Texas Instruments

PCIbus Solutions

Integrated Solutions for the PCIbus

Data Book

1996

Advanced System Logic Products

PCIbus Solutions

Integrated Solutions for the PCIbus

printed on recycled paper

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software 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, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

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

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI 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.

Printed in U.S.A. by
Custom Printing Company
Owensville, Missouri

INTRODUCTION

The PCIbus solutions group develops connectivity and integration solutions for notebooks, desktops, and end equipment in emerging markets that require both design flexibility and superior performance.

A major highlight in this data book is the inclusion of the industry's first CardBus controller, the PCI1130. We also have included a product preview of the PCI10XX, an alternative solution that enables easy migration to CardBus, and a product preview of our first PCI-to-PCI bridge chip, the PCI20XX. Our book also features TI's first PCI-to-PCMCIA controller, the PCI1050, which employs earlier PC Card technologies. All four devices are based on a commonality of features requested by our customers.

Texas Instruments (TI) is active in many of the standards committees that are working to meet the computer industry's requirements for a total integrated solution. All of our PCIbus devices address the needs of existing and emerging technologies. Future products will continue to address dynamic requirements for design flexibility, power-management options, and integration of other available technologies.

More than ever, TI technologies are changing the way people throughout the world live, learn, and work. We believe you will find our first edition of the PCIbus data book resourceful and forward-looking. If you have any questions after reviewing this data book, please contact your local TI representative or call the Advanced System Logic hotline at (903) 868-5202.

PRODUCT STAGE STATEMENTS

Product stage statements are used on Texas Instruments data sheets to indicate the development stage(s) of the product(s) specified in the data sheets.

If all products specified in a data sheet are at the same development stage, the appropriate statement from the following list is placed in the lower left corner of the first page of the data sheet.

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.

ADVANCE INFORMATION concerns new products in the sampling or preproduction phase of development. Characteristic data and other specifications are subject to change without notice.

PRODUCT PREVIEW information concerns products in the formative or design phase of development. Characteristic data and other specifications are design goals. Texas Instruments reserves the right to change or discontinue these products without notice.

If not all products specified in a data sheet are at the PRODUCTION DATA stage, then the first statement below is placed in the lower left corner of the first page of the data sheet. Subsequent pages of the data sheet containing PRODUCT PREVIEW information or ADVANCE INFORMATION are then marked in the lower left-hand corner with the appropriate statement given below:

UNLESS OTHERWISE NOTED this document contains PRODUCTION DATA information 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.

ADVANCE INFORMATION concerns new products in the sampling or preproduction phase of development. Characteristic data and other specifications are subject to change without notice.

PRODUCT PREVIEW information concerns products in the formative or design phase of development. Characteristic data and other specifications are design goals. Texas Instruments reserves the right to change or discontinue these products without notice.

Contents

INTRODUCTION

These symbols, terms, and definitions are in accordance with those currently agreed upon by the JEDEC Council of the Electronic Industries Association (EIA) for use in the USA and by the International Electrotechnical Commission (IEC) for international use.

operating conditions and characteristics (in sequence by letter symbols)

C_i **Input capacitance**

The internal capacitance at an input of the device

C_{io} **Input/output capacitance**

Input-to-output internal capacitance; transcapacitance

C_o **Output capacitance**

The internal capacitance at an output of the device

C_{pd} **Power dissipation capacitance**

Used to determine the no-load dynamic power dissipation per logic function (see individual circuit pages): $P_D = C_{pd} V_{CC}^2 f + I_{CC} V_{CC}$

ΔI_{CC} **Supply current change**

The increase in supply current for each input that is at one of the specified TTL voltage levels rather than 0 V or V_{CC}

I_{IH} **High-level input current**

The current into* an input when a high-level voltage is applied to that input

I_{IL} **Low-level input current**

The current into* an input when a low-level voltage is applied to that input

I_{OH} **High-level output current**

The current into* an output with input conditions applied that, according to the product specification, establishes a high level at the output

I_{OL} **Low-level output current**

The current into* an output with input conditions applied that, according to the product specification, establishes a low level at the output

t_a **Access time**

The time interval between the application of a specified input pulse and the availability of valid signals at an output

t_c **Clock cycle time**

Clock cycle time is $1/f_{max}$.

t_{dis} **Disable time (of a 3-state or open-collector output)**

The propagation time between the specified reference points on the input and output voltage waveforms with the output changing from either of the defined active levels (high or low) to a high-impedance (off) state

NOTE: For 3-state outputs, $t_{dis} = t_{PHZ}$ or t_{PLZ}. Open-collector outputs change only if they are low at the time of disabling, so $t_{dis} = t_{PLH}$.

*Current out of a terminal is given as a negative value.

t_{en} **Enable time (of a 3-state or open-collector output)**

The propagation time between the specified reference points on the input and output voltage waveforms with the output changing from a high-impedance (off) state to either of the defined active levels (high or low)

NOTE: For 3-state outputs, $t_{en} = t_{PZH}$ or t_{PZL}. Open-collector outputs change only if they are responding to data that would cause the output to go low, so $t_{en} = t_{PHL}$.

t_h **Hold time**

The time interval during which a signal is retained at a specified input terminal after an active transition occurs at another specified input terminal

NOTES: 1. The hold time is the actual time interval between two signal events and is determined by the system in which the digital circuit operates. A minimum value is specified that is the shortest interval for which correct operation of the digital circuit is to be expected.

2. The hold time may have a negative value in which case the minimum limit defines the longest interval (between the release of the signal and the active transition) for which correct operation of the digital circuit is to be expected.

t_{pd} **Propagation delay time**

The time between the specified reference points on the input and output voltage waveforms with the output changing from one defined level (high or low) to the other defined level ($t_{pd} = t_{PHL}$ or t_{PLH})

t_{PHL} **Propagation delay time, high-to-low level output**

The time between the specified reference points on the input and output voltage waveforms with the output changing from the defined high level to the defined low level

t_{PHZ} **Disable time (of a 3-state output) from high level**

The time interval between the specified reference points on the input and the output voltage waveforms with the 3-state output changing from the defined high level to the high-impedance (off) state

t_{PLH} **Propagation delay time, low-to-high level output**

The time between the specified reference points on the input and output voltage waveforms with the output changing from the defined low level to the defined high level

t_{PLZ} **Disable time (of a 3-state output) from low level**

The time interval between the specified reference points on the input and the output voltage waveforms with the 3-state output changing from the defined low level to the high-impedance (off) state

t_{PZH} **Enable time (of a 3-state output) to high level**

The time interval between the specified reference points on the input and output voltage waveforms with the 3-state output changing from the high-impedance (off) state to the defined high leve

t_{PZL} **Enable time (of a 3-state output) to low level**

The time interval between the specified reference points on the input and output voltage waveforms with the 3-state output changing from the high-impedance (off) state to the defined low level

$t_{sk(o)}$ **Output skew**

The difference between any two propagation delay times when a single switching input or multiple inputs switching simultaneously cause multiple outputs to switch, as observed across all switching output. This parameter is used to describe the fanout capability of a clock driver and is of concern when making decisions on clock buffering and distribution networks.

t_{su} **Setup time**

The time interval between the application of a signal at a specified input terminal and a subsequent active transition at another specified input terminal

NOTES: 1. The setup time is the actual time interval between two signal events and is determined by the system in which the digital circuit operates. A minimum value is specified that is the shortest interval for which correct operation of the digital circuit is to be expected.

2. The setup time may have a negative value, in which case the minimum limit defines the longest interval (between the active transition and the application of the other signal) for which correct operation of the digital circuit is to be expected.

t_t **Transition time**

The time interval between tow specified levels, one near the beginning and one near the end of the same pulse edge

t_w **Pulse duration (width)**

The time interval between specified reference points on the leading and trailing edges of the pulse waveform

V_{IH} **High-level input voltage**

An input voltage within the more positive (less negative) of the two ranges of values used to represent the binary variables

NOTE: A minimum is specified that is the least-positive value of high-level input voltage for which operation of the logic element within specification limits is to be expected.

V_{IL} **Low-level input voltage**

An input voltage within the less positive (more negative) of the two ranges of values used to represent the binary variables

NOTE: A maximum is specified that is the most-positive value of low-level input voltage for which operation of the logic element within specification limits is to be expected.

V_{OH} **High-level output voltage**

The voltage at an output terminal with input conditions applied that, according to product specification, establishes a high level at the output

V_{OL} **Low-level output voltage**

The voltage at an output terminal with input conditions applied that, according to product specification, establishes a low level at the output

EXPLANATION OF FUNCTION TABLES

The following symbols are used in function tables on TI data sheets:

H	=	high level (steady state)
L	=	low level (steady state)
↑	=	transition from low to high level
↓	=	transition from high to low level
→	=	value/level or resulting value/level is routed to indicated destination
↶	=	value/level is re-entered
X	=	irrelevant (any input, including transitions)
Z	=	off (high-impedance) state of a 3-state output
a . . . h	=	the level of steady-state inputs A through H respectively
Q_0	=	level of Q before the indicated steady-state input conditions were established
$\overline{Q}_0$	=	complement of Q_0 or level of $\overline{Q}$ before the indicated steady-state input conditions were established
Q_n	=	level of Q before the most recent active transition indicated by ↓ or ↑
⎍	=	one high-level pulse
⎏	=	one low-level pulse
Toggle	=	each output changes to the complement of its previous level on each active transition indicated by ↓ or ↑

If, in the input columns, a row contains only the symbols H, L, and/or X, this means the indicated output is valid whenever the input configuration is achieved and regardless of the sequence in which it is achieved. The output persists so long as the input configuration is maintained.

If, in the input columns, a row contains H, L, and/or X together with ↑ and/or ↓, this means the output is valid whenever the input configuration is achieved but the transition(s) must occur following the achievement of the steady-state levels. If the output is shown as a level (H, L, Q_0, or $\overline{Q}_0$), it persists so long as the steady-state input levels and the levels that terminate indicated transitions are maintained. Unless otherwise indicated, input transitions in the opposite direction to those shown have no effect at the output. (If the output is shown as a pulse, ⎍ or ⎏, the pulse follows the indicated input transition and persists for an interval dependent on the circuit.)

Among the most complex function tables are those of the shift registers. These embody most of the symbols used in any of the function tables, plus more. Below is the function table of a 4-bit bidirectional universal shift register, e.g., type SN74194.

FUNCTION TABLE

INPUTS										OUTPUTS			
CLEAR	MODE		CLOCK	SERIAL		PARALLEL				Q_A	Q_B	Q_C	Q_D
	S1	S0		LEFT	RIGHT	A	B	C	D				
L	X	X	X	X	X	X	X	X	X	L	L	L	L
H	X	X	L	X	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
H	H	H	↑	X	X	a	b	c	d	a	b	c	d
H	L	H	↑	X	H	H	H	H	H	H	Q_{An}	Q_{Bn}	Q_{Cn}
H	L	H	↑	X	L	L	L	L	L	L	Q_{An}	Q_{Bn}	Q_{Cn}
H	H	L	↑	H	X	X	X	X	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	H
H	H	L	↑	L	X	X	X	X	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	L
H	L	L	X	X	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}

The first line of the table represents a synchronous clearing of the register and says that if clear is low, all four outputs will be reset low regardless of the other inputs. In the following lines, clear is inactive (high) and so has no effect.

The second line shows that so long as the clock input remains low (while clear is high), no other input has any effect and the outputs maintain the levels they assumed before the steady-state combination of clear high and clock low was established. Since on other lines of the table only the rising transition of the clock is shown to be active, the second line implicitly shows that no further change in the outputs will occur while the clock remains high or on the high-to-low transition of the clock.

The third line of the table represents synchronous parallel loading of the register and says that if S1 and S0 are both high then, without regard to the serial input, the data entered at A will be at output Q_A, data entered at B will be at Q_B, and so forth, following a low-to-high clock transition.

The fourth and fifth lines represent the loading of high- and low-level data, respectively, from the shift-right serial input and the shifting of previously entered data one bit; data previously at Q_A is now at Q_B, the previous levels of Q_B and Q_C are now at Q_C and Q_D, respectively, and the data previously at Q_D is no longer in the register. This entry of serial data and shift takes place on the low-to-high transition of the clock when S1 is low and S0 is high and the levels at inputs A through D have no effect.

The sixth and seventh lines represent the loading of high- and low-level data, respectively, from the shift-left serial input and the shifting of previously entered data one bit; data previously at Q_B is now at Q_A, the previous levels of Q_C and Q_D are now at Q_B and Q_C, respectively, and the data previously at Q_A is no longer in the register. This entry of serial data and shift takes place on the low-to-high transition of the clock when S1 is high and S0 is low and the levels at inputs A through D have no effect.

The last line shows that as long as both inputs are low, no other input has any effect and, as in the second line, the outputs maintain the levels they assumed before the steady-state combination of clear high and both mode inputs low was established.

The function table functional tests do not reflect all possible combinations or sequential modes.

Current Texas Instruments Advanced Logic Publications

Listed below is the current collection of Texas Instruments logic technical documentation. The alphanumeric code is the literature number for each item listed. These documents can be ordered through a TI representative or authorized distributor by referencing the literature number.

Document	*Literature Number*
Advanced BiCMOS Technology Data Book (1994)	SCBD002B
Advanced Bus-Interface SPICE I/O Models Data Book (1995)	SCBD004A
Advanced CMOS Logic Data Book (1993)	SCAD001C
Advanced Logic and Bus-Interface Logic Data Book (1991)	SCYD001
ALS/AS Logic Data Book (1995)	SDAD001C
BCT BiCMOS Bus-Interface Logic Data Book (1994)	SCBD001B
Boundary-Scan Logic, IEEE Std. 1149.1 (JTAG) 5-V and 3.3-V Bus-Interface and Scan-Support Products Data Book (1994)	SCTD002
CBT Bus Switches Crossbar Technology Data Book (1995)	SCDD001
CDC Clock-Distribution Circuits Data Book (1994)	SCAD004
F Logic Data Book (1994)	SDFD001B
High-Performance FIFO Memories Data Book (1996)	SCAD003C
High-Performance FIFO Memories Designer's Handbook (1996)	SCAA012A
High-Speed CMOS Logic Data Book (1989)	SCLD001C
Low-Voltage Logic Data Book (1996)	SCBD003B
TTL Logic Data Book (TTL, LS, S) (1988)	SDLD001A
Semiconductor Group Package Outlines Reference Guide	SSYU001A

In addition to the books listed above, the following documents are available only in Europe.

Document	*Literature Number*
Advanced BiCMOS Technology Data Book (1994)	SCBDE08
CBT Crossbar Technology Data Book (1995)	SCDDE01
CDC Data and Applications Manual (1995)	SCBTE07B
Packaging Data Book (1995)	SCYDE04
ASL SCOPE™ Products Data and Applications Manual (1994)	SCBDE09

PCI-TO-PC CARD CONTROLLERS

Signaling Environment

FEATURES	PCI1050 PCI-TO-PC CARD CONTROLLER UNIT	PCI10XX PCI-TO-PC CARD16 CONTROLLER UNIT	PCI1130 PCI-TO-CARDBUS CONTROLLER UNIT
Core logic environment	5 V	3.3 V	3.3 V
PCI signaling environment	3.3 V/5 V	3.3 V/5 V	3.3 V/5 V

Standards Compliance/Compatibility

FEATURES	PCI1050 PCI-TO-PC CARD CONTROLLER UNIT	PCI10XX PCI-TO-PC CARD16 CONTROLLER UNIT	PCI1130 PCI-TO-CARDBUS CONTROLLER UNIT
PCI specification compliance	PCI specification 2.0	PCI specification 2.1	PCI specification 2.1
PC Card™ standard compliance	PCMCIA 2.1/JEIDA 4.1	PC Card standard, February 1995	PC Card standard, February 1995
Intel™ 82365SL-DF register compatible	Yes	Yes	Yes
ExCA™ register compatible	Yes	Yes	Yes

Slots Support

FEATURES	PCI1050 PCI-TO-PC CARD CONTROLLER UNIT	PCI10XX PCI-TO-PC CARD16 CONTROLLER UNIT	PCI1130 PCI-TO-CARDBUS CONTROLLER UNIT
Number of slots supported	2	2	2
Slot voltages supported	3.3 V/5 V	3.3 V/5 V	3.3-V/5-V PC Card16 cards and 3.3-V CardBus cards
Hot insertion/removal	Yes	Yes	Yes

Interrupts

FEATURES	PCI1050 PCI-TO-PC CARD CONTROLLER UNIT	PCI10XX PCI-TO-PC CARD16 CONTROLLER UNIT	PCI1130 PCI-TO-CARDBUS CONTROLLER UNIT
PCI style	Yes	Yes	Yes
ISA serial	Yes	Yes	Yes
ISA parallel	Yes	Yes	Yes

DMA

FEATURES	PCI1050 PCI-TO-PC CARD CONTROLLER UNIT	PCI10XX PCI-TO-PC CARD16 CONTROLLER UNIT	PCI1130 PCI-TO-CARDBUS CONTROLLER UNIT
PCIWay DMA	No	Yes	Yes
PC/PCI DMA	No	Yes	Yes

PCI-TO-PC CARD CONTROLLERS

Other Features

FEATURES	PCI1050 PCI-TO-PC CARD CONTROLLER UNIT	PCI10XX PCI-TO-PC CARD16 CONTROLLER UNIT	PCI1130 PCI-TO-CARDBUS CONTROLLER UNIT
Burst transfer support	No	Yes	Yes
PCI clock run support	No	Yes	Yes
Programmable windows for PC Card16 sockets	Five memory and two I/O	Five memory and two I/O	Five memory and two I/O
Programmable windows for CardBus sockets	Two memory and two I/O	Two memory and two I/O	Two memory and two I/O
Zoom video support	No	Yes	Yes
Package type	208-pin PQFP	208-pin TQFP	208-pin TQFP

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

SCPS001F – OCTOBER 1994 – REVISED MARCH 1996

- **Supports Two PC Card™ Slots With Hot Insertion or Removal**
- **Supports Any Combination of 3.3-V and 5-V PC Cards™**
- **PCMCIA 2.1/JEIDA 4.1/ExCA™ Standard Compliant**
- **Intel™ 82365SL-DF Register Compatible**
- **Programmable PCI/ISA Interrupt Routing**
- **Programmable V_{PP} and V_{CC} Control for Each Slot**
- **Supports AT Attachment (ATA) Interface**
- **Device Selection for Cascading Done Through PCI Configuration**
- **Exchangeable Card (ExCA™)-Compatible Registers Mapped In the PCI Configuration Space and I/O Space**
- **Texas Instruments (TI) Extension Registers Mapped in the PCI Configuration Space**
- **Four-Deep, 32-Bit Write Buffers**
- **Low-Power Advanced Submicron CMOS Technology**
- **Packaged in 208-Pin Plastic Quad Flatpack (PPM)**

description

The TI PCI1050 is a high-performance card controller that interfaces two PC Card slots to the peripheral component interconnect (PCI) bus. The core logic and PCI interface are powered at 5 V while the card interfaces can be powered at the card V_{CC} to support any combination of 3.3-V and 5-V PC Cards.

All card signals are individually buffered to allow hot insertion and removal without external buffering. The PCI1050 is register compatible with the Intel 82365SL-DF ExCA controller and can be cascaded to support up to eight PC Card slots. The PCI1050 internal datapath logic allows the host to access 8-bit and 16-bit cards using full 32-bit PCI cycles for maximum performance. Independent four-deep by 32-bit write buffers allow fast posted writes to improve system bus utilization.

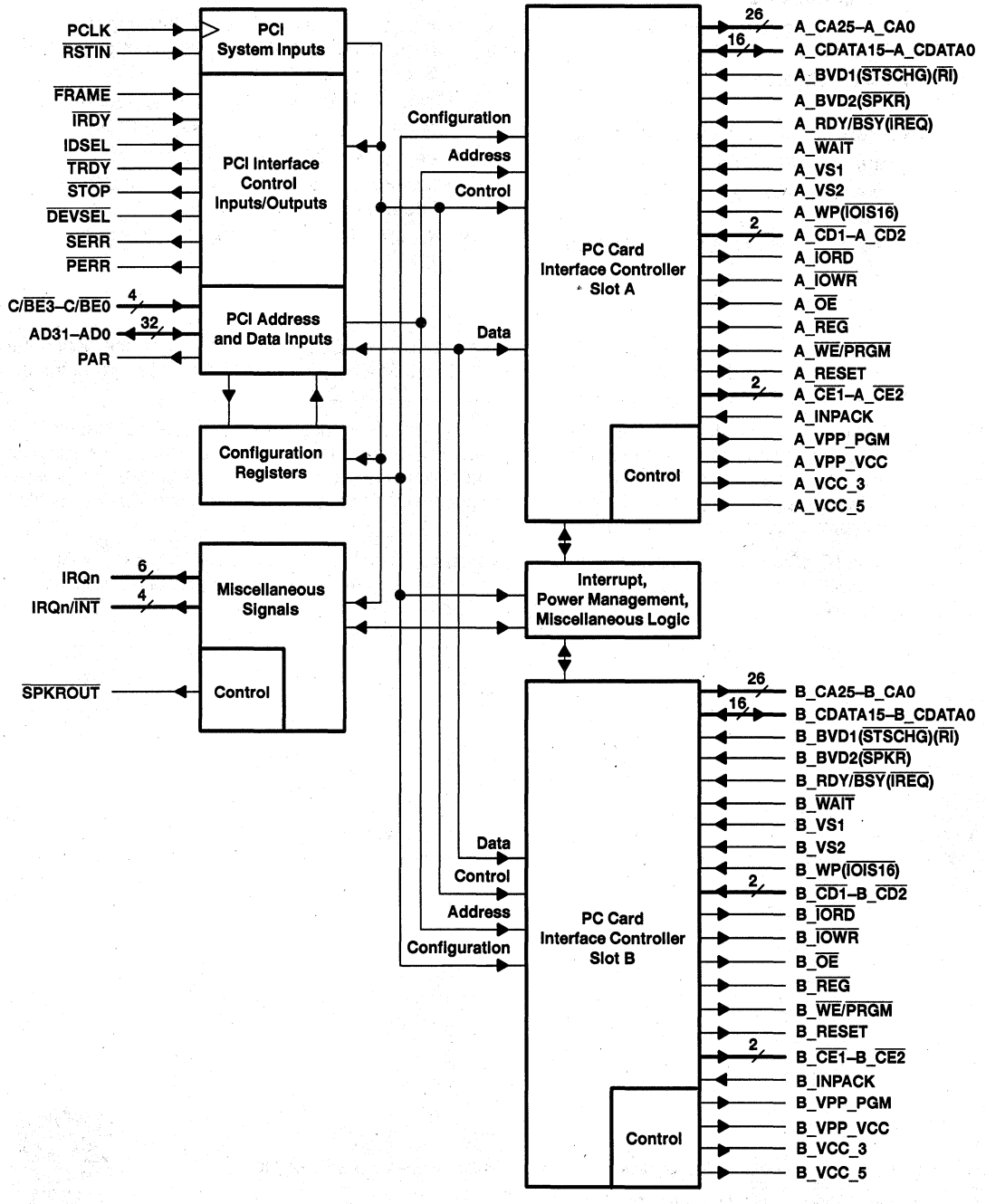
An advanced CMOS process is utilized to achieve low system-power consumption while operating at PCI clock rates up to 33 MHz. A power-down mode allows host software to reduce power consumption further while preserving internal register contents and allowing PC Cards to interrupt the host.

Table of Contents

ExCA and PC Card are trademarks of Personal Computer Memory Card International Association (PCMCIA).
Intel is a trademark of Intel Corp.
Lotus is a registered trademark of Lotus Development Corporation.
Microsoft is a registered trademark of Microsoft Corporation.

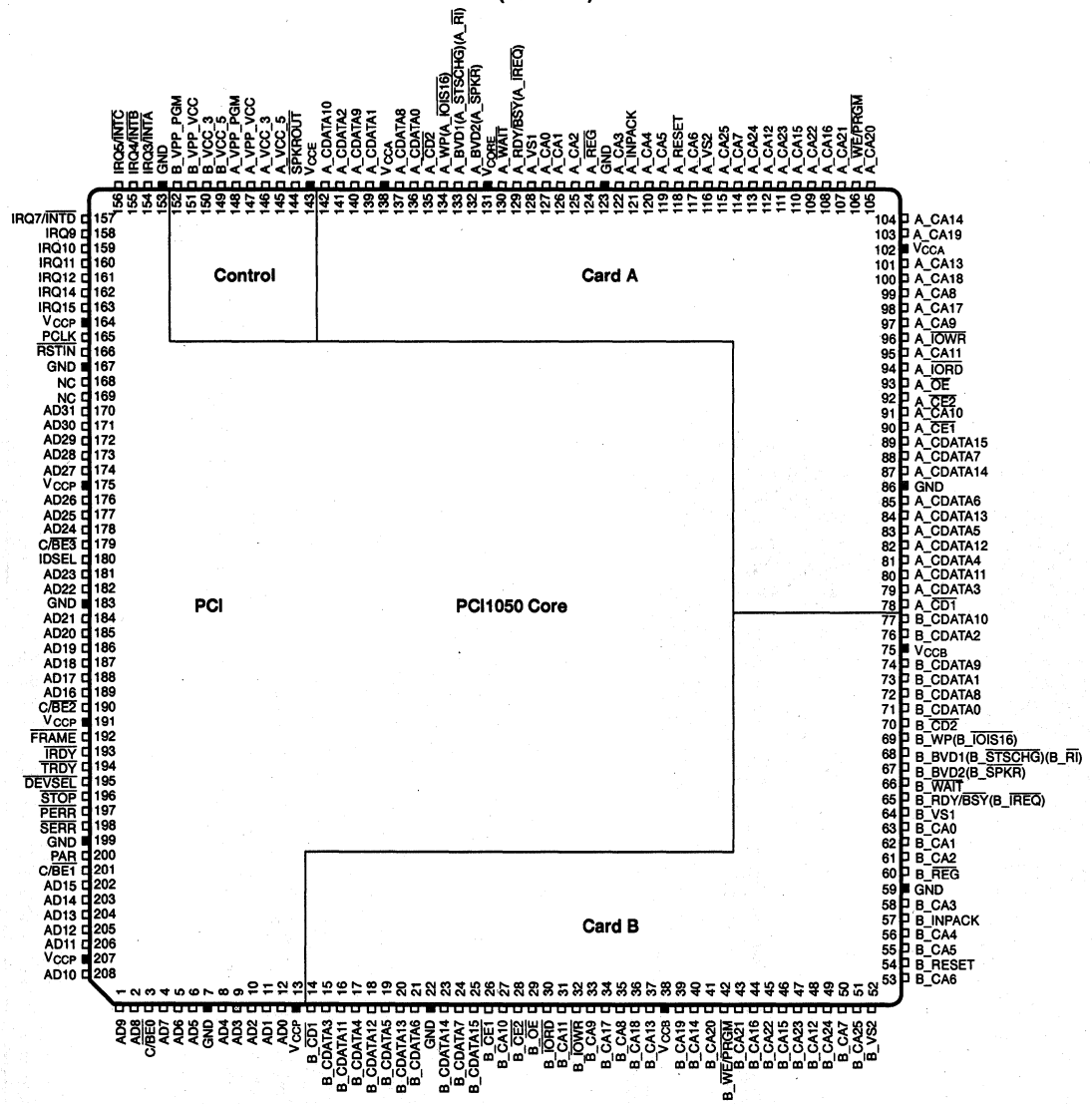
TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

functional block diagram



terminal assignments

PPM PACKAGE
(TOP VIEW)



NC – No internal connection

Table 1. Signals Sorted by Signal Name

SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.
A_BVD1(A_STSCHG)(A_RI)	133	A_IOWR	96	B_CDATA10	77	AD28	173
A_BVD2(A_SPKR)	132	A_OE	93	B_CDATA11	16	AD29	172
A_CA0	127	A_RDY/BSY(A_IREQ)	129	B_CDATA12	18	AD30	171
A_CA1	126	A_REG	124	B_CDATA13	20	AD31	170
A_CA2	125	A_RESET	118	B_CDATA14	23	C/BE0	3
A_CA3	122	A_VCC_3	146	B_CDATA15	25	C/BE1	201
A_CA4	120	A_VCC_5	145	B_CE1	26	C/BE2	190
A_CA5	119	A_VPP_PGM	148	B_CE2	28	C/BE3	179
A_CA6	117	A_VPP_VCC	147	B_VS1	64	DEVSEL	195
A_CA7	114	A_WAIT	130	B_VS2	52	FRAME	192
A_CA8	99	A_WE/PRGM	106	B_INPACK	57	GND	7
A_CA9	97	A_WP(A_IOSI16)	134	B_IORD	30	GND	22
A_CA10	91	B_BVD1(B_STSCHG)(B_RI)	68	B_IOWR	32	GND	59
A_CA11	95	B_BVD2(B_SPKR)	67	B_OE	29	GND	86
A_CA12	112	B_CA0	63	B_RDY/BSY(B_IREQ)	65	GND	123
A_CA13	101	B_CA1	62	B_REG	60	GND	153
A_CA14	104	B_CA2	61	B_RESET	54	GND	167
A_CA15	110	B_CA3	58	B_VCC_3	150	GND	183
A_CA16	108	B_CA4	56	B_VCC_5	149	GND	199
A_CA17	98	B_CA5	55	B_VPP_PGM	152	IDSEL	180
A_CA18	100	B_CA6	53	B_VPP_VCC	151	IRDY	193
A_CA19	103	B_CA7	50	B_WAIT	66	IRQ3/INTA	154
A_CA20	105	B_CA8	35	B_WE/PRGM	42	IRQ4/INTB	155
A_CA21	107	B_CA9	33	B_WP(B_IOIS16)	69	IRQ5/INTC	156
A_CA22	109	B_CA10	27	AD0	12	IRQ7/INTD	157
A_CA23	111	B_CA11	31	AD1	11	IRQ9	158
A_CA24	113	B_CA12	48	AD2	10	IRQ10	159
A_CA25	115	B_CA13	37	AD3	9	IRQ11	160
A_CD1	78	B_CA14	40	AD4	8	IRQ12	161
A_CD2	135	B_CA15	46	AD5	6	IRQ14	162
A_CDATA0	136	B_CA16	44	AD6	5	IRQ15	163
A_CDATA1	139	B_CA17	34	AD7	4	NC	168
A_CDATA2	141	B_CA18	36	AD8	2	NC	169
A_CDATA3	79	B_CA19	39	AD9	1	PAR	200
A_CDATA4	81	B_CA20	41	AD10	208	PCLK	165
A_CDATA5	83	B_CA21	43	AD11	206	RSTIN	166
A_CDATA6	85	B_CA22	45	AD12	205	PERR	197
A_CDATA7	88	B_CA23	47	AD13	204	SPKROUT	144
A_CDATA8	137	B_CA24	49	AD14	203	STOP	196
A_CDATA9	140	B_CA25	51	AD15	202	SERR	198
A_CDATA10	142	B_CD1	14	AD16	189	TRDY	194
A_CDATA11	80	B_CD2	70	AD17	188	V_{CCA}	102
A_CDATA12	82	B_CDATA0	71	AD18	187	V_{CCA}	138
A_CDATA13	84	B_CDATA1	73	AD19	186	V_{CCB}	35
A_CDATA14	87	B_CDATA2	76	AD20	185	V_{CCB}	75
A_CDATA15	89	B_CDATA3	15	AD21	184	V_{CCE}	143
A_CE1	90	B_CDATA4	17	AD22	182	V_{CCP}	13
A_CE2	92	B_CDATA5	19	AD23	181	V_{CCP}	164
A_VS1	128	B_CDATA6	21	AD24	178	V_{CCP}	175
A_VS2	116	B_CDATA7	24	AD25	177	V_{CCP}	191
A_INPACK	121	B_CDATA8	72	AD26	176	V_{CCP}	207
A_IORD	94	B_CDATA9	74	AD27	174	V_{CORE}	131

Table 2. Signals Sorted by Pin Number

NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME
1	AD9	53	B_CA6	105	A_CA20	157	IRQ7/INTD
2	AD8	54	B_RESET	106	A_WE/PRGM	158	IRQ9
3	C/BE0	55	B_CA5	107	A_CA21	159	IRQ10
4	AD7	56	B_CA4	108	A_CA16	160	IRQ11
5	AD6	57	B_INPACK	109	A_CA22	161	IRQ12
6	AD5	58	B_CA3	110	A_CA15	162	IRQ14
7	GND	59	GND	111	A_CA23	163	IRQ15
8	AD4	60	B_REG	112	A_CA12	164	V_{CCP}
9	AD3	61	B_CA2	113	A_CA24	165	PCLK
10	AD2	62	B_CA1	114	A_CA7	166	RSTIN
11	AD1	63	B_CA0	115	A_CA25	167	GND
12	AD0	64	B_VS1	116	A_VS2	168	NC
13	V_{CCP}	65	B_RDY/BSY(B_IREQ)	117	A_CA6	169	NC
14	B_CD1	66	B_WAIT	118	A_RESET	170	AD31
15	B_CDATA3	67	B_BVD2(B_SPKR)	119	A_CA5	171	AD30
16	B_CDATA11	68	B_BVD1(B_STSCHG)(B_RI)	120	A_CA4	172	AD29
17	B_CDATA4	69	B_WP(B_IOIS16)	121	A_INPACK	173	AD28
18	B_CDATA12	70	B_CD2	122	A_CA3	174	AD27
19	B_CDATA5	71	B_CDATA0	123	GND	175	V_{CCP}
20	B_CDATA13	72	B_CDATA8	124	A_REG	176	AD26
21	B_CDATA6	73	B_CDATA1	125	A_CA2	177	AD25
22	GND	74	B_CDATA9	126	A_CA1	178	AD24
23	B_CDATA14	75	V_{CCB}	127	A_CA0	179	C/BE3
24	B_CDATA7	76	B_CDATA2	128	A_VS1	180	IDSEL
25	B_CDATA15	77	B_CDATA10	129	A_RDY/BSY(A_IREQ)	181	AD23
26	B_CE1	78	A_CD1	130	A_WAIT	182	AD22
27	B_CA10	79	A_CDATA3	131	V_{CORE}	183	GND
28	B_CE2	80	A_CDATA11	132	A_BVD2(A_SPKR)	184	AD21
29	B_OE	81	A_CDATA4	133	A_BVD1(A_STSCHG)(A_RI)	185	AD20
30	B_IORD	82	A_CDATA12	134	A_WP(A_IOIS16)	186	AD19
31	B_CA11	83	A_CDATA5	135	A_CD2	187	AD18
32	B_IOWR	84	A_CDATA13	136	A_CDATA0	188	AD17
33	B_CA9	85	A_CDATA6	137	A_CDATA8	189	AD16
34	B_CA17	86	GND	138	V_{CCA}	190	C/BE2
35	B_CA8	87	A_CDATA14	139	A_CDATA1	191	V_{CCP}
36	B_CA18	88	A_CDATA7	140	A_CDATA9	192	FRAME
37	B_CA13	89	A_CDATA15	141	A_CDATA2	193	IRDY
38	V_{CCB}	90	A_CE1	142	A_CDATA10	194	TRDY
39	B_CA19	91	A_CA10	143	V_{CCE}	195	DEVSEL
40	B_CA14	92	A_CE2	144	SPKROUT	196	STOP
41	B_CA20	93	A_OE	145	A_VCC_5	197	PERR
42	B_WE/PRGM	94	A_IORD	146	A_VCC_3	198	SERR
43	B_CA21	95	A_CA11	147	A_VPP_VCC	199	GND
44	B_CA16	96	A_IOWR	148	A_VPP_PGM	200	PAR
45	B_CA22	97	A_CA9	149	B_VCC_5	201	C/BE1
46	B_CA15	98	A_CA17	150	B_VCC_3	202	AD15
47	B_CA23	99	A_CA8	151	B_VPP_VCC	203	AD14
48	B_CA12	100	A_CA18	152	B_VPP_PGM	204	AD13
49	B_CA24	101	A_CA13	153	GND	205	AD12
50	B_CA7	102	V_{CCA}	154	IRQ3/INTA	206	AD11
51	B_CA25	103	A_CA19	155	IRQ4/INTB	207	V_{CCP}
52	B_VS2	104	A_CA14	156	IRQ5/INTC	208	AD10

Terminal Functions

PCI system

TERMINAL NAME	TERMINAL NO.	I/O TYPE	BUFFER TYPE	FUNCTION
PCLK	165	I	CMOS	Bus clock. PCLK provides timing for all transactions on the PCI bus.
$\overline{\text{RSTIN}}$	166	I	CMOS	Reset. $\overline{\text{RSTIN}}$ forces the PCI1050 to a known state.

PCI address and data

TERMINAL NAME	TERMINAL NO.	I/O TYPE	BUFFER TYPE	FUNCTION
AD31	170	I/O	CMOS/12 mA	Address/data bus. During the address phase of a PCI cycle, AD31–AD0 contain a 32-bit address. During the data phase, AD31–AD0 contain data.
AD30	171			
AD29	172			
AD28	173			
AD27	174			
AD26	176			
AD25	177			
AD24	178			
AD23	181			
AD22	182			
AD21	184			
AD20	185			
AD19	186			
AD18	187			
AD17	188			
AD16	189			
AD15	202			
AD14	203			
AD13	204			
AD12	205			
AD11	206			
AD10	208			
AD9	1			
AD8	2			
AD7	4			
AD6	5			
AD5	6			
AD4	8			
AD3	9			
AD2	10			
AD1	11			
AD0	12			
C/$\overline{\text{BE3}}$	179	I	CMOS	Bus commands and byte enables. C/$\overline{\text{BE3}}$–C/$\overline{\text{BE0}}$ are multiplexed on the same PCI terminals. During the address phase, C/$\overline{\text{BE3}}$–C/$\overline{\text{BE0}}$ define the bus command. During the data phase, C/$\overline{\text{BE3}}$–C/$\overline{\text{BE0}}$ are used as byte enables. The byte enables determine which byte lanes carry meaningful data. C/$\overline{\text{BE0}}$ applies to byte 0 and C/$\overline{\text{BE3}}$ applies to byte 3.
C/$\overline{\text{BE2}}$	190			
C/$\overline{\text{BE1}}$	201			
C/$\overline{\text{BE0}}$	3			
PAR	200	I/O	12 mA	Parity. During the data phase of PCI reads, the chip calculates even parity across AD31–AD0 and C/$\overline{\text{BE3}}$–C/$\overline{\text{BE0}}$ and outputs the result on PAR.

Terminal Functions (Continued)

PCI interface control

TERMINAL NAME	TERMINAL NO.	I/O TYPE	BUFFER TYPE	FUNCTION
$\overline{\text{DEVSEL}}$	195	O	12 mA	Device select. When asserted, $\overline{\text{DEVSEL}}$ indicates that the PCI1050 decodes its address as the target of the current access.
$\overline{\text{FRAME}}$	192	I	CMOS	Cycle frame. Driven by the current master to indicate the beginning and duration of an access, $\overline{\text{FRAME}}$ is asserted to indicate that a bus transaction is beginning. While $\overline{\text{FRAME}}$ is asserted, data transfers continue. When $\overline{\text{FRAME}}$ is deasserted, the transaction is in the final data phase.
IDSEL	180	I	CMOS	Initialization device select. IDSEL selects the PCI1050 during configuration accesses and can be connected to one of the upper 24 PCI address lines.
$\overline{\text{IRDY}}$	193	I	CMOS	Initiator ready. $\overline{\text{IRDY}}$ indicates the bus master's ability to complete the current data phase of the transaction. $\overline{\text{IRDY}}$ is used with $\overline{\text{TRDY}}$. A data phase is completed on any clock where both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are sampled asserted. During a write, $\overline{\text{IRDY}}$ indicates that valid data is present on AD31–AD0. During a read, $\overline{\text{IRDY}}$ indicates that the master is prepared to accept data. Wait cycles are inserted until both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are asserted together.
$\overline{\text{STOP}}$	196	O	12 mA	Stop. $\overline{\text{STOP}}$ indicates that the PCI1050 is requesting the master to stop the current transaction.
$\overline{\text{PERR}}$	197	O	12 mA	Parity error. $\overline{\text{PERR}}$ is pulsed from the PCI1050, indicating a data parity error has occurred during a write phase.
$\overline{\text{SERR}}$	198	O	12 mA	System error. $\overline{\text{SERR}}$ is pulsed from the PCI1050, indicating an address parity error has occurred.
$\overline{\text{TRDY}}$	194	O	12 mA	Target ready. $\overline{\text{TRDY}}$ indicates the ability of the PCI1050 to complete the current data phase of the transaction. $\overline{\text{TRDY}}$ is used with $\overline{\text{IRDY}}$. A data phase is completed on any clock where both $\overline{\text{TRDY}}$ and $\overline{\text{IRDY}}$ are sampled asserted. During a read, $\overline{\text{TRDY}}$ indicates that valid data is present on AD31–AD0. During a write, $\overline{\text{TRDY}}$ indicates that the PCI1050 is prepared to accept data. Wait cycles are inserted until both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are asserted together.

Terminal Functions (Continued)

PC Card interface controller (slots A and B)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	BUFFER TYPE	FUNCTION
BVD1 ($\overline{\text{STSCHG}}$) ($\overline{\text{RI}}$)	133	68	I	CMOS	Battery voltage detect 1. BVD1 is generated by 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 needs to be replaced. When BVD1 is low, the battery is no longer serviceable and the data in the memory PC Card is lost. Status change. $\overline{\text{STSCHG}}$ alerts the system to a change in the RDY/$\overline{\text{BSY}}$, WP, or BVD condition of the I/O PC Card. Ring indicate. $\overline{\text{RI}}$ is used by modem cards to indicate ring detection.
BVD2 ($\overline{\text{SPKR}}$)	132	67	I	CMOS	Battery voltage detect 2. BVD2 is generated by 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 needs to be replaced. When BVD1 is low, the battery is no longer serviceable and the data in the memory PC Card is lost. Speaker. $\overline{\text{SPKR}}$ is an optional binary-audio signal available only when the card and socket have been configured for the I/O interface. The audio signals from cards A and B can be combined by the PCI1050 and output on $\overline{\text{SPKROUT}}$.
CA25	115	51	O	2 mA	Card address. CA25–CA0 drive PC Card address lines. CA25 is the most-significant bit.
CA24	113	49			
CA23	111	47			
CA22	109	45			
CA21	107	43			
CA20	105	41			
CA19	103	39			
CA18	100	36			
CA17	98	34			
CA16	108	44			
CA15	110	46			
CA14	104	40			
CA13	101	37			
CA12	112	48			
CA11	95	31			
CA10	91	27			
CA9	97	33			
CA8	99	35			
CA7	114	50			
CA6	117	53			
CA5	119	55			
CA4	120	56			
CA3	122	58			
CA2	125	61			
CA1	126	62			
CA0	127	63			

† Terminal name is preceded with A_. For example, the full name for terminal 133 is A_BVD1(A_$\overline{\text{STSCHG}}$)(A_$\overline{\text{RI}}$).
‡ Terminal name is preceded with B_. For example, the full name for terminal 68 is B_BVD1(B_$\overline{\text{STSCHG}}$)(B_$\overline{\text{RI}}$).

Terminal Functions (Continued)

PC Card interface controller (slots A and B) (continued)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	BUFFER TYPE	FUNCTION
$\overline{CD1}$ $\overline{CD2}$	78 135	14 70	I	CMOS	PC Card detect 1 and 2. $\overline{CD1}$ and $\overline{CD2}$ are internally connected to ground on the PC Card. When a PC Card is inserted into a socket, these signals are driven low. The signal status is available by reading the interface status register.
CDATA15 CDATA14 CDATA13 CDATA12 CDATA11 CDATA10 CDATA9 CDATA8 CDATA7 CDATA6 CDATA5 CDATA4 CDATA3 CDATA2 CDATA1 CDATA0	89 87 84 82 80 142 140 137 88 85 83 81 79 141 139 136	25 23 20 18 16 77 74 72 24 21 19 17 15 76 73 71	I/O	CMOS/2 mA	PC Card data. CDATA15 is the most-significant bit.
$\overline{CE1}$ $\overline{CE2}$	90 92	26 28	O	2 mA	PC Card enable 1. $\overline{CE1}$ enables even-numbered address bytes. PC Card enable 2. $\overline{CE2}$ enables odd-numbered address bytes.
INPACK	121	57	I	CMOS	Input acknowledge. Although not applicable in a PCI environment, INPACK is connected to a PC Card socket labeled INPACK.
$\overline{IORD}$	94	30	O	2 mA	I/O read. $\overline{IORD}$ is driven low by the PCI1050 to enable I/O PC Card data output during host I/O read cycles.
$\overline{IOWR}$	96	32	O	2 mA	I/O write. $\overline{IOWR}$ is driven low by the PCI1050 to strobe write data into I/O PC Cards during host I/O write cycles.
$\overline{OE}$	93	29	O	2 mA	Output enable. $\overline{OE}$ is driven low by the PCI1050 to enable memory PC Card data output during host-memory read cycles.
RDY/$\overline{BSY}$ ($\overline{IREQ}$)	129	65	I	CMOS	Ready/busy. RDY/$\overline{BSY}$ provides the ready/busy function when the PC Card and the host socket are configured for the memory-only interface. RDY/$\overline{BSY}$ is driven low by the memory PC Cards to indicate that the memory card circuits are busy processing a previous write command. RDY/$\overline{BSY}$ is set high when memory PC Cards are ready to accept a new data transfer command. Interrupt request. $\overline{IREQ}$ is asserted by an I/O PC Card indicating to the host that a device on the I/O PC Card requires service by the host software. $\overline{IREQ}$ is held at the inactive level when no interrupt is requested.
$\overline{REG}$	124	60	O	2 mA	Attribute memory select. $\overline{REG}$ remains high for all common memory accesses. When $\overline{REG}$ is asserted, access is limited to attribute memory ($\overline{OE}$ or $\overline{WE}$ active) and to the I/O space ($\overline{IORD}$ or $\overline{IOWR}$ 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.

† Terminal name is preceded with A_. For example, the full name for terminal 78 is A_$\overline{CD1}$.
‡ Terminal name is preceded with B_. For example, the full name for terminal 14 is B_$\overline{CD1}$.

Terminal Functions (Continued)

PC Card interface controller (slots A and B) (continued)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	BUFFER TYPE	FUNCTION
RESET	118	54	O	2 mA	PC Card reset. RESET forces a hard reset to a PC Card.
VCC_3	146	150	O	2 mA	3.3-V V_{CC}. VCC_3 enables a 3.3-V supply onto the card V_{CC} terminal. VCC_3 is mutually exclusive with VCC_5.
VCC_5	145	149	O	2 mA	5-V V_{CC}. VCC_5 enables a 5-V supply onto the card V_{CC} terminal. VCC_5 is mutually exclusive with VCC_3.
VPP_PGM	148	152	O	2 mA	V_{PP} program. VPP_PGM enables the programming voltage onto the card V_{PP} terminal. VPP_PGM is mutually exclusive with VPP_VCC.
VPP_VCC	147	151	O	2 mA	V_{PP} is V_{CC}. VPP_VCC enables the socket V_{CC} supply onto the card V_{PP} terminal. VPP_VCC is mutually exclusive with VPP_PGM.
VS1 VS2	128 116	64 52	I	CMOS	Voltage sense 1 and 2. VS1 and VS2, when used together, determine the operating voltage of the PC Card and are internally tied to the 5-V power terminals.
$\overline{\text{WAIT}}$	130	66	I	CMOS	Bus cycle wait. $\overline{\text{WAIT}}$ is driven by a PC Card to delay completion of the memory or I/O cycle that is in progress.
$\overline{\text{WE}}/\overline{\text{PRGM}}$	106	42	O	2 mA	Write enable/program. $\overline{\text{WE}}/\overline{\text{PRGM}}$ is used for strobing memory write data into memory PC Cards. $\overline{\text{WE}}/\overline{\text{PRGM}}$ is also used for memory PC Cards that employ programmable-memory technologies.
WP ($\overline{\text{IOIS16}}$)	134	69	I	CMOS	Write protect. WP reflects the status of the write-protect switch on memory PC Cards. For I/O cards, WP is used for the 16-bit port ($\overline{\text{IOIS16}}$) function. The status of the signal can be read in the interface status register. I/O is 16 bits. $\overline{\text{IOIS16}}$ is asserted by the PC Card when the address on the bus corresponds to an address to which the PC Card responds and when the I/O port addressed is capable of 16-bit accesses.

† Terminal name is preceded with A_. For example, the full name for terminal 129 is A_RDY/$\overline{\text{BSY}}$(A_$\overline{\text{IREQ}}$).
‡ Terminal name is preceded with B_. For example, the full name for terminal 65 is B_RDY/$\overline{\text{BSY}}$(B_$\overline{\text{IREQ}}$).

Terminal Functions (Continued)

miscellaneous

TERMINAL NAME	TERMINAL NO.	I/O TYPE	BUFFER TYPE	FUNCTION
IRQ3/$\overline{\text{INTA}}$ IRQ4/$\overline{\text{INTB}}$ IRQ5/$\overline{\text{INTC}}$ IRQ7/$\overline{\text{INTD}}$	154 155 156 157	O	4 mA	Interrupt request 3, 4, 5, 7. These terminals can be connected to either PCI or ISA interrupts.
IRQ9 IRQ10 IRQ11 IRQ12 IRQ14 IRQ15	158 159 160 161 162 163	O	4 mA	Interrupt request 9, 10, 11, 12, 14, 15. These terminals indicate an interrupt request from one of the PC Cards.
$\overline{\text{SPKROUT}}$	144	O	2 mA	Speaker. $\overline{\text{SPKROUT}}$ carries the digital audio signal from the PC Card.

power supply

TERMINAL NAME	TERMINAL NO.	I/O TYPE	BUFFER TYPE	FUNCTION
GND	7, 22, 59, 86, 123, 153, 167, 183, 199	I	—	Device ground terminals
NC	168, 169	—	—	No internal connection. These are for use on products to be developed by TI.
V_{CCA}	102, 138	I	—	Power-supply terminals for card A
V_{CCB}	38, 75	I	—	Power-supply terminals for card B
V_{CCE}	137	I	—	Power-supply terminal for control interface
V_{CCP}	13, 164, 175, 191, 207	I	—	Power-supply terminals for PCI interface and core logic
V_{CORE}	131	I	—	Power-supply terminal for PCI core logic

architecture

This section provides the following: an introduction to PCMCIA, descriptions of the PC Card 3.3-V/5-V operation, PCI and PC Card interfaces, memory mapping, I/O mapping, interrupts, and PCI configuration space.

introduction to PCMCIA

PCMCIA 2.1 provides a hardware- and software-interface standard for connecting credit-card-sized memory and I/O cards to personal computers. By implementing compliant card slots, PC manufacturers allow customers to use industry-standard PCMCIA memory and I/O cards from many different vendors. The PCMCIA 2.1 standard is an extension of the previous PCMCIA 1.01 and JEIDA 4.1 standards.

PCMCIA cards, also called PC Cards, can have two types of memory: attribute memory containing card configuration registers and data, and common memory used by the application. Attribute memory contains the card information structure (CIS) defined by PCMCIA 2.1. CIS is read by PC system software to determine the capabilities of the card. To allow applications to access card memory, the PC Card adapter should support a window-mapping scheme similar to the expanded memory managers used in PCs. This allows the PC to map areas of card memory into unused areas of the PC memory space. The ExCA standard (also called quick swap) requires card adapters to implement five memory windows for each card slot.

introduction to PCMCIA (continued)

By reading the CIS, the PC can determine if a card is memory only or is I/O capable. Since I/O cards typically need to interrupt the host, the PC Card-slot hardware should include logic to route the single card-interrupt output to one of the PC IRQ interrupt lines. I/O cards generally have a small number of I/O ports that need to be mapped into the PC I/O space for access by device drivers and applications.

3.3-V/5-V operation

The PCI1050 is designed to operate at 5 V with card interfaces powered at 3.3 V or 5 V. The PC Card A and B interfaces have separate V_{CC} terminals that should be connected to the card V_{CC}. This means that both 3.3-V and 5-V cards can be connected directly to the PCI1050 (no external level-shifting buffers are needed). Because the card V_{CC} terminals are completely independent, one card can be powered at 5 V while the other is powered at 3.3 V.

The PCI1050 outputs two V_{CC} control signals each for PC Cards A and B that can be used to control external card power supplies. This allows software to dynamically change PCI1050 card interface V_{CC} during device operation. The PCI1050 control interface also can be powered at 3.3 V or 5 V; however, this is normally hardwired in the system and does not change dynamically.

host interface

The PCI1050 interfaces directly to the PCI bus with no external buffering. From a software standpoint, the PCI1050 occupies PCI configuration space, I/O space, and memory space.

- PCI configuration space. The PCI1050 implements a single PCI configuration space with a standard 64-byte header region as defined by PCI revision 2.0.
- I/O space. Host software can program PC Card I/O windows at any byte boundary in the first 64K bytes of host I/O space.
- Memory space. Host software can program PC Card windows at any 4K-byte boundary in a 16M-byte page of PCI memory space. The 16M-byte page is selected using the memory window page register in the PCI as configuration space.

PCI interface

The PCI1050 conforms to the signal timing given in the PCI specification for all outputs valid within a maximum of 11 ns after the PCI clock rising edges. All PCI outputs are driven with output buffers that conform to PCI ac-switching performance requirements.

The PCI1050 uses positive address decode to determine if the PCI address falls within any enabled card memory or I/O window, or matches the I/O data/index port used to access the compatibility registers. If a match is detected, the PCI1050 asserts $\overline{\text{DEVSEL}}$ at the start of clock 4 as a medium-speed peripheral.

The PCI1050 does not support multiple data phases and always forces a disconnect by asserting $\overline{\text{STOP}}$ and $\overline{\text{TRDY}}$ during the first data phase. The PCI1050 signals a target abort only for I/O cycles where the byte-enable outputs by the master correspond to addresses outside the decode hit range. In this case, the PCI1050 deasserts $\overline{\text{DEVSEL}}$ and asserts $\overline{\text{STOP}}$ without asserting $\overline{\text{TRDY}}$. The PCI retry mechanism is not supported and the PCI1050 only terminates cycles with a disconnect or target abort.

PCI-to-PC Card cycle translation

All valid PCI cycles that represent a hit to one of the card interfaces are translated to appropriate PC Card cycle(s). This includes 8-, 16-, and 32-bit read/write cycles (contiguous or split). Translation depends on the type of card and the card size. Table 3 can be used to determine the number of PC Card cycles generated for a given PCI cycle. Once the translation is selected, the cycle is serialized to the PC Card.

Table 3. PC Card Cycle Count for Given PCI Cycle

PCI BYTE ENABLES				NO. OF PC Card CYCLES	
$\overline{BE3}$	$\overline{BE2}$	$\overline{BE1}$	$\overline{BE0}$	8-BIT CARD	16-BIT CARD
1	1	1	1	0	0
1	1	1	0	1	1
1	1	0	1	1	1
1	1	0	0	2	1
1	0	1	1	1	1
1	0	1	0	2	2
1	0	0	1	2	2
1	0	0	0	3	2
0	1	1	1	1	1
0	1	1	0	2	2
0	1	0	1	2	2
0	1	0	0	3	2
0	0	1	1	2	1
0	0	1	0	3	2
0	0	0	1	3	2
0	0	0	0	4	2

PCI cycles can be aborted under two conditions connected with I/O cycles. All cycles on the PCI bus are accepted or rejected based on the address phase of the cycle. For I/O cycles, the $\overline{BE3}$–$\overline{BE0}$ bits can cause a cycle to become invalid. Since an I/O window setup for each card has byte granularity, it is possible for an I/O cycle to stretch across window boundaries once byte lanes are enabled. For example, the first word of a 32-bit PCI I/O cycle might be within an I/O window boundary, but the second word could cross the boundary and not be mapped by the card. In this case, the cycle is aborted by the device using standard PCI bus-cycle abort protocol. This scenario never occurs for memory cycles that have a 4K-byte window granularity. The second possible cause of a cycle abort is the validity of the I/O cycle itself. The internal state machine performs a check that compares $\overline{BE3}$–$\overline{BE0}$ to AD1–AD0 and confirms the validity of the cycle (i.e., the byte enables agree with the lower two bits of the address).

PCI write buffer

A four-deep write buffer, which can be configured as either one deep or four deep, is maintained for each card interface and can be used as a buffer in case of consecutive writes to the same card. This write buffer can be turned on/off and can be configured to accept only memory cycles or both memory and I/O cycles. The write buffer is configured using the IOBUF, FDEP, and FEN bits in the write-buffer-control register. This register also contains two additional bits (FULL and EMPTY) that indicate the current status of the write buffer.

An additional buffer is maintained on each card interface so that even if the write buffer is turned off, single-card writes (both memory and I/O writes) are buffered, freeing the PCI bus as soon as possible. An additional write holds the PCI bus until the previous cycle is completed. With the buffer on, this additional write fills the first write buffer and frees the PCI bus.

PCI latency

The latency of the cycle is dependent on a number of factors, including the size of the card, the type of card, the status of the write buffers, and the cycle type and size. Minimum latency occurs on a write that can be placed immediately in the write buffer. Maximum latency for a given card setup occurs on a read with the four-deep write buffer full. In this case, the write buffer must first be emptied before proceeding with the read. Latency is longer for a greater cycle size and a smaller card size; therefore, a 32-bit read from an 8-bit card represents worst case. The card assertion of $\overline{\text{WAIT}}$ also increases latency.

PC Card interface

The PCI1050 performs the bus-cycle conversion between the PCI bus and the PCMCIA bus and generates all the card address and control signals. When the PC reads or writes to an enabled memory or I/O window, the PCI1050 enables the appropriate card and executes a PCMCIA read or write cycle.

I/O and memory PC Cards

The PCMCIA release 2.0 standard specifies that all cards when first inserted must behave as memory cards. This means that cards must not respond to I/O cycles and that they must drive the dual-function PC Card signals as memory-card signals. After the host system reads the CIS from the card attribute memory, it can enable I/O-capable cards by writing to the on-card configuration option register. At this point, the I/O card starts to drive the dual-function signals in their I/O mode of operation. The PCI1050 interprets these card signals as either memory or I/O mode, depending on the value of bit CTYPE in the interrupt and general control register.

ATA hard-drive support

The PCI1050 supports the ATA interface defined by PCMCIA release 2.0. I/O addresses 3F7h and 377h are configured as read only so that the PCI1050 does not respond during writes to a floppy disk. This feature is enabled by setting bit ATAEN in the PCI1050 initialization register.

power-down mode

To enter the power-down mode, software sets b0 PWRDN in the Intel-compatible global control register. In power-down mode, all PC Card outputs and bidirectionals are in the high-impedance state. These terminals are:

A_CA25–A_CA0, A_CDATA15–A_CDATA0, A_$\overline{\text{IORD}}$, A_$\overline{\text{IOWR}}$, A_$\overline{\text{REG}}$, A_$\overline{\text{OE}}$, A_$\overline{\text{WE/PRGM}}$, A_RESET, A_$\overline{\text{CE1}}$, A_$\overline{\text{CE2}}$

B_CA25–B_CA0, B_CDATA15–B_CDATA0, B_$\overline{\text{IORD}}$, B_$\overline{\text{IOWR}}$, B_$\overline{\text{REG}}$, B_$\overline{\text{OE}}$, B_$\overline{\text{WE/PRGM}}$, B_RESET, B_$\overline{\text{CE1}}$, B_$\overline{\text{CE2}}$

All other terminals function as in normal operation. All internal registers retain their contents and are fully accessible via PCI. All card- and status-change interrupts remain enabled. The device responds to PCI accesses, but does not execute the cycles on the PC Card interface.

PC Card cycle timing

The PCI1050 generates PC Card cycles with Intel 82365SL-DF-compatible timing. In an Intel 82365SL-DF-based system, PC Card cycle timing is determined by the duration of the host cycle on the AT bus. To change the PC Card cycle timing, the host programs the wait-state bits in the memory and I/O window registers to lengthen the AT bus cycle. For compatibility with the Intel 82365SL-DF, the PCI1050 also uses the wait-state bits to increase the PC Card cycle duration. For each possible setting of the wait-state bits, the PCI1050 reproduces the same cycle timing as the Intel 82365SL-DF connected to an 8.33-MHz PC/AT bus. This ensures that software written for the Intel 82365SL-DF device runs as the PCI1050 without modification.

pullup resistors

The PCMCIA 2.1 standard specifies that all input signals to the PCI1050 have external pullup resistors added to minimize sink current to 400 μA. The pullup resistors must have a value that is greater than 10 kΩ and should be connected to the PC Cards A and B V_{CC} supplies, respectively. Table 4 shows the signals that are inputs to the PCI1050.

Table 4. PC Card Interface-Controller Pullup-Resistor Inputs

SIGNAL NAME	SIGNAL NUMBER SLOT A†	SIGNAL NUMBER SLOT B‡
BVD1($\overline{\text{STSCHG}}$)($\overline{\text{RI}}$)	133	68
BVD2($\overline{\text{SPKR}}$)	132	67
RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$)	129	65
$\overline{\text{CD1}}$	78	14
$\overline{\text{CD2}}$	135	70
VS1	128	64
VS2	116	52
$\overline{\text{WAIT}}$	130	66
WP ($\overline{\text{IOIS16}}$)	134	69
INPACK	121	57

† The signal name is preceded with A_. For example, the full name for signal 133 is A_BVD1(A_$\overline{\text{STSCHG}}$)(A_$\overline{\text{RI}}$).

‡ The signal name is preceded with B_. For example, the full name for signal 68 is B_BVD1(B_$\overline{\text{STSCHG}}$)(B_$\overline{\text{RI}}$).

memory mapping

A principal feature of the PCI1050 is its ability to map areas of card memory into the host memory space. The PCI1050 implements a total of ten independent memory windows with five dedicated to each of the PC Cards A and B. Each window can start and stop on any 4K-byte address boundary above the first 64K bytes in host memory and can access 16-bit or 8-bit card memory. Programmable address offsets allow each window to be located anywhere in the 64M-byte card memory space, whatever its position in host memory space.

Memory windows can be mapped to either the card attribute or common memory space. This means that for PC Cards A and B, the host can set up one window to access the CIS located in attribute memory and another window to access data stored in common memory.

Each memory window has a set of six internal registers associated with it that defines its size, location, offset, data width, and cycle attributes. Most of the register bits are used to program the host memory window start and end addresses and the card memory offset. The window start and end addresses are 14 bits and correspond to host address bits AD23–AD12 to give a minimum window resolution of 4K bytes. The offset address is two bits longer and corresponds to card address bits CA25–CA12.

The PCI1050 also contains two page registers, one each for PC Cards A and B, that allow the memory windows to be located above the first 16M bytes of system address space. The system address bits AD31–AD24 are compared with the page-register values, and if they match, the PCI1050 memory window decode logic is enabled. This allows the PC Card memory windows to be located in any of the 256 separate 16M-byte pages that comprise the 4G-byte PCI address space.

memory mapping (continued)

The start and end addresses for the window in card memory are calculated by adding the offset to the host memory start and end addresses. For each host memory window access, the PCI1050 adds the offset to the incoming host address to generate the correct 26-bit card address. The PCI1050 adder wraps around to 0 at the top of the 64M-byte card address space to allow both positive and negative address offsets using two's complement arithmetic.

I/O mapping

The PCI1050 can map areas of the card I/O space into the host I/O space. The PCI1050 implements four independent I/O windows with two each dedicated to PC Cards A and B. Each window can start and stop on any byte address boundary inside the first 64K bytes in host memory and can access 16-bit or 8-bit card ports.

To allow I/O remapping, the PC Cards should decode only the minimum number of card address lines required to address the number of I/O locations they have. This means that an I/O card with six I/O locations decodes CA0, CA1, and CA2 and ignores all higher address bits. The PC Card INPACK signal is not used for I/O address decode and is implemented for compatibility with products to be developed by TI.

Each I/O window has a set of four internal registers associated with it that define its size, location, data width, and cycle attributes. Most of the register bits are used to program the host I/O window start and end addresses. The window start and end addresses are 16 bits and correspond to host address bits AD15–AD0, giving a minimum window resolution of one byte.

interrupts

The PCI1050 provides ten host interrupt terminals that can be configured either as PCI open-drain outputs or positive-edge ISA IRQ outputs. Host interrupts can be triggered by two types of events:

- I/O card interrupts. PC Cards configured in I/O-mode output level or pulse-mode interrupts on device terminal RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$).
- Status-change interrupts. These occur when PC Card signals RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$), BVD1($\overline{\text{STSCHG}}$), and/or BVD2($\overline{\text{SPKR}}$), or the PC Card detect ($\overline{\text{CD1}}$, $\overline{\text{CD2}}$) a change state.

I/O card interrupts

The PCI1050 can independently route card A and B I/O interrupts to any of ten host-interrupt terminals connected to PCI or ISA host interrupts.

The PC Card interrupts on RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$) can be level or pulse mode as defined in the PCMCIA 2.1 specification. In both cases, the PC Card drives RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$) active low to request an interrupt. In pulse mode, the PC Card releases RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$) after a fixed time interval; whereas in level mode, the PC Card continues to assert RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$) until the interrupt is serviced.

Each PCI1050 interrupt terminal can be configured either as a totem-pole output for connection to an ISA-type host interrupt or as an open-drain output for connection to a PCI-type host interrupt. Each interrupt terminal has a control bit in the TI interrupt-mode registers that, when set, configures the interrupt for ISA-type operation. Following reset, these bits are cleared for default PCI open-drain operation.

For compatibility with the Intel 82365SL-DF, active-low open-drain operation also can be selected by setting bits AIREQLM and BIREQLM in the global control register. However, ISA-type operation can be enabled only if the relevant TI interrupt-mode register bits are set.

For an ISA-configured interrupt, the PCI1050 drives IRQ low until the PC Card asserts RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$). The PCI1050 responds by driving IRQ high, causing a positive-edge-triggered host interrupt. When the PC Card deasserts RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$), the PCI1050 drives IRQ low again.

I/O card interrupts (continued)

For a PCI-configured interrupt, the PCI1050 places IRQ in the high-impedance state until the PC Card asserts RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$). The PCI1050 responds by driving IRQ low, causing a level-mode host interrupt. When the PC Card deasserts RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$), the PCI1050 places IRQ in the high-impedance state again. The PC Card should be configured for level-mode operation to allow host-interrupt sharing.

The PCI1050 provides a card-interrupt flag in the miscellaneous register that is set when a PC Card asserts RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$). This feature allows software to determine the source of a pulse-mode card interrupt without interrogating the PC Card.

status-change interrupts

The PCI1050 can independently route status-change interrupts to any of the ten host-interrupt terminals connected to PCI or ISA host interrupts. The events that can cause status-change interrupts depend on whether the PC Card is configured as I/O or memory:

- I/O-configured PC Cards: battery-voltage-detect input BVD1($\overline{\text{STSCHG}}$) low, indicating a change in battery voltage, write protect, or ready/busy status
- Memory-configured PC Cards: one or both of battery-voltage-detect inputs BVD1($\overline{\text{STSCHG}}$) and/or BVD2($\overline{\text{SPKR}}$) low, indicating battery deterioration; or ready/busy input RDY/$\overline{\text{BSY}}$($\overline{\text{IREQ}}$) transition, indicating ready/busy status change
- All cards: one or both PC Card detect inputs ($\overline{\text{CD1}}$ and/or $\overline{\text{CD2}}$) transition, indicating PC Card insertion or removal

Each PCI1050 interrupt terminal can be configured either as a totem-pole output for connection to an ISA-type host interrupt or as an open-drain output for connection to a PCI-type host interrupt. Each interrupt terminal has a control bit in the TI interrupt-mode registers that, when set, configures the interrupt for ISA-type operation. Following reset, these bits are cleared for default PCI open-drain operation.

For compatibility with the Intel 82365SL-DF, active-low open-drain operation also can be selected by setting bit CSC in the global control register. However, ISA-type operation can be enabled only if the relevant TI interrupt-mode register bits are set.

For an ISA-configured interrupt, the PCI1050 drives IRQ low until a card status change occurs. The PCI1050 responds by driving IRQ high, causing a positive-edge-triggered host interrupt. When the interrupt service routine clears the status-change flag in the card status-change register, the PCI1050 drives IRQ low again.

For a PCI-configured interrupt, the PCI1050 places IRQ in the high-impedance state until a card status change occurs. The PCI1050 responds by driving IRQ low, causing a level-mode host interrupt. When the interrupt service routine clears the status-change flag in the card status-change register, the PCI1050 places IRQ in the high-impedance state again.

To determine the source of any card status-change interrupt, the host can read the flag bits in the card status-change register. The flags can either be cleared automatically by the read operation or explicitly by writing a 1 to the set flag. This option is controlled by bit XWBCSC in the global control register. When all flags are cleared, the selected IRQ is returned to the inactive state.

parity generation and checking

Per the PCI specification, parity generation is required and checking is optional. The PCI1050 supports both parity generation and checking in both address and data phases. When a parity error occurs during a bus transaction, $\overline{\text{PERR}}$ is asserted. If either a PC Card interface-system error or an address-parity error occurs, $\overline{\text{SERR}}$ is asserted for one clock cycle.

power management

The primary power dissipation in the PCI1050 is dynamic and occurs each time the internal and external logic levels in the device change state. Within the device, static power dissipation is insignificant since the PCI1050 utilizes CMOS technology; however, at the system- or device-interface level, static power dissipation needs to be considered, primarily where the pullup or pulldown resistors are required.

The PCI1050 implements two modes of power management, suspend and disable, as shown in Figure 1.

- Suspend mode is a power-managed mode of operation in which complete device functionality is maintained, but allows portions of the device to be powered down completely. The internal state in the device is maintained and PC Card interrupts are passed through to PCI interface. This mode is obtained through software (Card and Socket Services) by programming the PWRDN bit located in the global control register. Programming this bit allows the host software to force the device into or out of suspend mode. The PCI interface on the PCI1050 is always active in this state. The PC Card interfaces on the controller are power managed. In the suspend mode, the following signals are in the high-impedance state when a PC Card is present in the socket: CADR[25–0], CDATA[15–0], $\overline{CE1}$, $\overline{CE2}$, $\overline{IORD}$, $\overline{IOWR}$, $\overline{OE}$, $\overline{REG}$, $\overline{RESET}$, and $\overline{WE}$.
- Disable mode, except for the complete power down of the device, offers the most power savings, but the device is nonfunctional. This mode is obtained when the PCI $\overline{RSTIN}$ signal is asserted and all pins on the device are in the high-impedance state. When the $\overline{RSTIN}$ signal is deasserted, the PCI1050 returns to its normal mode.

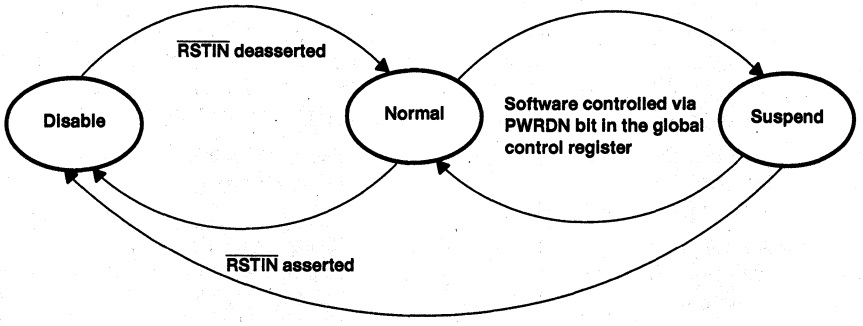


Figure 1. PCI1050 Power-Management Modes of Operation

PCI configuration space

The PCI1050 implements a single PCI configuration-register space with a standard header region defined by PCI revision 2.0 occupying the bottom 64 bytes. The PCI1050 maps all the Intel compatibility registers into the top 128 bytes of configuration space to provide an alternative means of access without conflict with other devices. Some of the remaining 64 bytes are used for TI-specific registers.

The host accesses the configuration-register space using PCI configuration read and write cycles. During the address phase of a configuration cycle, the host PCI bridge asserts one of AD31–AD11, depending on which PCI device the host wants to access. The system designer should connect the PCI1050 input IDSEL to the AD line corresponding to the physical PCI device number assigned to the PCI1050. Address bits AD10–AD8 carry the functional PCI device number and are ignored by the PCI1050, which is a single-function device. Address bits AD7–AD2 carry the double-word address of the particular configuration register and are decoded internally by the PCI1050. The 256-byte configuration space contains three sets of registers: Intel compatibility registers (also accessible via the data/index port in I/O space), TI extension registers, and the standard PCI configuration header registers (see Figure 2).

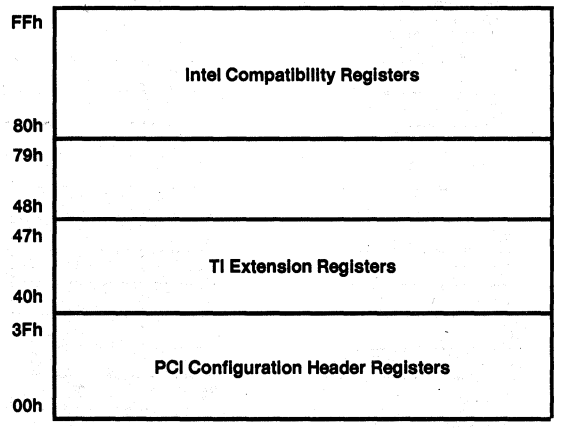


Figure 2. PCI Configuration-Space Registers

compatibility registers

The PCI1050 is fully register compatible with the Intel 82365SL-DF card-controller chip. The compatibility registers can be accessed via the host I/O space or via the PCI configuration header register space. For I/O access, the PCI1050 uses the same index and data I/O port scheme introduced by Intel.

One I/O port is a read/write 8-bit index register and the other port is an 8-bit read/write data register. To access any register, the host writes an index value to the I/O index port and then either reads or writes the register contents to or from the I/O data port. Table 5 shows the index offset for each compatibility register with its corresponding address in the PCI configuration header register space.

compatibility registers (continued)

Table 5. Compatibility Registers

NAME	PCI ADDRESS		ExCA OFFSET	
	SOCKET A	SOCKET B	SOCKET A	SOCKET B
General Setup Registers				
Identification and revision register	80	C0	00	40
Interface status register	81	C1	01	41
Power and RESETDRV control register	82	C2	02	42
Card status-change register	84	C4	04	44
Address window-enable register	86	C6	06	46
Global control register	9E	DE	1E	5E
Card detect and general control register	96	D6	16	56
Interrupt Registers				
Interrupt and general control register	83	C3	03	43
Card status-change interrupt configuration register	85	C5	05	45
I/O Window Registers				
I/O window control register	87	C7	07	47
I/O window 0 start-address low-byte register	88	C8	08	48
I/O window 0 start-address high-byte register	89	C9	09	49
I/O window 0 end-address low-byte register	8A	CA	0A	4A
I/O window 0 end-address high-byte register	8B	CB	0B	4B
I/O window 1 start-address low-byte register	8C	CC	0C	4C
I/O window 1 start-address high-byte register	8D	CD	0D	4D
I/O window 1 end-address low-byte register	8E	CE	0E	4E
I/O window 1 end-address high-byte register	8F	CF	0F	4F
Memory Window Registers				
Memory window 0 start-address low-byte register	90	D0	10	50
Memory window 0 start-address high-byte register	91	D1	11	51
Memory window 0 end-address low-byte register	92	D2	12	52
Memory window 0 end-address high-byte register	93	D3	13	53
Memory window 0 offset-address low-byte register	94	D4	14	54
Memory window 0 offset-address high-byte register	95	D5	15	55
Reserved	97	D7	17	57
Memory window 1 start-address low-byte register	98	D8	18	58
Memory window 1 start-address high-byte register	99	D9	19	59
Memory window 1 end-address low-byte register	9A	DA	1A	5A
Memory window 1 end-address high-byte register	9B	DB	1B	5B
Memory window 1 offset-address low-byte register	9C	DC	1C	5C
Memory window 1 offset-address high-byte register	9D	DD	1D	5D
Reserved	9F	DF	1F	5F

compatibility registers (continued)

Table 5. Compatibility Registers (Continued)

NAME	PCI ADDRESS		ExCA OFFSET	
	SOCKET A	SOCKET B	SOCKET A	SOCKET B
Memory Window Registers (continued)				
Memory window 2 start-address low-byte register	A0	E0	20	60
Memory window 2 start-address high-byte register	A1	E1	21	61
Memory window 2 end-address low-byte register	A2	E2	22	62
Memory window 2 end-address high-byte register	A3	E3	23	63
Memory window 2 offset-address low-byte register	A4	E4	24	64
Memory window 2 offset-address high-byte register	A5	E5	25	65
Reserved	A6	E6	26	66
Reserved	A7	E7	27	67
Memory window 3 start-address low-byte register	A8	E8	28	68
Memory window 3 start-address high-byte register	A9	E9	29	69
Memory window 3 end-address low-byte register	AA	EA	2A	6A
Memory window 3 end-address high-byte register	AB	EB	2B	6B
Memory window 3 offset-address low-byte register	AC	EC	2C	6C
Memory window 3 offset-address high-byte register	AD	ED	2D	6D
Reserved	AE	EE	2E	6E
Reserved	AF	EF	2F	6F
Memory window 4 start-address low-byte register	B0	F0	30	70
Memory window 4 start-address high-byte register	B1	F1	31	71
Memory window 4 end-address low-byte register	B2	F2	32	72
Memory window 4 end-address high-byte register	B3	F3	33	73
Memory window 4 offset-address low-byte register	B4	F4	34	74
Memory window 4 offset-address high-byte register	B5	F5	35	75
Reserved	B6	F6	36	76
Reserved	B7	F7	37	77
Reserved	B8	F8	38	78
Reserved	B9	F9	39	79
Reserved	BA	FA	3A	7A
Reserved	BB	FB	3B	7B
Reserved	BC	FC	3C	7C
Reserved	BD	FD	3D	7D
Reserved	BE	FE	3E	7E
Reserved	BF	FF	3F	7F

general setup registers

Identification and revision register

PCI addresses (hex): Socket A: 80, Socket B: C0 — ExCA offset (hex): Socket A: 00, Socket B: 40

This read-only register is used by the system software to determine the Intel revision compatibility.

Bit	7	6	5	4	3	2	1	0
Name	IFTYPE1	IFTYPE0	—	—	IREV3	IREV2	IREV1	IREV0
Default	1	0	0	0	0	1	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–6	IFTYPE1–IFTYPE10	R	This field is hardwired to 10b for compatibility with the Intel 82365SL-DF.
5–4	—	R	Reserved
3–0	COMP3–COMP30	R	This field is hardwired to 0100b for compatibility with the Intel 82365SL-DF.

Interface status register

PCI addresses (hex): Socket A: 81 Socket B: C1 ExCA offset (hex): Socket A: 01 Socket B: 41

The read-only interface status register provides the current status of the PC Card socket interface signals.

Bit	7	6	5	4	3	2	1	0
Name	—	CPOWER	RDY/BSY	CWP	$\overline{CD2}$	$\overline{CD1}$	BVD2	BVD1(STS)
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	—	R	Reserved
6	CPOWER	R	PC Card power active. CPOWER indicates the current power status of the socket. 0 = Power to the socket is turned off. 1 = Power is provided to the socket. Outputs VCC_5, VCC_3, VPP_PGM, VPP_VCC are set according to the power control register bits.
5	RDY/BSY	R	Ready/busy. RDY/BSY indicates the ready/busy condition of the PC Card. 0 = PC Card is busy. 1 = PC Card is ready.
4	CWP	R	Memory write protect. Bit value is the logic level of the WP signal on the memory PC Card interface. 0 = WP input is 0. Card is read and write. 1 = WP input is 1. Card is read only. Memory write access to the slot is not blocked unless the write-protect bit in the associated card memory offset-address high-byte register is also set to 1.
3	$\overline{CD2}$	R	Card detect 2. With card detect 1, $\overline{CD2}$ indicates that a card is present in the socket and fully seated. 0 = $\overline{CD2}$ input is 1. 1 = $\overline{CD2}$ input is 0 (card inserted).
2	$\overline{CD1}$	R	Card detect 1. With card detect 2, $\overline{CD1}$ indicates that a card is present in the socket and fully seated. 0 = $\overline{CD1}$ input is 1. 1 = $\overline{CD1}$ input is 0 (card inserted).
1–0	BVD2–BVD1	R	Battery voltage detect 2 and 1. BVD2–BVD1 reflect the value of input terminals BVD1($\overline{STSCHG}$) and BVD2($\overline{SPKR}$). **BVD2 BVD1 Battery Voltage** 0 0 Battery dead 1 0 Battery dead 0 1 Battery replacement warning 1 1 Battery good
0	STS	R	For I/O PC Cards, STS indicates the status of the $\overline{STSCHG}$/$\overline{RI}$ signal from the PC Card when the ring-indicate enable bit in the interrupt and general control register is set to 0.

power and RESETDRV control register

PCI addresses (hex): Socket A: 82 Socket B: C2 ExCA offset (hex): Socket A: 02 Socket B: 42

This read/write register controls the PC Card power.

Bit	7	6	5	4	3	2	1	0
Name	COE	—	AUTOPWR	VCC1	VCC0	—	VPP1	VPP0
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	COE	R/W	Output enable 0 = PC Card outputs CADR25–CADR0, $\overline{CE1}$, $\overline{CE2}$, $\overline{IORD}$, $\overline{OE}$, $\overline{REG}$, RESET, $\overline{WE}$, DATA15–DATA0 in the high-impedance state. 1 = PC Card outputs (as above) are enabled.
6	—	R/W	Reserved
5	AUTOPWR	R/W	Automatic-power switch enable 0 = Automatic-socket-power switching is disabled. 1 = Automatic-socket-power switching is enabled.
4–3	VCC1–VCC0	R/W	V_{CC} control bits. VCC1–VCC0 control card V_{CC} using outputs VCC_5 and VCC_3. **VCC1 VCC0 VCC_5 VCC_3 V_{CC} Voltage** 0 0 0 0 No connect 0 1 0 0 Reserved 1 0 1 0 5 V 1 1 0 1 3.3 V
2	—	R/W	Reserved
1–0	VPP1–VPP0	R/W	V_{PP} control bits. VPP1–VPP0 control card V_{PP} using outputs VPP_PGM and VPP_VCC. **VCC1 VPP1 VPP0 VPP_PGM VPP_VCC V_{PP} Voltage** 0 X X 0 0 No connect 1 0 0 0 0 No connect 1 0 1 0 V_{CC} 1 1 0 1 0 12 V 1 1 1 0 0 Reserved

card status-change register

PCI addresses (hex): Socket A: 84, Socket B: C4 — ExCA offset (hex): Socket A: 04, Socket B: 44

The card status-change register contains flag bits for each type of card status change. Each flag can be enabled as an interrupt source by writing to the card status-change interrupt configuration register. The status flags can be cleared automatically by a register read or explicitly by writing a 1 to each set flag. The method used depends on whether bit XWBCSC in the global control register is set or clear.

If the card status-change interrupt is enabled to one of the system bus-interrupt request lines, the corresponding IRQ signal remains high until the register is read.

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	CDCHG	RDYCHG	BWARN	BDEAD
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–4	—	R/W	Reserved
3	CDCHG	R	Card detect change 0 = No change is detected on either $\overline{CD1}$ or $\overline{CD2}$ signals. 1 = A change is detected on either $\overline{CD1}$ or $\overline{CD2}$ signals.
2	RDYCHG	R	Ready change 0 = No low-to-high change is detected on RDY/$\overline{BSY}$. 1 = Detected low-to-high change of the RDY/$\overline{BSY}$ signal indicates that the memory PC Card is ready to accept a new data transfer.
1	BWARN	R	Battery warning 0 = Battery warning condition is not detected. 1 = Battery warning condition is detected.
0	BDEAD	R	Battery dead 0 = Battery dead condition is not detected. 1 = Battery dead condition is detected.

address window-enable register

PCI addresses (hex): Socket A: 86, Socket B: C6 — ExCA offset (hex): Socket A: 06, Socket B: 46

The address window-enable register controls the enabling of the memory and I/O mapping windows to the PC Card memory or I/O space.

I/O window enables control I/O accesses within the I/O address for the window specified. When PC Card enables are generated, I/O accesses transmit addresses from the system bus directly through to the PC Card.

Bit	7	6	5	4	3	2	1	0
Name	IW1EN	IW0EN	—	MW4EN	MW3EN	MW2EN	MW1EN	MW0EN
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	IW1EN	R/W	I/O window 1 enable 0 = Disable 1 = Enable
6	IW0EN	R/W	I/O window 0 enable 0 = Disable 1 = Enable
5	—	R/W	Reserved
4	MW4EN	R/W	Memory window 4 enable 0 = Disable 1 = Enable
3	MW3EN	R/W	Memory window 3 enable 0 = Disable 1 = Enable
2	MW2EN	R/W	Memory window 2 enable 0 = Disable 1 = Enable
1	MW1EN	R/W	Memory window 1 enable 0 = Disable 1 = Enable
0	MW0EN	R/W	Memory window 0 enable 0 = Disable 1 = Enable

global control register

PCI addresses (hex): Socket A: 9E, Socket B: DE

ExCA offset (hex): Socket A: 1E, Socket B: 5E

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	BIREQLM	AIREQLM	XWBCSC	CSCLM	PWRDN
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–5	—	R/W	Reserved
4	BIREQLM	R/W	Card B interrupt-mode select 0 = Host IRQ is a positive-edge ISA-style interrupt. 1 = Host IRQ is a open-drain active-low PCI-style interrupt.
3	AIREQLM	R/W	Card A interrupt-mode select 0 = Host IRQ is a positive-edge ISA-style interrupt. 1 = Host IRQ is a open-drain active-low PCI-style interrupt.
2	XWBCSC	R/W	Explicit writeback of card status-change interrupt acknowledge 0 = CSC interrupts are cleared by read of the card status-change register. 1 = CSC interrupts are cleared by explicit writeback of 1 to status flags in the card status-change register.
1	CSCLM	R/W	Card status-change interrupt-mode select 0 = Host IRQ is a positive-edge ISA-style interrupt. 1 = Host IRQ is a open-drain active-low PCI-style interrupt.
0	PWRDN	R/W	Chip power down 0 = Normal operation 1 = Power down enabled

card detect and general control register

PCI addresses (hex): Socket A: 96, Socket B: D6

ExCA offset (hex): Socket A: 16, Socket B: 56

Bit	7	6	5	4	3	2	1	0
Name	VS2	VS1	—	—	—	—	CONFRES	—
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	VS2	R	Voltage sense 2
6	VS1	R	Voltage sense 1 Indicates the state of PC Card input VS1
5–2	—	R/W	Reserved
1	CONFRES	R/W	Configuration reset enable 0 = Normal operation 1 = Reset configuration registers for slot when $\overline{CD1}$ and $\overline{CD2}$ go high
0	—	R/W	Reserved

Interrupt registers

interrupt and general control register

PCI addresses (hex): Socket A: 83, Socket B: C3 — ExCA offset (hex): Socket A: 03, Socket B: 43

This read/write interrupt and general control register controls the interrupt steering for the PC Card I/O interrupt as well as general control of the PCI1050.

Bit	7	6	5	4	3	2	1	0
Name	—	CRESET	CTYPE	—	CINT3	CINT2	CINT1	CINT0
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	CRIEN	R/W	Card ring indicate enable. Enables ring indicate function of PCI1050. CRIEN is encoded as: 0 = Ring indicate disabled (default) 1 = Rind indicate enabled
6	CRESET	R/W	PC Card reset. This is a software reset to the PC Card. 0 = Drive card RESET high 1 = Drive card RESET low
5	CTYPE	R/W	PC Card type (memory card or I/O card) 0 = Selects a memory PC Card 1 = Selects an I/O PC Card and enables the PC Card interface multiplexer for routing of PC Card I/O signals
4	INTR	R/W	PCI interrupt-CSC routing enable bit. When INTR is set (1) and the PCI interrupt bit in the device control register is enabled, the card status-change interrupts are routed to the PCI interrupt for the socket ($\overline{INTA}$ or $\overline{INTB}$). When low, the card status-change interrupts are routed using bits 7–4 in the ExCA card status-change interrupt configuration register. To use PCI interrupt-CSC routing, the ISA IRQ signaling method must be enabled (bits 2–1 of the device control register must not be 0). INTR is encoded as: 0 = CSC interrupts routed by ExCA registers (default) 1 = CSC interrupts routed to PCI interrupts
3–0	CINT3–CINT0	R/W	This field selects the routing for PC Card I/O interrupts.

CINT3	CINT2	CINT1	CINT0	Level
0	0	0	0	IRQ not selected
0	0	0	1	Reserved
0	0	1	0	Reserved
0	0	1	1	IRQ3 enabled
0	1	0	0	IRQ4 enabled
0	1	0	1	IRQ5 enabled
0	1	1	0	Reserved
0	1	1	1	IRQ7 enabled
1	0	0	0	Reserved
1	0	0	1	IRQ9 enabled
1	0	1	0	IRQ10 enabled
1	0	1	1	IRQ11 enabled
1	1	0	0	IRQ12 enabled
1	1	0	1	Reserved
1	1	1	0	IRQ14 enabled
1	1	1	1	IRQ15 enabled

card status-change interrupt configuration register

PCI addresses (hex): Socket A: 85 Socket B: C5 ExCA offset (hex): Socket A: 05 Socket B: 45

This register controls interrupt steering of the card status-change interrupt and the card status-change interrupt enables.

Bit	7	6	5	4	3	2	1	0
Name	SINT3	SINT2	SINT1	SINT0	CDEN	RDYEN	BWRNEN	BDEADEN
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–4	SINT3–SINT0	R/W	This field selects the routing for CSC interrupts. This field is ignored if INTR in the interrupt and general control register is set to 1. SINT3 SINT2 SINT1 SINT0 Level 0 0 0 0 IRQ not selected 0 0 0 1 Reserved 0 0 1 0 Reserved 0 0 1 1 IRQ3 enabled 0 1 0 0 IRQ4 enabled 0 1 0 1 IRQ5 enabled 0 1 1 0 Reserved 0 1 1 1 IRQ7 enabled 1 0 0 0 Reserved 1 0 0 1 IRQ9 enabled 1 0 1 0 IRQ10 enabled 1 0 1 1 IRQ11 enabled 1 1 0 0 IRQ12 enabled 1 1 0 1 Reserved 1 1 1 0 IRQ14 enabled 1 1 1 1 IRQ15 enabled
3	CDEN	R/W	Card detect enable 0 = Disables the generation of a card status-change interrupt when the card-detect signals change state 1 = Enables a card status-change interrupt when a change is detected on the $\overline{CD1}$ or $\overline{CD2}$ signals
2	RDYEN	R/W	Ready enable for memory PC Cards. RDYEN is ignored when the interface is configured for I/O PC Cards. 0 = Disables the generation of a card status-change interrupt when a low-to-high transition is detected on the RDY/$\overline{BSY}$ signal 1 = Enables a card status-change interrupt when a low-to-high transition is detected on the RDY/$\overline{BSY}$ signal
1	BWRNEN	R/W	Battery warning enable for memory PC Cards. BWRNEN is ignored when the interface is configured for I/O PC Cards. 0 = Disables the generation of a card status-change interrupt when a battery warning condition is detected 1 = Enables a card status-change interrupt when a battery warning condition is detected
0	BDEADEN	R/W	Battery dead enable ($\overline{STSCHG}$) 0 = Disables the generation of a card status-change interrupt 1 = Enables a card status-change interrupt when a battery dead condition is detected

I/O window registers

I/O window control register

PCI addresses (hex): Socket A: 87 Socket B: C7 ExCA offset (hex): Socket A: 07 Socket B: 47

This register configures I/O window 0 and I/O window 1.

Bit	7	6	5	4	3	2	1	0
Name	IW1WS	IW1ZWS	IW1ADS	IW1DS	IW0WS	IW0ZWS	IW0ADS	IW0DS
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	IW1WS	R/W	Window 1 wait state. The PCI1050 emulates the ISA wait-state mechanism used by the Intel 82365SL-DF to control PC Card cycle timing. 0 = The 16-bit and 8-bit cycles have standard length. 1 = The 16-bit cycles are extended by equivalent of one ISA wait state; the 8-bit cycles are unchanged.
6	IW1ZWS	R/W	Window 1 zero wait state. The PCI1050 emulates the ISA wait-state mechanism used by the Intel 82365SL-DF to control PC Card cycle timing. 0 = The 16-bit and 8-bit cycles have standard length. 1 = The 8-bit cycles are reduced to equivalent of three ISA clock cycles; the 16-bit cycles are unchanged.
5	IW1ADS	R/W	I/O window 1 auto data size 0 = Window data width is determined by bit IW1DS. 1 = Window data width is determined by input $\overline{\text{IOIS16}}$ from the PC Card.
4	IW1DS	R/W	I/O window 1 data size. Bit 4 is ignored if IW1ADS is set. 0 = Window data width is 8 bits. 1 = Window data width is 16 bits.
3	IW0WS	R/W	Window 0 wait state. The PCI1050 emulates the ISA wait-state mechanism used by the Intel 82365SL-DF to control PC Card cycle timing. 0 = The 16-bit and 8-bit cycles have standard length. 1 = The 16-bit cycles are extended by equivalent of one ISA wait state; the 8-bit cycles are unchanged.
2	IW0ZWS	R/W	Window 0 zero wait state. The PCI1050 emulates the ISA wait-state mechanism used by the Intel 82365SL-DF to control PC Card cycle timing. 0 = The 16-bit and 8-bit I/O cycles have standard length. 1 = The 8-bit I/O cycles are reduced to equivalent of three ISA clock cycles; the 16-bit I/O cycles are unchanged.
1	IW0ADS	R/W	I/O window 0 auto data size 0 = Window data width is determined by bit IW0DS. 1 = Window data width is determined by input $\overline{\text{IOIS16}}$ from the PC Card.
0	IW0DS	R/W	I/O window 0 data size. Bit 0 is ignored if IW0ADS is set. 0 = Window data width is 8 bits. 1 = Window data width is 16 bits.

I/O window 0 start-address low-byte register

PCI addresses (hex): Socket A: 88, Socket B: C8 — ExCA offset (hex): Socket A: 08, Socket B: 48

This register contains the low-order address bits used to determine the start address of I/O address window 0. This provides a minimum 1-byte window for I/O address window 0.

Bit	7	6	5	4	3	2	1	0
Name	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	SA7–SA0	R/W	I/O window 0 start address A7–A0

I/O window 0 start-address high-byte register

PCI addresses (hex): Socket A: 89, Socket B: C9 — ExCA offset (hex): Socket A: 09, Socket B: 49

This register contains the high-order address bits used to determine the start address of I/O address window 0.

Bit	7	6	5	4	3	2	1	0
Name	SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	SA15–SA8	R/W	I/O window 0 start address A15–A8

I/O window 0 end-address low-byte register

PCI addresses (hex): Socket A: 8A, Socket B: CA — ExCA offset (hex): Socket A: 0A, Socket B: 4A

This register contains the low-order address bits used to determine the end address of I/O address window 0. This provides a minimum 1-byte window for I/O address window 0.

Bit	7	6	5	4	3	2	1	0
Name	EA7	EA6	EA5	EA4	EA3	EA2	EA1	EA0
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	EA7–EA0	R/W	I/O window 0 end address A7–A0

I/O window 0 end-address high-byte register

PCI addresses (hex): Socket A: 8B, Socket B: CB — ExCA offset (hex): Socket A: 0B, Socket B: 4B

This register contains the high-order address bits used to determine the end address of I/O address window 0.

Bit	7	6	5	4	3	2	1	0
Name	EA15	EA14	EA13	EA12	EA11	EA10	EA9	EA8
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	EA15–EA8	R/W	I/O window 0 end address A15–A8

I/O window 1 configuration registers

System I/O window 1 register functions duplicate I/O window 0. The addresses of each of these registers are shown below.

I/O window 1 start-address low-byte register

PCI addresses (hex): Socket A: 8C, Socket B: CC — ExCA offset (hex): Socket A: 0C, Socket B: 4C

I/O window 1 start-address high-byte register

PCI addresses (hex): Socket A: 8D, Socket B: CD — ExCA offset (hex): Socket A: 0D, Socket B: 4D

I/O window 1 end-address low-byte register

PCI addresses (hex): Socket A: 8E, Socket B: CE — ExCA offset (hex): Socket A: 0E, Socket B: 4E

I/O window 1 end-address high-byte register

PCI addresses (hex): Socket A: 8F, Socket B: CF — ExCA offset (hex): Socket A: 0F, Socket B: 4F

memory window registers

memory window 0 start-address low-byte register

PCI addresses (hex): Socket A: 90, Socket B: D0 — ExCA offset (hex): Socket A: 10, Socket B: 50

This register contains the low-order address bits used to determine the start address of the corresponding system memory address mapping window. This provides a minimum memory-mapping window of 4K bytes.

A memory PC Card is selected when the following conditions are satisfied:

- Memory window is enabled.
- PCI address bits A23–A12 are greater than or equal to the memory window start address.
- PCI address bits A23–A12 are less than or equal to the memory window end address.
- PCI address bits AD31–AD24 are equal to the memory window page register value (default is 0).

The system memory address mapping windows can be configured by software to be used independently, or used together to perform mapping for special memory-mapping requirements such as LIM/EMS (Lotus®-Intel™-Microsoft®/extended memory specification) or XIP (execute in place).

Bit	7	6	5	4	3	2	1	0
Name	SA19	SA18	SA17	SA16	SA15	SA14	SA13	SA12
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	SA19–SA12	R/W	System memory window start address A19–A12

memory window 0 start-address high-byte register

PCI addresses (hex): Socket A: 91, Socket B: D1 — ExCA offset (hex): Socket A: 11, Socket B: 51

This register contains the high-order address bits used to determine the start address of the corresponding system memory address mapping window. Each system memory window has a datapath size associated with it that is controlled by a bit in this register.

Bit	7	6	5	4	3	2	1	0
Name	DSIZE	ZWS	SCRATCH	SCRATCH	SA23	SA22	SA21	SA20
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	DSIZE	R/W	Memory window data size 0 = Window data width is 8 bits. 1 = Window data width is 16 bits.
6	ZWS	R/W	Window 0 zero wait state. The PCI1050 emulates the ISA wait-state mechanism used by the Intel 82365SL-DF to control PC Card cycle timing. 0 = The 16-bit and 8-bit memory cycles have standard length. 1 = The 8-bit memory cycles are reduced to equivalent of three ISA clock cycles; the 16-bit memory cycles are reduced to equivalent of two ISA clock cycles.
5–4	SCRATCH	R/W	Scratch bits. General-purpose storage and retrieval.
3–0	SA23–SA20	R/W	System-memory window start address A23–A20

memory window 0 end-address low-byte register

PCI addresses (hex): Socket A: 92, Socket B: D2 ExCA offset (hex): Socket A: 12, Socket B: 52

This register contains the low-order address bits used to determine the end address of the corresponding system memory address mapping window. This provides a minimum memory-mapping window of 4K bytes.

Bit	7	6	5	4	3	2	1	0
Name	EA19	EA18	EA17	EA16	EA15	EA14	EA13	EA12
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	EA19–EA12	R/W	System memory window end address A19–A12

memory window 0 end-address high-byte register

PCI addresses (hex): Socket A: 93, Socket B: D3 ExCA offset (hex): Socket A: 13, Socket B: 53

This register contains the high-order address bits used to determine the end address of the corresponding system memory address mapping window.

Bit	7	6	5	4	3	2	1	0
Name	WS1	WS0	—	—	EA23	EA22	EA21	EA20
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–6	WS1–WS0	R/W	Window wait state. The PCI1050 emulates the ISA wait-state mechanism used by the Intel 82365SL-DF to control PC Card cycle timing. WS1 and WS0 determine the number of equivalent ISA wait states added to 16-bit memory cycles (8-bit memory cycles are unchanged).
5–4	—	R/W	Reserved
3–0	EA23–EA20	R/W	System memory window end address A23–A20

memory window 0 offset-address low-byte register

PCI addresses (hex): Socket A: 94, Socket B: D4 ExCA offset (hex): Socket A: 14, Socket B: 54

This register contains the low-order address bits that are added to the system address bits A19–A12 to generate the memory address for the PC Card.

Bit	7	6	5	4	3	2	1	0
Name	OF19	OF18	OF17	OF16	OF15	OF14	OF13	OF12
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	OF19–OF12	R/W	Card memory offset address A19–A12

memory window 0 offset-address high-byte register

PCI addresses (hex): Socket A: 95, Socket B: D5 — ExCA offset (hex): Socket A: 15, Socket B: 55

This register contains the high-order address bits that are added to the system address bits A23–A20 to generate the memory address for the PC Card. The software write protect of the PC Card memory for the corresponding system memory window is controlled by this register. This register also controls whether the corresponding system memory window is mapped to attribute or common memory in the PC Card.

Bit	7	6	5	4	3	2	1	0
Name	WP	REG	OF25	OF24	OF23	OF22	OF21	OF20
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7	WP	R/W	Write protect. Write operations to the PC Card through the corresponding system memory window are controlled by WP. 0 = Write operations allowed 1 = Write operations inhibited
6	REG	R/W	Register active. Accesses to the system memory are controlled by REG. 0 = Accesses common memory on the PC Card 1 = Accesses attribute memory on the PC Card
5–0	OF25–OF20	R/W	Card memory offset address A25–A20

memory windows 1–4 configuration registers

System memory windows 1–4 register functions duplicate memory window 0. The register addresses of each of these registers are shown below.

memory window 1 start-address low-byte register

PCI addresses (hex): Socket A: 98, Socket B: D8 — ExCA offset (hex): Socket A: 18, Socket B: 58

memory window 1 start-address high-byte register

PCI addresses (hex): Socket A: 99, Socket B: D9 — ExCA offset (hex): Socket A: 19, Socket B: 59

memory window 1 end-address low-byte register

PCI addresses (hex): Socket A: 9A, Socket B: DA — ExCA offset (hex): Socket A: 1A, Socket B: 5A

memory window 1 end-address high-byte register

PCI addresses (hex): Socket A: 9B, Socket B: DB — ExCA offset (hex): Socket A: 1B, Socket B: 5B

memory window 1 offset-address low-byte register

PCI addresses (hex): Socket A: 9C, Socket B: DC — ExCA offset (hex): Socket A: 1C, Socket B: 5C

memory window 1 offset-address high-byte register

PCI addresses (hex): Socket A: 9D, Socket B: DD — ExCA offset (hex): Socket A: 1D, Socket B: 5D

memory window 2 start-address low-byte register

PCI addresses (hex):	Socket A: A0 Socket B: E0	ExCA offset (hex):	Socket A: 20 Socket B: 60

memory window 2 start-address high-byte register

PCI addresses (hex):	Socket A: A1 Socket B: E1	ExCA offset (hex):	Socket A: 21 Socket B: 61

memory window 2 end-address low-byte register

PCI addresses (hex):	Socket A: A2 Socket B: E2	ExCA offset (hex):	Socket A: 22 Socket B: 62

memory window 2 end-address high-byte register

PCI addresses (hex):	Socket A: A3 Socket B: E3	ExCA offset (hex):	Socket A: 23 Socket B: 63

memory window 2 offset-address low-byte register

PCI addresses (hex):	Socket A: A4 Socket B: E4	ExCA offset (hex):	Socket A: 24 Socket B: 64

memory window 2 offset-address high-byte register

PCI addresses (hex):	Socket A: A5 Socket B: E5	ExCA offset (hex):	Socket A: 25 Socket B: 65

memory window 3 start-address low-byte register

PCI addresses (hex):	Socket A: A8 Socket B: E8	ExCA offset (hex):	Socket A: 28 Socket B: 68

memory window 3 start-address high-byte register

PCI addresses (hex):	Socket A: A9 Socket B: E9	ExCA offset (hex):	Socket A: 29 Socket B: 69

memory window 3 end-address low-byte register

PCI addresses (hex):	Socket A: AA Socket B: EA	ExCA offset (hex):	Socket A: 2A Socket B: 6A

memory window 3 end-address high-byte register

PCI addresses (hex):	Socket A: AB Socket B: EB	ExCA offset (hex):	Socket A: 2B Socket B: 6B

memory window 3 offset-address low-byte register

PCI addresses (hex):	Socket A: AC Socket B: EC	ExCA offset (hex):	Socket A: 2C Socket B: 6C

memory window 3 offset-address high-byte register

PCI addresses (hex):	Socket A: AD Socket B: ED	ExCA offset (hex):	Socket A: 2D Socket B: 6D

memory window 4 start-address low-byte register

PCI addresses (hex):	Socket A: B0 Socket B: F0	ExCA offset (hex):	Socket A: 30 Socket B: 70

memory window 4 start-address high-byte register

PCI addresses (hex):	Socket A: B1 Socket B: F1	ExCA offset (hex):	Socket A: 31 Socket B: 71

memory window 4 end-address low-byte register

PCI addresses (hex):	Socket A: B2 Socket B: F2	ExCA offset (hex):	Socket A: 32 Socket B: 72

memory window 4 end-address high-byte register

PCI addresses (hex):	Socket A: B3 Socket B: F3	ExCA offset (hex):	Socket A: 33 Socket B: 73

memory window 4 offset-address low-byte register

PCI addresses (hex):	Socket A: B4 Socket B: F4	ExCA offset (hex):	Socket A: 34 Socket B: 74

memory window 4 offset-address high-byte register

PCI addresses (hex):	Socket A: B5 Socket B: F5	ExCA offset (hex):	Socket A: 35 Socket B: 75

PCI configuration headers

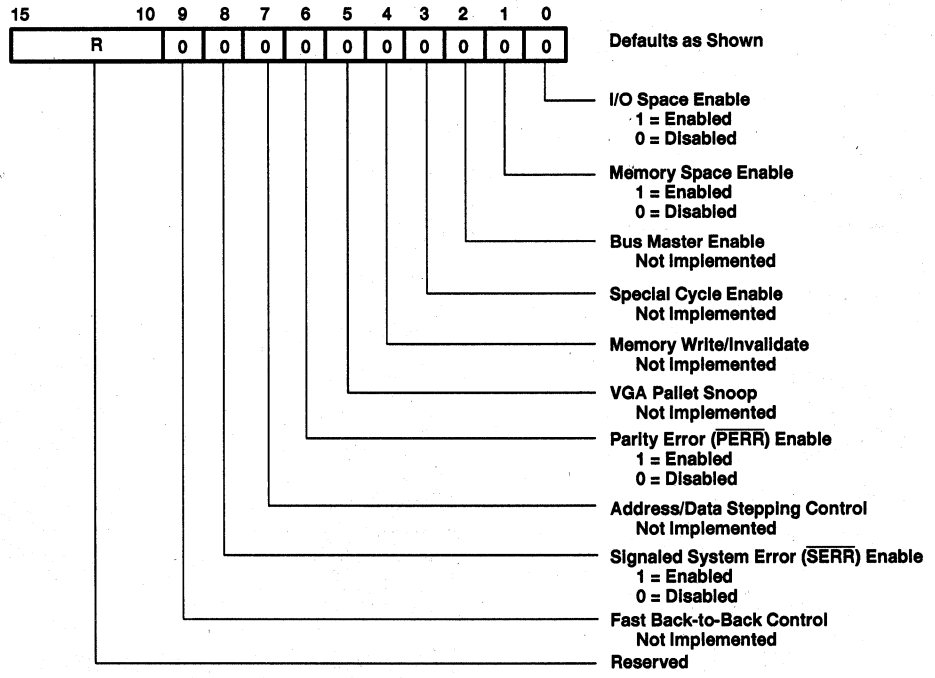
The PCI configuration headers and configuration registers are listed in Table 6.

Table 6. PCI Configuration Headers

NAME	PCI ADDRESS
Device identification 1 register	00h
Command register	04h
Device identification 2 register	08h
Miscellaneous function 1 register	0Ch
Base address registers 0–5	10h, 14h, 18h, 1Ch, 20h, 24h
Expansion ROM base address register	30h
Miscellaneous function 2 register	3Ch
PCI header reserved registers	28h, 2Ch, 34h, 38h

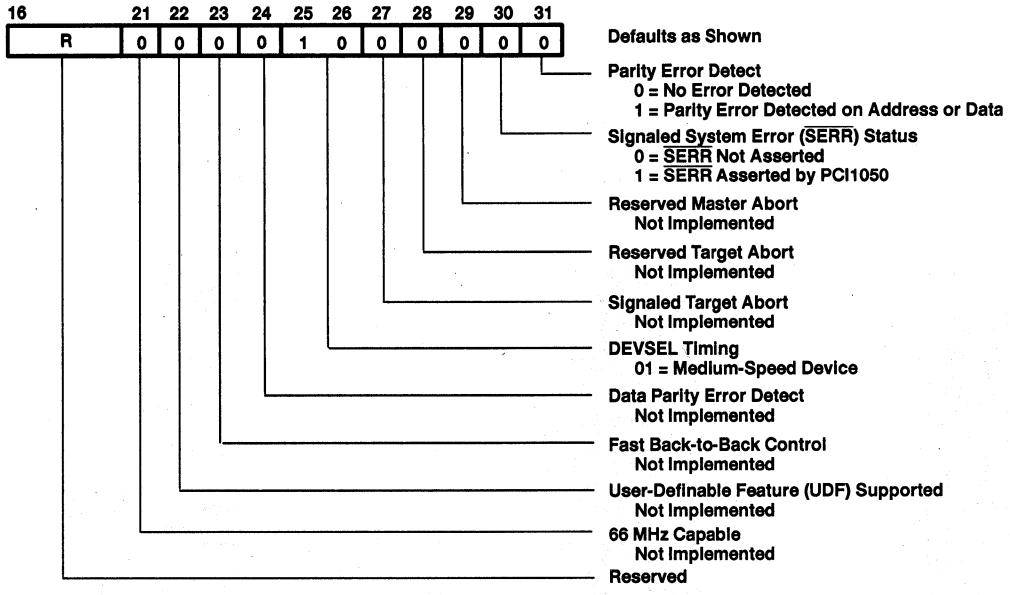
configuration header

The PCI supports the PCI-defined 64-byte header. Reads from registers that are reserved or that are not implemented return to 0.



PCI Command Register

configuration header (continued)



PCI Status Register

command/status register

PCI address (hex): 04

This 32-bit register contains the status and command fields.

Bit	31	30	29	28	27	26	25	24
Name	Status							
Default	0	0	0	0	0	0	0	0
Bit	23	22	21	20	19	18	17	16
Name	Status							
Default	0	0	0	0	0	0	0	0

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

BIT	NAME	ACCESS	DESCRIPTION
31	PED	R/W	Parity error detected 0 = No parity error detected 1 = Parity error detected on address or data
30	SER	R/W	Signaled system error ($\overline{\text{SERR}}$) status 0 = The PCI1050 does not generate a system error on the $\overline{\text{SERR}}$ line. 1 = The PCI1050 generates a system error on the $\overline{\text{SERR}}$ line.
29–28	—	R/W	Reserved
27	STA	R/W	Signaled target abort sequence 0 = Default 1 = PCI transaction terminates with a target abort.
26–25	DEV	R	$\overline{\text{DEVSEL}}$ timing Returns 01 indicating that PCI1050 is a medium-speed device for any bus command
24–9	—	R/W	Reserved
8	ENS	R/W	Signaled system error ($\overline{\text{SERR}}$) enable. ENS controls the PCI1050 response to parity errors that occur during an address cycle. Upon reset, default is 0. 0 = PCI1050 ignores PCI data parity errors. 1 = PCI1050 asserts $\overline{\text{SERR}}$ when a parity error occurs during an address cycle.
7	—	R/W	Address/data stepping control. This function is not implemented on the PCI1050; therefore, bit 7 is always read as 0.
6	ENP	R/W	Parity error ($\overline{\text{PERR}}$) enable. ENP controls the PCI1050 response to PCI data parity errors. Upon reset, default is 0, and parity checking is disabled. 0 = PCI1050 ignores PCI data parity errors. 1 = PCI1050 asserts $\overline{\text{PERR}}$ when a PCI data parity error occurs.
5–2	—	R/W	Bus master enable, special cycle enable, memory write/invalidate enable, and VGA pallet snoop enable. These functions are not implemented on the PCI1050; therefore, bits 5–2 are always read as 0.
1	ENM	R/W	Memory access enable. ENM allows PCI1050 to accept PCI-originated memory cycles. 0 = Memory access disabled. PCI1050 does not respond to PCI master memory cycles; $\overline{\text{DEVSEL}}$ is disabled during memory cycles. 1 = Memory access enabled
0	ENI	R/W	I/O access enable. ENI allows PCI1050 to accept PCI-originated I/O cycles. 0 = I/O access disabled. PCI1050 does not respond to PCI master I/O cycles; $\overline{\text{DEVSEL}}$ is disabled during I/O cycles. 1 = I/O access enabled

device identification 1 register

PCI address (hex): 00

This read-only 32-bit register contains the device and vendor ID that is required to uniquely identify a PCI device.

Bit	31	30	29	28	27	26	25	24
Name	Device ID							
Default	1	0	1	0	1	1	0	0
Bit	23	22	21	20	19	18	17	16
Name	Device ID							
Default	0	0	0	1	0	0	0	0

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

BIT	NAME	ACCESS	DESCRIPTION
31–16	Device ID	R	This 16-bit value is the device ID for the PCI1050, which is AC10h.
15–0	Vendor ID	R	This 16-bit value is the vendor ID assigned to TI, which is 104Ch.

device identification 2 register

PCI address (hex): 08

This 32-bit register contains the class code and revision ID fields.

Bit	31	30	29	28	27	26	25	24
Name	Class Code							
Default	0	0	0	0	0	1	1	0
Bit	23	22	21	20	19	18	17	16
Name	Class Code							
Default	0	0	0	0	0	1	0	1
Bit	15	14	13	12	11	10	9	8
Name	Class Code							
Default	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Name	Revision ID							
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
31–8	Class Code	R	Class code is <Base class> and <Sub class> and <Prog. interface>. PC Card bridge class code is 060500h.
7–0	Revision ID	R	First silicon revision ID is 00h.

miscellaneous function 1 register

PCI address (hex): 0C

This 32-bit register contains the BIST, header type, latency type, and cache line fields.

Bit	31	30	29	28	27	26	25	24
Name	BIST							
Default	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Name	Header Type							
Default	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Name	Latency Type							
Default	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Name	Cache Line							
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
31–24	BIST	R	0 = No built-in self test
23–16	Header Type	R	0 = Single function
15–8	Latency Type	R	0 = Target only
7–0	Cache Line	R	0 = Target only

base-address registers 0–5

PCI addresses (hex): 10, 14, 18, 1C, 20, 24

Base-address register 0 determines the I/O address of the index/data register pairs used to access the Intel 82365SL-DF-compatible registers. The base register is programmed with the address of the index register. The data register is mapped at the next higher byte address. Base-address registers 1–5 are reserved.

Bit	31	30	29	28	27	26	25	24
Name	Base Address							
Default	0	0	0	0	0	0	0	0
Bit	23	22	21	20	19	18	17	16
Name	Base Address							
Default	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Name	Base Address							
Default	0	0	0	0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
Name	Base Address							
Default	0	0	0	0	0	0	0	1

BIT	NAME	ACCESS	DESCRIPTION
31–2	Base Address	R/W	Base registers
1	—	R	Reserved
0	—	R	I/O indicator bit. Hardwired to 1.

expansion ROM base address register

PCI address (hex): 30PCI address (hex):30

PCI provides this 32-bit register to allow software remapping of device-expansion ROM. The PCI1050 does not implement this feature.

Bit	31	30	29	28	27	26	25	24
Name	Expansion ROM Base Address							
Default	0	0	0	0	0	0	0	0
Bit	23	22	21	20	19	18	17	16
Name	Expansion ROM Base Address							
Default	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Name	Expansion ROM Base Address							
Default	0	0	0	0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
Name	Expansion ROM Base Address							
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
31–0	Expansion ROM Base Address	R	Not implemented

miscellaneous function 2 register

PCI address (hex): 3C

This 32-bit register contains the MAX_LAT, MAX_GNT, INT PIN, and INT LINE fields.

Bit	31	30	29	28	27	26	25	24
Name	MAX_LAT							
Default	0	0	0	0	0	0	0	0
Bit	23	22	21	20	19	18	17	16
Name	MAX_GNT							
Default	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Name	INT PIN							
Default	0	0	0	0	0	0	0	1
Bit	7	6	5	4	3	2	1	0
Name	INT LINE							
Default	1	1	1	1	1	1	1	1

BIT	NAME	ACCESS	DESCRIPTION
31–24	MAX_LAT	R	0 = Target only
23–16	MAX_GNT	R	0 = Target only
15–8	INT PIN	R	Interrupt routing is programmable in the interrupt register. Hardwired to 1.
7–0	INT LINE	R/W	Used to communicate interrupt line routing but does not affect device function. Field is written to by the host software after resource allocation and is available to be read by device drivers and operating systems.

PCI header reserved registers

PCI addresses (hex): 28, 2C, 34, 38

These 32-bit registers are defined by PCI as reserved and are read only with a hardwired value of 0.

Bit	31	30	29	28	27	26	25	24
Name	—	—	—	—	—	—	—	—
Default	0	0	0	0	0	0	0	0
Bit	23	22	21	20	19	18	17	16
Name	—	—	—	—	—	—	—	—
Default	0	0	0	0	0	0	0	0

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

TI extension registers

The TI extension registers are accessible only in PCI configuration space and are used to control features not found in the Intel 82365SL-DF. The six registers are listed in Table 7.

Table 7. TI Extension Registers

NAME	PCI ADDRESS	
	SOCKET A	SOCKET B
Initialization register	40h	44h
Write buffer control register	41h	45h
Miscellaneous register	42h	46h
Memory window page register	43h	47h
Interrupt-mode register 1	48h	
Interrupt-mode register 2	49h	

initialization register

PCI addresses (hex): Socket A: 40
Socket B: 44

This register controls device I/O addressing and software reset.

Bit	7	6	5	4	3	2	1	0
Name	—	—	TS1	TS0	—	DEVID	SRES	—
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–6	—	R/W	Reserved
5–4	TS1–TS0	R/W	PCI clock frequency. Bits 5–4 are programmed at power up to indicate the frequency of the PCI clock. The PC Card cycle generator uses PCI clock for waveform timing and needs to know the frequency. **TS1 TS0 PCI Clock Frequency** 0 0 25 MHz 0 1 33 MHz 1 0 Reserved 1 1 Reserved
3	—	R/W	Reserved
2	DEVID	R/W	Device number 0 = Valid index range is 00h to 3Fh for socket A and 40h to 7Fh for socket B. 1 = Valid index range is 80h to BFh for socket A and C0h to FFh for socket B.
1	SRES	R/W	Soft reset 0 = Normal operation 1 = Reset PCI1050
0	—	R/W	Reserved

write buffer control register

PCI addresses (hex): Socket A: 41
Socket B: 45

This register controls the internal write buffer.

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	IOBUF	FDEP	FEN	FULL	EMPTY
Default	0	0	0	0	0	0	0	1

BIT	NAME	ACCESS	DESCRIPTION
7–5	—	R/W	Reserved
4	IOBUF	R/W	Write buffer cycle select 0 = Memory writes only to write buffer 1 = Memory and I/O writes to write buffer
3	FDEP	R/W	Write buffer depth 0 = Four deep 1 = One deep
2	FEN	R/W	Write buffer enable/disable 0 = Write buffer off 1 = Write buffer on
1	FULL	R	Write buffer full 0 = Write buffer not full 1 = Write buffer full
0	EMPTY	R/W	When read, EMPTY indicates write buffer status. Read 0 = Write buffer not empty Read 1 = Write buffer empty When written to, EMPTY allows software to flush the write buffer. Write 0 = No change Write 1 = Flush write buffer

miscellaneous register

PCI addresses (hex): Socket A: 42
Socket B: 46

This register controls socket PC Card ring indicate, speaker, and card-voltage detection.

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	ATAEN	—	—	SPKEN	BIFG
Default	0	0	0	0	0	0	1	0

BIT	NAME	ACCESS	DESCRIPTION
7–5	—	R/W	Reserved
4	ATAEN	R/W	ATA special feature enable 0 = Normal operation 1 = I/O window addresses 3F7h and 377h are read only. Input FDC_D7 is routed to AD31 during reads from I/O 3F7h and 377h.
3–2	—	R/W	Reserved
1	SPKEN	R/W	Speaker-to-speaker out enable 0 = $\overline{\text{SPKR}}$ routing to $\overline{\text{SPKROUT}}$ disabled 1 = $\overline{\text{SPKR}}$ routing to $\overline{\text{SPKROUT}}$ enabled
0	BIFG	R/W	Card-interrupt flag. BIFG is set when the card asserts IREQ. Writing a 1 to this bit clears the flag. 0 = Clear 1 = Card interrupt has occurred.

memory window page register

PCI addresses (hex): Socket A: 43
Socket B: 47

This register contains an 8-bit page number that is compared with PCI address signals AD31–AD24 during memory cycles. If the page bits P7–P0 match AD31–AD24, the PCI1050 memory-window-decode logic is enabled. This allows the memory windows to be located above the first 16M bytes of system address space, which is a limitation of the ISA bus. By using the page register, the programmer can locate the PC Card memory windows in any of the 256 separate 16M-byte pages that comprise the 4G bytes of PCI address space.

Bit	7	6	5	4	3	2	1	0
Name	P7	P6	P5	P4	P3	P2	P1	P0
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–0	P7–P0	R/W	Memory window page register

interrupt-mode register 1

PCI address (hex): 48

This register determines whether interrupts IRQ3/$\overline{\text{INTA}}$, IRQ4/$\overline{\text{INTB}}$, IRQ5/$\overline{\text{INTC}}$, and IRQ7/$\overline{\text{INTD}}$ are configured as totem-pole or open-drain outputs.

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	IRQ7	IRQ5	IRQ4	IRQ3
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–4	—	R/W	Reserved
3	IRQ7/$\overline{\text{INTD}}$	R/W	Bit 3 configures IRQ7/$\overline{\text{INTD}}$ as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
2	IRQ5/$\overline{\text{INTC}}$	R/W	Bit 2 configures IRQ5/$\overline{\text{INTC}}$ as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
1	IRQ4/$\overline{\text{INTB}}$	R/W	Bit 1 configures IRQ4/$\overline{\text{INTB}}$ as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
0	IRQ3/$\overline{\text{INTA}}$	R/W	Bit 0 configures IRQ3/$\overline{\text{INTA}}$ as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output

interrupt-mode register 2

PCI address (hex): 49

This register determines whether interrupts IRQ9, IRQ10, IRQ11, IRQ12, IRQ14, and IRQ15 are configured as totem-pole or open-drain outputs.

Bit	7	6	5	4	3	2	1	0
Name	—	—	IRQ15	IRQ14	IRQ12	IRQ11	IRQ10	IRQ9
Default	0	0	0	0	0	0	0	0

BIT	NAME	ACCESS	DESCRIPTION
7–6	—	R/W	Reserved
5	IRQ15	R/W	Bit 5 configures IRQ15 as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
4	IRQ14	R/W	Bit 4 configures IRQ14 as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
3	IRQ12	R/W	Bit 3 configures IRQ12 as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
2	IRQ11	R/W	Bit 2 configures IRQ11 as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
1	IRQ10	R/W	Bit 1 configures IRQ10 as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output
0	IRQ9	R/W	Bit 0 configures IRQ9 as an open-drain output for connection to PCI bus interrupts. 0 = Open-drain PCI-type interrupt 1 = Totem-pole ISA-type output

absolute maximum ratings over recommended operating free-air temperature range (unless otherwise noted)†

Supply voltage range, V_{CC} ..... –0.5 V to 6 V
Input voltage range, V_I: Standard ..... –0.5 V to V_{CC} + 0.5 V
Fail safe ..... –0.5 V to 6.5 V
Output voltage range, V_O: Standard ..... –0.5 V to V_{CC} + 0.5 V
Fail safe ..... –0.5 V to 6.5 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 to external input and bidirectional buffers. $V_I > V_{CC}$ does not apply to fail-safe terminals.
2. Applies to external output and bidirectional buffers. $V_O > V_{CC}$ does not apply to fail-safe terminals.

recommended operating conditions

			MIN	NOM	MAX	UNIT
t_t	Input transition (rise and fall) time	CMOS compatible	0		25	ns
T_A	Operating free-air temperature	Commercial	0	25	70	°C
T_J‡	Virtual junction temperature	Commercial	0	25	115	°C

‡ These junction temperatures reflect simulation conditions. Customer is responsible for verifying junction temperature.

recommended operating conditions for PCI interface

			MIN	NOM	MAX	UNIT
V_{CCE}	Supply voltage for control interface	Commercial	4.75	5	5.25	V
V_{CORE}	Core voltage	Commercial	4.75	5	5.25	V
V_{CCP}	PCI supply voltage	Commercial	4.75	5	5.25	V
V_I	Input voltage		0		V_{CCP}	V
V_O§	Output voltage		0		V_{CCP}	V
V_{IH}¶	High-level input voltage	CMOS compatible	0.7V_{CCP}			V
V_{IL}¶	Low-level input voltage	CMOS compatible			0.2 V_{CCP}	V

§ Applies to external output buffers
¶ Applies to external input and bidirectional buffers without hysteresis

recommended operating conditions for PC Cards A and B and miscellaneous inputs and outputs

			OPERATION	MIN	NOM	MAX	UNIT
$V_{CC(A/B)}$	PC Card supply voltage	Commercial	3.3 V	3	3.3	3.6	V
			5 V	4.75	5	5.25	V
V_I	Input voltage		3.3 V	0		$V_{CC(A/B)}$	V
			5 V	0		$V_{CC(A/B)}$	V
V_O†	Output voltage		5 V	0		$V_{CC(A/B)}$	V
			3.3 V	0		$V_{CC(A/B)}$	V
V_{IH}‡	High-level input voltage	CMOS compatible	3.3 V	$0.7V_{CC(A/B)}$§			V
			5 V	$0.7V_{CC(A/B)}$§			V
V_{IL}‡	Low-level input voltage	CMOS compatible	3.3 V			$0.3V_{CC(A/B)}$§	V
			5 V			$0.2V_{CC(A/B)}$§	V

† Applies to external output buffers
‡ Applies to external input and bidirectional buffers without hysteresis
§ Meets TTL levels, V_{IH} MIN = 2 V and V_{IL} MAX = 0.8 V

electrical characteristics over recommended operating conditions (unless otherwise noted)

PARAMETER		SIDE	OPERATION	TEST CONDITIONS	MIN	MAX	UNIT
V_{OH}	High-level output voltage	PCI	5 V	I_{OH} = –2 mA	V_{CC}–0.55		V
		PC Card	3.3 V	I_{OH} = See Note 3	V_{CC}–0.55		
			5 V	I_{OH} = See Note 3	V_{CC}–0.8		
V_{OL}	Low-level output voltage	PCI	5 V	IOH = = 3 mA, 6 mA		0.5	V
		PC Card	3.3 V	I_{OL} = See Note 4		0.5	
			5 V	I_{OL} = See Note 4		0.5	
I_{OZ}	High-impedance output current			V_I = V_{CC} or GND¶		±10	µA
I_{IL}	Low-level input current			V_I = GND		–1	µA
I_{IH}	High-level input current			V_I = V_{CC}#		1	µA

NOTES: 3. I_{OH} = –0.9 mA for all PC Card outputs and bidirectionals, I_{OH} = –1.8 mA for all miscellaneous outputs.
4. I_{OL} = 1.62 mA for all PC Card outputs and bidirectionals, I_{OL}= 3.24 mA for all miscellaneous outputs.
¶ The 3-state or open-drain outputs must be in the high-impedance state.
# Applies to all inputs except TEST

PCI clock/reset timing requirements over recommended ranges of supply voltage and operating free-air temperature (see Figures 3, 4, and 5)

		ALTERNATE SYMBOL	MIN	MAX	UNIT
t_c	Cycle time, PCLK	t_{cyc}	30	∞	ns
t_{wH}	Pulse duration, PCLK high	t_{high}	12		ns
t_{wL}	Pulse duration, PCLK low	t_{low}	12		ns
Δv/Δt	Slew rate, PCLK	t_r, t_f	1	4	V/ns
t_w	Pulse duration, $\overline{RSTIN}$	t_{rst}	1		ms
t_{su}	Setup time, PCLK active at end of $\overline{RSTIN}$	$t_{rst\text{-}clk}$	100		µs

PCI timing requirements over recommended ranges of supply voltage and operating free-air temperature (see Figure 6)

			ALTERNATE SYMBOL	TEST CONDITIONS	MIN	MAX	UNIT
t_{pd}	Propagation delay time	PCLK to shared signal valid delay time (see Note 5)	t_{val}	C_L = 50 pF		11	ns
		PCLK to shared signal invalid delay time (see Note 5)	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, PCI valid before PCLK		t_{su}		7		ns
t_h	Hold time after PCLK high		t_h		0		ns

NOTE 5: PCI shared signals are AD31–AD0, C/$\overline{BE3}$–$\overline{BE0}$, $\overline{FRAME}$, $\overline{TRDY}$, $\overline{IRDY}$, $\overline{STOP}$, IDSEL, $\overline{DEVSEL}$, and PAR.

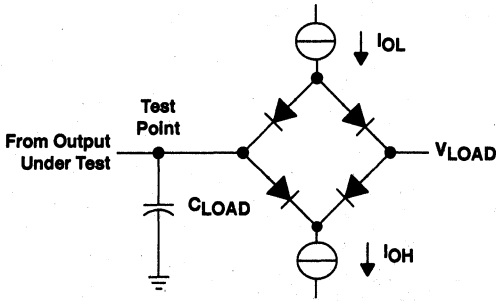
PARAMETER MEASUREMENT INFORMATION

LOAD CIRCUIT PARAMETERS

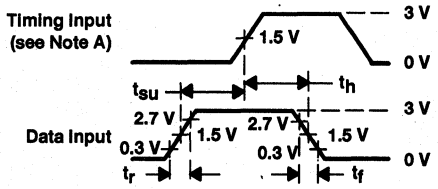
TIMING PARAMETER		C_{LOAD}† (pF)	I_{OL} (mA)	I_{OH} (mA)	V_{LOAD} (V)
t_{en}	t_{PZH}	50	8	−8	0
	t_{PZL}				3
t_{dis}	t_{PHZ}	50	8	−8	1.5
	t_{PLZ}				
t_{pd}		50	8	−8	‡

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

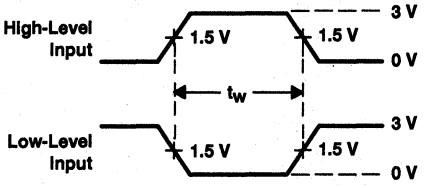
‡ $\frac{V_{LOAD} - V_{OL}}{I_{OL}}$ = 50 Ω, where V_{OL} = 0.6 V, I_{OL} = 8 mA



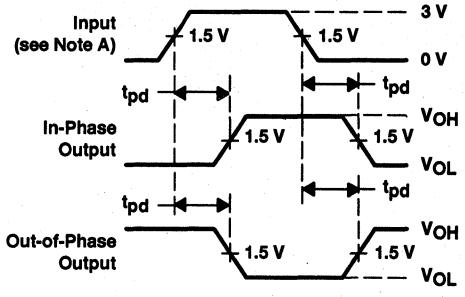
LOAD CIRCUIT



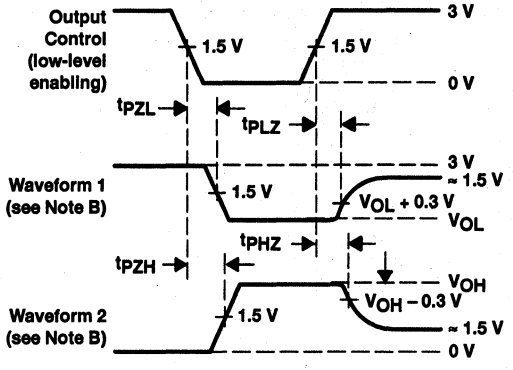
VOLTAGE WAVEFORMS
SETUP AND HOLD TIMES
INPUT RISE AND FALL TIMES



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, $t_r \le 6$ ns, $t_f \le 6$ 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 3. Load Circuit and Voltage Waveforms

PCIBUS PARAMETER MEASUREMENT INFORMATION

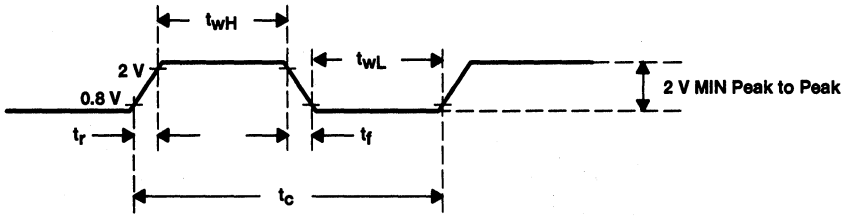


Figure 4. PCLK Timing Waveform

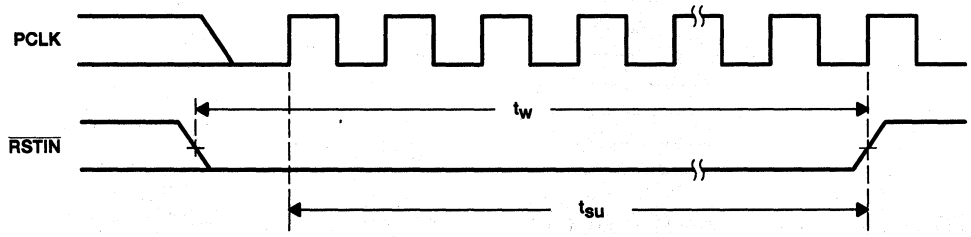


Figure 5. PCLK and $\overline{\text{RSTIN}}$ Timing Waveforms

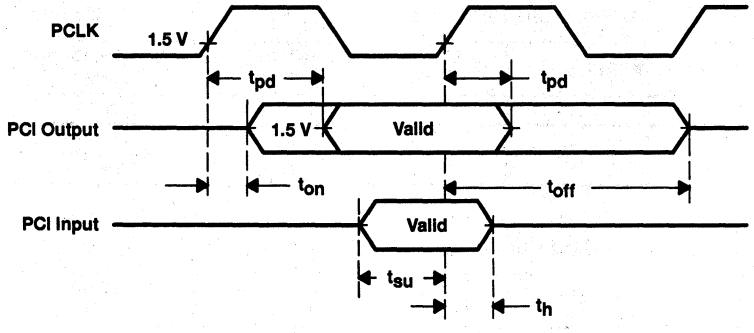


Figure 6. Shared-Signals Timing Waveforms

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. When the PCI1050 is used in systems with different PCI clock frequencies, the PC Card cycle generator must know the maximum PCI clock frequency to optimize the cycle timing. To communicate this information to the cycle generator, there are two additional register bits implemented in the TI initialization register. These bits (TS1 and TS0) should be programmed by software according to maximum PCI clock frequency as shown in Table 8. 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.

Table 8. PCI Clock Programming

TS1	TS0	MAXIMUM PCI CLOCK FREQUENCY (MHz)
0	0	25
0	1	33
1	0	50
1	1	Reserved

The PC Card address setup and hold times are functions of the wait-state bits and the PCI clock-frequency bits (TS1 and TS0). Table 9 shows address setup time in PCLK cycles and nanoseconds for I/O and memory cycles. Tables 10 and 11 show command active time in PCLK cycles and nanoseconds for I/O and memory cycles. Table 12 shows address hold time in PCLK cycles and nanoseconds for I/O and memory cycles.

Table 9. PC Card Address Setup Time, $t_{su(A)}$, 8-Bit and 16-Bit PCI Cycles

WAIT-STATE BITS			TS1–0 = 00 (PCLK/ns)	TS1–0 = 01 (PCLK/ns)	TS1–0 = 10 (PCLK/ns)
I/O			2/80	3/90	4/80
Memory	WS1	0	1/40	2/60	4/80
Memory	WS1	1	3/120	4/120	5/100

Table 10. PC Card Command Active Time, $t_{c(A)}$, 8-Bit PCI Cycles

WAIT-STATE BITS			TS1–0 = 00 (PCLK/ns)	TS1–0 = 01 (PCLK/ns)	TS1–0 = 10 (PCLK/ns)
I/O WS, ZWS	0	0	15/600	19/570	29/580
	1	X	18/720	23/690	35/700
	0	1	5/200	7/210	10/200
Memory WS1, WS0, ZWS	00	0	15/600	19/570	29/580
	01	X	18/720	23/690	35/700
	10	X	18/720	23/690	35/700
	11	X	18/720	23/690	35/700
	00	1	5/200	7/210	10/250

PC Card cycle timing (continued)

Table 11. PC Card Command Active Time, $t_{c(A)}$, 16-Bit PCI Cycles

WAIT-STATE BITS			TS1–0 = 00 (PCLK/ns)	TS1–0 = 01 (PCLK/ns)	TS1–0 = 10 (PCLK/ns)
I/O WS, ZWS	0	0	5/200	7/210	10/200
	1	X	8/320	11/330	16/320
	0	1	N/A	N/A	N/A
Memory WS1, WS0, ZWS	00	0	7/280	9/270	13/260
	01	X	10/400	13/390	19/380
	10	X	13/520	17/510	25/500
	11	X	16/640	21/630	32/640
	00	1	4/160	5/150	7/140

Table 12. PC Card Address Hold Time, $t_{h(A)}$, 8-Bit and 16-Bit PCI Cycles

WAIT-STATE BITS			TS1–0 = 00 (PCLK/ns)	TS1–0 = 01 (PCLK/ns)	TS1–0 = 10 (PCLK/ns)
I/O			1/40	2/60	2/50
Memory	WS1	0	1/40	2/60	2/50
Memory	WS1	1	2/80	3/90	4/80

timing requirements over recommended ranges of supply voltage and operating free-air temperature, memory cycle (for 100-ns common memory) (see Note 6, Figure 7, and Tables 9 and 10)

		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 6: These times are dependent on the register settings associated with ISA wait states and data size, and 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 cycle (see Figure 8 and Tables 9 and 10)

		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 9)

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

PC CARD PARAMETER MEASUREMENT INFORMATION

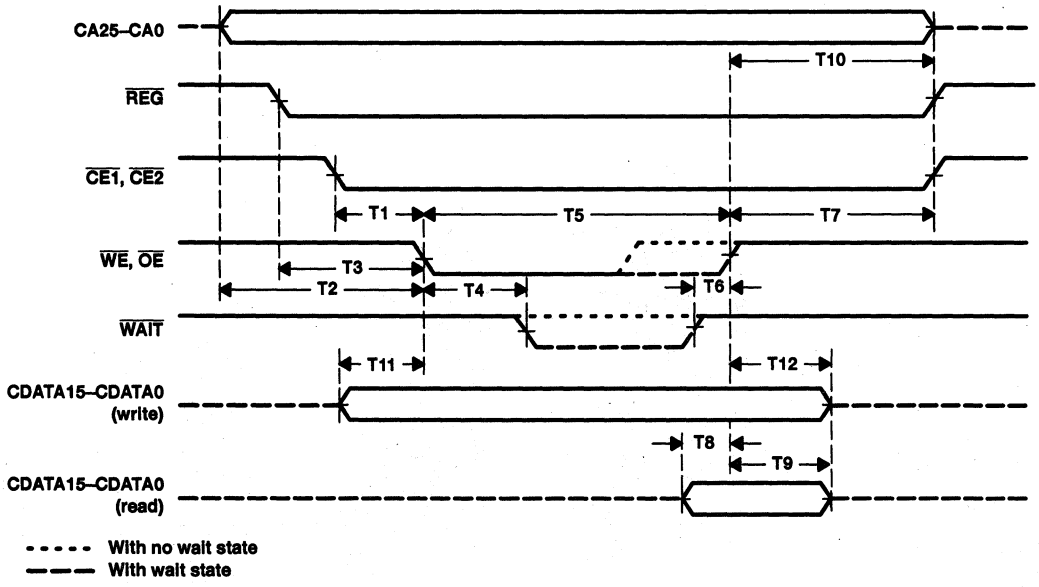


Figure 7. PC Card Memory Cycle

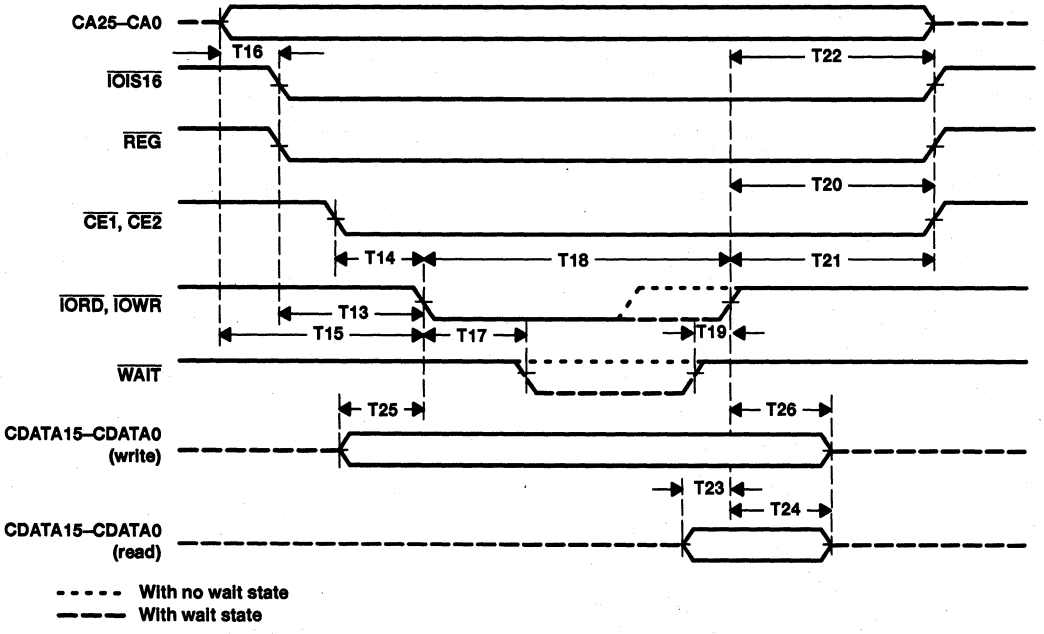


Figure 8. PC Card I/O Cycle

PC CARD PARAMETER MEASUREMENT INFORMATION

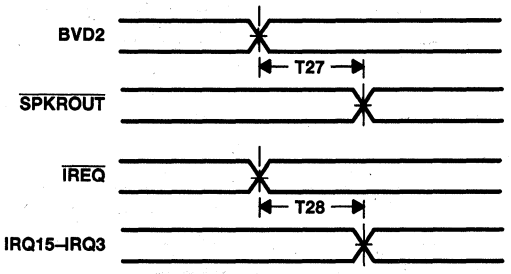


Figure 9. Miscellaneous PC Card Delay Times

SCPS008 – FEBRUARY 1996

- 3.3-V Core Logic With Universal PCI Interface Compatible With 3.3-V or 5-V PCI Signaling Environments
- Supports PCI Local Bus Specification 2.1
- Mix and Match 3.3-V/5-V PC Card16 Cards
- Supports Two PC Card™ Slots With Hot Insertion and Removal
- 1995 PC Card Standard Compliant
- Low-Power Advanced Submicron CMOS Technology
- Uses Serial Interface to Texas Instruments (TI) TPS2202A Dual Power Switch
- System Interrupts Can Be Programmed as PCI-Style or ISA IRQ-Style Interrupts
- ISA IRQ-Style Interrupts Can Be Serialized Onto a Single IRQSER Pin
- Five PCI Memory Windows and Two I/O Windows Available to Each PC Card16 Socket
- Exchangeable Card (ExCA™)-Compatible Registers Are Mapped In Memory and I/O Space
- TI Extension Registers Are Mapped in the PCI Configuration Space
- Intel™ 82365SL-DF Register Compatible
- Supports 16-Bit DMA on Both PC Card Sockets
- Supports Zoom Video Mode
- Supports Ring Indicate
- Packaged in 208-Pin Thin Plastic Quad Flatpack (TQFP)

Table of Contents

PRODUCT PREVIEW

ExCA and PC Card are trademarks of Personal Computer Memory Card International Association (PCMCIA).
Intel is a trademark of Intel Corp.

description

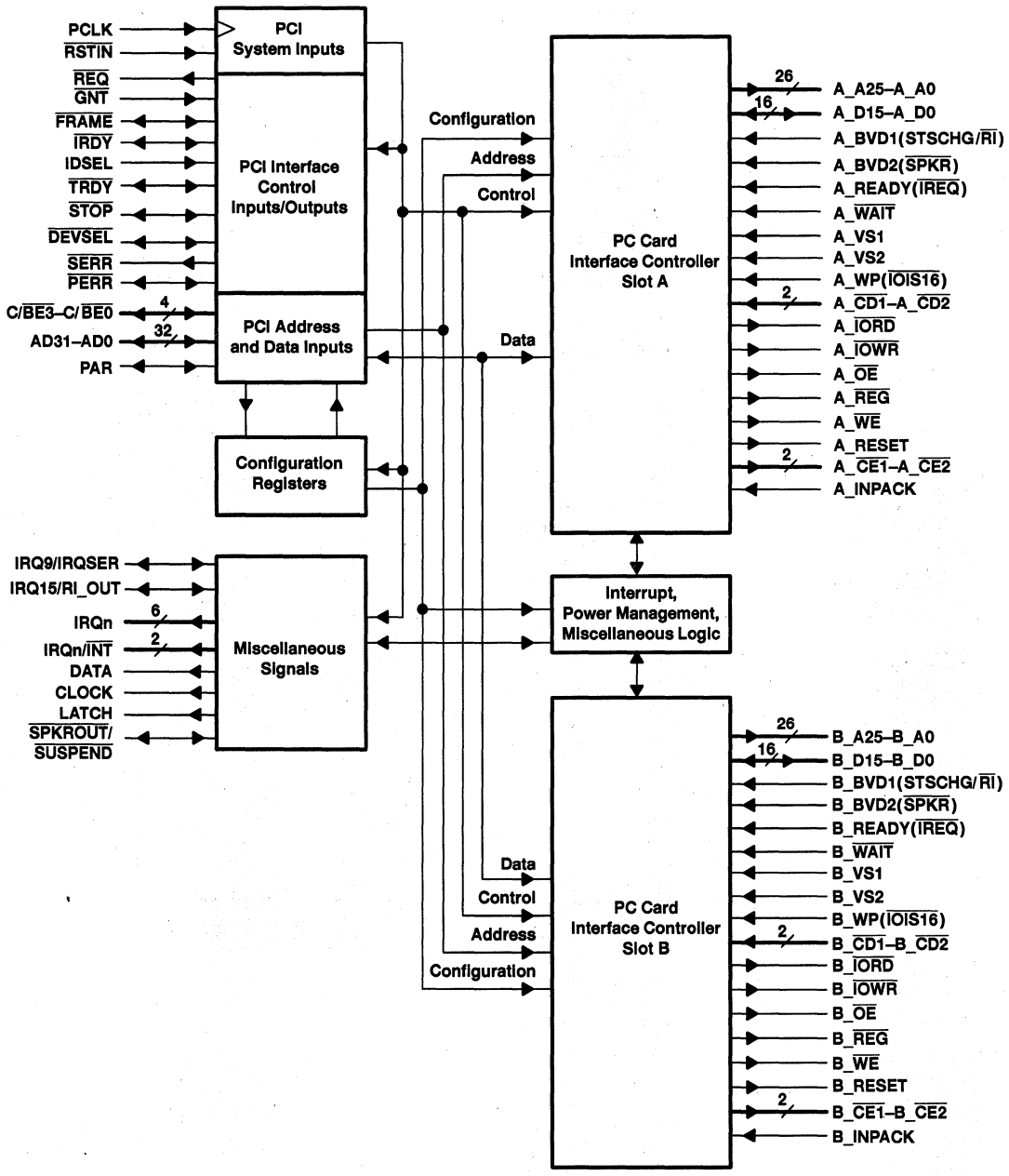
The TI PCI10XX is a high-performance PCI-to-PC Card controller that supports two independent PC Card sockets compliant with the1995 PC Card standard. The PCI10XX provides a set of features that make it ideal for bridging between PCI and PC Cards in both notebook and desktop computers. The 1995 PC Card standard retains the 16-bit PC Card specification defined in PCMCIA release 2.1. The 16-bit PC Card is referred to in this document as R2 PC Card (release 2.1 compliant). The PCI10XX supports any combination of 16-bit PC Card (R2) cards in its two sockets, powered at 3.3 V or 5 V as required.

The PCI10XX is compliant with the PCI local bus specification revision 2.1, and its PCI interface can act as both a PCI master or slave device. The PCI bus mastering is initiated during R2 PC Card DMA transfers.

All card signals are individually buffered to allow hot insertion and removal without external buffering. The PCI10XX is register compatible with the Intel 82365SL-DF ExCA controller. The PCI10XX internal datapath logic allows the host to access 8-bit and 16-bit PCI cycles for maximum performance. The 32-bit write buffers allow fast-posted writes to improve system-bus utilization.

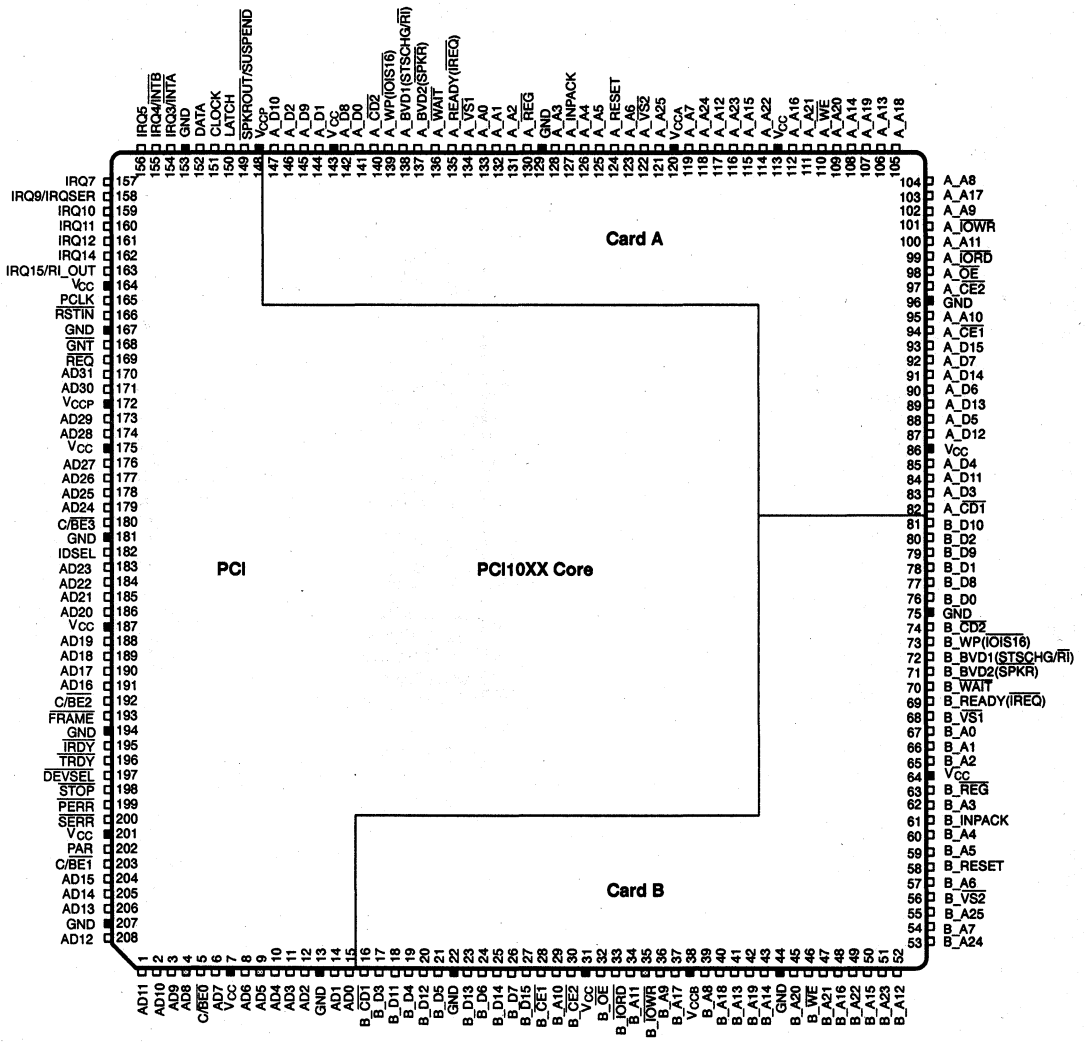
An advanced CMOS process is utilized to achieve low system-power consumption while operating at PCI clock rates up to 33 MHz. Several low-power modes allow the host power-management system to further reduce power consumption.

functional block diagram – R2 PC Card interface



PRODUCT PREVIEW

terminal assignments – PCI-to-PC Card (R2)



PRODUCT PREVIEW

Table 1. Signal Names Sorted by Pin Number – R2 PC Card

NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME
1	AD11	53	B_A24	105	A_A18	157	IRQ7
2	AD10	54	B_A7	106	A_A13	158	IRQ9/IRQSER
3	AD9	55	B_A25	107	A_A19	159	IRQ10
4	AD8	56	B_$\overline{VS2}$	108	A_A14	160	IRQ11
5	C/$\overline{BE0}$	57	B_A6	109	A_A20	161	IRQ12
6	AD7	58	B_RESET	110	A_$\overline{WE}$	162	IRQ14
7	V_{CC}	59	B_A5	111	A_A21	163	IRQ15/RI_OUT
8	AD6	60	B_A4	112	A_A16	164	V_{CC}
9	AD5	61	B_INPACK	113	V_{CC}	165	PCLK
10	AD4	62	B_A3	114	A_A22	166	$\overline{RSTIN}$
11	AD3	63	B_$\overline{REG}$	115	A_A15	167	GND
12	AD2	64	V_{CC}	116	A_A23	168	$\overline{GNT}$
13	GND	65	B_A2	117	A_A12	169	$\overline{REQ}$
14	AD1	66	B_A1	118	A_A24	170	AD31
15	AD0	67	B_A0	119	A_A7	171	AD30
16	B_$\overline{CD1}$	68	B_$\overline{VS1}$	120	V_{CCA}	172	V_{CCP}
17	B_D3	69	B_READY($\overline{IREQ}$)	121	A_A25	173	AD29
18	B_D11	70	B_$\overline{WAIT}$	122	A_$\overline{VS2}$	174	AD28
19	B_D4	71	B_BVD2($\overline{SPKR}$)	123	A_A6	175	V_{CC}
20	B_D12	72	B_BVD1(STSCHG/$\overline{RI}$)	124	A_RESET	176	AD27
21	B_D5	73	B_WP($\overline{IOIS16}$)	125	A_A5	177	AD26
22	GND	74	B_$\overline{CD2}$	126	A_A4	178	AD25
23	B_D13	75	GND	127	A_INPACK	179	AD24
24	B_D6	76	B_D0	128	A_A3	180	C/$\overline{BE3}$
25	B_D14	77	B_D8	129	GND	181	GND
26	B_D7	78	B_D1	130	A_$\overline{REG}$	182	IDSEL
27	B_D15	79	B_D9	131	A_A2	183	AD23
28	B_$\overline{CE1}$	80	B_D2	132	A_A1	184	AD22
29	B_A10	81	B_D10	133	A_A0	185	AD21
30	B_$\overline{CE2}$	82	A_$\overline{CD1}$	134	A_$\overline{VS1}$	186	AD20
31	V_{CC}	83	A_D3	135	A_READY($\overline{IREQ}$)	187	V_{CC}
32	B_$\overline{OE}$	84	A_D11	136	A_$\overline{WAIT}$	188	AD19
33	B_$\overline{IORD}$	85	A_D4	137	A_BVD2($\overline{SPKR}$)	189	AD18
34	B_A11	86	V_{CC}	138	A_BVD1(STSCHG/$\overline{RI}$)	190	AD17
35	B_$\overline{IOWR}$	87	A_D12	139	A_WP($\overline{IOIS16}$)	191	AD16
36	B_A9	88	A_D5	140	A_$\overline{CD2}$	192	C/$\overline{BE2}$
37	B_A17	89	A_D13	141	A_D0	193	$\overline{FRAME}$
38	V_{CCB}	90	A_D6	142	A_D8	194	GND
39	B_A8	91	A_D14	143	V_{CC}	195	$\overline{IRDY}$
40	B_A18	92	A_D7	144	A_D1	196	$\overline{TRDY}$
41	B_A13	93	A_D15	145	A_D9	197	$\overline{DEVSEL}$
42	B_A19	94	A_$\overline{CE1}$	146	A_D2	198	$\overline{STOP}$
43	B_A14	95	A_A10	147	A_D10	199	$\overline{PERR}$
44	GND	96	GND	148	V_{CCP}	200	$\overline{SERR}$
45	B_A20	97	A_$\overline{CE2}$	149	$\overline{SPKROUT/SUSPEND}$	201	V_{CC}
46	B_$\overline{WE}$	98	A_$\overline{OE}$	150	LATCH	202	PAR
47	B_A21	99	A_$\overline{IORD}$	151	CLOCK	203	C/$\overline{BE1}$
48	B_A16	100	A_A11	152	DATA	204	AD15
49	B_A22	101	A_$\overline{IOWR}$	153	GND	205	AD14
50	B_A15	102	A_A9	154	IRQ3/$\overline{INTA}$	206	AD13
51	B_A23	103	A_A17	155	IRQ4/$\overline{INTB}$	207	GND
52	B_A12	104	A_A8	156	IRQ5	208	AD12

PRODUCT PREVIEW

introduction to the PCI10XX

The PCI10XX is a bridge between the PCI local bus and two PC Card sockets supporting 16-bit R2 PC Cards and is compliant with the latest revisions of the PCI local bus specification revision 2.1 and the PCMCIA 1995 PC Card standard. Support for 16-bit PC Card features such as multifunction cards, 3.3-V cards, and direct memory access (DMA) are included in the PCI10XX. The PCI10XX core is powered at 3.3 V to provide low power dissipation, but can independently support either 3.3-V or 5-V signaling on the PCI and PC Card interfaces.

Host software interacts with the PCI10XX through a variety of internal registers that provide status and control information about the PC Cards currently in use and the internal operation of the PCI10XX. These internal registers are accessed by application software either through the PCI configuration header or through programmable windows mapped into PCI memory or I/O address space. The PCI10XX uses a windows format to pass cycles between PCI and PC Card address spaces. Host software must program the location and size of these windows when the PCI10XX or PC Card is initialized.

The PCI10XX also communicates via a three-line serial protocol to the TI TPS2202 dual-power switch. The TPS2202 switches V_{CC} and V_{PP} supply voltage to the two PC Card sockets independently. Host software has indirect control over the TPS2202 by writing to internal PCI10XX registers. To prevent damage to low-voltage CardBus PC Cards, the PCI10XX allows only valid V_{CC} settings to be applied to such cards.

The PCI10XX can notify the host system via interrupts when an event occurs that requires attention from the host. Such events are either card status change (CSC) events or functional interrupts from a PC Card. CSC events occur within the PCI10XX or at the PC Card interface and indicate a change in the status of the socket such as a card insertion or card removal. Functional interrupts originate from the PC Card application and pass from the card to the host system. Both CSC and functional interrupts can be individually masked and routed to a variety of system interrupts. The PCI10XX can signal the system interrupt controller via PCI-style interrupts, ISA IRQs, or with the serialized IRQ protocol.

PCI configuration headers

A number of registers found in the PCI10XX PCI configuration space are defined in the PCI-to-PCI bridge architecture specification revision 1.0 and are common to the PCI local bus specification revision 2.1. Registers common to both are the device ID, vendor ID, status, command, class code, revision ID, BIST, header type, latency timer, cache line size, interrupt pin, and interrupt line registers. The special needs of a PCI-to-PCI bridge and the PCI10XX require additional registers in the form of the CardBus latency timer, subordinate bus number, CardBus bus number, PCI bus number, secondary-status and bridge-control registers.

Most of the registers are implemented in the PCI10XX as defined in either the PCI local bus specification revision 2.1 or the PCI-to-PCI bridge architecture specification revision 1.0. References to these documents are made where appropriate.

Host software exerts control and retrieves status information on PC Cards via a standard set of internal ExCA registers. The PCI10XX maps these registers into PCI address space for access by host software. The locations of these registers are set by the CardBus socket registers/ExCA registers base-address register in PCI configuration space, which locates a 4K-byte nonprefetchable memory window in PCI memory-address space. Within this memory window, the PCI10XX maps both the socket registers and the ExCA registers. Each socket has a separate CardBus socket register/ExCA registers base-address register for accessing the ExCA registers.

The 16-bit PC Cards use the ExCA register set for card status and control purposes. These registers are accessed by host software through an index/data register pair. Software writes the index of the desired ExCA register to the index register and reads or writes the desired data to the data register. The PCI10XX also supports the index/data scheme of accessing the ExCA registers through the use of the PC Card 16-bit IF legacy-mode base-address register. An address written to this register becomes the address for the index register and the address+1 becomes the address for the data address. Using this access method, applications requiring index/data type ExCA access can be supported.

PCI configuration headers (continued)

This PC Card 16-bit legacy-mode base address is shared by both sockets and the ExCA registers run contiguously from offset 00h–3Fh for socket A and 40h–7Fh for socket B. The PCI10XX also maps the ExCA register set to a 4K-byte memory window located by the CardBus socket registers/ExCA registers base address. The ExCA registers are offset from this base address by 800h.

The 16-bit PC Cards use a set of six 32-bit socket registers, defined by the PC Card 1995 specification, which the PCI10XX maps into PCI memory space. The location of these registers in PCI memory space is set by the CardBus socket registers/ExCA registers base-address register, also in PCI configuration space. This 32-bit base address allows the CardBus socket registers to be anywhere within the 4G-byte PCI memory address range. The CardBus socket registers occupy 48 bytes of PCI memory space, and the CardBus socket registers/ExCA registers base address mark the beginning of a 4K-byte block of addresses for which the PCI10XX claims PCI memory cycles. Bit 0 of the PCI10XX command register must be set to claim PCI memory cycles.

The TI extension registers are specific to PCI10XX value-added features that are not part of currently defined industry specifications. The registers, which reside in the TI extension registers, are a collection of control and status bits that are required to support various PCI10XX functionality. This functionality typically does not exist within the register models implemented elsewhere within this device. Table 2 shows the TI extension registers and their location in PCI configuration space.

Table 2. TI Extension Registers

REGISTER NAME	OFFSET
System control register†	80h
Retry status register†	90h
Card control register†	91h
Device control register†	92h
Buffer control register†	93h

† Indicates registers that are common to both PC Card socket configuration space 0 and 1

The PCI10XX supports the DMA specification defined in the 1995 PC Card standard by providing one DMA channel per socket. The PC Card standard stipulates the signaling and timing associated with DMA transfers to and from a PC Card. This defines DMA transfers from the PC Card to the socket only. On the PCI side, the PCI10XX implements a set of status and control registers similar to the programming model of the original dual 8237 DMA controller found in PC-ATs. These registers comply with the specification for DMA in a PCI environment, particularly as it defines slave DMA devices. The PCI10XX provides two registers in its configuration header that set up both the PCI interface and PC Card socket for DMA.

Host software must program the PCI10XX socket DMA registers 0 and 1 to set up the socket for DMA transfers. Socket DMA register 0 applies to the PC Card portion of DMA transfers. Socket DMA register 1 applies to the PCI portion of DMA transfers, specifically to set up the slave DMA support required in distributed DMA (DMA support on the PCIWay). Socket DMA register 1 provides register bits to program the DMA transfer width. This transfer width refers to both the PC Card interface and the PCI interface.

Descriptions of each of the registers follow. Before writing data to each of the TI extension registers, host software must first read the register, modify the contents, and write back the data. This preserves current register settings and prevents unpredictable or undesired behavior.

ExCA registers

The ExCA architecture registers implemented in the PCI10XX are register compatible with the Intel 82365SL-DF PCMCIA controller. The PCI10XX controller makes the ExCA registers for each socket available by directly mapping them into PCI memory space. They are accessed through the CardBus socket registers/ExCA registers base-address register at offset 800h. Each socket has a separate CardBus socket register/ExCA registers base-address register for accessing the ExCA registers (see Figure 1). The ExCA offset is the offset from the PC Card 16-bit IF legacy-mode base address. This PC Card 16-bit legacy-mode base address is shared by both sockets. The ExCA registers run contiguously from offset 00h–3Fh for socket A and 40h–7Fh for socket B (see Figure 2). Table 3 identifies each ExCA register and its respective ExCA offset and PCI configuration header address.

The ExCA general setup registers, as defined in the Intel 82365SL-DF specification, provide status and control information on a variety of R2 PC Card functions. These registers are concerned with V_{CC}/V_{PP} control, PC Card status, memory and I/O window control, and global card status. This set of registers includes those registers at offsets 800h, 801h, 802h, 804h, 806h, 816h, 81Eh, and 840h.

The interrupt registers in the ExCA register set (as defined in the Intel 82365SL-DF specification) control such card functions as reset, type, interrupt routing, and interrupt enables. Special attention must be paid to the interrupt routing registers and the host-interrupt signaling method selected for the PCI10XX. Certain IRQs are available only if the serial interrupt scheme is selected. This scheme is a method by which IRQ information is communicated serially to the host-interrupt controller through a common, wired-OR terminal on the PCI10XX. If discrete IRQ signaling is selected, only a subset of the possible IRQs are available for interrupt routing. Host software should first select the interrupt signaling method to be used, then route the PC Card interrupt sources to host interrupts. This set of registers includes those registers at ExCA offsets 803h and 805h.

The R2 I/O PC Cards are available to the host system via I/O windows. These are regions of the host I/O address space into which the card I/O space is mapped. These windows are defined by start and end addresses programmed in the ExCA registers described in this section. I/O windows have byte granularity.

The R2 memory PC Cards are available to the host system via memory windows. These are regions of host memory address 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. Memory windows have double-word granularity.

ExCA registers (continued)

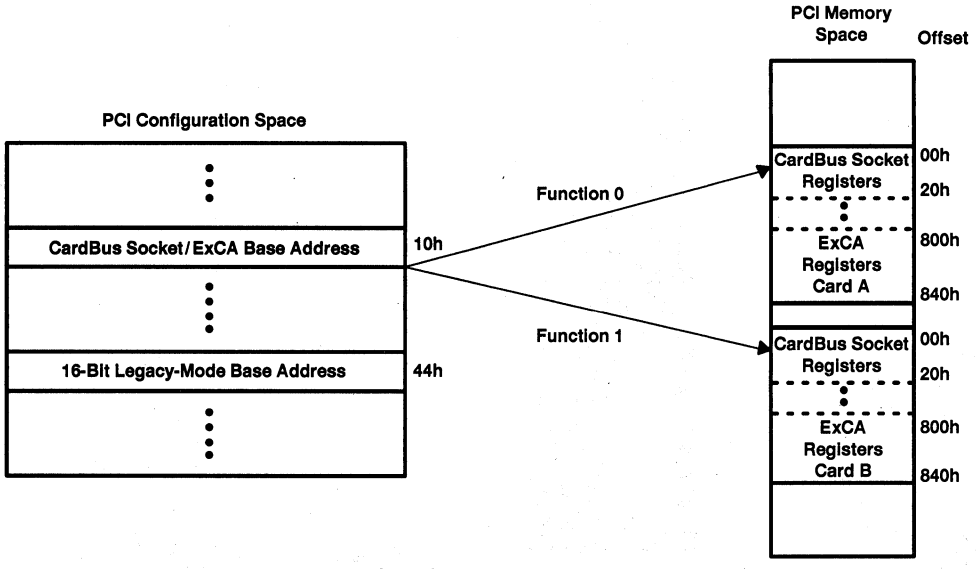


Figure 1. ExCA PCI Memory Access Method

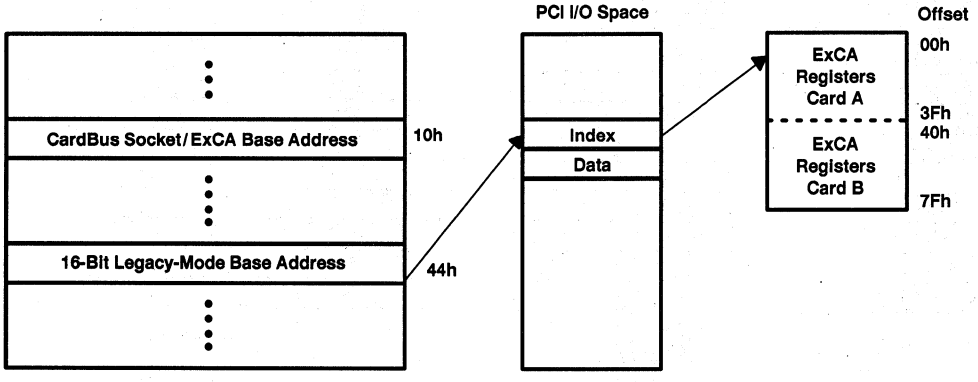


Figure 2. ExCA PCI I/O Legacy Access Method

ExCA registers (continued)

Table 3. ExCA Registers

NAME	PCI MEMORY ADDRESS OFFSET	ExCA OFFSET	
		CARD A	CARD B
Identification and revision register	800	00	40
Interface status register	801	01	41
Power control register	802	02	42
Interrupt and general control register	803	03	43
Card status-change register	804	04	44
Card status-change interrupt configuration register	805	05	45
Address window enable register	806	06	46
I/O window-control register	807	07	47
I/O window 0 start-address low-byte register	808	08	48
I/O window 0 start-address high-byte register	809	09	49
I/O window 0 end-address low-byte register	80A	0A	4A
I/O window 0 end-address high-byte register	80B	0B	4B
I/O window 1 start-address low-byte register	80C	0C	4C
I/O window 1 start-address high-byte register	80D	0D	4D
I/O window 1 end-address low-byte register	80E	0E	4E
I/O window 1 end-address high-byte register	80F	0F	4F
Memory window 0 start-address low-byte register	810	10	50
Memory window 0 start-address high-byte register	811	11	51
Memory window 0 end-address low-byte register	812	12	52
Memory window 0 end-address high-byte register	813	13	53
Memory window 0 offset-address low-byte register	814	14	54
Memory window 0 offset-address high-byte register	815	15	55
Card detect and general control register	816	16	56
Reserved	817	17	57
Memory window 1 start-address low-byte register	818	18	58
Memory window 1 start-address high-byte register	819	19	59
Memory window 1 end-address low-byte register	81A	1A	5A
Memory window 1 end-address high-byte register	81B	1B	5B
Memory window 1 offset-address low-byte register	81C	1C	5C
Memory window 1 offset-address high-byte register	81D	1D	5D
Global control register	81E	1E	5E
Reserved	81F	1F	5F
Memory window 2 start-address low-byte register	820	20	60
Memory window 2 start-address high-byte register	821	21	61
Memory window 2 end-address low-byte register	822	22	62
Memory window 2 end-address high-byte register	823	23	63
Memory window 2 offset-address low-byte register	824	24	64
Memory window 2 offset-address high-byte register	825	25	65
Reserved	826	26	66
Reserved	827	27	67

ExCA registers (continued)

Table 3. ExCA Registers (Continued)

NAME	PCI MEMORY ADDRESS OFFSET	ExCA OFFSET	
		CARD A	CARD B
Memory window 3 start-address low-byte register	828	28	68
Memory window 3 start-address high-byte register	829	29	69
Memory window 3 end-address low-byte register	82A	2A	6A
Memory window 3 end-address high-byte register	82B	2B	6B
Memory window 3 offset-address low-byte register	82C	2C	6C
Memory window 3 offset-address high-byte register	82D	2D	6D
Reserved	82E	2E	6E
Reserved	82F	2F	6F
Memory window 4 start-address low-byte register	830	30	70
Memory window 4 start-address high-byte register	831	31	71
Memory window 4 end-address low-byte register	832	32	72
Memory window 4 end-address high-byte register	833	33	73
Memory window 4 offset-address low-byte register	834	34	74
Memory window 4 offset-address high-byte register	835	35	75
Reserved	836	36	76
Reserved	837	37	77
Reserved	838	38	78
Reserved	839	39	79
Reserved	83A	3A	7A
Reserved	83B	3B	7B
Reserved	83C	—	—
Reserved	83D	3D	7D
Reserved	83E	3E	7E
Reserved	83F	3F	7F
Memory window page register	840	3C	7C

PRODUCT PREVIEW

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 PCI10XX provides the CardBus socket base-address register to locate these CardBus socket registers in PCI memory address space. Each socket has a separate CardBus socket register/ExCA registers base-address register for accessing the CardBus socket registers (see Figure 3). This base-address register is located at offset 10h in the PCI10XX configuration space. Table 3 shows the location of the socket registers in relation to the CardBus socket base address. The socket power management register is an extended register that provides control and status information related to power management.

CardBus socket registers (continued)

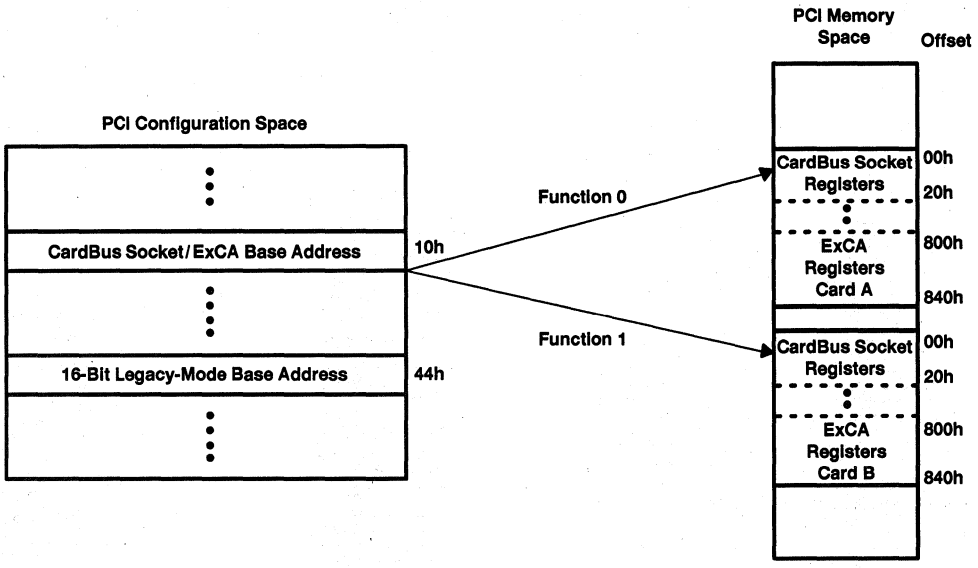


Figure 3. CardBus Socket/ExCA PCI Memory Access Method

Table 4. CardBus Socket Registers

REGISTER NAME	OFFSET
Socket event register	00h
Socket mask register	04h
Socket present state register	08h
Socket force event register	0Ch
Socket control register	10h
Reserved	14–1Fh
Socket power management register	20h

DMA registers

The DMA base-address register, located in PCI configuration space at offset 98h, points to a 16-byte region in PCI I/O space where the DMA registers reside. The names and locations of these registers are summarized in Table 5. These DMA registers are identical in function to the 8237 DMA controller. The similarity between the register models retains compatibility with legacy DMA and simplifies the translation required by the master DMA device when forwarding legacy DMA writes to slave DMA channels.

While the slave DMA register definitions are identical to those in the 8237 DMA controller with 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 PCI10XX implements these obsolete register bits as read-only, nonfunctional bits. The reserved registers shown in Table 5 are implemented as read only and return 0s when read. Writes to reserved registers have no effect.

DMA registers (continued)

Table 5. DMA Registers

R/W	REGISTER NAME				DMA BASE ADDRESS OFFSET
R	Reserved	Page	Current address		00h
W			Base address		
R	Reserved	Reserved	Current word		04h
W			Base word		
R	NA	Reserved	NA	Status	08h
W	Mode		Request	Command	
R	Multichannel mask	Reserved	NA	Reserved	0Ch
W			Master clear		

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

Supply voltage range, V_{CC} ..... −0.5 V to 6 V
Input voltage range, V_I: Standard ..... −0.5 V to V_{CC} + 0.5 V
Fail safe ..... −0.5 V to 6.5 V
Output voltage range, V_O: Standard ..... −0.5 V to V_{CC} + 0.5 V
Fail safe ..... −0.5 V to 6.5 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 to external input and bidirectional buffers. $V_I > V_{CC}$ does not apply to fail-safe terminals.
2. Applies to external output and bidirectional buffers. $V_O > V_{CC}$ does not apply to fail-safe terminals.

recommended operating conditions

			MIN	NOM	MAX	UNIT
t_t	Input transition (rise and fall) time	CMOS compatible	0		25	ns
T_A	Operating ambient temperature	Commercial	0	25	70	°C
T_J‡	Virtual junction temperature	Commercial	0	25	115	°C

‡ These junction temperatures reflect simulation conditions. Customer is responsible for verifying junction temperature.

recommended operating conditions for PCI interface

			OPERATION	MIN	NOM	MAX	UNIT
V_{CC}	Core voltage	Commercial	3.3 V	3	3.3	3.6	V
V_{CCP}	PCI supply voltage	Commercial	3.3 V	3	3.3	3.6	V
			5 V	4.75	5	5.25	
V_I	Input voltage		3.3 V	0		V_{CCP}	V
			5 V	0		V_{CCP}	
V_O§	Output voltage		3.3 V	0		V_{CCP}	V
			5 V	0		V_{CCP}	
V_{IH}¶	High-level input voltage	CMOS compatible	3.3 V	0.7 V_{CC}			V
			5 V	2			
V_{IL}¶	Low-level input voltage	CMOS compatible	3.3 V			0.3 V_{CC}	V
			5 V			0.8	

§ Applies to external output buffers
¶ Applies to external input and bidirectional buffers without hysteresis

PRODUCT PREVIEW

recommended operating conditions for PC Cards A and B and miscellaneous inputs and outputs

			OPERATION	MIN	NOM	MAX	UNIT
$V_{CC(A/B)}$	PC Card supply voltage	Commercial	3.3 V	3	3.3	3.6	V
			5 V	4.75	5	5.25	
V_I	Input voltage		3.3 V	0		$V_{CC(A/B)}$	V
			5 V	0		$V_{CC(A/B)}$	
V_O†	Output voltage		3.3 V	0		$V_{CC(A/B)}$	V
			5 V	0		$V_{CC(A/B)}$	
V_{IH}‡	High-level input voltage	CMOS compatible	3.3 V	2			V
			5 V	2.4			
V_{IL}‡	Low-level input voltage	CMOS compatible	3.3 V			0.8	V
			5 V			0.8	

† Applies to external output buffers
‡ Applies to external input and bidirectional buffers without hysteresis

electrical characteristics over recommended operating conditions (unless otherwise noted)

PARAMETER		SIDE	OPERATION	TEST CONDITIONS	MIN	MAX	UNIT
V_{OH}	High-level output voltage	PCI	3.3 V		0.9 V_{CC}		V
			5 V		2.4		
		PC Card	3.3 V	See Note 3	0.9 V_{CC}		
			5 V	See Note 3	2.4		
V_{OL}	Low-level output voltage	PCI	3.3 V			0.1 V_{CC}	V
			5 V			0.55	
		PC Card	3.3 V	See Note 4		0.1 V_{CC}	
			5 V	See Note 4		0.55	
I_{OZ}	High-impedance output current			V_I = V_{CC} or GND§		±10	µA
I_{IL}	Low-level input current			V_I = GND		–1	µA
I_{IH}	High-level input current			V_I = V_{CC}¶		1	µA

§ The 3-state or open-drain outputs must be in the high-impedance state.
¶ Applies to all inputs except TEST
NOTES: 3. I_{OH} = –0.9 mA for all PC Card outputs and bidirectionals; I_{OH} = –1.8 mA for all miscellaneous outputs
4. I_{OL} = 1.62 mA for all PC Card outputs and bidirectionals; I_{OL} = 3.24 mA for all miscellaneous outputs

PCI clock/reset timing requirements over recommended ranges of supply voltage and operating free-air temperature (see Figure 5 and Figure 6)

		ALTERNATE SYMBOL	TEST CONDITIONS	MIN	MAX	UNIT
t_c	Cycle time, PCLK	t_{cyc}		30	∞	ns
t_{wH}	Pulse duration, PCLK high	t_{high}		12		ns
t_{wL}	Pulse duration, PCLK low	t_{low}		12		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

PRODUCT PREVIEW

PCI timing requirements over recommended ranges of supply voltage and operating free-air temperature (see Figure 4 and Figure 7)

			ALTERNATE SYMBOL	TEST CONDITIONS	MIN	MAX	UNIT
t_{pd}	Propagation delay time	PCLK to shared signal valid delay time	t_{val}	C_L = 50 pF, See Note 5		11	ns
		PCLK to shared signal invalid delay time	t_{inv}		2		
t_{en}	Enable time, high-impedance-to-active delay time from PCLK		t_{on}		2		ns
t_{dis}	Disable time, active-to-high-impedance delay time from PCLK		t_{off}			28	ns
t_{su}	Setup time before PCLK valid		t_{su}		7		ns
t_h	Hold time after PCLK high		t_h		0		ns

NOTE 5: PCI shared signals are AD31–AD0, C/$\overline{BE3}$–C/$\overline{BE0}$, $\overline{FRAME}$, $\overline{TRDY}$, $\overline{IRDY}$, $\overline{STOP}$, IDSEL, $\overline{DEVSEL}$, and PAR.

PRODUCT PREVIEW

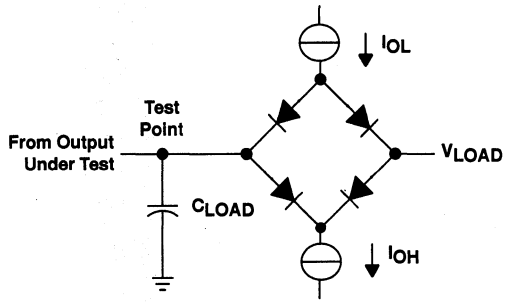
PARAMETER MEASUREMENT INFORMATION

LOAD CIRCUIT PARAMETERS

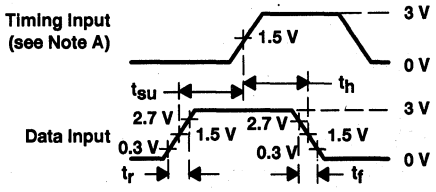
TIMING PARAMETER		C_{LOAD}† (pF)	I_{OL} (mA)	I_{OH} (mA)	V_{LOAD} (V)
t_{en}	t_{PZH}	50	8	−8	0
	t_{PZL}				3
t_{dis}	t_{PHZ}	50	8	−8	1.5
	t_{PLZ}				
t_{pd}		50	8	−8	‡

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

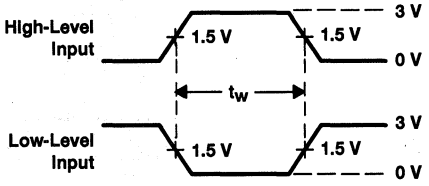
‡ $\frac{V_{LOAD} - V_{OL}}{I_{OL}}$ = 50 Ω, where V_{OL} = 0.6 V, I_{OL} = 8 mA



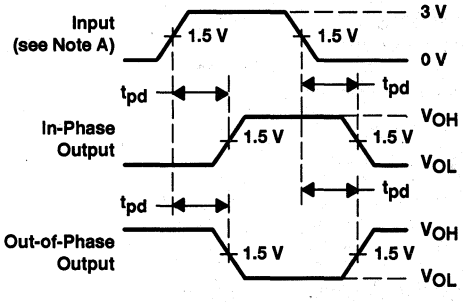
LOAD CIRCUIT



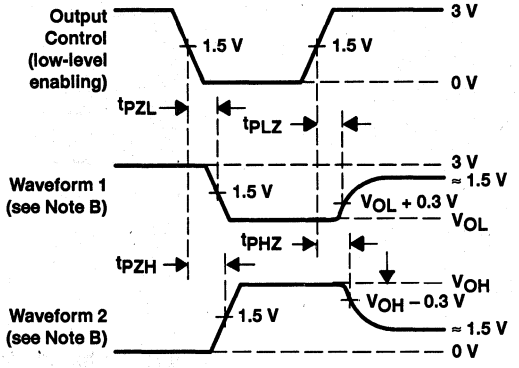
VOLTAGE WAVEFORMS
SETUP AND HOLD TIMES
INPUT RISE AND FALL TIMES



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, $t_r \le 6$ ns, $t_f \le 6$ 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 4. Load Circuit and Voltage Waveforms

PRODUCT PREVIEW

PCIBUS PARAMETER MEASUREMENT INFORMATION

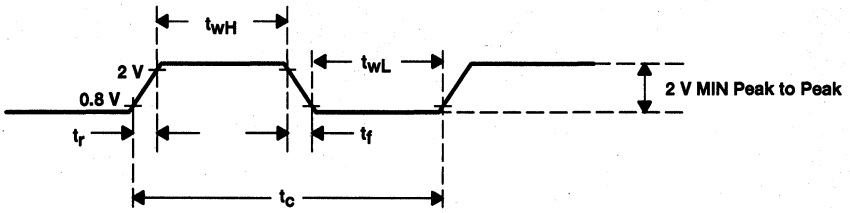


Figure 5. PCLK Timing Waveform

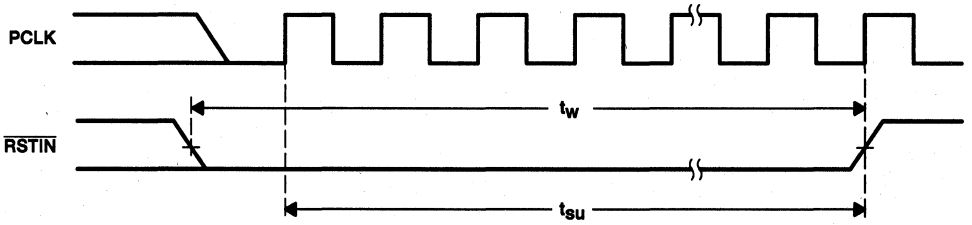


Figure 6. $\overline{\text{RSTIN}}$ Timing Waveforms

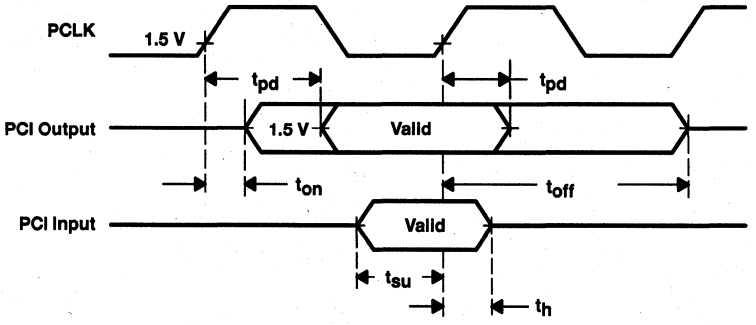


Figure 7. Shared Signals Timing Waveforms

PRODUCT PREVIEW

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 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 6 shows address setup time in PCLK cycles and nanoseconds for I/O and memory cycles. Tables 7 and 8 show command active time in PCLK cycles and nanoseconds for I/O and memory cycles. Table 9 shows address hold time in PCLK cycles and nanoseconds for I/O and memory cycles.

Table 6. 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 7. PC Card Command Active 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 8. PC Card Command Active 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	21/630
	00	1	5/150

Table 9. 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

PRODUCT PREVIEW

timing requirements over recommended ranges of supply voltage and operating free-air temperature, memory cycles (for 100-ns common memory) (see Note 6 and Figure 8)

		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 6: 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 9)

		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 10)

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

PC CARD PARAMETER MEASUREMENT INFORMATION

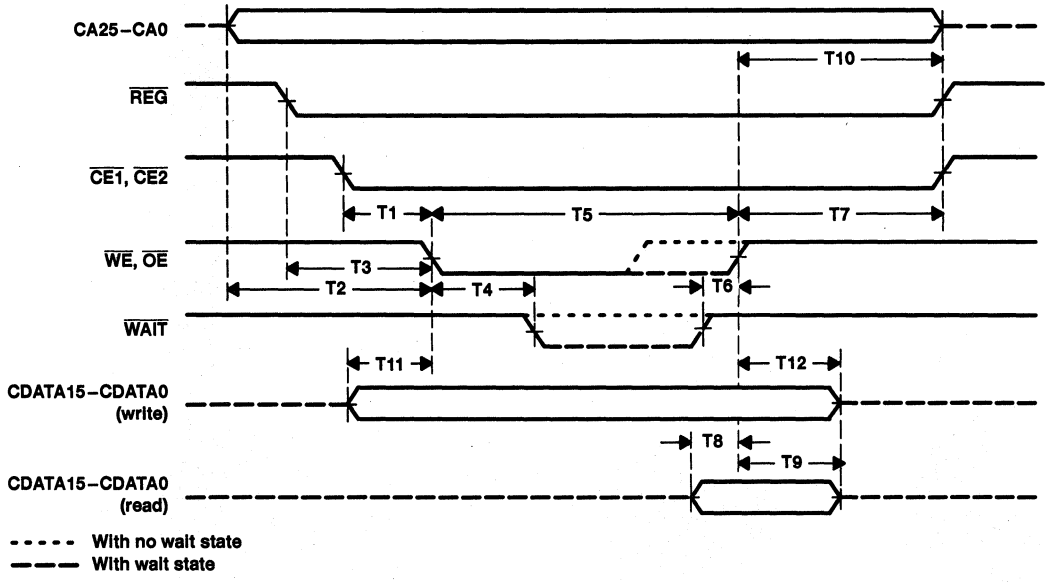


Figure 8. PC Card Memory Cycle

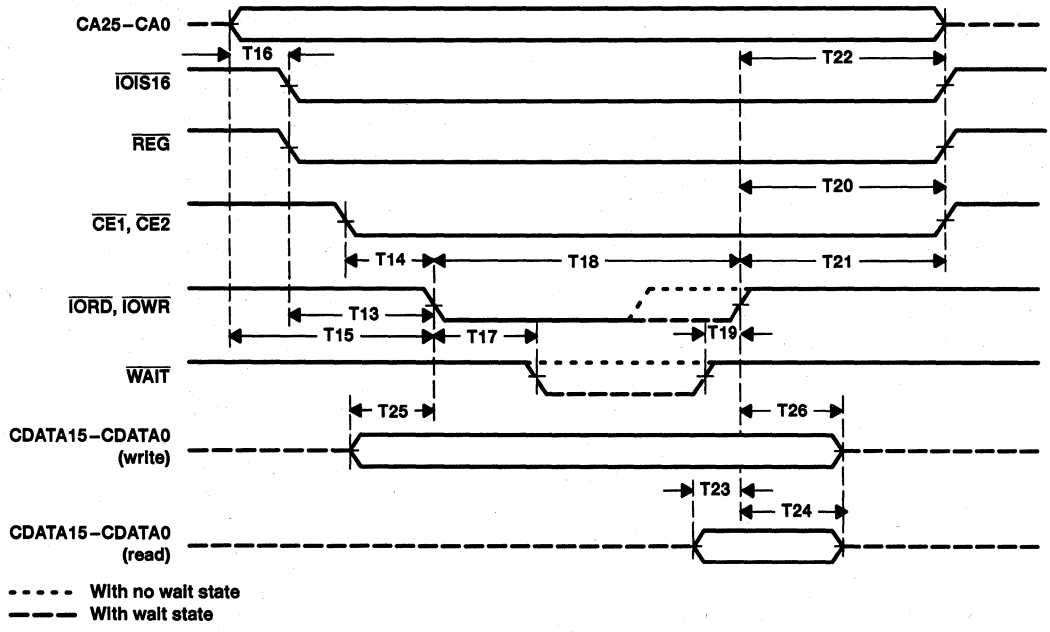


Figure 9. PC Card I/O Cycle

PRODUCT PREVIEW

PC CARD PARAMETER MEASUREMENT INFORMATION

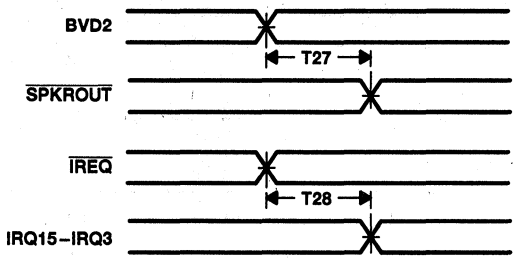


Figure 10. Miscellaneous PC Card Delay Times

4 PCI1130

SCPS003A – FEBRUARY 1996 – REVISED MARCH 1996

- 3.3-V Core Logic With Universal PCI Interface Compatible With 3.3-V or 5-V PCI Signaling Environments
- Supports PCI Local Bus Specification 2.1
- Mix and Match 3.3-V/5-V PC Card16 Cards and 3.3-V CardBus Cards
- Supports Two PC Card™ or CardBus Slots With Hot Insertion and Removal
- 1995 PC Card Standard Compliant
- Low-Power Advanced Submicron CMOS Technology
- Uses Serial Interface to Texas Instruments (TI) TPS2202A Dual Power Switch
- System Interrupts Can Be Programmed as PCI-Style or ISA IRQ-Style Interrupts
- ISA IRQ Interrupts Can Be Serialized Onto a Single IRQSER Pin
- Independent Read and Write Buffers for Each Direction
- Supports Burst Transfers to Maximize Data Throughput on the PCI and CardBus Bus
- Multifunction PCI Device With Separate Configuration Spaces for Each Socket
- Five PCI Memory Windows and Two I/O Windows Available to Each PC Card16 Socket
- Two I/O Windows and Two Memory Windows Available to Each CardBus Socket
- CardBus Memory Windows Can Be Individually Selected Prefetchable or Nonprefetchable
- Exchangeable Card (ExCA™)-Compatible Registers Are Mapped In Memory and I/O Space
- TI Extension Registers Are Mapped in the PCI Configuration Space
- Intel™ 82365SL-DF Register Compatible
- Supports 16-Bit Distributed Direct Memory Access (DMA) on Both PC Card Sockets
- Supports PC/PCI DMA on Both PC Card Sockets
- Supports Zoom Video Mode
- Supports Ring Indicate
- Packaged in 208-Pin Thin Plastic Quad Flatpack (PDV)

PRODUCT PREVIEW

Table of Contents

ExCA and PC Card are trademarks of Personal Computer Memory Card International Association (PCMCIA).
Intel and MPIIX are trademarks of Intel Corp.

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

description

The TI PCI1130 is a high-performance PCI-to-PC Card controller that supports two independent PC Card sockets compliant with the 1995 PC card standard. The PCI1130 provides a set of features that make it ideal for bridging between PCI and PC Cards in both notebook and desktop computers. The 1995 PC Card standard retains the 16-bit PC Card specification defined in PCMCIA release 2.1 and defines the new 32-bit PC Card, called CardBus, capable of full 32-bit data transfers at 33 MHz. The PCI1130 supports any combination of 16-bit and CardBus PC Cards in its two sockets, powered at 3.3 V or 5 V as required.

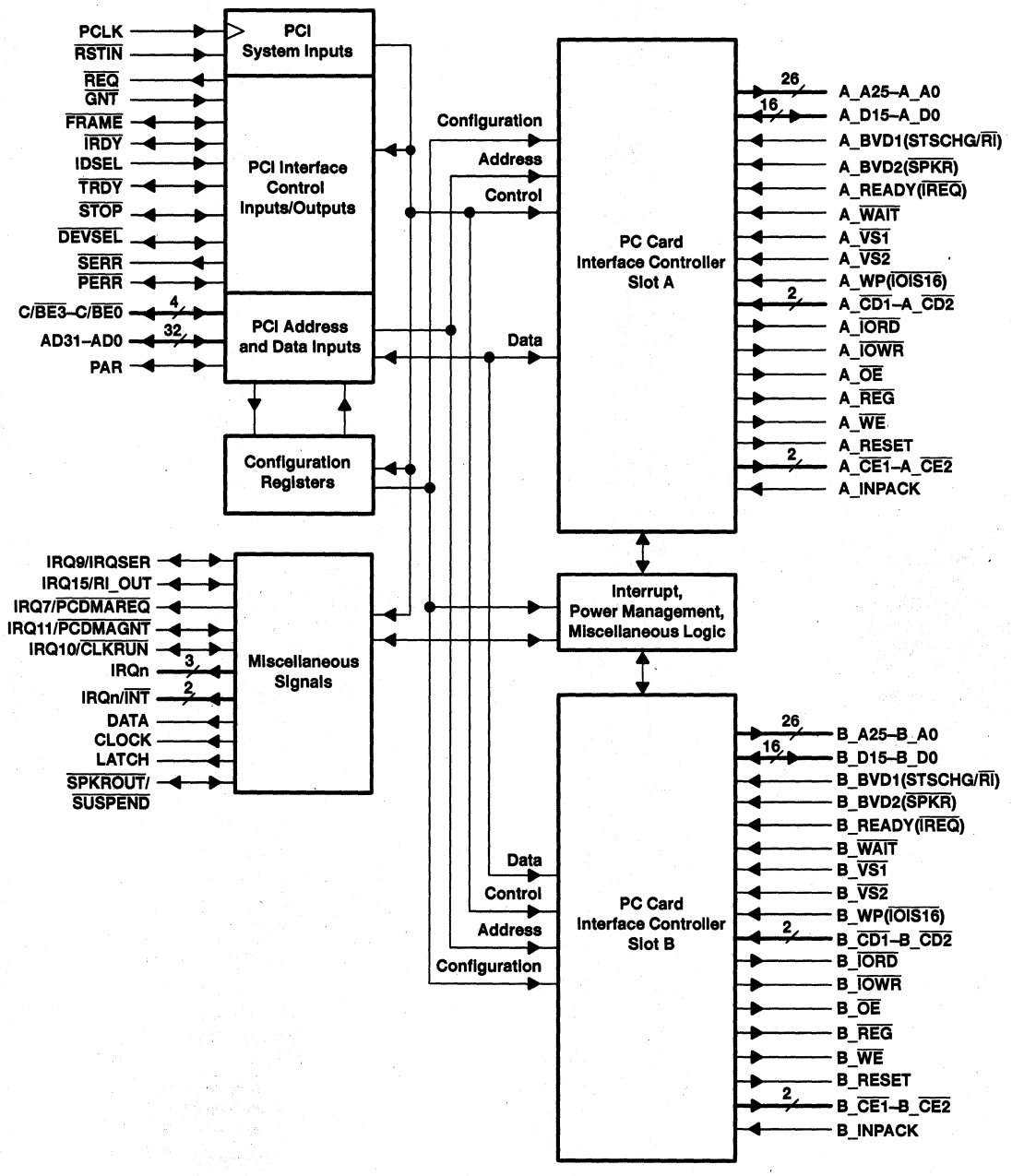
The PCI1130 is compliant with the PCI local bus specification revision 2.1, and its PCI interface can act as either a PCI master device or a PCI slave device. The PCI bus mastering is initiated during 16-bit PC Card DMA transfers or CardBus PC Card bus mastering cycles.

All card signals are internally buffered to allow hot insertion and removal without external buffering. The PCI1130 is register compatible with the Intel 82365SL-DF ExCA controller. The PCI1130 internal datapath logic allows the host to access 8-, 16-, and 32-bit cards using full 32-bit PCI cycles for maximum performance. Independent 32-bit write buffers allow fast-posted writes to improve system-bus utilization.

An advanced CMOS process is used to achieve low system-power consumption while operating at PCI clock rates up to 33 MHz. Several low-power modes allow the host power-management system to further reduce power consumption.

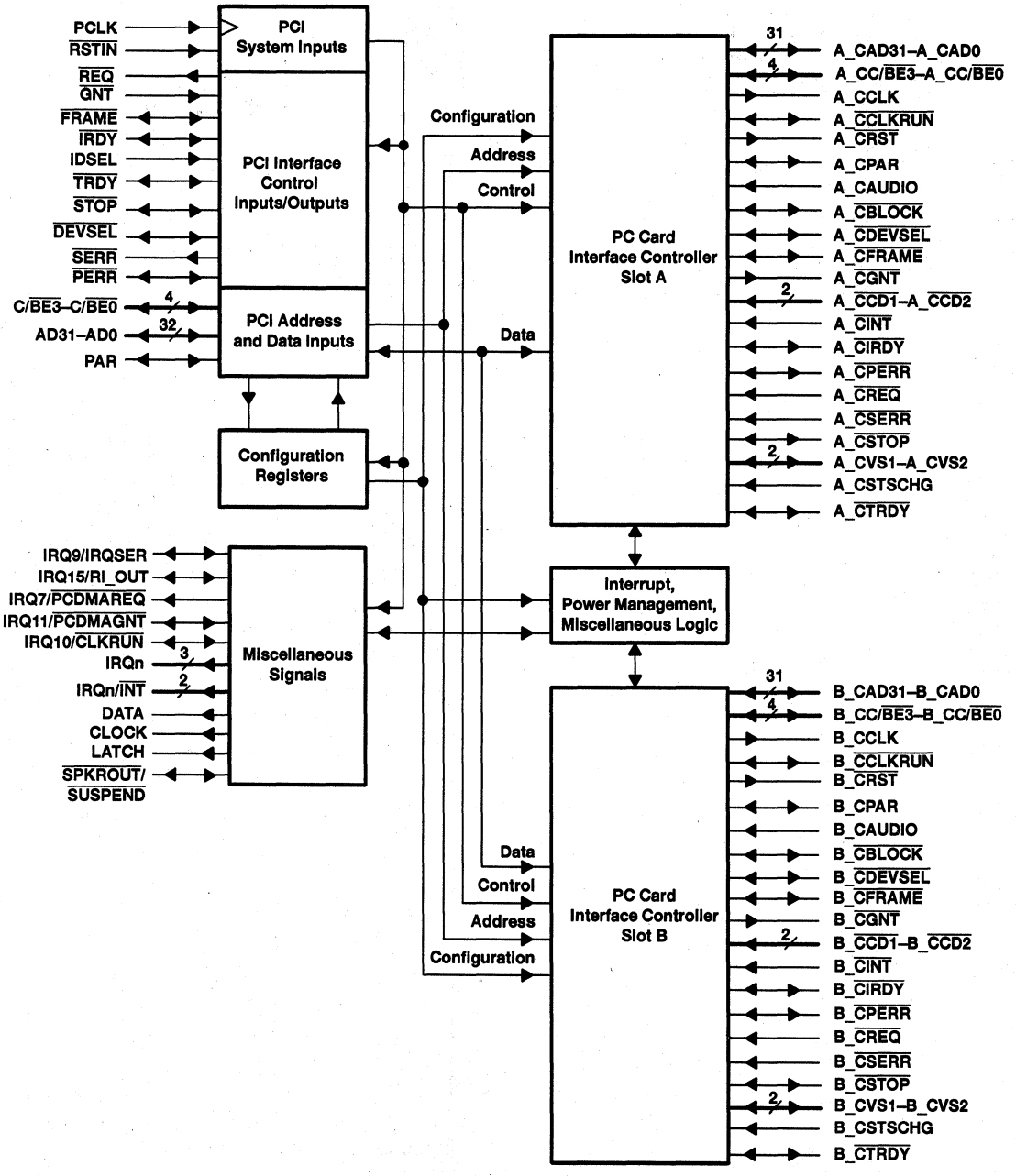
PRODUCT PREVIEW

functional block diagram – 16-bit PC Card interface



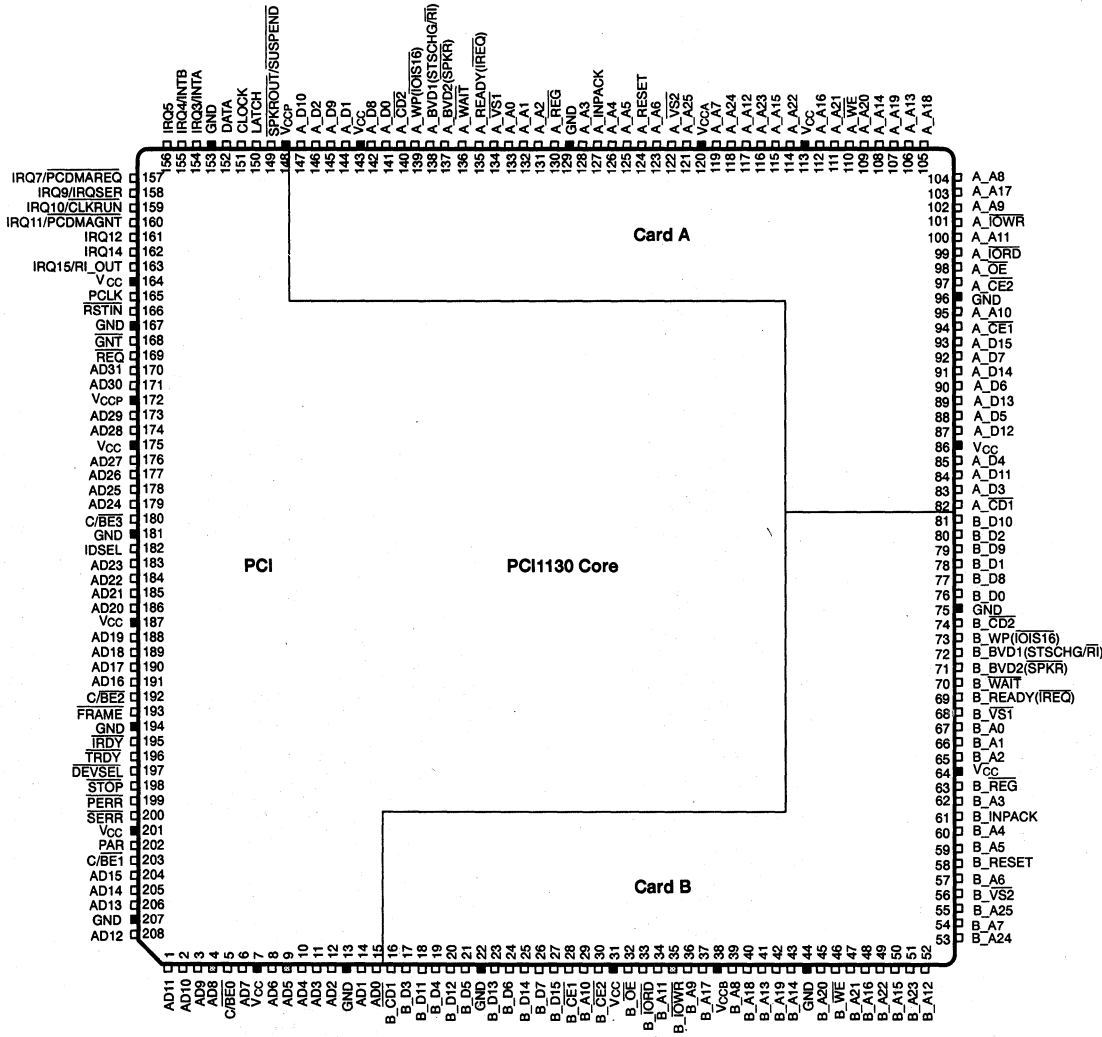
PRODUCT PREVIEW

functional block diagram – CardBus Card interface



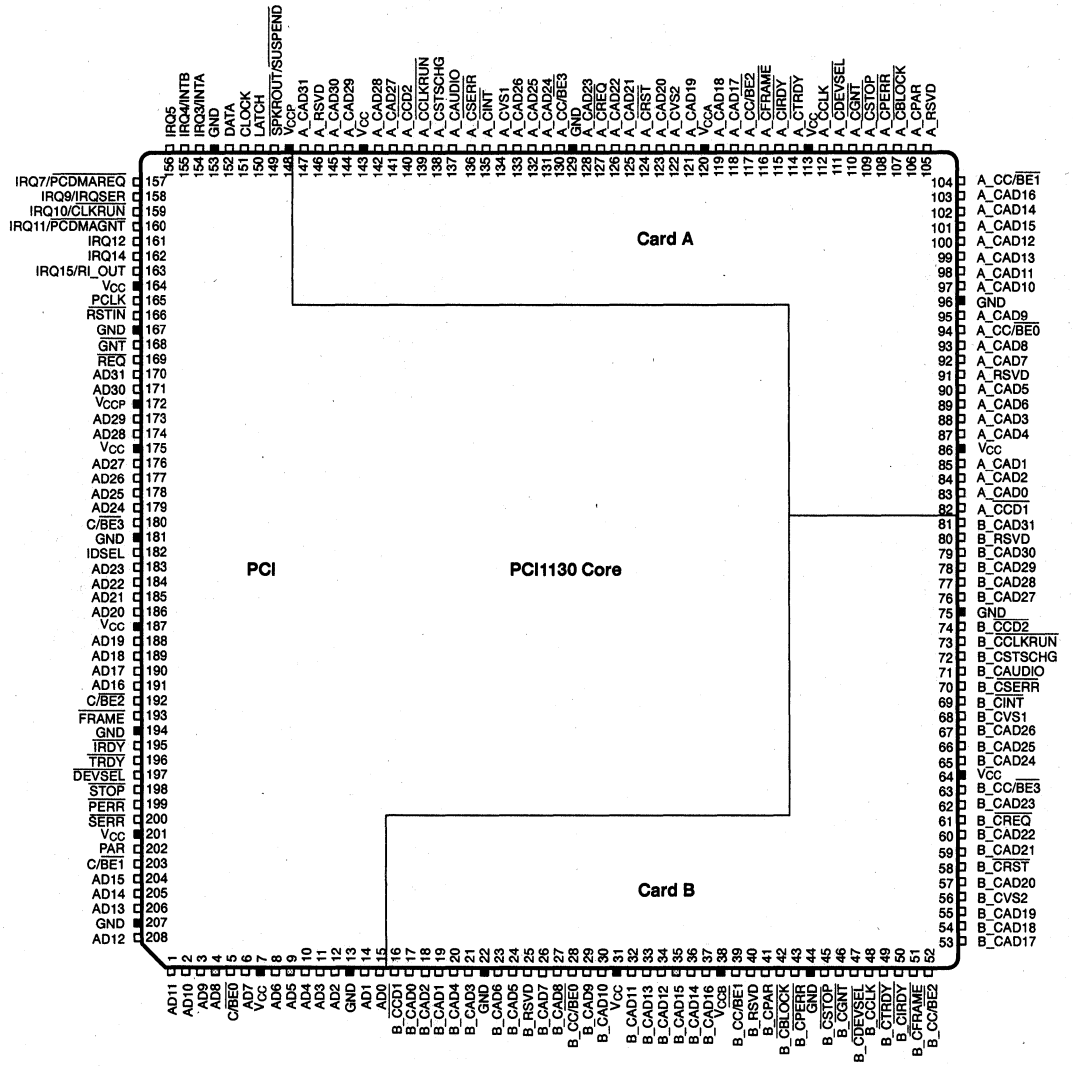
PRODUCT PREVIEW

terminal assignments – PCI-to-PC Card (16 bit)



PRODUCT PREVIEW

terminal assignments – PCI-to-CardBus



PRODUCT PREVIEW

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

Table 1. Signal Names Sorted Alphabetically – 16-bit PC Card

SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.
A_A0	133	A_READY(IREQ)	135	B_A12	52	CLOCK	151
A_A1	132	A_REG	130	B_A13	41	DATA	152
A_A2	131	A_RESET	124	B_A14	43	DEVSEL	197
A_A3	128	A_VS1	134	B_A15	50	FRAME	193
A_A4	126	A_VS2	122	B_A16	48	GND	13
A_A5	125	A_WAIT	136	B_A17	37	GND	22
A_A6	123	A_WE	110	B_A18	40	GND	44
A_A7	119	A_WP(IOSI16)	139	B_A19	42	GND	75
A_A8	104	AD0	15	B_A20	45	GND	96
A_A9	102	AD1	14	B_A21	47	GND	129
A_A10	95	AD2	12	B_A22	49	GND	153
A_A11	100	AD3	11	B_A23	51	GND	167
A_A12	117	AD4	10	B_A24	53	GND	181
A_A13	106	AD5	9	B_A25	55	GND	194
A_A14	108	AD6	8	B_BVD1(STSCHG/RI)	72	GND	207
A_A15	115	AD7	6	B_BVD2(SPKR)	71	GNT	168
A_A16	112	AD8	4	B_CD1	16	IDSEL	182
A_A17	103	AD9	3	B_CD2	74	IRDY	195
A_A18	105	AD10	2	B_CE1	28	IRQ3/INTA	154
A_A19	107	AD11	1	B_CE2	30	IRQ4/INTB	155
A_A20	109	AD12	208	B_D0	76	IRQ5	156
A_A21	111	AD13	206	B_D1	78	IRQ7/PCDMAREQ	157
A_A22	114	AD14	205	B_D2	80	IRQ9/IRQSER	158
A_A23	116	AD15	204	B_D3	17	IRQ10/CLKRUN	159
A_A24	118	AD16	191	B_D4	19	IRQ11/PCDMAGNT	160
A_A25	121	AD17	190	B_D5	21	IRQ12	161
A_BVD1(STSCHG/RI)	138	AD18	189	B_D6	24	IRQ14	162
A_BVD2(SPKR)	137	AD19	188	B_D7	26	IRQ15/RI_OUT	163
A_CD1	82	AD20	186	B_D8	77	LATCH	150
A_CD2	140	AD21	185	B_D9	79	PAR	202
A_CE1	94	AD22	184	B_D10	81	PCLK	165
A_CE2	97	AD23	183	B_D11	18	PERR	199
A_D0	141	AD24	179	B_D12	20	REQ	169
A_D1	144	AD25	178	B_D13	23	RSTIN	166
A_D2	146	AD26	177	B_D14	25	SPKROUT/SUSPEND	149
A_D3	83	AD27	176	B_D15	27	STOP	198
A_D4	85	AD28	174	B_INPACK	61	SERR	200
A_D5	88	AD29	173	B_IORD	33	TRDY	196
A_D6	90	AD30	171	B_IOWR	35	V_{CC}	7
A_D7	92	AD31	170	B_OE	32	V_{CC}	31
A_D8	142	B_A0	67	B_READY(IREQ)	69	V_{CC}	64
A_D9	145	B_A1	66	B_REG	63	V_{CC}	86
A_D10	147	B_A2	65	B_RESET	58	V_{CC}	113
A_D11	84	B_A3	62	B_VS1	68	V_{CC}	143
A_D12	87	B_A4	60	B_VS2	56	V_{CC}	164
A_D13	89	B_A5	59	B_WAIT	70	V_{CC}	175
A_D14	91	B_A6	57	B_WE	46	V_{CC}	187
A_D15	93	B_A7	54	B_WP(IOSI16)	73	V_{CC}	201
A_INPACK	127	B_A8	39	C/BE0	5	V_{CCA}	120
A_IORD	99	B_A9	36	C/BE1	203	V_{CCB}	38
A_IOWR	101	B_A10	29	C/BE2	192	V_{CCP}	148
A_OE	98	B_A11	34	C/BE3	180	V_{CCP}	172

PRODUCT PREVIEW

PRODUCT PREVIEW

Table 2. Signal Names Sorted Alphabetically – CardBus PC Card

SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.
A_CAD0	83	A_CSTOP	109	B_CAD12	34	CLOCK	151
A_CAD1	85	A_CSTSCHG	138	B_CAD13	33	DATA	152
A_CAD2	84	A_CTRDY	114	B_CAD14	36	DEVSEL	197
A_CAD3	88	A_CVS1	134	B_CAD15	35	FRAME	193
A_CAD4	87	A_CVS2	122	B_CAD16	37	GND	13
A_CAD5	90	A_RSVD	91	B_CAD17	53	GND	22
A_CAD6	89	A_RSVD	105	B_CAD18	54	GND	44
A_CAD7	92	A_RSVD	146	B_CAD19	55	GND	75
A_CAD8	93	AD0	15	B_CAD20	57	GND	96
A_CAD9	95	AD1	14	B_CAD21	59	GND	129
A_CAD10	97	AD2	12	B_CAD22	60	GND	153
A_CAD11	98	AD3	11	B_CAD23	62	GND	167
A_CAD12	100	AD4	10	B_CAD24	65	GND	181
A_CAD13	99	AD5	9	B_CAD25	66	GND	194
A_CAD14	102	AD6	8	B_CAD26	67	GND	207
A_CAD15	101	AD7	6	B_CAD27	76	GNT	168
A_CAD16	103	AD8	4	B_CAD28	77	IDSEL	182
A_CAD17	118	AD9	3	B_CAD29	78	IRDY	195
A_CAD18	119	AD10	2	B_CAD30	79	IRQ3/INTA	154
A_CAD19	121	AD11	1	B_CAD31	81	IRQ4/INTB	155
A_CAD20	123	AD12	208	B_CAUDIO	71	IRQ5	156
A_CAD21	125	AD13	206	B_CBLOCK	42	IRQ7/PCDMAREQ	157
A_CAD22	126	AD14	205	B_CC/BE0	28	IRQ9/IRQSER	158
A_CAD23	128	AD15	204	B_CC/BE1	39	IRQ10/CLKRUN	159
A_CAD24	131	AD16	191	B_CC/BE2	52	IRQ11/PCDMAGNT	160
A_CAD25	132	AD17	190	B_CC/BE3	63	IRQ12	161
A_CAD26	133	AD18	189	B_CCD1	16	IRQ14	162
A_CAD27	141	AD19	188	B_CCD2	74	IRQ15/RI_OUT	163
A_CAD28	142	AD20	186	B_CCLK	48	LATCH	150
A_CAD29	144	AD21	185	B_CCLKRUN	73	PAR	202
A_CAD30	145	AD22	184	B_CDEVSEL	47	PCLK	165
A_CAD31	147	AD23	183	B_CFRAME	51	PERR	199
A_CAUDIO	137	AD24	179	B_CGNT	46	REQ	169
A_CBLOCK	107	AD25	178	B_CINT	69	RSTIN	166
A_CC/BE0	94	AD26	177	B_CIRDY	50	SPKROUT/SUSPEND	149
A_CC/BE1	104	AD27	176	B_CPAR	41	STOP	198
A_CC/BE2	117	AD28	174	B_CPERR	43	SERR	200
A_CC/BE3	130	AD29	173	B_CREQ	61	TRDY	196
A_CCD1	82	AD30	171	B_CRST	58	V_{CC}	7
A_CCD2	140	AD31	170	B_CSERR	70	V_{CC}	31
A_CCLK	112	B_CAD0	17	B_CSTOP	45	V_{CC}	64
A_CCLKRUN	139	B_CAD1	19	B_CSTSCHG	72	V_{CC}	86
A_CDEVSEL	111	B_CAD2	18	B_CTRDY	49	V_{CC}	113
A_CFRAME	116	B_CAD3	21	B_CVS1	68	V_{CC}	143
A_CGNT	110	B_CAD4	20	B_CVS2	56	V_{CC}	164
A_CINT	135	B_CAD5	24	B_RSVD	25	V_{CC}	175
A_CIRDY	115	B_CAD6	23	B_RSVD	40	V_{CC}	187
A_CPAR	106	B_CAD7	26	B_RSVD	80	V_{CC}	201
A_CPERR	108	B_CAD8	27	C/BE0	5	V_{CCA}	120
A_CREQ	127	B_CAD9	29	C/BE1	203	V_{CCB}	38
A_CRST	124	B_CAD10	30	C/BE2	192	V_{CCP}	148
A_CSERR	136	B_CAD11	32	C/BE3	180	V_{CCP}	172

Table 3. Signal Names Sorted by Pin Number – 16-bit PC Card

NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME
1	AD11	53	B_A24	105	A_A18	157	IRQ7/PCDMAREQ
2	AD10	54	B_A7	106	A_A13	158	IRQ9/IRQSER
3	AD9	55	B_A25	107	A_A19	159	IRQ10/CLKRUN
4	AD8	56	B_VS2	108	A_A14	160	IRQ11/PCDMAGNT
5	C/BE0	57	B_A6	109	A_A20	161	IRQ12
6	AD7	58	B_RESET	110	A_WE	162	IRQ14
7	V_{CC}	59	B_A5	111	A_A21	163	IRQ15/RI_OUT
8	AD6	60	B_A4	112	A_A16	164	V_{CC}
9	AD5	61	B_INPACK	113	V_{CC}	165	PCLK
10	AD4	62	B_A3	114	A_A22	166	RSTIN
11	AD3	63	B_REG	115	A_A15	167	GND
12	AD2	64	V_{CC}	116	A_A23	168	GNT
13	GND	65	B_A2	117	A_A12	169	REQ
14	AD1	66	B_A1	118	A_A24	170	AD31
15	AD0	67	B_A0	119	A_A7	171	AD30
16	B_CD1	68	B_VS1	120	V_{CCA}	172	V_{CCP}
17	B_D3	69	B_READY(IREQ)	121	A_A25	173	AD29
18	B_D11	70	B_WAIT	122	A_VS2	174	AD28
19	B_D4	71	B_BVD2(SPKR)	123	A_A6	175	V_{CC}
20	B_D12	72	B_BVD1(STSCHG/RI)	124	A_RESET	176	AD27
21	B_D5	73	B_WP(IOIS16)	125	A_A5	177	AD26
22	GND	74	B_CD2	126	A_A4	178	AD25
23	B_D13	75	GND	127	A_INPACK	179	AD24
24	B_D6	76	B_D0	128	A_A3	180	C/BE3
25	B_D14	77	B_D8	129	GND	181	GND
26	B_D7	78	B_D1	130	A_REG	182	IDSEL
27	B_D15	79	B_D9	131	A_A2	183	AD23
28	B_CE1	80	B_D2	132	A_A1	184	AD22
29	B_A10	81	B_D10	133	A_A0	185	AD21
30	B_CE2	82	A_CD1	134	A_VS1	186	AD20
31	V_{CC}	83	A_D3	135	A_READY(IREQ)	187	V_{CC}
32	B_OE	84	A_D11	136	A_WAIT	188	AD19
33	B_IORD	85	A_D4	137	A_BVD2(SPKR)	189	AD18
34	B_A11	86	V_{CC}	138	A_BVD1(STSCHG/RI)	190	AD17
35	B_IOWR	87	A_D12	139	A_WP(IOIS16)	191	AD16
36	B_A9	88	A_D5	140	A_CD2	192	C/BE2
37	B_A17	89	A_D13	141	A_D0	193	FRAME
38	V_{CCB}	90	A_D6	142	A_D8	194	GND
39	B_A8	91	A_D14	143	V_{CC}	195	IRDY
40	B_A18	92	A_D7	144	A_D1	196	TRDY
41	B_A13	93	A_D15	145	A_D9	197	DEVSEL
42	B_A19	94	A_CE1	146	A_D2	198	STOP
43	B_A14	95	A_A10	147	A_D10	199	PERR
44	GND	96	GND	148	V_{CCP}	200	SERR
45	B_A20	97	A_CE2	149	SPKROUT/SUSPEND	201	V_{CC}
46	B_WE	98	A_OE	150	LATCH	202	PAR
47	B_A21	99	A_IORD	151	CLOCK	203	C/BE1
48	B_A16	100	A_A11	152	DATA	204	AD15
49	B_A22	101	A_IOWR	153	GND	205	AD14
50	B_A15	102	A_A9	154	IRQ3/INTA	206	AD13
51	B_A23	103	A_A17	155	IRQ4/INTB	207	GND
52	B_A12	104	A_A8	156	IRQ5	208	AD12

PRODUCT PREVIEW

PRODUCT PREVIEW

Table 4. Signal Names Sorted by Pin Number – CardBus PC Card

NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME
1	AD11	53	B_CAD17	105	A_RSVD	157	IRQ7/PCDMAREQ
2	AD10	54	B_CAD18	106	A_CPAR	158	IRQ9/IRQSER
3	AD9	55	B_CAD19	107	A_CBLOCK	159	IRQ10/CLKRUN
4	AD8	56	B_CVS2	108	A_CPERR	160	IRQ11/PCDMAGNT
5	C/BE0	57	B_CAD20	109	A_CSTOP	161	IRQ12
6	AD7	58	B_CRST	110	A_CGNT	162	IRQ14
7	V_{CC}	59	B_CAD21	111	A_CDEVSEL	163	IRQ15/RI_OUT
8	AD6	60	B_CAD22	112	A_CCLK	164	V_{CC}
9	AD5	61	B_CREQ	113	V_{CC}	165	PCLK
10	AD4	62	B_CAD23	114	A_CTRDY	166	RSTIN
11	AD3	63	B_CC/BE3	115	A_CIRDY	167	GND
12	AD2	64	V_{CC}	116	A_CFRAME	168	GNT
13	GND	65	B_CAD24	117	A_CC/BE2	169	REQ
14	AD1	66	B_CAD25	118	A_CAD17	170	AD31
15	AD0	67	B_CAD26	119	A_CAD18	171	AD30
16	B_CCD1	68	B_CVS1	120	V_{CCA}	172	V_{CCP}
17	B_CAD0	69	B_CINT	121	A_CAD19	173	AD29
18	B_CAD2	70	B_CSERR	122	A_CVS2	174	AD28
19	B_CAD1	71	B_CAUDIO	123	A_CAD20	175	V_{CC}
20	B_CAD4	72	B_CSTSCHG	124	A_CRST	176	AD27
21	B_CAD3	73	B_CCLKRUN	125	A_CAD21	177	AD26
22	GND	74	B_CCD2	126	A_CAD22	178	AD25
23	B_CAD6	75	GND	127	A_CREQ	179	AD24
24	B_CAD5	76	B_CAD27	128	A_CAD23	180	C/BE3
25	B_RSVD	77	B_CAD28	129	GND	181	GND
26	B_CAD7	78	B_CAD29	130	A_CC/BE3	182	IDSEL
27	B_CAD8	79	B_CAD30	131	A_CAD24	183	AD23
28	B_CC/BE0	80	B_RSVD	132	A_CAD25	184	AD22
29	B_CAD9	81	B_CAD31	133	A_CAD26	185	AD21
30	B_CAD10	82	A_CCD1	134	A_CVS1	186	AD20
31	V_{CC}	83	A_CAD0	135	A_CINT	187	V_{CC}
32	B_CAD11	84	A_CAD2	136	A_CSERR	188	AD19
33	B_CAD13	85	A_CAD1	137	A_CAUDIO	189	AD18
34	B_CAD12	86	V_{CC}	138	A_CSTSCHG	190	AD17
35	B_CAD15	87	A_CAD4	139	A_CCLKRUN	191	AD16
36	B_CAD14	88	A_CAD3	140	A_CCD2	192	C/BE2
37	B_CAD16	89	A_CAD6	141	A_CAD27	193	FRAME
38	V_{CCB}	90	A_CAD5	142	A_CAD28	194	GND
39	B_CC/BE1	91	A_RSVD	143	V_{CC}	195	IRDY
40	B_RSVD	92	A_CAD7	144	A_CAD29	196	TRDY
41	B_CPAR	93	A_CAD8	145	A_CAD30	197	DEVSEL
42	B_CBLOCK	94	A_CC/BE0	146	A_RSVD	198	STOP
43	B_CPERR	95	A_CAD9	147	A_CAD31	199	PERR
44	GND	96	GND	148	V_{CCP}	200	SERR
45	B_CSTOP	97	A_CAD10	149	SPKROUT/SUSPEND	201	V_{CC}
46	B_CGNT	98	A_CAD11	150	LATCH	202	PAR
47	B_CDEVSEL	99	A_CAD13	151	CLOCK	203	C/BE1
48	B_CCLK	100	A_CAD12	152	DATA	204	AD15
49	B_CTRDY	101	A_CAD15	153	GND	205	AD14
50	B_CIRDY	102	A_CAD14	154	IRQ3/INTA	206	AD13
51	B_CFRAME	103	A_CAD16	155	IRQ4/INTB	207	GND
52	B_CC/BE2	104	A_CC/BE1	156	IRQ5	208	AD12

Terminal Functions

PCI system

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
PCLK	165	I	PCI bus clock. PCLK provides timing for all transactions on the PCI bus. All PCI signals are sampled at the rising edge of PCLK.
$\overline{\text{RSTIN}}$	166	I	PCI reset. When the $\overline{\text{RSTIN}}$ signal is asserted low, the PCI1130 forces all output buffers to the high-impedance state and resets all internal registers. When asserted, the PCI1130 is nonfunctional. After deasserting $\overline{\text{RSTIN}}$, the PCI1130 returns to the default state. When the PCI1130 $\overline{\text{SUSPEND}}$ mode is enabled, the device is protected from any $\overline{\text{RSTIN}}$ reset (i.e., the PCI1130 internal register contents are preserved). See *power management*.

PCI address and data

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
AD31	170	I/O	Address/data bus. AD31–AD0 are the multiplexed PCI address and data bus. During the address phase of a PCI cycle, AD31–AD0 contain a 32-bit address or other destination information. During the data phase, AD31–AD0 contain data.
AD30	171		
AD29	173		
AD28	174		
AD27	176		
AD26	177		
AD25	178		
AD24	179		
AD23	183		
AD22	184		
AD21	185		
AD20	186		
AD19	188		
AD18	189		
AD17	190		
AD16	191		
AD15	204		
AD14	205		
AD13	206		
AD12	208		
AD11	1		
AD10	2		
AD9	3		
AD8	4		
AD7	6		
AD6	8		
AD5	9		
AD4	10		
AD3	11		
AD2	12		
AD1	14		
AD0	15		
C/$\overline{\text{BE3}}$	180	I/O	Bus commands and byte enables. C/$\overline{\text{BE3}}$–C/$\overline{\text{BE0}}$ are multiplexed on the same PCI terminals. During the address phase, C/$\overline{\text{BE3}}$–C/$\overline{\text{BE0}}$ define the bus command. During the data phase, C/$\overline{\text{BE3}}$–C/$\overline{\text{BE0}}$ are used as byte enables. The byte enables determine which byte lanes carry meaningful data. C/$\overline{\text{BE0}}$ applies to byte 0 (AD7–AD0), C/$\overline{\text{BE1}}$ applies to byte 1 (AD15–AD8), C/$\overline{\text{BE2}}$ applies to byte 2 (AD23–AD16), and C/$\overline{\text{BE3}}$ applies to byte 3 (AD31–AD24).
C/$\overline{\text{BE2}}$	192		
C/$\overline{\text{BE1}}$	203		
C/$\overline{\text{BE0}}$	5		
PAR	202	I/O	Parity. As a PCI target during PCI read cycles, or as PCI bus master during PCI write cycles, the PCI1130 calculates even parity across the AD and C/$\overline{\text{BE}}$ buses and outputs the results on PAR, delayed by one clock.

PRODUCT PREVIEW

Terminal Functions (Continued)

PCI interface control

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
$\overline{\text{DEVSEL}}$	197	I/O	Device select. As a PCI target, the PCI1130 asserts $\overline{\text{DEVSEL}}$ to claim the current cycle. As a PCI master, the PCI1130 monitors $\overline{\text{DEVSEL}}$ until a target responds or a time-out occurs.
$\overline{\text{FRAME}}$	193	I/O	Cycle frame. $\overline{\text{FRAME}}$ is driven by the current master to indicate the beginning and duration of an access, $\overline{\text{FRAME}}$ is low (asserted) to indicate that a bus transaction is beginning. While $\overline{\text{FRAME}}$ is asserted, data transfers continue. When $\overline{\text{FRAME}}$ is sampled high (deasserted), the transaction is in the final data phase.
$\overline{\text{GNT}}$	168	I	Grant. $\overline{\text{GNT}}$ is driven by the PCI arbiter to grant the PCI1130 access to the PCI bus after the current data transaction is complete.
IDSEL	182	I	Initialization device select. IDSEL selects the PCI1130 during configuration accesses. IDSEL can be connected to one of the upper 24 PCI address lines.
$\overline{\text{IRDY}}$	195	I/O	Initiator ready. $\overline{\text{IRDY}}$ indicates the bus master's ability to complete the current data phase of the transaction. $\overline{\text{IRDY}}$ is used with $\overline{\text{TRDY}}$. A data phase is completed on any clock where both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are sampled low (asserted). During a write, $\overline{\text{IRDY}}$ indicates that valid data is present on AD31–AD0. During a read, $\overline{\text{IRDY}}$ indicates that the master is prepared to accept data. Wait cycles are inserted until both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are low (asserted) at the same time. This signal is an output when the PCI1130 is the PCI bus master and an input when the PCI bus is the target.
$\overline{\text{PERR}}$	199	I/O	Parity error. $\overline{\text{PERR}}$ is driven by the PCI target during a write to indicate that a data parity error has been detected.
$\overline{\text{REQ}}$	169	O	Request. $\overline{\text{REQ}}$ is asserted by the PCI1130 to request access to the PCI bus as a master.
$\overline{\text{SERR}}$	200	O	System error. $\overline{\text{SERR}}$ pulsed from the PCI1130 indicates an address parity error has occurred.
$\overline{\text{STOP}}$	198	I/O	Stop. $\overline{\text{STOP}}$ is driven by the current PCI target to request the master to stop the current transaction.
$\overline{\text{TRDY}}$	196	I/O	Target ready. $\overline{\text{TRDY}}$ indicates the ability of the PCI1130 to complete the current data phase of the transaction. $\overline{\text{TRDY}}$ is used with $\overline{\text{IRDY}}$. A data phase is completed on any clock where both $\overline{\text{TRDY}}$ and $\overline{\text{IRDY}}$ are sampled asserted. During a read, $\overline{\text{TRDY}}$ indicates that valid data is present on AD31–AD0. During a write, $\overline{\text{TRDY}}$ indicates that the PCI1130 is prepared to accept data. Wait cycles are inserted until both $\overline{\text{IRDY}}$ and $\overline{\text{TRDY}}$ are asserted together. This signal is an output when the PCI1130 is the PCI target and an input when the PCI1130 is the PCI bus master.

Terminal Functions (Continued)

16-bit PC Card address and data (slots A and B)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	FUNCTION
A25	121	55	O	PC Card address. 16-bit PC Card address lines. A25 is the most-significant bit.
A24	118	53		
A23	116	51		
A22	114	49		
A21	111	47		
A20	109	45		
A19	107	42		
A18	105	40		
A17	103	37		
A16	112	48		
A15	115	50		
A14	108	43		
A13	106	41		
A12	117	52		
A11	100	34		
A10	95	29		
A9	102	36		
A8	104	39		
A7	119	54		
A6	123	57		
A5	125	59		
A4	126	60		
A3	128	62		
A2	131	65		
A1	132	66		
A0	133	67		
D15	93	27	I/O	PC Card data. 16-bit PC Card data lines. D15 is the most-significant bit.
D14	91	25		
D13	89	23		
D12	87	20		
D11	84	18		
D10	147	81		
D9	145	79		
D8	142	77		
D7	92	26		
D6	90	24		
D5	88	21		
D4	85	19		
D3	83	17		
D2	146	80		
D1	144	78		
D0	141	76		

† Terminal name is preceded with A_. For example, the full name for terminal 121 is A_A25.
‡ Terminal name is preceded with B_. For example, the full name for terminal 55 is B_A25.

Terminal Functions (Continued)

16-bit PC Card interface control signals (slots A and B)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	FUNCTION
BVD1 (STSCHG/$\overline{RI}$)	138	72	I	Battery voltage detect 1. 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 needs to 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* for enable bits. See *ExCA card status-change register* and *ExCA interface status register* for the status bits for this signal. 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. Ring indicate. $\overline{RI}$ is used by 16-bit modem cards to indicate ring detection.
BVD2($\overline{SPKR}$)	137	71	I	Battery voltage detect 2. 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 needs to 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* for enable bits. See *ExCA card status-change register* and *ExCA interface status register* for the status bits for this signal. Speaker. $\overline{SPKR}$ 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 can be combined by the PCI1130 and output on $\overline{SPKROUT}$. DMA request. BVD2 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 BVD2 to request a DMA operation.
$\overline{CD1}$ $\overline{CD2}$	82 140	16 74	I	PC Card detect 1 and PC Card detect 2. $\overline{CD1}$ and $\overline{CD2}$ are internally connected to ground on the PC Card. When a PC Card is inserted into a socket, $\overline{CD1}$ and $\overline{CD2}$ are pulled low. For signal status, see *ExCA interface status register.*
$\overline{CE1}$ $\overline{CE2}$	94 97	28 30	O	Card enable 1 and card enable 2. $\overline{CE1}$ and $\overline{CE2}$ enable even- and odd-numbered address bytes. $\overline{CE1}$ enables even-numbered address bytes, and $\overline{CE2}$ enables odd-numbered address bytes.
INPACK	127	61	I	Input acknowledge. INPACK is asserted by the PC Card when it can respond to an I/O read cycle at the current address. DMA request. INPACK 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 INPACK to indicate a request for a DMA operation.
$\overline{IORD}$	99	33	O	I/O read. $\overline{IORD}$ is asserted by the PCI1130 to enable 16-bit I/O PC Card data output during host I/O read cycles. DMA write. $\overline{IORD}$ is used as the DMA write strobe during DMA operations from a 16-bit PC Card that supports DMA. The PCI1130 asserts $\overline{IORD}$ during DMA transfers from the PC Card to host memory.
$\overline{IOWR}$	101	35	O	I/O write. $\overline{IOWR}$ is driven low by the PCI1130 to strobe write data into 16-bit I/O PC Cards during host I/O write cycles. DMA read. $\overline{IOWR}$ is used as the DMA read strobe during DMA operations to a 16-bit PC Card that supports DMA. The PCI1130 asserts $\overline{IOWR}$ during DMA transfers from host memory to the PC Card.
$\overline{OE}$	98	32	O	Output enable. $\overline{OE}$ is driven low by the PCI1130 to enable 16-bit memory PC Card data output during host memory read cycles. DMA terminal count. $\overline{OE}$ is used as terminal count (TC) during DMA operations to a 16-bit PC Card that supports DMA. The PCI1130 asserts $\overline{OE}$ to indicate TC for a DMA write operation.

† Terminal name is preceded with A_. For example, the full name for terminal 138 is A_BVD1.
‡ Terminal name is preceded with B_. For example, the full name for terminal 72 is B_BVD1.

PRODUCT PREVIEW

Terminal Functions (Continued)

16-bit PC Card interface control signals (slots A and B) (continued)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	FUNCTION
READY(IREQ)	135	69	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. Interrupt request. 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. IREQ is high (deasserted) when no interrupt is requested.
REG	130	63	O	Attribute memory select. REG remains high for all common memory accesses. When REG is asserted, access is limited to attribute memory (OE or WE active) and to the I/O space (IORD or 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. DMA acknowledge. REG is used as a DMA acknowledge (DACK) during DMA operations to a 16-bit PC Card that supports DMA. The PCI1130 asserts REG to indicate a DMA operation. REG is used with the DMA read (IOWR) or DMA write (IORD) strobes to transfer data.
RESET	124	58	O	PC Card reset. RESET forces a hard reset to a 16-bit PC Card.
WAIT	136	70	I	Bus cycle wait. 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.
WE	110	46	O	Write enable. WE is used to strobe memory write data into 16-bit memory PC Cards. WE also is used for memory PC Cards that employ programmable memory technologies. DMA terminal count. WE is used as TC during DMA operations to a 16-bit PC Card that supports DMA. The PCI1130 asserts WE to indicate TC for a DMA read operation.
WP(IOIS16)	139	73	I	Write protect. This signal 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 (IOIS16) function. The status of WP can be read from the ExCA interface status register. I/O is 16 bits. WP applies to 16-bit I/O PC Cards. 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. 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 request a DMA operation.
VS1 VS2	134 122	68 56	I/O	Voltage sense 1 and voltage sense 2. VS1 and VS2, when used together, determine the operating voltage of the 16-bit PC Card.

† Terminal name is preceded with A_. For example, the full name for terminal 98 is A_OE.
‡ Terminal name is preceded with B_. For example, the full name for terminal 32 is B_OE.

Terminal Functions (Continued)

CardBus PC Card address and data signals (slots A and B)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	FUNCTION
CAD31	147	81	I/O	CardBus PC Card address and data. CAD31–CAD0 are multiplexed address and data signals. A bus transaction consists of an address phase followed by one or more data phases. The PCI1130 supports both read and write bursts. The address phase is the clock cycle in which $\overline{\text{CFRAME}}$ is asserted. During the address phase, CAD31-CAD0 contain a physical address (32 bits). For I/O, this is a byte address; for configuration and memory, it is a DWORD address. During data phases, CAD7–CAD0 contain the least-significant byte and CAD31–CAD24 contain the most-significant byte. Write data is stable and valid when $\overline{\text{CIRDY}}$ is asserted. Read data is stable and valid when $\overline{\text{CTRDY}}$ is asserted. Data is transferred during those clocks when $\overline{\text{CIRDY}}$ and $\overline{\text{CTRDY}}$ are asserted.
CAD30	145	79		
CAD29	144	78		
CAD28	142	77		
CAD27	141	76		
CAD26	133	67		
CAD25	132	66		
CAD24	131	65		
CAD23	128	62		
CAD22	126	60		
CAD21	125	59		
CAD20	123	57		
CAD19	121	55		
CAD18	119	54		
CAD17	118	53		
CAD16	103	37		
CAD15	101	35		
CAD14	102	36		
CAD13	99	33		
CAD12	100	34		
CAD11	98	32		
CAD10	97	30		
CAD9	95	29		
CAD8	93	27		
CAD7	92	26		
CAD6	89	23		
CAD5	90	24		
CAD4	87	20		
CAD3	88	21		
CAD2	84	18		
CAD1	85	19		
CAD0	83	17		
CC/$\overline{\text{BE0}}$	94	28	I/O	CardBus PC Card command and byte enables. CC/BE0–CC/BE3 are multiplexed on the same pin. During the address phase of the transaction, CC/$\overline{\text{BE3}}$–CC/$\overline{\text{BE0}}$ define the bus command. During the data phase transaction, CC/$\overline{\text{BE3}}$–CC/$\overline{\text{BE0}}$ are used as byte enables. Byte enables are valid during the entire data phase and determine the byte lanes that carry the data. CC/$\overline{\text{BE0}}$ applies to byte 0, CC/$\overline{\text{BE1}}$ applies to byte 1, CC/$\overline{\text{BE2}}$ applies to byte 2, and CC/$\overline{\text{BE3}}$ applies to byte 3.
CC/$\overline{\text{BE1}}$	104	39		
CC/$\overline{\text{BE2}}$	117	52		
CC/$\overline{\text{BE3}}$	130	63		
CPAR	106	41	I/O	CardBus PC Card parity. Even parity across CAD31–CAD0 and CC/$\overline{\text{BE3}}$–CC/$\overline{\text{BE0}}$ is calculated and driven by this signal. CPAR is stable and valid for one clock after the address phase. For data phases, CPAR is stable and valid one clock after either $\overline{\text{CIRDY}}$ is asserted on a write transaction or $\overline{\text{CTRDY}}$ is asserted on a read transaction. Once CPAR is valid, it remains valid for one clock after the completion of the current data phase. NOTE: CPAR has the same timing as CAD31–CAD0 but delays by one clock. When the PCI1130 is acting as an initiator, it drives CPAR for address and write data phases; and when acting as a target, the PCI1130 drives CPAR for read data phases.

† Terminal name is preceded with A_. For example, the full name for terminal 147 is A_CAD31.
‡ Terminal name is preceded with B_. For example, the full name for terminal 81 is B_CAD31.

PRODUCT PREVIEW

Terminal Functions (Continued)

CardBus PC Card interface system signals (slots A and B)

NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	FUNCTION
CCLK	112	48	O	CardBus PC card clock. CCLK provides synchronous timing for all transactions on the CardBus PC Card interface. All signals except $\overline{\text{CRST}}$ (upon assertion) $\overline{\text{CCLKRUN}}$, $\overline{\text{CINT}}$, CSTSCHG, CAUDIO, $\overline{\text{CCD2}}$–$\overline{\text{CCD1}}$, and CVS2–CVS1 are sampled on the rising edge of the clock, and all timing parameters are defined with the rising edge of CCLK. The CardBus clock operates at 33 MHz but can be stopped in the low state.
$\overline{\text{CCLKRUN}}$	139	73	I/O	CardBus PC Card clock run. $\overline{\text{CCLKRUN}}$ is used by a CardBus PC Card to request an increase in the CCLK frequency. It is used by the PCI1130 to indicate that the CCLK frequency is decreased.
$\overline{\text{CRST}}$	124	58	O	CardBus PC Card reset. $\overline{\text{CRST}}$ is used to bring CardBus PC Card specific registers, sequencers, and signals to the a consistent state. When $\overline{\text{CRST}}$ is asserted, all CardBus PC Card signals must be driven to the high-impedance state, but the PCI1130 drives these signals to a valid logic level. Assertion can be asynchronous to CCLK, but deassertion must be synchronous to CCLK.

CardBus PC Card interface control signals (slots A and B)

NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	FUNCTION
CAUDIO	137	71	I	CardBus audio. CAUDIO is an optional digital input signal from a PC Card to the system speaker. CardBus cards support two types of audio: single amplitude, binary waveform and/or pulsewidth modulation (PWM) encoded signal. The PCI1130 supports the binary audio mode and can output a binary audio signal from the PC Card to $\overline{\text{SPKROUT}}$.
$\overline{\text{CBLOCK}}$	107	42	I/O	CardBus lock. $\overline{\text{CBLOCK}}$ is an optional signal used to lock a particular address, ensuring a bus initiator exclusive access. This signal is not supported on the PCI1130.
$\overline{\text{CCD1}}$ $\overline{\text{CCD2}}$	82 140	16 74	I	CardBus detect 1 and CardBus detect 2. $\overline{\text{CCD1}}$ and $\overline{\text{CCD2}}$ are used with CVS1 and CVS2 to determine the type and voltage of the CardBus PC Card. For signal status, see *ExCA interface status register.*
$\overline{\text{CDEVSEL}}$	111	47	I/O	CardBus device select. When actively driven, $\overline{\text{CDEVSEL}}$ indicates that the PCI1130 has decoded its address as the target of the current access. As an input, $\overline{\text{CDEVSEL}}$ indicates whether any device on the bus has been selected.
$\overline{\text{CFRAME}}$	116	51	I/O	CardBus cycle frame. $\overline{\text{CFRAME}}$ is driven by the PCI1130 when it is acting as an initiator to indicate the beginning and duration of a transaction. $\overline{\text{CFRAME}}$ is asserted to indicate a bus transaction is beginning, and while it is asserted, data transfer is continuous. When $\overline{\text{CFRAME}}$ is high (deasserted), the transaction is in its final data phase.
$\overline{\text{CGNT}}$	110	46	O	CardBus grant. $\overline{\text{CGNT}}$ is driven by the PCI1130 to grant a CardBus PC Card access to the CardBus bus after after the current data transaction is complete.
$\overline{\text{CINT}}$	135	69	I	CardBus interrupt. $\overline{\text{CINT}}$ is asserted low by a CardBus PC Card to request interrupt servicing from the host.

† Terminal name is preceded with A_. For example, the full name for terminal 112 is A_CCLK.
‡ Terminal name is preceded with B_. For example, the full name for terminal 48 is B_CCLK.

PRODUCT PREVIEW

Terminal Functions (Continued)

CardBus PC Card interface control signals (slots A and B) (continued)

TERMINAL NAME	TERMINAL NUMBER SLOT A†	TERMINAL NUMBER SLOT B‡	I/O TYPE	FUNCTION
$\overline{\text{CIRDY}}$	115	50	I/O	CardBus initiator ready. $\overline{\text{CTRDY}}$ indicates that the PCI1130 is initiating the ability of the bus initiator to complete a current data phase of the transaction. It is used with $\overline{\text{CTRDY}}$. When both $\overline{\text{CTRDY}}$ and $\overline{\text{CTRDY}}$ are sampled asserted, a data phase is completed on any clock. During a write, $\overline{\text{CIRDY}}$ indicates that valid data is present on CAD31–CAD0. During a read, $\overline{\text{CIRDY}}$ indicates the PCI1130, as an initiator, is prepared to accept the data. Wait cycles are inserted until $\overline{\text{CTRDY}}$ and $\overline{\text{CTRDY}}$ are both low (asserted).
$\overline{\text{CPERR}}$	108	43	I/O	CardBus parity error. $\overline{\text{CPERR}}$ reports errors during all CardBus PC Card transactions except during special cycles. $\overline{\text{CPERR}}$ is sustained in the high-impedance state and must be driven active by the agent receiving data, two clocks following the data, when a data parity error is detected. $\overline{\text{CPERR}}$ must be driven active for a minimum duration of one clock for each data phase. $\overline{\text{CPERR}}$ must be driven high for one clock before it is returned to the high-impedance state. An agent cannot report a $\overline{\text{CPERR}}$ until it claims the access by asserting $\overline{\text{CDEVSEL}}$ and completes a data phase.
$\overline{\text{CREQ}}$	127	61	I	CardBus request. $\overline{\text{CREQ}}$ indicates to the arbiter that the CardBus PC Card requires use of the CardBus bus.
$\overline{\text{CSERR}}$	136	70	I	CardBus system error. $\overline{\text{CSERR}}$ reports address parity error, data errors on the special cycle command, or any other system error such that the CardBus card can no longer operate correctly. $\overline{\text{CSERR}}$ is open drain and is actively driven for a single CardBus PC Card clock by the agent reporting the error. The assertion of $\overline{\text{CSERR}}$ is synchronous to the clock and meets the setup and hold times of all bused signals. Restoring $\overline{\text{CSERR}}$ to the deasserted state is accomplished by a weak pullup provided by the system designer. This pullup can take two to three clock periods to fully restore $\overline{\text{CSERR}}$. The PCI1130 reports $\overline{\text{CSERR}}$ to the operating system any time it is sampled low (asserted).
$\overline{\text{CSTOP}}$	109	45	I/O	CardBus stop. $\overline{\text{CSTOP}}$ indicates the current target is requesting the initiator to stop the current transaction.
CSTSCHG	138	72	I	CardBus status change. CSTSCHG is used to alert the system to a change in the READY, WP, or BVD condition of the I/O CardBus PC Card.
$\overline{\text{CTRDY}}$	114	49	I/O	CardBus target ready. $\overline{\text{CTRDY}}$ indicates that the PCI1130, as a selected target, can complete a current data phase of the transaction. $\overline{\text{CTRDY}}$ is used with $\overline{\text{CIRDY}}$. When both of these signals are sampled asserted, a data phase is completed on any clock. During a read, $\overline{\text{CTRDY}}$ indicates that valid data is present on CAD31–CAD0. During a write, $\overline{\text{CIRDY}}$ indicates the PCI1130, as a target, is prepared to accept the data. Wait cycles are inserted until $\overline{\text{CIRDY}}$ and $\overline{\text{CTRDY}}$ are both low (asserted).
CVS1 CVS2	134 122	68 56	I/O	CardBus voltage sense 1 and voltage sense 2. CVS1 and CVS2, together with $\overline{\text{CCD1}}$ and $\overline{\text{CCD2}}$, determine the operating voltage of the CardBus PC Card.

† Terminal name is preceded with A_. For example, the full name for terminal 115 is A_$\overline{\text{CIRDY}}$.
‡ Terminal name is preceded with B_. For example, the full name for terminal 50 is B_$\overline{\text{CIRDY}}$.

PRODUCT PREVIEW

Terminal Functions (Continued)

interrupt

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
IRQ3/INTA IRQ4/INTB	154 155	O	Interrupt request 3 and interrupt request 4. IRQ3/INTA–IRQ4/INTB can be connected to either PCI or ISA interrupts. IRQ3/INTA–IRQ4/INTB are software configurable as IRQ3 or INTA and as IRQ4 or INTB. When configured for IRQ3 and IRQ4, IRQ3/INTA–IRQ4/INTB must be connected to the ISA IRQ programmable interrupt controller. When IRQ3/INTA–IRQ4/INTB are configured for INTA and INTB, IRQ3/INTA–IRQ4/INTB must be connected to available interrupts on the PCI bus.
IRQ7/$\overline{\text{PCDMAREQ}}$	157	O	Interrupt request 7. IRQ7/$\overline{\text{PCDMAREQ}}$ is software configurable and is used by the PCI1130 to request PC/PCI DMA transfers from chipsets that support the PC/PCI DMA scheme. When IRQ7/$\overline{\text{PCDMAREQ}}$ is configured for PC/PCI DMA request (IRQ7), it must be connected to the appropriate request ($\overline{\text{REQ}}$) pin on the Intel Mobile Trition PCI I/O accelerator (MPIIX™) (see *PC/PCI DMA*).
IRQ9/IRQSER	158	O	Interrupt request 9. IRQ9/IRQSER is software configurable and indicates an interrupt request from one of the PC Cards. When IRQ9/IRQSER is configured for IRQ9, it must be connected to the IRQ programmable interrupt controller. IRQSER allows all IRQ signals to be serialized onto one pin. This signal is configured in the device control register of the TI extension registers (see *device control register*).
IRQ10/$\overline{\text{CLKRUN}}$	159	O	Interrupt requests 10. IRQ10/$\overline{\text{CLKRUN}}$ is software configurable and is used by the PCI1130 to support the PCI $\overline{\text{CLKRUN}}$ protocol. When configured as $\overline{\text{CLKRUN}}$ by setting bit 0 in the system control register 80h, this terminal is an open drain output.
IRQ11/$\overline{\text{PCDMAGNT}}$	160	I/O	Interrupt request 11. IRQ11/$\overline{\text{PCDMAGNT}}$ is software configurable and is used by the PCI1130 to accept a grant for PC/PCI DMA transfers from chipsets that support the PC/PCI DMA scheme. When IRQ11/$\overline{\text{PCDMAGNT}}$ is configured for PC/PCI DMA grant (IRQ11), it must be connected to the appropriate grant ($\overline{\text{GNT}}$) pin on the Intel MPIIX controller (see *PC/PCI DMA*).
IRQ5 IRQ12 IRQ14	156 161 162	O	Interrupt requests 5, 12, and 14. These signals are ISA interrupts. These terminals indicate an interrupt request from one of the PC Cards. The interrupt mode is selected in the device control register of the TI extension registers.
IRQ15/RI_OUT	163	I/O	Interrupt request 15. IRQ15/RI_OUT indicates an interrupt request from one of the PC Cards. RI_OUT allows the RI input from the 16-bit PC Card to be output to the system. IRQ15/RI_OUT is configured in the card control register of the TI extension registers (see *card control register*).

PC Card power switch

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
CLOCK	151	O	Power switch clock. Information on the DATA line is sampled at the rising edge of CLOCK. The frequency of the clock is derived from dividing PCICLK by 36. The maximum frequency of CLOCK is 2 MHz (see *TPS2202 PC Card power control interface*).
DATA	152	O	Power switch data. DATA is used by the PCI1130 to serially communicate socket power control information.
LATCH	150	O	Power switch latch. LATCH is asserted by the PCI1130 to indicate to the PC Card power switch that the data on the DATA line is valid.

speaker control

TERMINAL NAME	TERMINAL NO.	I/O TYPE	DESCRIPTION
$\overline{\text{SPKROUT}}$/ $\overline{\text{SUSPEND}}$	149	O	Speaker. $\overline{\text{SPKROUT}}$ carries the digital audio signal from the PC Card. $\overline{\text{SUSPEND}}$ places the PCI1130 in suspend mode (see *PCI1130 suspend mode*). $\overline{\text{SPKROUT}}$/$\overline{\text{SUSPEND}}$ is configured in the card control register (see *card control register*) of the TI extension registers.

PRODUCT PREVIEW

Terminal Functions (Continued)

power supply

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
GND	13, 22, 44, 75, 96, 129, 153, 167, 181, 194, 207	I	Device ground terminals
V_{CC}	7, 31, 64, 86, 113, 143, 164, 175, 187, 201	I	Power-supply terminals for core logic (3.3 V)
V_{CCA}	120	I	Power-supply terminal for PC Card A (5 V or 3.3 V)
V_{CCB}	38	I	Power-supply terminal for PC Card B (5 V or 3.3 V)
V_{CCP}	148, 172	I	Power-supply terminals for PCI interface (5 V or 3.3 V)

architecture

This section provides an overview of the PCI1130 PCI-to-PC Card/CardBus controller, followed by detailed descriptions of PCI and PC Card interfaces, the TPS2202 interface, and interrupt support. Both hardware protocols and software programming models are discussed.

introduction to the PCI1130

The PCI1130 is a bridge between the PCI local bus and two PC Card sockets supporting both 16-bit and 32-bit CardBus PC Cards. It is compliant with the PCI local bus specification revision 2.1 and PCMCIA's 1995 PC Card standard. The PCI1130 PC Card interface recognizes and identifies PC Cards installed at power up or run-time, and automatically switches protocols to accommodate 16-bit and 32-bit cards. The PCI1130 includes support for 16-bit PC Card features such as multifunction cards, 3.3 V cards, and DMA, as well as backward compatibility to the PCMCIA release 2.1-compliant PC Cards. CardBus cards operating at up to 33 MHz and with a 32-bit datapath offer higher performance, and the PCI1130 allows applications to take full advantage of this bandwidth. The PCI1130 core is powered at 3.3 V to provide low power dissipation, but can independently support either 3.3-V or 5-V signaling on the PCI and PC Card interfaces.

Host software interacts with the PCI1130 through a variety of internal registers that provide status and control information about the PC Cards currently in use and the internal operation of the PCI1130 itself. These internal registers are accessed by application software either through the PCI configuration header, or through programmable windows mapped into PCI memory or I/O address space. The PCI1130 uses a windows format to pass cycles between PCI and PC Card address spaces. Host software must program the location and size of these windows when the PCI1130 or PC Card is initialized.

The PCI1130 also communicates via a three-line serial protocol to the TI TPS2202 dual PCMCIA power switch. The TPS2202 switches V_{CC} and V_{PP} supply voltage to the two PC Card sockets independently. Host software has indirect control over the TPS2202 by writing to internal PCI1130 registers. To prevent damage to low-voltage CardBus PC Cards, the PCI1130 allows only valid V_{CC} settings to be applied to such cards.

The PCI1130 can notify the host system via interrupts when an event occurs that requires attention from the host. Such events are either card status change (CSC) events or functional interrupts from a PC Card. CSC events occur within the PCI1130 or at the PC Card interface, and indicate a change in the status of the socket (i.e., card insertion or removal). Functional interrupts originate from the PC Card application and are passed from the card to the host system. Both CSC and functional interrupts can be individually masked and routed to a variety of system interrupts. The PCI1130 can signal the system interrupt controller via PCI-style interrupts, ISA IRQs, or with the serialized IRQ protocol.

The following sections describe how the PCI1130 interacts at electrical, protocol, and software levels at its PCI interface, PC Cards, TPS2202 PC Card power control, and interrupt interfaces.

PRODUCT PREVIEW

PCI Interface

This section describes the PCI1130 PCI interface, how the device responds and participates in PCI bus cycles, and how the major internal registers appear in the PCI address space. The PCI1130 provides all required signals for PCI master/slave device(s) and can operate in either 3.3-V or 5-V PCI signaling environments by connecting the two V_{CCP} terminals to the desired switching level.

The PCI1130 is a true multifunction PCI device, with two different PCI functions residing within the device. PCI function 0 is associated with PC Card socket A and PCI function 1 is associated with PC Card socket B. The PCI1130 behaves in accordance with the PCI specification for multifunction devices. Functions 0 and 1 have separately addressable PCI configuration headers and can use PCI INTA and INTB, respectively.

The PCI1130 responds as a PCI target device to PCI bus cycles based on its decode of the address phase of each cycle and internal register settings of the device. Table 5 lists the valid PCI bus cycles and their encoding on the 4-bit C/$\overline{BE}$ bus during the address phase of a bus cycle. The most common PCI bus commands are read and write cycles to one of the three PCI address spaces: memory, I/O, and configuration address spaces.

Table 5. PCI Command Definition

C/$\overline{BE3}$–C/$\overline{BE0}$	COMMAND
0000	Interrupt acknowledge
0001	Special cycle
0010	I/O read
0011	I/O write
0100	Reserved
0101	Reserved
0110	Memory read
0111	Memory write
1000	Reserved
1001	Reserved
1010	Configuration read
1011	Configuration write
1100	Memory read multiple
1101	Dual address cycle
1110	Memory read line
1111	Memory write and invalidate

The PCI1130 never responds as a PCI target device to the interrupt acknowledge, special cycle, dual address cycle, or reserved commands, nor initiates them as a PCI master device. The remaining PCI commands address one of the three PCI address spaces mentioned earlier, and each is described in the following three sections. The PCI1130 accepts PCI cycles by asserting $\overline{DEVSEL}$ as a medium-speed device.

The ability of the PCI1130 to respond to PCI memory or I/O bus cycles is dictated by register bits in the PCI command register. This register is located in the PCI configuration header at offset 04h and is required by the PCI local bus specification. Bits 0 and 1 of this register enable the PCI1130 to respond to I/O and memory cycles, respectively. Host software must set these bits during initialization of the device. Bit 2 of this register enables/disables the bus-mastering capability of the PCI1130 on the PCI bus. Host software also must set this bit during device initialization.

The PCI1130 can accept and generate PCI burst cycles during transfers to and from the CardBus. The PCI and CardBus interfaces can burst 32-bit data without wait states until their internal FIFOs are emptied/filled.

PCI configuration address space and bus hierarchy

The PCI local bus specification defines two types of PCI configuration read and write cycles: type 0 and type 1. The PCI1130 decodes each type differently. Type 0 configuration cycles are intended for devices on the current bus, while type 1 configuration cycles are intended for devices at a subordinate bus. The difference between these two types of cycles is the encoding of the PCI address AD bus during the address phase of the cycle. The address AD bus encoding during the address phase of a type 0 configuration cycle is shown in Figure 1. The 6-bit register number field represents an 8-bit address but with two lower bits masked to 0. This results in a 256-byte configuration address space (per PCI function) with a 32-bit, or double-word granularity. Individual byte addresses can be selected for read/write using the C/$\overline{BE}$ signals during the data phase of the cycle.

31 – 11	10 – 8	7 – 2	1	0
Reserved	Function number	Register number	0	0

Figure 1. PCI AD31–AD0 During Address Phase of a Type 0 Configuration Cycle

The PCI1130 claims type 0 configuration cycles only when IDSEL is asserted during the address phase of the cycle. The PCI function number encoded in the cycle is 0 or 1. If the function number is 2 or greater, the PCI1130 does not recognize the configuration command. The PCI1130 services valid type 0 configuration read or write cycles by accessing internal registers from the appropriate configuration header. Table 6 shows a PCI configuration header in the PCI1130.

Table 6 represents either PCI1130 function. Blocks with a dagger (†) represent registers that are, in whole or in part, common between the two functions. Blocks without a dagger are registers that are separate and distinct between the two functions. Refer to *PCI configuration header register* for a complete description of all of the registers shown in Table 6.

Because type 1 configuration cycles are issued to devices on subordinate buses, the PCI1130 claims type 1 configuration cycles based on the bus number of the destination bus. The AD bus encoding during the address phase of a type 1 configuration cycle is shown in Figure 2. The device number and bus number fields define the destination bus and device for the cycle.

31 – 24	23 – 16	15 – 11	10 – 8	7 – 2	1	0
Reserved	Bus number	Device number	Function number	Register number	0	1

Figure 2. PCI AD31–AD0 During Address Phase of a Type 1 Configuration Cycle

Several PCI1130 configuration registers in Table 6 are significant when decoding and claiming type 1 configuration cycles. The destination bus number encoded on the AD bus is compared to the values programmed in the PCI1130 configuration registers 18h, 19h, and 1Ah. These registers are named PCI bus number, CardBus bus number, and subordinate bus number (see *PCI bus number, CardBus bus number, and subordinate bus number*). These registers default to 00h and are programmed by host software to reflect the bus hierarchy in the system (see Figure 3 for an example of a system bus hierarchy and how the PCI1130 bus number registers would be programmed in this case).

PCI configuration address space and bus hierarchy (continued)

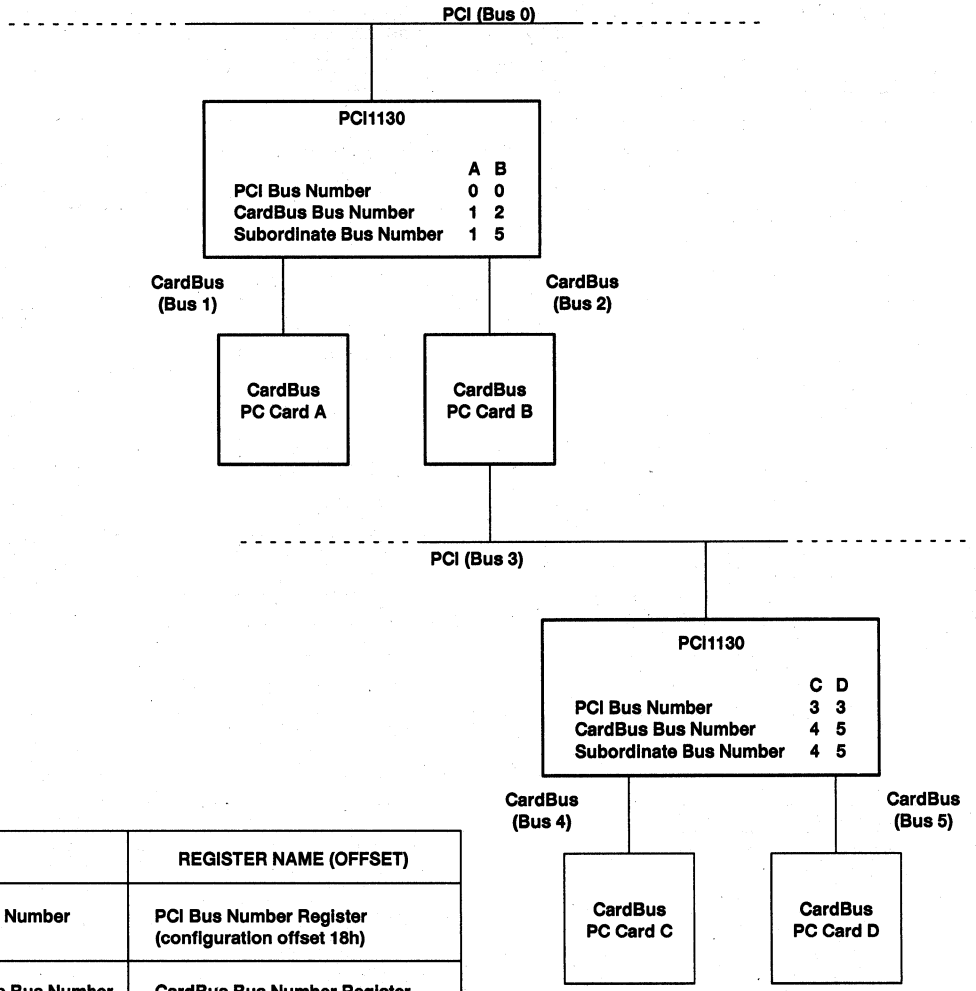
Table 6. PCI1130 Configuration Header

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 registers/ExCA base address register				10h
Secondary status		Reserved		14h
CardBus latency timer†	Subordinate bus number	CardBus bus number	PCI bus number†	18h
Memory base register 0				1Ch
Memory limit register 0				20h
Memory base register 1				24h
Memory limit register 1				28h
I/O base register 0				2Ch
I/O limit register 0				30h
I/O base register 1				34h
I/O limit register 1				38h
Bridge control†		Interrupt pin	Interrupt line†	3Ch
Subsystem ID		Subsystem vendor ID		40h
PC Card 16-bit IF legacy-mode base address†				44h
Reserved				48h – 7Ch
System control register†				80h
Reserved				84h – 8Ch
Buffer control†	Device control†	Card control†	Retry status†	90h
Socket DMA register 0				94h
Socket DMA register 1				98h
Reserved				9Ch – FFh

† One or more bits in the register are common to both PC Card socket configuration spaces 0 and 1.

PRODUCT PREVIEW

PCI configuration address space and bus hierarchy (continued)



	REGISTER NAME (OFFSET)
PCI Bus Number	PCI Bus Number Register (configuration offset 18h)
CardBus Bus Number	CardBus Bus Number Register (configuration offset 19h)
Subordinate Bus Number	Subordinate Bus Number Register (configuration offset 1Ah)

Figure 3. Bus Hierarchy and Numbering

PRODUCT PREVIEW

PCI configuration address space and bus hierarchy (continued)

Based on the system bus hierarchy in Figure 3, host software programs the PCI1130 bus number registers for each socket as indicated. The PCI bus number register is set to the bus number assigned to the PCI bus and should be the same for both sockets. The CardBus bus number register is set to the bus number assigned to the CardBus bus, and the subordinate bus number register is set to the number of the highest numbered bus below the CardBus bus. Once this programming is complete, the PCI1130 performs one of the following actions when a type 1 configuration cycle is issued on the primary bus:

Type 1-to-type 0 conversion: If the destination bus number encoded on the AD bus during the address phase of the cycle matches the value in the CardBus bus number register and the destination device number encoded on the AD bus is 00h, the PCI1130 claims the cycle on the PCI bus and converts it to a type 0 cycle on the CardBus bus. If the destination device number is not 00h, the cycle is claimed but the PCI1130 does not assert $\overline{\text{DEVSEL}}$. Type 1 cycles to devices other than 00h are claimed but are not passed on. Reads return all 1s.

Type 1-to-type 1 conversion: If the destination bus number encoded on the AD bus during the address phase of the cycle is greater than the value in the CardBus bus number register and less than or equal to the value in the subordinate bus number register, the PCI1130 claims the cycle on the PCI bus and passes it unchanged as a type 1 cycle on the CardBus bus.

The behavior described above assumes that the socket is occupied by a CardBus card. If the socket is either empty or occupied by a 16-bit PC Card, type 1 cycles are not passed to that socket regardless of the programming of configuration registers 18h – 1Ah. If the type 1 configuration write cycle is decoded because of the values in the configuration registers 18h – 1Ah, the cycle is accepted but no information is passed through the PCI1130. In the case of a type 1 configuration read cycle, the PCI1130 returns all 1s. Type 1 cycles to devices other than 00h are claimed but are not passed on. Reads return all 1s. The PCI1130 never issues PCI configuration read or write cycles on the PCI bus as a PCI bus master.

PCI I/O address space

The PCI local bus specification defines an I/O address space accessed using 32-bit addresses, yielding a 4G-byte usable address space. The PCI1130 decodes PCI I/O cycles as a PCI target device only if host software has enabled it to do so (see bit 0 of the PCI command register). If so enabled, the PCI1130 positively decodes the address on the PCI AD bus and claims the cycle if a hit is detected to a programmed I/O window. Such a window can be mapped either to internal PCI1130 registers or to PC Card address space.

There are two instances in which the PCI1130 maps internal registers to PCI I/O address space. The first is the legacy 16-bit PC Card index/data registers (used to access the ExCA registers), and the second is DMA socket registers (used to access registers in distributed DMA). In both cases, the locations of these windows are programmed by base address registers in PCI configuration space. The legacy 16-bit PC Card base address (see *PC Card 16-bit IF legacy-mode base address*) is located at configuration offset 44h and is common to both PCI1130 functions 0 and 1. This base address locates a 4-byte window in I/O space anywhere in the 32-bit I/O address space. The socket DMA base address register (see *socket DMA register 1*) is located at configuration offset 98h and is separate and distinct for functions 0 and 1. This base address locates a 16-byte window in I/O space in the lower 64K bytes of PCI I/O address space. For a complete description of this base address register and the socket DMA registers, see *socket DMA register* and *DMA registers*.

The PCI1130 enables host software to program PCI I/O windows to PC Card address spaces. These windows provide the bounds upon which the PCI1130 positively decodes I/O cycles from PCI to a PC Card and are the primary means for applications to communicate with PC Cards (see *16-bit PC Cards and windows*, *ExCA registers*, and *CardBus PC Cards and windows*).

As a PCI bus master, the PCI1130 initiates a PCI I/O cycle only when a bus-mastering CardBus card has previously initiated the identical cycle on the CardBus bus.

PCI memory address space

The PCI local bus specification also defines a memory address space accessed using 32-bit addresses, yielding a 4G-byte usable address space. The PCI1130 decodes PCI memory cycles as a PCI target device only if host software has enabled it to do so (see bit 1 of the *PCI command register*). If so enabled, the PCI1130 positively decodes the address on the PCI AD bus and claims the cycle if a hit is detected to a programmed memory window. Such a window can be mapped either to internal PCI1130 registers or to PC Card address space.

The only case in which the PCI1130 maps internal registers to PCI memory address space is the CardBus/ExCA registers that are mapped into a 4K-byte window for each socket. The location of these windows is programmed by a base address register in PCI configuration space. The CardBus socket/ExCA base address is located at configuration offset 10h and is separate and distinct between functions 0 and 1. Each base address locates a 4K-byte window in memory space anywhere in the 32-bit memory address space. For a description of this base address register and the CardBus socket registers, see *CardBus socket registers/ExCA registers base address register.*

The PCI1130 enables host software to program PCI memory windows to PC Card address spaces. These windows provide the bounds upon which the PCI1130 positively decodes memory cycles from PCI to a PC Card and are the primary means for applications to communicate with PC Cards (see *16-bit PC Cards and windows* and *ExCA registers*). When passing memory cycles between PCI and CardBus, the PCI1130 distinguishes between the three different types of PCI memory read commands: memory read, memory read line, and memory read multiple. The PCI1130 uses the value in the cache line size register (see *cache line size*) to determine how it responds to these cycles. The PCI1130 read and write FIFOs must be programmed for their full depth by setting the appropriate bits in the buffer control register (see *buffer control register*), PCI configuration offset 93h. A memory read always disconnects after the first data phase.

compliance to PCI local bus specification revision 2.1

The most significant additions to the PCI local bus specification revision 2.1 are the latency requirements on PCI peripherals. Minimum response times are specified for a PCI device to respond with valid data. These requirements are intended to improve throughput and reduce latencies on the PCI bus. The PCI1130 is fully compliant with these guidelines.

Other additions to revision 2.1 of the PCI local bus specification include the addition of the subsystem ID and subsystem vendor ID registers in the PCI configuration header.

PC Cards

The 1995 PC Card standard provides a hardware- and software-interface standard for connecting credit-card-sized memory and I/O cards to personal computers. By implementing compliant card slots, PC manufacturers allow customers to use industry-standard PCMCIA memory and I/O cards from many different vendors. The 1995 PC card standard defines 16-bit and 32-bit PC Cards. The 16-bit PC Cards are an extension of the PCMCIA 2.1/JEIDA 4.1 standards and are sometimes referred to as 16-bit cards or as R2 cards. The 32-bit PC Cards are a newly defined architecture called CardBus cards with all 60 signals on the PC Card interface redefined for a synchronous, 32-bit bus environment patterned after PCI.

PC card insertion/removal and recognition

Prior to the PCMCIA 1995 PC Card standard, only two types of PC Cards existed: 16-bit memory cards and 16-bit I/O cards. Both types of cards were designed for 5-V V_{CC} supply and could be hot-inserted into a fully powered socket. Upon insertion, 16-bit I/O cards were required to use the memory card signaling conventions until host software had read the card information structure (CIS) and switched the socket and card to an I/O mode.

PC card insertion/removal and recognition (continued)

The 1995 PC Card standard introduced several features, such as CardBus and 3.3-V/5-V card support, which have challenged the idea of hot insertion and introduced a new card recognition scheme. Both CardBus cards and 16-bit PC Cards can now be designed for 3.3-V V_{CC} supply, which offers power savings, but could result in card damage if such a card were inserted into a socket powered at 5 V. Similarly, the socket can no longer automatically power a PC Card to 5-V V_{CC}, so a method of detecting the voltage requirements and card type is needed. The 1995 PC Card standard addresses this by describing an interrogation procedure that the socket must initiate upon card insertion into a cold, unpowered socket.

This scheme uses the card $\overline{CD1}$, $\overline{CD2}$, $\overline{VS1}$, and $\overline{VS2}$ signals (called $\overline{CCD1}$, $\overline{CCD2}$, CVS1, and CVS2 for CardBus cards). A PC Card designer connects these four pins in a certain configuration depending on the type of card (16 bit or CardBus) and the supply voltage (5 V, 3.3 V, X.X V and/or Y.Y V). The encoding scheme for this is defined in the 1995 PC Card standard and in Table 7.

Table 7. PC Card Card Detect and Voltage Sense Connections

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

Based on the information described in Table 7, the PCI1130 executes an algorithm upon card insertion that alternatively drives the $\overline{VS1}$ and $\overline{VS2}$ pins to low and high levels to determine which of the card types has been inserted. This process is completed without V_{CC} being applied to the socket. Once the PCI1130 has successfully determined the card type and voltage requirements, it updates the appropriate status bits in the CardBus socket present state register (see *CardBus socket present state register*) and asserts a CSC interrupt to the host system. Host software must then read the CardBus socket registers to determine the card type and voltage requirements and respond accordingly.

16-bit PC cards and windows

PCMCIA revision 1.0 defined the original 16-bit memory card, and the later PCMCIA revisions 2.0 and 2.1 defined the 16-bit I/O card. Both types of 16-bit PC Cards have 16-bit datapaths and a 26-bit address bus defined. Status and control signals differ between the two card types. The PCI1130 fully supports both types of cards. The ExCA register set is implemented in the PCI1130, which provides the industry standard Intel 82365SL-DF programming model.

The 16-bit memory cards can have two types of memory address space: attribute memory and common memory. The attribute memory address space contains the card information structure (CIS), and common memory is the memory space used by the application.

16-bit PC cards and windows (continued)

The CIS is defined by PCMCIA and contains a variety of information about the card capabilities and resource requirements. Host software reads and parses the CIS to set up the system resources to use the card application. Both attribute and common memory are accessed with 26-bit addresses, resulting in a total addressable memory address space of 64M bytes.

The 16-bit I/O cards can possess attribute and common memory, but also have an I/O address space. This address space is accessed via 16-bit I/O addresses, resulting in a 64K-byte I/O address space.

The PCI1130 provides a windowing mechanism to link the PCI address space to 16-bit PC Card address space. Both of these memory and I/O windows are programmed by host software in the ExCA registers (windows to CardBus address spaces are provided separately and are discussed in the following sections). The PCI1130 provides up to five memory windows per socket and two I/O windows per socket. Once enabled, the PCI1130 positively decodes and claims bus cycles that fall within these windows. Bus cycles to the PC Card are then initiated to write data to the card (in the case of a PCI write cycle) or to read data from the card (in the case of a PCI read cycle).

Memory and I/O windows to 16-bit PC Cards have several programmable options associated with them. Host software can choose among these options by setting the appropriate bits in the appropriate ExCA registers. These options include:

- Window start address
- Window end address
- Window offset address
- Page address (for 16-bit PC Card memory windows only)
- Attribute or common memory access (for 16-bit PC Card memory windows only)
- PC Card datapath width (8 bit or 16 bit)
- Wait state timing (ISA bus timing or minimum)
- Write protection (enable/disable writes to memory windows)

The start, end, offset, and page addresses define the bounds of the memory window in PCI and PC Card memory address spaces. The page address is necessary to take into account the difference in addressable memory between PCI (4G bytes) and 16-bit PC Cards (64M bytes). The 8-bit page address appended to the 26-bit start and end addresses define the bounds of the window in PCI memory address space. When a PCI memory cycle is decoded and claimed, the PCI1130 adds the offset address to the PCI address before passing the lower 26 bits to the PC Card. The memory windows need not be aligned between the two address spaces.

ExCA registers

The PCI1130 is fully register compatible with the Intel 82365SL-DF PC Card interface controller. The ExCA compatibility registers can be accessed indirectly via PCI I/O address space or directly via PCI memory address space. For I/O access, the PCI1130 uses the same index and data I/O port scheme introduced by Intel. This index/data window is located in PCI I/O space by the PC Card 16-bit IF legacy base address (see *PC Card 16-bit IF legacy-mode base address*), found at offset 44h in PCI configuration space. The PC Card 16-bit IF legacy-mode base address is shared by both sockets and the ExCA registers run contiguously from index 00h–3Fh for socket A and 40h–7Fh for socket B. Accesses to ExCA indices 80–FFh return 0s when read. Writes have no effect.

The compatibility registers also can be accessed directly through the CardBus socket/ExCA register window. This window in PCI memory address space is located by the CardBus socket registers/ExCA registers base address register (see *CardBus socket registers/ExCA registers base address register*), found at offset 10h in PCI configuration space. The ExCA compatibility registers are directly mapped into this memory window, starting at an offset of 800h from the bottom of this window. Each socket has a separate CardBus socket register/ExCA register base address register for accessing the ExCA registers. ExCA I/O windows are accessed on word (16-bit) boundaries.

ExCA registers (continued)

The ExCA registers provide bits to control many 16-bit PC Card functions. These functions include:

- Explicit writeback/clear on read of interrupt flag mode selection
- PC Card CSC and functional interrupt control
- Interrupt mode select: level/edge interrupt modes
- PC Card socket status information
- ExCA registers configuration after PC Card removal – reset upon card removal or save the register values upon card removal
- Memory and I/O windows configuration for 16-bit PC Cards

Table 8 classifies the basic functionality of each register in the ExCA register set. The functional classifications are: card status register, card control register, memory window, and I/O window. Some registers are classified as both card status and card control since some bits within the register provide status information and other bits provide card control.

When a 16-bit PC Card is installed in a socket, the entire ExCA register set associated with that socket is enabled. Some status and control functions in the CardBus socket registers are maintained when a 16-bit PC Card is present, such as the socket power control register. When a CardBus PC Card is installed in a socket, the entire ExCA register set associated with that socket is enabled. Some status and control functions in the CardBus socket registers are maintained when a CardBus PC Card is present, such as the socket power control register and interrupt routing. Software is expected to use either ExCA or CardBus socket registers to control socket power, but not both. The intent is to be fully backward compatible with present card and socket services, but take advantage of the easy access of some of the newly defined CardBus registers.

Table 8. ExCA Registers

REGISTER NAME	STATUS	CONTROL	MEMORY WINDOW	I/O WINDOW	EXCA OFFSET
Identification and revision register	X				00
Interface status register	X				01
Power control register		X			02
Interrupt and general control register		X			03
Card status-change register	X				04
Card status-change interrupt configuration register		X			05
Address window enable register			X	X	06
I/O window control register				X	07
I/O window 0 start-address low-byte register				X	08
I/O window 0 start-address high-byte register				X	09
I/O window 0 end-address low-byte register				X	0A
I/O window 0 end-address high-byte register				X	0B
I/O window 1 start-address low-byte register				X	0C
I/O window 1 start-address high-byte register				X	0D
I/O window 1 end-address low-byte register				X	0E
I/O window 1 end-address high-byte register				X	0F

PRODUCT PREVIEW

ExCA registers (continued)

Table 8. ExCA Registers (Continued)

REGISTER NAME	STATUS	CONTROL	MEMORY WINDOW	I/O WINDOW	EXCA OFFSET
Memory window 0 start-address low-byte register			X		10
Memory window 0 start-address high-byte register			X		11
Memory window 0 end-address low-byte register			X		12
Memory window 0 end-address high-byte register			X		13
Memory window 0 offset-address low-byte register			X		14
Memory window 0 offset-address high-byte register			X		15
Card detect and general control register	X	X			16
Reserved					17
Memory window 1 start-address low-byte register			X		18
Memory window 1 start-address high-byte register			X		19
Memory window 1 end-address low-byte register			X		1A
Memory window 1 end-address high-byte register			X		1B
Memory window 1 offset-address low-byte register			X		1C
Memory window 1 offset-address high-byte register			X		1D
Global control register		X			1E
Reserved					1F
Memory window 2 start-address low-byte register			X		20
Memory window 2 start-address high-byte register			X		21
Memory window 2 end-address low-byte register			X		22
Memory window 2 end-address high-byte register			X		23
Memory window 2 offset-address low-byte register			X		24
Memory window 2 offset-address high-byte register			X		25
Reserved					26
Reserved					27
Memory window 3 start-address low-byte register			X		28
Memory window 3 start-address high-byte register			X		29
Memory window 3 end-address low-byte register			X		2A
Memory window 3 end-address high-byte register			X		2B
Memory window 3 offset-address low-byte register			X		2C
Memory window 3 offset-address high-byte register			X		2D
Reserved					2E
Reserved					2F
Memory window 4 start-address low-byte register			X		30
Memory window 4 start-address high-byte register			X		31
Memory window 4 end-address low-byte register			X		32
Memory window 4 end-address high-byte register			X		33
Memory window 4 offset-address low-byte register			X		34
Memory window 4 offset-address high-byte register			X		35
I/O window 0 offset address low-byte register				X	36
I/O window 0 offset address high-byte register				X	37
I/O window 1 offset address low-byte register				X	38
I/O window 1 offset address high-byte register				X	39

ExCA registers (continued)

Table 8. ExCA Registers (Continued)

REGISTER NAME	STATUS	CONTROL	MEMORY WINDOW	I/O WINDOW	EXCA OFFSET
Reserved					3A
Reserved					3B
Reserved					—
Reserved					3D
Reserved					3E
Reserved					3F
Memory window page register			X		3C

CardBus PC Cards and windows

The 1995 PC Card standard defines CardBus as a 32-bit address and data bus with a maximum clock speed of 33 MHz. CardBus uses a synchronous protocol modeled after the PCI bus. The CardBus bus uses a multiplexed 32-bit address and data bus, with a 4-bit command/byte-enable bus. Many other PCI control signals appear on the CardBus bus in the same capacity. Because of the similarities between the two buses, the PCI1130 is very much like a PCI-to-PCI bridge during transactions between PCI and a CardBus PC Card, which is evident in the PCI1130 PCI configuration header. The PCI1130 controls the CardBus clock source and is the CardBus central resource and arbiter; thus, it also controls access to the CardBus bus.

When a CardBus card is installed, access is made through PCI1130 memory and I/O windows that are full 32-bit addresses, defined by base and limit addresses (analogous to the 16-bit start and end addresses), and are mapped directly into the 32-bit CardBus memory address space, so page registers are not required. Offset registers are unnecessary because the CardBus PC Cards must relocate their memory windows to match PCI memory windows programmed in the PCI1130 configuration header. The memory windows allow a minimum granularity of 4K bytes and are aligned on natural 4K-byte boundaries. The base and limit registers must not be equal to 0.

The PCI1130 has another use for the PCI memory windows to CardBus PC Cards. It can determine when to pass a CardBus memory cycle, initiated by a CardBus bus master, to the PCI bus. In this case, the decoding is the opposite of that used on the PCI interface. When decoding the address phase of a CardBus memory read or write cycle, the PCI1130 accepts cycles that address memory locations outside the base and limit addresses of the PCI memory windows defined for the CardBus card.

When a CardBus I/O card is installed, access is made through the PCI I/O windows that are full 32-bit addresses, defined by base and limit addresses (analogous to the 16-bit start and end addresses), and are mapped directly into the 32-bit CardBus I/O address space, so page registers are not required. Base and limit registers for two I/O windows are programmed in the PCI1130 configuration header. These windows allow a minimum double-word granularity and are aligned on natural double-word boundaries.

PCI I/O windows to CardBus PC Cards determines when to pass a CardBus I/O cycle, initiated by a CardBus bus master, to the PCI bus. The decoding is opposite of that used on the PCI interface. When decoding the address phase of a CardBus I/O read or write cycle, the PCI1130 accepts cycles that address I/O locations outside the base and limit addresses of the PCI I/O windows defined for the CardBus card.

Windows to CardBus PC Cards have several programmable options associated with them. These options include:

- Memory I/O base register
- Memory I/O limit register
- Control of CardBus bus master capabilities

CardBus PC Cards and windows (continued)

When the PCI1130 receives an access from the CardBus master, the request is translated internally into a PCI master access onto the PCI bus. During this translation, the data is buffered through a FIFO that increases the data throughput while preventing the data from being lost.

For a read access from a CardBus PC Card, the PCI1130 issues a retry to the CardBus while issuing a request to the PCI bus, then waits for the PCI interface to accept the access for data to be passed on to the CardBus. When the CardBus device returns to retrieve the data from the retry, the data passes through to the CardBus PC Card. This process repeats until no retries occur and all data is transferred.

CardBus socket registers

The PCMCIA CardBus specification requires a CardBus socket controller to provide five 32-bit registers that report and control the socket-specific functions. The PCI1130 provides the CardBus socket base address register to locate these CardBus socket registers in PCI memory address space. Each socket has a separate CardBus socket register/ExCA register base address register for accessing the CardBus socket registers. This base address register is located at offset 10h in the PCI1130 configuration space.

The CardBus socket registers provide control and status to the PC Card socket. Some of the options implemented include:

- PC Card socket status
- PC Card socket interrupt control
- PC Card voltage capability determination and reporting
- PC Card type determination and reporting
- PC Card power control
- Power management control and status

Table 9 lists the CardBus socket registers and their respective offset from the CardBus socket register base address.

Table 9. CardBus Socket Registers

REGISTER NAME	OFFSET
Socket event register	00h
Socket mask register	04h
Socket present state register	08h
Socket force event register	0Ch
Socket control register	10h
Reserved	14–1Fh
Socket power management register	20h

TPS2202 PC card power control interface

The attribute of PC Card technology that enables PC Cards to be inserted and removed in a system during run time requires that power to the PC Card sockets be managed. The TI TPS2202 PC card power switch performs this duty by switching V_{CC} and V_{PP} to two card sockets under the control of the PCI1130. Another TI power switch, the TPS2202A, also can be used. (References in this document to the TPS2202 apply identically to the TPS2202A.) Both the TPS2202 and TPS2202A are pin compatible and provide the same signaling interface from the PCI1130. The TPS2202A provides RESET and $\overline{\text{RESET}}$ pins that allow the socket V_{CC} and V_{PP} to be shut down via external control from either system reset or a power supervisory device in the system.

TPS2202 PC card power control interface (continued)

The PCI1130 and TPS2202 communicate via a 3-line serial interface called P^2C (PCMCIA peripheral control). This serial interface is a significant savings in pin count over the 8-line signaling convention. The P^2C signaling is transparent to host software; the PCI1130 generates the proper signal protocols when its internal V_{CC}/V_{PP} control registers are written. Figure 4 illustrates the protocol used to communicate from the PCI1130 to the TPS2202.

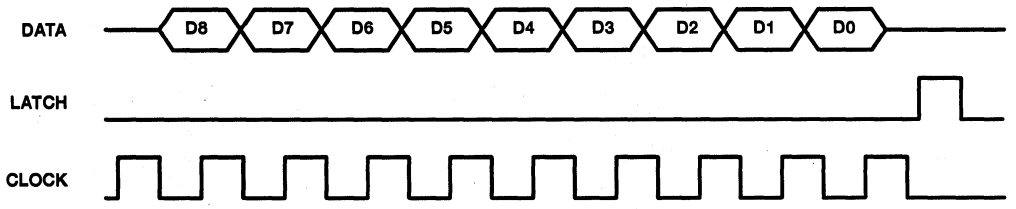


Figure 4. Serial-Interface Timing

The DATA, LATCH, and CLOCK terminals on the PCI1130 are connected to the terminals of the same name on the TPS2202. The PCI1130 generates the TPS2202 CLOCK signal by dividing the PCI CLK input by 36. A PCI CLK frequency of 33 MHz results in a TPS2202 CLOCK frequency of approximately 1 MHz. To conserve power, the PCI1130 switches the TPS2202 CLOCK signal only when transmitting information to the power switch; otherwise, the PCI1130 stops the clock in a logic low state.

The encoding of the serial data stream is shown in Table 10. The ninth data bit, D8, is not shown. This bit (D8) is the active low shutdown ($\overline{SHDN}$) bit, and when reset to 0, the values of bits D0 through D7 are ignored and the power switch removes all power to both PC Card sockets. The PCI1130 sets this bit to a logic high value at all times.

Table 10. TPS2202 Control Logic

CONTROL SIGNALS											
D0	D1	A V_{PP}	D2	D3	A V_{CC}	D4	D5	B V_{PP}	D6	D7	B V_{CC}
0	0	0 V	0	0	0 V	0	0	0 V	0	0	0 V
0	1	A V_{CC}	0	1	5 V	0	1	B V_{CC}	0	1	3.3 V
1	0	12 V	1	0	3.3 V	1	0	12 V	1	0	5 V
1	1	Hi–Z	1	1	0 V	1	1	Hi–Z	1	1	0 V

Interrupts

Interrupts are an integral component in any computer architecture. The dynamic nature of PCMCIA and the abundance of PC Card I/O applications mean that interrupts are an integral part of the PCI1130. The PCI1130 provides several interrupt signaling schemes to accommodate 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 PCMCIA CardBus specification provides interrupt control for the CardBus PC Card functions. The PCI1130 is backward compatible with existing register definitions and defines new ones where required.

The PCI1130 detects interrupts and/or events at the PC Card interface and notifies the host interrupt controller via one of several interrupt signaling protocols. To simplify the discussion and use of interrupts in the PCI1130, PC Card interrupts are classified as either CSC interrupts or functional interrupts. Functional interrupts are explicit requests for interrupt servicing directly from the PC Card application itself. Such requests are communicated over a dedicated PC Card signal defined for this purpose. CSC interrupts indicate a change in the state of the PC Card (i.e., card removal or insertion, or power up complete). All sources of functional and CSC interrupts are discussed in detail, as well as any specific options to be configured by host software.

interrupts (continued)

The method by which either type of PC Card interrupt is communicated to the host interrupt controller varies from system to system. The PCI1130 offers system designers the choice of using PCI interrupt signaling, traditional ISA IRQ signaling, the serialized IRQ protocol, or PCI with ISA interrupts.

functional and CSC interrupts

Functional interrupts are 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 CardBus cards. CSC interrupts are defined to be events at the PC Card interface that are detected by the PCI1130 and can warrant notification of host software for service. Such events include transitions on certain PC Card signals or card removal/insertion. The specific examples of functional and CSC interrupts depend on the type of PC Card(s) installed in the socket. A CardBus card has entirely different methods for signaling interrupts than a 16-bit card, and the 16-bit interrupt sources differ between memory and I/O PC Cards.

Table 11 summarizes the sources of interrupts and the type of PC Card associated with them. The functional interrupt events are valid only for 16-bit I/O and CardBus PC Cards. Card insertion and removal events are independent of the card type since the same card detect signals are used in both cases and the PCI1130 cannot distinguish between card types.

Table 11. 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.
		CSC	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 (CINT)	Functional	READY(IREQ) // CINT	The assertion of $\overline{\text{CINT}}$ indicates an interrupt request from the PC Card.
	Power cycle complete	CSC	N/A	An interrupt is generated when a PC Card power-up cycle is complete.
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 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 is complete.

The signal-naming convention for PC Card signals describes the function for 16-bit memory and I/O cards, as well as CardBus. The 16-bit memory card signal name is first, with the I/O card signal name second, enclosed in parentheses. The CardBus signal name follows after a double forward slash (//). The 16-bit I/O and CardBus PC Cards both have similar methods for signaling interrupts. Both use two signals: one to indicate a change in card status and another dedicated to request interrupt servicing from the host. A 16-bit memory PC Card uses the BVD1 and BVD2 signals to indicate changes in battery conditions on the card and the READY signal to insert wait states during memory card data transfers.

functional and CSC interrupts (continued)

The PC Card standard describes the power-up sequence that must be followed by the PCI1130 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 PCI1130 interrupt scheme can be used to notify the host system denoted by "power cycle complete" (see Table 11). This interrupt source is considered a PCI1130 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.

Host software can individually mask (disable) each of the potential CSC interrupt sources listed in Table 11 by setting the appropriate bits in the PCI1130. By individually masking the interrupt sources listed in the Table 11, host software can control which events cause a PCI1130 interrupt. Host software has some control over which system interrupt the PCI1130 asserts by programming the appropriate routing registers. The PCI1130 allows host software to route PC Card CSC and functional interrupts to separate system interrupts. Interrupt routing is specific to the interrupt signaling method used and is discussed in the following sections.

When an interrupt is signaled by the PCI1130, the interrupt service routine must be able to discern which of the events in Table 11 caused the interrupt. This is of particular interest with CSC interrupts, where a variety of events at the card interface can cause interrupts. Internal registers in the PCI1130 provide flags that report to the host-interrupt-service routine which of the interrupt sources was the cause of an interrupt. By first reading these status bits, the interrupt-service routine can determine which action to take.

Table 12 describes the valid PC Card interrupt events and details the internal PCI1130 registers associated with masking and reporting them.

Table 12. PC Card Interrupt Mask and Flag Registers

CARD TYPE	EVENT	MASK	FLAG
16-bit memory	Battery conditions (BVD1, BVD2)	ExCA offset 05h/45h/805h Bits 1 and 0	ExCA offset 04h/44h/804h Bits 1 and 0
	Wait states (READY)	ExCA offset 05h/45h/805h Bit 2	ExCA offset 04h/44h/804h Bit 2
16-bit I/O	Change in card status (STSCHG)	ExCA offset 05h/45h/805h Bit 0 Always enabled	ExCA offset 04h/44h/804h Bit 0
	Interrupt request (IREQ) Power cycle complete	Always enabled	PCI configuration offset 91h Bit 0
CardBus	Change in card status (CSTSCHG)	Socket mask register Bit 0	Socket event register Bit 0
	Interrupt request (CINT)	Always enabled	Socket present state register Bit 6
	Power cycle complete	Socket mask register Bit 3	Socket event register Bit 3
All PC Cards	Card insertion or removal	Socket mask register Bits 2 and 1	Socket event register Bits 2 and 1 ExCA offset 04h/44h/804h Bit 3

There are various methods of clearing the interrupt flag bit. ExCA provides two methods to clear 16-bit PC Card-related interrupt flags. One is an explicit write of 1 to the bit, and the other is a simple read from the register. This selection is made by bit 2 in ExCA offset 1Eh/5Eh/81Eh (see *ExCA I/O window 0–1 offset-address high-byte register*).

NOTE:
The CardBus specification requires an explicit write of 1 to clear CardBus-related interrupt flags.

PRODUCT PREVIEW

functional and CSC interrupts (continued)

There is a single exception to Table 12, when PCI interrupt signaling is used. The enable/disable bits for functional and CSC interrupts are found in separate registers in PCI configuration register 91h, bits 4 and 3 (see *card control register*). Refer to the section on PCI interrupt signaling for details.

ISA IRQ interrupts

Among the PCI1130 interrupt signaling schemes is the traditional ISA IRQ signaling, available in most x86 PCs. Dedicated terminals on the PCI1130 can be used to assert ten of the fifteen ISA IRQ: IRQ3, IRQ4, IRQ5, IRQ7, IRQ9, IRQ10, IRQ11, IRQ12, IRQ14, and IRQ15. These IRQs represent the common interrupts expected by PC Card applications and several free IRQs for CSC routing.

In a system using ISA IRQ, the host software must first configure the PCI1130 to use ISA signaling by setting bits 2–1 of PCI configuration register, offset 92h, to 01b (see *device control register*). The ten IRQ terminals remain in the high-impedance state until the ExCA CSC and functional interrupt routing registers are set to a valid state. The step-by-step series of events that host software must follow to successfully configure the PCI1130 for ISA IRQ signaling follows. These steps assume that the system has powered up and $\overline{\text{RSTIN}}$ is high (deasserted). In cases where only selected bits of a register are to be modified, host software must leave the remaining register bits unchanged by reading the current contents of the register first, modifying the desired bits, then writing the new value back to the respective PCI1130 register.

- Set bits 2–1 of PCI configuration register 92h (function 0) to 01b for interrupt mode selection.
- Write to the upper four bits of ExCA register 05h/45h/805h for desired CSC routing for each socket (note the restrictions placed on interrupt routing with ISA IRQ signaling; only ten IRQs are valid in this mode).
- If a PC Card is installed in the socket and requires functional interrupts, write to the lower nibble of ExCA register 03h/43h/803h for desired functional interrupt routing for the socket (note the restrictions placed on interrupt routing with ISA IRQ signaling).
- Using Table 12, write to the appropriate mask register bits to enable interrupt generation for desired events.
- Upon card removal events, host software masks any functional interrupts that were set for that socket.
- Upon card insertion events, host software reconfigures the mask and routing registers to support the new card requirements.

PCI interrupts

The PCI1130 also supports interrupt signaling compliant with the PCI local bus specification. Consistent with this specification, the PCI1130 can use one PCI interrupt for each of its functions: INTA is used for PC Card socket A interrupts and INTB for socket B. These pins are on the PCI1130 at pins 154 and 155 and are dual-function pins with the ISA-mode interrupts IRQ3 and IRQ4. When the PCI1130 is configured for PCI interrupt signaling, these pins behave as open-drain PCI interrupts. Systems that prefer a single interrupt line from the PCI1130 can connect these two interrupt terminals together.

NOTE:
PCI interrupts can be used with ISA interrupts.

PCI Interrupts (continued)

The PCI configuration register offset 91h must be written in order to route CSC and functional interrupts from each socket. The step-by-step series of events for host software to successfully configure the PCI1130 for PCI signaling follows. These steps assume that the system has powered up and $\overline{\text{RSTIN}}$ is high (deasserted). In cases where only selected bits of a register are to be modified, host software must leave the remaining register bits unchanged by reading the current contents of the register first, modifying the desired bits, then writing the new value back to the register.

1. Set bit 5 of PCI configuration register 91h (function 0) to a value of 1 (enabled).
2. Set bit 3 of PCI configuration register 91h (functions 0 and 1 separately) to route CSC interrupts to INTA (for socket A) or INTB (for socket B).
3. If a PC Card is installed in the socket and requires functional interrupts, write to bit 4 of the PCI Card control register 91h (for the socket) to route functional interrupts from the PC Card to INTA (for socket A) or INTB (for socket B).
4. Using Table 12, write to the appropriate mask register bits to enable interrupt generation for desired events.
5. Upon card-removal events, host software masks any functional interrupts that were set for that socket.
6. Upon card-insertion events, host software reconfigures the mask and routing registers to support the new card requirements.

serialized IRQ signaling

The serialized interrupt protocol implemented in the PCI1130 uses a single PCI1130 terminal to communicate all interrupt status information to the host interrupt controller. The protocol defines a serial packet consisting of a start cycle, a stop cycle, and multiple interrupt cycles. All data in the packet is synchronous with PCLK. The duration of the stop and interrupt cycles is a fixed number of clock periods, but the start cycle is variable (four, six, or eight clock periods). This allows the serial packet to retain coherence on either side of a PCI-to-PCI bridge.

Figures 5 and 6 illustrate how the serialized IRQ protocol works. Figure 5 shows the start cycle and the first several IRQ sampling periods, and Figure 6 shows the final IRQ sampling periods and the stop cycle. The intermediate IRQ sampling periods are not shown, but the sampling periods occur in ascending IRQ order: IRQ0, IRQ1, SMI, IRQ3, IRQ4 . . . IRQ15, and $\overline{\text{IOCHK}}$. The IRQ signals are active high. In the following illustrations, IRQ1 and IRQ15 are sampled deasserted. The stop cycle can occur no sooner than after the $\overline{\text{IOCHK}}$ period, but can be extended to allow more sampling periods for platform-specific functions.

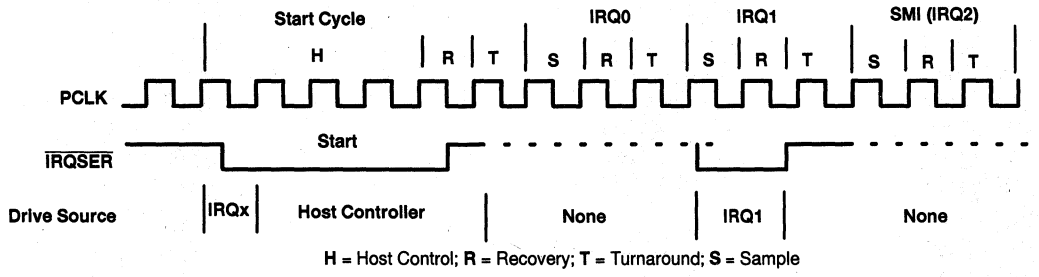


Figure 5. Serial-Interrupt Timing – Start Cycle and IRQ Sampling Periods

PRODUCT PREVIEW

serialized IRQ signaling (continued)

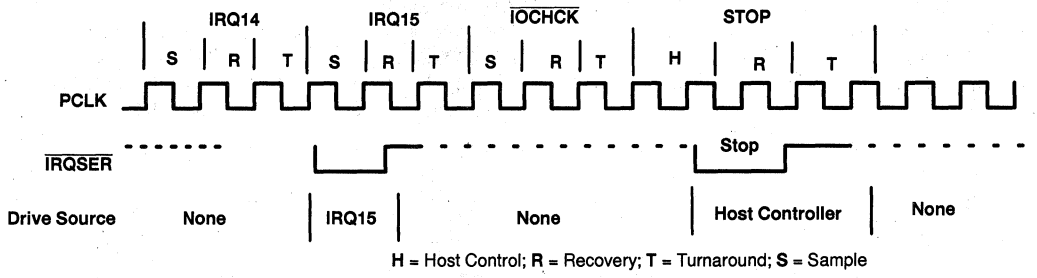


Figure 6. Serial-Interrupt Timing – Stop Cycle

In a system using the serialized IRQ protocol, the host software must configure the PCI1130 to use serialized IRQs by setting bits 2–1 of the PCI configuration register at offset 92h to 10b. The step-by-step series of events that host software must follow to successfully configure the PCI1130 for serialized IRQ signaling is listed below. These steps assume that the system has powered up and $\overline{\text{RSTIN}}$ is high (deasserted). In cases where only select bits of a register are to be modified, host software must leave the remaining register bits unchanged by reading the current contents of the register first, modifying the desired bits, then writing the new value back to the register.

1. Set bits 2–1 of PCI device control register 92h (function 0) to 10b.
2. Write to the upper nibble of ExCA register 05h/45h/805h for desired CSC routing for each socket (all 15 IRQs are available for routing when serialized IRQ signaling has been selected).
3. If a PC Card is installed in the socket and requires functional interrupts, write to the lower nibble of ExCA register 03h/43h/803h for desired functional interrupt routing for the socket.
4. Using Table 12, write to the appropriate mask register bits to enable interrupt generation for desired events.
5. On card-removal events, host software masks any functional interrupts that were set for that socket.
6. Upon card-insertion events, host software reconfigures the mask and routing registers to support the new card requirements.

PCI clock run

The PCI1130 supports PCI clock run (CLKRUN). CLKRUN is an optional signal that is used as an input to determine the status of CLK and as an open-drain output to request CLK to start or to speed up. PCI CLKRUN is enabled by setting bit 0 in the system control register at offset 80h (see *system control register*). When the PCI central resource manager informs the PCI1130 that the PCI clock is stopped or slowed, the PCI1130 ensures that no transactions are in progress for either of the two PC Card sockets before allowing the central resource manager to stop or slow the PCI clock. CLKRUN shares the IRQ10 pin on the PCI1130.

CLKRUN configuration

Bits 1–0 in the TI extension registers at offset 80h are used to enable and configure CLKRUN. Bit 0 enables CLKRUN and bit 1; when set, keeps the PCI clock running in response to a PCI CLKRUN deassertion (see *system control register*).

conditions for stopping/slowing the PCI clock

Before allowing the central resource manager to slow or stop the PCI clock, the following conditions are checked:

- The PCI CLKRUN enable bit is set and the KEEP CLOCK bit is cleared (see *system control register,* bit 1).
- Neither socket is in the process of powering up or powering down.
- The 16-bit resource managers are not busy.
- The CardBus master is not busy.
- The PCI master is not busy.
- No socket interrogation is underway.
- No card interrupts are pending very large scale integration (VLSI).
- The CardBus clock is stopped.

conditions for restarting the PCI clock

The PCI clock restarts when any PC Card is installed in a socket or removed from a socket. For 16-bit cards, if the PCI clock stops or slows, the PCI1130 requests that the clock be restarted under the following conditions:

- A 16-bit I/O card asserts $\overline{\text{IREQ}}$.
- A 16-bit I/O card asserts STSCH/$\overline{\text{RI}}$.
- A 16-bit DMA card asserts $\overline{\text{DREQ}}$.

NOTE:
The 16-bit cards must be powered to restart the PCI clock.

For CardBus cards, if the PCI clock stops or slows, the PCI1130 requests that the clock be restarted under the following conditions:

- A CardBus card asserts $\overline{\text{CINT}}$.
- A CardBus card asserts $\overline{\text{CREQ}}$.
- A CardBus card asserts $\overline{\text{CCLKRUN}}$.
- A CardBus card asserts $\overline{\text{CSTSCHG}}$ (wakeup).

NOTE:
For the first three conditions, the CardBus card must be powered to restart the PCI clock.

PC Card DMA and distributed DMA

DMA is a concept with many different interpretations and implementations, depending on the context and application. In fact, DMA support within the PCI1130 has different connotations, depending on whether the subject is PCI or PC Card DMA. On the PC Card side, the PCI1130 supports the DMA protocol defined in the 1995 PC Card standard on both sockets. On the PCI side, the PCI1130 supports a distributed DMA protocol, compliant with the distributed DMA on the PCIWay, revision 6.0, specification. It also supports PC/PCI DMA in systems designed with the Intel MPIIX.

DMA on PCI is accomplished by compliance with the distributed DMA specification. The PCI1130 complies with this specification as it applies to DMA devices and implements two DMA channels; one per socket. Each DMA channel is controlled by the host via a 16-byte window in PCI I/O address space. This window is mapped in internal PCI1130 registers that are similar, but not identical, to the 8237 DMA controller programming model. By programming these registers, the PCI1130 services DMA requests from PC Card applications by initiating PCI bus mastering cycles to host memory address space.

DMA configuration

Host software must program the PCI1130 socket DMA registers 0 and 1 to set up the socket for DMA transfers. These registers are found in the PCI configuration header, offsets 94h and 98h (see *buffer control register* and *socket DMA register 0*). Socket DMA register 0 applies to the PC Card portion of DMA transfers. Socket DMA register 1 applies to the PCI portion of DMA transfers and complies with the distributed DMA specification.

Socket DMA register 0 has only two significant bits. Bits1–0 encode the $\overline{\text{DREQ}}$ signal used by the PC Card. This field must be programmed with a valid value before the PCI1130 initiates a DMA transfer. Socket DMA register 1 has 16 significant bits, and the encoding is shown in Table 13. The most important field in socket DMA register 1 is the base address that locates the DMA registers in PCI I/O address space. This is how the host communicates and configures the DMA transfer process.

Table 13. Socket DMA Register 1

BIT	TYPE	FUNCTION
31–16	R	Reserved. Bits 31–16 are read only and return 0s when read. Writes have no effect.
15–4	R/W	DMA base address. Bits 15–4 locate the socket 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 four bits are hardwired to 0, forcing the window to a natural 16-byte boundary.
3	R	Nonlegacy extended addressing. This is not supported on the PCI1130 and always returns a 0.
2–1	R/W	Transfer size. Bits 2–1 specify the width of the DMA transfer on the PCI interface. The field is encoded as: 00 = 8-bit transfer (default) 01 = 16-bit transfer 10 = Reserved 11 = Reserved
0	R/W	Decode enable. Enables the decoding of the DMA base address by the PCI1130. This bit is encoded as: 0 = Disabled (default) 1 = Enabled

When host software initializes the PCI1130, the base address in socket DMA register 1 can be programmed, but not enabled. When a particular DMA-capable PC Card is installed in the socket, host software can proceed to program the $\overline{\text{DREQ}}$ signaling option, the datapath width, and enable the DMA register decode in I/O space. These options are specific to the PC Card and must be set when the card is configured, but not when the socket is configured. After setting these options and enabling the DMA register decode, the DMA registers can be programmed. The DMA register programming model is shown in Table 14.

Table 14. DMA Registers

<table>
<tr><th>R/W</th><th colspan="4">REGISTER NAME</th><th>DMA BASE ADDRESS OFFSET</th></tr>
<tr><td>R</td><td rowspan="2">Reserved</td><td rowspan="2">Page</td><td colspan="2">Current address</td><td rowspan="2">00h</td></tr>
<tr><td>W</td><td colspan="2">Base address</td></tr>
<tr><td>R</td><td rowspan="2">Reserved</td><td rowspan="2">Reserved</td><td colspan="2">Current word</td><td rowspan="2">04h</td></tr>
<tr><td>W</td><td colspan="2">Base word</td></tr>
<tr><td>R</td><td>NA</td><td rowspan="2">Reserved</td><td>NA</td><td>Status</td><td rowspan="2">08h</td></tr>
<tr><td>W</td><td>Mode</td><td>Request</td><td>Command</td></tr>
<tr><td>R</td><td rowspan="2">Multichannel mask</td><td rowspan="2">Reserved</td><td>NA</td><td rowspan="2">Reserved</td><td rowspan="2">0Ch</td></tr>
<tr><td>W</td><td>Master clear</td></tr>
</table>

PRODUCT PREVIEW

DMA configuration (continued)

The DMA registers contain control and status information consistent with the 8237 DMA controller; however, the register locations are reordered and expanded in some cases. Refer to *DMA registers* for a detailed description of the individual bits contained in the DMA registers. While the DMA 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 PCI1130 implements these obsolete register bits as read-only, nonfunctional bits. The reserved registers shown in Table 14 are implemented as read only and return 0s when read. Writes to reserved registers have no effect.

DMA transfers

The DMA 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:

1. Setting the proper DMA request ($\overline{\text{DREQ}}$) signal assignment in the PCI configuration, offset 94h (bits 1–0)
2. Setting the proper data width of the DMA transfer in the PCI configuration, offset 98h (bits 2–1)
3. Enabling I/O window decoding of the DMA registers by setting bit 0 in the PCI configuration offset 98h

These steps assume that host software has already powered the PC Card, interrogated its CIS and set the appropriate bits in the PCI1130 that identify the card as a 16-bit I/O PC Card. Also, both I/O access and bus mastering must be enabled in the PCI command register. Host software can then program the DMA registers with the transfer count, direction of the transfer, and memory location of the data. Once this programming is complete, the PCI1130 awaits the assertion of $\overline{\text{DREQ}}$ to initiate the transfer.

DMA writes transfer data from the PC Card to PCI memory addresses. The PCI1130 accepts data 8 or 16 bits at a time (depending on the programming of the data width register field), then requests access to the PCI bus by asserting its $\overline{\text{REQ}}$ signal. Once granted access to the bus and the bus returns to an idle state, the PCI1130 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 PCI1130 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 PCI1130 asserts its PCI $\overline{\text{REQ}}$ signal to request access to the PCI bus. Once access is granted and the bus is idle, the PCI1130 initiates a PCI memory read operation to the current memory address and accepts 8 or 16 bits of data (depending on the programming of the socket DMA register 1 field). After terminating the PCI cycle, the data is passed on to the PC Card. After terminating the PC Card cycle, the PCI1130 requests access to the PCI bus again until the transfer count has expired.

PCI I/O read and write cycles to the DMA registers are accepted and serviced during DMA transfers. If, while a DMA transfer is in progress, the host resets the DMA channel, the PCI1130 asserts TC and ends the PC Card cycle(s). TC is indicated in the DMA status register. At the PC Card interface, the PCI1130 supports demand mode transfers. The PCI1130 asserts DACK the entire duration of the transfer unless $\overline{\text{DREQ}}$ is high (deasserted) before TC. There is no performance penalty for long wait states during this mode of operation, as there is in the legacy ISA system, because the DMA channel is a dedicated resource localized at the PC Card socket.

PC/PCI DMA

The PC/PCI DMA protocol provides a way for legacy I/O devices to do DMA transfers on the PCI bus in systems equipped with the Intel MPIIX. The Intel MPIIX supports PC/PCI DMA expansion for docking station applications where I/O devices require DMA transfers between the docking station PCI bus or extended I/O bus and a PCI bus in the notebook docking computer.

PC/PCI DMA (continued)

In the PC/PCI DMA protocol, the PCI1130 acts as a PCI slave device. The Intel MPIIX DMA controller uses request/grant pairs, $\overline{\text{REQ}}$[A–B] and $\overline{\text{GNT}}$[A–B], which are configured to support a PCI DMA slave device such as the PCI1130. The Intel MPIIX $\overline{\text{REQ}}$ and $\overline{\text{GNT}}$ pins correspond to the PCI1130 IRQ7 and IRQ11 pins, respectively.

Under the PC/PCI protocol, a PCI DMA slave device requests a DMA transfer using a serialized protocol on $\overline{\text{REQ}}$. The Intel MPIIX, as a bus master, arbitrates for the PCI bus. When the Intel MPIIX gets control of the PCI bus, it asserts $\overline{\text{GNT}}$ on the PCI1130 and, for the selected DMA channel, runs the DMA I/O cycles and memory cycles on the PCI bus.

PC/PCI DMA is enabled for each PC Card16 slot by setting bit 19 in the respective system control register (see Table 21). On power up, this bit is cleared, disabling PC/PCI DMA. Bit 3 of each PCI1130 system control register is a global PC/PCI enable bit. When bit 3 is set, the PCI1130 can request a DMA transfer by asserting IRQ7 ($\overline{\text{REQ}}$) and encoding the channel request information using the serialized protocol. When the Intel MPIIX gets control of the PCI bus, it encodes the granted channel on PCI1130 IRQ11 ($\overline{\text{GNT}}$). On power up, bit 3 is cleared and PC/PCI DMA is disabled. When the PCI1130 receives a $\overline{\text{GNT}}$ signal, the PCI1130 looks at the DMA I/O address to determine the type of transfer. The cycle types are as follows:

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

To perform PC/PCI DMA transfers, the following conditions must be met:

- Bit 3 in the system control register must be set to enable the PCI1130 to do PC/PCI DMA transfers.
- The desired DMA channel for each PC Card16 slot (slot A and slot B) must be configured via bits 18–16 in the respective system control register (see Table 21). The Intel MPIIX uses this channel to do the DMA transfers. The channels are configured as follows:

BITS			DMA CHANNEL
18	17	16	
0	0	0	Channel 0
0	0	1	Channel 1
0	1	0	Channel 2
0	1	1	Channel 3
1	0	0	Channel 4
1	0	1	Channel 5
1	1	0	Channel 6
1	1	1	Channel 7

- Each PC Card16 slot must be enabled by setting bit 19 of the respective system control register.

DMA channels 0–3 are used for 8-bit DMA transfers and channels 5–7 are used for 16-bit DMA transfers. On power up, the system control register bits 18–16 default to 100 (channel 4). DMA channel 4 is used by PCI master devices to request the bus; hence, PC/PCI DMA is not the default mode.

The $\overline{\text{REQ}}$ and $\overline{\text{GNT}}$ signal pairs can be configured to support slave devices on the primary bus (i.e., the same bus as the Intel MPIIX) or slave devices on a secondary bus such as a PCI-to-ISA bridge. The $\overline{\text{REQ}}$/$\overline{\text{GNT}}$ pairs are configured by setting the PCI DMA expansion register (offset 088h and 089h, respectively). If the $\overline{\text{REQ}}$/$\overline{\text{GNT}}$ pairs are configured to support a slave device on a secondary bus, the signals must be properly routed to the Intel MPIIX DMA controller, either through the docking station bridge chip or through the docking station connector.

ring indicate

When a 16-bit I/O PC Card is inserted into a socket, the PCI1130 can be configured to allow a ring detect signal to be passed from the PC Card to the system on the IRQ15/RI_OUT pin. This is accomplished by first enabling the RI_OUT function on IRQ15 by setting bit 7 of the card control register (see *card control register*) of the TI extension registers. Next, bit 7 of the ExCA interrupt and general control register (see *ExCA interrupt and general control register*) of the ExCA registers must be set to enable the $\overline{RI}$ Input for the 16-bit I/O PC Card to support the $\overline{RI}$ function. When $\overline{RI}$ sees a low, it is passed through to IRQ15/RI_OUT (see Figure 7). The status of $\overline{RI}$ is reflected in bit 0 of the ExCA card status-change register (see *ExCA card status-change register*) of the ExCA registers.

When a CardBus PC Card is inserted, a status change interrupt can still be output on the RI_OUT pin from CSTSCHG. When in the CardBus mode, the RI_OUT output can still be configured as in the 16-bit PC Card; however, the input must be enabled in the socket mask register (see *CardBus socket mask register*) of the CardBus socket registers.

Setting bit 0 of this register enables the CSTSCHG interrupt to be seen on the RI_OUT pin. This input causes an interrupt when it sees a low-to-high transition. The state of CSTSCHG is reflected in bit 0 of the CardBus socket present state register (see *CardBus socket present state register*) of the CardBus socket registers.

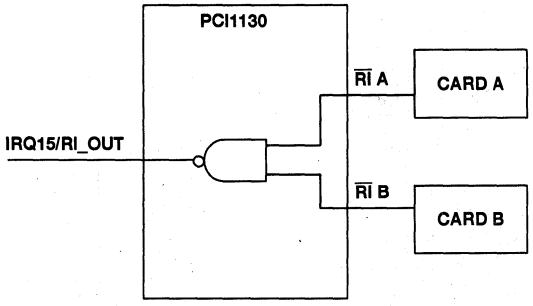


Figure 7. Ring Indicate Enabled on PCI1130

zoom video

The PCI1130 allows the implementation of the zoom video proposal before the PCMCIA. Zoom video is supported by setting bit 6 of the card control register (see *card control register*) in the TI extension registers. Setting this bit puts 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. Address lines A3-A0 can still be used by the PCI1130 to access PC Card CIS registers for PC Card configuration (see Figure 8).

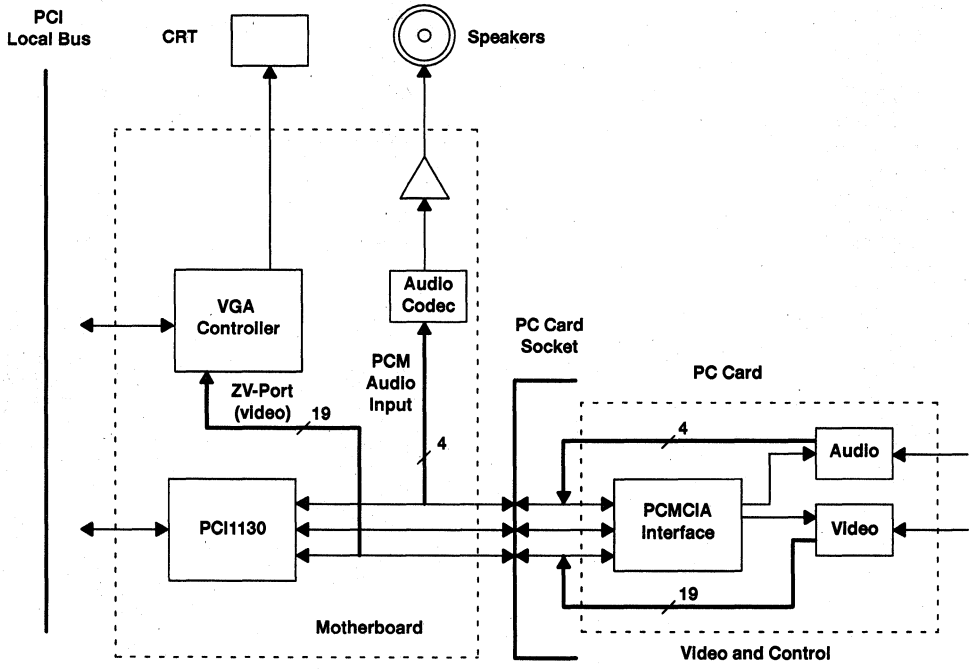


Figure 8. Zoom-Video Implementation on the PCI1130

power management

The PCI1130 provides four methods of power management. These methods relate to the primary bus (PCI) and the secondary bus (CardBus and PC Card16). Managing the PCI clock and CardBus clock is the main method of conserving power on the PCI1130.

PCI power management

The PCI clock run feature is the primary method of power management on the PCIBus side of the PCI1130. To enable the PCI1130 to operate correctly in the suspend and resume configurations of different chipsets, the PCI1130 implements $\overline{\text{SUSPEND}}$ that allows $\overline{\text{RSTIN}}$ ($\overline{\text{PCIRST}}$) to be asserted as the system resumes, while preserving the state of the PCI1130 internal registers.

PCI clock run

The PCI1130 supports the PCI clock run protocol as defined in the PCI mobile design guide revision 1.0. When the system's central resource signals the system to stop the PCI clock by driving $\overline{\text{CLKRUN}}$ high, the PCI1130 either signals that it is acceptable to stop the PCI clock by not driving $\overline{\text{CLKRUN}}$ or signals the system to keep the clock running by pulling $\overline{\text{CLKRUN}}$ low.

The PCI1130 $\overline{\text{CLKRUN}}$ is multiplexed on the IRQ10 interrupt line. The PCI1130 clock run feature is enabled by setting bit 0 in the system control register 80h (see *system control register*). Bit 0 enables/disables the PCI clock run functionality of the multiplexed pin IRQ10/$\overline{\text{CLKRUN}}$. Bit 1 of the system control register allows software to enable the PCI1130 keep clock running mode to prevent the system from stopping the PCI clock. When bit 1 of the system control register is set, the PCI1130 signals back to the system to keep the PCI clock running (not to stop the clock).

If a CardBus PC Card is inserted, the PCI1130 needs to use the clock run protocol on the CardBus interface to stop the CardBus clock. If the CardBus PC Card allows the CardBus clock to be stopped, the PCI1130 allows the PCI clock to be stopped. If the CardBus clock is not stopped on the first attempt, the PCI1130 signals to keep the PCI clock running. This gives the PCI1130 the time required to stop the CardBus clock. If the CardBus clock is already stopped, the PCI1130 can allow the PCI clock to be stopped. Figure 9 shows a diagram of the PCI bus states and the logic level of $\overline{\text{CLKRUN}}$ for each state.

The PCI1130 signals the system to restart the clock when one of the following events occur:

- A card is inserted or removed. The PCI1130 signals to start the PCI clock and generates a card status-change interrupt on the CSC interrupt routing.
- A functional interrupt is generated by a PC Card. The PCI1130 signals to start the PCI clock and generates a functional interrupt on the appropriate routing.
- A ring indicate ($\overline{\text{RI}}$) signal is detected by a PC Card16. The PCI1130 signals to start the PCI clock and a ring indicate output (RI_OUT) signal is provided to the system.
- A CardBus PC Card signals to the PCI1130 to start the clock using the clock run protocol. The PCI1130 signals to start the PCI clock.

PCI clock run (continued)

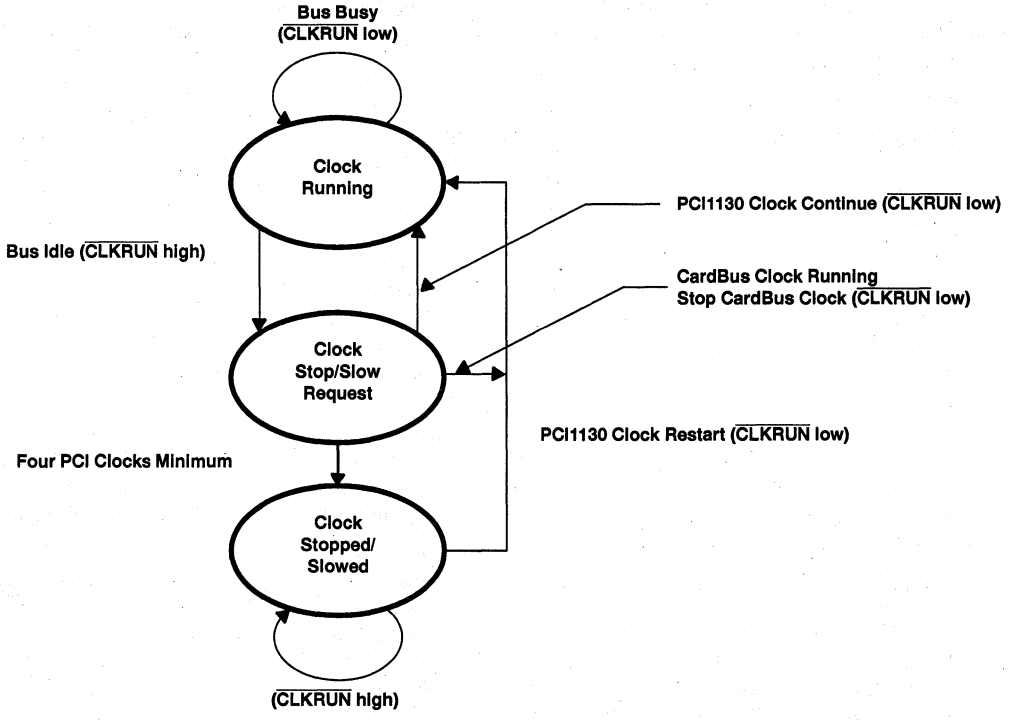


Figure 9. Clock Run and Bus States

PCI suspend/resume

The PCI1130 implements a suspend feature that allows RSTIN to be asserted without the resetting PCI1130 internal registers. SPKROUT is multiplexed with SUSPEND. The multiplex control is provided in the PCI configuration space by setting bit 1 of the card control register 92h (see *card control register*). Some chipsets provide a PCIRESET signal that is asserted when the system resumes after a suspend period. With these particular chipsets, the PCI1130 suspend feature should be implemented to allow the system to activate suspend without clearing the internal registers on the PCI1130 (see Figure 10). If a chipset does not require suspend, SUSPEND can be pulled high or SPKROUT can be activated. The PCI1130 default state for SUSPEND is active.

Any bus contention between SPKROUT and SUSPEND is avoided because the PCI1130 implements a three-PCI-clock-cycle delay after the control bit in the card control register has been changed. This allows the pullup resistor on the pin to pull the line high so that an erroneous suspend mode does not occur.

PCI suspend/resume (continued)

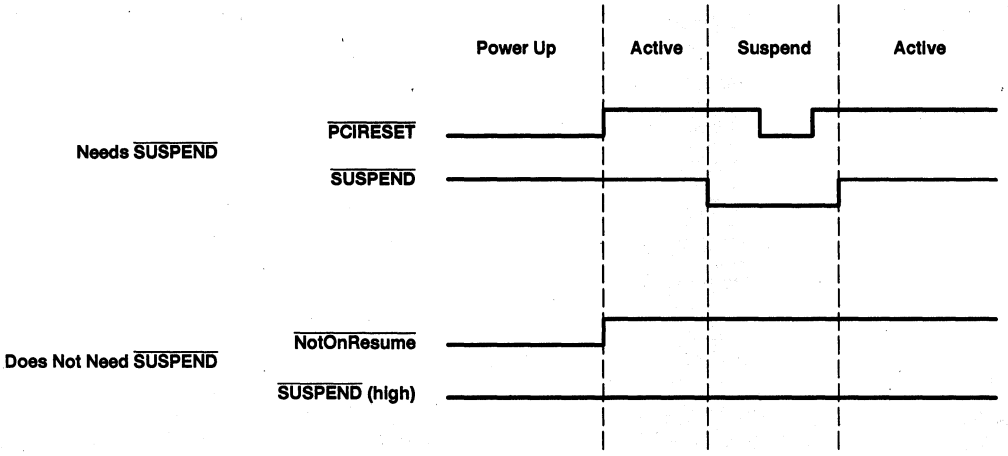


Figure 10. PCI Reset and Suspend Mode

CardBus clock run mode

When the PCI1130 is configured to support CardBus PC Card sleep mode, it allows CardBus PC Card clock (CCLK) to be stopped (0 MHz), or to be slowed down to a frequency equal to a divide-by-16 of the PCI clock frequency, using the CCLKRUN protocol as specified in the PC Card standard. The PC Card must be powered, reset deasserted, and no activity on the socket for eight CardBus clock cycles before the requesting to slow or stop the CardBus PC Card clock. Activity on the socket is determined by monitoring signals from the PC Card. These signals are: CFRAME, CIRDY, CREQ, and CBLOCK. Any transaction requests from the PCIBus before completion of eight inactive clock cycles keeps the CardBus clock from slowing down or stopping.

If the CardBus socket is in sleep mode, any activity on the PC Card interface or accesses to the PC Card from the PCIBus causes the CardBus PC Card clock to return to the PCI clock frequency. The CardBus PC Card clock also can be restarted using CCLKRUN on the PC Card interface. The CardBus PC Card clock is returned to the PCI clock frequency after two PCI clock cycles. If no activity occurs for eight PCI clock cycles after the CardBus PC Card clock has returned to the PCI clock frequency, the PCI1130 requests the PC Card to place the CardBus socket in sleep mode using CCLKRUN. If the PC Card does not signal the PCI1130 (using CCLKRUN) to stop or slow the CardBus PC Card clock within eight PCI clock cycles, the CardBus PC Card clock returns to the configured sleep mode.

The following configuration steps are required to enable the different forms of the CardBus PC Card sleep mode. The first configuration places the CardBus PC Card clock (CCLK) into a divide-by-16 PCI clock frequency mode:

1. Ensure that bit 7 in the CardBus socket control register is cleared (low).†
2. Set bit 16 in the CardBus socket power management register.‡
3. Clear bit 0 in the CardBus socket power management register.‡

† See *CardBus socket control register*
‡ See *CardBus socket power management register*

PRODUCT PREVIEW

CardBus clock run mode (continued)

There are two configurations to place the PCI1130 in a sleep mode that stops the CardBus PC Card clock (CCLK). The first is:

1. Ensure that bit 7 in the CardBus socket control register is cleared (low).†
2. Set bit 16 in the CardBus socket power management register.‡
3. Set bit 0 in the CardBus socket power management register.‡

The second method for placing the socket in stopped clock sleep mode is:

1. Ensure that bit 16 in the CardBus socket power management register is cleared (low).‡
2. Set bit 7 in the CardBus socket control register.†

† See *CardBus socket control register*
‡ See *CardBus socket power management register*

PC Card16 mode

When a 16-bit legacy PC Card is inserted into a socket, there are two options for minimizing power consumption. The first is to use the card output enable (COE) bit (see bit 7 of the *ExCA power control register*). When bit 7 is set, the outputs on the PC Card socket are placed in the high-impedance state. Bit 7 is software controlled. Socket services have to clear this bit to activate the socket. The second method is to set the power-down bit (see bit 0 of the *ExCA global control register*). When bit 0 is set, it enables an automated COE bit. When a card access to a PC Card16 card is complete, the PCI1130 automatically places the card outputs in the high-impedance state. When there is any activity on the socket, the outputs are automatically enabled.

The major difference between the use of the COE bit and the POWERDWN bit is that the COE bit resets the PC Card16 PC Card and the POWERDWN bit does not. The POWERDWN bit continues to drive the Card RESET line inactive, while the COE bit puts the RESET line in the high-impedance state.

PCI configuration headers

A number of registers found in the PCI1130 PCI configuration space are defined in the PCI-to-PCI bridge architecture specification revision 1.0, which, in turn, are common to the PCI local bus specification revision 2.1. Registers common to both are the device ID, vendor ID, status, command, class code, revision ID, BIST, header type, latency timer, cache line size, interrupt pin, and interrupt line registers. The special needs of a PCI-to-PCI bridge and the PCI1130 require additional registers in the form of the CardBus latency timer, subordinate bus number, CardBus bus number, PCI bus number, secondary status and bridge control registers. Conceptually, the CardBus bus that the PCI1130 controls is analogous to the secondary bus of a PCI-to-PCI bridge.

The PCI specific registers listed in the previous paragraph are applicable to the entire device and are not specific to any one PCI function (i.e., PC Card socket) on the PCI1130. Each register is mapped to the same location in both PCI configuration spaces. Access is possible by addressing the configuration space of either function, but host software should consistently access PCI specific registers through a single function. Detailed descriptions of the PCI specific registers follow and are listed in Table 16. Most of the registers are implemented in the PCI1130 as defined in either the PCI local bus specification revision 2.1, the PCI-to-PCI bridge architecture specification revision 1.0, or the Yenta specification revision 2.1. References to these documents are made where appropriate. Additional register bits defined in the bridge control register (see *bridge control*) enable features specific to CardBus memory windows.

PCI configuration headers (continued)

Host software exerts control and retrieves status information on PC Cards via a standard set of internal PCI1130 registers: ExCA registers for 16-bit PC Cards and socket registers for CardBus PC Cards. The PCI1130 maps these registers into PCI address space for access by host software. The locations of these registers are set by the CardBus socket registers/ExCA registers base address register (see *CardBus socket registers/ExCA registers base address register*) register in PCI configuration space, which locates a 4K-byte nonprefetchable memory window in PCI memory address space. Within this memory window, the PCI1130 maps both the CardBus socket registers and the ExCA registers. Each socket has a separate CardBus socket register/ExCA registers base address register for accessing the ExCA registers.

The 16-bit PC Cards use the ExCA register set for card status and control purposes. Traditionally, these registers have been accessed by host software through an index/data register pair. Software would write the index of the desired ExCA register to the index register, and read or write the desired data to the data register. The PCI1130 departs from this scheme by directly mapping the ExCA register set to a 4K-byte memory window located by the CardBus socket registers/ExCA registers base address register. The ExCA registers are offset from this base address by 800h. The PCI1130 also supports the index/data scheme of accessing the ExCA registers through the use of the PC Card 16-bit IF legacy-mode base-address register (see *PC Card 16-bit IF legacy-mode base address*). An address written to this register becomes the address for the index register and the address+1 becomes the address for the data address. Using this access method, applications requiring index/data type ExCA access can be supported. This PC Card 16-bit legacy-mode base address is shared by both sockets and the ExCA registers run contiguously from offset 00h–3Fh for socket A and 40h–7Fh for socket B.

The PCI1130 implements a set of six 32-bit socket registers, defined by the PCMCIA CardBus specification, which the PCI1130 maps into PCI memory space. The location of these registers in PCI memory space is set by the CardBus socket registers/ExCA registers base address register (see *CardBus socket registers/ExCA registers base address register*), also in PCI configuration space. This 32-bit base address allows the CardBus socket registers to be anywhere within the 4G-byte PCI memory address range. The CardBus socket registers/ExCA registers base address register marks the beginning of a 4K-byte block of addresses for which the PCI1130 claims PCI memory cycles. Bit 1 of the PCI1130 command register (see *PCI command*) must be set to claim PCI memory cycles.

Memory and I/O windows to CardBus are defined by base and limit registers in the configuration header. The PCI1130 provides two memory and two I/O windows for each CardBus PC Card, each with 4K-byte and double-word granularity, respectively. The PCI1130 uses these registers to decode and claim PCI or CardBus memory or I/O cycles and pass them to the appropriate bus. Note that bits 0 and 1 of the PCI1130 command register must be set to claim PCI I/O and memory cycles. This is because these memory and I/O windows are used to decode CardBus cycles. Bits 0 and 1 in the command register can be cleared to prevent the PCI1130 from claiming CardBus cycles destined for PCI.

The PCI1130 implements two PCI configuration headers, one for each PC Card socket; therefore, all memory and I/O window functionality for socket A are repeated, but separate from, socket B. It is the responsibility of host software to program nonoverlapping memory and I/O resources for each socket.

The TI extension registers are specific PCI1130 value-added features that are not part of currently defined industry specifications. The TI extension registers are a collection of control and status bits that are required to support various PCI1130 functionalities. This functionality typically does not exist within the register models implemented elsewhere within the device. Tables 15 and 16 show the TI extension registers and their locations in PCI configuration space.

PCI configuration headers (continued)

Table 15. TI Extension Registers

REGISTER NAME	OFFSET
System control register†	80h
Retry status register†	90h
Card control register†	91h
Device control register†	92h
Buffer control register†	93h

† One or more bits in the register are common to PCI functions 0 and 1.

The PCI1130 supports the DMA specification defined in the 1995 PC card standard by providing one DMA channel per socket. The PC card standard stipulates the signaling and timing associated with DMA transfers to and from a PC Card. This defines DMA transfers from the PC Card to the socket only. On the PCI side, the PCI1130 implements a set of status and control registers similar, but not identical, to the programming model of the original dual 8237 DMA controller found in PC-AT systems. These registers comply with the specification for distributed DMA in a PCI environment, particularly as it defines DMA devices. The PCI1130 provides two registers in its configuration header that set up both the PCI interface and PC Card socket for DMA. See *PC Card DMA and distributed DMA* for a complete discussion of DMA support on the PCI1130.

Host software must program the PCI1130 socket DMA registers 0 and 1 to set up the socket for DMA transfers. Socket DMA register 0 applies to the PC Card portion of DMA transfers. Socket DMA register 1 applies to the PCI portion of DMA transfers specifically to set up the DMA support required in distributed DMA. Socket DMA register 1 provides register bits to program the DMA transfer width. This transfer width refers to both the PC Card interface and the PCI interface.

PCI configuration headers (continued)

Descriptions of each of the registers follow. Before writing data to each of the TI extension registers, host software must first read the register to preserve the current contents. After reading the register, software can modify the desired bits and write back the new data. This preserves current register settings and prevents unpredictable or undesirable behavior.

The PCI1130 configuration header is shown in Table 16.

Table 16. PCI1130 Configuration Header

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 registers/ExCA base-address register				10h
Secondary status		Reserved		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 IF legacy-mode base address†				44h
Reserved				48h–7Ch
System control register†				80h
Reserved				84h–8Ch
Buffer control†	Device control†	Card control†	Retry status†	90h
Socket DMA register 0				94h
Socket DMA register 1				98h
Reserved				9Ch–FFh

† One or more bits in the register are common to PCI functions 0 and 1.

PCI vendor ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PCI 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: **PCI Vendor ID**
Type: Read Only
Offset: 00h
Default: 104Ch
Description: This 16-bit value is allocated by the PCI SIG (special interest group) and identifies TI as the manufacturer of this device. The vendor ID assigned to TI is 104Ch.

PCI device ID

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PCI 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	0	0	1	0

Register: **PCI Device ID**
Type: Read Only
Offset: 02h
Default: AC12h
Description: This 16-bit value is allocated by the vendor. The device ID for the PCI1130 is AC12h.

PCI command

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PCI 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: **PCI Command**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 04h
Default: 0000h
Description: The PCI command register provides control over the PCI1130's ability to generate and respond to PCI cycles. In its default state, or when 0000h is written, the PCI1130 can respond to PCI configuration cycles only; all other PCI functionality is disabled. The PCI1130 does not claim PCI cycles as a target, nor request access to the bus as an initiator in this state. Table 17 describes each bit in the command register.

PCI command (continued)

Table 17. PCI Command Register

BIT	TYPE	FUNCTION
15–10	R	Reserved. Bits 15–10 are read only and return 0s when read. Writes have no effect.
9	R	Fast back-to-back enable. Bit 9 indicates whether the device is enabled for the fast back-to-back transaction function. The PCI1130 does not support fast back-to-back PCI cycles. Bit 9 is read only and returns 0s when read.
8	R/W	System error ($\overline{\text{SERR}}$) enable. Both bit 8 and bit 6 must be set for the PCI1130 to report address parity errors. 0 = Disable the $\overline{\text{SERR}}$ output driver (default) 1 = Enable the $\overline{\text{SERR}}$ output driver
7	R	Wait cycle control. Bit 7 indicates whether a PCI device is capable of address/data stepping. The PCI1130 does not support address/data stepping; therefore, this bit is hardwired to 0. Bit 7 is read only and returns 0s when read. Writes to this bit have no effect.
6	R/W	Parity error response. Data parity errors are indicated by asserting $\overline{\text{PERR}}$, while address parity errors are indicated by asserting $\overline{\text{SERR}}$. 0 = PCI1130 ignores detected parity error (default) 1 = PCI1130 responds to detected parity errors
5	R	VGA palette snoop. Bit 5 controls how PCI devices handle accesses to VGA palette registers. The PCI1130 does not support VGA palette snooping; therefore, this bit is hardwired to 0. Bit 5 is read only and returns 0s when read. Writes to this bit have no effect.
4	R	Memory write and invalidate enable. Bit 4 controls whether a PCI initiator device can generate memory write and invalidate commands. The PCI1130 uses memory-write commands instead of memory-write-and-invalidate commands; therefore, this bit is hardwired to 0. Bit 4 is read only and returns 0s when read. Writes to this bit have no effect.
3	R	Special cycles. Bit 3 controls whether or not a PCI device ignores PCI special cycles. The PCI1130 does not monitor special cycle operations; therefore, this bit is hardwired to 0. Bit 3 is read only and returns 0s when read. Writes to this bit have no effect.
2	R/W	Bus initiator control. Bit 2 controls whether or not a PCI device can act as a PCI bus initiator. 0 = Disables the PCI1130's ability to generate PCI bus accesses (default) 1 = Enables the PCI1130's ability to generate PCI bus accesses
1	R/W	Memory space control. Bit 1 controls whether or not a PCI device can claim cycles in PCI memory space. 0 = Disables the PCI1130's response to memory space accesses (default) 1 = Enables the PCI1130's response to memory space accesses
0	R/W	I/O space control. Bit 0 controls whether or not a PCI device can claim cycles in PCI I/O space. 0 = Disables the PCI1130's response to I/O space accesses (default) 1 = Enables the PCI1130's response to I/O space accesses

PCI status

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PCI Status															
Type	R/W	R/W	R/W	R/W	R/W	R	R	R/W	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: **PCI Status**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 06h
Default: 0200h
Description: The PCI status register provides PCI-related device information to the host system. Bits in this register can be read normally; however, writes behave differently. 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. Table 18 describes each bit in the PCI status register.

Table 18. PCI Status Register

BIT	TYPE	FUNCTION
15	R/W	Parity error status 0 = PCI1130 does not detect a parity error (default). 1 = PCI1130 detects a parity error.
14	R/W	System error status 0 = PCI1130 does not generate a system error on the $\overline{\text{SERR}}$ line (default). 1 = PCI1130 generates a system error on the $\overline{\text{SERR}}$ line.
13	R/W	Initiator abort status 0 = A bus initiator abort does not terminate a bus initiator's transaction (default). 1 = A bus initiator abort terminates a bus initiator's transaction.
12	R/W	Target abort status. A target abort terminates a PCI1130 bus master transaction. 0 = A target abort does not terminate a PCI1130 bus master transaction (default). 1 = A target abort terminates a bus master transaction.
11	R/W	Target abort status. The PCI1130 target abort terminates a bus master transaction. 0 = A PCI1130 target does not terminate a bus master transaction (default). 1 = A PCI1130 target terminates a bus master transaction.
10–9	R	Device select timing status. Bits 10–9 are encoded with the $\overline{\text{DEVSEL}}$ timing. Bits 10–9 are hardwired as 01b, indicating a medium speed device.
8	R/W	Data parity status 0 = No data parity errors occur (default). 1 = Data parity errors occur; the following conditions are met: a. $\overline{\text{PERR}}$ is asserted by the bus initiator or the bus initiator observed $\overline{\text{PERR}}$ asserted. b. The agent that set the bit is the bus initiator during the transaction when the error occurred. c. Parity error response (bit 6 in the command register) is enabled.
7	R	Fast back-to-back capable. The PCI1130 cannot accept fast back-to-back transactions; therefore, bit 7 is hardwired to 0.
6	R	User-definable feature (UDF) support. The PCI1130 does not support the UDF option; therefore, bit 6 is hardwired to 0.
5	R	66 MHz capable. The PCI1130 operates at a maximum frequency of 33 MHz; therefore, bit 5 is hardwired to 0.
4–0	R	Reserved. Bits 4–0 are read only and return 0s when read. Writes have no effect.

PRODUCT PREVIEW

PCI revision ID

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

Register: **PCI Revision ID**
Type: Read Only
Offset: 08h
Default: 02h
Description: The revision ID register is selected by TI and indicates the silicon revision. The PCI1130 initial silicon revision is 02h.

PCI class code

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
Byte	Base Class								Sub Class								Programming Interface							
Name	PCI Class Code																							
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: The class code indicates that the PCI1130 is a bridge device (06h), CardBus bridge (07h), with 00h programming interface.

cache line size

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 used by host software to program the system cache line size. The cache line size register decodes values as follows:

00h – 03h – Burst mode disabled
04h – 07h – Four double-word burst-mode enabled
08h – FFh – Eight double-word burst-mode enabled

PRODUCT PREVIEW

PCI latency timer

Bit	7	6	5	4	3	2	1	0
Name	PCI 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: **PCI Latency Timer**
Type: Read/Write
Offset: 0Dh
Default: 00h
Description: This register specifies the latency timer for the PCI1130 in units of PCI clock cycles. When the PCI1130 is a bus initiator and asserts $\overline{\text{FRAME}}$, the latency timer begins counting from 0. If the latency timer expires before the PCI1130 transaction is terminated, the PCI1130 terminates the transaction when its $\overline{\text{GNT}}$ is high (deasserted). A recommended minimum value for this register is 20h. This allows most transactions to be completed.

PCI header type

Bit	7	6	5	4	3	2	1	0
Name	PCI Header Type							
Type	R	R	R	R	R	R	R	R
Default	1	0	0	0	0	0	1	0

Register: **PCI Header Type**
Type: Read Only
Offset: 0Eh
Default: 82h
Description: The PCI header type register indicates that the PCI1130 uses a CardBus bridge configuration header. It also identifies the PCI1130 as a multifunction device.

BIST

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: The PCI1130 does not support built-in self test (BIST); therefore, this register is considered reserved. The BIST register is read only and returns 0s when read. Writes to this register have no effect.

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 Registers/ExCA Registers Base Address Register															
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 Registers/ExCA Registers Base Address Register															
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 Registers/ExCA Registers Base Address Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 10h
Default: 0000 0000h
Description: This register points to the nonprefetchable memory window where the PCI1130 maps both the CardBus socket registers and the ExCA registers. The register is separated into two fields. Bits 31–12 are read/write and allow the CardBus socket registers/ExCA registers to be located anywhere in the 32-bit PCI I/O address space on 4K-byte boundaries. Bits 11–0 are read only and are hardwired to 0 to indicate that this register represents a memory base address. When software writes a value of all 1s to this register, the value read back is FFFF F000h, indicating that at least 4K bytes of memory address space are required.

NOTE:

CardBus status and control registers start at offset 000h and the 16-bit card registers begin at offset 800h.

secondary status

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Secondary Status															
Type	R/W	R/W	R/W	R/W	R/W	R	R	R/W	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 (see individual bit descriptions)
Offset: 16h
Default: 0200h
Description: This register complies with the secondary status register definition found in the PCI-to-PCI bridge architecture specification revision 1.0. Writing a 1 to a bit clears it. Table 19 describes each bit in the secondary status register.

Table 19. Secondary Status Register

BIT	TYPE	FUNCTION
15	R/W	Parity error detect status 0 = PCI1130 does not detect a parity error (default). 1 = PCI1130 detects a parity error.
14	R/W	System error 0 = PCI1130 does not receive a system error on the $\overline{\text{SERR}}$ line (default). 1 = PCI1130 receives a system error on the $\overline{\text{SERR}}$ line.
13	R/W	Initiator abort status 0 = A bus initiator abort does not terminate a bus initiator's transaction (default). 1 = A bus initiator abort terminates a bus initiator's transaction.
12	R/W	Target abort status. A CardBus target abort terminates a PCI1130 bus master transaction. 0 = A CardBus target abort does not terminate a PCI1130 bus master transaction (default). 1 = A CardBus target abort terminates a PCI1130 bus master transaction.
11	R/W	Target abort status. A PCI1130 target abort terminates a CardBus bus master transaction. 0 = A PCI1130 target abort does not terminate a CardBus bus master transaction (default). 1 = A PCI1130 device target abort terminates a CardBus bus master transaction.
10–9	R	Device select timing status. Bits 10–9 are encoded with the $\overline{\text{DEVSEL}}$ timing. Bits 10–9 are hardwired to 01b, indicating a medium speed device.
8	R/W	Data parity status 0 = No data parity errors occur (default). 1 = Data parity errors occur; the following conditions are met: a. $\overline{\text{PERR}}$ is asserted by the bus initiator or the bus initiator observed $\overline{\text{PERR}}$ asserted. b. The agent that set the bit was the bus initiator during the transaction when the error occurred. c. Parity error response (bit 6 in the command register) is enabled.
7	R	Fast back-to-back capable. The PCI1130 cannot accept fast back-to-back transactions; therefore, bit 7 is hardwired to 0.
6	R	User-definable feature (UDF) support. The PCI1130 does not support the UDF option; therefore, bit 6 is hardwired to 0.
5	R	66 MHz capable. The PCI1130 operates at a maximum frequency of 33 MHz; therefore, bit 5 is hardwired to 0.
4–0	R	Reserved. Bits 4–0 are read only and return 0s when read. Writes have no effect.

PCI bus number

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 represents the bus number of the PCI bus (also referred to as the primary bus) to which the PCI1130 is connected. The PCI1130 uses this register, the CardBus bus number and subordinate bus number register to determine when to forward PCI configuration cycles to its secondary buses.

CardBus bus number

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 represents the bus number of the CardBus bus (also referred to as the secondary bus) to which the PCI1130 is connected. The PCI1130 uses this register, the PCI bus number and subordinate bus number registers to determine when to forward PCI configuration cycles to its secondary buses.

subordinate bus number

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 represents the bus number of the highest numbered bus below the CardBus bus. The PCI1130 uses this register, the PCI bus number and CardBus bus number registers to determine when to forward PCI configuration cycles to its secondary buses.

CardBus latency timer

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 specifies the CardBus latency timer for the PCI1130 in units of CCLK cycles. When the PCI1130 is a CardBus initiator and asserts $\overline{\text{CFRAME}}$, the CardBus latency timer begins counting from 0. If this latency timer expires before the PCI1130 transaction is terminated, the PCI1130 terminates the transaction at the end of the next data phase. A recommended minimum value for this register is 20h. This value 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 (see individual bit descriptions)
Offset: 1Ch, 24h
Default: 0000 0000h
Description: The memory base registers indicate the lower address of a PCI memory address range and are used by the PCI1130 to determine when to forward a memory transaction to the CardBus bus and when to forward a CardBus cycle to PCI. Bits 31–12 of these registers are read/write and allow the memory base address to be located anywhere in the 32-bit PCI memory space on 4K-byte boundaries. Bits 11–0 are read only and always return 0s. Writes to these bits have no effect. The PCI1130 assumes that the lower 12 bits of the base address are 0. Bits 8 and 9 of the bridge control register (see *bridge control*) specify whether memory windows 0 and 1 are prefetchable or nonprefetchable.

NOTE:
The memory base register or the memory limit register must not be 0 for the PCI1130 to claim any memory transactions.

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 (see individual bit descriptions)
Offset: 20h, 28h
Default: 0000 0000h
Description: The memory limit registers indicate the upper address of a PCI memory address range and are used by the PCI1130 to determine when to forward a memory transaction to the CardBus bus and when to forward a CardBus cycle to PCI. Bits 31–12 of these registers are read/write and allow the memory base address to be located anywhere in the 32-bit PCI memory space on 4K-byte boundaries. Bits 11–0 are read only and always return 0s. Writes to these bits have no effect. The PCI1130 assumes that the lower 12 bits of the limit address are 1s. Bits 8 and 9 of the bridge control register (see *bridge control*) specify whether memory windows 0 and 1 are prefetchable or nonprefetchable.

NOTE:
The memory base register or the memory limit register must not be 0 for the PCI1130 to claim any memory transactions.

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 (see individual bit descriptions)
Offset: 2Ch, 34h
Default: 0000 0000h
Description: The I/O base registers indicate the lower address of a PCI I/O address range and are used by the PCI1130 to determine when to forward a I/O transaction to the CardBus bus and when to forward a CardBus cycle to PCI. The lower 16 bits of this register locate the bottom of the I/O window within a 64K-byte page, and the upper 16 bits are a page register that locates this 64K-byte page in the 32-bit PCI I/O address space. Bits 31–2 are read/write. Bits 1–0 are read only and always return 0s, forcing the I/O window to be aligned on a natural double-word boundary. Writes to these bits have no effect. The PCI1130 assumes that the lower two bits of the base address are 0.

NOTE:
The I/O base register or the I/O limit register must not be 0 for the PCI1130 to claim any I/O transactions.

PRODUCT PREVIEW

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 (see individual bit descriptions)
Offset: 30h, 38h
Default: 0000 0000h
Description: The I/O limit registers indicate the upper address of a PCI I/O address range and are used by the PCI1130 to determine when to forward a I/O transaction to the CardBus bus and when to forward a CardBus cycle to PCI. Bits 15–2 of these registers 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 register) on double-word boundaries. Bits 31–16 and bits1–0 are read only and always return 0s. Writes to these bits have no effect. The PCI1130 assumes that the lower two bits of the limit address are 1s.

NOTE:
The I/O limit register or the I/O limit register must not be 0 for the PCI1130 to claim any I/O transactions.

interrupt line

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: The contents of this register default to the FFh, or unknown, condition. This register is not used by the PCI1130, due to the number of interrupt signaling options and the variety of interrupt sources within the device. Host software must use the proper internal registers to monitor and set interrupt routing. This register is considered reserved; however, host software can read and write to this register.

Interrupt pin

Bit	7	6	5	4	3	2	1	0
Name	Interrupt Pin							
Type	R	R	R	R	R	R	R	R
Function 0 (Socket A)	0	0	0	0	0	0	0	1
Function 1 (Socket B)	0	0	0	0	0	0	1	0

Register: **Interrupt Pin**
Type: Read Only
Offset: 3Dh
Default: 01h for function 0 (socket A) and 02h for function 1 (socket B)
Description: This register is hardwired and writes to the register have no effect. The return values for the register are 01h for function 0 (socket A) and 02h for function 1 (socket B).

bridge control

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/W
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 PCI1130 bridging functions. Table 20 describes each bit in the bridge control register.

bridge control (continued)

Table 20. Bridge Control Register

BIT	TYPE	FUNCTION
15–11	R	Reserved. Bits 15–11 are read only and return 0s when read. Writes have no effect.
10	R/W	Write posting enable. Enables posting of write data to and from the socket. If bit 10 is not set, the bridge must drain any data in its buffers before accepting data for or from the socket. Each data word must then be accepted by the target before the bridge can accept the next word from the source master. The bridge must not release the source master until the last word is accepted by the target. Operating with write posting disabled inhibits system performance. Bit 10 is encoded as: 0 = Write posting is disabled (default). 1 = Write posting is enabled.
9	R/W	Memory window 1 type. Bit 9 specifies whether memory window 1 is prefetchable or nonprefetchable. When the PCI1130 gets a memory read, all byte enables are enabled on the target bus. Bit 9 is encoded as: 0 = Memory window 1 is nonprefetchable. 1 = Memory window 1 is prefetchable (default).
8	R/W	Memory window 0 type. Bit 8 specifies whether memory window 0 is prefetchable or nonprefetchable. When the PCI1130 gets a memory read, all byte enables are enabled on the target bus. Bit 8 is encoded as: 0 = Memory window 0 is nonprefetchable. 1 = Memory window 0 is prefetchable (default).
7	R/W	PCI interrupt-IREQ routing enable bit. When bit 7 is 0 and the PCI interrupt bit in device control register (see *device control register*) is enabled, the functional card interrupts are routed to the PCI interrupt for the socket (INTA or INTB). When bit 7 is 1, the functional card interrupt is routed to an IRQ pin, using the routing selected in the ExCA card interrupt and general control register (see *ExCA interrupt and general control register*). Bit 7 is encoded as: 0 = Functional interrupts are routed to PCI interrupts (default). 1 = Functional interrupts are routed by ExCA registers.
6	R/W	CardBus reset. When bit 6 is set to 1, the PCI1130 asserts and holds $\overline{\text{CRST}}$. When cleared, the bridge deasserts $\overline{\text{CRST}}$. Bit 6 can be set by software. It is also set by hardware when the PCI1130 executes the power-down sequence. Bit 6 is cleared only by software. CRST is a wired-OR of bit 6 and $\overline{\text{RSTIN}}$. Bit 6 is encoded as: 0 = $\overline{\text{CRST}}$ is deasserted. 1 = $\overline{\text{CRST}}$ is asserted (default).
5	R/W	Master abort mode. Bit 5 controls how the PCI1130 responds to a master abort when the PCI1130 is a master. Bit 5 is common between each socket. Bit 5 is encoded as: 0 = Master aborts not reported (default) 1 = Signal target abort and $\overline{\text{SERR}}$, if enabled
4	R	Reserved. Bit 4 is read only and returns 0 when read. Writes have no effect.
3	R/W	VGA enable. Bit 3 affects how the PCI1130 responds to VGA addresses. Bit 3 is common between each socket. Bit 3 is encoded as: 0 = Normal operation (default) 1 = Accesses to VGA addresses are forwarded.
2	R/W	ISA enable. Bit 2 affects how the PCI1130 passes I/O cycles within the 64K-byte ISA range. Bit 2 is not common between each socket. Bit 2 is encoded as: 0 = Normal operation (default) 1 = Last 768 bytes of each 1K are not forwarded to CardBus.
1	R/W	$\overline{\text{SERR}}$ enable. Bit 1 controls the response of the PCI1130 to $\overline{\text{SERR}}$ signals on the CardBus bus. Bit 1 is common between each socket. Bit 1 is encoded as: 0 = $\overline{\text{CSERR}}$ is not forwarded to PCI (default). 1 = $\overline{\text{CSERR}}$ is forwarded to PCI.
0	R/W	Parity error response enable. Bit 0 controls the response of the PCI1130 to parity errors. Bit 0 is common between each socket. Bit 0 is encoded as: 0 = Parity errors are ignored (default) 1 = Parity errors are reported using $\overline{\text{PERR}}$.

subsystem vendor ID

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
Offset: 40h
Default: 0000h
Description: The PCI1130 does not support this function. This register is read only and returns 0s when read.

subsystem ID

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
Offset: 42h
Default: 0000h
Description: The PCI1130 does not support this function. This register is read only and returns 0s when read.

PC Card 16-bit IF legacy-mode base address

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	PC Card 16-Bit IF 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 IF 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	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Register: **PC Card 16-Bit IF Legacy-Mode Base Address**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 44h
Default: 0000 0001h
Description: The PCI1130 supports the index/data scheme of accessing the ExCA registers through the use of the PC Card 16-bit IF legacy-mode base-address register. An address written to this register becomes the address for the index register and the address+1 becomes the address for the data address. Using this access method, applications requiring index/data type ExCA access can be supported.

PRODUCT PREVIEW

system control register

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	System Control Register															
Type	R	R	R	R	R	R/W	R/W	R/W	R	R	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	System Control Register															
Type	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	X	X	X	X	0	0	0	0	0	0	0	0

Register: **System Control Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 80h
Default: 0004 1X00h
Description: The system control register provides status and control for system functions unique to the PCI1130. The functionality of each bit of the system control register is defined in Table 21.

Table 21. System Control Register

BIT	TYPE	FUNCTION
31–27	R	Reserved. Bits 31–27 are read only and return 0s when read. Writes have no effect.
26	R/W	System maintenance interrupt (SMI) routing selected. This is a global bit. Bit 26 is encoded as: 0 = SMI interrupts are routed to IRQ2 (default). 1 = A card status change interrupt is generated while the SMI interrupt bit is a 1.
25	R/W	SMI interrupt status bit. Bit 25 is set to 1 when a write to either CardBus or ExCA power control for the socket and the SMI interrupt mode is enabled in bit 24. Writing a 1 to this bit clears the status bit. Bit 25 is encoded as: 0 = SMI interrupts are not active (default). 1 = SMI interrupts are active.
24	R/W	SMI interrupt mode enable. When enabled, SMI interrupts are generated when a write to the socket power control occurs. This is a global bit. Bit 24 is encoded as: 0 = SMI interrupts are disabled (default). 1 = SMI interrupts are enabled.
23–22	R	Reserved. Bits 23–22 are read only and return 0s when read. Writes have no effect.
21	R/W	V_{CC} protection enable. In the default state (0), V_{CC} protection for 16-bit PC Cards is enabled. When bit 24 is set, V_{CC} protection for 16-bit PC Cards is disabled and bad V_{CC} request for 16-bit PC Cards is also disabled. Bit 24 is encoded as follows: 0 = V_{CC} protection for 16-bit PC Cards is enabled (default). 1 = V_{CC} protection and BadV_{CC}Req for 16-bit PC Cards is disabled.
20	R/W	Reduced zoom video enable. When enabled, A25–A22 of the card interface for PC Card16 cards is placed in the high-impedance state. Bit 20 is encoded as: 0 = Reduced zoom video is disabled (default). 1 = Reduced zoom video is enabled.
19	R/W	PC/PCI DMA card enable. When enabled, allows PC Card16 cards to start requesting PC/PCI DMA bus cycles using request/grant sequence. Bit 19 is encoded as: 0 = PC/PCI DMA is disabled (default). 1 = PC/PCI DMA is enabled.
18–16	R/W	PC/PCI DMA channel assignment. The valid channels for PC/PCI DMA are: 0–3 8-bit DMA channels 4 PCI master; not used (default) 5–7 16-bit DMA channels
15–14	R	Reserved. Bits 15–14 are read only and return 0s when read. Writes have no effect.
13	R	Socket activity status bit. When set, bit 13 indicates access has been performed to or from a CardBus card or a PC Card16 card has been accessed by the PCI interface or DMA. Bit 13 is cleared upon a read of the status bit. Bit 13 is encoded as: 0 = No socket activity (default) 1 = Socket activity

system control register (continued)

Table 21. System Control Register (Continued)

BIT	TYPE	FUNCTION
12	R	Reserved. Bit 12 is read only and returns 1 when read. Writes have no effect.
11	R	Power stream in progress status bit. When high, this bit indicates that a power stream to the TPS2202A is in progress and power has been requested. Bit 11 is cleared when the power stream is finished. This is a global bit. Bit 11 is encoded as: 0 = No power stream in progress 1 = Power stream in progress
10	R	Power-down delay in progress status bit. When high, bit 10 indicates that a power-down stream is sent to the TPS2202A but power is not yet stable. Bit 10 is cleared when the power-down delay has expired. This is a global bit. Bit 10 is encoded as: 0 = Power-down delay not in effect 1 = Power-down delay in effect
9	R	Power-up delay in progress status bit. When high, bit 9 indicates that a power-up stream is sent to the TPS2202A but power is not yet stable. Bit 9 is cleared when the power-up delay has expired. This is a global bit. Bit 9 is encoded as: 0 = Power-up delay not in effect 1 = Power-up delay in effect
8	R	Interrogation in progress status. When high, bit 8 indicates an interrogation is in progress. Bit 8 is cleared when the interrogation has completed. Bit 8 is encoded as: 0 = Interrogation not in progress 1 = Interrogation in progress
7–6	R	Reserved. Bits 7–6 are read only and return 0s when read. Writes have no effect.
5	R/W	ExCA identification and revision register read only enable. When bit 5 is set, the entire ExCA identification and revision register is read only. This bit is encoded as: 0 = ExCA identification and revision register are read/write. 1 = ExCA identification and revision register are read only (default).
4	R/W	CardBus data parity SERR signaling enable bit. This is a global bit. Bit 4 is encoded as: 0 = CardBus data parity not signaled on $\overline{\text{SERR}}$ of the PCI interface (default) 1 = CardBus data parity signaled on $\overline{\text{SERR}}$ of the PCI interface
3	R/W	PC/PCI DMA enable bit. Enables PC/PCI DMA. When enabled, the PC/PCI DMA request is output on the IRQ7 pin and the PC/PCI DMA grant is input on the IRQ11 pin. This is a global bit. Bit 3 is encoded as: 0 = PC/PCI DMA disabled (default) 1 = PC/PCI DMA enabled
2	R/W	Asynchronous interrupt mode enable bit. When enabled, bit 2 allows asynchronous card status-change events to cause an interrupt without the PCI clock running. The only card status-change interrupt that requires a clock in this mode is the power status, since a clock is required to send the power stream to the TPS2202A. This is a global bit. Bit 2 is encoded as: 0 = Asynchronous interrupt mode disabled (default) 1 = Asynchronous interrupt mode enabled
1	R/W	Keep Clock. Keep PCI clock running bit. When bit 1 is set (keep clock run enabled) and PCI clock run is enabled (bit 0 is set), the PCI1130 requests that the PCI clock continue running in response to PCI clock run deassertion. If this bit is cleared, the internal status of the PCI1130 determines if the clock can be stopped. This is a global bit. Bit 1 is encoded as: 0 = Keep PCI clock running disabled (default) 1 = Keep PCI clock running enabled
0	R/W	PCI clock run enable. When enabled, bit 0 defines IRQ10/$\overline{\text{CLKRUN}}$ as the PCI clock run pin and allows the PCI1130 to support PCI CLKRUN. When bit 0 is cleared, the PCI1130 ignores the PCI CLKRUN signal. This is a global bit. Bit 0 is encoded as: 0 = PCI clock run disabled (default) 1 = PCI clock run enabled

retry status register

Bit	7	6	5	4	3	2	1	0
Name	Retry Status Register							
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: **Retry Status Register**
Type: Read/Write
Offset: 90h
Default: 00h
Description: This register displays the retry expiration status. The flags are cleared by writing a 1 to the bit. The entire register is shared between each socket. The bit definitions for this register are found in Table 22.

Table 22. Retry Status Register

BIT	TYPE	FUNCTION
7	R/W	PCI retry timeout counter enable. Bit 7 is encoded as: 0 = Disabled (default) 1 = Enabled
6	R/W	CardBus retry timeout counter enable. Bit 6 is encoded as: 0 = Disabled (default) 1 = Enabled
5	R/W	CardBus B retry expired status. Write a 1 to clear this bit. Bit 5 encoded as: 1 = Retry has expired 0 = Inactive (default) 1 = Retry has expired
4	R/W	CardBus master B retry expired status. Write a 1 to clear this bit. Bit 4 encoded as: 0 = Inactive (default) 1 = Retry has expired
3	R/W	CardBus A retry expired status. Write a 1 to clear this bit. Bit 3 encoded as: 0 = Inactive (default) 1 = Retry has expired
2	R/W	CardBus master A retry expired status. Write a 1 to clear this bit. Bit 2 encoded as: 0 = Inactive (default) 1 = Retry has expired
1	R/W	PCI retry expired status. Write a 1 to clear this bit. Bit 1 encoded as: 0 = Inactive (default) 1 = Retry has expired
0	R/W	PCI master retry expired status. Write a 1 to clear this bit. Bit 0 encoded as: 0 = Inactive (default) 1 = Retry has expired

card control register

Bit	7	6	5	4	3	2	1	0
Name	Card Control Register							
Type	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Default	0	0	0	0	0	0	0	0

Register: **Card Control Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 91h
Default: 00h
Description: This register provides separate card control for socket 0 and socket 1. Bit 7 is the only shared bit in this register, all others are specific to each socket. The bits in this register are defined in Table 23.

Table 23. Card Control Register

BIT	TYPE	FUNCTION
7	R/W	Ring indicate output enable. Bit 7 configures the IRQ15/RI_OUT pin as RI_OUT, which is the output for the $\overline{RI}$ input. Bit 7 is encoded as: 0 = Disabled (default) 1 = Enabled
6	R/W	Zoom video mode enable. Bit 6 enables the zoom video mode application. Bit 6 is encoded as: 0 = Disabled (default) 1 = Enabled
5	R/W	PCI interrupt enable. Bit 5 enables the PCI interrupt INTA (INTB). Bit 5 is encoded as: 0 = Disabled (default) 1 = Enabled
4	R/W	Functional interrupt routing enable. If bit 5 is enabled, bit 4 routes the IREQ/$\overline{CINT}$ from card A (B) to the PCI interrupt INTA (INTB). Bit 4 is encoded as: 0 = Disabled (default) 1 = Enabled
3	R/W	Card status change (CSC) interrupt routing enable. If bit 5 is enabled, bit 3 routes the CSC interrupts to the PCI interrupt INTA (INTB). Bit 3 is encoded as: 0 = Disabled (default) 1 = Enabled
2	R	Reserved. Bit 2 is read only and returns 0s when read. Writes have no effect.
1	R/W	SpeakerOut/suspend enable. When set, bit 1 enables $\overline{SPKR}$ on the PC Card and routes it to $\overline{SPKROUT}$ on the PCIBus. When cleared, this bit enables the suspend mode for the PCI1130, see *power management* for details concerning PCI1130 suspend mode. Bit 1 is encoded as: 0 = SUSPEND mode enabled (default) 1 = $\overline{SPKR}$ to $\overline{SPKROUT}$ enabled
0	R/W	IFG. Bit 0 is the PC Card interrupt flag. Write a 1 to clear this bit. Bit 0 is encoded as: 0 = No PC Card interrupt (default) 1 = PC Card interrupt detected

PRODUCT PREVIEW

device control register

Bit	7	6	5	4	3	2	1	0
Name	Device Control Register							
Type	R	R/W	R/W	R/W	R/W	R/W	R/W	R
Default	0	1	1	1	0	0	0	0

Register: **Device Control Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 92h
Default: 70h
Description: This register is common for socket A and socket B and can be accessed from both configuration spaces. The bit definitions for this register are found in Table 24.

NOTE:
When bit 5 is set, the PCI1130 cannot program a dual-voltage socket to 5 V.

Table 24. Device Control Register

BIT	TYPE	FUNCTION
7	R	Reserved. Bit 7 is read only and returns 0s when read. Only write a value of 0b to bit 7.
6	R/W	5-V socket capable force bit. Bit 6 is read/write. Bit 6 is encoded as: 0 = Not 5-V capable 1 = 5-V capable (default)
5	R/W	3-V socket capable force bit. Bit 5 is read/write. Bit 5 is encoded as: 0 = Not 3-V capable 1 = 3-V capable (default)
4	R/W	Reserved. Bit 4 defaults to a 1. Only write 1s to bit 4.
3	R/W	Reserved. For internal TI test purposes only; bit 3 must always write a 0.
2–1	R/W	Interrupt mode. Bits 2–1 select the interrupt mode used by the PCI1130. Bits 2–1 are encoded as: 00 = No interrupts enabled (default) 01 = ISA 10 = Compaq type interrupt scheme 11 = Reserved
0	R	Reserved. For internal TI test purposes only.

PRODUCT PREVIEW

buffer control register

Bit	7	6	5	4	3	2	1	0
Name	Buffer Control Register							
Type	R	R	R	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0

Register: **Buffer Control Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: 93h
Default: 00h
Description: This register is common for socket A and socket B and can be accessed from both configuration spaces. It allows diagnostic software to control the depth of each FIFO. The bit definitions for this register are found in Table 25.

Table 25. Buffer Control Register

BIT	TYPE	FUNCTION
7–5	R	Reserved. Bits 7–5 are read only and return 0s when read. Writes have no effect.
4	R/W	Reserved. Bit 4 is for internal TI use only. Host software must always write 0 to this bit. CAUTION: Unpredictable behavior can result from setting this bit to 1.
3	R/W	CardBus read buffer depth. Bit 3 is encoded as: 0 = Full depth (default) 1 = 1 deep
2	R/W	CardBus write buffer depth. Bit 2 is encoded as: 0 = Full depth (default) 1 = 1 deep
1	R/W	PCI read buffer depth. Bit 1 is encoded as: 0 = Full depth (default) 1 = 1 deep
0	R/W	PCI write buffer depth. Bit 0 is encoded as: 0 = Full depth (default) 1 = 1 deep

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
Size: Four bytes
Description: This register provides control over the PC Card DMA signaling. Table 26 describes each bit in this register.

Table 26. Socket DMA Register 0

BIT	TYPE	FUNCTION
31–2	R	Reserved. Bits 31–2 are read only and return 0s when read. Only write 0s to these bits.
1–0	R/W	DMA enable/$\overline{\text{DREQ}}$ pin. Bits 1–0 indicate which pin on the PC Card interface acts as the $\overline{\text{DREQ}}$ (DMA request) signal during DMA transfers. This field is encoded as: 00 = Socket not configured for DMA (default) 01 = $\overline{\text{DREQ}}$ uses $\overline{\text{SPKR}}$ 10 = $\overline{\text{DREQ}}$ uses $\overline{\text{IOIS16}}$ 11 = $\overline{\text{DREQ}}$ uses INPACK

PRODUCT PREVIEW

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
Size: Four bytes
Description: This register provides control over the DMA registers and the PCI portion of DMA transfers. Table 27 describes each bit in this register. The DMA base address locates the DMA registers in a 16-byte region within the first 64K bytes of PCI I/O address space. Note that 32-bit transfers are not supported; the maximum transfer width possible for a 16-bit PC Card is 16 bits.

Table 27. Socket DMA Register 1

BIT	TYPE	FUNCTION
31–16	R	Reserved. Bits 31–16 are read only and return 0s when read. Writes have no effect.
15–4	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 four bits are hardwired to 0 and are included in the address decode, forcing the window to a natural 16-byte boundary.
3	R	Nonlegacy extended addressing. This is not supported on the PCI1130 and always returns a 0.
2–1	R/W	Transfer size. Bits 2–1 specify the width of the DMA transfer on the PC Card interface. The field is encoded as: 00 = 8-bit transfer (default) 01 = 16-bit transfer 10 = Reserved 11 = Reserved
0	R/W	Decode enable. Enables the decoding of the DMA base address by the PCI1130. Bit 0 is encoded as: 0 = Disabled (default) 1 = Enabled

PRODUCT PREVIEW

ExCA registers

ExCA architecture registers implemented in the PCI1130 are register compatible with the Intel 82365SL-DF PCMCIA controller. The PCI1130 makes the ExCA registers for each socket available by directly mapping them into PCI memory space. They are located through the CardBus socket registers/ExCA registers base address register at offset 800h. Each socket has a separate CardBus socket register/ExCA registers base address register for accessing the ExCA registers (see Figure 11). The ExCA offset is the offset from the PC Card 16-bit IF legacy-mode base address. This PC Card 16-bit legacy-mode base address is shared by both sockets. The ExCA registers run contiguously from offset 00h–3Fh for socket A and 40h–7Fh for socket B (see Figure 12). Table 3 identifies each ExCA register and its respective ExCA offset and PCI configuration header address.

The ExCA general setup registers (defined in the Intel 82365SL-DF specification) provide status and control information on a variety of 16-bit PC Card functions. These registers are concerned with V_{CC}/V_{PP} control, PC Card status, memory and I/O window control, and global card status. This set of registers includes those registers at offsets 800h, 801h, 802h, 804h, 806h, 816h, 81Eh and 840h.

The interrupt registers in the ExCA register set (defined in the Intel 82365SL-DF specification) control such card functions as reset, type, interrupt routing, and interrupt enables. Special attention must be paid to the interrupt routing registers and the host interrupt signaling method selected for the PCI1130. Certain IRQs are available only if the serial interrupt scheme is selected. This scheme is a method by which IRQ information is communicated serially to the host interrupt controller through a common, wired-OR terminal on the PCI1130. If discrete IRQ signaling is selected, only a subset of the possible IRQs are available for interrupt routing. Host software should first select the interrupt signaling method to be used, then route the PC Card interrupt sources to host interrupts. This set of registers includes those registers at ExCA offsets 803h and 805h.

The 16-bit I/O PC Cards are available to the host system via 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.

The 16-bit memory PC Cards are available to the host system via memory windows. These are regions of host memory address 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. Memory windows have 4K-byte granularity.

ExCA registers (continued)

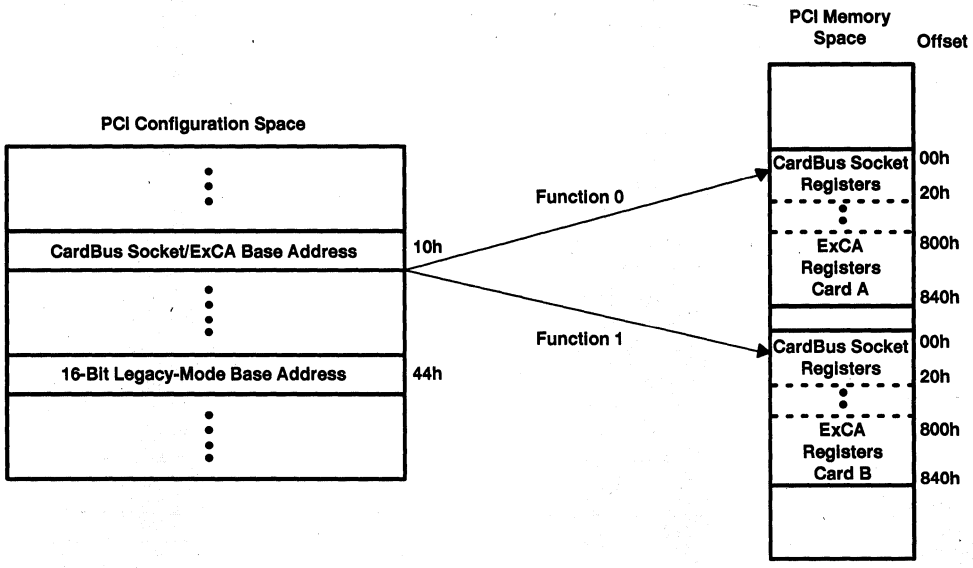


Figure 11. ExCA PCI Memory Access Method

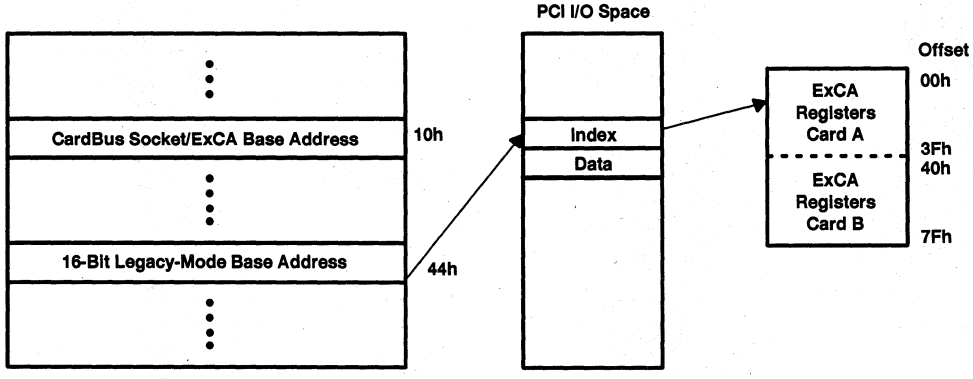


Figure 12. ExCA PCI I/O Legacy Access Method

PRODUCT PREVIEW

ExCA registers (continued)

Table 28. ExCA Registers

NAME	PCI MEMORY ADDRESS OFFSET	ExCA OFFSET	
		CARD A	CARD B
Identification and revision register	800	00	40
Interface status register	801	01	41
Power control register	802	02	42
Interrupt and general control register	803	03	43
Card status-change register	804	04	44
Card status-change interrupt configuration register	805	05	45
Address window enable register	806	06	46
I/O window-control register	807	07	47
I/O window 0 start-address low-byte register	808	08	48
I/O window 0 start-address high-byte register	809	09	49
I/O window 0 end-address low-byte register	80A	0A	4A
I/O window 0 end-address high-byte register	80B	0B	4B
I/O window 1 start-address low-byte register	80C	0C	4C
I/O window 1 start-address high-byte register	80D	0D	4D
I/O window 1 end-address low-byte register	80E	0E	4E
I/O window 1 end-address high-byte register	80F	0F	4F
Memory window 0 start-address low-byte register	810	10	50
Memory window 0 start-address high-byte register	811	11	51
Memory window 0 end-address low-byte register	812	12	52
Memory window 0 end-address high-byte register	813	13	53
Memory window 0 offset-address low-byte register	814	14	54
Memory window 0 offset-address high-byte register	815	15	55
Card detect and general control register	816	16	56
Reserved	817	17	57
Memory window 1 start-address low-byte register	818	18	58
Memory window 1 start-address high-byte register	819	19	59
Memory window 1 end-address low-byte register	81A	1A	5A
Memory window 1 end-address high-byte register	81B	1B	5B
Memory window 1 offset-address low-byte register	81C	1C	5C
Memory window 1 offset-address high-byte register	81D	1D	5D
Global control register	81E	1E	5E
Reserved	81F	1F	5F
Memory window 2 start-address low-byte register	820	20	60
Memory window 2 start-address high-byte register	821	21	61
Memory window 2 end-address low-byte register	822	22	62
Memory window 2 end-address high-byte register	823	23	63
Memory window 2 offset-address low-byte register	824	24	64
Memory window 2 offset-address high-byte register	825	25	65
Reserved	826	26	66
Reserved	827	27	67
Memory window 3 start-address low-byte register	828	28	68
Memory window 3 start-address high-byte register	829	29	69
Memory window 3 end-address low-byte register	82A	2A	6A
Memory window 3 end-address high-byte register	82B	2B	6B
Memory window 3 offset-address low-byte register	82C	2C	6C
Memory window 3 offset-address high-byte register	82D	2D	6D

ExCA registers (continued)

Table 28. ExCA Registers (Continued)

NAME	PCI MEMORY ADDRESS OFFSET	CARD A	CARD B
Reserved	82E	2E	6E
Reserved	82F	2F	6F
Memory window 4 start-address low-byte register	830	30	70
Memory window 4 start-address high-byte register	831	31	71
Memory window 4 end-address low-byte register	832	32	72
Memory window 4 end-address high-byte register	833	33	73
Memory window 4 offset-address low-byte register	834	34	74
Memory window 4 offset-address high-byte register	835	35	75
I/O window 0 offset-address low-byte register	836	36	76
I/O window 0 offset-address high-byte register	837	37	77
I/O window 1 offset-address low-byte register	838	38	78
I/O window 1 offset-address high-byte register	839	39	79
Reserved	83A	3A	7A
Reserved	83B	3B	7B
Reserved	83C	—	—
Reserved	83D	3D	7D
Reserved	83E	3E	7E
Reserved	83F	3F	7F
Memory window page register	840	3C	7C

ExCA identification and revision register (index 00h)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Identification and Revision Register							
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 Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 800h; Card A ExCA Offset 00h
Card B ExCA Offset 40h
Default: 84h
Description: This register provides host software with information on 16-bit PC Card support and Intel 82365SL-DF compatibility. Table 29 describes each bit in the ExCA identification and revision register.

NOTE:
This entire register is read only when bit 5 of the system control register is set (see Table 21).

Table 29. ExCA Identification and Revision Register (Index 00h)

BIT	TYPE	FUNCTION
7–6	R	Interface type. Bits 7–6, which are hardwired as 10b, identify the 16-bit PC Card support provided by the PCI1130. The PCI1130 supports both I/O and memory 16-bit PC Cards.
5–4	R/W	Reserved. Bits 5–4 can be used for Intel 82365SL-DF emulation.
3–0	R/W	Intel 82365SL-DF revision. Bits 3–0 store the Intel 82365SL-DF revision supported by the PCI1130. Host software can read this field to determine compatibility to the Intel 82365SL-DF register set. This field defaults to 0100b upon PCI1130 reset.

PRODUCT PREVIEW

ExCA interface status register (Index 01h)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Interface Status Register							
Type	R	R	R	R	R	R	R	R
Default	0	0	X	X	X	X	X	X

Register: **ExCA Interface Status Register**
Type: Read Only
Offset: CardBus Socket Address + 801h; Card A ExCA Offset 01h
Card B ExCA Offset 41h
Default: 00XX XXXXb (See Table 30 for detailed default information for bits 5–0; "X" indicates that value of the bit after reset depends on the state of the PC Card interface.)
Description: This register provides information on the current status of the PC Card interface. Table 30 describes each bit in the ExCA interface status register.

Table 30. ExCA Interface Status Register (Index 01h)

BIT	TYPE	FUNCTION
7	R	Reserved. Bit 7 is read only and returns 0s when read. Writes have no effect.
6	R	Card power. Bit 6 indicates the current power status of the PC Card socket. Bit 6 reflects how the ExCA power control register has been programmed. Bit 6 is encoded as: 0 = V_{CC} and V_{PP} to the socket is turned off (default). 1 = V_{CC} and V_{PP} to the socket is turned on.
5	R	READY. Bit 5 indicates the current status of the READY signal at the PC Card interface. This signal reports to the PCI1130 that the card is ready for another data transfer. Bit 5 is encoded as: 0 = PC Card is not ready for a data transfer. 1 = PC Card is ready for a data transfer.
4	R	Card write protect. Bit 4 indicates the current status of the WP signal at the PC Card interface. This signal reports to the PCI1130 whether or not the memory card is write protected. Further, write protection for an entire PCI1130 16-bit memory window is available by setting the appropriate bit in the memory window offset high-byte register. Bit 4 is encoded as: 0 = WP signal is 0. PC Card is R/W. 1 = WP signal is 1. PC Card is read only.
3	R	Card detect 2. Bit 3 indicates the current status of the $\overline{CD2}$ signal at the PC Card interface and does not have a default value. Host software can use this bit and the card detect 1 ($\overline{CD1}$) bit to determine if a PC Card is present in the socket and is fully seated. Bit 3 is encoded as: 0 = $\overline{CD2}$ signal is 1. No PC Card is inserted. 1 = $\overline{CD2}$ signal is 0. PC Card is inserted.
2	R	Card detect 1. Bit 2 indicates the current status of the $\overline{CD1}$ signal at the PC Card interface and does not have a default value. Host software can use bit 2 and the card detect 2 ($\overline{CD2}$) bit to determine if a PC Card is present in the socket and is fully seated. Bit 2 is encoded as: 0 = $\overline{CD1}$ signal is 1. No PC Card is inserted. 1 = $\overline{CD1}$ signal is 0. PC Card is inserted.
1–0	R	Battery voltage detect. Bits 1–0 have meanings that depend on the type of 16-bit PC Card inserted in the socket. 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 BVD1 status and bit 0 reflects the BVD2 status. In this case, bits 1–0 are encoded as: 00 = Battery is dead. 01 = Battery is dead. 10 = Battery is low; warning. 11 = Battery is good. When a 16-bit I/O card is inserted, this field indicates the status of $\overline{SPKR}$ (bit 1) and STSCHG (bit 0) at the PC Card interface. In this case, bits 1–0 directly reflect the current state of these card outputs.

PRODUCT PREVIEW

ExCA power control register (Index 02h)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Power Control Register							
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 Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 802h; Card A ExCA Offset 02h
Card B ExCA Offset 42h

Default: 00h
Description: This register provides PC Card power control. Bit 7 of this register controls the 16-bit outputs on the socket interface. Table 31 describes each bit in the ExCA power control register.

Table 31. ExCA Power Control Register (Index 02h)

BIT	TYPE	FUNCTION
7	R/W	Card outputs enable. Bit 7 controls the state of all 16-bit outputs on the PCI1130. Bit 7 is encoded as: 0 = 16-bit PC Card outputs are disabled (default). 1 = 16-bit PC Card outputs are enabled.
6–5	R	Reserved. Bits 6–5 are read only and return 0s when read. Writes have no effect.
4–3	R/W	V_{CC}. Bits 4–3 are used to request changes to card V_{CC}. This field is encoded as: 00 = 0 V (default) 01 = 0 V (reserved) 10 = 5 V 11 = 3 V
2	R	Reserved. Bit 2 is read only and returns 0s when read. Writes have no effect.
1–0	R/W	V_{PP}. Bits 1–0 set the V_{PP} level applied to the socket. Changes to this socket are relayed to the TPS2202 power switch. The PCI1130 ignores this field unless V_{CC} to the socket is enabled (i.e., 5 V or 3.3 V). This field is encoded as: 00 = 0 V (default) 01 = V_{CC} 10 = 12 V 11 = 0 V (reserved)

PRODUCT PREVIEW

ExCA interrupt and general control register (index 03h)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Interrupt and General Control Register							
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 Register**
Type: Read/Write
Offset: CardBus Socket Address + 803h; Card A ExCA Offset 03h
Card B ExCA Offset 43h
Default: 00h
Description: This register controls interrupt routing for I/O interrupts, as well as PC Card resets and card types. Table 32 describes each bit in the ExCA interrupt and general control register.

Table 32. ExCA Interrupt and General Control Register (Index 03h)

BIT	TYPE	FUNCTION
7	R/W	Card ring indicate enable. Bit 7 enables ring indicate function of the BVD1/$\overline{RI}$ pins. Bit 7 is encoded as: 0 = Ring indicate is disabled (default). 1 = Ring indicate is enabled.
6	R/W	Card reset. Bit 6 controls the PC Card RESET signal and allows host software to force a card reset. Bit 6 affects 16-bit cards only. Bit 6 is encoded as: 0 = RESET signal is asserted (default). 1 = RESET signal is deasserted.
5	R/W	Card type. Bit 5 indicates the PC Card type. Bit 5 is encoded as: 0 = Memory PC Card is installed (default). 1 = I/O PC Card is installed.
4	R/W	PCI interrupt-CSC routing enable bit. When bit 4 is set high and the PCI interrupt bit in the device control register (see *device control register*) is enabled, the card status change interrupts are routed to the PCI interrupt for the socket (INTA or INTB). When low, the card status change interrupts are routed using bits 7–4 in the ExCA card status change interrupt configuration register (see *ExCA card status change interrupt configuration register*). In order to use PCI interrupt-CSC routing, the ISA IRQ signaling method must be enabled (bits 2–1 of the device control register, offset 92h must not be 0). Bit 4 is encoded as: 0 = CSC interrupts routed by ExCA registers (default) 1 = CSC interrupts routed to PCI interrupts
3–0	R/W	Card interrupt select for 16-bit I/O PC Card interrupts. Bits 3–0 select the interrupt routing for I/O PC Card interrupts. This field is encoded as: 0000 = No interrupt routing (default) 0001 = IRQ1 enabled† 0010 = SMI enabled † 0011 = IRQ3 enabled 0100 = IRQ4 enabled 0101 = IRQ5 enabled 0110 = IRQ6 enabled† 0111 = IRQ7 enabled 1000 = IRQ8 enabled† 1001 = IRQ9 enabled 1010 = IRQ10 enabled 1011 = IRQ11 enabled 1100 = IRQ12 enabled 1101 = IRQ13 enabled† 1110 = IRQ14 enabled 1111 = IRQ15 enabled

† Valid when the serialized interrupt scheme is selected in the TI extension registers. There is no dedicated pin for these interrupts.

PRODUCT PREVIEW

ExCA card status-change register (index 04)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Card Status-Change Register							
Type	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0

Register: **ExCA Card Status-Change Register**
Type: Read Only
Offset: CardBus Socket Address + 804h; Card A ExCA Offset 04h
Card B ExCA Offset 44h

Default: 00h
Description: This register reflects the status of PC Card interrupt sources. The ExCA card status-change interrupt configuration 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 as 0. When an interrupt source is enabled, the corresponding bit in this register is set to indicate that the interrupt source is active. After generating the interrupt to the host, the interrupt service routine must read this register to determine the source of the interrupt. The interrupt service routine is also responsible for resetting the bits in this register.

Resetting the bit is accomplished by one of two methods. The choice of these two methods is based on the interrupt flag clear mode select, bit 2 in the ExCA global control register (see *ExCA global control register*). When this interrupt flag clear mode select bit is set, the bits in the ExCA card status-change register are reset by writing a 1 to the respective bit locations. When the interrupt flag clear mode select bit is cleared (0), the bits in the ExCA card status-change register are reset by a read cycle to the register. Table 33 describes each bit in the ExCA card status-change register.

Table 33. ExCA Card Status-Change Register (Index 04h)

BIT	TYPE	FUNCTION
7–4	R	Reserved. Bits 7–4 are read only and return 0s when read. Writes have no effect.
3	R	Card detect change. Bit 3 indicates whether a change on the $\overline{CD1}$ or $\overline{CD2}$ signals occurred at the PC Card interface. Bit 3 is encoded as: 0 = No change detected on either $\overline{CD1}$ or $\overline{CD2}$ (default) 1 = A change was detected on either $\overline{CD1}$ or $\overline{CD2}$
2	R	Ready change. When a 16-bit memory card is installed in the socket, bit 2 indicates whether the source of a PCI1130 interrupt was due to a change on the READY signal at the PC Card interface, indicating that a PC Card is now ready to accept new data. Bit 2 is encoded as: 0 = No low-to-high transition detected on READY (default) 1 = Detected a low-to-high transition on READY When a 16-bit I/O card is installed, this bit is always 0.
1	R	Battery warning change. When a 16-bit memory card is installed in the socket, bit 1 indicates whether the source of a PCI1130 interrupt was due to a battery low warning condition. Bit 1 is encoded as: 0 = No battery warning condition (default) 1 = Detected a battery warning condition When a 16-bit I/O card is installed, this bit is always 0.
0	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 PCI1130 interrupt is due to a battery dead condition. Bit 0 is encoded as: 0 = No battery dead condition (default) 1 = Detected a battery dead condition When a 16-bit I/O card is installed, this bit indicates whether the source of a PCI1130 interrupt is due to the assertion of the $\overline{STSCHG}$ signal at the PC Card interface. Bit 0 is encoded as: 0 = $\overline{STSCHG}$ deasserted (default) 1 = $\overline{STSCHG}$ asserted Ring indicate. When the PCI1130 is configured for ring indicate operation (see *ring indicate*), bit 0 indicates the status of the $\overline{RI}$ pin.

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

Bit	7	6	5	4	3	2	1	0
Name	ExCA Card 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 Register**
Type: Read/Write
Offset: CardBus Socket Address + 805h; Card A ExCA Offset 05h
Card B ExCA Offset 45h
Default: 00h
Description: This register controls interrupt routing for card status change interrupts, as well as masking PC Card interrupt sources. Table 34 describes each bit in the ExCA card status-change interrupt configuration register.

Table 34. ExCA Card Status-Change Interrupt Configuration Register (Index 05h)

BIT	TYPE	FUNCTION
7–4	R/W	Interrupt select for card status change. Bits 7–4 select the interrupt routing for card status-change interrupts. This field is encoded as: 0000 = No interrupt routing (default) 0001 = IRQ1 enabled† 0010 = SMI enabled† 0011 = IRQ3 enabled 0100 = IRQ4 enabled 0101 = IRQ5 enabled 0110 = IRQ6 enabled† 0111 = IRQ7 enabled 1000 = IRQ8 enabled† 1001 = IRQ9 enabled 1010 = IRQ10 enabled 1011 = IRQ11 enabled 1100 = IRQ12 enabled 1101 = IRQ13 enabled† 1110 = IRQ14 enabled 1111 = IRQ15 enabled
3	R/W	Card detect enable. Enables interrupts on $\overline{CD1}$ or $\overline{CD2}$ changes. Bit 3 is encoded as: 0 = Disables interrupts on changes on the $\overline{CD1}$ or $\overline{CD2}$ lines (default) 1 = Enables interrupts on changes on the $\overline{CD1}$ or $\overline{CD2}$ lines
2	R/W	Ready enable. Bit 2 enables/disables a low-to-high transition on the PC Card READY signal to generate a host interrupt. This interrupt source is considered a card status change. Bit 2 is encoded as: 0 = Disables host interrupt generation (default) 1 = Enables host interrupt generation
1	R/W	Battery warning enable. Bit 1 enables/disables a battery warning condition to generate a host interrupt. This interrupt source is considered a card status change. Bit 1 is encoded as: 0 = Disables host interrupt generation (default) 1 = Enables host interrupt generation
0	R/W	Battery dead enable. Bit 0 enables/disables a battery dead condition on a memory PC Card or assertion of the $\overline{STSCHG}$ I/O PC Card signal to generate a host interrupt. This interrupt source is considered a card status change. Bit 0 is encoded as: 0 = Disables host interrupt generation (default) 1 = Enables host interrupt generation

† Valid when the serialized interrupt scheme is selected in the TI extension registers. There is no dedicated pin for these interrupts.

PRODUCT PREVIEW

ExCA address window enable register (index 06h)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Address Window Enable Register							
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 Register**
Type: Read Only, Read/Write
Offset: CardBus Socket Address + 806h; Card A ExCA Offset 06h
Card B ExCA Offset 46h
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 PCI1130 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. Table 35 describes each bit in the ExCA address window enable register.

Table 35. ExCA Address Window Enable Register (Index 06h)

BIT	TYPE	FUNCTION
7	R/W	I/O window 1 enable. Bit 7 enables/disables I/O window 1 for the PC Card. Bit 7 is encoded as: 0 = I/O window 1 disabled (default) 1 = I/O window 1 enabled
6	R/W	I/O window 0 enable. Bit 6 enables/disables I/O window 0 for the PC Card. Bit 6 is encoded as: 0 = I/O window 0 disabled (default) 1 = I/O window 0 enabled
5	R	Reserved. Bit 5 is read only and returns 0 when read. Writes have no effect.
4	R/W	Memory window 4 enable. Bit 4 enables/disables memory window 4 for the PC Card. Bit 4 is encoded as: 0 = Memory window 4 disabled (default) 1 = Memory window 4 enabled
3	R/W	Memory window 3 enable. Bit 3 enables/disables memory window 3 for the PC Card. Bit 3 is encoded as: 0 = Memory window 3 disabled (default) 1 = Memory window 3 enabled
2	R/W	Memory window 2 enable. Bit 2 enables/disables memory window 2 for the PC Card. Bit 2 is encoded as: 0 = Memory window 2 disabled (default) 1 = Memory window 2 enabled
1	R/W	Memory window 1 enable. Bit 1 enables/disables memory window 1 for the PC Card. Bit 1 is encoded as: 0 = Memory window 1 disabled (default) 1 = Memory window 1 enabled
0	R/W	Memory window 0 enable. Bit 0 enables/disables memory window 0 for the PC Card. Bit 0 is encoded as: 0 = Memory window 0 disabled (default) 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 Register							
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 Register**
Type: Read/Write
Offset: CardBus Socket Address + 807h; Card A ExCA Offset 07h
Card B ExCA Offset 47h
Default: 00h
Description: The I/O window control register contains parameters related to I/O window sizing and cycle timing. Table 36 describes each bit in this register.

Table 36. ExCA I/O Window Control Register (Index 07h)

BIT	TYPE	FUNCTION
7	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. Bit 7 is encoded as: 0 = 16-bit cycles have standard length (default). 1 = 16-bit cycles are extended by one equivalent ISA wait state.
6	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. Bit 6 is encoded as: 0 = 8-bit cycles have standard length (default). 1 = 8-bit cycles are reduced to equivalent of three ISA cycles.
5	R/W	I/O window 1 $\overline{\text{IOIS16}}$ source. Bit 5 controls the I/O window 1 automatic data sizing feature that uses the $\overline{\text{IOIS16}}$ signal from the PC Card to determine the data width of the I/O data transfer. Bit 5 is encoded as: 0 = Window data width is determined by I/O window 1 data sizing bit, bit 4 (default). 1 = Window data width is determined by $\overline{\text{IOIS16}}$.
4	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. Bit 4 is encoded as: 0 = Window data width is 8 bits (default). 1 = Window data width is 16 bits.
3	R/W	I/O window zero 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. Bit 3 is encoded as: 0 = 16-bit cycles have standard length (default). 1 = 16-bit cycles are extended by one equivalent ISA wait state.
2	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. Bit 2 is encoded as: 0 = 8-bit cycles have standard length (default). 1 = 8-bit cycles are reduced to equivalent of three ISA cycles.
1	R/W	I/O window 0 $\overline{\text{IOIS16}}$ source. Bit 1 controls the I/O window 0 automatic data sizing feature that uses the $\overline{\text{IOIS16}}$ signal from the PC Card to determine the data width of the I/O data transfer. Bit 1 is encoded as: 0 = Window data width is determined by I/O window 0 data sizing bit, bit 0 (default). 1 = Window data width is determined by $\overline{\text{IOIS16}}$.
0	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. Bit 0 is encoded as: 0 = Window data width is 8 bits (default). 1 = Window data width is 16 bits.

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

Register: **ExCA I/O Window 0 Start-Address Low-Byte Register**
Offset: CardBus Socket Address + 808h; Card A ExCA Offset 08h
Card B ExCA Offset 48h
Register: **ExCA I/O Window 1 Start-Address Low-Byte Register**
Offset: CardBus Socket Address + 80Ch; Card A ExCA Offset 0Ch
Card B ExCA Offset 4Ch
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 8 bits of these registers correspond to the lower 8 bits of the start address.

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

Register: **ExCA I/O Window 0 Start-Address High-Byte Register**
Offset: CardBus Socket Address + 809h; Card A ExCA Offset 09h
Card B ExCA Offset 49h
Register: **ExCA I/O Window 1 Start-Address High-Byte Register**
Offset: CardBus Socket Address + 80Dh; Card A ExCA Offset 0Dh
Card B ExCA Offset 4Dh
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 8 bits of these registers correspond to the upper 8 bits of the start address.

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

Register: **ExCA I/O Window 0 End-Address Low-Byte Register**
Offset: CardBus Socket Address + 80Ah; Card A ExCA Offset 0Ah
Card B ExCA Offset 4Ah
Register: **ExCA I/O Window 1 End-Address Low-Byte Register**
Offset: CardBus Socket Address + 80Eh; Card A ExCA Offset 0Eh
Card B ExCA Offset 4Eh
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 8 bits of these registers correspond to the lower 8 bits of the end address.

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

Register: **ExCA I/O Window 0 End-Address High-Byte Register**
Offset: CardBus Socket Address + 80Bh; Card A ExCA Offset 0Bh
Card B ExCA Offset 4Bh
Register: **ExCA I/O Window 1 End-Address High-Byte Register**
Offset: CardBus Socket Address + 80Fh; Card A ExCA Offset 0Fh
Card B ExCA Offset 4Fh
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 8 bits of these registers correspond to the upper 8 bits of the end address.

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

Register: **ExCA Memory Window 0 Start-Address Low-Byte Register**
Offset: CardBus Socket Address + 810h; Card A ExCA Offset 10h
Card B ExCA Offset 50h
Register: **ExCA Memory Window 1 Start-Address Low-Byte Register**
Offset: CardBus Socket Address + 818h; Card A ExCA Offset 18h
Card B ExCA Offset 58h
Register: **ExCA Memory Window 2 Start-Address Low-Byte Register**
Offset: CardBus Socket Address + 820h; Card A ExCA Offset 20h
Card B ExCA Offset 60h
Register: **ExCA Memory Window 3 Start-Address Low-Byte Register**
Offset: CardBus Socket Address + 828h; Card A ExCA Offset 28h
Card B ExCA Offset 68h
Register: **ExCA Memory Window 4 Start-Address Low-Byte Register**
Offset: CardBus Socket Address + 830h; Card A ExCA Offset 30h
Card B ExCA Offset 70h
Type: Read/Write
Default: 00h
Size: One byte
Description: These registers contain the low byte of the memory window start address for I/O windows 0, 1, 2, 3, and 4. The 8 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)

Register: **ExCA Memory Window 0 Start-Address High-Byte Register**
Offset: CardBus Socket Address + 811h; Card A ExCA Offset 11h
Card B ExCA Offset 51h
Register: **ExCA Memory Window 1 Start-Address High-Byte Register**
Offset: CardBus Socket Address + 819h; Card A ExCA Offset 19h
Card B ExCA Offset 59h
Register: **ExCA Memory Window 2 Start-Address High-Byte Register**
Offset: CardBus Socket Address + 821h; Card A ExCA Offset 21h
Card B ExCA Offset 61h
Register: **ExCA Memory Window 3 Start-Address High-Byte Register**
Offset: CardBus Socket Address + 829h; Card A ExCA Offset 29h
Card B ExCA Offset 69h
Register: **ExCA Memory Window 4 Start-Address High-Byte Register**
Offset: CardBus Socket Address + 831h; Card A ExCA Offset 31h
Card B ExCA Offset 71h
Type: Read/Write
Default: 00h
Size: One byte
Description: These registers contain the high byte of the memory window start address for memory windows 0, 1, 2, 3 and 4. In addition, the memory window data width and wait states are set in this register. Table 37 describes each bit in the ExCA memory window start-address high-byte register.

Table 37. ExCA Memory Window Start-Address High-Byte Register

BIT	TYPE	FUNCTION
7	R/W	Data size. Bit 7 controls the memory window data width. Bit 7 is encoded as: 0 = Window data width is 8 bits (default). 1 = Window data width is 16 bits.
6	R/W	0 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. Bit 6 is encoded as: 0 = 8- and 16-bit cycles have standard length (default). 1 = 8-bit cycles are reduced to equivalent of three ISA cycles. 16-bit cycles are reduced to equivalent of two ISA cycles.
5–4	R/W	Scratch pad bits. Bits 5–4 are read/write and have no effect on memory window operation.
3–0	R/W	Start-address high-byte. Bits 3–0 represent the upper address bits A23–A20 of the memory window start address.

PRODUCT PREVIEW

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

Register: **ExCA Memory Window 0 End-Address Low-Byte Register**
Offset: CardBus Socket Address + 812h; Card A ExCA Offset 12h
Card B ExCA Offset 52h
Register: **ExCA Memory Window 1 End-Address Low-Byte Register**
Offset: CardBus Socket Address + 81Ah; Card A ExCA Offset 1Ah
Card B ExCA Offset 5Ah
Register: **ExCA Memory Window 2 End-Address Low-Byte Register**
Offset: CardBus Socket Address + 822h; Card A ExCA Offset 22h
Card B ExCA Offset 62h
Register: **ExCA Memory Window 3 End-Address Low-Byte Register**
Offset: CardBus Socket Address + 82Ah; Card A ExCA Offset 2Ah
Card B ExCA Offset 6Ah
Register: **ExCA Memory Window 4 End-Address Low-Byte Register**
Offset: CardBus Socket Address + 832h; Card A ExCA Offset 32h
Card B ExCA Offset 72h

Type: Read/Write
Default: 00h
Size: One byte
Description: These registers contain the low byte of the memory window end address for memory windows 0, 1, 2, 3 and 4. The 8 bits of these registers correspond to bits A19–A12 of the end address.

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

Register: **ExCA Memory Window 0 End-Address High-Byte Register**
Offset: CardBus Socket Address + 813h; Card A ExCA Offset 13h
Card B ExCA Offset 53h
Register: **ExCA Memory Window 1 End-Address High-Byte Register**
Offset: CardBus Socket Address + 81Bh; Card A ExCA Offset 1Bh
Card B ExCA Offset 5Bh
Register: **ExCA Memory Window 2 End-Address High-Byte Register**
Offset: CardBus Socket Address + 823h; Card A ExCA Offset 23h
Card B ExCA Offset 63h
Register: **ExCA Memory Window 3 End-Address High-Byte Register**
Offset: CardBus Socket Address + 82Bh; Card A ExCA Offset 2Bh
Card B ExCA Offset 6Bh
Register: **ExCA Memory Window 4 End-Address High-Byte Register**
Offset: CardBus Socket Address + 833h; Card A ExCA Offset 33h
Card B ExCA Offset 73h

Type: Read Only, Read/Write
Default: 00h
Size: One byte
Description: These registers contain the high byte of the memory window end address for memory windows 0, 1, 2, 3 and 4. In addition, the memory window wait states are set in this register. Table 38 describes each bit in the ExCA memory window end-address high-byte register.

Table 38. ExCA Memory Window End-Address High-Byte Register

BIT	TYPE	FUNCTION
7–6	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	R	Reserved. Bits 5–4 are read only and return 0s when read. Writes have no effect.
3–0	R/W	End-address high-byte. 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)

Register: **ExCA Memory Window 0 Offset-Address Low-Byte Register**
Offset: CardBus Socket Address + 814h; Card A ExCA Offset 14h
Card B ExCA Offset 54h
Register: **ExCA Memory Window 1 Offset-Address Low-Byte Register**
Offset: CardBus Socket Address + 81Ch; Card A ExCA Offset 1Ch
Card B ExCA Offset 5Ch
Register: **ExCA Memory Window 2 Offset-Address Low-Byte Register**
Offset: CardBus Socket Address + 824h; Card A ExCA Offset 24h
Card B ExCA Offset 64h
Register: **ExCA Memory Window 3 Offset-Address Low-Byte Register**
Offset: CardBus Socket Address + 82Ch; Card A ExCA Offset 2Ch
Card B ExCA Offset 6Ch
Register: **ExCA Memory Window 4 Offset-Address Low-Byte Register**
Offset: CardBus Socket Address + 834h; Card A ExCA Offset 34h
Card B ExCA Offset 74h

Type: Read/Write
Default: 00h
Size: One byte
Description: These registers contain the low byte of the memory window offset address for memory windows 0, 1, 2, 3 and 4. The 8 bits of these registers correspond to bits A19–A12 of the offset address.

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

Register: **ExCA Memory Window 0 Offset-Address High-Byte Register**
Offset: CardBus Socket Address + 815h; Card A ExCA Offset 15h
Card B ExCA Offset 55h
Register: **ExCA Memory Window 1 Offset-Address High-Byte Register**
Offset: CardBus Socket Address + 81Dh; Card A ExCA Offset 1Dh
Card B ExCA Offset 5Dh
Register: **ExCA Memory Window 2 Offset-Address High-Byte Register**
Offset: CardBus Socket Address + 825h; Card A ExCA Offset 25h
Card B ExCA Offset 65h
Register: **ExCA Memory Window 3 Offset-Address High-Byte Register**
Offset: CardBus Socket Address + 82Dh; Card A ExCA Offset 2Dh
Card B ExCA Offset 6Dh
Register: **ExCA Memory Window 4 Offset-Address High-Byte Register**
Offset: CardBus Socket Address + 835h; Card A ExCA Offset 35h
Card B ExCA Offset 75h

Type: Read Only, Read/Write
Default: 00h
Size: One byte
Description: These registers contain the high byte of the memory window offset address for memory windows 0, 1, 2, 3 and 4. In addition, the memory window write protection and common/attribute memory configurations are set in this register. Table 39 describes each bit in the ExCA memory window offset-address high-byte register.

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

Table 39. ExCA Memory Window Offset-Address High-Byte Register

BIT	TYPE	FUNCTION
7	R/W	Write protect. Bit 7 specifies whether write operations to this memory window are enabled. Bit 7 is encoded as: 0 = Write operations are allowed (default). 1 = Write operations are not allowed.
6	R/W	REG. Bit 6 specifies whether this memory window is mapped to card attribute or common memory. Bit 6 is encoded as: 0 = Memory window is mapped to common memory (default). 1 = Memory window is mapped to attribute memory.
5–0	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 0–1 offset-address low-byte register (index 36h, 38h)

Register: **ExCA I/O Window 0 Offset-Address Low-Byte Register**
Offset: CardBus Socket Address + 836h; Card A ExCA Offset 36h
Card B ExCA Offset 76h
Register: **ExCA I/O Window 1 Offset-Address Low-Byte Register**
Offset: CardBus Socket Address + 838h; Card A ExCA Offset 38h
Card B ExCA Offset 78h
Type: Read/Write
Default: 00h
Size: One byte
Description: These registers contain the low byte of the I/O window offset address for I/O windows 0 and 1. Bit 0 is always 0.

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

Register: **ExCA I/O Window 0 Offset-Address High-Byte Register**
Offset: CardBus Socket Address + 837h; Card A ExCA Offset 37h
Card B ExCA Offset 77h
Register: **ExCA I/O Window 1 Offset-Address High-Byte Register**
Offset: CardBus Socket Address + 839h; Card A ExCA Offset 39h
Card B ExCA Offset 79h
Type: Read/Write
Default: 00h
Size: One byte
Description: These registers contain the high byte of the I/O window offset address for I/O windows 0 and 1.

ExCA card detect and general control register (Index 16h)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Card Detect and General Control Register							
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 Register**
Type: Read Only, Write Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 816h; Card A ExCA Offset 16h
Card B ExCA Offset 56h

Default: XX00 0000b
Description: This register controls how the ExCA registers for the socket respond to card removal, as well as reporting the status of the $\overline{VS1}$ and $\overline{VS2}$ signals at the PC Card interface. Table 40 describes each bit in the ExCA card detect and general control register.

Table 40. ExCA Card Detect and General Control Register (Index 16h)

BIT	TYPE	FUNCTION
7	R	$\overline{VS2}$. Bit 7 reports the current state of the $\overline{VS2}$ signal at the PC Card interface and does not have a default value. Bit 7 is encoded as: 0 = $\overline{VS2}$ is low. 1 = $\overline{VS2}$ is high.
6	R	$\overline{VS1}$. Bit 6 reports the current state of the $\overline{VS1}$ signal at the PC Card interface and does not have a default value. Bit 6 is encoded as: 0 = $\overline{VS1}$ is low. 1 = $\overline{VS1}$ is high.
5	W	Software card detect interrupt. If the card detect enable bit in the card status change interrupt configuration register (see *ExCA 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 operation of this bit always returns 0. Bit 5 is encoded as: 0 = Software card detect interrupt disabled (default) 1 = Software card detect interrupt enabled
4	R/W	Card detect resume enable. If bit 4 is set to 1 and detect change has been detected on the $\overline{CD1}$ and $\overline{CD2}$ inputs, RI_OUT output goes from high to low. The RI_OUT remains low until the card status change bit in the ExCA card status-change register (see *ExCA card status-change register*) is cleared. If bit 4 is a 0, the card detect resume functionality is disabled. Bit 4 is encoded as: 0 = Card detect resume disabled (default) 1 = Card detect resume enabled
3–2	R	Reserved. Bits 3–2 are read only and return 0s when read. Writes have no effect.
1	R/W	Register configuration upon card removal. Bit 1 determines how the ExCA registers for the socket react to a card removal event. Bit 1 is encoded as: 0 = No change to ExCA registers upon card removal (default) 1 = Reset ExCA registers upon card removal
0	R	Reserved. Bit 0 is read only and returns 0s when read. Writes have no effect.

PRODUCT PREVIEW

ExCA global control register (Index 1Eh)

Bit	7	6	5	4	3	2	1	0
Name	ExCA Global Control Register							
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 Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 81Eh; Card A ExCA Offset 1Eh
Card B ExCA Offset 5Eh
Default: 00h
Description: This register controls both PC Card sockets and is not duplicated for each socket. The host interrupt mode bits in this register (retained for Intel 82365SL-DF compatibility) must also agree with the interrupt mode registers found in the TI extension registers. Host software is responsible for maintaining coherence between these registers. Table 41 describes each bit in the ExCA global control register.

Table 41. ExCA Global Control Register (Index 1Eh)

BIT	TYPE	FUNCTION
7–5	R	Reserved. Bits 7–5 are read only and return 0s when read. Writes have no effect.
4	R/W	Level/edge interrupt mode select – Card B. Bit 4 selects the signaling mode for the PCI1130 host interrupt for Card B interrupts. Bit 4 is encoded as: 0 = Host interrupt is in edge mode (default). 1 = Host interrupt is in level mode.
3	R/W	Level/edge interrupt mode select – Card A. Bit 3 selects the signaling mode for the PCI1130 host interrupt for Card A interrupts. Bit 3 is encoded as: 0 = Host interrupt is in edge mode (default). 1 = Host interrupt is in level mode.
2	R/W	Interrupt flag clear mode select. Bit 2 selects explicit writeback of card status-change interrupt acknowledges. Bit 2 is encoded as: 0 = Card status-change interrupt flags are cleared by a read of ExCA card status-change register (default). 1 = Card status-change interrupt flags are cleared by explicit writeback of one to card status-change register.
1	R/W	Card status-change level/edge mode select. Bit 1 selects the signaling mode for the PCI1130 host interrupt for card status changes. Bit 1 is encoded as: 0 = Host interrupt is in edge mode (default). 1 = Host interrupt is in level mode.
0	R/W	PWRDWN mode select. When bit 0 is set to 1, the PCI1130 is in power-down mode. In power-down mode, the PCI1130 outputs are placed in a high-impedance state until an active cycle is executed on the card interface. Following an active cycle, the outputs are again placed in a high-impedance state. The PCI1130 still receives DMA requests, functional interrupts and/or card status change interrupts; however, an actual card access is required to wake up the interface. In power-down mode, the PCI1130 is in a minimum power consumption state. Bit 0 is encoded as: 0 = Power-down mode is disabled (default). 1 = Power-down mode is enabled.

ExCA memory window page register

Bit	7	6	5	4	3	2	1	0
Name	ExCA Memory Window Page Register							
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 Page Register**
Type: Read/Write
Offset: CardBus Socket Address + 840h; Card A ExCA Offset 3Ch
Card B ExCA Offset 7Ch
Default: 00h
Description: The upper 8 bytes of a PCI memory address are compared to the contents of this register when decoding addresses for 16-bit memory windows. By programming this register to a value other than 0, host software can locate 16-bit memory windows in any one of 256 16M-byte regions in the 4G-byte PCI address space. The default register values, 00h, locates 16-bit memory windows in the first 16M bytes of address space.

CardBus socket registers

The PCMCIA CardBus specification requires a CardBus socket controller to provide five 32-bit registers that report and control the socket-specific functions. The PCI1130 provides the CardBus socket base address register (see *CardBus socket registers/ExCA register base address register*) to locate these CardBus socket registers in PCI memory address space. Each socket has a separate CardBus socket register/ExCA registers base address register for accessing the CardBus socket registers (see Figure 13). This base address register is located at offset 10h in the PCI1130 configuration space. Table 42 illustrates the location of the CardBus socket registers in relation to the CardBus socket base address. The CardBus socket power management register (see *CardBus socket power management register*) is an extended register that provides control and status information related to power management. This register is described in detail in *power management*.

Table 42. CardBus Socket Registers

REGISTER NAME	OFFSET
Socket event register	00h
Socket mask register	04h
Socket present state register	08h
Socket force event register	0Ch
Socket control register	10h
Reserved	14–1Fh
Socket power management register	20h

CardBus socket registers (continued)

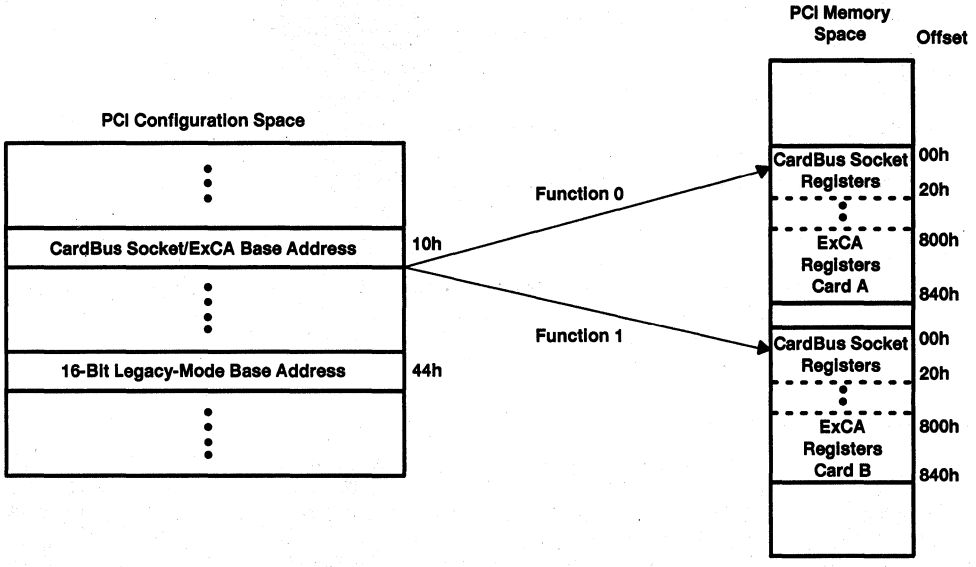


Figure 13. CardBus Socket/ExCA PCI Memory Access Method

CardBus socket event register

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CardBus Socket Event Register															
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	CardBus Socket Event Register															
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: **CardBus Socket Event Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 00h
Default: 0000 0000h
Size: Four bytes
Description: The CardBus socket event register indicates a change in socket status has occurred. These bits do not indicate what the change is, only that 1 has occurred. Software must read the CardBus socket present state register for current status. Each bit in this register can be cleared by writing a 1 to that bit. These bits can be set to a 1 by software through writing a 1 to the corresponding bit in the CardBus socket force event register. All bits in this register are cleared by PCI reset. If, when coming out of PC Card reset, the bridge finds the status unchanged (i.e., CSTSCHG reasserted or card detect is still true), they can be set again. Software needs to clear this register before enabling interrupts. If it is not cleared when interrupts are enabled, an interrupt is generated based on any bit set but not masked. Table 43 describes each bit in this register.

Table 43. CardBus Socket Event Register

BIT	TYPE	FUNCTION
31–4	R	Reserved. Bits 31–4 are read only and return 0s when read. Writes have no effect.
3	R/W	PowerCycle. Bit 3 is set when the PCI1130 detects that the PowerCycle bit in the CardBus present state register has changed. Bit 3 is reset by writing a 1.
2	R/W	$\overline{\text{CCD2}}$. Bit 2 is set whenever the $\overline{\text{CCD2}}$ field in the socket's CardBus socket-present state register changes state. Bit 2 is reset by writing a 1.
1	R/W	$\overline{\text{CCD1}}$. Bit 1 is set whenever the $\overline{\text{CCD1}}$ field in the socket's CardBus socket-present state register changes state. Bit 1 is reset by writing a 1.
0	R/W	CSTSCHG. Bit 0 is set whenever the CSTSCHG field in the socket's CardBus socket-present state register changes state. For CardBus cards, bit 0 is set on the rising edge of the CSTSCHG signal. For 16-bit PC Cards, bit 0 is set on both transitions of the CSTSCHG signal. Bit 0 is reset by writing a 1.

PRODUCT PREVIEW

CardBus socket mask register

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CardBus Socket Mask Register															
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	CardBus Socket Mask Register															
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: **CardBus Socket Mask Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 04h
Default: 0000 0000h
Size: Four bytes
Description: This register allows host software to control the CardBus card events that generate a status change interrupt. Table 44 describes each bit in this register. The state of these mask bits does not prevent the analogous bits from reacting in the CardBus socket event register.

Table 44. CardBus Socket Mask Register

BIT	TYPE	FUNCTION
31–4	R	Reserved. Bits 31–4 are read only and return 0s when read. Writes have no effect.
3	R/W	PowerCycle. Bit 3 masks the PowerCycle bit in the socket's CardBus socket-event register from causing a status change interrupt. Bit 3 is set by writing a 1. Bit 3 is encoded as: 0 = PowerCycle event does not cause a status-change interrupt (default). 1 = PowerCycle event causes a status-change interrupt.
2–1	R/W	CardDetect. When reset (00b), bits 2–1 mask the $\overline{\text{CCD1}}$ and $\overline{\text{CCD2}}$ bits in the socket's CardBus socket-event register from causing a status-change interrupt. Bits 2–1 are set by writing an 11. This field is encoded as: 00 = Card insertion/removal events does not cause a status-change interrupt (default). 01 = Undefined condition 10 = Undefined condition 11 = Card insertion/removal events causes a status-change interrupt.
0	R/W	CSTSCHG. When reset, bit 0 masks the CSTSCHG from the CardBus PC Card from causing a status-change interrupt. Bit 0 is set by writing a 1. Bit 0 is encoded as: 0 = CSTSCHG event does not cause a status-change interrupt (default). 1 = CSTSCHG event causes a status-change interrupt.

CardBus socket present state register

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CardBus Socket Present State Register															
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	CardBus Socket Present State Register															
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	1	1	0

Register: **CardBus Socket Present State Register**
Type: Read Only
Offset: CardBus Socket Address + 08h
Default: 3000 0006h
Size: Four bytes
Description: This register reports information about the socket interface. Writes to the CardBus socket force event register are reflected here. Table 45 describes each bit in this register. Information about supported V_{CC} are hardwired (unless overridden by the CardBus socket force event register), while information about PC Card VCC support is dynamic and updated at each insertion. The PCI1130 uses the $\overline{CCD1}$ and $\overline{CCD2}$ signals during card identification, and changes on these signals during this operation are not reflected in this register.

Table 45. CardBus Socket Present State Register

BIT	TYPE	FUNCTION
31	R	YVsocket. Bit 31 indicates whether or not the socket can supply V_{CC} = Y.Y V to PC Cards. The PCI1130 does not support Y.Y V V_{CC}; therefore, bit 31 is always reset unless overridden by the CardBus socket force event register. Bit 31 is hardwired to 0.
30	R	XVsocket. Bit 30 indicates whether or not the socket can supply V_{CC} = X.X V to PC Cards. The PCI1130 does not support X.X V V_{CC}; therefore, bit 30 is always reset unless overridden by the CardBus socket force event register. Bit 30 is hardwired to 0.
29	R	3Vsocket. Bit 29 indicates whether or not the socket can supply V_{CC} = 3.3 V to PC Cards. The PCI1130 supports 3.3 V V_{CC}; therefore, bit 29 is always set unless overridden by the CardBus socket force event register. Bit 29 is encoded as: 0 = Socket cannot supply V_{CC} = 3.3 V. 1 = Socket can supply V_{CC} = 3.3 V (default).
28	R	5Vsocket. Bit 28 indicates whether or not the socket can supply V_{CC} = 5.0 V to PC Cards. The PCI1130 supports 5.0 V V_{CC}; therefore, bit 28 is always set unless overridden by the CardBus socket force event register. Bit 28 is encoded as: 0 = Socket cannot supply V_{CC} = 5.0 V. 1 = Socket can supply V_{CC} = 5.0 V (default).
27–14	R	Reserved. Bits 27–14 are read only and return 0s when read. Writes have no effect.
13	R	YVCard. Bit 13 indicates whether or not the PC Card currently inserted in the socket supports V_{CC} = Y.Y V. Bit 13 is encoded as: 0 = PC Card does not function at V_{CC} = Y.Y V (default). 1 = PC Card functions at V_{CC} = Y.Y V.
12	R	XVCard. Bit 12 indicates whether or not the PC Card currently inserted in the socket supports V_{CC} = X.X V. Bit 12 is encoded as: 0 = PC Card does not function at V_{CC} = X.X V (default). 1 = PC Card functions at V_{CC} = X.X V.
11	R	3VCard. Bit 11 indicates whether or not the PC Card currently inserted in the socket supports V_{CC} = 3.3 V. Bit 11 is encoded as: 0 = PC Card does not function at V_{CC} = 3.3 V (default). 1 = PC Card functions at V_{CC} = 3.3 V.

PRODUCT PREVIEW

CardBus socket present state register (continued)

Table 45. CardBus Socket Present State Register (Continued)

BIT	TYPE	FUNCTION
10	R	5VCard. Bit 10 indicates whether or not the PC Card currently inserted in the socket supports V_{CC} = 5.0 V. Bit 10 is encoded as: 0 = PC Card does not function at V_{CC} = 5.0 V (default) 1 = PC Card functions at V_{CC} = 5.0 V
9	R	BadV_{CC}Req. Bit 9 indicates that host software has requested that the socket be powered at an invalid voltage. Bit 9 is encoded as: 0 = Normal operation (default) 1 = Invalid V_{CC} requested by host software
8	R	DataLost. Bit 8 indicates that a PC Card removal event may have caused lost data because the cycle did not terminate properly or write data still resides in the PCI1130. Bit 8 is encoded as: 0 = Normal operation (default) 1 = Potential data loss due to card removal
7	R	NotACard. Bit 7 indicates that an unrecognizable PC Card has been inserted in the socket. Bit 7 is not updated until a valid PC Card is inserted in the socket. Bit 7 is encoded as: 0 = Normal operation (default) 1 = Unrecognizable PC Card detected
6	R	READY($\overline{IREQ}$)/$\overline{CINT}$. Bit 6 indicates the current status of the READY($\overline{IREQ}$)/$\overline{CINT}$ signal at the PC Card interface. Bit 6 is encoded as: 0 = READY($\overline{IREQ}$)/$\overline{CINT}$ is low (default) 1 = READY($\overline{IREQ}$)/$\overline{CINT}$ is high The READY signal applies to 16-bit memory PC Cards. $\overline{IREQ}$ applies to 16-bit I/O PC Cards and $\overline{CINT}$ applies to 32-bit CardBus cards.
5	R	CBcard. Bit 5 indicates that a CardBus PC Card is inserted in the socket. Bit 5 is not updated until a subsequent removal and insertion event. Bit 5 is encoded as: 0 = CardBus PC Card not detected (default) 1 = CardBus PC Card detected
4	R	16-bit card. Bit 4 indicates that a 16-bit PC Card is inserted in the socket. Bit 4 is not updated until a subsequent removal and insertion event. Bit 4 is encoded as: 0 = 16-bit PC Card not detected (default) 1 = 16-bit PC Card detected
3	R	PowerCycle. Bit 3 indicates the status of each power-up/power-down request. Bit 3 is encoded as: 0 = Socket is powered down (default). 1 = Socket has successfully powered up.
2	R	$\overline{CCD2}$. Bit 2 reflects the current status of the $\overline{CCD2}$ signal at the PC Card interface. Changes to this signal during card interrogation are not reflected here. Bit 2 is encoded as: 0 = $\overline{CCD2}$ is low; PC Card may be present. 1 = $\overline{CCD2}$ is high; no PC Card is present (default).
1	R	$\overline{CCD1}$. Bit 1 reflects the current status of the $\overline{CCD1}$ signal at the PC Card interface. Changes to this signal during card interrogation are not reflected here. Bit 1 is encoded as: 0 = $\overline{CCD1}$ is low; PC Card may be present. 1 = $\overline{CCD1}$ is high; no PC Card is present (default).
0	R	CSTSCHG. Bit 0 reflects the current status of the CSTSCHG signal at the PC Card interface. Bit 0 is encoded as: 0 = CSTSCHG is low (deasserted) (default). 1 = CSTSCHG is high (asserted).

CardBus socket force event register

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CardBus Socket Force Event Register															
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	CardBus Socket Force Event Register															
Type	R	W	W	W	W	W	W	W	W	R	W	W	W	W	W	W
Default	0	X	X	X	X	X	X	X	X	0	X	X	X	X	X	X

Register: **CardBus Socket Force Event Register**
Type: Read Only, Write Only (see individual bit descriptions)
Offset: CardBus Socket Address + 0Ch
Default: NA
Size: Four bytes
Description: This register is not physically implemented, but is an address to which changes to the CardBus socket event and CardBus socket present state registers can be written. Host software can write to this register to simulate events. When host software modifies the XVCard bits in this register, the PCI1130 does not update the TPS2202 power switch until the CVSTEST bit is set. Table 46 describes each bit in this register.

NOTE:
When writing to this register, always write to the CVSTEST bit.

Table 46. CardBus Socket Force Event Register

BIT	TYPE	FUNCTION
31–15	R	Reserved. Bits 31–15 are read only and return 0s when read. Writes have no effect.
14	W	CVSTEST. When bit 14 is set, the PCI1130 reinterrogates the PC Card, updates the XVCard fields in the CardBus socket present state register and reenables the socket power control.
13	W	YVCard. Writes to bit 13 cause the YVCard bit in the CardBus socket present state register to be written. When set, bit 13 disables the socket power control.
12	W	XVCard. Writes to bit 12 cause the XVCard bit in the CardBus socket present state register to be written. When set, bit 12 disables the socket power control.
11	W	3VCard. Writes to bit 11 cause the 3VCard bit in the CardBus socket present state register to be written. When set, bit 11 disables the socket power control.
10	W	5VCard. Writes to bit 10 cause the 5VCard bit in the CardBus socket present state register to be written. When set, bit 10 disables the socket power control.
9	W	BadVccReq. Writes to bit 9 cause the BadVccReq bit in the CardBus socket present state register to be written.
8	W	DataLost. Writes to bit 8 cause the DataLost bit in the CardBus socket present state register to be written.
7	W	NotACard. Writes to bit 7 cause the NotACard bit in the CardBus socket present state register to be written.
6	R	Reserved. Bit 6 is read only and returns 0s when read. Writes have no effect.
5	W	CBcard. Writes to bit 5 cause the CBcard bit in the CardBus socket present state register to be written. Writes to bit 5 are ignored if a card is present in the socket.
4	W	16-bitcard. Writes to bit 4 cause the 16-bitcard bit in the CardBus socket present state register to be written. Writes to bit 4 are ignored if a card is present in the socket.
3	W	PowerCycle. Setting bit 3 causes the PowerCycle bit in the CardBus socket event register to be set. The PowerCycle bit in the CardBus socket present state register is unaffected by writes to bit 3.
2	W	$\overline{\text{CCD2}}$. Setting bit 2 causes the $\overline{\text{CCD2}}$ bit in the CardBus socket event register to be set. The $\overline{\text{CCD2}}$ bit in the CardBus socket present state register is unaffected by writes to bit 2.
1	W	$\overline{\text{CCD1}}$. Setting bit 1 causes the $\overline{\text{CCD1}}$ bit in the CardBus socket event register to be set. The $\overline{\text{CCD1}}$ bit in the CardBus socket present state register is unaffected by writes to bit 1.
0	W	CSTSCHG. Setting bit 0 causes the CSTSCHG bit in the CardBus socket event register to be set. The CSTSCHG bit in the CardBus socket present state register is unaffected by writes to bit 0.

PRODUCT PREVIEW

CardBus socket control register

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CardBus Socket Control Register															
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	CardBus Socket Control Register															
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: **CardBus Socket Control Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 10h
Default: 0000 0000h
Size: Four bytes
Description: This register provides control of the voltages applied to the socket's V_{PP} and V_{CC}. The PCI1130 ensures that the socket is powered up only at acceptable voltages when a CardBus card is inserted. Table 47 describes each bit in this register.

Table 47. CardBus Socket Control Register

BIT	TYPE	FUNCTION
31–8	R	Reserved. Bits 31–8 are read only and return 0s when read. Writes have no effect.
7	R/W	Stop clock. When bit 7 is set, it causes the bridge to stop the CardBus clock (CCLK) using the CLKRUN protocol. Bit 7 is encoded as: 0 = Clock stopping disabled (default) 1 = Clock stopping enabled
6–4	R/W	V_{CC}Control. Bits 6–4 are used to request changes to card V_{CC}. Bits 6–4 are encoded as: 000 = Request V_{CC} power off (default) 001 = Reserved 010 = Request V_{CC} = 5.0 V 011 = Request V_{CC} = 3.3 V 100 = Request V_{CC} = X.X V 101 = Request V_{CC} = Y.Y V 110 = Reserved 111 = Reserved
3	R	Reserved. Bit 3 is read only and returns 0s when read. Writes have no effect.
2–0	R/W	V_{PP}Control. Bits 2–0 are used to request changes to card V_{PP}. Bits 2–0 are encoded as: 000 = Request V_{PP} power off (default) 001 = Request V_{PP} = 12.0 V 010 = Request V_{PP} = 5.0 V 011 = Request V_{PP} = 3.3 V 100 = Request V_{PP} = X.X V 101 = Request V_{PP} = Y.Y V 110 = Reserved 111 = Reserved

CardBus socket power management register

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CardBus Socket Power Management Register															
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	CardBus Socket Power Management Register															
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: **CardBus Socket Power Management Register**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: CardBus Socket Address + 20h
Default: 0000h
Size: Four bytes
Description: This register provides control over power management for the socket. It provides a mechanism for slowing or stopping the clock on the card interface when the card is idle. Table 48 describes each bit in this register.

Table 48. CardBus Socket Power Management Register

BIT	TYPE	FUNCTION
31–26	R	Reserved. Bits 31–26 are read only and return 0s when read. Writes have no effect.
25	R	Socket access status. Bit 25 provides information on when a socket access has occurred. Reading bit 25 clears it. Bit 25 is encoded as: 0 = No PC Card access has occurred (default). 1 = Host system has accessed PC Card.
24	R	Socket mode status bit. Bit 24 provides clock mode information. Write a 1 to clear this bit. Bit 24 is encoded as: 0 = Normal operation (default) 1 = Clock frequency has changed.
23–17	R	Reserved. Bits 23–17 are read only and return 0s when read. Writes have no effect.
16	R/W	CardBus PC Card clock control enable bit. Bit 16 enables the PC Card clock control bit, bit 0, to be enabled. Bit 16 is encoded as: 0 = Disabled (default) 1 = Enabled
15–1	R	Reserved. Bits 15–1 are read only and return 0s when read. Writes have no effect.
0	R/W	CardBus PC Card clock control bit. Bit 0, when enabled by bit 16, provides control over the PC Card clock. Bit 0 is encoded as: 0 = Clock stopped (default) 1 = Divide by 16

DMA registers

The DMA base address register, located in PCI configuration space at offset 98h (see *socket DMA register 0*), points to a 16-byte region in PCI I/O space where the DMA registers reside. The names and locations of these registers are summarized in Table 5. These registers are identical in function 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 forwarding legacy DMA writes to DMA channels.

While the DMA register definitions are identical to those in the 8237 DMA controller, some register bits defined in the 8237 DMA controller do not apply to distributed DMA in a PCI environment. In such cases, the PCI1130 implements these obsolete register bits as read-only, nonfunctional bits. The reserved registers shown in Table 5 are implemented as read only and return 0s when read. Writes to reserved registers have no effect.

Table 49. DMA Registers

R/W	REGISTER NAME				DMA BASE ADDRESS OFFSET
R	Reserved	Page	Current address		00h
W			Base address		
R	Reserved	Reserved	Current word		04h
W			Base word		
R	NA	Reserved	NA	Status	08h
W	Mode		Request	Command	
R	Multichannel mask	Reserved	NA	Reserved	0Ch
W			Master clear		

DMA current address/base address

Register: **DMA Current Address/Base Address**
Type: Read/Write
Offset: DMA Base Address + 00h
Default: 00 0000h
Size: Three bytes
Description: Writes to this register set the starting (base) memory address of a DMA transfer. Reads from this register indicate the current memory address of a DMA transfer.

For 8-bit DMA transfer mode, the DMA 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 16-bit DMA transfer mode, the DMA current address register contents are presented on AD16–AD1 of the PCI bus during the address phase. AD0 is equal to 0. Bits 7–1 of the page register are presented on AD23–AD17 of the PCI bus during the address phase. Bit 0 of the page register is ignored.

DMA current word/base word

Register: **DMA Current Word/Base Word**
Type: Read/Write
Offset: DMA Base Address + 04h
Default: 0000h
Size: Two bytes
Description: Writes to this register set the total transfer count, in bytes, of a DMA transfer. Reads to this register indicate the current count of a DMA transfer. When nonlegacy addressing mode is disabled, the upper 8 bits of this register are reserved and behave as a reserved register. This addressing mode forces compliance with the transfer size in legacy 8237 DMA transfers. When nonlegacy addressing mode is enabled, the full 24-bit address range is used.

DMA status/command

Bit	7	6	5	4	3	2	1	0
Name	DMA Status							
Type	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
Name	DMA Command							
Type	R	R	R	R	R	R/W	R	R
Default	0	0	0	0	0	0	0	0

Register: **DMA Status/Command**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: DMA Base Address + 08h
Default: 00h
Size: One byte
Description: This address contains both the DMA status and command registers. During PCI I/O read cycles to this address, the PCI1130 returns the contents of the DMA status register. During PCI I/O write cycles to this address, the DMA command register is written. The DMA status and command registers remain in accordance with the 8237 DMA controller register definitions; however, certain bits are not implemented in the PCI1130. Table 50 describes the DMA status register bits; Table 51 describes the DMA command register bits.

Table 50. DMA Status Register

BIT	TYPE	FUNCTION
7–4	R	Channel request. In the 8237 DMA controller, bits 7–4 indicate the status of the $\overline{\text{DREQ}}$ signal of each DMA channel. In the PCI1130, the status register only reports information about a single DMA channel; therefore, all four of these register 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 its $\overline{\text{DREQ}}$ signal and are reset when $\overline{\text{DREQ}}$ is high (deasserted). The status of the mask bit in the DMA multichannel mask register has no effect on these bits.
3–0	R	Channel TC. The 8237 DMA controller uses bits 3–0 to indicate the TC status of each of its four DMA channels. In the PCI1130, the status register reports information about just 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 terminal count (TC) is reached by the DMA channel. Bits 3–0 are reset when read or when the DMA channel is reset.

PRODUCT PREVIEW

DMA status/command (continued)

Table 51. DMA Command Register

BIT	TYPE	FUNCTION
7	R	Reserved. Bit 7 is read only and returns 0s when read. Writes have no effect. The 8237 DMA controller uses this register bit to select the DACK signaling active high or low. In the PCI1130, the PC Card signal used as DACK is defined in the PC Card standard as active high; therefore, bit 7 is reserved.
6	R	Reserved. Bit 6 is read only and returns 0s when read. Writes have no effect. The 8237 DMA controller uses this register bit to select the $\overline{\text{DREQ}}$ signaling active high or low. In the PCI1130, the PC Card signal used as $\overline{\text{DREQ}}$ is defined in the PC Card standard as active low; therefore, bit 6 is reserved.
5	R	Reserved. Bit 5 is read only and returns 0s when read. Writes have no effect. In the 8237 DMA controller, this register bit selects late or extended write mode. These types of cycles have no meaning in the PCI or PC Card environment; therefore, bit 5 is reserved in the PCI1130.
4	R	Reserved. Bit 4 is read only and returns 0s when read. Writes have no effect. In the 8237 DMA controller, this register bit selects rotating or fixed priority between DMA channels. Priority servicing has no meaning in the PCI distributed DMA environment, so bit 4 is reserved in the PCI1130. Priority to a particular DMA channel on the PCI1130 is given when the device asserts its PCI $\overline{\text{REQ}}$ signal and is granted use of the PCI bus.
3	R	Reserved. Bit 3 is read only and returns 0s when read. Writes have no effect. The 8237 DMA controller uses this register bit to select normal or compressed timing. This functionality has no meaning on either the PCI or PC Card interfaces, where the transfer timing is rigorously defined. Therefore, bit 3 is reserved in the PCI1130.
2	R/W	DMA controller enable/disable. In the 8237 DMA controller, register bit 2 enables/disables the DMA controller. This functionality is retained in the PCI1130, but enables or disables only the particular DMA channel of the command register. Bit 2 defaults to the enabled state. 0 = DMA controller enabled (default) 1 = DMA controller disabled
1	R	Reserved. Bit 1 is read only and returns 0s when read. Writes have no effect. In the 8237 DMA controller, this register bit is used with memory-to-memory transfers. Memory-to-memory transfers are not supported in the distributed DMA specification; therefore, bit 1 is reserved in the PCI1130.
0	R	Reserved. Bit 0 is read only and returns 0s when read. Writes have no effect. In the 8237 DMA controller, this register bit enables or disables memory-to-memory transfers. Memory-to-memory transfers are not supported in the distributed DMA specification; therefore, bit 0 is reserved in the PCI1130.

DMA request

Bit	7	6	5	4	3	2	1	0
Name	DMA Request							
Type	W	W	W	W	W	W	W	W
Default	0	0	0	0	0	0	0	0

Register: **DMA Request**
Type: Write Only
Offset: DMA Base Address + 09h
Default: 00h
Size: One byte
Description: The request register is used in DMA requests. Writing a 1 to bit 2 of this register enables software requests for DMA transfers. This register is used in block mode only.

PRODUCT PREVIEW

DMA mode

Bit	7	6	5	4	3	2	1	0
Name	DMA 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: **DMA Mode**
Type: Read Only, Read/Write (see individual bit descriptions)
Offset: DMA Base Address + 0Bh
Default: 00h
Size: One byte
Description: The DMA mode register. Table 52 describes the mode register bits.

Table 52. DMA Mode Register

BIT	TYPE	FUNCTION
7–6	R/W	Mode select bits. The PCI1130 uses bits 7–6 to determine which DMA transfer mode to use: single, block or demand. This field is encoded as: 00 = Demand mode select (default) 01 = Single mode select 10 = Block mode select 11 = Reserved
5	R/W	Address increment/decrement. The PCI1130 uses bit 5 to select the memory address in the current/base register to increment or decrement after each data transfer. This is in accordance with the 8237 DMA controller use of this register bit. Bit 5 is encoded as: 0 = Addresses increment (default) 1 = Addresses decrement
4	R/W	Autoinitialization bit. In the PCI1130, bit 4 selects autoinitialization. Bit 4 is encoded as: 0 = Autoinitialization disabled (default) 1 = Autoinitialization enabled
3–2	R/W	Transfer type. Bits 3–2 select the type of DMA transfer to be performed. A DMA write transfer moves data from the PC Card to memory. A DMA read transfer moves data from memory to PC Card. This field is encoded as: 00 = No transfer selected (default) 01 = Write transfer 10 = Read transfer 11 = Reserved
1–0	R	Reserved. Bits 1–0 are read only and return 0s when read. Writes have no effect. The 8237 DMA controller uses register bits 1–0 to select the current channel number for programming. The master DMA device uses bits 1–0 to select the current device. Devices such as the PCI1130 do not require bits 1–0.

DMA master clear

Bit	7	6	5	4	3	2	1	0
Name	DMA Master Clear							
Type	W	W	W	W	W	W	W	W
Default	0	0	0	0	0	0	0	0

Register: **DMA Master Clear**
Type: Write Only
Offset: DMA Base Address + 0Dh
Default: 00h
Size: One byte
Description: The DMA master clear register is a write-only register that, when written with any data, resets the entire DMA channel to the socket and resets all registers to their default condition.

CAUTION:

The master DMA device must select byte enables during PCI writes to other registers within this double word to prevent inadvertent reset.

DMA multichannel mask

Bit	7	6	5	4	3	2	1	0
Name	DMA Multichannel Mask							
Type	R	R	R	R	R	R	R	R/W
Default	0	0	0	0	0	0	0	0

Register: **DMA Multichannel Mask**
Type: Read Only, Read/Write (see individual bit description)
Offset: DMA Base Address + 0Fh
Default: 01h
Size: One byte
Description: The PCI1130 uses only the least-significant bit of the DMA multichannel mask register. Bit 0 is used to mask the DMA channel. Table 53 describes the bits of this register. The PCI1130 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. The DMA controller for the socket is also internally masked by internal flags indicating that a 16-bit PC Card is present in the socket.

Table 53. DMA Multichannel Mask

BIT	TYPE	FUNCTION
7–1	R	Reserved. Bits 7–1 are read only and return 0s when read. Writes have no effect.
0	R/W	Mask select bit. 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. 0 = Mask bit cleared 1 = Mask bit set (default)

PRODUCT PREVIEW

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

Supply voltage range, V_{CC} ..... –0.5 V to 4.6 V
Supply voltage range, V_{CCP} ..... –0.5 V to 6 V
Input voltage range, V_I: Standard ..... –0.5 V to V_{CC} + 0.5 V
Card A ..... –0.5 to $V_{CC(A)}$ + 0.5 V
Card B ..... –0.5 to $V_{CC(B)}$ + 0.5 V
Fail safe ..... –0.5 V to V_{CC} + 0.5 V
Output voltage range, V_O: Standard ..... –0.5 V to V_{CC} + 0.5 V
Card A ..... –0.5 to $V_{CC(A)}$ + 0.5 V
Card B ..... –0.5 to $V_{CC(B)}$ + 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 to external input and bidirectional buffers. $V_I > V_{CC}$ does not apply to fail-safe terminals.
2. Applies to external output and bidirectional buffers. $V_O > V_{CC}$ does not apply to fail-safe terminals.

recommended operating conditions

			MIN	NOM	MAX	UNIT
t_t	Input transition (rise and fall) time	CMOS compatible	0		25	ns
T_A	Operating ambient temperature	Commercial	0	25	70	°C
T_J‡	Virtual junction temperature	Commercial	0	25	115	°C

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

recommended operating conditions for PCI interface

			OPERATION	MIN	NOM	MAX	UNIT
V_{CC}	Core voltage	Commercial	3.3 V	3	3.3	3.6	V
V_{CCP}	PCI supply voltage	Commercial	3.3 V	3	3.3	3.6	V
			5 V	4.75	5	5.25	
V_I	Input voltage		3.3 V	0		V_{CCP}	V
			5 V	0		V_{CCP}	
V_O§	Output voltage		3.3 V	0		V_{CCP}	V
			5 V	0		V_{CCP}	
V_{IH}¶	High-level input voltage	CMOS compatible	3.3 V	0.5 V_{CCP}			V
			5 V	2			
		Fail safe#	3.3 V	0.5 V_{CC}			V
V_{IL}¶	Low-level input voltage	CMOS compatible	3.3 V			0.3 V_{CCP}	V
			5 V			0.8	
		Fail safe#	3.3 V			0.3 V_{CC}	V

§ Applies to external output buffers
¶ Applies to external input and bidirectional buffers without hysteresis
# Fail-safe pins are 16, 56, 68, 72, 74, 82, 122, 134, 138, 140, 149, and 152.

PRODUCT PREVIEW

recommended operating conditions for PC Cards A and B and miscellaneous inputs and outputs

			OPERATION	MIN	NOM	MAX	UNIT
$V_{CC(A/B)}$	PC Card supply voltage	Commercial	3.3 V	3	3.3	3.6	V
			5 V	4.75	5	5.25	
V_I	Input voltage		3.3 V	0		$V_{CC(A/B)}$	V
			5 V	0		$V_{CC(A/B)}$	
V_O†	Output voltage		3.3 V	0		$V_{CC(A/B)}$	V
			5 V	0		$V_{CC(A/B)}$	
V_{IH}‡	High-level input voltage	CMOS compatible	3.3 V	.475 $V_{CC(A/B)}$¶			V
			5 V	2.4			
		Fail safe§	3.3 V	.475 $V_{CC(A/B)}$¶			V
V_{IL}‡	Low-level input voltage	CMOS compatible	3.3 V			.325 $V_{CC(A/B)}$¶	V
			5 V			0.8	
		Fail safe§	3.3 V			.325 $V_{CC(A/B)}$¶	V

† Applies to external output buffers
‡ Applies to external input and bidirectional buffers without hysteresis
§ Fail-safe pins are 16, 56, 68, 72, 74, 82, 122, 134, 138, 140, 149, and 152.
¶ Meets TTL levels, V_{IH} MIN =1.65 V and V_{IL} MAX = 0.99 V

PRODUCT PREVIEW

electrical characteristics over recommended operating conditions (unless otherwise noted)

PARAMETER		SIDE	OPERATION	TEST CONDITIONS	MIN	MAX	UNIT
V_{OH}	High-level output voltage	PCI	3.3 V	I_{OH} = –0.5 mA	0.9 V_{CC}		V
			5 V	I_{OH} = –2 mA	2.4		
		PC Card	3.3 V	I_{OH} = –0.15 mA	0.9 V_{CC}		
			5 V	I_{OH} = –0.15 mA	2.4		
V_{OL}	Low-level output voltage	PCI	3.3 V	I_{OL} = 1.5 mA		0.1 V_{CC}	V
			5 V	I_{OL} = 6 mA		0.55	
		PC Card	3.3 V	I_{OL} = 0.7 mA		0.1 V_{CC}	
			5 V	I_{OL} = 0.7 mA		0.55	
I_{OZ}	High-impedance output current			V_O = V_{CC} or GND#		±10	µA
I_{IL}	Low-level input current*			V_I = GND		–1	µA
I_{IH}	High-level input current*			V_I = V_{CC}‖		1	µA

The 3-state or open-drain outputs must be in the high-impedance state.
‖ Applies to all inputs except TEST
* I_{IL} not tested on DATA (pin 152) due to internal pulldown resistor and I_{IH} not tested on $\overline{\text{SPKROUT}}$ (pin 149) due to internal pullup resistor.

PCI clock/reset timing requirements over recommended ranges of supply voltage and operating free-air temperature (see Figure 15 and Figure 16)

		ALTERNATE SYMBOL	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 Figure 14 and Figure 17)

			ALTERNATE SYMBOL	TEST CONDITIONS	MIN	MAX	UNIT
t_{pd}	Propagation delay time	PCLK to shared signal valid delay time	t_{val}	C_L=50 pF, See Note 5		11	ns
		PCLK to shared signal invalid delay time	t_{inv}		2		
t_{en}	Enable time, high-impedance-to-active delay time from PCLK		t_{on}		2		ns
t_{dis}	Disable time, active-to-high-impedance delay time from PCLK		t_{off}			28	ns
t_{su}	Setup time before PCLK valid		t_{su}		7		ns
t_h	Hold time after PCLK high		t_h		0		ns

NOTE 3: PCI shared signals are AD31–AD0, C/$\overline{BE3}$–C/$\overline{BE0}$, $\overline{FRAME}$, $\overline{TRDY}$, $\overline{IRDY}$, $\overline{STOP}$, IDSEL, $\overline{DEVSEL}$, and PAR.

PRODUCT PREVIEW

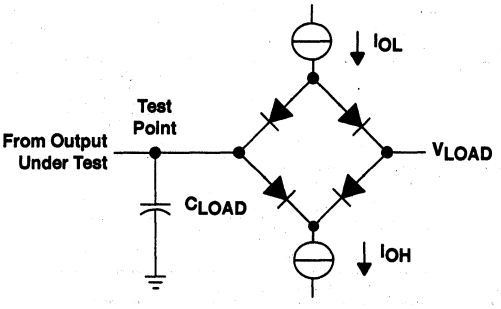
PARAMETER MEASUREMENT INFORMATION

LOAD CIRCUIT PARAMETERS

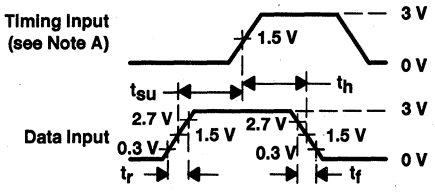
TIMING PARAMETER		C_{LOAD}† (pF)	I_{OL} (mA)	I_{OH} (mA)	V_{LOAD} (V)
t_{en}	t_{PZH}	50	8	– 8	0
	t_{PZL}				3
t_{dis}	t_{PHZ}	50	8	– 8	1.5
	t_{PLZ}				
t_{pd}		50	8	– 8	‡

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

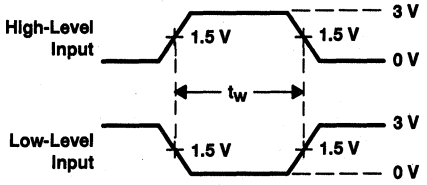
‡ $\frac{V_{LOAD} - V_{OL}}{I_{OL}}$ = 50 Ω, where V_{OL} = 0.6 V, I_{OL} = 8 mA



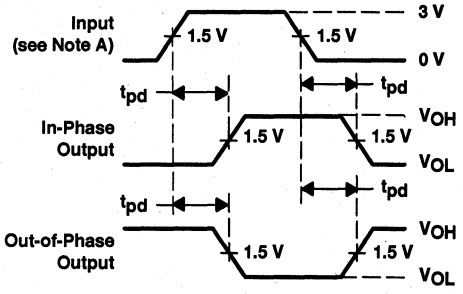
LOAD CIRCUIT



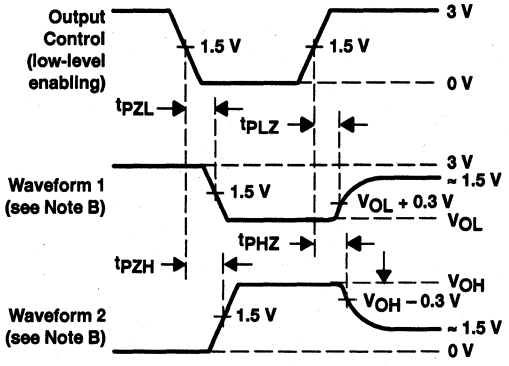
VOLTAGE WAVEFORMS
SETUP AND HOLD TIMES
INPUT RISE AND FALL TIMES



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, $t_r \le 6$ ns, $t_f \le 6$ 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 14. Load Circuit and Voltage Waveforms

PRODUCT PREVIEW

PCIBUS PARAMETER MEASUREMENT INFORMATION

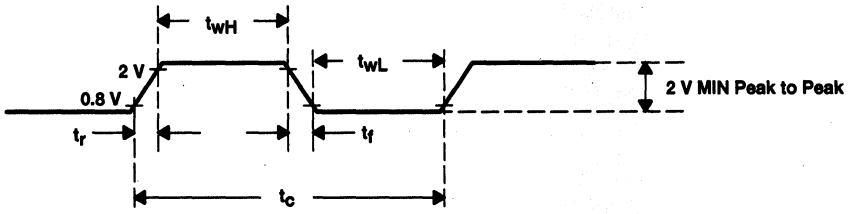


Figure 15. PCLK Timing Waveform

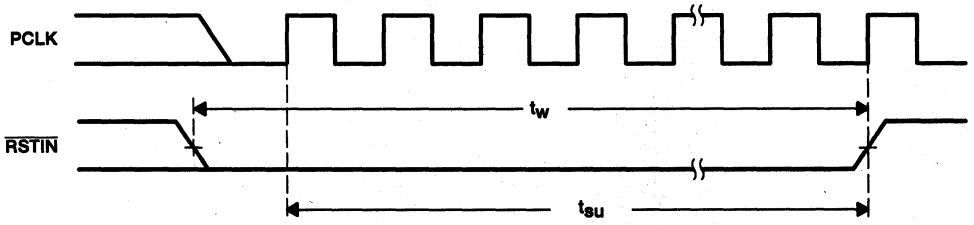


Figure 16. $\overline{\text{RSTIN}}$ Timing Waveforms

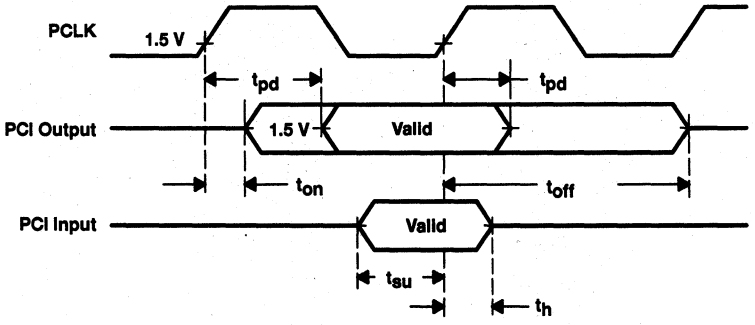


Figure 17. Shared Signals Timing Waveforms

PRODUCT PREVIEW

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 54 shows address setup time in PCLK cycles and nanoseconds for I/O and memory cycles. Table 55 and Table 56 show command active time in PCLK cycles and nanoseconds for I/O and memory cycles. Table 57 shows address hold time in PCLK cycles and nanoseconds for I/O and memory cycles.

Table 54. 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 55. PC Card Command Active 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 56. PC Card Command Active 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 57. 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 6 and Figure 18)

		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 4: 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 19)

		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 10)

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

PRODUCT PREVIEW

PC CARD PARAMETER MEASUREMENT INFORMATION

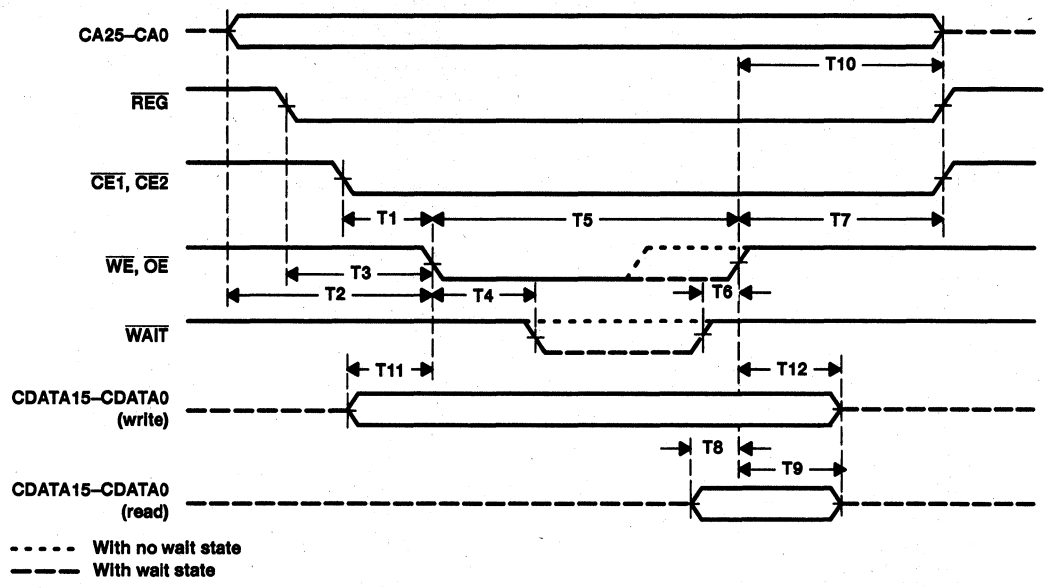


Figure 18. PC Card Memory Cycle

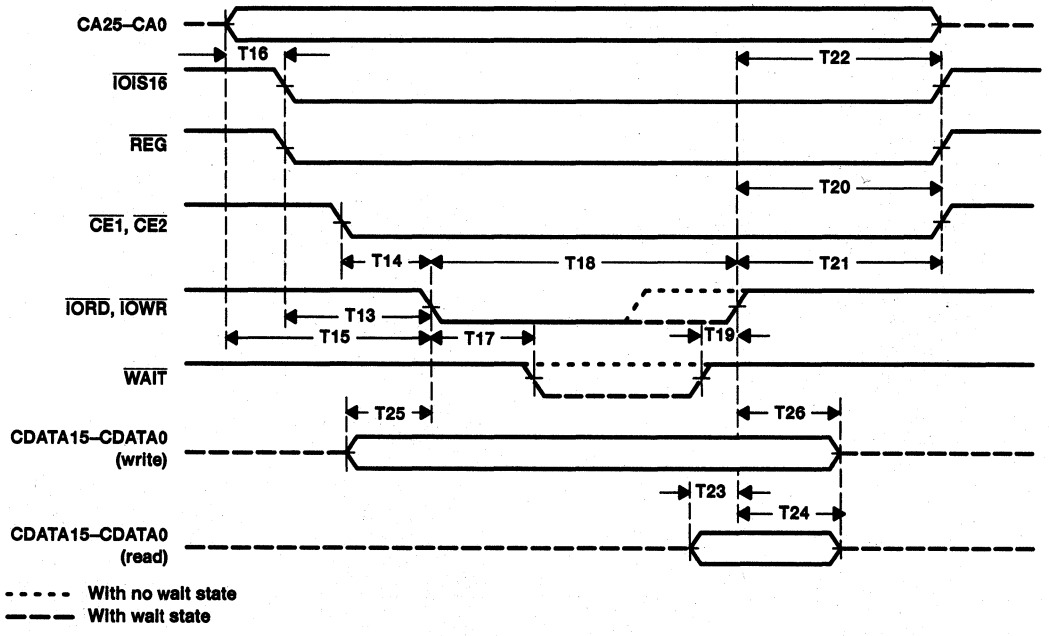


Figure 19. PC Card I/O Cycle

PRODUCT PREVIEW

PC CARD PARAMETER MEASUREMENT INFORMATION

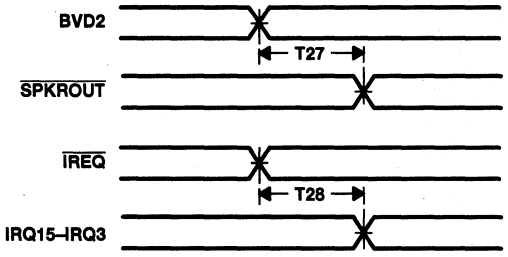


Figure 20. Miscellaneous PC Card Delay Times

SCPS009 – FEBRUARY 1996

- **3.3-V Core Logic With Universal PCI Interface Compatible With 3.3-V or 5-V PCI Signaling Environments**
- **Supports PCI Local Bus Specification 2.1**
- **Supports Two 32-Bit, 33-MHz PCI Buses Providing Concurrent Operation**
- **Provides VGA/Palette Memory and I/O and Subtractive Decoding Options**
- **Supports Burst Transfers to Maximize Data Throughput on Both PCI Buses**
- **Propagates Bus Locking**
- **Provides Six Secondary PCI Bus Clock Outputs**
- **Provides Internal Arbitration for up to Seven Secondary Bus Masters With Programmable Control**
- **Low-Power Advanced Submicron CMOS Technology**
- **Independent Read and Write Buffers for Each Direction**
- **Predictable Latency: 16 Clocks on Initial Data Phase and 8 Clocks on Subsequent Data Phases**
- **Supports PCI Clock Run**
- **Secondary Bus Is Driven Low During Reset**

description

The Texas Instruments (TI) PCI20XX provides a high-performance connection path between two peripheral component interconnect (PCI) buses. Transactions can occur between a master on one PCI bus and a target on another PCI bus. The PCI20XX supports burst mode transfers to maximize data throughput. The two bus traffic paths through the PCI20XX act independently.

The PCI20XX is compliant with the PCI local bus specification revision 2.1 and can be used to overcome the electrical loading limit of ten devices per PCI bus by creating hierarchical buses. Furthermore, add-in cards requiring multiple PCI devices can use the PCI20XX to overcome the electrical loading limit of one PCI device per slot.

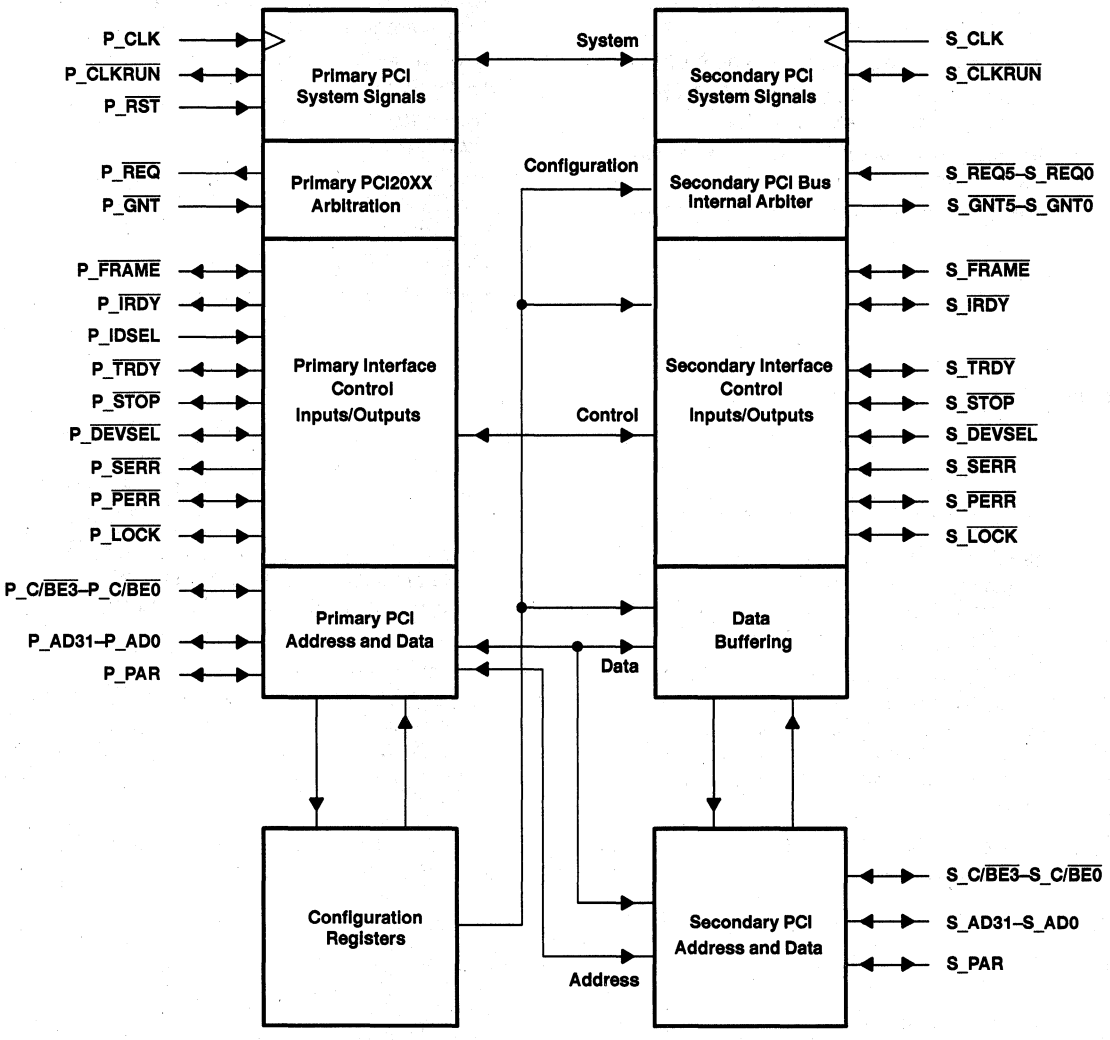
The PCI20XX is also compliant with the PCI-to-PCI bridge architecture specification 1.0 and implements many additional features that make it ideal for bridging two PCI buses. The PCI20XX can be configured for subtractive decoding, and negative decoding can be disabled on the secondary interface. Two extension windows are included for specialty decoding. The serial and parallel port addresses can be programmed for positive decoding on the primary interface. The PCI20XX implements many other features that add performance and flexibility.

An advanced CMOS process is utilized to achieve low system-power consumption while operating at PCI clock rates up to 33 MHz.

PRODUCT PREVIEW

Table of Contents

functional block diagram



PRODUCT PREVIEW

Table 1. Signal Names Sorted Alphabetically

SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.	SIGNAL NAME	NO.
GND		P_AD12		S_AD5		S_REQ0	
GND		P_AD13		S_AD6		S_REQ1	
GND		P_AD14		S_AD7		S_REQ2	
GND		P_AD15		S_AD8		S_REQ3	
GND		P_AD16		S_AD9		S_REQ4	
GND		P_AD17		S_AD10		S_REQ5	
GND		P_AD18		S_AD11		S_SERR	
GND		P_AD19		S_AD12		S_STOP	
GND		P_AD20		S_AD13		S_TRDY	
GND		P_AD21		S_AD14		VCC	
GND		P_AD22		S_AD15		VCC	
GND		P_AD23		S_AD16		VCC	
GND		P_AD24		S_AD17		VCC	
GND		P_AD25		S_AD18		VCC	
GND		P_AD26		S_AD19		VCC	
GND		P_AD27		S_AD20		VCC	
GND		P_AD28		S_AD21		VCC	
GND		P_AD29		S_AD22		VCC	
GND		P_AD30		S_AD23		VCC	
GND		P_AD31		S_AD24		VCC	
NC		P_C/BE0		S_AD25		VCC	
NC		P_C/BE1		S_AD26		VCC	
NC		P_C/BE2		S_AD27		VCC	
NC		P_C/BE3		S_AD28		VCC	
NC		P_CLK		S_AD29		VCC	
NC		P_CLKRUN		S_AD30		VCC	
NC		P_DEVSEL		S_AD31		VCC	
NC		P_FRAME		S_C/BE0		VCC	
NC		P_GNT		S_C/BE1		VCC	
NC		P_IDSEL		S_C/BE2		VCCP	
NC		P_IRDY		S_C/BE3		VCCP	
NC		P_LOCK		S_CLK		VCCP	
NC		P_PAR		S_CLKRUN		VCCP	
P_AD1		P_PERR		S_DEVSEL		VCCP	
P_AD2		P_REQ		S_FRAME		VCCP	
P_AD3		P_RST		S_GNT0		VCCS	
P_AD4		P_SERR		S_GNT1		VCCS	
P_AD5		P_STOP		S_GNT2		VCCS	
P_AD6		P_TRDY		S_GNT3		VCCS	
P_AD7		S_AD0		S_GNT4		VCCS	
P_AD8		S_AD1		S_GNT5		VCCS	
P_AD9		S_AD2		S_IRDY			
P_AD10		S_AD2		S_PAR			
P_AD11		S_AD4		S_PERR			

PRODUCT PREVIEW

Terminal Functions

primary PCI system

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
P_CLK			Primary PCI bus clock. P_CLK provides timing for all transactions on the primary PCI bus. All primary PCI signals are sampled at rising edge of P_CLK.
P_$\overline{\text{RST}}$			PCI reset. Assertion on the P_$\overline{\text{RST}}$ signal causes the PCI20XX to place all output buffers in the high-impedance state and reset all internal registers. When asserted, the device is completely nonfunctional. After deasserting P_$\overline{\text{RST}}$, the PCI20XX returns to its default state.

primary PCI address and data

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
P_AD31–P_AD0			Primary address/data bus. P_AD31–P_AD0 comprise the multiplexed PCI address and data bus on the primary interface. During the address phase of a primary bus PCI cycle, P_AD31–P_AD0 contain a 32-bit address or other destination information. During the data phase, P_AD31–P_AD0 contain data.
P_C/$\overline{\text{BE3}}$ P_C/$\overline{\text{BE2}}$ P_C/$\overline{\text{BE1}}$ P_C/$\overline{\text{BE0}}$			Primary bus commands and byte enables. P_C/$\overline{\text{BE3}}$–P_C/$\overline{\text{BE0}}$ are multiplexed on the same PCI terminals. During the address phase of a primary bus PCI cycle, P_C/$\overline{\text{BE3}}$–P_C/$\overline{\text{BE0}}$ 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. P_C/$\overline{\text{BE0}}$ applies to byte 0 (P_AD7–P_AD0).
P_PAR			Primary parity. In all primary bus read and write cycles, the PCI20XX calculates even parity across the P_AD and P_C/$\overline{\text{BE}}$ buses. As an initiator during PCI write cycles, the PCI20XX outputs P_PAR with a one P_CLK delay. As a target during PCI read cycles, the calculated parity is compared to the initiator's parity indicator. A miscompare can result in a parity error assertion (P_$\overline{\text{PERR}}$)

power supply

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
GND			Device ground terminals
V_{CC}			Power-supply terminal for core logic
V_{CCP}			Primary bus signaling environment supply. This power-supply input is used in protection circuitry on primary bus I/O signals.
V_{CCS}			Secondary bus signaling environment supply. This power-supply input is used in protection circuitry on secondary bus I/O signals.

primary PCI interface control

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
P_$\overline{\text{DEVSEL}}$			Primary device select. The PCI20XX asserts P_$\overline{\text{DEVSEL}}$ to claim a PCI cycle as the target device. As a PCI initiator on the primary bus, the PCI20XX monitors P_$\overline{\text{DEVSEL}}$ until a target responds. If no target responds before timeout occurs, then the PCI20XX terminates the cycle with an initiator abort.
P_$\overline{\text{FRAME}}$			Primary cycle frame. P_$\overline{\text{FRAME}}$ is driven by the initiator of a primary bus cycle. P_$\overline{\text{FRAME}}$ is asserted to indicate that a bus transaction is beginning, and data transfers continue while P_$\overline{\text{FRAME}}$ is asserted. When P_$\overline{\text{FRAME}}$ is deasserted, the primary bus transaction is in the final data phase.
P_$\overline{\text{GNT}}$			Primary bus grant to PCI20XX. P_$\overline{\text{GNT}}$ is driven by the primary PCI bus arbiter to grant the PCI20XX access to the primary PCI bus after the current data transaction has completed. P_$\overline{\text{GNT}}$ follows a primary bus request depending upon the primary bus parking algorithm.
P_IDSEL			Initialization device select. P_IDSEL selects the PCI20XX during configuration space accesses. P_IDSEL can be connected to one of the upper 24 PCI address lines on the primary PCI bus. There is no IDSEL signal interfacing the secondary PCI bus; thus, the entire configuration space of the PCI20XX can be accessed only from the primary bus.

Terminal Functions (Continued)

primary PCI interface control (continued)

TERMINAL NAME	NO.	I/O TYPE	FUNCTION
P_$\overline{\text{IRDY}}$			Primary initiator ready. P_$\overline{\text{IRDY}}$ indicates the primary bus initiator's ability to complete the current data phase of the transaction. A data phase is completed upon a rising edge of P_CLK where both P_$\overline{\text{IRDY}}$ and P_$\overline{\text{TRDY}}$ are asserted until wait states are inserted.
P_$\overline{\text{LOCK}}$			Primary lock. P_$\overline{\text{LOCK}}$ is used to lock the primary bus and gain exclusive access as an initiator.
P_$\overline{\text{PERR}}$			Primary parity-error indicator. P_$\overline{\text{PERR}}$ is driven by a primary bus PCI device to indicate that calculated parity does not match P_PAR when enabled through the command register.
P_$\overline{\text{REQ}}$			Primary PCI bus request. P_$\overline{\text{REQ}}$ is asserted by the PCI20XX to request access to the primary PCI bus as an initiator.
P_$\overline{\text{STOP}}$			Primary cycle stop signal. P_$\overline{\text{STOP}}$ is driven by a PCI target to request the initiator to stop the current primary bus transaction. P_$\overline{\text{STOP}}$ is used for target disconnects and is commonly asserted by target devices that do not support burst data transfers.
P_$\overline{\text{SERR}}$			Primary system error. If enabled through the command register, P_$\overline{\text{SERR}}$ is pulsed from the PCI20XX to indicate that an address parity error has occurred. The PCI20XX need not be the target of the primary PCI cycle to assert this signal. If enabled through the bridge control register, P_$\overline{\text{SERR}}$ pulses to indicate that an address parity error occurred on one of the subordinate buses downstream from the PCI20XX.
P_$\overline{\text{TRDY}}$			Primary target ready. P_$\overline{\text{TRDY}}$ indicates the primary bus target's ability to complete the current data phase of the transaction. A data phase is completed upon a rising edge of P_CLK where both P_$\overline{\text{IRDY}}$ and P_$\overline{\text{TRDY}}$ are asserted until wait states are inserted.

secondary PCI system

TERMINAL NAME	NO.	I/O TYPE	FUNCTION
S_CLK			Secondary PCI bus clock. S_CLK provides timing for all transactions on the secondary PCI bus. All secondary PCI signals are sampled at rising edge of S_CLK .
S_$\overline{\text{CLKRUN}}$			Secondary PCI bus clock run. S_$\overline{\text{CLKRUN}}$ is output by the PCI20XX to indicate that the S_CLK frequency is decreased and is driven by secondary bus PCI devices to request an increase in the S_CLK frequency.

secondary PCI address and data

TERMINAL NAME	NO.	I/O TYPE	FUNCTION
S_AD31–S_AD0			Secondary address/data bus. S_AD31–S_AD0 comprise the multiplexed PCI address and data bus on the secondary interface. During the address phase of a secondary bus PCI cycle, S_AD31–S_AD0 contain a 32-bit address or other destination information. During the data phase, S_AD31–S_AD0 contain data.
S_C/$\overline{\text{BE3}}$ S_C/$\overline{\text{BE2}}$ S_C/$\overline{\text{BE1}}$ S_C/$\overline{\text{BE0}}$			Secondary bus commands and byte enables. S_C/$\overline{\text{BE3}}$–S_C/$\overline{\text{BE0}}$ are multiplexed on the same PCI terminals. During the address phase of a secondary bus PCI cycle, S_C/$\overline{\text{BE3}}$–S_C/$\overline{\text{BE0}}$ 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. S_C/$\overline{\text{BE0}}$ applies to byte 0 (S_AD7–S_AD0).
S_PAR			Secondary parity. In all secondary bus read and write cycles, the PCI20XX calculates even parity across the S_AD and S_C/$\overline{\text{BE}}$ buses. As an initiator during PCI write cycles, the PCI20XX outputs S_PAR with a one S_CLK delay. As a target during PCI read cycles, the calculated parity is compared to the initiator's parity indicator. A miscompare can result in a parity-error assertion (S_$\overline{\text{PERR}}$).

Terminal Functions (Continued)

secondary PCI interface control

TERMINAL NAME	TERMINAL NO.	I/O TYPE	FUNCTION
S_DEVSEL			Secondary device select. The PCI20XX asserts S_DEVSEL to claim a PCI cycle as the target device. As a PCI initiator on the secondary bus, the PCI20XX monitors S_DEVSEL until a target responds. If no target responds before timeout occurs, then the PCI20XX terminates the cycle with an initiator abort.
S_FRAME			Secondary cycle frame. S_FRAME is driven by the initiator of a secondary bus cycle. S_FRAME is asserted to indicate that a bus transaction is beginning, and data transfers continue while S_FRAME is asserted. When S_FRAME is deasserted, the secondary bus transaction is in the final data phase.
S_GNT5–S_GNT0			Secondary bus grant signals. The PCI20XX provides internal arbitration. S_GNT5–S_GNT0 are used to grant potential secondary PCI bus masters access to the bus. A total of seven potential initiators (including the PCI20XX) can be located on the secondary PCI bus.
S_IRDY			Secondary initiator ready. S_IRDY indicates the secondary bus initiator's ability to complete the current data phase of the transaction. A data phase is completed upon a rising edge of S_CLK where both S_IRDY and S_TRDY are asserted until wait states are inserted.
S_LOCK			Secondary lock. S_LOCK is used to lock the secondary bus and gain exclusive access as an initiator.
S_PERR			Secondary parity error indicator. S_PERR is driven by a secondary bus PCI device to indicate that calculated parity does not match S_PAR when enabled through the bridge control register.
S_REQ5–S_REQ0			Secondary bus request signals. The PCI20XX provides internal arbitration. S_REQ5–S_REQ0 are used as inputs from secondary PCI bus initiators requesting the bus. A total of seven potential initiators (including the PCI20XX) can be located on the secondary PCI bus.
S_SERR			Secondary system error. S_SERR passes through to the primary interface by the PCI20XX if enabled through the bridge control register. S_SERR is never asserted by the PCI20XX.
S_STOP			Secondary cycle stop signal. S_STOP is driven by a PCI target to request the initiator to stop the current secondary bus transaction. S_STOP is used for target disconnects and is commonly asserted by target devices that do not support burst data transfers.
S_TRDY			Secondary target ready. S_TRDY indicates the secondary bus target's ability to complete the current data phase of the transaction. A data phase is completed upon a rising edge of S_CLK where both S_IRDY and S_TRDY are asserted until wait states are inserted.

architecture

This section provides an overview of the PCI20XX PCI-to-PCI bridge functionality and its applications.

Introduction to the PCI20XX

The PCI20XX is a bridge between two PCI buses and is compliant with both the PCI local bus specification revision 2.1 and the PCI-to-PCI bridge architecture specification revision 1.0. The PCI20XX supports two 32-bit PCI buses operating at a maximum of 33 MHz. The primary and secondary buses can operate independently in either a 3.3-V or 5-V signaling environment. The core logic of the PCI20XX, however, is powered at 3.3 V to reduce power consumption.

Host software interacts with the PCI20XX through internal registers that provide the PCI standard status and control for both the primary and secondary buses. There are many vendor-specific features included in the PCI20XX that exist in the TI extension register set. The PCI configuration header of the PCI20XX is accessible only from the primary PCI interface.

The PCI20XX provides internal arbitration for the seven possible secondary bus masters and provides each with a dedicated $\overline{\text{REQ}}/\overline{\text{GNT}}$ pair. The arbiter features a two-tier rotational scheme with the PCI20XX defaulting to the highest priority tier. The bus parking scheme is also configurable and can be set to park $\overline{\text{GNT}}$ either on the bridge or on the last mastering device.

Upon system power up, host software configures the PCI20XX according to the devices that exist on subordinate buses and enables performance-enhancing features of the PCI20XX. In a typical system, this is the only communication with the bridge's internal register set.

PCI commands

The PCI20XX responds as a PCI target device to PCI bus cycles based on the decoding of each address phase and internal register settings. Table 2 lists the valid PCI bus cycles and their encoding on the C/$\overline{\text{BE}}$ bus during the address phase of a bus cycle.

Table 2. PCI Command Definition

C/$\overline{\text{BE}}$3–C/$\overline{\text{BE}}$0	COMMAND
.0000	Interrupt acknowledge
0001	Special cycle
0010	I/O read
0011	I/O write
0100	Reserved
0101	Reserved
0110	Memory read
0111	Memory write
1000	Reserved
1001	Reserved
1010	Configuration read
1011	Configuration write
1100	Memory read multiple
1101	Dual address cycle
1110	Memory read line
1111	Memory write and invalidate

PCI commands (continued)

The PCI20XX never responds as a PCI target to the interrupt acknowledge, special cycle, or reserved commands. The PCI20XX does, however, initiate special cycles on the secondary interface when a type 1 configuration cycle issues the special cycle request feature. The remaining PCI commands address either memory, I/O, or configuration space. The PCI20XX accepts PCI cycles by asserting $\overline{\text{DEVSEL}}$ as a medium-speed device; that is, $\overline{\text{DEVSEL}}$ is asserted two clock cycles after the address phase.

configuration cycles

The PCI local bus specification 2.1 defines two types of PCI configuration read and write cycles: type 0 and type 1. The PCI20XX decodes each type differently. Type 0 configuration cycles are intended for devices on the primary bus, while type 1 configuration cycles are intended for devices at a subordinate bus. The difference between these two types of cycles is the encoding of the PCI P_AD bus during the address phase of the cycle. The P_AD bus encoding during the address phase of a type 0 configuration cycle is shown in Figure 1. The 6-bit register number field represents an 8-bit address with the two lower bits masked to 0, indicating a double-word boundary. This results in a 256-byte configuration address space, per function, per device. Individual byte accesses can be selected within a double word by using the P_C/$\overline{\text{BE}}$ signals during the data phase of the cycle.

31 11	10 8	7 2	1	0
Reserved	Function number	Function number	0	0

Figure 1. PCI P_AD31–P_AD0 During Address Phase of a Type 0 Configuration Cycle

The PCI20XX claims type 0 configuration cycles only when its P_IDSEL terminal is asserted during the address phase of the cycle and the PCI function number encoded in the cycle is 0. If the function number is 1 or greater, the PCI20XX does not recognize the configuration command. The PCI20XX services valid type 0 configuration read or write cycles by accessing internal registers from the configuration header (see *PCI20XX configuration header*).

Because type 1 configuration cycles are issued to devices on subordinate buses, the PCI20XX claims type 1 cycles based on the bus number of the destination bus. The P_AD bus encoding during the address phase of a type 1 cycle is shown in Figure 2. Device number and bus number fields are also defined.

31 24	23 16	15 11	10 8	7 2	1	0
Reserved	Bus number	Device number	Function number	Register number	0	1

Figure 2. PCI P_AD31–P_AD31 During Address Phase of a Type 1 Configuration Cycle

Several PCI20XX configuration registers in Table 3 are significant when decoding and claiming type 1 configuration cycles. The destination bus number encoded on the P_AD bus is compared to the values programmed in the PCI20XX configuration registers 18h, 19h, and 1Ah, which are the primary bus number, secondary bus number, and subordinate bus number registers, respectively. These registers default to 00h and are programmed by host software to reflect the bus hierarchy in the system (see Figure 3 for an example of a system bus hierarchy and how the PCI20XX registers 18h, 19h, and 1Ah are programmed in this case).

configuration cycles (continued)

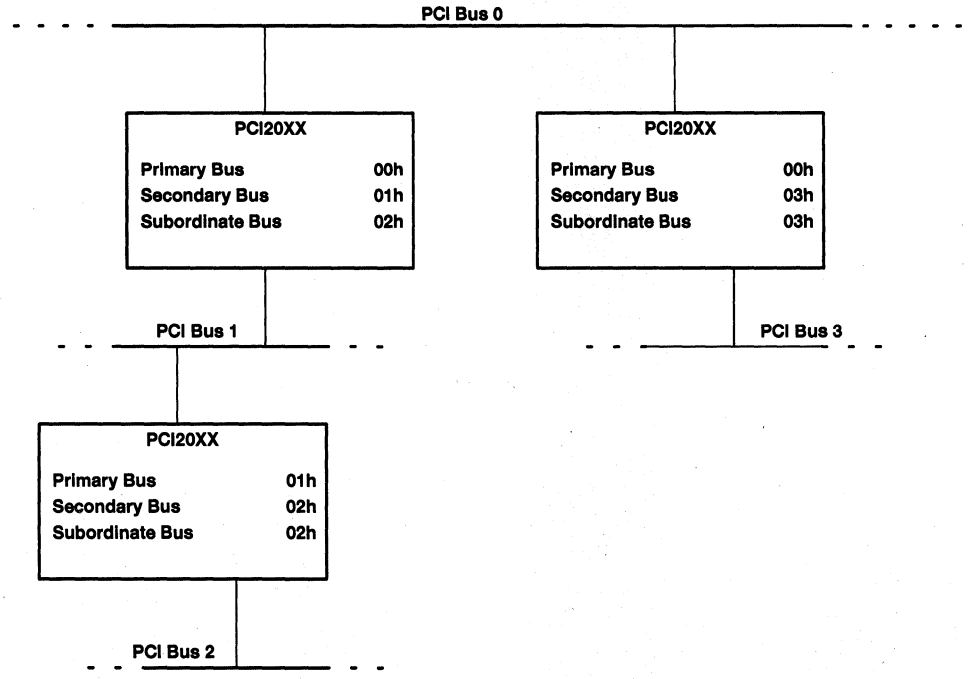


Figure 3. Bus Hierarchy and Numbering

special cycle generation

The PCI20XX is designed to generate special cycles on the secondary bus through a type 1 cycle conversion. If the following conditions exist during a type 1 configuration cycle, the PCI20XX generates a special cycle on the secondary bus with a message that matches the type 1 configuration cycle data: the bus number matches the PCI20XX secondary bus number, the device is programmed as 1Fh, and the function is programmed 7h. If the bus number is a subordinate bus and not the secondary, then the PCI20XX passes the type 1 special cycle request through to the secondary interface along with the proper message.

Special cycles are passed through only to subordinate buses; that is, a device on a subordinate bus cannot generate configuration cycles directed upstream and messages cannot be passed via special cycles upstream through the PCI20XX.

compliance to PCI local bus specification revision 2.1

The most significant additions to the PCI local bus specification revision 2.1 are the latency requirements on PCI peripherals. Minimum response times are specified for a PCI device to respond with valid data. These requirements are intended to improve throughput and reduce latencies on the PCI bus. The PCI20XX is fully compliant with these guidelines.

Other additions to revision 2.1 of the PCI specification include the subsystem ID and subsystem vendor ID registers in the PCI configuration header. The PCI20XX also includes these features.

typical applications

Figure 4 shows two typical applications for the PCI20XX PCI-to-PCI bridge. A system that requires more than ten PCI loads requires a PCI-to-PCI bridge to overcome the electrical loading limits in the PCI local bus specification revision 2.1. Since option card slots require two loads each, it is clear that bridging is necessary on large and expandable systems. Option cards using more than one PCI device require a PCI-to-PCI bridge to limit the load to the option slot.

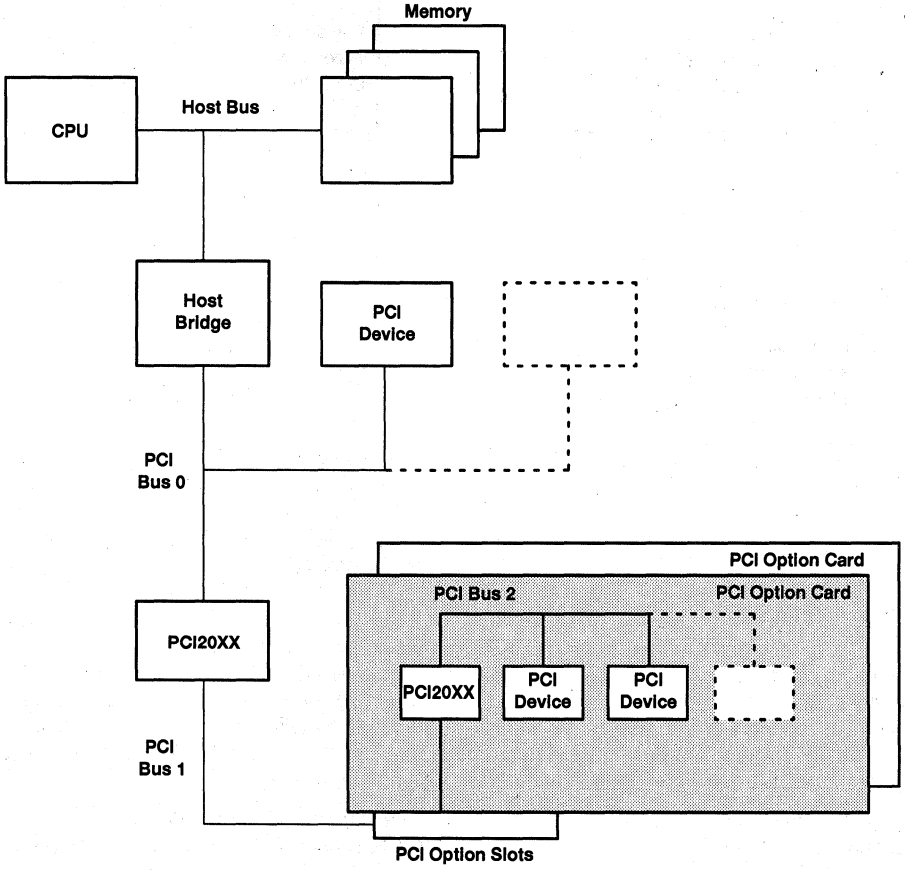


Figure 4. Typical PCI20XX Applications

PCI configuration headers

The PCI20XX is a single-function PCI device. The configuration header is in compliance with the PCI-to-PCI bridge architecture specification, revision 1.0. Table 3 shows the PCI configuration headers, which includes the predefined portion of the bridge configuration space.

Table 3. PCI Configuration Headers

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
Internal base address register 0				10h
Internal base address register 1				14h
Secondary bus latency timer†	Subordinate bus number†	Secondary bus number†	Primary bus number†	18h
Secondary status register†		I/O limit†	I/O base†	1Ch
Memory limit†		Memory base†		20h
Prefetchable memory limit†		Prefetchable memory base†		24h
Prefetchable base upper 32 bits				28h
Prefetchable limit upper 32 bits				2Ch
I/O limit upper 16 bits†		I/O base upper 16 bits†		30h
Reserved				34h
Expansion ROM base address				38h
Bridge control †		Interrupt pin†	Interrupt line†	3Ch
Subsystem ID		Subsystem vendor ID		40h
Extension window base 0				44h
Extension window limit 0				48h
Extension window base 1				4Ch
Extension window limit 1				50h
Reserved				54h
Reserved				58h
Reserved				5Ch
Reserved				60h
Primary decode	Secondary decode	Window map	Window control	64h
Reserved	Diagnostic reg	Reserved	Port enable reg	68h
Clock control	Arbitration cfg	Buffer control	Retry status	6Ch
Reserved				70h-FFh

† Used by the PCI20XX

PRODUCT PREVIEW

vendor ID

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 16-bit register contains a value allocated by the PCI SIG (special interest group) and identifies TI as the manufacturer of this device. The vendor ID assigned to TI is 104Ch.

device ID

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	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

Register: **Device ID**
Type: Read Only
Offset: 02h
Default: UUUUh
Description: This 16-bit register contains a value allocated by the vendor and identifies TI as the manufacturer of this device. The device ID for the PCI20XX is UUUUh.

TI extension registers

The TI extension registers lie outside the standard PCI-to-PCI bridge device configuration space (i.e., registers 40h-FFh in PCI configuration space in the PCI20XX). These registers can be accessed through configuration reads and writes and I/O space if the internal base address register (10h) is programmed and I/O space response is enabled in the command register (04h). The TI extension registers add flexibility and performance benefits to the standard PCI-to-PCI bridge, and can be accessed from the secondary PCI bus through I/O reads and writes.

Contents

Electrical characteristics presented in this data book, unless otherwise noted, apply for the circuit type(s) listed in the page heading regardless of package. The availability of a circuit function in a particular package is denoted by an alphabetical reference above the pin-connection diagram(s). These alphabetical references refer to mechanical outline drawings shown in this section.

Factory orders for circuits described in this catalog should include a three-part type number as explained in the following example.

EXAMPLE: PCI 1130 PDV

Prefix

PCI = Standard prefix

Unique Circuit Description

MUST CONTAIN FOUR TO TWELVE CHARACTERS

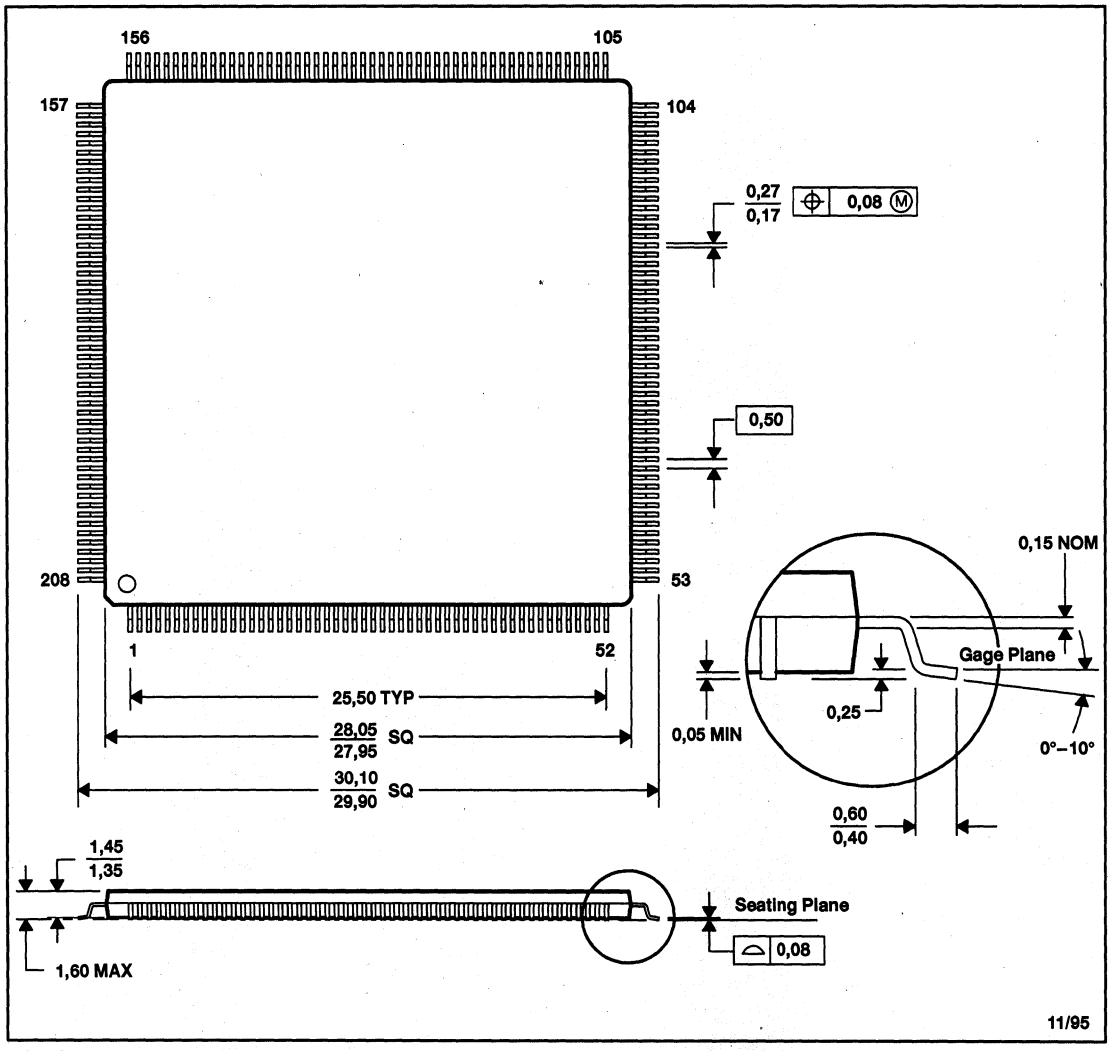
Examples: 1130
1050
10XX

Package

MUST CONTAIN ONE TO THREE LETTERS

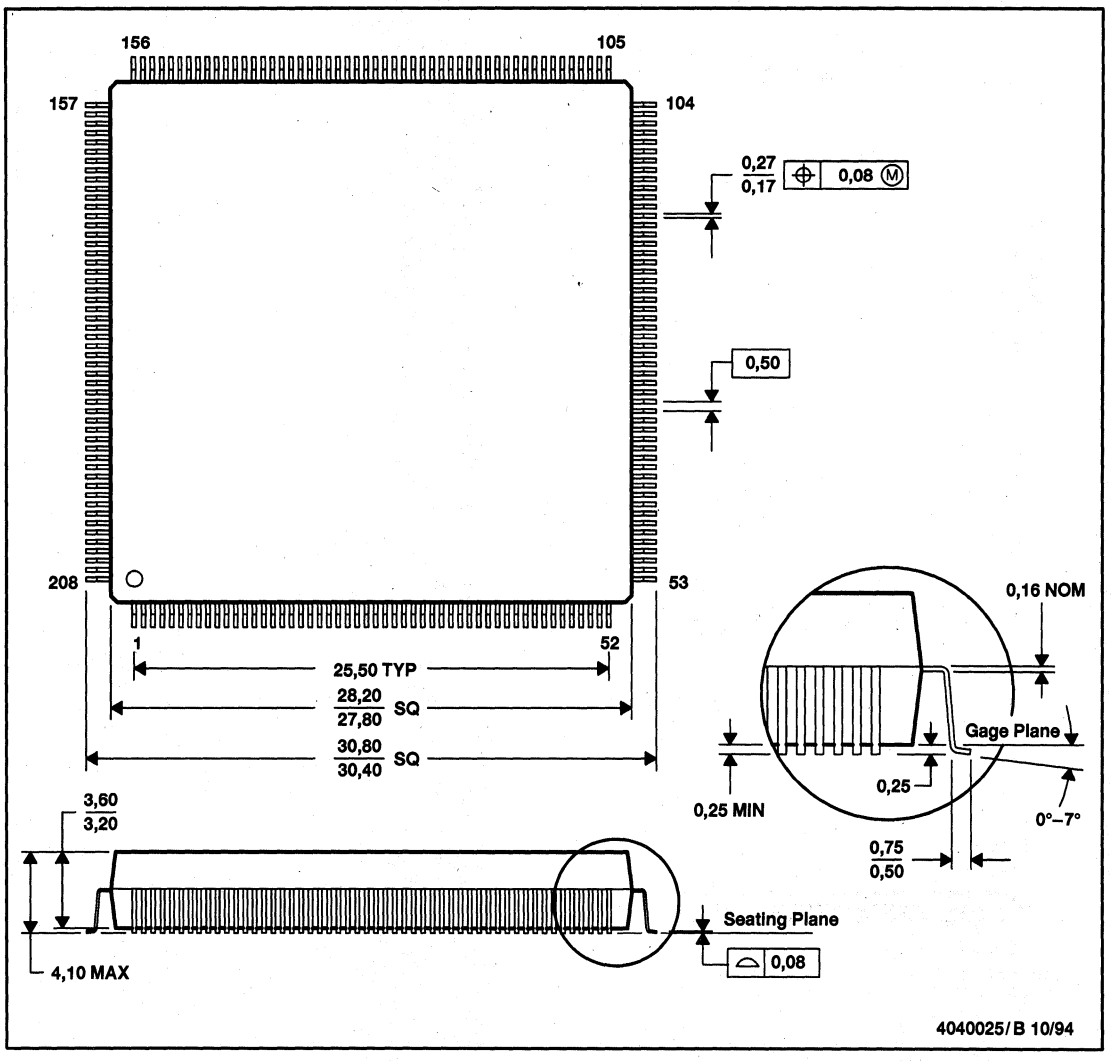
PDV = thin plastic quad flatpack
PPM = plastic quad flatpack

PDV (S–TQFP–G208) **THIN PLASTIC QUAD FLATPACK**



NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Falls within JEDEC MO-143

PPM (S-PQFP-G208) **PLASTIC QUAD FLATPACK**



NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Falls within JEDEC MO-143

NOTES

TI Worldwide Sales and Representative Offices

AUSTRALIA / NEW ZEALAND: Texas Instruments Australia Ltd.: Melbourne [61] 3-696-1211, Fax 3-696-4446; **Sydney** 2-910-3100, Fax 2-805-1186.
BELGIUM: Texas Instruments Belgium S.A./N.V.: Brussels [32] (02) 726 75 80, Fax (02) 726 72 76.
BRAZIL: Texas Instrumentos Electronicos do Brasil Ltda.: Sao Paulo [55] 11-535-5133.
CANADA: Texas Instruments Canada Ltd.: Montreal (514) 421-2750; **Ottawa** (613) 726-3201; Fax 726-6363; **Toronto** (905) 884-9181; Fax 884-0062.
DENMARK: Texas Instruments A/S: Ballerup [45] (44) 68 74 00.
FRANCE/MIDDLE EAST/AFRICA: Texas Instruments France: Velizy-Villacoublay [33] (1) 30 70 10 01, Fax (1) 30 70 10 54.
GERMANY: Texas Instruments Deutschland GmbH.: Freising [49] (08161) 800, Fax (08161) 80 45 16; **Hannover** (0511) 90 49 60, Fax (0511) 64 90 331; **Ostfildern** (0711) 340 30, Fax (0711) 340 32 57.
HONG KONG: Texas Instruments Hong Kong Ltd.: Kowloon [852] 2956-7288, Fax 2956-2200.
HUNGARY: Texas Instruments Representation: Budapest [36] (1) 269 83 10, Fax (1) 267 13 57.
IRELAND: Texas Instruments Ireland Ltd.: Dublin [353] (01) 475 52 33, Fax (01) 478 14 63.
ITALY: Texas Instruments Italia S.p.A.: Agrate Brianza [39] (039) 684 21, Fax (039) 684 29 12; **Rome** (06) 657 26 51.
JAPAN: Texas Instruments Japan Ltd.: Kanazawa [81] 0762-23-5471, Fax 0762-23-1583; **Kita Kanto** 0485-22-2440, Fax 0485-23-5787; **Kyoto** 075-341-7713, Fax 075-341-7724; **Kyushu** 0977-73-1557, Fax 0977-73-1583; **Matsumoto** 0263-33-1060, Fax 0263-35-1025; **Nagoya** 052-232-5601, Fax 052-232-7888; **Osaka** 06-204-1881, Fax 06-204-1895; **Tachikawa** 0425-27-6760, Fax 0425-27-6426; **Tokyo** 03-3769-8700, Fax 03-3457-6777; **Yokohama** 045-338-1220, Fax 045-338-1255.
KOREA: Texas Instruments Korea Ltd.: Seoul [82] 2-551-2804, Fax 2-551-2828.
MAINLAND CHINA: Texas Instruments China Inc.: Beijing [86] 10-500-2255, Ext. 3750, Fax 10-500-2705.
MALAYSIA: Texas Instruments Malaysia Sdn Bhd: Kuala Lumpur [60] 3-208-6001, Fax 3-230-6605.
MEXICO: Texas Instruments de Mexico S.A. de C.V.: Colonia del Valle [52] 5-639-9740.
NORWAY: Texas Instruments Norge A/S: Oslo [47] (02) 264 75 70.
PHILIPPINES: Texas Instruments Asia Ltd.: Metro Manila [63] 2-636-0980, Fax 2-631-7702.
SINGAPORE (& INDIA, INDONESIA, THAILAND): Texas Instruments Singapore (PTE) Ltd.: Singapore [65] 390-7100, Fax 390-7062.
SPAIN/PORTUGAL: Texas Instruments España S.A.: Madrid [34] (1) 372 80 51, Fax (1) 307 68 64.
SUOMI/FINLAND: Texas Instruments/OY: Espoo [358] (0) 43 54 20 33, Fax (0) 46 73 23.
SWEDEN: Texas Instruments International Trade Corporation (Sverigefilialen): Kista [46] (08) 752 58 00, Fax (08) 751 97 15.
SWITZERLAND: Texas Instruments Switzerland AG: Dietikon [41] 886-2-3771450.
TAIWAN: Texas Instruments Taiwan Limited: Taipei [886] 2-378-6800, Fax 2-377-2718.
THE NETHERLANDS: Texas Instruments Holland, B.V. Amsterdam [31] (020) 546 98 00, Fax (020) 646 31 36.
UNITED KINGDOM: Texas Instruments Ltd.: Northampton [44] (01604) 66 30 00, Fax (01604) 66 30 01.

UNITED STATES: Texas Instruments Incorporated: ALABAMA: Huntsville (205) 430-0114; **ARIZONA: Phoenix** (602) 224-7800; **CALIFORNIA: Irvine** (714) 660-1200; **Los Angeles** (818) 704-8100; **San Diego** (619) 278-9600; **San Jose** (408) 894-9000; **COLORADO: Denver** (303) 488-9300; **CONNECTICUT: Wallingford** (203) 269-0074; **FLORIDA: Fort Lauderdale** (214) 644-5580 **Orlando** (407) 667-5308; **Tampa** (813) 573-0331; **GEORGIA: Atlanta** (770) 662-7967; **ILLINOIS: Chicago** (708) 517-4500; **INDIANA: Indianapolis** (317) 573-6400; **KANSAS: Kansas City** (913) 451-4511; **MARYLAND: Baltimore** (410) 312-7900; **MASSACHUSETTS: Boston** (617) 895-9100; **MICHIGAN: Detroit** (810) 305-5700; **MINNESOTA: Minneapolis** (612) 828-9300; **NEW JERSEY: Edison** (908) 906-0033; **NEW MEXICO: Albuquerque** (505) 345-2555; **NEW YORK: Long Island** (516) 454-6600; **Poughkeepsie** (914) 897-2900; **Rochester** (716) 385-6770; **NORTH CAROLINA: Charlotte** (704) 522-5487; **Raleigh** (919) 876-2725; **OHIO: Cleveland** (216) 328-2149; **Dayton** (513) 427-6200; **OREGON: Portland** (503) 643-6758; **PENNSYLVANIA: Philadelphia** (610) 825-9500; **PUERTO RICO: Hato Rey** (809) 753-8700; **TEXAS: Austin** (512) 250-6769; **Dallas** (214) 917-1264; **Houston** (713) 778-6592; **Midland** (915)561-6521; **WISCONSIN: Milwaukee** (414) 798-5021.

North American Authorized Distributors

COMMERCIAL	
Almac / Arrow	800-426-1410 / 800-452-9185 Oregon only
Anthem Electronics	800-826-8436
Arrow / Schweber	800-777-2776
Future Electronics (Canada)	800-388-8731
Hamilton Hallmark	800-332-8638
Marshall Industries	800-522-0084 or www.marshall.com
Wyle	800-414-4144
OBSOLETE PRODUCTS	
Rochester Electronics	508-462-9332
MILITARY	
Alliance Electronics Inc	800-608-9494
Future Electronics (Canada)	800-388-8731
Hamilton Hallmark	800-332-8638
Zeus, An Arrow Company	800-524-4735
TI DIE PROCESSORS	
Chip Supply	(407) 298-7100
Elmo Semiconductor	(818) 768-7400
Minco Technology Labs	(512) 834-2022
CATALOG	
Allied Electronics	800-433-5700
Arrow Advantage	800-777-2776
Newark Electonics	800-367-3573

For Distributors outside North America, contact your local Sales Office.

A022396

Printed in the USA

TEXAS INSTRUMENTS

Printed in U.S.A.
0396–CP

SCPD001