC Card-16 DMA mechanism may not provide the performance levels of DDMA; however, the design of a PCI target implementing PC/PCI DMA is considerably less complex. No bus master state machine is required to support PC/PCI DMA since the DMA control is centralized in the chipset. This DMA scheme is often referred to as centralized DMA for this reason.

### CardBus socket registers

The PCI1211 contains all registers for compatibility with the latest PCI-to-PCMCIA CardBus bridge specification. These registers exist as the CardBus socket registers, and are listed in Table 9.

**Table 9. 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 |
| Reserved | 18h |
| Reserved | 1Ch |
| Socket power management | 20h |

## serial bus interface

The PCI1211 provides a serial bus interface to accommodate loading subsystem identification and select register defaults through a serial EEPROM. The PCI1211 serial bus interface is compatible with various $I^2C$ and SMBus components.

### serial bus interface implementation

The PCI1211 defaults to serial bus interface disabled. To enable the serial interface, appropriate pullup resistors must be implemented on the SDA and SCL signals, i.e., the MFUNC1 and MFUNC4 terminals. In addition, pullup resistors must be implemented on $\overline{\text{VCCD0}}$ and $\overline{\text{VCCD1}}$. When the interface is detected, the SBDETECT bit in the system control register is set. The SBDETECT bit is cleared by a write back of 1.

The PCI1211 implements a two-pin serial interface with one clock signal (SCL) and one data signal (SDA). The SCL signal is mapped to the MFUNC4 terminal and the SDA signal is mapped to the MFUNC1 terminal. The PCI1211 drives SCL at nearly 100 kHz during data transfers, which is the maximum specified frequency for standard mode $I^2C$. Figure 8 illustrates an example application implementing the two-wire serial bus.

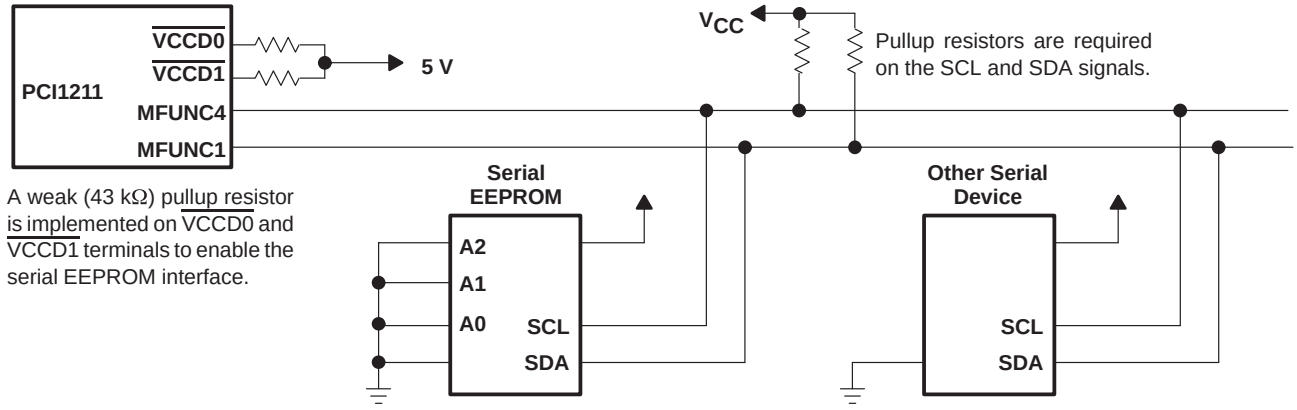



**Figure 8. Serial EEPROM Application**

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.

### serial bus interface protocol

The SCL and SDA signals are bidirectional, open-drain signals and require pullup resistors as shown in Figure 8. The PCI1211 supports up to 100 kb/s data transfer rate and is compatible with standard mode $I^2C$ using seven-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 signalled when the SDA line transitions to a low state while SCL is in the high state as illustrated in Figure 9. The end of a requested data transfer is indicated by a stop condition, which is signalled by a low-to-high transition of SDA while SCL is in the high state as shown in Figure 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 is interpreted as control signals, that is, a start or a stop condition.

**serial bus interface protocol (continued)**

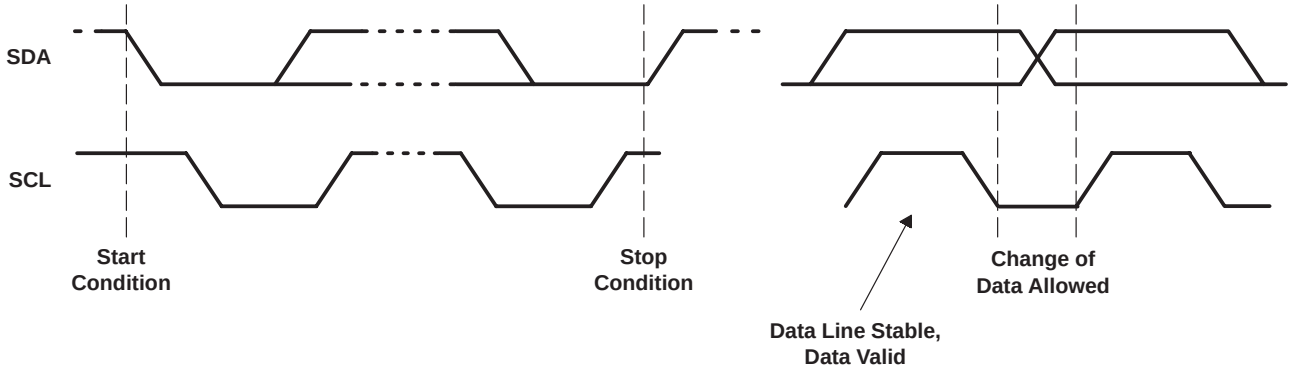



**Figure 9. Serial Bus Start/Stop Conditions and Bit Transfers**

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 10 illustrates the acknowledge protocol.

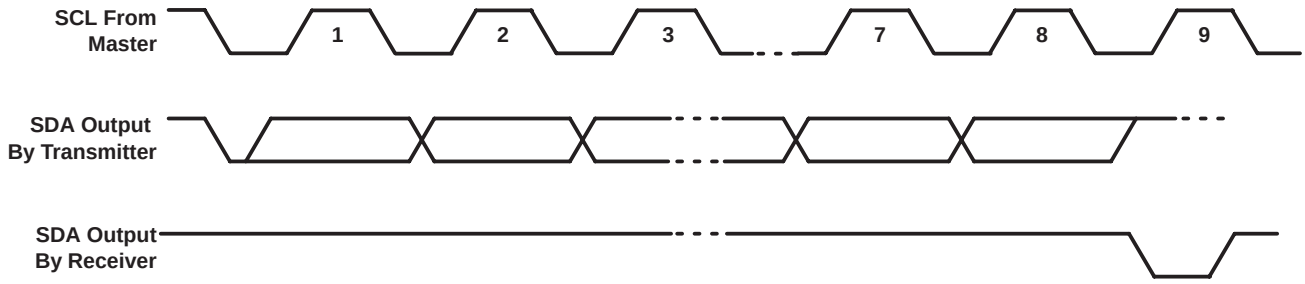



**Figure 10. Serial Bus Protocol Acknowledge**

The PCI1211 is a serial bus master; all other devices connected to the serial bus external to the PCI1211 are slave devices. As the bus master, the PCI1211 drives the SCL clock at nearly 100 kHz during bus cycles, and 3-states SCL (zero frequency) during idle states.

Typically, the PCI1211 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. Refer to *serial bus EEPROM application* on page 32 for details on how the PCI1211 automatically loads the subsystem identification and other register defaults through a serial bus EEPROM.

Figure 11 illustrates a byte write operation. The PCI1211 issues a start condition and sends the seven-bit slave device address and the command bit zero. A zero 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 there is no acknowledgment received by the PCI1211, then an appropriate status bit is set in the serial bus control and status register. The word address byte is then sent by the PCI1211 and another slave acknowledgment is expected. Then the PCI1211 delivers the data byte MSB first and expects a final acknowledgment before issuing the stop condition.

### serial bus interface protocol (continued)

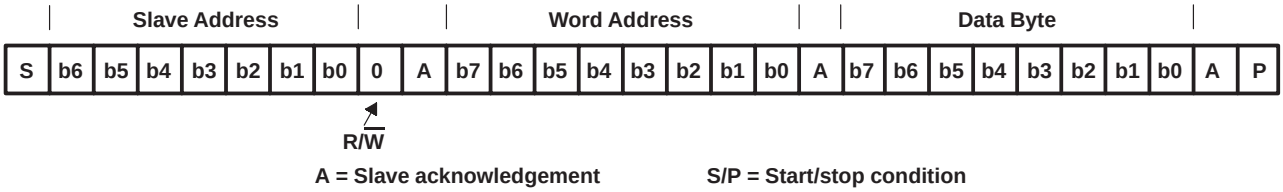



**Figure 11. Serial Bus Protocol – Byte Write**

Figure 12 illustrates a byte read operation. The read protocol is very similar to the write protocol except the R/$\overline{W}$ command bit must be set to one to indicate a read-data transfer. In addition, the PCI1211 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 PCI1211 master.

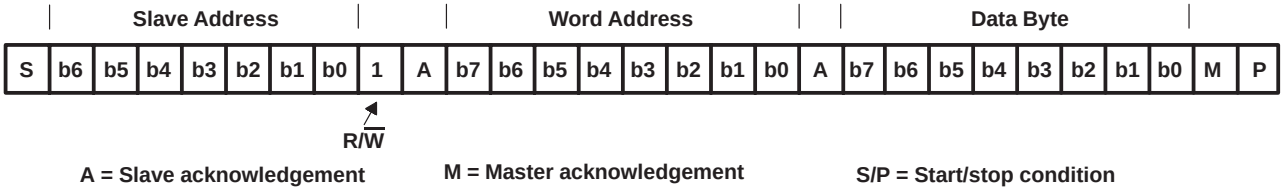



**Figure 12. Serial Bus Protocol – Byte Read**

### serial bus EEPROM application

When the PCI bus is reset, and the serial bus interface is detected, the PCI1211 attempts to read the subsystem identification and other register defaults from a serial EEPROM. The registers and corresponding bits that may be loaded with defaults through the EEPROM are provided in Table 10.

**Table 10. Registers and Bits Loadable Through Serial EEPROM**

| PCI OFFSET | OFFSET REFERENCE | REGISTER | BITS LOADED FROM EEPROM |
|---|---|---|---|
| 40h | 01h | Subsystem identification | 31–0 |
| 80h | 02h | System control register | 31–30, 27, 26, 24, 15–14, 6–3, 1 |
| 8Ch | 03h | Multifunction routing register | 27–0 |
| 90h | 04h | Retry status, Card control, device control, diagnostic | 31, 28–24, 22, 19–16, 15, 7–6 |

Figure 13 details the EEPROM data format. This format must be followed for the PCI1211 to properly load initializations from a serial EEPROM. Any undefined condition results in a terminated load and sets the ROM_ERR bit in the serial bus control and status register.

**serial bus EEPROM application (continued)**

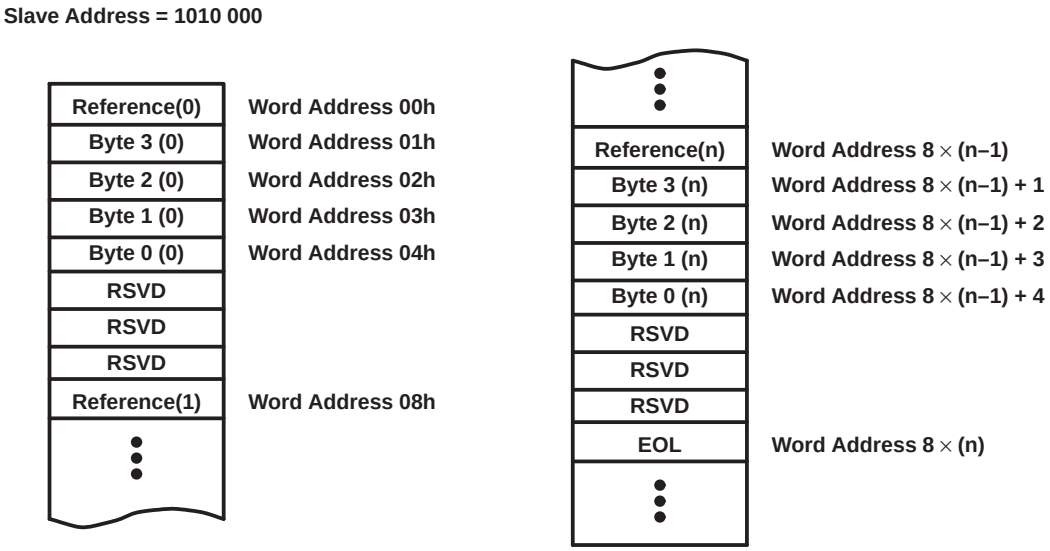



**Figure 13. EEPROM Data Format**

The byte at the EEPROM word address 00h must either contain a valid PCI offset, as listed in Table 10, or an end-of-list (EOL) indicator. The EOL indicator is a byte value of FFh, and indicates the end of the data to load from the EEPROM. Only doubleword registers are loaded from the EEPROM, and all bit fields must be considered when programming the EEPROM.

The serial EEPROM is addressed at slave address 1010000b by the PCI1211. 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 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.

When a valid offset reference is read, four bytes are read from the EEPROM, MSB first, as illustrated in Figure 14. The address auto-increments after every byte transfer according to the doubleword read protocol. The word addresses align with the data format illustrated in Figure 13. The PCI1211 continues to load data from the serial EEPROM until an end-of-list indicator is read. Three reserved bytes are stuffed to maintain eight-byte data structures.

The eight-byte data structure is important to provide correct addressing per the doubleword read format shown in Figure 14. In addition, the reference offsets must be loaded in the EEPROM in sequential order, that is 01h, 02h, 03h, 04h. If the offsets are not sequential, then the registers may be loaded incorrectly.

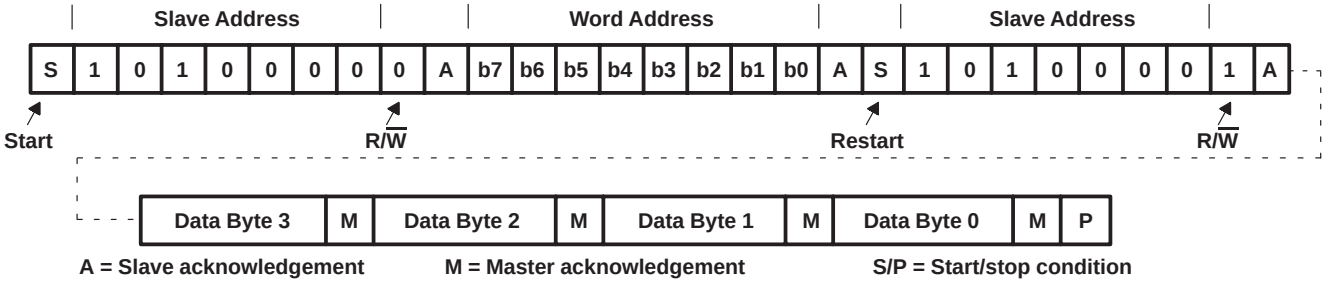



**Figure 14. EEPROM Interface Doubleword Data Collection**

### accessing serial bus devices through software

The PCI1211 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 11 illustrates the registers used to program a serial bus device through software.

**Table 11. PCI1211 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 | Writes 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. |

## 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 PCI1211. The PCI1211 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 PCI1211 is, therefore, backward compatible with existing interrupt control register definitions, and new registers have been defined where required.

The PCI1211 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 PCI1211, PC Card interrupts are classified as either card status change (CSC) or as functional interrupts.

The method by which any type of PCI1211 interrupt is communicated to the host interrupt controller varies from system to system. The PCI1211 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 signalling is provided through the seven multifunction terminals, MFUNC0–MFUNC6.

### 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 PCI1211 and may warrant notification of host card and socket services software for service. CSC events include both card insertion and removal from PC Card sockets, as well as transitions of certain PC Card signals.

Table 12 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

**PC Card functional and card status change interrupts (continued)**

**Table 12. Interrupt Mask and Flag Registers**

| CARD TYPE | EVENT | MASK | FLAG |
|---|---|---|---|
| 16-bit memory | Battery conditions (BVD1, BVD2) | ExCA offset 05h/805h bits 1 and 0 | ExCA offset 04h/804h bits 1 and 0 |
| | Wait states (READY) | ExCA offset 05h/805h bit 2 | ExCA offset 04h/804h bit 2 |
| 16-bit I/O | Change in card status ($\overline{\text{STSCHG}}$) | ExCA offset 05h/805h bit 0 | ExCA offset 04h/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/805h bit 3 | ExCA offset 04h/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 13 describes the PC Card interrupt events.

**Table 13. 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 (STSCHG) | CSC | BVD1($\overline{\text{STSCHG}}$)//CSTSCHG | The assertion of $\overline{\text{STSCHG}}$ indicates a status change on the PC Card. |
| | Interrupt request (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//CardBus PC Card. |
| | Power cycle complete | CSC | N/A | An interrupt is generated when a PC Card power-up cycle has completed. |

**PC Card functional and card status change interrupts (continued)**

The naming convention for PC Card signals describes the function for 16-bit memory and I/O cards, as well as CardBus. For example, READY(IREQ)//CINT includes READY for 16-bit memory cards, IREQ for 16-bit I/O cards, and 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 forward double slash (//).

The PC Card standard describes the power-up sequence that must be followed by the PCI1211 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 PCI1211 interrupt scheme can be used to notify the host system (see Table 13), denoted by the power cycle complete event. This interrupt source is considered a PCI1211 internal event because it does not depend on a signal change at the PC Card interface, but rather the completion of applying power to the socket.

**interrupt masks and flags**

Host software may individually mask (or disable) most of the potential interrupt sources listed in Table 13 by setting the appropriate bits in the PCI1211. By individually masking the interrupt sources listed, software can control those events that cause a PCI1211 interrupt. Host software has some control over the system interrupt the PCI1211 asserts by programming the appropriate routing registers. The PCI1211 allows host software to route PC Card CSC and PC Card functional interrupts to separate system interrupts. A discussion of interrupt routing is somewhat specific to the interrupt signaling method used, and is discussed in more detail in the following sections.

When an interrupt is signaled by the PCI1211, the interrupt service routine must determine which of the events in Table 12 caused the interrupt. Internal registers in the PCI1211 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 12 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 PCI1211 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.

There are various methods of clearing the interrupt flag bits listed in Table 12. 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 1 to the flag bit to clear, and the other is by reading the flag bit register. The selection of flag bit clearing is made by bit 2 in the global control register (ExCA offset 1Eh/81Eh), and defaults to the *flag cleared on read* method.

The CardBus-related interrupt flags can be cleared by an explicit write of 1 to the interrupt flag in the socket event register. 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.

**using parallel IRQ interrupts**

The seven multifunction terminals, MFUNC6–MFUNC0, implemented in the PCI1211 may 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, located at PCI offset 92h, to select the parallel IRQ signaling scheme. Refer to the *multifunction routing register* description on page 61 for details on configuring the multifunction terminals.

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

**using parallel IRQ interrupts (continued)**

As an example, suppose the six IRQs used by legacy PC Card applications are IRQ3, IRQ4, IRQ5, IRQ10, IRQ11, and IRQ15. The multifunction control register must be programmed to a value of 0x0FBA5432. This value routes the MFUNC0 terminal to $\overline{\text{INTA}}$ signaling, and routes the remaining terminals as illustrated in Figure 15. 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.

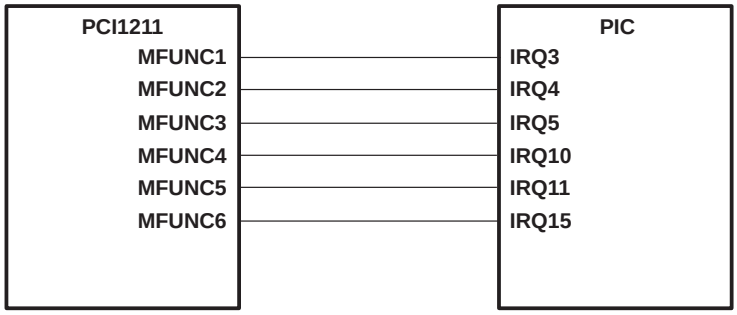



**Figure 15. Example of IRQ Implementation**

Power-on software is responsible for programming the multifunction routing register to reflect the IRQ configuration of a system implementing the PCI1211. Refer to the *multifunction routing register* description on page 61 for details on configuring the multifunction terminals.

The parallel ISA type IRQ signaling from the MFUNC6–MFUNC0 terminals is compatible with those input directly into the 8259 PIC. The parallel IRQ option is provided for system designs that require legacy ISA IRQs. There may be design constraints that demand more MFUNC6–MFUNC0 IRQ terminals than the PCI1211 makes available. A system designer may choose to implement an IRQSER deserializer companion chip, such as the Texas Instruments PCI950. To use a deserializer, the MFUNC3 terminal must be configured as IRQSER and connected to the deserializer, which outputs all 15 ISA IRQ's and four PCI interrupts as decoded from the IRQSER stream.

**using parallel PCI interrupts**

Parallel PCI interrupts are available when exclusively in parallel PCI interrupt mode, parallel ISA IRQ signaling mode, and when only IRQs are serialized with the IRQSER protocol. The socket function interrupts are routed to $\overline{\text{INTA}}$ (MFUNC0).

**using serialized IRQSER interrupts**

The serialized interrupt protocol implemented in the PCI1211 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*.

**SMI support in the PCI1211**

The PCI1211 provides a mechanism of interrupting the system when power changes have been made to the PC Card socket interface. The interrupt mechanism is designed to fit into a system maintenance interrupt (SMI) scheme. SMI interrupts are generated by the PCI1211, when enabled, after a write cycle to either the socket control register of the CardBus register set or the power control register of the ExCA register set causes a power cycle change sequence sent on the power switch interface.

The SMI control is programmed through three bits in the system control register. These bits are SMIROUTE, SMISTATUS, and SMIENB. The SMI control bits function as described in Table 14.

**SMI support in the PCI1211 (continued)**

**Table 14. SMI Control**

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

If CSC SMI interrupts are selected, then the SMI interrupt is sent as the CSC. The CSC interrupt can be either level or edge mode depending upon the CSCMODE bit in the ExCA global control register.

If IRQ2 is selected by SMIROUTE, 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 MFUNC1, MFUNC3, or MFUNC6 through the multifunction routing register.

## power management overview

In addition to the low-power CMOS technology process used for the PCI1211, various features are designed into the device to allow implementation of popular power-saving techniques. These features and techniques are discussed in this section.

### $\overline{\text{CLKRUN}}$ protocol

The PCI $\overline{\text{CLKRUN}}$ feature is the primary method of power management on the PCI interface of the PCI1211. $\overline{\text{CLKRUN}}$ signalling is provided through the MFUNC6 terminal. Since some chipsets do not implement $\overline{\text{CLKRUN}}$, this is not always available to the system designer, alternate power savings features are provided. For details on the $\overline{\text{CLKRUN}}$ protocol refer to the *PCI Mobile Design Guide*.

The PCI1211 does not permit the central resource to stop the PCI clock under any of the following conditions:

- The KEEPCLK bit in the system control register is set.
- The PC Card-16 resource manager is busy.
- The PCI1211 CardBus master state machine is busy. A cycle may be in progress on CardBus.
- The PCI1211 master is busy. There may be posted data from CardBus to PCI in the PCI1211.
- There are pending interrupts.
- The CardBus CCLK has not been stopped by the PCI1211 PCI $\overline{\text{CCLKRUN}}$ manager.

The PCI1211 restarts the PCI clock using the clockk run protocol under any of the following conditions:

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

### CardBus PC Card Power Management

The PCI1211 implements its own card power management engine that can be used to turn off the CCLK to the socket when there is no activity to the CardBus PC Card. The PCI $\overline{\text{CCLKRUN}}$ protocol is followed on the CardBus interface to control this clock management.

### 16-Bit PC Card Power Management

The COE and PWRDOWN bits in the ExCA registers are provided for 16-bit PC Card power management. The COE bit three states the card interface to save power. The power savings when using this feature are minimal. The COE bit will reset the PC Card when used, and the PWRDOWN bit will not. Furthermore, the PWRDOWN bit is an automatic COE, that is, the PWRDOWN 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 PWRDOWN modes.

### suspend mode

The $\overline{\text{SUSPEND}}$ signal is provided for backward compatibility, and gates the PCI reset ($\overline{\text{RST}}$) signal from the PCI1211. However, additional functionality has been defined for $\overline{\text{SUSPEND}}$ to provide additional power-management options.

$\overline{\text{SUSPEND}}$ provides a mechanism to gate the PCLK from the PCI1211, as well as gate $\overline{\text{RST}}$. This can potentially save power while in an idle state; however, it requires substantial design effort to implement. Some issues to consider are:

- What if a card is present in the socket?
- What if the card in the socket is powered?
- How to pass CSC (insertion/removal) events.

Even without the PCI clock to the PCI1211 core, there are asynchronous-type functions (such as $\overline{\text{RI_OUT}}$) that can pass CSC events, wake-up events, etc., back to the system. Figure 16 is a functional implementation diagram for $\overline{\text{SUSPEND}}$.

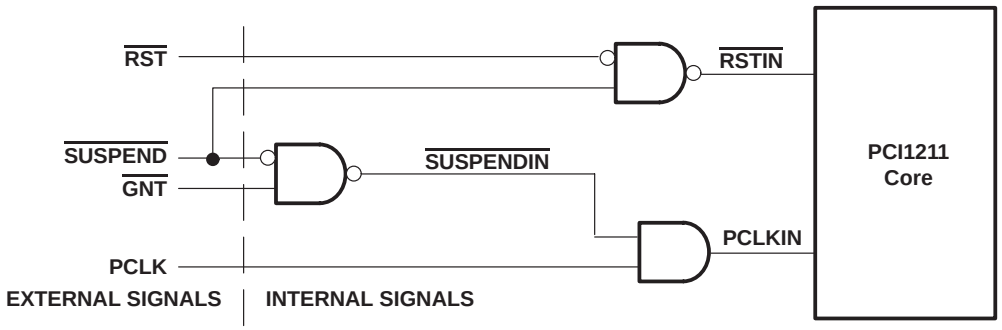



**Figure 16. $\overline{\text{SUSPEND}}$ Functional Implementation**

Figure 17 is a signal diagram of the suspend function.

**suspend mode (continued)**

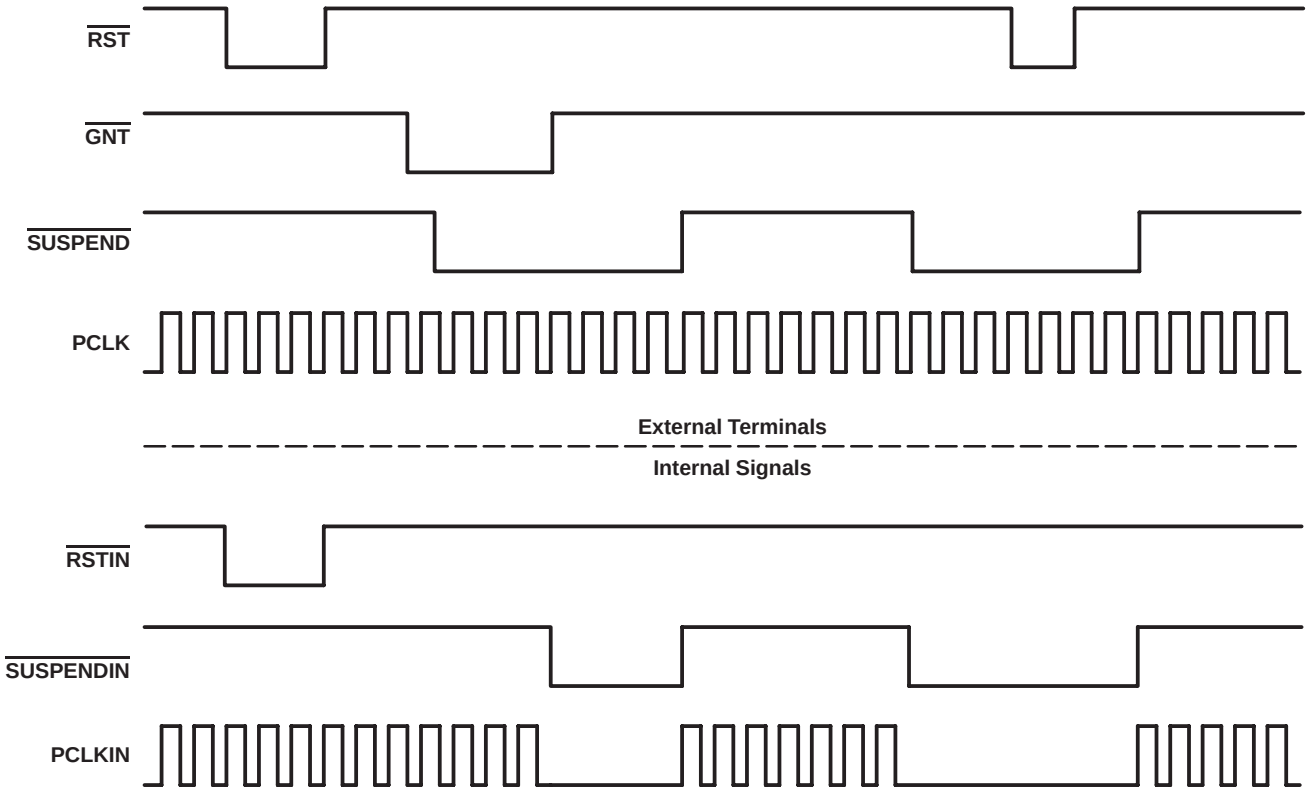



**Figure 17. Signal Diagram of Suspend Function**

**ring indicate**

The RI_OUT output is an important feature in power management and is basically used so that a system can go into a suspended mode and wake up on modem rings and other card events. RI_OUT on the PCI1211 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 CSC event occurs, such as insertion/removal of cards, battery voltage levels.

CSTSCHG from a powered CardBus card is indicated as a CSC event, not as a CBWAKE event. These two RI_OUT events are enabled separately. Figure 15 shows various enable bits for the PCI1211 RI_OUT function; however, it does not show the masking of CSC events. See *interrupt masks and flags*, on page 36, for a detailed description of CSC interrupt masks and flags.

**ring indicate (continued)**

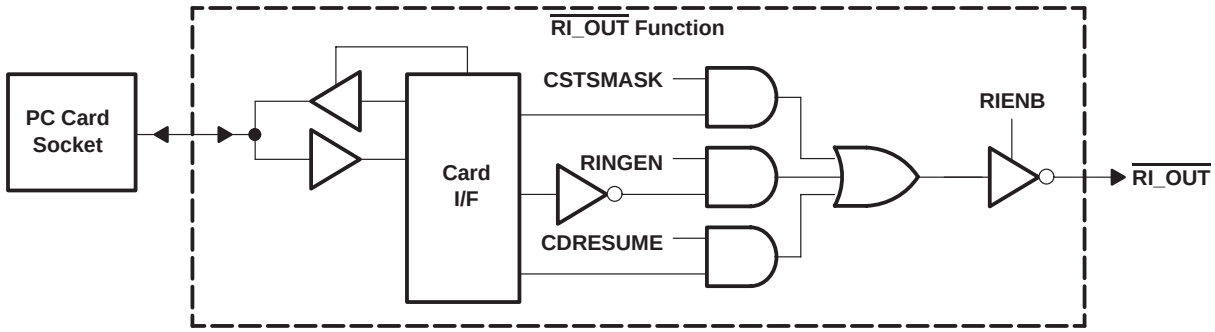



**Figure 18. $\overline{\text{RI_OUT}}$ Functional Diagram**

$\overline{\text{RI}}$ from the 16-bit PC Card interface is masked by the ExCA control bit RINGEN in the interrupt and general control register. This is programmed on a per-socket basis and is only applicable when a 16-bit card is powered in the socket.

The CBWAKE signaling to $\overline{\text{RI_OUT}}$ is enabled through the same mask as the CSC event for CSTSCHG. The mask bit, CSTSMASK, is programmed through the socket mask register in the CardBus socket registers.

**PCI power management (PCIPM)**

The PCI power-management (PCIPM) specification 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 four software-visible power-management states that result in varying levels of power savings.

The four power-management states of PCI functions are:

- D0 – Fully-on state
- D1 and D2 – Intermediate states
- 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 power 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 new capabilities is indicated by a 1 in the capabilities list (CAPLIST) bit in the status register (bit 4) and providing access to a capabilities list.

The capabilities pointer provides access to the first item in the linked list of capabilities. For the PCI1211, 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, the next item pointer should be set to 0. The registers following the next item pointer are specific to the function's capability. The PCIPM capability implements the register block outlined in Table 15.

### PCI power management (PCIPM) (continued)

**Table 15. Power-Management Registers**

| REGISTER NAME | | | | OFFSET |
|---|---|---|---|---|
| Power-management capabilities | | Next item pointer | Capability ID | 0h |
| Data | PMCSR bridge support extensions | Power-management control status (CSR) | | 4h |

The power management capabilities register is a static read-only register that provides information on the capabilities of the function related to power management. The PMCSR register 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 refer to the *PCI Bus Power Management Interface Specification, Revision 1.0*.

### ACPI Support

The *Advanced Configuration and Power Management (ACPI) Specification* provides a mechanism that allows unique pieces of hardware to be described to the ACPI driver. The PCI1211 offers a generic interface that is compliant with ACPI design rules.

Two doublewords of general purpose ACPI programming bits reside in the PCI1211 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 GPE_STS and GPE_EN registers. The status and enable bits are implemented as defined by ACPI, and illustrated in Figure 19.

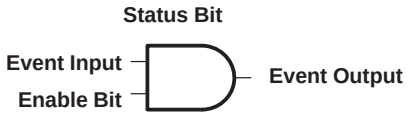



**Figure 19. Block Diagram of a Status/Enable Cell**

The status and enable bits are used to 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 on ACPI refer to the *Advanced Configuration and Power Interface Specification* at: http://www.teleport.com/~acpi/.

## PC Card controller programming model

This section describes the PCI1211 PCI configuration registers that make up the 256-byte PCI configuration header for each PCI1211 function.

## PCI configuration registers

The configuration header is compliant with the PCI specification as a CardBus bridge header, and is PC98/99 compliant as well. Table 16 shows the PCI configuration header, which includes both the predefined portion of the configuration space and the user-definable registers.

**Table 16. 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 |
| Socket DMA register 0 | | | | 94h |
| Socket DMA register 1 | | | | 98h |
| Reserved | | | | 9Ch |
| Power-management capabilities | | Next-item pointer | Capability ID | A0h |
| PM data | PMCSR 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 |

### vendor ID register

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

Register: **Vendor ID**
Type: Read-only
Offset: 00h
Default: 104Ch
Description: This register contains a value allocated by the PCI SIG (special interest group) and identifies the manufacturer of the PCI device. The vendor ID assigned to TI is 104Ch.

### device ID register

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

Register: **Device ID**
Type: Read-only
Offset: 02h
Default: AC1Eh
Description: This register contains a value assigned to the PCI1211 by TI. The device identification for the PCI1211 is AC1E.

### command register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Command | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R/W | R | R/W | R | R | R | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Command**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 04h
Default: 0000h
Description: This register provides control over the PCI1211 interface to the PCI bus. All bit functions adhere to the definitions in *PCI Local Bus Specification, Revision 2.2*. See Table 17 for the complete description of the register contents.

**Table 17. Command Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15–10 | RSVD | R | Reserved. Bits 15–10 return 0s when read. Writes have no effect. |
| 9 | FBB_EN | R | Fast back-to-back enable. The PCI1211 does not generate fast back-to-back transactions; therefore, bit 9 returns 0 when read. |
| 8 | SERR_EN | R/W | 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 bit 8 and bit 6 must be set for the PCI1211 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 PCI1211 does not support address/data stepping, and bit 7 is hardwired to 0. Writes to this bit have no effect. |
| 6 | PERR_EN | R/W | Parity error response enable. Bit 6 controls the PCI1211's 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 = PCI1211 ignores detected parity error (default)<br>1 = PCI1211 responds to detected parity errors |
| 5 | VGA_EN | R | VGA palette snoop. Bit 5 controls how PCI devices handle accesses to video graphics array (VGA) palette registers. The PCI1211 does not support VGA palette snooping; therefore, this bit is hardwired to 0. Bit 5 returns 0 when read. Writes to this bit have no effect. |
| 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 PCI1211 controller does not support memory write and invalidate commands, it uses memory write commands instead; therefore, this bit is hardwired to 0. Bit 4 returns 0 when read. Writes to this bit have no effect. |
| 3 | SPECIAL | R | Special cycles. Bit 3 controls whether or not a PCI device ignores PCI special cycles. The PCI1211 does not respond to special cycle operations; therefore, this bit is hardwired to 0. Bit 3 returns 0 when read. Writes to this bit have no effect. |
| 2 | MAST_EN | R/W | Bus master control. Bit 2 controls whether or not the PCI1211 can act as a PCI bus initiator (master). The PCI1211 can take control of the PCI bus only when this bit is set.<br>0 = Disables the PCI1211's ability to generate PCI bus accesses (default)<br>1 = Enables the PCI1211's ability to generate PCI bus accesses |
| 1 | MEM_EN | R/W | Memory space enable. Bit 1 controls whether or not the PCI1211 can claim cycles in PCI memory space.<br>0 = Disables the PCI1211's response to memory space accesses (default)<br>1 = Enables the PCI1211's response to memory space accesses |
| 0 | IO_EN | R/W | I/O space control. Bit 0 controls whether or not the PCI1211 can claim cycles in PCI I/O space.<br>0 = Disables the PCI1211 from responding to I/O space accesses (default)<br>1 = Enables the PCI1211 to respond to I/O space accesses |

### status register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Status | | | | | | | | | | | | | | | |
| Type | R/WC | R/WC | R/WC | R/WC | R/WC | R | R | R/WC | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Register: **Status**
Type: Read-only, Read/Write to Clear (see individual bit descriptions)
Offset: 06h
Default: 0210h
Description: This register provides device information to the host system. Bits in this register may be read normally. A bit in the status register is reset when a 1 is written to that bit location; a 0 written to a bit location has no effect. All bit functions adhere to the definitions in the *PCI Local Bus Specification, Revision 2.2*. PCI bus status is shown through each function. See Table 18 for the complete description of the register contents.

**Table 18. Status Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | PAR_ERR | R/WC | Detected parity error. Bit 15 is set when a parity error is detected (either address or data). |
| 14 | SYS_ERR | R/WC | Signaled system error. Bit 14 is set when $\overline{\text{SERR}}$ is enabled and the PCI1211 signals a system error to the host. |
| 13 | MABORT | R/WC | Received master abort. Bit 13 is set when a cycle initiated by the PCI1211 on the PCI bus has been terminated by a master abort. |
| 12 | TABT_REC | R/WC | Received target abort. Bit 12 is set when a cycle initiated by the PCI1211 on the PCI bus was terminated by a target abort. |
| 11 | TABT_SIG | R/WC | Signaled target abort. Bit 11 is set by the PCI1211 when it terminates a transaction on the PCI bus with a target abort. |
| 10–9 | PCI_SPEED | R | DEVSEL timing. These bits encode the timing of $\overline{\text{DEVSEL}}$ and are hardwired 01b, indicating that the PCI1211 asserts PCI_SPEED at a medium speed on nonconfiguration cycle accesses. |
| 8 | DATAPAR | R/WC | 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 PCI1211.<br>b. The PCI1211 was the bus master during the data parity error.<br>c. The parity error response bit is set in the command. |
| 7 | FBB_CAP | R | Fast back-to-back capable. The PCI1211 cannot accept fast back-to-back transactions; thus, bit 7 is hardwired to 0. |
| 6 | UDF | R | User-definable feature support. The PCI1211 does not support the user-definable features; thus, bit 6 is hardwired to 0. |
| 5 | 66MHZ | R | 66-MHz capable. The PCI1211 operates at a maximum PCLK frequency of 33 MHz; therefore, bit 5 is hardwired to 0. |
| 4 | CAPLIST | R | Capabilities list. Bit 4 returns 1 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 0s when read. |

### revision ID register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Revision ID | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Revision ID**
Type: Read-only
Offset: 08h
Default: 00h
Description: This register indicates the silicon revision of the PCI1211.

### PCI class code register

| 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 | Class code | | | | | | | | | | | | | | | | | | | | | | | |
| | Base class | | | | | | | | Sub class | | | | | | | | Programming interface | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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**
Type: Read-only
Offset: 09h
Default: 060700h
Description: This register recognizes the PCI1211 as a bridge device (06h), and CardBus bridge device (07h) with a 00h programming interface.

### cache line size register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Cache line size | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Cache line size**
Type: Read/Write
Offset: 0Ch
Default: 00h
Description: This register is programmed by host software to indicate the system cache line size.

### latency timer register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Latency timer | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Latency timer**
Type: Read/Write
Offset: 0Dh
Default: 00h
Description: This register specifies the latency timer for the PCI1211 in units of PCI clock cycles. When the PCI1211 is a PCI bus initiator and asserts $\overline{\text{FRAME}}$, the latency timer begins counting from zero. If the latency timer expires before the PCI1211 transaction has terminated, the PCI1211 terminates the transaction when its $\overline{\text{GNT}}$ is deasserted.

### header type register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Header type | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

Register: **Header type**
Type: Read-only
Offset: 0Eh
Default: 02h
Description: This register returns 02h when read, indicating that the PCI1211 configuration spaces adhere to the CardBus bridge PCI header. The CardBus bridge PCI header ranges from PCI register 0 to 7Fh, leaving 80h–FFh is user-definable extension registers.

### BIST register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | BIST | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **BIST**
Type: Read-only
Offset: 0Fh
Default: 00h
Description: Because the PCI1211 does not support a built-in self-test (BIST), this register returns the value of 00h when read.

**CardBus socket registers/ExCA registers base-address register**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | CardBus socket/ExCA registers base address | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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 |
| Name | CardBus socket/ExCA registers base address | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R | R | R | R | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **CardBus socket/ExCA registers base address**
Type: Read-only, Read/Write
Offset: 10h
Default: 0000 0000h
Description: This 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 4K-byte boundary. Bits 11–0 are read-only, returning 0s when read. When software writes all 1s to this register, the value readback is FFFF F000h, indicating that at least 4K bytes of memory address space are required. The CardBus registers start at offset 000h, and the memory-mapped ExCA registers begin at offset 800h.

**capability pointer register**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Capability pointer | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

Register: **Capability pointer**
Type: Read-only
Offset: 14h
Default: A0h
Description: This 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. The socket has its own capability pointer register. This register returns A0h when read.

### secondary status register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Secondary status | | | | | | | | | | | | | | | |
| Type | R/WC | R/WC | R/WC | R/WC | R/WC | R | R | R/WC | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Secondary status**
Type: Read-only, Read/Write to Clear (see individual bit descriptions)
Offset: 16h
Default: 0200h
Description: This 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 PCI status register (offset 06h), and status bits are cleared by writing a 1. See Table 19 for the complete description of the register contents.

**Table 19. Secondary Status Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | CBPARITY | R/WC | Detected parity error. Bit 15 is set when a CardBus parity error is detected (either address or data). |
| 14 | CBSERR | R/WC | Signaled system error. Bit 14 is set when $\overline{\text{CSERR}}$ is signaled by a CardBus card. The PCI1211 does not assert $\overline{\text{CSERR}}$. |
| 13 | CBMABORT | R/WC | Received master abort. Bit 13 is set when a cycle initiated by the PCI1211 on the CardBus bus has been terminated by a master abort. |
| 12 | REC_CBTA | R/WC | Received target abort. Bit 12 is set when a cycle initiated by the PCI1211 on the CardBus bus is terminated by a target abort. |
| 11 | SIG_CBTA | R/WC | Signaled target abort. Bit 11 is set by the PCI1211 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 PCI1211 asserts CB_SPEED at a medium speed. |
| 8 | CB_DPAR | R/WC | 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 PCI1211 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 PCI1211 cannot accept fast back-to-back transactions; thus, bit 7 is hardwired to 0. |
| 6 | CB_UDF | R | User-definable feature support. The PCI1211 does not support the user-definable features; thus, bit 6 is hardwired to 0. |
| 5 | CB66MHZ | R | 66-MHz capable. The PCI1211 CardBus interface operates at a maximum CCLK frequency of 33 MHz; therefore, bit 5 is hardwired to 0. |
| 4–0 | RSVD | R | Reserved. Bits 4–0 return 0s when read. |

### PCI bus number register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | PCI bus number | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **PCI bus number**
Type: Read/Write
Offset: 18h
Default: 00h
Description: This register is programmed by the host system to indicate the bus number of the PCI bus to which the PCI1211 is connected. The PCI1211 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.

### CardBus bus number register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CardBus bus number | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **CardBus bus number**
Type: Read/Write
Offset: 19h
Default: 00h
Description: This register is programmed by the host system to indicate the bus number of the CardBus bus to which the PCI1211 is connected. The PCI1211 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.

### subordinate bus number register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Subordinate bus number | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Subordinate bus number**
Type: Read/Write
Offset: 1Ah
Default: 00h
Description: This register is programmed by the host system to indicate the highest-numbered bus below the CardBus bus. The PCI1211 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.

### CardBus latency timer register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CardBus latency timer | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **CardBus latency timer**
Type: Read/Write
Offset: 1Bh
Default: 00h
Description: This register is programmed by the host system to specify the latency timer for the PCI1211 CardBus interface in units of CCLK cycles. When the PCI1211 is a CardBus initiator and asserts $\overline{\text{CFRAME}}$, the CardBus latency timer begins counting. If the latency timer expires before the PCI1211 transaction has terminated, then the PCI1211 terminates the transaction at the end of the next data phase. A recommended minimum value for this register is 20h, which allows most transactions to be completed.

### memory base registers 0, 1

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Memory base registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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 |
| Name | Memory base registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R | R | R | R | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Memory base registers 0, 1**
Type: Read-only, Read/Write
Offset: 1Ch, 24h
Default: 0000 0000h
Description: These registers indicate the lower address of a PCI memory address range and are used by the PCI1211 to determine when to forward a memory transaction to the CardBus bus, and likewise, 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 4K-byte boundaries. Bits 11–0 always return 0s when read. Writes 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 PCI1211 to claim any memory transactions through CardBus memory windows (i.e., these windows are not enabled by default to pass the first 4K bytes of memory to CardBus).

### memory limit registers 0, 1

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Memory limit registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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 |
| Name | Memory limit registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R | R | R | R | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Memory limit registers 0, 1**
Type: Read-only, Read/Write
Offset: 20h, 28h
Default: 0000 0000h
Description: These registers indicate the upper address of a PCI memory address range and are used by the PCI1211 to determine when to forward a memory transaction to the CardBus bus, and likewise, 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 4K-byte boundaries. Bits 11–0 always return 0s when read. Writes 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 PCI1211 to claim any memory transactions through CardBus memory windows (i.e., these windows are not enabled by default to pass the first 4K bytes of memory to CardBus).

### I/O base registers 0, 1

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | I/O base registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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 |
| Name | I/O base registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **I/O base registers 0, 1**
Type: Read-only, Read/Write
Offset: 2Ch, 34h
Default: 0000 0000h
Description: These registers indicate the lower address of a PCI I/O address range and are used by the PCI1211 to determine when to forward an I/O transaction to the CardBus bus, and likewise, 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 64K-byte page, and the upper 16 bits (31–16) are a page register which locates this 64K-byte page in 32-bit PCI I/O address space. Bits 31–2 are read/write. Bits 1–0 always return 0s when read, forcing I/O windows to be aligned on a natural doubleword boundary.

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

### I/O limit registers 0, 1

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | I/O limit registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | I/O limit registers 0, 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **I/O limit registers 0, 1**
Type: Read-only, Read/Write
Offset: 30h, 38h
Default: 0000 0000h
Description: These registers indicate the upper address of a PCI I/O address range and are used by the PCI1211 to determine when to forward an I/O transaction to the CardBus bus, and likewise, 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 64K-byte page, and the upper 16 bits are a page register which locates this 64K-byte 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 64K-byte page (indicated by bits 31–16 of the appropriate I/O base) on doubleword boundaries.

Bits 31–16 always return 0s when read. The page is set in the I/O base register. Bits 1–0 always return 0s when read, forcing I/O windows to be aligned on a natural doubleword boundary. Writes to read-only bits have no effect. The PCI1211 assumes that the lower 2 bits of the limit address are 1s.

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

### interrupt line register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Interrupt line | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Register: **Interrupt line**
Type: Read/Write
Offset: 3Ch
Default: FFh
Description: This register is used to communicate interrupt line routing information.

### interrupt pin register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Interrupt pin | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Register: **Interrupt pin**
Type: Read-only
Offset: 3Dh
Default: 01h
Description: The value read from the interrupt pin register is function dependent and reflects the interrupt signalling mode selected through the device control register (92h). The PCI1211 defaults to serialized PCI and ISA interrupt mode.

### bridge control register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Bridge control | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R/W | R/W | R/W | R/W | R/W | R/W | R | R/W | R/W | R/W | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Bridge control**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 3Eh
Default: 0340h
Description: This register provides control over various PCI1211 bridging functions. See Table 20 for a complete description of the register contents.

**Table 20. Bridge Control Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15–11 | RSVD | R | Reserved. Bits 15–11 return 0s when read. |
| 10 | POSTEN | R/W | Write posting enable. Enables write posting to and from the CardBus sockets. Write posting enables posting of write data on burst cycles. Operating with write posting disabled inhibits performance on burst cycles. Note that bursted write data can be posted, but various write transactions may not. |
| 9 | PREFETCH1 | R/W | Memory window 1 type. Bit 9 specifies whether or not memory window 1 is prefetchable. This bit is socket dependent. Bit 9 is encoded as:<br>0 = Memory window 1 is nonprefetchable.<br>1 = Memory window 1 is prefetchable (default). |
| 8 | PREFETCH0 | R/W | 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 | R/W | PCI interrupt – IREQ routing enable. Bit 7 is used to select 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 | R/W | 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 RST assertion to CardBus.<br>0 = $\overline{\text{CRST}}$ deasserted<br>1 = $\overline{\text{CRST}}$ asserted (default) |
| 5 | MABTMODE | R/W | Master abort mode. Bit 5 controls how the PCI1211 responds to a master abort when the PCI1211 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 0 when read. |
| 3 | VGAEN | R/W | VGA enable. Bit 3 affects how the PCI1211 responds to VGA addresses. When this bit is set, accesses to VGA addresses are forwarded. |
| 2 | ISAEN | R/W | ISA mode enable. Bit 2 affects how the PCI1211 passes I/O cycles within the 64K-byte ISA range. This bit is not common between sockets. When this bit is set, the PCI1211 does not forward the last 768 bytes of each 1K I/O range to CardBus. |
| 1 | CSERREN | R/W | $\overline{\text{CSERR}}$ enable. Bit 1 controls the response of the PCI1211 to $\overline{\text{CSERR}}$ signals on the CardBus bus. This bit is common between the two sockets.<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 | R | CardBus parity error response enable. Bit 0 controls the response of the PCI1211 to CardBus parity errors. This bit is common between the two sockets.<br>0 = CardBus parity errors are ignored.<br>1 = CardBus parity errors are reported using $\overline{\text{CPERR}}$. |

### subsystem vendor ID register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Subsystem vendor ID | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Subsystem vendor ID**
Type: Read-only (read/write when bit 5 in the system control register is 0)
Offset: 40h
Default: 0000h
Description: This 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. When bit 5 is 0, this register is read/write; when bit 5 is 1, this register is read-only. The default mode is read-only.

### subsystem ID register

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

Register: **Subsystem ID**
Type: Read-only (read/write when bit 5 in the system control register is 0)
Offset: 42h
Default: 0000h
Description: This 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. When bit 5 is 0, this register is read/write; when bit 5 is 1, this register is read-only. The default mode is read-only.

### PC Card 16-bit I/F legacy-mode base address register

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | PC Card 16-bit I/F legacy-mode base address | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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 |
| Name | PC Card 16-bit I/F legacy-mode base address | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| 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**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 44h
Default: 0000 0001h
Description: The PCI1211 supports the index/data scheme of accessing the ExCA registers, which is 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 1 when read. Refer to *ExCA compatibility registers* on page 80 for register offsets.

### system control register

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | System control | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R | R | R | R/W | R/W | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 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 |
| Name | System control | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R | R | R | R | R | R | R | R/W | R/W | R/W | R/W | R | R/W | R/W |
| Default | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Register: **System control**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 80h
Default: 0044 9060h
Description: System-level initializations are performed through programming this doubleword register. See Table 21 for a complete description of the register contents.

**Table 21. System Control Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–30 | SER_STEP | R/W | Serialized PCI interrupt routing step. Bits 31–30 are used to configure the serialized PCI interrupt stream signaling, and accomplish an even distribution of interrupts signaled on the four PCI interrupt slots. Bits 31–30 are encoded as follows:<br>00 = $\overline{\text{INTA}}$ is signaled in the $\overline{\text{INTA}}$ IRQSER slot.<br>01 = $\overline{\text{INTA}}$ is signaled in the $\overline{\text{INTB}}$ IRQSER slot.<br>10 = $\overline{\text{INTA}}$ is signaled in the $\overline{\text{INTC}}$ IRQSER slot.<br>11 = $\overline{\text{INTA}}$ is signaled in the $\overline{\text{INTD}}$ IRQSER slot. |
| 29–27 | RSVD | R | Reserved. These bits return 0s when read. |
| 26 | SMIROUTE | R/W | 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 | R/W | SMI interrupt status. This socket-dependent bit is set when a write occurs to set the socket power, and the SMIENB bit is set. Writing a 1 to bit 25 clears the status.<br>0= SMI interrupt signaled (default)<br>1 = SMI interrupt not signaled |
| 24 | SMIENB | R/W | SMI interrupt mode enable. When bit 24 is set, the SMI interrupt signaling is enabled and generates an interrupt when a write to the socket power control occurs. This bit defaults to 0 (disabled). |
| 23 | RSVD | R | Reserved. This bit returns 0 when read. |
| 22 | CBRSVD | R/W | CardBus reserved terminals signaling. When bit 22 is set, the RSVD CardBus terminals are driven low when a CardBus card is inserted. When this bit is low (as default), these signals are 3-stated.<br>0 = 3-state CardBus RSVD<br>1 = Drive Cardbus RSVD low (default) |
| 21 | VCCPROT | R/W | $V_{CC}$ 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 | R/W | Reduced Zoom Video Enable.When this bit is enabled, A25–A22 of the card interface for PC Card 16 cards is placed in the high impedance state. This bit should not be set for normal ZV operation. This bit is encoded as:<br>0 = Reduced zoom video disabled (default)<br>1 = Reduced zoom video enabled |
| 19 | CDREQEN | R/W | PC/PCI DMA card enable. When bit 19 is set, the PCI1211 allows 16-bit PC Cards to request PC/PCI DMA using the $\overline{\text{DREQ}}$ signaling. $\overline{\text{DREQ}}$ is selected through the socket DMA register 0.<br>0 = Ignore $\overline{\text{DREQ}}$ signaling from PC Cards (default)<br>1 = Signal DMA request on $\overline{\text{DREQ}}$ |
| 18–16 | CDMACHAN | R/W | PC/PCI DMA channel assignment. Bits 18–16 are encoded as:<br>0–3 = 8-bit DMA channels<br>4 = PCI master; not used (default).<br>5–7 = 16-bit DMA channels |
| 15 | MRBURSTDN | R/W | 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 | R/W | Memory read burst enable upstream. When bit 14 is set, the PCI1211 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. This bit is socket dependent.<br>0 = No socket activity (default)<br>1 = Socket activity |
| 12 | RSVD | R | Reserved. Bit 12 returns 1 when read. This is the power rail bit. |

**Table 21. System Control Register (Continued)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 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. |
| 10 | DELAYUP | R | Power-up delay in progress status. When set, bit 9 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 10 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 | RSVD | R | Reserved. Bit 7 returns 0 when read. |
| 6 | PWRSAVINGS | R/W | Power savings mode enable. When this bit is set, if a CB card is inserted, idle, and without a CB clock, the applicable CB state machine will not be clocked. |
| 5 | SUBSYSRW | R/W | Subsystem ID (SSID), subsystem vendor ID (SSVID), ExCA ID, and revision register read/write enable.<br>0 = SSID, SSVID, ExCA ID, and revision register are read/write.<br>1 = SSID, SSVID, ExCA ID, and revision register are read-only (default). |
| 4 | CB_DPAR | R/W | 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 | CDMA_EN | R/W | PC/PCI DMA enable. Bit 3 enables PC/PCI DMA when set if MFUNC routing is configured for centralized DMA.<br>0 = Centralized DMA disabled (default)<br>1 = Centralized DMA enabled |
| 2 | RSVD | R | Reserved. Bit 2 returns 0 when read. |
| 1 | KEEPCLK | R/W | 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 | R/W | $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ 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 are enabled at the same time, $\overline{\text{RI_OUT}}$ has precedence over $\overline{\text{PME}}$.<br>1 = Only $\overline{\text{PME}}$ is routed to the $\overline{\text{RI_OUT}}$/$\overline{\text{PME}}$ terminal. |

**multifunction routing register**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Multifunction routing | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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 |
| Name | Multifunction routing | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Multifunction routing**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 8Ch
Default: 0000 0000h
Description: This 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. Pullup resistors are required for terminals configured as outputs. This register is intended to be programmed once at power-on initialization. The default value for this register may also be loaded through a serial bus EEPROM. See Table 22 for a complete description of the register contents.

**Table 22. Multifunction Routing Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–28 | RSVD | R | Reserved. These bits return 0s when read. |
| 27–24 | MFUNC6 | R/W | Multifunction terminal 6 configuration. These bits control the internal signal mapped to the MFUNC6 terminal as follows:<br>0000 = RSVD, Reserved high impedance input (default)<br>0001 = $\overline{\text{CLKRUN}}$, PCI clock control signal<br>0010 = IRQ2, Parallel ISA type<br>0011 = IRQ3, Parallel ISA type<br>0100 = IRQ4, Parallel ISA type<br>0101 = IRQ5, Parallel ISA type<br>0110 = IRQ6, Parallel ISA type<br>0111 = IRQ7, Parallel ISA type<br>1000 = IRQ8, Parallel ISA type<br>1001 = IRQ9, Parallel ISA type<br>1010 = IRQ10, Parallel ISA type<br>1011 = IRQ11, Parallel ISA type<br>1100 = IRQ12, Parallel ISA type<br>1101 = IRQ13, Parallel ISA type<br>1110 = IRQ14, Parallel ISA type<br>1111 = IRQ15, Parallel ISA type |

**Table 22. Multifunction Routing Register (Continued)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 23–20 | MFUNC5 | R/W | Multifunction terminal 5 configuration. These bits control the internal signal mapped to the MFUNC5 terminal as follows:<br>0000 = GPI4, General-purpose input (default)<br>0001 = GPO4, General-purpose output<br>0010 = PCGNT, PC/PCI (centralized) DMA grant<br>0011 = IRQ3, Parallel ISA type<br>0100 = IRQ4, Parallel ISA type<br>0101 = IRQ5, Parallel ISA type<br>0110 = ZVSTAT, Zoom video status output<br>0111 = ZVSEL0, Zoom video select output<br>1000 = CAUDPWM, PWM output of CAUDIO CardBus terminal<br>1001 = IRQ9, Parallel ISA type<br>1010 = IRQ10, Parallel ISA type<br>1011 = IRQ11, Parallel ISA type<br>1100 = LED_SKT, Socket activity LED<br>1101 = LED_SKT, Socket activity LED<br>1110 = GPE, General-Purpose event signal<br>1111 = IRQ15, Parallel ISA type |
| 19–16 | MFUNC4 | R/W | Multifunction terminal 4 configuration. These bits control the internal signal mapped to the MFUNC4 terminal as follows:<br><br>NOTE: When the serial bus mode is implemented by pulling up the VPPD0 and VPPD1 terminals, the MFUNC4 terminal provides the SCL signaling.<br><br>0000 = GPI3, General-purpose input (default)<br>0001 = GPO3, General-purpose output<br>0010 = LOCK, PCI atomic transfer support mechanism<br>0011 = IRQ3, Parallel ISA type<br>0100 = IRQ4, Parallel ISA type<br>0101 = IRQ5, Parallel ISA type<br>0110 = ZVSTAT, Zoom video status output<br>0111 = ZVSEL0, Zoom video select output<br>1000 = CAUDPWM, PWM output of CAUDIO CardBus terminal<br>1001 = IRQ9, Parallel ISA type<br>1010 = IRQ10, Parallel ISA type<br>1011 = IRQ11, Parallel ISA type<br>1100 = RI_OUT, Ring-indicate output<br>1101 = LED_SKT, Socket activity LED<br>1110 = GPE, General-purpose event signal<br>1111 = IRQ15, Parallel ISA type |
| 15–12 | MFUNC3 | R/W | Multifunction terminal 3 configuration. These bits control the internal signal mapped to the MFUNC3 terminal as follows:<br>0000 = RSVD, Reserved high impedance input (default)<br>0001 = IRQSER, Serial interrupt stream, IRQ and optional PCI<br>0010 = IRQ2, Parallel ISA type<br>0011 = IRQ3, Parallel ISA type<br>0100 = IRQ4, Parallel ISA type<br>0101 = IRQ5, Parallel ISA type<br>0110 = IRQ6, Parallel ISA type<br>0111 = IRQ7, Parallel ISA type<br>1000 = IRQ8, Parallel ISA type<br>1001 = IRQ9, Parallel ISA type<br>1010 = IRQ10, Parallel ISA type<br>1011 = IRQ11, Parallel ISA type<br>1100 = IRQ12, Parallel ISA type<br>1101 = IRQ13, Parallel ISA type<br>1110 = IRQ14, Parallel ISA type<br>1111 = IRQ15, Parallel ISA type |

**Table 22. Multifunction Routing Register (Continued)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 11–8 | MFUNC2 | R/W | Multifunction terminal 2 configuration. These bits control the internal signal mapped to the MFUNC2 terminal as follows:<br>0000 = GPI2, General-purpose input (default)<br>0001 = GPO2, General-purpose output<br>0010 = PCREQ, PC/PCI (centralized) DMA request<br>0011 = IRQ3, Parallel ISA type<br>0100 = IRQ4, Parallel ISA type<br>0101 = IRQ5, Parallel ISA type<br>0110 = ZVSTAT, Zoom video status output<br>0111 = ZVSEL0, Zoom video select output<br>1000 = CAUDPWM, PWM output of CAUDIO CardBus terminal<br>1001 = IRQ9, Parallel ISA type<br>1010 = IRQ10, Parallel ISA type<br>1011 = IRQ11, Parallel ISA type<br>1100 = RI_OUT, Ring-indicate output<br>1101 = IRQ13, Parallel ISA type<br>1110 = GPE, General-purpose event signal<br>1111 = IRQ7, Parallel ISA type |
| 7–4 | MFUNC1 | R/W | Multifunction terminal 1 configuration. These bits control the internal signal mapped to the MFUNC1 terminal as follows:<br><br>NOTE: When the serial bus mode is implemented by pulling up the VPPD0 and VPPD1 terminals, the MFUNC1 terminal provides the SDA signaling.<br><br>0000 = GPI1, General-purpose input (default)<br>0001 = GPO1, General-purpose output<br>0010 = IRQ2, Parallel ISA type<br>0011 = IRQ3, Parallel ISA type<br>0100 = IRQ4, Parallel ISA type<br>0101 = IRQ5, Parallel ISA type<br>0110 = ZVSTAT, Zoom video status output<br>0111 = ZVSEL0, Zoom video select output<br>1000 = CAUDPWM, PWM output of CAUDIO CardBus terminal<br>1001 = IRQ9, Parallel ISA type<br>1010 = IRQ10, Parallel ISA type<br>1011 = IRQ11, Parallel ISA type<br>1100 = LED_SKT, Socket activity LED<br>1101 = IRQ13, Parallel ISA type<br>1110 = GPE, General-purpose event signal<br>1111 = IRQ15, Parallel ISA type |
| 3–0 | MFUNC0 | R/W | Multifunction terminal 0 configuration. These bits control the internal signal mapped to the MFUNC0 terminal as follows:<br>0000 = GPI0, General-purpose input (default)<br>0001 = GPO0, General-purpose output<br>0010 = INTA, PCI interrupt signal, INTA<br>0011 = IRQ3, Parallel ISA type<br>0100 = IRQ4, Parallel ISA type<br>0101 = IRQ5, Parallel ISA type<br>0110 = ZVSTAT, Zoom video status output<br>0111 = ZVSEL0, Zoom video select output<br>1000 = CAUDPWM, PWM output of CAUDIO CardBus terminal<br>1001 = IRQ9, Parallel ISA type<br>1010 = IRQ10, Parallel ISA type<br>1011 = IRQ11, Parallel ISA type<br>1100 = LED_SKT, Socket activity LED<br>1101 = IRQ13, Parallel ISA type<br>1110 = GPE, General-purpose event signal<br>1111 = IRQ15, Parallel ISA type |

### retry status register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Retry status | | | | | | | |
| Type | R/W | R/W | R | R | R/WC | R | R/WC | R |
| Default | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Retry status**
Type: Read-only, Read/Write, Read/Write to Clear (see individual bit descriptions)
Offset: 90h
Default: C0h
Description: This register enables the retry timeout counters and displays the retry expiration status. The flags are set when the PCI1211 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 1 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 23 for a complete description of the register contents.

**Table 23. Retry Status Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | PCIRETRY | R/W | PCI retry timeout counter enable. Bit 7 is encoded:<br>0 = PCI retry counter disabled<br>1 = PCI retry counter enabled (default) |
| 6 | CBRETRY | R/W | CardBus retry timeout counter enable. Bit 6 is encoded:<br>0 = CardBus retry counter disabled<br>1 = CardBus retry counter enabled (default) |
| 5–4 | RSVD | R | Reserved. These bits return 0s when read. |
| 3 | TEXP_CB | R/WC | CardBus target retry expired. Write a 1 to clear bit 3.<br>0 = Inactive (default)<br>1 = Retry has expired. |
| 2 | RSVD | R | Reserved. Bit 2 returns 0 when read. |
| 1 | TEXP_PCI | R/WC | PCI target retry expired. Write a 1 to clear bit 1.<br>0 = Inactive (default)<br>1 = Retry has expired. |
| 0 | RSVD | R | Reserved. Bit 0 returns 0 when read. |

### card control register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Card control | | | | | | | |
| Type | R/W | R/W | R/W | R | R | R/W | R/W | R/WC |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Card control**
Type: Read-only, Read/Write, Read/Write to Clear (see individual bit descriptions)
Offset: 91h
Default: 00h
Description: This register is provided for PCI1130 compatibility. $\overline{\text{RI_OUT}}$ is enabled through this register. See Table 24 for a complete description of the register contents.

**Table 24. Card Control Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | RIENB | R/W | 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 $\overline{\text{RI_OUT/PME}}$ terminal when RIMUX is set to 0, or for routing to MFUNC2/4. |
| 6 | ZVENABLE | R/W | Compatibility ZV mode enable. When set, the PC Card socket interface ZV terminals enter a high-impedance state. |
| 5 | No function | R/W | These bits have no assigned function. |
| 4–3 | RSVD | R | Reserved. Bits 4–3 default to 0. |
| 2 | AUD2MUX | R/W | CardBus Audio-to-CAUDPWM. When set, the CAUDIO signal (PWM) is routed to the CAUDPWM signal which can be routed to a multifunction terminal. |
| 1 | SPKROUTEN | R/W | Speaker out enable. This bit is the enable for routing PC Card SPKR through to the SPKROUT terminal. The SPKROUT terminal drives valid data only when the socket SPKROUTEN bit is set.<br>0 = $\overline{\text{SPKR}}$ to SPKROUT not enabled (default)<br>1 = $\overline{\text{SPKR}}$ to SPKROUT enabled |
| 0 | IFG | R/WC | 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 1 to clear this bit.<br>0 = No PC Card functional interrupt detected (default).<br>1 = PC Card functional interrupt detected. |

### device control register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Device control | | | | | | | |
| Type | R | R/W | R/W | R | R/W | R/W | R/W | R/W |
| Default | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |

Register: **Device control**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 92h
Default: 66h
Description: This register is provided for PCI1130 compatibility The mode select and socket-capable force bits are programmed through this register. See Table 25 for a complete description of the register contents.

**Table 25. Device Control Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | RSVD | R | Reserved. Bit 7 returns 0 when read. |
| 6 | 3VCAPABLE | R/W | 3-V socket capable force<br>0 = Not 3-V capable<br>1 = 3-V capable (default) |
| 5 | IO16R2 | R/W | Diagnostic bit. |
| 4 | RSVD | R | Reserved. Bit 4 returns 0 when read. Writes have no effect. |
| 3 | TEST | R/W | TI test. Only a 0 should be written to bit 3. This bit can be set to shorten the interrogation counter. |
| 2–1 | INTMODE | R/W | Interrupt mode. Bits 2–1 select the interrupt signaling mode. The interrupt 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 | R/W | Reserved. This bit is reserved for test purposes. Only 0 should be written to this bit. |

### diagnostic register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Diagnostic | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |

Register: **Diagnostic**
Type: Read/Write
Offset: 93h
Default: 61h
Description: This register is provided for internal TI test purposes. It is a read/write register, but should not be accessed during normal operation. See Table 26 for a complete description of the register contents.

**Table 26. Diagnostic Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | TRUE_VAL | R/W | True value. This bit defaults to 0 when read. 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 to the PCI vendor ID and PCI device ID registers |
| 6 | RSVD | R/W | Reserved. This bit has no function. |
| 5 | CSC | R/W | CSC Interrupt Routing Control.<br>0 = CSC interrupts routed to PCI if ExCA 803 bit 4 = 1<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 | R/W | Diagnostic RETRY_DIS. Delayed transaction disable. |
| 3 | DIAG3 | R/W | Diagnostic RETRY_EXT. Extends the latency from 16 to 64. |
| 2 | DIAG2 | R/W | Diagnostic DISCARD_TIM_SEL_CB. Set = $2^{10}$, reset = $2^{15}$. |
| 1 | DIAG1 | R/W | Diagnostic DISCARD_TIM_SEL_PCI. Set = $2^{10}$, reset = $2^{15}$. |
| 0 | ASYNCINT | R/W | Global asynchronous interrupt enable. When set to a 1, bit 0 enables the asynchronous generation of CSC interrupts. |

### socket DMA register 0

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket DMA register 0 | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | Socket DMA register 0 | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket DMA register 0**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 94h
Default: 0000 0000h
Description: This register provides control over the PC Card DMA request ($\overline{\text{DREQ}}$) signaling. See Table 27 for a complete description of the register contents.

**Table 27. Socket DMA Register 0**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–2 | RSVD | R | Reserved. Bits 31–2 return 0s when read. |
| 1–0 | DREQPIN | R/W | DMA request ($\overline{\text{DREQ}}$). Bits 1–0 indicate which pin on the 16-bit PC Card interface will be used as $\overline{\text{DREQ}}$ during DMA transfers. This field is encoded as:<br>00 = Socket not configured for DMA (default).<br>01 = $\overline{\text{DREQ}}$ uses $\overline{\text{SPKR}}$.<br>10 = $\overline{\text{DREQ}}$ uses $\overline{\text{IOIS16}}$.<br>11 = $\overline{\text{DREQ}}$ uses $\overline{\text{INPACK}}$. |

**socket DMA register 1**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket DMA register 1 | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | Socket DMA register 1 | | | | | | | | | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket DMA register 1**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: 98h
Default: 0000 0000h
Description: This register provides control over the distributed DMA (DDMA) registers and the PCI portion of DMA transfers. The DMA base address locates the DDMA registers in a 16-byte region within the first 64K bytes of PCI I/O address space. See Table 28 for a complete description of the register contents.

**NOTE:**
32-bit transfers are not supported; the maximum transfer possible for 16-bit PC Cards is 16 bits.

**Table 28. Socket DMA Register 1**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–16 | RSVD | R | Reserved. Bits 31–16 return 0s when read. |
| 15–4 | DMABASE | R/W | DMA base address. Locates the socket's DMA registers in PCI I/O space. This field represents a 16-bit PCI I/O address. The upper 16 bits of the address are hardwired to 0, forcing this window to within the lower 64K bytes of I/O address space. The lower 4 bits are hardwired to 0 and are included in the address decode. Thus, the window is aligned to a natural 16-byte boundary. |
| 3 | EXTMODE | R | Extended addressing. This feature is not supported by the PCI1211 and always returns a 0. |
| 2–1 | XFERSIZE | R/W | Transfer size. Bits 2–1 specify the width of the DMA transfer on the PC Card interface and are encoded as:<br>00 = Transfers are 8 bits (default).<br>01 = Transfers are 16 bits.<br>10 = Reserved<br>11 = Reserved |
| 0 | DDMAEN | R/W | DDMA registers decode enable. Enables the decoding of the distributed DMA registers based on the value of DMABASE.<br>0 = Disabled (default)<br>1 = Enabled |

### capability ID register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Capability ID | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

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

### next-item pointer register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Next-item pointer | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Next-item pointer**
Type: Read-only
Offset: A1h
Default: 00h
Description: This register is used to indicate the next item in the linked list of the PCI power management capabilities. Because the PCI1211 functions include only one capabilities item, this register returns 0s when read.

### power-management capabilities register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Power-management capabilities | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| Default | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

Register: **Power-management capabilities**
Type: Read-only (see individual bit descriptions)
Offset: A2h
Default: 7E21h
Description: This register contains information on the capabilities of the PC Card function related to power management. Both PCI1211 CardBus bridge functions support D0, D2, and D3 power states. See Table 29 for a complete description of the register contents.

**Table 29. Power-Management Capabilities Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15–11 | PME_CAP | R | PME support. This 5-bit field indicates the power states from which the PCI1211 supports asserting $\overline{\text{PME}}$. A 0 for any bit indicates that the CardBus function cannot assert $\overline{\text{PME}}$ from that power state. These five bits return 01111b when read. Each of these bits is described below:<br>Bit 15 contains the value 0, indicating that $\overline{\text{PME}}$ cannot be asserted from $D3_{cold}$ state.<br>Bit 14 contains the value 1, indicating that $\overline{\text{PME}}$ can be asserted from $D3_{hot}$ state.<br>Bit 13 contains the value 1, indicating that $\overline{\text{PME}}$ can be asserted from D2 state.<br>Bit 12 contains the value 1, indicating that $\overline{\text{PME}}$ can be asserted from D1 state.<br>Bit 11 contains the value 1, indicating that $\overline{\text{PME}}$ can be asserted from the D0 state. |
| 10 | D2_CAP | R | D2 support. Bit 10 returns a 1 when read, indicating that the CardBus function supports the D2 device power state. |
| 9 | D1_CAP | R | D1 support. Bit 9 returns a 1 when read, indicating that the CardBus function supports the D1 device power state. |
| 8–6 | RSVD | R | Reserved. These bits return 000b when read. |
| 5 | DSI | R | Device-specific initialization. Bit 5 returns 1 when read, indicating that the CardBus controller function require 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. This bit returns 0 when read, indicating that the function supplies its own auxiliary power source. |
| 3 | PMECLK | R | $\overline{\text{PME}}$ clock. Bit 3 returns 0 when read, indicating that no host bus clock is required for the PCI1211 to generate $\overline{\text{PME}}$. |
| 2–0 | VERSION | R | Version. Bits 2–0 return 001b when read, indicating that there are four bytes of general-purpose power management (PM) registers as described in the *PCI Bus Power Management Interface Specification*, *Revision 1.0*. |

**power-management control/status register**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Power-management control/status | | | | | | | | | | | | | | | |
| Type | R/WC | R | R | R | R | R | R | R/W | R | R | R | R | R | R | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Power-management control/status**
Type: Read-only, Read/Write, Read/Write to Clear (see individual bit descriptions)
Offset: A4h
Default: 0000h
Description: This register determines and changes the current power state of the PCI1211 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. See Table 30 for a complete description of the register contents.

**Table 30. Power-Management Control/Status Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | PMESTAT | R/WC | PME status. Bit 15 is set when the CardBus function would normally assert $\overline{PME}$, independent of the state of the PME_EN bit. Bit 15 is cleared by a write back of 1, and this also clears the $\overline{PME}$ signal if $\overline{PME}$ was asserted by this function. Writing a 0 to this bit has no effect. |
| 14–13 | DATASCALE | R | Data scale. This 2-bit field returns 0s when read. The CardBus function does not return any dynamic data as indicated by the DYN_DATA bit. |
| 12–9 | DATASEL | R | Data select. This 4-bit field returns 0s when read. The CardBus function does not return any dynamic data as indicated by the DYN_DATA bit. |
| 8 | PME_EN | R/W | PME enable. Bit 8 enables the function to assert $\overline{PME}$. If this bit is cleared, assertion of $\overline{PME}$ is disabled. |
| 7–5 | RSVD | R | Reserved. Bits 7–5 return 0s when read. |
| 4 | DYN_DATA_PME_EN | R | Dynamic data PME enable. Bit 4 returns 0 when read since the CardBus function does not report dynamic data. |
| 3–2 | RSVD | R | Reserved. Bits 3–2 return 0s when read. |
| 1–0 | PWR_STATE | R/W | 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 = $D3_{hot}$ |

**power-management control/status register bridge support extensions**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Power-management control/status register bridge support extensions | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Power-management control/status register bridge support extensions**
Type: Read-only
Offset: A6h
Default: C0h
Description: The power-management control/status register bridge support extensions supports PCI bridge specific functionality. See Table 31 for a complete description of the register contents.

**Table 31. Power-Management Control/Status Register Bridge Support Extensions**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | BPCC_EN | R | Bus power/clock control. When read, bit 7 returns a 1. |
| 6 | $\overline{\text{B2_B3}}$ | R | B2/B3 support for $D3_{hot}$. ThIs bit returns a 1 when read. |
| 5–0 | RSVD | R | Reserved. These bits return 0s when read. |

**power management data register**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Power management data | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Power management data**
Type: Read-only
Offset: A7h
Default: 00h
Description: This register returns zeros when read since the CardBus functions do not report dynamic data.

**general-purpose event status register**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Power-management control/status | | | | | | | | | | | | | | | |
| Type | R/WC | R | R | R | R/WC | R | R | R/WC | R | R | R | R/WC | R/WC | R/WC | R/WC | R/WC |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **General-purpose event status**
Type: Read-only, Read/Write to Clear (see individual bit descriptions)
Offset: A8h
Default: 0000h
Description: This 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 1 to the corresponding bit location. The status bits in this register do not depend upon the state of a corresponding bit in the general-purpose enable register. See Table 32 for a complete description of the register contents.

**Table 32. General-Purpose Event Status Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | ZV_STS | R/WC | PC card ZV Status. Bit 15 is set on a change in status of the ZVENABLE bit in the PC card controller function of the PCI1211. |
| 14–12 | RSVD | R | Reserved. These bits return 0s when read. |
| 11 | PWR_STS | R/WC | Power change status. Bit 11 is set when software has changed the power state the socket. A change in either $V_{CC}$ or $V_{PP}$ for the socket causes this bit to be set. |
| 10–9 | RSVD | R | Reserved. These bits return 0s when read. |
| 8 | VPP12_STS | R/WC | 12 Volt $V_{PP}$ request status. Bit 8 is set when software has changed the requested Vpp level to or from 12 Volts for the PC Card socket. |
| 7–5 | RSVD | R | Reserved. These bits return 0s when read. |
| 4 | GP4_STS | R/WC | GPI4 Status. Bit 4 is set on a change in status of the MFUNC5 terminal input level. |
| 3 | GP3_STS | R/WC | GPI3 Status. Bit 3 is set on a change in status of the MFUNC4 terminal input level . |
| 2 | GP2_STS | R/WC | GPI2 Status. Bit 2 is set on a change in status of the MFUNC2 terminal input level. |
| 1 | GP1_STS | R/WC | GPI1 Status. Bit 1 is set on a change in status of the MFUNC1 terminal input level. |
| 0 | GP0_STS | R/WC | GPI0 Status. Bit 0 is set on a change in status of the MFUNC0 terminal input level. |

### general-purpose event enable register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | General-purpose event enable | | | | | | | | | | | | | | | |
| Type | R/W | R | R | R | R/W | R | R | R/W | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **General-purpose event enable**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: AAh
Default: 0000h
Description: This 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 be signaled only if one of the multifunction terminals, MFUNC6–MFUNC0, are configured for $\overline{\text{GPE}}$ signaling. See Table 33 for a complete description of the register contents.

**Table 33. General-Purpose Event Enable Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15 | ZV_EN | R/W | PC card socket ZV enable. When bit 15 is set, a $\overline{\text{GPE}}$ is signaled on a change in status of ZVENABLE in the PC Card controller function of the PCI1211. |
| 14–12 | RSVD | R | Reserved. These bits return 0s when read. |
| 11 | PWR_EN | R/W | Power change event enable. When bit 11 is set, a $\overline{\text{GPE}}$ is signaled on when software has changed the power state of the socket. |
| 10–9 | RSVD | R | Reserved. These bits return 0s when read. |
| 8 | VPP12_EN | R/W | 12 Volt $V_{PP}$ request event 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 Volts for the card socket. |
| 7–5 | RSVD | R | Reserved. These bits return 0s when read. |
| 4 | GP4_EN | R/W | GPI4 event 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 | R/W | GPI3 event 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 | R/W | GPI2 event 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 | R/W | GPI1 event 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 | R/W | GPI0 event 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. |

**general-purpose input register**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | General-purpose input | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | X | X | X | X | X |

Register: **General-purpose input**
Type: Read-only (see individual bit descriptions)
Offset: ACh
Default: 00XXh
Description: This register provides the logical value of the data input from the GPI terminals, MFUNC5–MFUNC4 and MFUNC2–MFUNC0. See Table 34 for a complete description of the register contents.

**Table 34. General-Purpose Input Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15–5 | RSVD | R | Reserved. Bits 15–5 return 0s when read. Writes have no effect. |
| 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. Writes have no effect. |
| 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. Writes have no effect. |
| 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. Writes have no effect. |
| 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. Writes have no effect. |
| 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. Writes have no effect. |

### general-purpose output register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | General-purpose output | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **General-purpose output**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: AEh
Default: 0000h
Description: This register is used for control of the general-purpose outputs. See Table 35 for a complete description of the register contents.

**Table 35. General-Purpose Output Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 15–5 | RSVD | R | Reserved. Bits 15–5 return 0s when read. Writes have no effect. |
| 4 | GPO4_DATA | R/W | 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. Reads return the last data value written. |
| 3 | GPO3_DATA | R/W | 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. Reads return the last data value written. |
| 2 | GPO2_DATA | R/W | 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. Reads return the last data value written. |
| 1 | GPO1_DATA | R/W | 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. Reads return the last data value written. |
| 0 | GPO0_DATA | R/W | 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. Reads return the last data value written. |

### serial bus data register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Serial bus data | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial bus data**
Type: Read/Write
Offset: B0h
Default: 00h
Description: This 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, and the serial bus slave address must be programmed with both 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 must be programmed with both the 7-bit slave address and the read/write indicator bit must be set, and the REQBUSY bit in the serial bus control and status register must be polled until clear. Then the contents of this register are valid read data from the serial bus interface. See Table 36 for a complete description of the register contents.

**Table 36. Serial Bus Data Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–0 | SBDATA | R/W | 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. |

### serial bus index register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Serial bus index | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial bus index**
Type: Read/Write
Offset: B1h
Default: 00h
Description: This 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 must be programmed with both the 7-bit slave address and the read/write indicator.

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 must be set, and the REQBUSY bit in the serial bus control and status register 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 37 for a complete description of the register contents.

**Table 37. Serial Bus Index Register**

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

### serial bus slave address register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Serial bus slave address | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial bus slave address**
Type: Read/Write
Offset: B2h
Default: 00h
Description: This 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 must be set, and the REQBUSY bit in the serial bus control and status register 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 38 for a complete description of the register contents.

**Table 38. Serial Bus Slave Address Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–1 | SLAVADDR | R/W | Serial bus slave address. This bit field represents the slave address of a read or write transaction on the serial interface. |
| 0 | RWCMD | R/W | 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 |

### serial bus control and status register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | Serial bus control and status | | | | | | | |
| Type | R/W | R | R | R | R/WC | R/W | R/WC | R/WC |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Serial bus control and status**
Type: Read-only, Read/Write, Read/Write to Clear (see individual bit descriptions)
Offset: B3h
Default: 00h
Description: This register is used to communicate serial bus status information and select the quick command protocol. The REQBUSY bit 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 39 for a complete description of the register contents.

**Table 39. Serial Bus Control and Status Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | PROT_SEL | R/W | 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 is not output by the PCI1211 when bit 7 is set. |
| 6 | RSVD | R | Reserved. Bit 6 returns 0 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. 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 PCI1211 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 | R/WC | Serial bus detect. When bit 3 is set, it indicates that the serial bus interface is detected. Pullup resistors must be implemented on the MFUNC1 and MFUNC4 (SDA and SCL) terminals for bit 3 to be set. 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 | R/W | 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 | R/WC | Requested serial bus access error. Bit 1 indicates when a data error occurs on the serial interface during a requested cycle, and may be set due to a missing acknowledge. Bit 1 is cleared by a write back of 1.<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 | R/WC | 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 may be set due to a missing acknowledge. Bit 0 is also set on invalid EEPROM data formats. Refer to *serial bus interface* on page 30 for details on EEPROM data format. Bit 0 is cleared by a write back of 1.<br>0 = No error detected during auto-load from serial bus EEPROM<br>1 = Data error detected during auto-load from serial bus EEPROM |

## ExCA compatibility registers

The exchangeable card architecture (ExCA) registers implemented in the PCI1211 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. The offsets from this base address run contiguous from 00h to 3Fh for the socket. Refer to Figure 20 for an ExCA I/O mapping illustration.

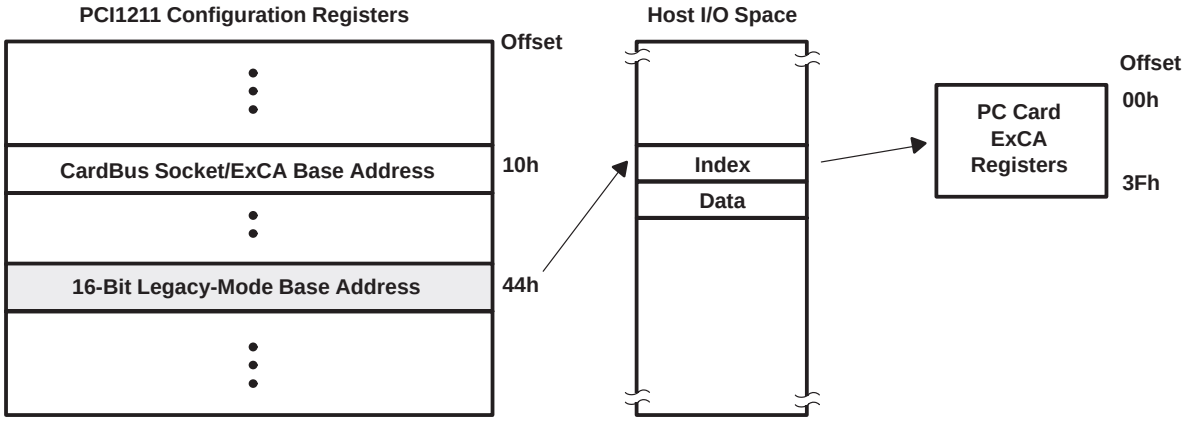



**Figure 20. ExCA Register Access Through I/O**

The TI PCI1211 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 registers/ExCA registers base address register (PCI register 10h) at memory offset 800h. Refer to Figure 21 for an ExCA memory mapping illustration. This illustration also identifies the CardBus socket register mapping, which is mapped into the same 4K window at memory offset 0h.

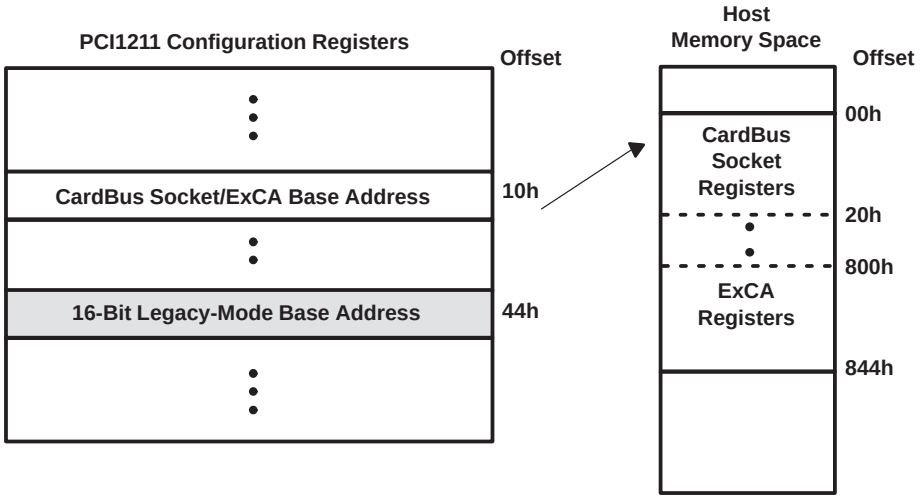



**Figure 21. ExCA Register Access Through Memory**

## ExCA compatibility registers (continued)

The interrupt registers, as defined by the 82365SL–DL Specification, in the ExCA register set 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 PCI1211 to ensure that all possible PCI1211 interrupts can potentially be routed to the programmable interrupt controller. The ExCA registers that are critical to the interrupt signaling are at memory address ExCA offset 803h and 805h.

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 40 identifies each ExCA register and its respective ExCA offset.) Memory windows have 4K-byte granularity.

**Table 40. 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 |

**Table 40. ExCA Registers and Offsets (Continued)**

| ExCA REGISTER NAME | PCI MEMORY ADDRESS OFFSET (HEX) | ExCA OFFSET (HEX) |
|---|---|---|
| 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 |
| 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 | – |

### ExCA identification and revision register (index 00h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA identification and revision | | | | | | | |
| Type | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

Register: **ExCA identification and revision**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 800h; ExCA offset 00h
Default: 84h
Description: This register provides host software with information on 16-bit PC Card support and Intel 82365SL-DF compatibility. See Table 41 for a complete description of the register contents.

**Table 41. ExCA Identification and Revision Register (Index 00h)**

| 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 PCI1211. The PCI1211 supports both I/O and memory 16-bit PC cards. |
| 5–4 | RSVD | R/W | Reserved. Bits 5–4 can be used for Intel 82365SL-DF emulation. |
| 3–0 | 365REV | R/W | Intel 82365SL-DF revision. This field stores the Intel 82365SL-DF revision supported by the PCI1211. Host software can read this field to determine compatibility to the Intel 82365SL-DF register set. This field defaults to 0100b upon PCI1211 reset. |

### ExCA interface status register (index 01h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA interface status | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | X | X | X | X | X | X |

Register: **ExCA interface status**
Type: Read-only (see individual bit descriptions)
Offset: CardBus socket address + 801h; ExCA offset 01h
Default: 00XX XXXXb
Description: This 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 42 for a complete description of the register contents.

**Table 42. ExCA Interface Status Register (Index 01h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | RSVD | R | Reserved. Bit 7 returns 0 when read. Writes have no effect. |
| 6 | CARDPWR | R | Card Power. Bit 6 indicates the current power status of the PC Card socket. This bit reflects how the power control register 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. Bit 4 indicates the current status of WP at the PC Card interface. This signal reports to the PCI1211 whether or not the memory card is write protected. Furthermore, write protection for an entire PCI1211 16-bit memory window is available by setting the appropriate bit in the memory window offset high-byte register.<br>0 = WP is 0. PC Card is R/W.<br>1 = WP is 1. PC Card is read-only. |
| 3 | CDETECT2 | R | Card detect 2. Bit 3 indicates the status of CD2 at the PC Card interface. Software may use this and CDETECT1 to determine if a PC Card is fully seated in the socket.<br>0 = CD2 is 1. No PC Card is inserted.<br>1 = CD2 is 0. PC Card is at least partially inserted. |
| 2 | CDETECT1 | R | Card detect 1. Bit 2 indicates the status of CD1 at the PC Card interface. Software may use this and CDETECT2 to determine if a PC Card is fully seated in the socket.<br>0 = CD1 is 1. No PC Card is inserted.<br>1 = CD1 is 0. 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 $\overline{\text{SPKR}}$ (bit 1) and 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. |

### ExCA power-control register (index 02h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA power control | | | | | | | |
| Type | R/W | R | R | R/W | R/W | R | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA power control**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 802h; ExCA offset 02h
Default: 00h
Description: This register provides PC Card power control. Bit 7 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 43 for a complete description of the register contents.

**Table 43. ExCA Power-Control Register (Index 02h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | COE | R/W | Card output enable. Bit 7 controls the state of all of the 16-bit outputs on the PCI1211. 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 6–5 return 0s when read. Writes have no effect. |
| 4–3 | EXCAVCC | R/W | $V_{CC}$. Bits 4–3 are used to 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 V |
| 2 | RSVD | R | Reserved. Bit 2 returns 0 when read. Writes have no effect. |
| 1–0 | EXCAVPP | R/W | $V_{PP}$. Bits 1–0 are used to request changes to card $V_{PP}$. The PCI1211 ignores this field unless $V_{CC}$ to the socket is enabled (i.e., 5 V or 3.3 V). This field is encoded as:<br>00 = 0 V (default)<br>01 = $V_{CC}$<br>10 = 12 V<br>11 = 0 V reserved |

### ExCA interrupt and general-control register (index 03h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA interrupt and general control | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA interrupt and general control**
Type: Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 803h; ExCA offset 03h
Default: 00h
Description: This register controls interrupt routing for I/O interrupts, as well as other critical 16-bit PC Card functions. See Table 44 for a complete description of the register contents.

**Table 44. ExCA Interrupt and General-Control Register (Index 03h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | RINGEN | R/W | Card ring indicate enable. Bit 7 enables the ring indicate function of BVD1/RI. This bit is encoded as:<br>0 = Ring indicate disabled (default)<br>1 = Ring indicate enabled |
| 6 | RESET | R/W | 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 | R/W | 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 | R/W | 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 in the ExCA card status change interrupt configuration register. 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 | R/W | 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 routed to PCI interrupts. This bit setting is OR'ed with ExCA bit 4 for backwards 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 |

### ExCA card status-change register (index 04h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA card status change | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA card status change**
Type: Read-only (see individual bit descriptions)
Offset: CardBus socket address + 804h; ExCA offset 04h
Default: 00h
Description: This register controls interrupt routing for I/O interrupts as well as other critical 16–bit PC Card functions. This register reflects the status of PC Card CSC interrupt sources. The card status change interrupt 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 0. When an interrupt source is enabled and that particular event occurs, 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 1 to the status bit. The choice of these two methods is based on the interrupt flag clear mode select, bit 2, in the global control register. See Table 45 for a complete description of the register contents.

**Table 45. ExCA Card Status-Change Register (Index 04h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–4 | RSVD | R | Reserved. Bits 7–4 return 0s when read. Writes have no effect. |
| 3 | CDCHANGE | R | Card detect change. Bit 3 indicates whether a change on CD1 or CD2 occurred at the PC Card interface. This bit is encoded as:<br>0 = No change detected on either CD1 or CD2<br>1 = Change detected on either CD1 or CD2 |
| 2 | READYCHANGE | R | Ready change. When a 16-bit memory is installed in the socket, bit 2 includes whether the source of a PCI1211 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 0. |
| 1 | BATWARN | R | Battery warning change. When a 16-bit memory card is installed in the socket, bit 1 indicates whether the source of a PCI1211 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 0. |
| 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 a PCI1211 interrupt was due to a battery dead condition. This bit is encoded as:<br>0 = STSCHG deasserted (default)<br>1 = STSCHG asserted<br>Ring indicate. When the PCI1211 is configured for ring indicate operation, bit 0 indicates the status of RI. |

### ExCA card status-change-interrupt configuration register (index 05h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA status-change-interrupt configuration | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA card status-change-interrupt configuration**
Type: Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 805h; ExCA offset 05h
Default: 00h
Description: This register controls interrupt routing for card status-change interrupts, as well as masking CSC interrupt sources. See Table 46 for a complete description of the register contents.

**Table 46. ExCA Card Status-Change-Interrupt Configuration Register (Index 05h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–4 | CSCSELECT | R/W | Interrupt select for card status change. Bits 7–4 select the interrupt routing for card status change interrupts. 0000 = CSC interrupts routed to PCI interrupts if bit 5 of the diagnostic register (PCI Offset 93h) is set to 1b. In this case bit 4 of ExCA 803 is a “don’t care”. This is the default setting.<br>0000 = No ISA interrupt routing if bit 5 of the diagnostic register (PCI Offset 93h) is set to 0b. In this case, CSC interrupts are routed to PCI interrupts by setting bit 4 of ExCA 803 to 1b. This field is encoded as:<br>0000 = No interrupt routing (default)<br>0001 = IRQ1 enabled<br>0010 = SMI enabled<br>0011 = IRQ3 enabled<br>0100 = IRQ4 enabled<br>0101 = IRQ5 enabled<br>0110 = 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 |
| 3 | CDEN | R/W | Card detect enable. Bit 3 enables interrupts on CD1 or CD2 changes. This bit is encoded as:<br>0 = Disables interrupts on CD1 or CD2 line changes (default)<br>1 = Enables interrupts on CD1 or CD2 line changes |
| 2 | READYEN | R/W | 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 | R/W | 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 | R/W | Battery dead enable. Bit 0 enables/disables a battery dead condition on a memory PC Card or assertion of the STSCHG I/O PC Card signal to generate a CSC interrupt.<br>0 = Disables host interrupt generation (default)<br>1 = Enables host interrupt generation |

### ExCA address window enable register (index 06h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA address window enable | | | | | | | |
| Type | R/W | R/W | R | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA address window enable**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 806h; ExCA offset 06h
Default: 00h
Description: This 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 PCI1211 does not acknowledge PCI memory or I/O cycles to the card if the corresponding enable bit in this register is 0, regardless of the programming of the memory or I/O window start/end/offset address registers. See Table 47 for a complete description of the register contents.

**Table 47. ExCA Address Window Enable Register (Index 06h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | IOWIN1EN | R/W | I/O window 1 enable. Bit 7 enables/disables I/O window 1 for the card. This bit is encoded as:<br>0 = I/O window 1 disabled (default)<br>1 = I/O window 1 enabled |
| 6 | IOWIN0EN | R/W | I/O window 0 enable. Bit 6 enables/disables I/O window 0 for the 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 0 when read. Writes have no effect. |
| 4 | MEMWIN4EN | R/W | Memory window 4 enable. Bit 4 enables/disables memory window 4 for the card. This bit is encoded as:<br>0 = Memory window 4 disabled (default)<br>1 = Memory window 4 enabled |
| 3 | MEMWIN3EN | R/W | Memory window 3 enable. Bit 3 enables/disables memory window 3 for the card. This bit is encoded as:<br>0 = Memory window 3 disabled (default)<br>1 = Memory window 3 enabled |
| 2 | MEMWIN2EN | R/W | Memory window 2 enable. Bit 2 enables/disables memory window 2 for the card. This bit is encoded as:<br>0 = Memory window 2 disabled (default)<br>1 = Memory window 2 enabled |
| 1 | MEMWIN1EN | R/W | Memory window 1 enable. Bit 1 enables/disables memory window 1 for the card. This bit is encoded as:<br>0 = Memory window 1 disabled (default)<br>1 = Memory window 1 enabled |
| 0 | MEMWIN0EN | R/W | Memory window 0 enable. Bit 0 enables/disables memory window 0 for the card. This bit is encoded as:<br>0 = Memory window 0 disabled (default)<br>1 = Memory window 0 enabled |

### ExCA I/O window control register (index 07h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O window control | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA I/O window control**
Type: Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 807h; ExCA offset 07h
Default: 00h
Description: This register contains parameters related to I/O window sizing and cycle timing. See Table 48 for a complete description of the register contents.

**Table 48. ExCA I/O Window Control Register (Index 07h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | WAITSTATE1 | R/W | 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 | R/W | 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 | IOSIS16W1 | R/W | I/O window 1 $\overline{\text{IOIS16}}$ source. Bit 5 controls the I/O window 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 DATASIZE1, bit 4 (default).<br>1 = Window data width determined by $\overline{\text{IOIS16}}$. |
| 4 | DATASIZE1 | R/W | I/O window 1 data size. Bit 4 controls the I/O window 1 data size. Bit 4 is ignored if the I/O window 1 $\overline{\text{IOIS16}}$ source bit (bit 5) 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 | R/W | 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 | R/W | 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 | IOSIS16W0 | R/W | 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 DATASIZE0, bit 0 (default).<br>1 = Window data width is determined by $\overline{\text{IOIS16}}$. |
| 0 | DATASIZE0 | R/W | I/O window 0 data size. Bit 0 controls the I/O window 0 data size. Bit 0 is ignored if the I/O window 0 $\overline{\text{IOIS16}}$ source bit (bit 1) is set. This bit is encoded as:<br>0 = Window data width is 8 bits (default).<br>1 = Window data width is 16 bits. |

### ExCA I/O window 0 and 1 start-address low-byte register (index 08h, 0Ch)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O window 0 and 1 start-address low byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the low byte of the 16-bit I/O window start address for I/O windows 0 and 1. The eight bits of these registers correspond to the lower eight bits of the start address.

### ExCA I/O window 0 and 1 start-address high-byte register (index 09h, 0Dh)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O window 0 and 1 start-address high byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the high byte of the 16-bit I/O window start address for I/O windows 0 and 1. The eight bits of these registers correspond to the upper eight bits of the end address.

### ExCA I/O window 0 and 1 end-address low-byte register (index 0Ah, 0Eh)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O window 0 and 1 end-address low byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the low byte of the 16-bit I/O window end address for I/O windows 0 and 1. The eight bits of these registers correspond to the lower eight bits of the end address.

### ExCA I/O window 0 and 1 end-address high-byte register (index 0Bh, 0Fh)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O window 0 and 1 end-address high byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the high byte of the 16-bit I/O window end address for I/O windows 0 and 1. The eight bits of these registers correspond to the upper eight bits of the end address.

### ExCA memory window 0–4 start-address low-byte register (index 10h, 18h, 20h, 28h, 30h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA memory window 0–4 start-address low byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the low byte of the 16-bit memory window start address for memory windows 0, 1, 2, 3, and 4. The eight bits of these registers correspond to bits A19–A12 of the start address.

**ExCA memory window 0–4 start-address high-byte register (index 11h, 19h, 21h, 29h, 31h)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA memory window 0–4 start-address high byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: 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 four 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 49 for a complete description of the register contents.

**Table 49. ExCA Memory Window 0–4 Start-Address High-Byte Register (Index 11h, 19h, 21h, 29h, 31h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | DATASIZE | R/W | 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 | R/W | 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 | R/W | Scratch pad bits. Bits 5–4 have no effect on memory window operation. |
| 3–0 | STAHN | R/W | Start-address high nibble. Bits 3–0 represent the upper address bits A23–A20 of the memory window start address. |

### ExCA memory window 0–4 end-address low-byte register (index 12h, 1Ah, 22h, 2Ah, 32h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA memory window 0–4 end-address low byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the low byte of the 16-bit memory window end address for memory windows 0, 1, 2, 3, and 4. The eight bits of these registers correspond to bits A19–A12 of the end address.

**ExCA memory window 0–4 end-address high-byte register (index 13h, 1Bh, 23h, 2Bh, 33h)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA memory window 0–4 end-address high byte | | | | | | | |
| Type | R/W | R/W | R | R | R/W | R/W | R/W | R/W |
| 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 (see individual bit descriptions)
Default: 00h
Size: One byte
Description: 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 four 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 50 for a complete description of the register contents.

**Table 50. ExCA Memory Window 0–4 End-Address High-Byte Register (Index 13h, 1Bh, 23h, 2Bh, 33h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–6 | MEMWS | R/W | Wait state. Bits 7–6 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 5–4 return 0s when read. Writes have no effect. |
| 3–0 | ENDHN | R/W | End-address high nibble. Bits 3–0 represent the upper address bits A23–A20 of the memory window end address. |

### ExCA memory window 0–4 offset-address low-byte register (index 14h, 1Ch, 24h, 2Ch, 34h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA memory window 0–4 offset-address low byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the low byte of the 16-bit memory window offset address for memory windows 0, 1, 2, 3 and 4. The eight bits of these registers correspond to bits A19–A12 of the offset address.

**ExCA memory window 0–4 offset-address high-byte register (index 15h, 1Dh, 25h, 2Dh, 35h)**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA memory window 0–4 offset-address high byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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; 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 (see individual bit descriptions)
Default: 00h
Size: One byte
Description: These registers contain the high six bits of the 16-bit memory window offset address for memory windows 0, 1, 2, 3 and 4. The lower six 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 51 for a complete description of the register contents.

**Table 51. ExCA Memory Window 0–4 Offset-Address High-Byte Register (Index 15h, 1Dh, 25h, 2Dh, 35h)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7 | WINWP | R/W | 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 | R/W | 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 | R/W | Offset-address high byte. Bits 5–0 represent the upper address bits A25–A20 of the memory window offset address. |

### ExCA I/O window 0 and 1 offset-address low-byte register (index 36h, 38h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O window 0 and 1 offset-address low byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| 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 (see description)
Default: 00h
Size: One byte
Description: These registers contain the low byte of the 16-bit I/O window offset address for I/O windows 0 and 1. The eight bits of these registers correspond to the lower eight bits of the offset address, and bit 0 is always 0.

### ExCA I/O window 0 and 1 offset-address high-byte register (index 37h, 39h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O window 0 and 1 offset-address high byte | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| 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
Size: One byte
Description: These registers contain the high byte of the 16-bit I/O window offset address for I/O windows 0 and 1. The eight bits of these registers correspond to the upper eight bits of the offset address.

### ExCA card detect and general-control register (index 16h)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA I/O card detect and general control | | | | | | | |
| Type | R | R | W | R/W | R | R | R/W | R |
| Default | X | X | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA card detect and general control**
Type: Read-only, Write-only, Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 816h; ExCA offset 16h
Default: XX00 0000b
Description: This 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 52 for a complete description of the register contents.

**Table 52. ExCA Card Detect and General-Control Register (Index 16h)**

| 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 | W | Software card detect interrupt. If the card detect enable bit in the card status change interrupt configuration register is set, writing a 1 to bit 5 causes a card-detect card-status change interrupt for the associated card socket. If the card detect enable bit is cleared to 0 in the card status change interrupt configuration register, writing a 1 to the software card detect interrupt bit has no effect. Bit 5 is write-only. A read always returns 0. |
| 4 | CDRESUME | R/W | Card detect resume enable. If bit 4 is set to 1, 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 the card status change bit in the card status change register is cleared. If this bit is a 0, 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 3–2 return 0s when read. Writes have no effect. |
| 1 | REGCONFIG | R/W | 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 0 when read. Writes have no effect. |

### ExCA global-control register (index 1Eh)

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA global control | | | | | | | |
| Type | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA global control**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 81Eh; ExCA offset 1Eh
Default: 00h
Description: This register controls the PC Card socket. The host interrupt mode bits in this register are retained for Intel 82365SL-DF compatibility. See Table 53 for a complete description of the register contents.

**Table 53. ExCA Global-Control Register (Index 1Eh)**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–5 | RSVD | R | Reserved. Bits 7–5 return 0s when read. Writes have no effect. |
| 4 | No function | R/W | This bit has no assigned function. |
| 3 | INTMODE | R/W | Level/edge interrupt mode select. Bit 3 selects the signaling mode for the PCI1211 host interrupt PC. This bit is encoded as:<br>0 = Host interrupt is edge mode (default).<br>1 = Host interrupt is level mode. |
| 2 | IFCMODE | R/W | Interrupt flag clear mode select. Bit 2 selects the interrupt flag clear mechanism for the flags in the ExCA card status change register. 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 write back of 1. |
| 1 | CSCMODE | R/W | Card status change level/edge mode select. Bit 1 selects the signaling mode for the PCI1211 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 | R/W | Power-down mode select. When bit 0 is set to 1, the PCI1211 is in power-down mode. In power-down mode, the PCI1211 card outputs are 3-stated until an active cycle is executed on the card interface. Following an active cycle, the outputs are again 3-stated. The PCI1211 still receives DMA requests, 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. |

### ExCA memory window 0–4 page register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ExCA memory window 0–4 page | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **ExCA memory window 0–4 page**
Type: Read/Write
Offset: CardBus socket address + 840h 841h, 842h, 843h, 844h
Default: 00h
Description: 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 one of 256 16M-byte regions in the 4G-byte PCI address space. These registers are accessible only when the ExCA registers are memory mapped, i.e., these registers can not be accessed using the index/data I/O scheme.

## CardBus socket registers

The PCMCIA CardBus specification requires a CardBus socket controller to provide five 32-bit registers that report and control socket-specific functions. The PCI1211 provides the CardBus socket/ExCA base address register (PCI offset 10h) to locate these CardBus socket registers in PCI memory address space. Each socket has a separate base address register for accessing the CardBus socket registers (see Figure 22). Table 54 gives the location of the socket registers in relation to the CardBus socket/ExCA base address.

The PCI1211 implements an additional register at offset 20h that provides power management control for the socket.

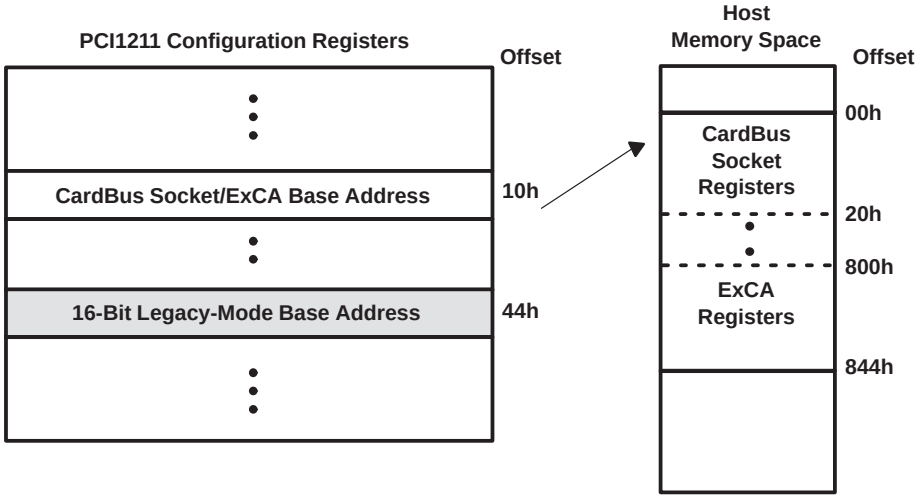



**Figure 22. Accessing CardBus Socket Registers Through PCI Memory**

**Table 54. 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 |
| Reserved | 18h |
| Reserved | 1Ch |
| Socket power management | 20h |

**socket event register**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket event | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | Socket event | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R/WC | R/WC | R/WC | R/WC |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket event**
Type: Read-only, Read/Write to Clear (see individual bit descriptions)
Offset: CardBus socket address + 00h
Default: 0000 0000h
Description: This register indicates that 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 for current status. Each bit in this register can be cleared by writing a 1 to that bit. The bits in this register can be set to a 1 by software by writing a 1 to the corresponding bit in the socket force event register. 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 (i.e., 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, an interrupt is generated (but not masked) based on any bit set. See Table 55 for a complete description of the register contents.

**Table 55. Socket Event Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–4 | RSVD | R | Reserved. Bits 31–4 return 0s when read. |
| 3 | PWREVENT | R/WC | Power cycle. Bit 3 is set when the PCI1211 detects that the PWRCYCLE bit in the socket present-state register has changed. This bit is cleared by writing a 1. |
| 2 | CD2EVENT | R/WC | $\overline{\text{CCD2}}$. Bit 2 is set when the PCI1211 detects that the CDETECT2 field in the socket present-state register has changed. This bit is cleared by writing a 1. |
| 1 | CD1EVENT | R/WC | $\overline{\text{CCD1}}$. Bit 3 is set when the PCI1211 detects that the CDETECT1 field in the socket present-state register has changed. This bit is cleared by writing a 1. |
| 0 | CSTSEVENT | R/WC | CSTSCHG. Bit 0 is set when the CARDSTS field in the socket present-state register has changed state. For CardBus cards, bit 0 is set on the rising edge of CSTSCHG. For 16-bit PC Cards, bit 0 is set on both transitions of CSTSCHG. This bit is reset by writing a 1. |

### socket mask register

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket mask | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | Socket mask | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket mask**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 04h
Default: 0000 0000h
Description: This 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. See Table 56 for a complete description of the register contents.

**Table 56. Socket Mask Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–4 | RSVD | R | Reserved. Bits 31–4 return 0s when read. |
| 3 | PWRMASK | R/W | Power cycle. Bit 3 masks the PWRCYCLE bit in the socket present state register 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 | R/W | Card detect mask. Bits 2–1 mask the CDETECT1 and CDETECT2 bits in the socket present-state register 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 | R/W | CSTSCHG mask. Bit 0 masks the CARDSTS field in the socket present-state register from causing a CSC interrupt.<br>0 = CARDSTS event does not cause CSC interrupt (default).<br>1 = CARDSTS event causes CSC interrupt. |

**socket present-state register**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket present state | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | Socket present state | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | X | 0 | 0 | 0 | X | X | X |

Register: **Socket present state**
Type: Read-only
Offset: CardBus socket address + 08h
Default: 3000 00XXh
Description: This register reports information about the socket interface. Writes to the socket force event register are reflected here, as well as general socket interface status. Information about PC Card $\overline{V_{CC}}$ support and card type is updated only at each insertion. Also note that the PCI1211 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 57 for a complete description of the register contents.

**Table 57. Socket Present-State Register**

| 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 PCI1211 does not support Y.Y-V $V_{CC}$; therefore, this bit is always reset unless overridden by the socket force event register. |
| 30 | XVSOCKET | R | XV socket. Bit 30 indicates whether or not the socket can supply $V_{CC}$ = X.X V to PC Cards. The PCI1211 does not support X.X-V $V_{CC}$; therefore, this bit is always reset unless overridden by the socket force event register. |
| 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 PCI1211 does support 3.3-V $V_{CC}$; therefore, this bit is always set unless overridden by the socket force event register. |
| 28 | 5VSOCKET | R | 5-V socket. Bit 28 indicates whether or not the socket can supply $V_{CC}$ = 5 V to PC Cards. The PCI1211 does support 5-V $V_{CC}$; therefore, this bit is always set unless overridden by the socket force event register. |
| 27–14 | RSVD | R | Reserved. Bits 27–14 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. |
| 12 | XVCARD | R | XV card. Bit 12 indicates whether or not the PC Card inserted in the socket supports $V_{CC}$ = X.X V. |
| 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. |
| 10 | 5VCARD | R | 5-V card. Bit 10 indicates whether or not the PC Card inserted in the socket supports $V_{CC}$ = 5 V. |
| 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 PCI1211.<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 that 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 | CSTSCHG. Bit 0 reflects the current status of CSTSCHG at the PC Card interface.<br>0 = CSTSCHG low<br>1 = CSTSCHG high |

### socket force event register

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket force event | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | Socket force event | | | | | | | | | | | | | | | |
| Type | R | W | W | W | W | W | W | W | W | R | W | W | W | W | W | W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket force event**
Type: Read-only, Write-only (see individual bit descriptions)
Offset: CardBus socket address + 0Ch
Default: 0000 0000h
Description: This register is used to force changes to the socket event register and the socket present state register. The CVSTEST bit in this register must be written when forcing changes that require card interrogation. See Table 58 for a complete description of the register contents.

**Table 58. Socket Force Event Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–15 | RSVD | R | Reserved. Bits 31–15 return 0s when read. |
| 14 | CVSTEST | W | Card VS test. When bit 14 is set, the PCI1211 reinterrogates the PC Card, updates the socket present state register, and reenables the socket power control. |
| 13 | FYVCARD | W | Force YV card. Writes to bit 13 cause the YVCARD bit in the socket present state register to be written. When set, this bit disables the socket power control. |
| 12 | FXVCARD | W | Force XV card. Writes to bit 12 cause the XVCARD bit in the socket present state register to be written. When set, this bit disables the socket power control. |
| 11 | F3VCARD | W | Force 3-V card. Writes to bit 11 cause the 3VCARD bit in the socket present state register to be written. When set, this bit disables the socket power control. |
| 10 | F5VCARD | W | Force 5-V card. Writes to bit 10 cause the 5VCARD bit in the socket present state register to be written. When set, this bit disables the socket power control. |
| 9 | FBADVCCREQ | W | Force bad $V_{CC}$ request. Changes to the BADVCCREQ bit in the socket present state register can be made by writing to bit 9. |
| 8 | FDATALOST | W | Force data lost. Writes to bit 8 cause the DATALOST bit in the socket present state register to be written. |
| 7 | FNOTACARD | W | Force not a card. Writes to bit 7 cause the NOTACARD bit in the socket present state register to be written. |
| 6 | RSVD | R | Reserved. Bit 6 returns 0 when read. |
| 5 | FCBCARD | W | Force CardBus card. Writes to bit 5 cause the CBCARD bit in the socket present state register to be written. |
| 4 | F16BITCARD | W | Force 16-bit card. Writes to bit 4 cause the 16BITCARD bit in the socket present state register to be written. |
| 3 | FPWRCYCLE | W | Force power cycle. Writes to bit 3 cause the PWREVENT bit in the socket event register to be written, and the PWRCYCLE bit in the socket present state register is unaffected. |
| 2 | FCDETECT2 | W | Force $\overline{\text{CCD2}}$. Writes to bit 2 cause the CD2EVENT bit in the socket event register to be written, and the CDETECT2 bit in the socket present state register is unaffected. |
| 1 | FCDETECT1 | W | Force $\overline{\text{CCD1}}$. Writes to bit 1 cause the CD1EVENT bit in the socket event register to be written, and the CDETECT1 bit in the socket present state register is unaffected. |
| 0 | FCARDSTS | W | Force CSTSCHG. Writes to bit 0 cause the CSTSEVENT bit in the socket event register to be written, and the CARDSTS bit in the socket present state register is unaffected. |

### socket control register

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket control | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| 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 |
| Name | Socket control | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R/W | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **Socket control**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: CardBus socket address + 10h
Default: 0000 0000h
Description: This register provides control of the voltages applied to the socket and instructions for CB $\overline{\text{CLKRUN}}$ protocol. The PCI1211 ensures that the socket is powered up only at acceptable voltages when a CardBus card is inserted. See Table 59 for a complete description of the register contents.

**Table 59. Socket Control Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–8 | RSVD | R | Reserved. Bits 31–8 return 0s when read. |
| 7 | STOPCLK | R/W | 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. (default)<br>1 = CB $\overline{\text{CLKRUN}}$ protocol can attempt to stop/slow the CB clock if the socket is idle. |
| 6–4 | VCCCTRL | R/W | $V_{CC}$ control. Bits 6–4 are used to 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 0 when read. |
| 2–0 | VPPCTRL | R/W | $V_{PP}$ control. Bits 2–0 are used to 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 |

**socket power management register**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Socket power management | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W |
| 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 |
| Name | Socket power management | | | | | | | | | | | | | | | |
| Type | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W |
| 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 (see individual bit descriptions)
Offset: CardBus socket address + 20h
Default: 0000 0000h
Description: 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 60 for a complete description of the register contents.

**Table 60. Socket Power Management Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 31–26 | RSVD | R | Reserved. Bits 31–26 return 0s 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 0s when read. |
| 16 | CLKCTRLEN | R/W | CardBus clock control enable. When bit 16 is set, clock control (CLKCTRL bit 0) 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 | R/W | CardBus clock control. The bit determines whether the CB $\overline{\text{CLKRUN}}$ protocol will attempt to stop or slow the CB clock during idle states. Bit 16 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. |

## distributed DMA (DDMA) registers

The DMA base address, programmable in PCI configuration space at offset 98h, points to a 16-byte region in PCI I/O space where the DDMA registers reside. The names and locations of these registers are summarized in Table 61. These PCI1211 register definitions are identical in function, but differ in location, to the 8237 DMA controller. The similarity between the register models retains some level of compatibility with legacy DMA and simplifies the translation required by the master DMA device when it forwards legacy DMA writes to DMA channels.

While the DMA register definitions are identical to those in the 8237 of the same name, some register bits defined in the 8237 do not apply to distributed DMA in a PCI environment. In such cases, the PCI1211 implements these obsolete register bits as read-only nonfunctional bits. The reserved registers shown in Table 61 are implemented as read-only and return 0s when read. Writes to reserved registers have no effect.

**Table 61. Distributed DMA Registers**

| TYPE | REGISTER NAME | | | | DMA BASE ADDRESS OFFSET (HEX) |
|---|---|---|---|---|---|
| R | Reserved | Page | Current address | | 00 |
| W | | | Base address | | |
| R | Reserved | Reserved | Current count | | 04 |
| W | | | Base count | | |
| R | N/A | Reserved | N/A | Status | 08 |
| W | Mode | | Request | Command | |
| R | Multichannel | Reserved | N/A | Reserved | 0C |
| W | Mask | | Master clear | | |

**DDMA current address/base address register**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA current address/base address | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | DDMA current address/base address | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA current address/base address**
Type: Read/Write
Offset: DDMA base address + 00h
Default: 0000h
Size: Two bytes
Description: This register is used to set the starting (base) memory address of a DDMA transfer. Reads from this register indicate the current memory address of a direct memory transfer.

For the 8-bit DDMA transfer mode, the current address register contents are presented on AD15–AD0 of the PCI bus during the address phase. Bits 7–0 of the page register are presented on AD23–AD16 of the PCI bus during the address phase.

For the 16-bit DDMA transfer mode, the current address register contents are presented on AD16–AD1 of the PCI bus during the address phase, and AD0 is driven to logic 0. Bits 7–1 of the page register are presented on AD23–AD17 of the PCI bus during the address phase, and bit 0 is ignored.

**DDMA page register**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA page | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA page**
Type: Read/Write
Offset: DDMA base address + 02h
Default: 00h
Size: One byte
Description: This register is used to set the upper byte of the address of a DDMA transfer. Details of the address represented by this register are explained in *DDMA current address/base address register*.

### DDMA current count/base count register

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA current count/base count | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Name | DMA current count/base count | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA current count/base count**
Type: Read/Write
Offset: DDMA base address + 04h
Default: 0000h
Size: Two bytes
Description: This register is used to set the total transfer count, in bytes, of a direct memory transfer. Reads to this register indicate the current count of a direct memory transfer. In the 8-bit transfer mode, the count is decremented by 1 after each transfer. Likewise, the count is decremented by 2 in the 16-bit transfer mode.

### DDMA command register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA command | | | | | | | |
| Type | R | R | R | R | R | R/W | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA command**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: DDMA base address + 08h
Default: 00h
Size: One byte
Description: This register is used to enable and disable the DMA controller. Bit 2, the only read/write bit, defaults to 0 enabling the DMA controller. All other bits are reserved. See Table 62 for a complete description of the register contents.

**Table 62. DDMA Command Register**

| BIT | TYPE | TYPE | FUNCTION |
|---|---|---|---|
| 7–3 | RSVD | R | Reserved. Bits 7–3 return 0s when read. |
| 2 | DMAEN | R/W | DMA controller enable. Bit 2 enables and disables the distributed DMA slave controller in the PCI1211 and defaults to the enabled state.<br>0 = DMA controller enabled (default)<br>1 = DMA controller disabled |
| 1–0 | RSVD | R | Reserved. Bits 1–0 return 0s when read. |

### DDMA status register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA status | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA status**
Type: Read-only (see individual bit descriptions)
Offset: DDMA base address + 08h
Default: 00h
Size: One byte
Description: This register indicates the terminal count and DMA request ($\overline{\text{DREQ}}$) status. See Table 63 for a complete description of the register contents.

**Table 63. DDMA Status Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–4 | DREQSTAT | R | Channel request. In the 8237, bits 7–4 indicate the status of $\overline{\text{DREQ}}$ of each DMA channel. In the PCI1211, these bits indicate the $\overline{\text{DREQ}}$ status of the single socket being serviced by this register. All four bits are set when the PC Card asserts $\overline{\text{DREQ}}$ and are reset when $\overline{\text{DREQ}}$ is deasserted. The status of the mask bit in the multichannel mask register has no effect on these bits. |
| 3–0 | TC | R | Channel terminal count. The 8327 uses bits 3–0 to indicate the TC status of each of its four DMA channels. In the PCI1211, these bits report information about a single DMA channel; therefore, all four of these register bits indicate the TC status of the single socket being serviced by this register. All four bits are set when the TC is reached by the DMA channel. These bits are reset when read or the DMA channel is reset. |

### DDMA request register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA request | | | | | | | |
| Type | W | W | W | W | W | W | W | W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA request**
Type: Write-only
Offset: DDMA base address + 09h
Default: 00h
Size: One byte
Description: This register is used to request a DDMA transfer through software. Any write to this register enables software requests, and this register is to be used in block mode only.

### DDMA mode register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA mode | | | | | | | |
| Type | R/W | R/W | R/W | R/W | R/W | R/W | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA mode**
Type: Read-only, Read/Write (see individual bit descriptions)
Offset: DDMA base address + 0Bh
Default: 00h
Size: One byte
Description: This register is used to set the DDMA transfer mode. See Table 64 for a complete description of the register contents.

**Table 64. DDMA Mode Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–6 | DMAMODE | R/W | Mode select. The PCI1211 uses bits 7–6 to determine the transfer mode.<br>00 = Demand mode select (default)<br>01 = Single mode select<br>10 = Block mode select<br>11 = Reserved |
| 5 | INCDEC | R/W | Address increment/decrement. The PCI1211 uses bit 5 to select the memory address in the current address/base address register to increment or decrement after each data transfer. This is in accordance with the 8237 use of this register bit, and is encoded as follows:<br>0 = Addresses increment (default).<br>1 = Addresses decrement. |
| 4 | AUTOINIT | R/W | Auto initialization<br>0 = Auto initialization disabled (default)<br>1 = Auto initialization enabled |
| 3–2 | XFERTYPE | R/W | Transfer type. Bits 3–2 select the type of direct memory transfer to be performed. A memory write transfer moves data from the PCI1211 PC Card interface to memory, and a memory read transfer moves data from memory to the PCI1211 PC Card interface. The field is encoded as:<br>00 = No transfer selected (default)<br>01 = Write transfer<br>10 = Read transfer<br>11 = Reserved |
| 1–0 | RSVD | R | Reserved. Bits 1–0 return 0s when read. |

### DDMA master clear register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA master clear | | | | | | | |
| Type | W | W | W | W | W | W | W | W |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA master clear**
Type: Write-only
Offset: DDMA base address + 0Dh
Default: 00h
Size: One byte
Description: This register is used to reset the DMA controller and resets all DDMA registers.

### DDMA multichannel/mask register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | DDMA multichannel/mask | | | | | | | |
| Type | R | R | R | R | R | R | R | R |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register: **DDMA multichannel/mask**
Type: Read-only (see individual bit descriptions)
Offset: DDMA base address + 0Fh
Default: 00h
Size: One byte
Description: The PCI1211 uses only the least-significant bit of this register to mask the PC Card DMA channel. The PCI1211 sets the mask bit when the PC Card is removed. Host software is responsible for either resetting the socket's DMA controller or reenabling the mask bit. See Table 65 for a complete description of the register contents.

**Table 65. DDMA Multichannel/Mask Register**

| BIT | SIGNAL | TYPE | FUNCTION |
|---|---|---|---|
| 7–1 | RSVD | R | Reserved. Bits 7–1 return 0s when read. |
| 0 | MASKBIT | R | Mask select. Bit 0 masks incoming $\overline{\text{DREQ}}$ signals from the PC Card. When set, the socket ignores DMA requests from the card. When cleared (or when reset), incoming $\overline{\text{DREQ}}$ assertions are serviced normally.<br>0 = DDMA service provided on card $\overline{\text{DREQ}}$<br>1 = Socket $\overline{\text{DREQ}}$ signal ignored (default) |

**absolute maximum ratings over operating temperature ranges** (unless otherwise noted)†

| | | |
|---|---|---|
| Supply voltage range, $V_{CC}$ | | –0.5 V to 4 V |
| Clamping voltage range, $V_{CCCB}$, $V_{CCI}$, $V_{CCP}$ | | –0.5 V to 6 V |
| Input voltage range, $V_I$: | PCI | –0.5 V to $V_{CCP}$ + 0.5 V |
| | Card | –0.5 to $V_{DD0}$ + 0.5 V |
| | MISC | –0.5 to $V_{CCI}$ + 0.5 V |
| | Fail safe | –0.5 V to $V_{CC}$ + 0.5 V |
| Output voltage range, $V_O$: | PCI | –0.5 V to $V_{CCP}$ + 0.5 V |
| | Card | –0.5 to $V_{DD0}$ + 0.5 V |
| | MISC | –0.5 to $V_{CCI}$ + 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 are measured with respect to $V_{CCP}$ instead of $V_{CC}$. PC Card terminals are measured with respect to $V_{CCCB}$. Miscellaneous signals are measured with respect to $V_{CCI}$. 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 are measured with respect to $V_{CCP}$ instead of $V_{CC}$. PC Card terminals are measured with respect to $V_{CCCB}$. Miscellaneous signals are measured with respect to $V_{CCI}$. The limit specified applies for a dc condition.

## recommended operating conditions (see Note 3)

| | | | OPERATION | MIN | NOM | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage (core) | Commercial | 3.3 V | 3 | 3.3 | 3.6 | V |
| $V_{CCP}$ | PCI I/O clamping voltage | Commercial | 3.3 V | 3 | 3.3 | 3.6 | V |
| | | | 5 V | 4.75 | 5 | 5.25 | |
| $V_{CCCB}$ | PC Card I/O clamping voltage | Commercial | 3.3 V | 3 | 3.3 | 3.6 | V |
| | | | 5 V | 4.75 | 5 | 5.25 | |
| $V_{CCI}$ | Miscellaneous I/O clamping 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 | $0.5V_{CCP}$ | | $V_{CCP}$ | V |
| | | | 5 V | 2 | | $V_{CCP}$ | |
| | | PC Card | 3.3 V | $0.475V_{CCCB}$ | | $V_{CCCB}$ | |
| | | | 5 V | 2.4 | | $V_{CCCB}$ | |
| | | MISC‡ | | 2 | | $V_{CCI}$ | |
| | | Fail safe§ | | 2 | | $V_{CC}$ | |
| | | CD Pins* | | 2.2 | | $V_{CC}$ | |
| $V_{IL}$† | Low-level input voltage | PCI | 3.3 V | 0 | | $0.3V_{CCP}$ | V |
| | | | 5 V | 0 | | 0.8 | |
| | | PC Card | 3.3 V | 0 | | $0.325V_{CCCB}$ | |
| | | | 5 V | 0 | | 0.8 | |
| | | MISC‡ | | 0 | | 0.8 | |
| | | Fail safe§ | | 0 | | 0.8 | |
| | | CD Pins* | | 0 | | 0.75 | |
| $V_I$ | Input voltage | PCI | | 0 | | $V_{CCP}$ | V |
| | | PC Card | | 0 | | $V_{CCCB}$ | |
| | | MISC‡ | | 0 | | $V_{CCI}$ | |
| | | Fail safe§ | | 0 | | $V_{CC}$ | |
| $V_O$¶ | Output voltage | PCI | | 0 | | $V_{CC}$ | V |
| | | PC Card | | 0 | | $V_{CC}$ | |
| | | MISC‡ | | 0 | | $V_{CC}$ | |
| | | Fail safe§ | | 0 | | $V_{CC}$ | |
| $t_t$ | Input transition time ($t_r$ and $t_f$) | PCI and PC Card | | 1 | | 4 | ns |
| | | Miscellaneous, and fail safe | | 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 pins are 70, 62, 59, 60, 61, 64, 65, 67, 68, and 69 for the PGE packaged device and L11, M9, L8, K8, N9, K9, N10, L10, N11, and M11 for the GGU packaged device ($\overline{\text{SUSPEND}}$, SPKROUT, $\overline{\text{RI_OUT}}$, multifunction terminals (MFUNC0–MFUNC6), and power switch control pins).

§ Fail-safe pins are 75, 117, 131, and 137 for the PGE packaged device and L12, D9, C6, and A4 for the GGU packaged device (card detect and voltage sense pins).

* CD Pins are 75 and 137.

¶ Applies to external output buffers

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

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

## electrical characteristics over recommended operating conditions (unless otherwise noted)

| PARAMETER | | PINS | OPERATION | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | High-level output voltage (see Note 4) | PCI and Misc1§ | 3.3 V | $I_{OH}$ = –0.5 mA | $0.9V_{CC}$ | | V |
| | | | 5 V | $I_{OH}$ = –2 mA | 2.4 | | |
| | | PC Card | 3.3 V | $I_{OH}$ = –0.15 mA | $0.9V_{CC}$ | | |
| | | | 5 V | $I_{OH}$ = –0.15 mA | 2.4 | | |
| | | Misc2¶ | | $I_{OH}$ = –12 mA | 2.1 | | |
| | | Misc3# | | $I_{OH}$ = –4 mA | 2.1 | | |
| $V_{OL}$ | Low-level output voltage | PCI and Misc1§ | 3.3 V | $I_{OL}$ = 1.5 mA | | $0.1V_{CC}$ | V |
| | | | 5 V | $I_{OL}$ = 6 mA | | 0.55 | |
| | | PC Card | 3.3 V | $I_{OL}$ = 0.7 mA | | $0.1V_{CC}$ | |
| | | | 5 V | $I_{OL}$ = 0.7 mA | | 0.55 | |
| | | Misc2¶ | | $I_{OL}$ = 12 mA | | 0.5 | |
| | | Misc3# | | $I_{OL}$ = 4 mA | | 0.5 | |
| $I_{OZL}$ | 3-state output, high-impedance state output current (see Note 4) | Output pins | 3.6 V | $V_I = V_{CC}$ | | –1 | µA |
| | | | 5.25 V | $V_I = V_{CC}$ | | –1 | |
| $I_{OZH}$ | 3-state output, high-impedance state output current | Output pins | 3.6 V | $V_I = V_{CC}$† | | 10 | µA |
| | | | 5.25 V | $V_I = V_{CC}$† | | 25 | |
| $I_{IL}$ | Low-level input current (see Note 5) | Input pins | | $V_I$ = GND | | –1 | µA |
| | | I/O pins | | $V_I$ = GND | | –10 | |
| $I_{IH}$ | High-level input current | Input pins | 3.6 V | $V_I = V_{CC}$‡ | | 10 | µA |
| | | | 5.25 V | $V_I = V_{CC}$‡ | | 20 | |
| | | I/O pins | 3.6 V | $V_I = V_{CC}$‡ | | 10 | |
| | | | 5.25 V | $V_I = V_{CC}$‡ | | 25 | |
| | | Fail-safe pins | 3.6 V | $V_I = V_{CC}$ | | 10 | |
| | | $V_{CCD}$ pins$ | 3.6 V | $V_I = V_{CC}$ | | 300 | |

§ Misc1 includes MFUNC6(69), MFUNC5(68), MFUNC4(67), MFUNC3(65), and MFUNC2(64).
¶ Misc2 includes MFUNC1(61), MFUNC0(60), and SERR(35).
# Misc3 includes SPKROUT(62) and RI_OUT(59).
† For PCI pins, $V_I = V_{CCP}$. For PC Card pins, $V_I = V_{CCCB}$. For miscellaneous pins, $V_I = V_{CCI}$
‡ For I/O pins, input leakage ($I_{IL}$ and $I_{IH}$) includes $I_{OZ}$ leakage of the disabled output.
$ $V_{CCD}$ pins include VCCD0(73) and VCCD1(74).

NOTES: 4. $V_{OH}$ and $I_{OL}$ are not tested on SERR(35, M1) and RI_OUT(59, L8) because they are open drain outputs.
5. $I_{IL}$ is not tested on VCCD0 (73, N13) and VCCD1(74, M13) because they are pulled down with an internal resistor.

**PCI clock/reset timing requirements over recommended ranges of supply voltage and operating free-air temperature** (see Figure 24 and Figure 25)

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

**PCI timing requirements over recommended ranges of supply voltage and operating free-air temperature** (see Note 6 and Figures 19 and 22)

| PARAMETER | | | ALTERNATE SYMBOL | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{pd}$ | Propagation delay time, See Note 7 | PCLK-to-shared signal valid delay time | $t_{val}$ | $C_L$ = 50 pF, See Note 7 | | 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 |

NOTES: 6. This data sheet 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, $t_d$ = delay time, $t_{su}$ = setup time, and $t_h$ = hold time.

7. PCI shared signals are AD31–AD0, C/$\overline{\text{BE3}}$–$\overline{0}$, $\overline{\text{FRAME}}$, $\overline{\text{TRDY}}$, $\overline{\text{IRDY}}$, $\overline{\text{STOP}}$, IDSEL, $\overline{\text{DEVSEL}}$, and PAR.

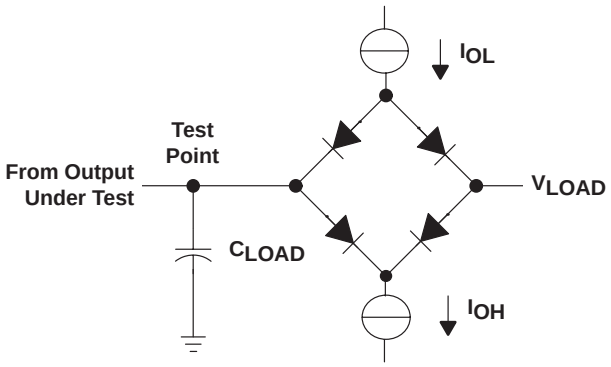
## PARAMETER MEASUREMENT INFORMATION

**LOAD CIRCUIT PARAMETERS**

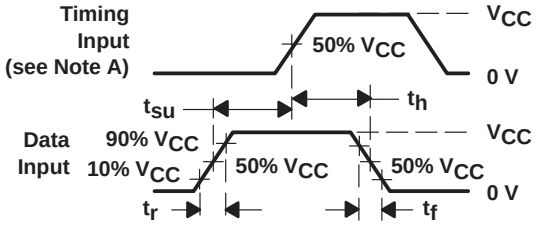
| TIMING PARAMETER | | $C_{LOAD}$† (pF) | $I_{OL}$ (mA) | $I_{OH}$ (mA) | $V_{LOAD}$ (V) |
|---|---|---|---|---|---|
| $t_{en}$ | $t_{PZH}$ | 50 | 8 | –8 | 0 |
| | $t_{PZL}$ | | | | 3 |
| $t_{dis}$ | $t_{PHZ}$ | 50 | 8 | –8 | 1.5 |
| | $t_{PLZ}$ | | | | |
| $t_{pd}$ | | 50 | 8 | –8 | ‡ |

† $C_{LOAD}$ includes the typical load-circuit distributed capacitance

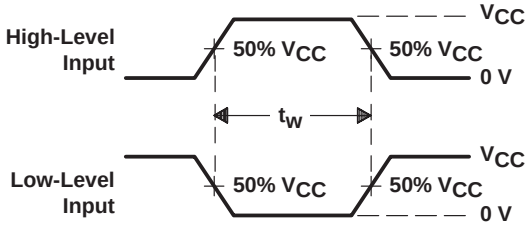
‡ $\frac{V_{LOAD} - V_{OL}}{I_{OL}}$ = 50 Ω, where $V_{OL}$ = 0.6 V, $I_{OL}$ = 8 mA



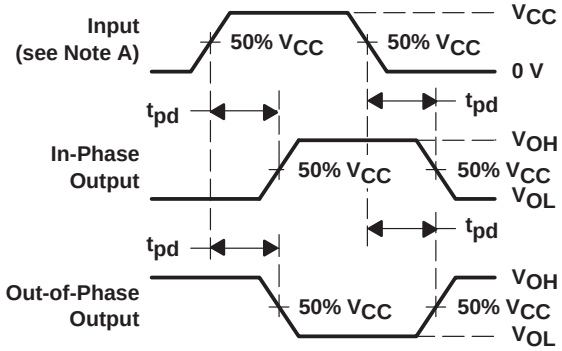

**LOAD CIRCUIT**



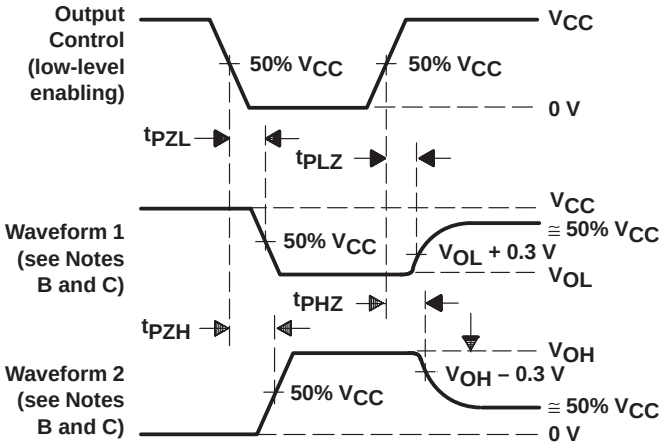

**VOLTAGE WAVEFORMS**
**SETUP AND HOLD TIMES**
**INPUT RISE AND FALL TIMES**




**VOLTAGE WAVEFORMS**
**PULSE DURATION**




**VOLTAGE WAVEFORMS**
**PROPAGATION DELAY TIMES**




**VOLTAGE WAVEFORMS**
**ENABLE AND DISABLE TIMES, 3-STATE OUTPUTS**

NOTES: A. Phase relationships between waveforms were chosen arbitrarily. All input pulses are supplied by pulse generators having the following characteristics: PRR = 1 MHz, $Z_O$ = 50 Ω, $t_r$ = 6 ns.
B. Waveform 1 is for an output with internal conditions such that the output is low except when disabled by the output control. Waveform 2 is for an output with internal conditions such that the output is high except when disabled by the output control.
C. For $t_{PLZ}$ and $t_{PHZ}$, $V_{OL}$ and $V_{OH}$ are measured values.

**Figure 23. Load Circuit and Voltage Waveforms**

## PCI BUS PARAMETER MEASUREMENT INFORMATION

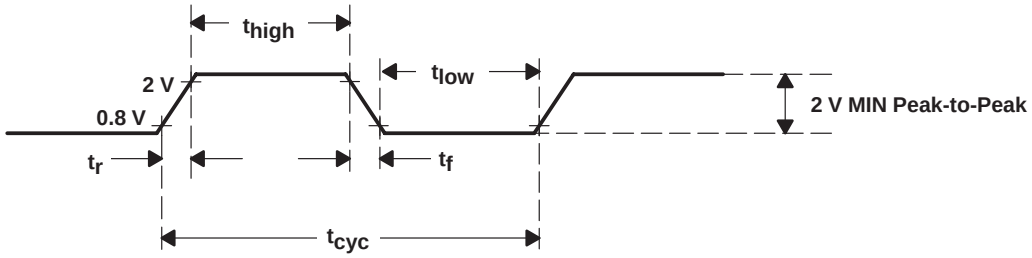



**Figure 24. PCLK Timing Waveform**

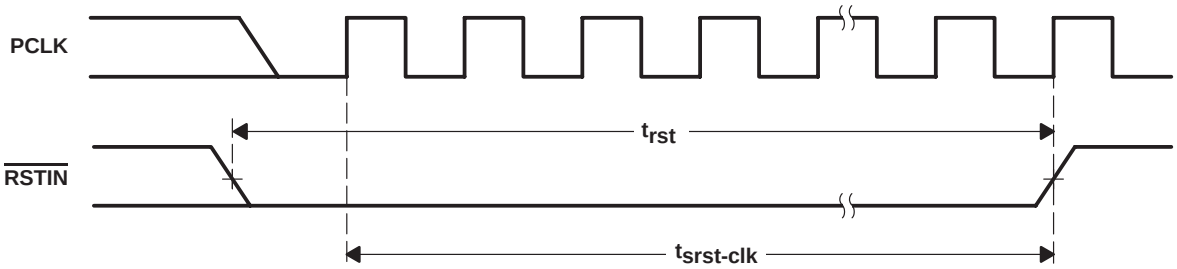



**Figure 25. $\overline{\text{RSTIN}}$ Timing Waveforms**

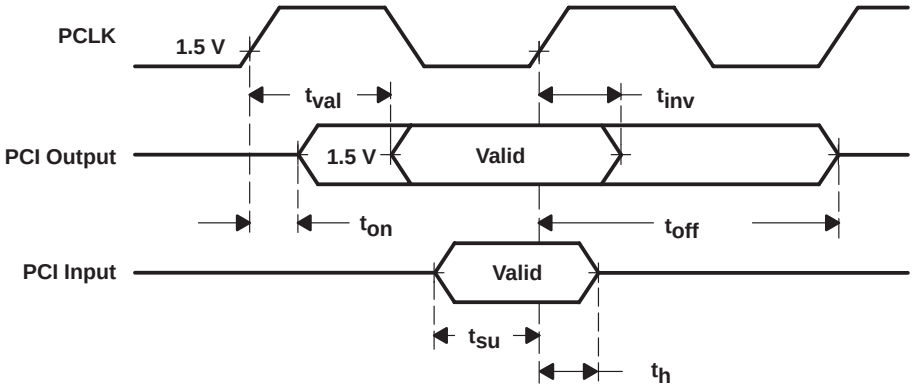



**Figure 26. Shared Signals Timing Waveforms**

## PC Card cycle timing

The PC Card cycle timing is controlled by the wait-state bits in the Intel 82365SL-DF compatible memory and I/O window registers. The PC Card cycle generator uses the PCI clock to generate the correct card address setup and hold times and the PC Card command active (low) interval. This allows the cycle generator to output PC Card cycles that are as close to the Intel 82365SL-DF timing as possible while always slightly exceeding the Intel 82365SL-DF values. This ensures compatibility with existing software and maximizes throughput.

The PC Card address setup and hold times are a function of the wait-state bits. Table 66 shows address setup time in PCLK cycles and nanoseconds for I/O and memory cycles. Table 67 and Table 68 show command active time in PCLK cycles and nanoseconds for I/O and memory cycles. Table 69 shows address hold time in PCLK cycles and nanoseconds for I/O and memory cycles.

**Table 66. PC Card Address Setup Time, $t_{su(A)}$, 8-Bit and 16-Bit PCI Cycles**

| WAIT-STATE BITS | | | TS1 – 0 = 01 (PCLK/ns) |
|---|---|---|---|
| I/O | | | 3/90 |
| Memory | WS1 | 0 | 2/60 |
| Memory | WS1 | 1 | 4/120 |

**Table 67. PC Card Command Active Cycle Time, $t_{C(A)}$, 8-Bit PCI Cycles**

| WAIT-STATE BITS | | | TS1 – 0 = 01 (PCLK/ns) |
|---|---|---|---|
| | WS | ZWS | |
| I/O | 0 | 0 | 19/570 |
| | 1 | X | 23/690 |
| | 0 | 1 | 7/210 |
| Memory | 00 | 0 | 19/570 |
| | 01 | X | 23/690 |
| | 10 | X | 23/690 |
| | 11 | X | 23/690 |
| | 00 | 1 | 7/210 |

**Table 68. PC Card Command Active Cycle Time, $t_{C(A)}$, 16-Bit PCI Cycles**

| WAIT-STATE BITS | | | TS1 – 0 = 01 (PCLK/ns) |
|---|---|---|---|
| | WS | ZWS | |
| I/O | 0 | 0 | 7/210 |
| | 1 | X | 11/330 |
| | 0 | 1 | N/A |
| Memory | 00 | 0 | 9/270 |
| | 01 | X | 13/390 |
| | 10 | X | 17/510 |
| | 11 | X | 23/630 |
| | 00 | 1 | 5/150 |

**Table 69. PC Card Address Hold Time, $t_{h(A)}$, 8-Bit and 16-Bit PCI Cycles**

| WAIT-STATE BITS | | | TS1 – 0 = 01 (PCLK/ns) |
|---|---|---|---|
| I/O | | | 2/60 |
| Memory | WS1 | 0 | 2/60 |
| Memory | WS1 | 1 | 3/90 |

**timing requirements over recommended ranges of supply voltage and operating free-air temperature, memory cycles (for 100-ns common memory)** (see Note 8 and Figure 27)

| | | ALTERNATE SYMBOL | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $t_{su}$ | Setup time, $\overline{CE1}$ and $\overline{CE2}$ before $\overline{WE}$/$\overline{OE}$ low | T1 | 60 | | ns |
| $t_{su}$ | Setup time, CA25–CA0 before $\overline{WE}$/$\overline{OE}$ low | T2 | $t_{su(A)}$+2PCLK | | ns |
| $t_{su}$ | Setup time, $\overline{REG}$ before $\overline{WE}$/$\overline{OE}$ low | T3 | 90 | | ns |
| $t_{pd}$ | Propagation delay time, $\overline{WE}$/$\overline{OE}$ low to $\overline{WAIT}$ low | T4 | | | ns |
| $t_w$ | Pulse duration, $\overline{WE}$/$\overline{OE}$ low | T5 | 200 | | ns |
| $t_h$ | Hold time, $\overline{WE}$/$\overline{OE}$ low after $\overline{WAIT}$ high | T6 | | | ns |
| $t_h$ | Hold time, $\overline{CE1}$ and $\overline{CE2}$ after $\overline{WE}$/$\overline{OE}$ high | T7 | 120 | | ns |
| $t_{su}$ | Setup time (read), CDATA15–CDATA0 valid before $\overline{OE}$ high | T8 | | | ns |
| $t_h$ | Hold time (read), CDATA15–CDATA0 valid after $\overline{OE}$ high | T9 | 0 | | ns |
| $t_h$ | Hold time, CA25–CA0 and $\overline{REG}$ after $\overline{WE}$/$\overline{OE}$ high | T10 | $t_{h(A)}$+1PCLK | | ns |
| $t_{su}$ | Setup time (write), CDATA15–CDATA0 valid before $\overline{WE}$ low | T11 | 60 | | ns |
| $t_h$ | Hold time (write), CDATA15–CDATA0 valid after $\overline{WE}$ low | T12 | 240 | | ns |

NOTE 8: These times are dependent on the register settings associated with ISA wait states and data size. They are also dependent on cycle type (read/write, memory/I/O) and $\overline{WAIT}$ from PC Card. The times listed here represent absolute minimums (the times that would be observed if programmed for zero wait state, 16-bit cycles) with a 33-MHz PCI clock.

**timing requirements over recommended ranges of supply voltage and operating free-air temperature, I/O cycles** (see Figure 28)

| | | ALTERNATE SYMBOL | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $t_{su}$ | Setup time, $\overline{REG}$ before $\overline{IORD}$/$\overline{IOWR}$ low | T13 | 60 | | ns |
| $t_{su}$ | Setup time, $\overline{CE1}$ and $\overline{CE2}$ before $\overline{IORD}$/$\overline{IOWR}$ low | T14 | 60 | | ns |
| $t_{su}$ | Setup time, CA25–CA0 valid before $\overline{IORD}$/$\overline{IOWR}$ low | T15 | $t_{su(A)}$+2PCLK | | ns |
| $t_{pd}$ | Propagation delay time, $\overline{IOIS16}$ low after CA25–CA0 valid | T16 | | 35 | ns |
| $t_{pd}$ | Propagation delay time, $\overline{IORD}$ low to $\overline{WAIT}$ low | T17 | 35 | | ns |
| $t_w$ | Pulse duration, $\overline{IORD}$/$\overline{IOWR}$ low | T18 | $T_{cA}$ | | ns |
| $t_h$ | Hold time, $\overline{IORD}$ low after $\overline{WAIT}$ high | T19 | | | ns |
| $t_h$ | Hold time, $\overline{REG}$ low after $\overline{IORD}$ high | T20 | 0 | | ns |
| $t_h$ | Hold time, $\overline{CE1}$ and $\overline{CE2}$ after $\overline{IORD}$/$\overline{IOWR}$ high | T21 | 120 | | ns |
| $t_h$ | Hold time, CA25–CA0 after $\overline{IORD}$/$\overline{IOWR}$ high | T22 | $t_{h(A)}$+1PCLK | | ns |
| $t_{su}$ | Setup time (read), CDATA15–CDATA0 valid before $\overline{IORD}$ high | T23 | 10 | | ns |
| $t_h$ | Hold time (read), CDATA15–CDATA0 valid after $\overline{IORD}$ high | T24 | 0 | | ns |
| $t_{su}$ | Setup time (write), CDATA15–CDATA0 valid before $\overline{IOWR}$ low | T25 | 90 | | ns |
| $t_h$ | Hold time (write), CDATA15–CDATA0 valid after $\overline{IOWR}$ high | T26 | 90 | | ns |

**switching characteristics over recommended ranges of supply voltage and operating free-air temperature, miscellaneous** (see Figure 29)

| PARAMETER | | | ALTERNATE SYMBOL | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $t_{pd}$ | Propagation delay time | BVD2 low to SPKROUT low | T27 | | 30 | ns |
| | | BVD2 high to SPKROUT high | | | 30 | |
| | | $\overline{\text{IREQ}}$ to IRQ15–IRQ3 | T28 | | 30 | |
| | | $\overline{\text{STSCHG}}$ to IRQ15–IRQ3 | | | 30 | |

## PC Card PARAMETER MEASUREMENT INFORMATION

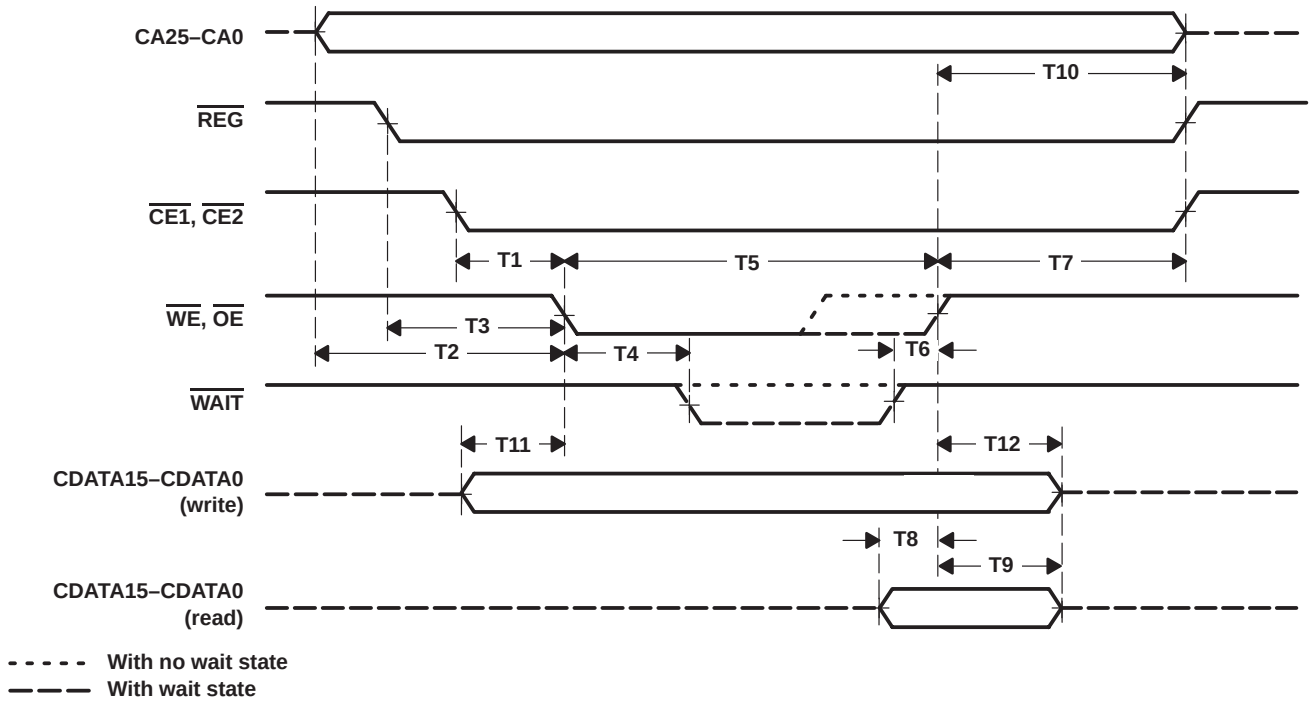



**Figure 27. PC Card Memory Cycle**

## PC Card PARAMETER MEASUREMENT INFORMATION

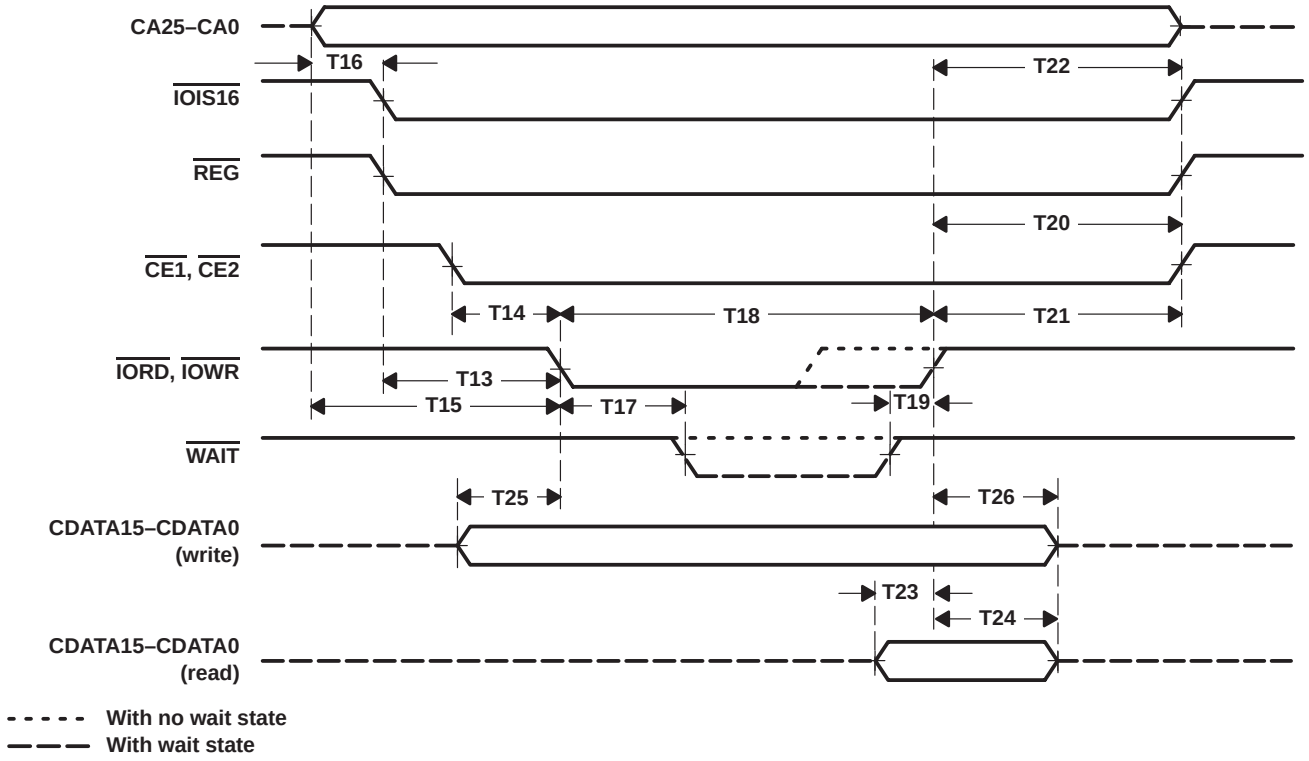



**Figure 28. PC Card I/O Cycle**

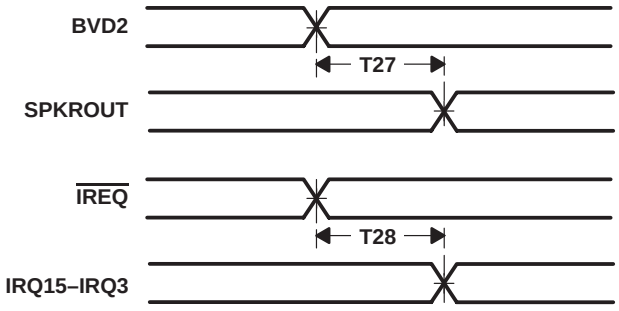
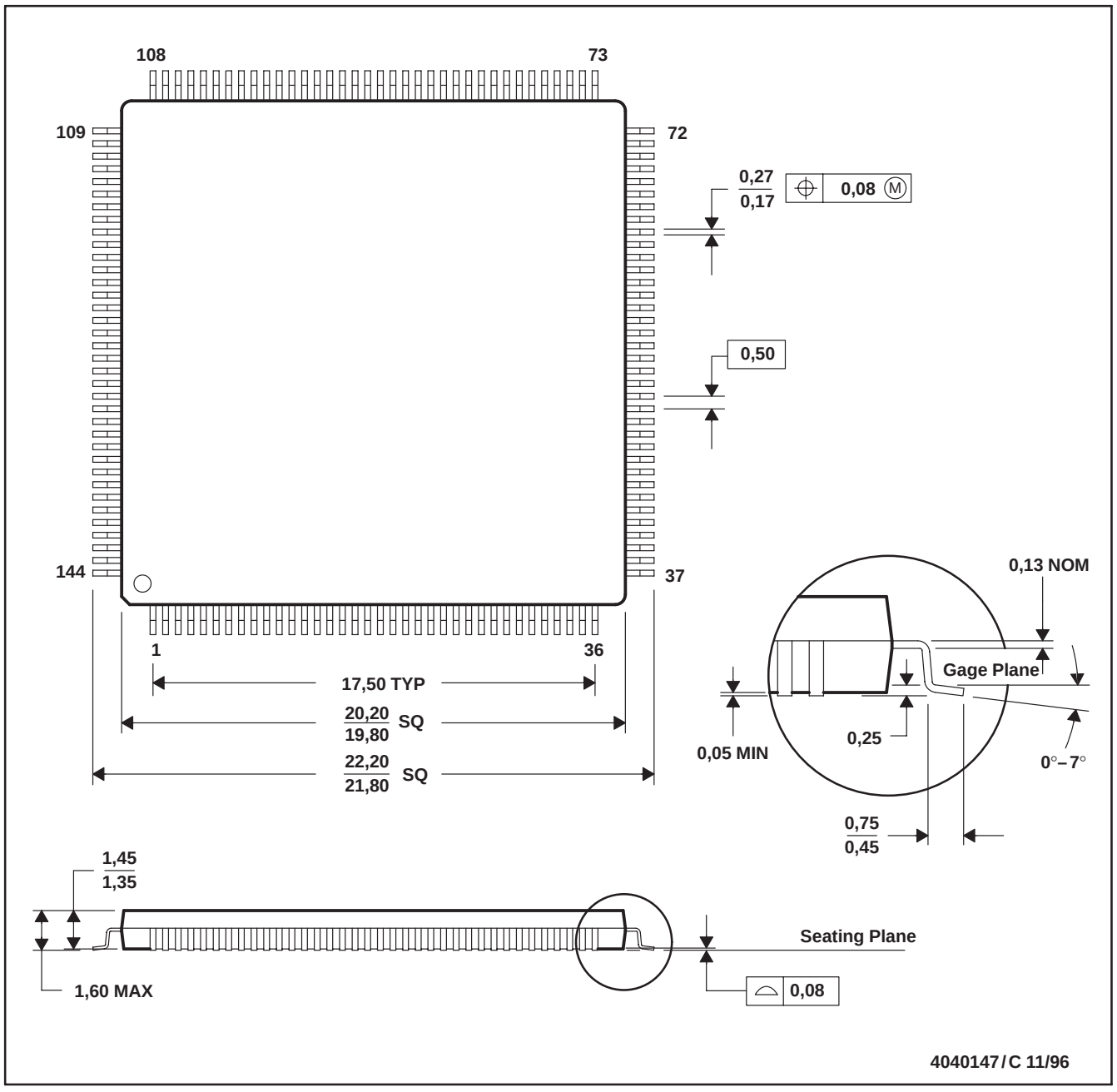



**Figure 29. Miscellaneous PC Card Delay Times**

## MECHANICAL DATA

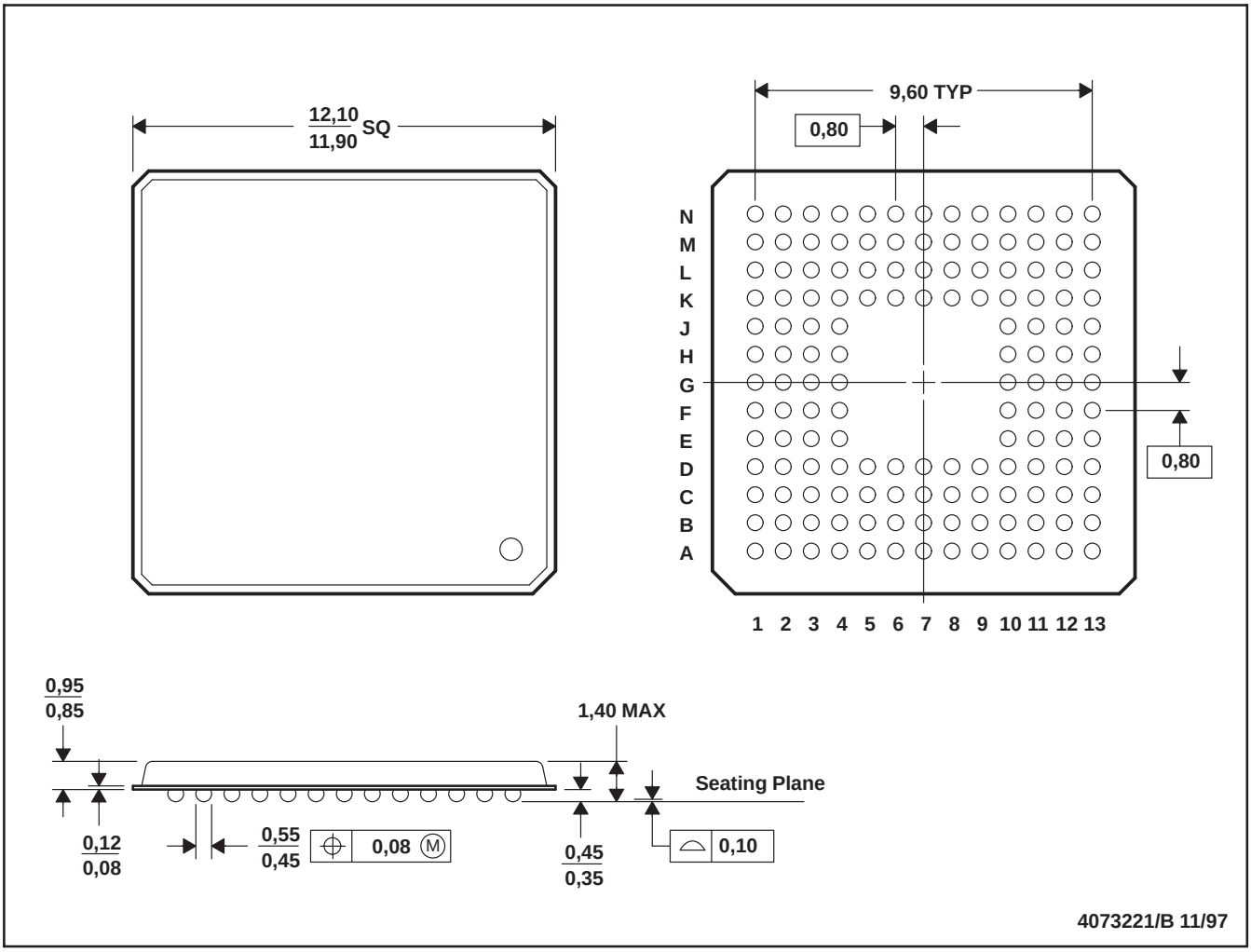
**PGE (S-PQFP-G144)** **PLASTIC QUAD FLATPACK**




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

## MECHANICAL DATA

**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. Micro Star™ BGA configuration

Micro Star is a trademark of Texas Instruments Incorporated.

## IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

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

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE ("CRITICAL APPLICATIONS"). TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF TI PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.

Copyright © 1998, Texas Instruments Incorporated