O$_2$Micro
Breathing Life into Computing

Preliminary

OZ6812

Designed for
Microsoft®
Windows NT®
Windows®98

Challenger ACPI CardBus Controller

FEATURES

- ACPI-PCI Bus Power Management Interface Specification Rev1.0 Compliant
- Supports OnNow LAN wakeup, OnNow Ring Indicate, PCI CLKRUN#, PME#, and CardBus CCLKRUN#
- Compliant with PCI specification v2.1S, 1998 PC Card Standard 6.1 and JEIDA 4.1
- Yenta™ PCI to PCMCIA CardBus Bridge register compatible
- ExCA (Exchangeable Card Architecture) compatible registers mappable in memory and I/O space
- Intel^{TM} 82365SL PCIC Register Compatible
- Supports PCMCIA_ATA Specification
- Supports 5V/3.3V PC Cards and 3.3V Cardbus cards
- Supports single PC Card or CardBus slot with hot insertion and removal
- Supports multiple FIFOs for PCI/CardBus data transfer
- Supports Direct Memory Access for PC/PCI and PC/Way on PC Card socket
- Programmable interrupt protocol: PCI, PCI+ISA, PCI/Way, or PC/PCI interrupt signaling modes
- Win'98 IRQ and PC-97/98 compliant
- Parallel or Serial interface for socket power control devices (Micrel or standard)
- Zoomed Video Support
- Integrated PC 98 -Subsystem Vendor ID support, with auto lock bit
- LED Activity Pins

ORDERING INFORMATION

OZ6812T - 144pin LQFP
OZ6812B - 144pin Mini-BGA

GENERAL DESCRIPTION

The OZ6812 Challenger is an ACPI/PC98 ready, high performance, single slot PC Card controller with a synchronous 32-bit bus master/target PCI interface. This PC Card to PCI bridge host controller is compliant with the 1997 PC Card Standard. This standard incorporates the new 32-bit CardBus while retaining the 16-bit PC Card specification as defined by PCMCIA release 2.1. CardBus is intended to support "temporal" add-in functions on PC Cards, such as Memory cards, Network interfaces, FAX/Modems and other wireless communication cards, etc. The high performance and capability of the CardBus interface will enable the new development of many new functions and applications.

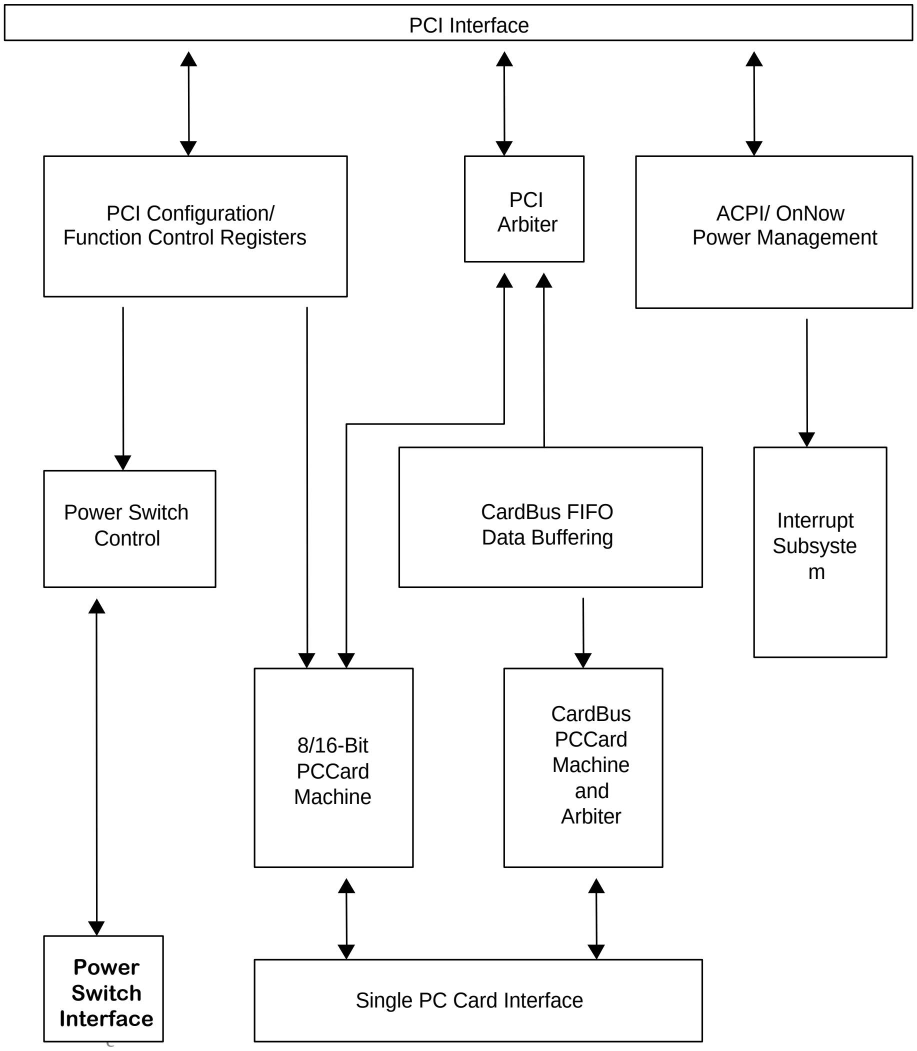
The OZ6812 CardBus controller is compliant with the latest ACPI-PCI Bus Power Management Interface Specification. It supports all four power states and the PME# function for maximum power savings and ACPI compliance. The device also provides a power-down mode to allow host software to reduce power consumption further by stopping internal clock distribution as well as the PC Card socket clock. In addition, an advanced CMOS process is utilized to minimize system power consumption.

The OZ6812 single PCMCIA socket supports a mix and match 3.3V/5V 8/16-bit PC Card R2 card or 32-bit CardBus R3 card. The R2 card support is compatible to the Intel 82365SL PCIC controller, and the R3 card support is fully compliant with the 1997 PC Card Standard CardBus specification. The OZ6812 is a stand alone device, which means that it does not require an additional buffer chip for the PC Card socket interface. In addition, the OZ6812 supports dynamic PC Card hot insertion and removal, with auto configuration capabilities.

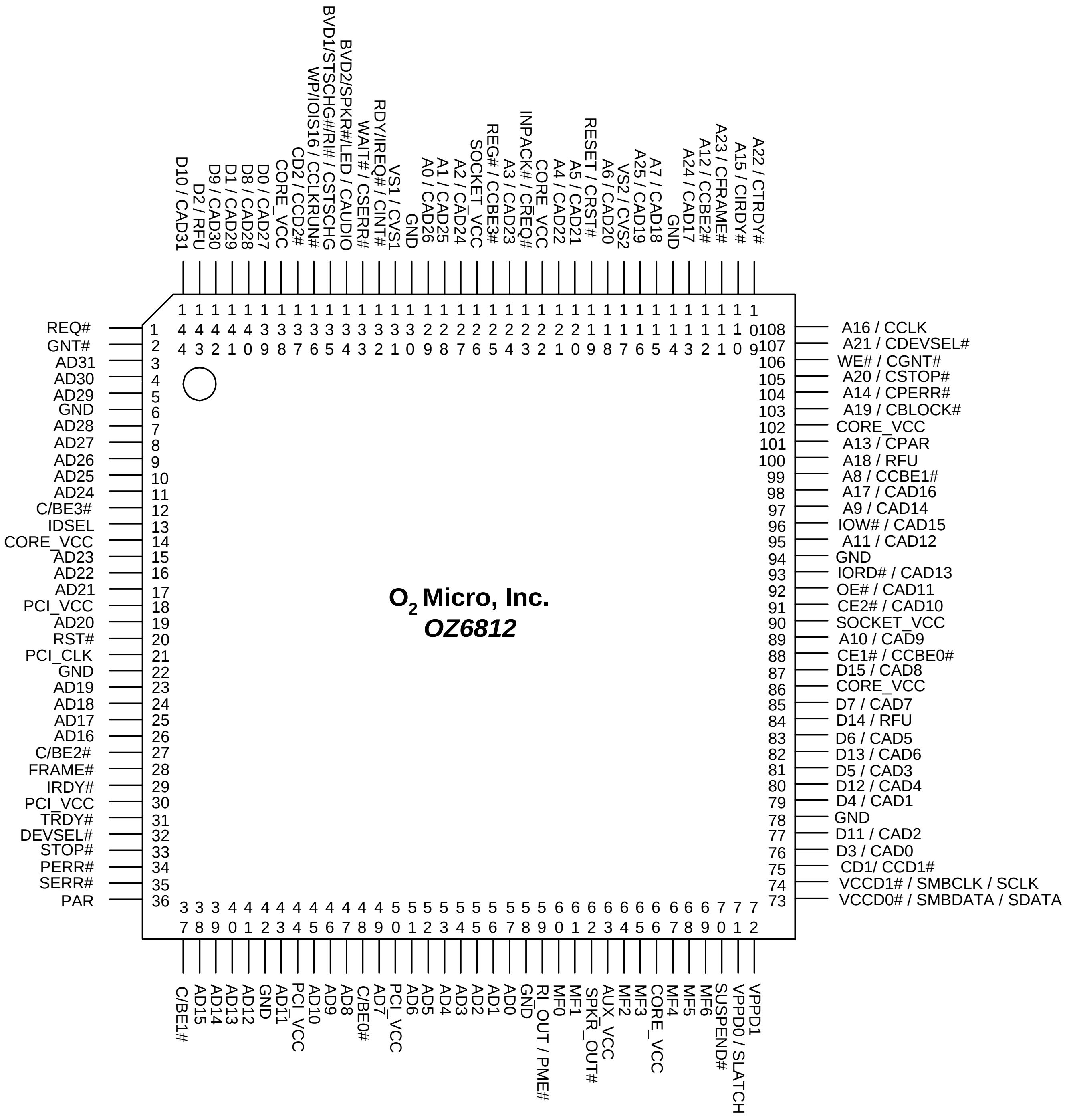
The OZ6812 is fully compliant with the 33Mhz PCI Bus specification, V2.1S. It supports a master device with inter CardBus direct data transfer. The OZ6812 implements a FIFO data buffer architecture between the PCI bus and CardBus socket interface to enhance data transfers to CardBus devices. The bi-directional FIFO buffer (composed of 16 double words) permits the OZ6812 to accept data from a target bus (PCI or CardBus interface) while simultaneously transferring data. This architecture not only speeds up data transfers but also prevents system deadlocks.

The OZ6812 is a PCMCIA R2/CardBus controller, providing the most advanced design flexibility for PC Cards which interface to advanced notebook designs.

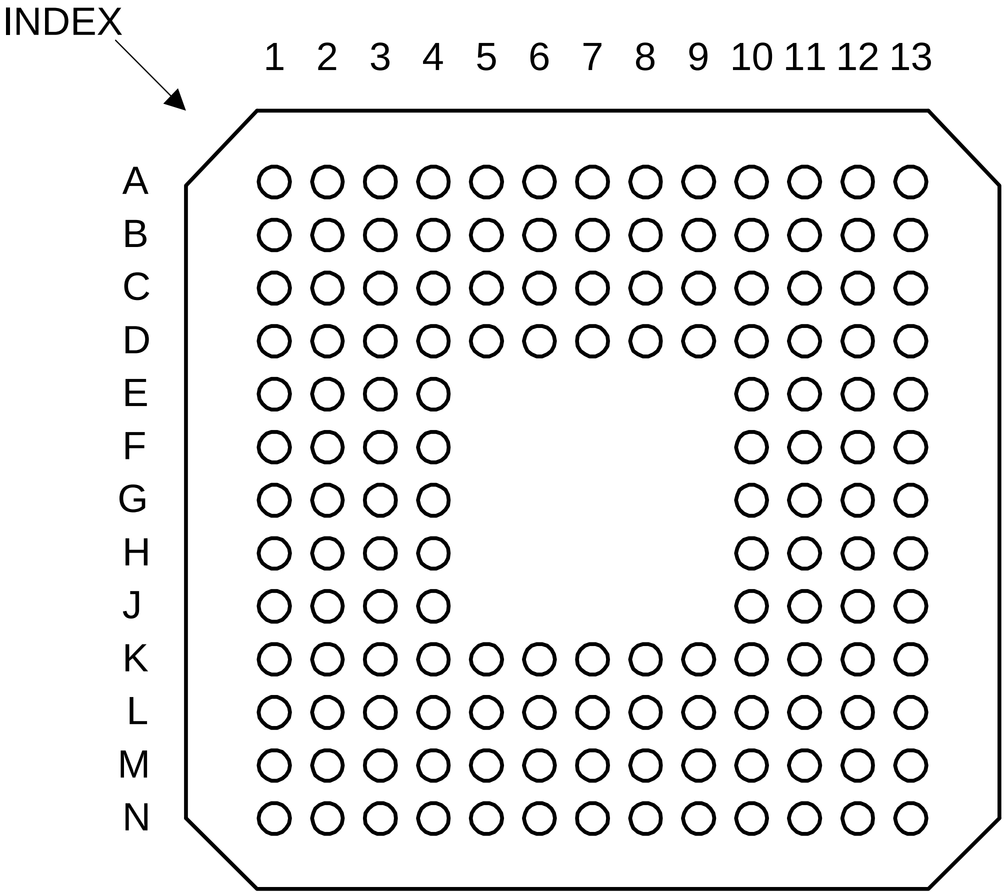
Functional Block Diagram



PIN DIAGRAM - 144 Pin LQFP



144 Pin - Mini BGA



(Top View)

OZ6812 Mini-BGA Pin List

BGA 144	Pin Name
A1	REQ#
A2	D9/CAD30
A3	D0/CAD27
A4	WP/IOIS16/CCLKRUN#
A5	WAIT#/CSERR#
A6	GND
A7	A2/CAD24
A8	A3/CAD23
A9	A4/CAD22
A10	A6/CAD20
A11	A7/CAD18
A12	A12/CCBE2#
A13	A22/CTRDY#
B1	AD30
B2	GNT#
B3	D2/RFU
B4	CORE_VCC
B5	BVD1/STSCHG#/RI# / CSTSCHG
B6	VS1/CVS1
B7	A1/CAD25
B8	REG#/CCBE3#
B9	A5/CAD21
B10	A25/CAD19
B11	A24/CAD17
B12	A15/CIRDY#
B13	WE#/CGNT#
C1	AD28
C2	AD29
C3	D10/CAD31
C4	D1/CAD29
C5	CD2/CCD2#
C6	RDY/REQ# / CINT#
C7	SOCKET_VCC
C8	INPACK#/CREQ#
C9	RESET/CRST#
C10	GND
C11	A16/CCLK#
C12	A21/CDEVSEL#
C13	A19/CBLOCK#
D1	AD25
D2	AD27
D3	GND
D4	AD31
D5	D8/CAD28
D6	BVD2/SPKR#/LED / CAUDIO
D7	A0/CAD26
D8	CORE_VCC
D9	VS2/CVS2
D10	A23/CFRAME#
D11	A20/CSTOP#
D12	CORE_VCC
D13	A18/RFU
E1	IDSEL
E2	C/BE3#
E3	AD24
E4	AD26
E10	A14/CPERR#
E11	A13/CPAR
E12	A8/CCBE1#
E13	A9/CAD14
F1	AD22
F2	AD21
F3	AD23
F4	CORE_VCC
F10	A17/CAD16
F11	IOW#/CAD15
F12	A11/CAD12
F13	GND
G1	AD20
G2	RST#
G3	PCI_VCC
G4	PCI_CLK
G10	IORD#/CAD13
G11	SOCKET_VCC
G12	OE#/CAD11
G13	CE2#CAD10
H1	GND
H2	AD19
H3	AD18
H4	AD16
H10	CORE_VCC
H11	D15/CAD8
H12	A10/CAD9
H13	CE1#/CCBE0#
J1	AD17
J2	C/BE2#
J3	IRDY#
J4	DEVSEL#
J10	D5/CAD3
J11	D6/CAD5
J12	D14/RFU
J13	D7/CAD7
K1	FRAME#
K2	PCI_VCC
K3	STOP#
K4	AD14
K5	AD10
K6	PCI_VCC
K7	AD0
K8	SPKR_OUT#
K9	MF5
K10	CD1/CCD1#
K11	GND
K12	D12/CAD4
K13	D13/CAD6
L1	TRDY#
L2	SERR#
L3	PAR
L4	GND
L5	AD8
L6	AD6
L7	AD3
L8	MF0
L9	MF3
L10	MF6
L11	VPPD1
L12	D11/CAD2
L13	D4/CAD1
M1	PERR#
M2	AD15
M3	AD12
M4	PCI_VCC
M5	C/BE0#
M6	AD4
M7	AD1
M8	RI_OUT/PME#
M9	AUX_VCC
M10	CORE_VCC
M11	VPPD0/SLATCH
M12	VCCD1#/SMBCLK/SCLK
M13	D3/CAD0
N1	C/BE1#
N2	AD13
N3	AD11
N4	AD9
N5	AD7
N6	AD5
N7	AD2
N8	GND
N9	MF1
N10	MF2
N11	MF4
N12	SUSPEND#
N13	VCCD0#/SMBDATA/SDATA

Pin List

Bold Text = Normal Default Pin Name

PCI Bus Interface Pins

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
AD[31:0]	**PCI Bus Address Input / Data:** These pins connect to PCI bus signals AD[31:0]. A Bus transaction consists of an address phase follow by one or more data phases.	3-5, 7-11, 15-17, 19, 23-26, 38-41, 43, 45-47, 49, 51-57	D4, B1, C2-1, D2, E4, D1, E3, F3, F1, F2, G1, H2-3, J1, H4, M2, K4, N2, M3, N3, K5, N4, L5, N5, L6, N6, M6, L7, N7, M7, K7	TTL	I/O	PCI_Vcc	PCI Spec
C/BE[3:0]#	**PCI Bus Command / Byte Enable:** The command signaling and byte enables are multiplexed on the same pins. During the address phase of a transaction, C/BE[3:0]# are interpreted as the bus commands. During the data phase, C/BE[3:0]# are interpreted as byte enables. The byte enables are to be valid for the entirety of each data phase, and they indicate which bytes in the 32-bit data path are to carry meaningful data for the current data phase.	12, 27, 37, 48	E2, J2, N1, M5	TTL	I/O	PCI_Vcc	PCI Spec
FRAME#	**Cycle Frame:** This input indicates to the OZ6812 that a bus transaction is beginning. While FRAME# is asserted, data transfers continue. When FRAME# is de-asserted, the transaction is in its final phase.	28	K1	TTL	I/O	PCI_Vcc	PCI Spec
IRDY#	**Initiator Ready:** This input indicates the initiating agent’s ability to complete the current data phase of the transaction. IRDY# is used in conjunction with TRDY#.	29	J3	TTL	I/O	PCI_Vcc	PCI Spec
TRDY#	**Target Ready:** This output indicates target Agent's the OZ6812’s ability to complete the current data phase of the transaction. TRDY# is used in conjunction with IRDY#.	31	L1	TTL	I/O	PCI_Vcc	PCI Spec
STOP#	**Stop:** This output indicates the current target is requesting the master to stop the current transaction.	33	K3	TTL	I/O	PCI_Vcc	PCI Spec
IDSEL	**Initialization Device Select:** This input is used as a chip select during configuration read and write transactions. This is a point-to-point signal. IDSEL can be used as a chip select during configuration read and write transactions.	13	E1	TTL	I	PCI_Vcc	PCI Spec
DEVSEL#	**Device Select:** This output is driven active LOW when the PCI address is recognized as supported, thereby acting as the target for the current PCI cycle. The Target must respond before timeout occurs or the cycle will terminate.	32	J4	TTL	I/O	PCI_Vcc	PCI Spec
PERR#	**Parity Error:** The output is driven active LOW when a data parity error is detected during a write phase.	34	M1	-	TO	PCI_Vcc	PCI Spec
SERR#	**System Error:** This output is driven active LOW to indicate an address parity error.	35	L2	-	TO	PCI_Vcc	PCI Spec

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
PAR	**Parity:** This pin generates PCI parity and ensures even parity across AD[31:0] and C/BE[3:0]#. During the address phase, PAR is valid after one clock. With data phases, PAR is stable one clock after a write or read transaction.	36	L3	TTL	I/O	PCI_Vcc	PCI Spec
PCI_CLK	**PCI Clock:** This input provides timing for all transactions on the PCI bus to and from the OZ6812. All PCI bus signals, except RST#, are sampled and driven on the rising edge of PCI_CLK. This input can be operated at frequencies from 0 to 33 MHz.	21	G4	-	I	PCI_Vcc	-
RST#	**Device Reset:** This input is used to initialize all registers and internal logic to their reset states and place most OZ6812 pins in a HIGH-impedance state.	20	G2	-	I	AUX_Vcc	-
GNT#	**Grant** : This signal indicates that access to the bus has been granted.	2	B2	TTL	I	PCI_Vcc	PCI Spec
REQ#	**Request** : This signal indicates to the arbiter that the OZ6812 requests use of the bus.	1	A1	-	TO	PCI_Vcc	PCI Spec

Power Control and General Interface Pins

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
RI_OUT/ PME#	**Ring Indicate Out:** This pin is Ring Indicate when the following occurs while O_2 Mode Control B Register (index 2Eh) bit 7 is set to 1: 1) Power Control (Index+02h) bit 7 set to 1 2) Interrupt and General Control (Index+03h) bit 7 set to 1 3) PCI O2Micro Control 2 (Offset: D4h) bit X = 0 **Power Management Event:** A power management event is the process by which the OZ6812 can request a change of its power consumption state. Usually, a PME occurs during a request to change from a power saving state to the fully operational state.	59	M8	-	TO	Aux_Vcc	4mA
SPKR_OUT#	**Speaker Output:** This output can be used to support PCCard audio output. See O2 Mode E Register (Index + 3Eh), bit 1.	62	K8	TTL	I/O	Aux_Vcc	12mA
MF[6:0]	**Multifunction Terminal [6:0] :** See PCI Multifunction MUX Register (Offset:08h).	69-67, 65-64, 61-60	L10, K9, N11, L9, N10-9, L8	TTL	I/O	Aux_Vcc	12mA
SUSPEND#	**Suspend:** This signal is used to protect the internal registers from clearing when the PCI RST# signal is asserted. When low, this signal is used to mask the PCI RESET during suspend. This pin can be used during suspend to prevent controller reset.	70	N12	TTL	I	Aux_Vcc	-

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
VPPD0 /SLATCH	**VPPD0:** This power input is used with parallel power control chip **SLATCH:** This output controls a serial interface power control chip.	71	M11	TTL	I/O	Aux_Vcc	12mA
VPPD1	**VPPD1:** This power input is used a parallel power interface chip.	72	L11	-	TO	Aux_Vcc	12mA
VCCD0# /SMBDATA /SDATA	**VCCD0# :** Rail power inputs for use with a parallel power control chip. **SMBus Data:** This pin serves as a bi-directional data pin when used with the SMBus socket power control chip. **Serial Data:** This pin serves as output DATA pin when used with a serial interface of serial power control chip.	73	N13	TTL	I/O	Aux_Vcc	12mA
VCCD1# /SMBCLK /SCLK	**VCCD1#:** Rail power inputs for use with a parallel power control chip. **SMBus Clock:** This pin serves as the clock when used with a SMBus socket power control chip. **Serial Clock:** The input is used as a reference clock (10-100kHz, usually 32kHz) to control a serial power control chips. By setting PCI O2Micro Control 2 register (Offset:D4h) bit 13 to 1, SCLK is an output. Default is input mode.	74	M12	TTL	I/O	Aux_Vcc	12mA

OZ6812 Slot Power Table

VCCD0#	VCCD1#	VPPD0	VPPD1	Slot_VCC	Slot_VPP
0	1	0	0	5	Switch Dependent
0	1	0	1	5	5
0	1	1	0	5	12
0	1	1	1	5	Switch Dependent
1	0	0	0	3.3	Switch Dependent
1	0	0	1	3.3	3.3
1	0	1	0	3.3	12
1	0	1	1	3.3	Switch Dependent

PCCard Socket Interface Pins

Refer to PCI Bus Interface pin descriptions for details on CardBus function.
EXCEPTIONS: CCD[2:1]#, CAUDIO, CSTSCHG, CVS[2:1]

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
REG#/ CCBE3#	**Register Access:** During PCCard memory cycles, this output chooses between Attribute and Common Memory. During I/O cycles for non-DMA transfers, this signal is active (low). During ATA mode, this signal is always inactive. For DMA cycles on the OZ6812 to a DMA-capable card, REG# becomes DACK to the PCMCIA card. **CardBus Command Byte Enable:** In CardBus mode, this pin is the CCBE3#.	125	B8	TTL	I/O	Socket _Vcc	CardBus spec.
A[25:24]/ CAD[19, 17]	**Address:** PCCard socket address 25:24 outputs. **CardBus Address/Data:** CardBus mode, these pins are the CAD bits 19 and 17.	116, 113	B10, B11	TTL	I/O	Socket _Vcc	CardBus spec.
A23/ CFRAME#	**Address:** PCCard socket address 23 output. **CardBus Frame:** In CardBus mode, this pin is the CFRAME# signal.	111	D10	TTL	I/O	Socket _Vcc	CardBus spec.
A22/ CTRDY#	**Address:** PCCard socket address 22 output. **CardBus Target Ready:** In CardBus mode, this pin is the CTRDY# signal.	109	A13	TTL	I/O-PU	Socket _Vcc	CardBus spec.
A21/ CDEVSEL#	**Address:** PCCard socket address 21 output. **CardBus Device Select:** In CardBus mode, this pin is the CDEVSEL# signal.	107	C12	TTL	I/O-PU	Socket _Vcc	CardBus spec.
A20/ CSTOP#	**Address:** PCCard socket address 20 output. **CardBus Stop:** In CardBus mode, this pin is the CSTOP# signal.	105	D11	TTL	I/O-PU	Socket _Vcc	CardBus spec.
A19/ CBLOCK#	**Address:** PCCard socket address 19 output. **CardBus Lock:** In CardBus mode, this signal is the CBLOCK# signal used for locked transactions.	103	C13	TTL	I/O-PU	Socket _Vcc	CardBus spec.
A18/ RFU	**Address:** PCCard socket address 18 output. **Reserved:** In CardBus mode, this pin is reserved for future use.	100	D13	TTL	TO	Socket _Vcc	CardBus spec.
A17/ CAD16	**Address:** PCCard socket address 17 output. **CardBus Address/Data:** In CardBus mode, this pin is the CAD bit 16.	98	F10	TTL	I/O	Socket _Vcc	CardBus spec.
A16/ CCLK#	**Address:** PCCard socket address 16 output. **CardBus Clock:** In CardBus mode, this pin supplies the clock to the inserted card.	108	C11	TTL	I/O	Socket _Vcc	CardBus spec.

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
A15/ CIRDY#	**Address:** PCCard socket address 15 output. **CardBus Initiator Ready:** In CardBus mode, this pin is the CIRDY# signal.	110	B12	TTL	I/O-PU	Socket _Vcc	CardBus spec.
A14/ CPERR#	**Address:** PCCard socket address 14 output. **CardBus Parity Error:** CardBus mode, this pin is the CPERR# signal.	104	E10	TTL	I/O-PU	Socket _Vcc	CardBus spec.
A13/ CPAR	**Address:** PCCard socket address 13 output. **CardBus Parity:b** In CardBus mode, this pin is the CPAR signal.	101	E11	TTL	I/O	Socket _Vcc	CardBus spec.
A12/ CCBE2#	**Address:** PCCard socket address 12 output. **CardBus Command/Byte Enable:** In CardBus mode, this pin is the CCBE2# signal.	112	A12	TTL	I/O	Socket _Vcc	CardBus spec.
A[11:9]/ CAD [12,9,14]	**Address:** PCCard socket address 11:9 output. **CardBus Address/Data:** In CardBus mode, these pin are the CAD bits 12, 9 and 14.	95, 89, 97	F12,H12, E13	TTL	I/O	Socket _Vcc	CardBus spec.
A8/ CCBE1#	**Address:** PCCard socket address 8 output. **CardBus Command/Byte Enable:** In CardBus mode, this pin is the CCBE1# signal.	99	E12	TTL	I/O	Socket _Vcc	CardBus spec.
A[7:0]/ CAD[18] [20:26]	**Address:** PCCard socket address 7:0 outputs. **CardBus Address/Data:** In CardBus mode, these pins are the CAD bits 18 and 20:26.	115, 118, 120, 121, 124, 127, 128, 129	A11-10, B9, A9-7, B7, D7	TTL	I/O	Socket _Vcc	CardBus spec.
D15/ CAD8	**Data:** PCCard socket I/O data bit 15. **CardBus Address/Data:** In CardBus mode, this pin is the CAD bit 8.	87	H11	TTL	I/O	Socket _Vcc	CardBus spec.
D14/ RFU	**Data:** PCCard socket I/O data bit 14. **Reserved:** In CardBus mode, this pin is reserved for future use.	84	J12	TTL	I/O	Socket _Vcc	CardBus spec.
D[13:3]/ CAD[6, 4, 2, 31, 30, 28, 7, 5, 3, 1, 0]	**Data:** PCCard socket I/O data Ibits 13:3. **CardBus Address/Data:** In CardBus mode, this pin is the CAD bit 6 4, 2, 31, 30, 28, 7, 5, 3, 1, and 0, respectively.	82, 80, 77, 144, 142, 140, 85, 83, 81, 79, 76	K13-12, L12, C3, A2, D5, J13, J11, J10, L13, M13	TTL	I/O	Socket _Vcc	CardBus spec.
D2/ RFU	**Data:** PCCard socket I/O data bit 2. **Reserved:** In CardBus mode, this pin is reserved for future use.	143	B3	TTL	I/O	Socket _Vcc	CardBus spec.
D[1:0]/ CAD[29,27]	**Data:** PCCard socket I/O data bits 1:0. **CardBus Address/Data:** In CardBus mode, these pins are the CAD bits 29 and 27, respectively.	141, 139	C4, A3	TTL	I/O	Socket _Vcc	CardBus spec.
OE#/ CAD11	**Output Enable**: This output goes active (low) to indicate a memory read from the OZ6812 to PCCard. **CardBus Address/Data:** In CardBus mode, this pin is the CAD bit 11.	92	G12	TTL	I/O	Socket _Vcc	CardBus spec.
WE#/ CGNT#	**Write Enable**: This output goes active (low) to indicate a memory write from the OZ6812 to the PCCard socket. **CardBus Grant:** In CardBus mode, this pin is the CGNT# signal.	106	B13	TTL	TO	Socket _Vcc	CardBus spec.

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
IORD#/ CAD13	**I/O Read**: This output goes active (low) for I/O reads from the OZ6812 to the socket. **CardBus Address/Data:** In CardBus mode, this pin is the CAD bit 13.	93	G10	TTL	I/O	Socket _Vcc	CardBus spec.
IOW#/ CAD15	**I/O Write**: This output goes active (low) for I/O writes from the OZ6812 to the socket. **CardBus Address/Data:** In CardBus mode, this pin is the CAD bit 15.	96	F11	TTL	I/O	Socket _Vcc	CardBus spec.
WP/ IOIS16#/ CCLKRUN#	**Write Protect / I/O is 16-Bit:** In Memory mode, this input is indicates the status of the write protect switch on the PCCard. In I/O mode, this input indicates the size of current data transfer on the PCCard. **CardBus Clock Run:** In CardBus mode, this pin is the CCLKRUN# signal, which starts and stops the CardBus CCLK. To enable the CLKRUN# signal, ExCA register 3Bh bit[3:2] must be enabled.	136	A4	TTL	I/O-PU	Socket _Vcc	CardBus spec.
INPACK#/ CREQ#	**Input Acknowledge:** The INPACK# function is not applicable in PCI bus environments. This pin is provided for Legacy card compatibility. **CardBus Request:** In CardBus mode, this pin is the CREQ# signal.	123	C8	-	I-PU	Socket _Vcc	CardBus spec.
RDY/IREQ#/ CINT#	**Ready / Interrupt Request:** In Memory mode, this input indicates that the card is ready or busy. In I/O mode, this input indicates a card interrupt request. **CardBus Interrupt:** In CardBus mode, this pin is the CINT# signal. This signal is active-low and level-sensitive.	132	C6	-	I-PU	Socket _Vcc	CardBus spec.
WAIT#/ CSERR#	**Wait:** This pin is driven by the PCCard to delay completion of the current cycle. **CardBus System Error:** In CardBus mode, this pin is the CSERR# signal.	133	A5	TTL	I-PU	Socket _Vcc	CardBus spec.
CD[2:1]/ CCD[2:1]#	**Card Detect**: These inputs indicate a card is present in the socket. They are internally pulled high to AUX_VCC. **CardBus Card Detect:** In CardBus mode, these inputs are used with CVS[2:1] to detect presence and type of card.	137, 75	C5, K10	TTL	I-PU-Schmitt	Aux_Vcc	CardBus spec.
CE2#/ CAD10	**Card Enable 2:** This pin is driven low to control byte/word card access. CE2# enables odd-numbered address bytes. **CardBus Address/Data:** In CardBus mode, this pin is the CAD bit 10.	91	G13	TTL	I/O	Socket _Vcc	CardBus spec.
CE1#/ CCBE0#	**Card Enable 1:** This pin is driven low to control byte/word card access. CE1# enables even-numbered address bytes. When configured for 8-bit cards, CE1# is active and A0 is used to indicate access of odd- or even-numbered bytes. **CardBus Command/Byte Enable:** In CardBus mode, this pin is the CCBEO# signal.	88	H13	TTL	I/O	Socket _Vcc	CardBus spec.

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
RESET/CRST#	**Reset:** This active high output resets the card. To prevent reset glitches, this signal is high-impedance unless a card is seated in the socket, card power is applied, and the card's interface signals are enabled. **CardBus Reset:** In CardBus mode, this pin is the CRST# output.	119	C9	TTL	TO	Socket_Vcc	CardBus spec.
BVD2/SPKR#/LED / CAUDIO	**Battery Voltage Detect 2 / Speaker / LED:** In Memory mode, this input serves as the BVD2 (battery warning status) input. In I/O mode, this input can be configured as the card's SPKR# audio input or drive-active LED input. **CardBus Audio:** In CardBus mode, this pin is the CAUDIO input.	134	D6	-	I-PU	Socket_Vcc	-
BVD1/STSCHG#/RI# / CSTSCHG	**Battery Voltage Detect 1 / Status Change / Ring Indicate:** In Memory mode, this is the BVD1 (battery-dead status) input. In I/O mode, this is the STSCHG# input indicating that the card's internal status has changed, or the ring indicate input for wakeup-on-ring system power management support. See bit 7 of the Interrupt and General Control register (03h). **CardBus Status Change:** In CardBus mode, this pin is the CSTSCHG. This pin can be used to generate PME#.	135	B5	-	I-PU	Socket_Vcc	-
VS[2:1]/CVS[2:1]	**Voltage Sense:** These pins are used in conjunction with CD[2:1] to determine the type and voltage of a card. These pins are internally pulled high to AUX_VCC. See Table 1. **CardBus Voltage Sense:** In CardBus mode, these pins are the CVS[2:1] pins.	117, 131	D9, B6	TTL	I/O-PU	Aux_Vcc	CardBus spec.
SOCKET_VCC	**Socket Power:** These pins are the power rail input for the socket interface control logic. These pins can be 0, 3.3, or 5 V,. The socket interface outputs will operate at the voltage applied to these pins.	90, 126	G11, C7	-	PWR	-	-

Power, Ground, and Reserved Pins

Pin Name	Description	Pin Number		Input	Type	Power Rail	Drive
		LQFP	BGA				
Aux_VCC	**Auxiliary VCC**: This pin is connected to the system's 3.3/5V power supply. For the device to 5V tolerant, connect to +5V power.	63	M9	-	PWR	-	-
CORE_VCC	**CORE_VCC:** This pin provides power to the core circuitry of the OZ6812. It must be connected to a 3.3V power supply.	14, 66, 86, 102, 122, 138	F4, M10, H10, D12, D8, B4	-	PWR	-	-
PCI_VCC	**PCI Bus VCC:** These pins can be connected to either a 3.3V or5V power supply. The PCI bus interface will operate at the voltage applied to these pins, independent of the voltage applied to other OZ6812 pin groups.	18, 30, 44, 50	G3, K2, M4, K6	-	PWR	-	-
GND	**System Ground**	6, 22, 42, 58, 78, 94, 114, 130	D3, H1, L4, N8, K11, F13, C10, A6	-	GND	-	-

Legend

I/O Type	Description
I	Input Pin
I-PU	Input pin with internal pull-up
I-PU Schmitt	Input pin with internal pull-up and Schmitt trigger
O	Output
OD	Open-drain
TO	Tri-state output
TO-PU	Tri-state output with internal pull-up
OD-PU	Open-drain output with internal pull-up
PWR	Power pin

Power Rail	Source of Output's Power
1	AUX_VCC: outputs powered from AUX_VCC
2	SOCKET_VCC: outputs powered from the socket
3	PCI_VCC: outputs powered from PCI bus power supply
4	CORE_VCC: outputs powered from the CORE_VCC

FUNCTIONAL INFORMATION

1. INTRODUCTION TO OZ6812

1.1. Architectural Overview

O2Micro's OZ6812 is the solution for today's notebook PCs. It interfaces directly to the PC card socket and PCI bus without additional buffers, thus requiring minimal PC board real estate. It is a bridge between the PCI local bus and the PCMCIA socket, supporting both 16-bit PCCard and 32-bit CardBus devices. The OZ6812 provides a 32-bit point-to-point connection between the card socket and CardBus bridge, enabling a maximum theoretical throughput of 132MB/second. The OZ6812's core logic is powered at 3.3V to minimize power dissipation.

The OZ6812 allows PC socket insertion of memory and I/O devices as exchangeable peripherals. Five programmable memory windows and 2 programmable I/O windows are available to map inserted PC cards into system memory and I/O space. PC cards may have both attribute and common memory. Attribute memory indicates the PC Card's capabilities of the PC Card to host software and to allow host software to change configuration. Common memory can be used by host software for any purpose (flash file system, system memory, floppy emulation, etc.). I/O PC Cards, such as Modem and LAN Cards, are supported as if they were I/O devices on the system motherboard.

PC Card I/O device interrupts need not be limited to PCI interrupts, and each generated interrupt can be steered by the OZ6812 to a variety of system interrupts. The OZ6812 can notify the host system via interrupts when an event such as card status change (CSC) or interrupts from the PC Card occur. Both CSC and functional interrupts can be individually masked and routed to the system interrupt controller via PCI-style, ISA type, or serialized IRQ protocol.

The OZ6812 also has a flexible interface to the socket power control device. It establishes either a parallel or serial communication protocol and communicates in conventional parallel mode or three possible serial modes, standard external serial hardware, or System Bus Management mode.

1.2. PCI Interface

The OZ6812 employs the same basic transaction protocol as the PCI bus. See Appendix 1 for a comparison of CardBus to PCI bus. All OZ6812 CardBus bus transactions are synchronized to the CCLK# signal and consist of an address phase followed by one or more data phases. Each address phase is one CCLK# in duration while each data phase has a minimum duration of one CCLK#. The number of data phases depends on how many data transfers take place during the overall burst transfer. A wait state inserted in a data phase will extend the data phase by an additional CCLK#. The OZ6812 responds as a PCI target device to PCI bus cycles decoded from the address phase of each cycle. Refer to a PCI reference book such as "PCI Hardware and Software Architecture and Design" for valid PCI bus cycle types and their encoding on the 4-bit C/BE bus during the address phase of a bus cycle. The most common PCI bus commands are read and write cycles to memory, I/O, and configuration address spaces.

1.3. Configuration Address Space

Like all PCI devices, the OZ6812 has configuration registers which must be programmed during system initialization. This key feature enables Plug and Play capability. The ability to identify the OZ6812 and dynamically assign system resources helps optimize system memory and I/O address space maps, avoiding system resource conflicts.

The OZ6812 is configured essentially in the same manner as any other device residing on the PCI bus. It is the OS software's responsibility to scan the PCI bus, and detect and configure the present devices. The OZ6812 is configured during the power-up sequence before recognizing PCCard installation and PCMCIA software notification that a PCCard is present.

Configuring the CardBus card is the PCMCIA software's responsibility. When a PCCard is inserted into a socket, the OZ6812 detects its presence and generates an interrupt to notify PCMCIA configuration software (client driver, Card Services and Socket Services). This software will then apply appropriate power to the socket and configure the PC Card and OZ6812 socket interface.

The PCI local bus specification defines two types of PCI configuration read and write cycles: type 0 and type 1. Type 0 configuration cycles are intended for devices on the main PCI bus, while type 1 configuration cycles are intended for devices residing on the CardBus socket bus. The difference between these two types is the PCI address AD bus encoding during the address phase of the cycle. The address AD bus encoding during the address phase of a type 0 configuration cycle, AD[10:8] selects the OZ6812. The contents of AD[7:2] accesses the configuration register. The OZ6812 claims Type 0 configuration cycles when its IDSEL pin is asserted during a configuration access cycle.

Type 1 configuration cycles are intended for PCI devices residing on either of the CardBus socket interfaces with each assigned their own CardBus bus number. The OZ6812 will claim Type 1 configuration cycles based on the destination bus number embedded in the configuration cycle address phase. The destination bus number encoded in the AD bus is compared to the values programmed in the OZ6812 configuration registers; PCI bus number (18h), CardBus bus number (19h), and subordinate bus number (1Ah). It is the responsibility of the PCI operating system to program these registers with the proper bus hierarchy, they are 00h by default.

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. The OZ6812 never issues PCI configuration read or write cycles on the PCI bus as a PCI bus master.

1.4. PCMCIA Socket Interface

All PC Cards, 16-bit and CardBus, communicate with the OZ6812 through a 68 pin PCMCIA/JEIDA socket interface through 60 signals and 8 power pins. The power pin definitions remain unchanged for both types of PC Cards, with the only difference in signal pin definition.

The OZ6812 can directly support one PC card socket, with an option to allow up to eight OZ6812 devices per system. The OZ6812 supports two PC card types (either memory or I/O) interchangeably. It accomplishes this by multiplexing some of the static signals that are defined differently for memory and I/O PC cards. These signals are configured appropriately by accessing the PC card’s configuration registers.

1.5. Memory and I/O Window Mapping

Memory address mappings for the OZ6812 bridge are on 4K byte boundaries with a minimum mapping of 4 Kbytes. OZ6812 may be mapped anywhere within the address space assigned to the bridge. The OZ6812 provides two memory base and limit register pairs which may be used for mapping memory mapped I/O or prefetchable memory space.

Two I/O mapping register pairs are provided for each socket, allowing some fragmenting of I/O space on a card and interleaving of I/O space with other I/O devices.

Multiple PC cards in a system can conflict if they try to utilize the same system memory and I/O range. The OZ6812 allows mapping of each PC card into separate memory and I/O ranges through the use of 5 programmable memory windows and 2 programmable I/O windows for 16-bit PC cards, or through two memory or I/O windows for 32-bit CardBus cards. This function avoids system configuration conflicts. The OZ6812 has 5 memory windows with memory start address, end address, offset address and upper address register to select 16-Mbyte page.

The OZ6812 provides memory paging, memory address mapping for both PC card attribute and common memory, and I/O address mapping. The OZ6812 includes registers allowing access to the card information structure and card configuration registers within the attribute memory described by the PCMCIA/JEIDA PC Card Standard.

1.6. Zoomed Video Port

The Zoomed Video (ZV) Port provides a low-cost functionality for direct video data transfer from a PC Card to the VGA frame buffer. This uni-directional bus enables the transfer of uninterrupted video data at rates up to 30 frames per second without loading the system PCI bus. Both video and audio data are transferred real-time with minimal CPU intervention. The PC Card transfers audio data to the host system using Pulse Code Modulation (PCM) in a serial I2S format. Video data is transferred by the ZV port, in compliance with CCIR601 timing, enabling direct interfacing with industry standard video graphic devices.

1.7. DVD Enhanced ZV PORT Offset D0h

The ZV port specification was originally developed with ISA MPEG1 data rates in mind. Today, DVD MPEG II playback has 4x bandwidth requirements which easily overwhelms the traditional 8Mhz ISA type interface. O2Micro addresses this deficiency in the ZV port specification by introducing a 1x, 2x and 4x clock rate control in the O2Micro Control1 Register (D0h) bits 18-19. Enabling the DVD Enhanced ZV port will increases the PC Card clock by a factor of 1x, 2x, and 4x, greatly increasing the available bandwidth for video data.

2. PC Card Detect and Power Management

Prior to 1995, only 5V PC Card slots existed until the 1995 PCMCIA PC Card standard introduced the 3.3V CardBus and 3.3V/5V card slots. It is now possible to design CardBus cards and 16-bit PC Cards which use a 3.3V supply for added power savings. Unfortunately, this complicates the hot insertion and card recognition schemes, since it is now possible to damage a 3.3V card by inserting it into a socket pre-configured for 5V. A PC Card socket can no longer maintain default 5V operation configuration. Voltage requirements and card type must now be detected before applying power. The OZ6812 addresses this by implementing a hardware/software interrogation procedure initiated at card insertion.

2.1. PC Card Sensing

The OZ6812 supports hot insertion and detection of either CardBus or 16-bit PC Cards. Card insertion detection, determining card type and setting the initial socket Vcc consists of a two step process. The pins CD2, CD1 for 16-bit cards, and CCD1# and CCD2# for CardBus cards are utilized to detect card insertion. The pins VS2 and VS1 for 16-bit cards, and CVS2 and CVS1 for CardBus cards are used to determine card type and the required initial socket voltage. It is the PC Card designer's responsibility to connect these four pins in the proper configuration corresponding to the type of card (16 bit or CardBus) and the supply voltage requirements.

Table 1 shows how the OZ6812 would interpret a PC Card's detect and voltage pins for initial socket voltage and card type. The x.x and y.y operating voltages are supported, even though they are not yet defined by the PC Card Specification. The OZ6812 assumes a low-voltage key (CardBus-capable socket in system).

CD1/ CCD1#	CD2/ CCD2#	VS1/ CVS1	VS2/ CVS2	Card Type	Initial Vcc
ground	Ground	open	Open	16-Bit	5 vdc
ground	Ground	ground	Open	16-Bit	3.3 vdc
ground	Ground	open	Ground	16-Bit	x.x vdc
ground	Ground	ground	Ground	16-Bit	3.3 & x.x vdc
CVS1	Ground	CCD1#	Open	CardBus	3.3 vdc
ground	CVS2	open	CCD2#	CardBus	x.x vdc
ground	CVS1	CCD2#	Open	CardBus	y.y vdc
ground	CVS2	ground	CCD2#	CardBus	3.3 & x.x
CVS2	Ground	open	CCD1#	CardBus	x.x & y.y
ground	CVS1	CCD2#	Ground	CardBus	3.3, x.x & y.y

Table 1. Card detect and Voltage Sense

2.2. Socket Power Control

Socket power is automatically disabled on power up reset or when no card is detected in the socket. Software can enable socket power when a card is detected on power up (card detect input pins CD[2:1] asserted low) or from a card insertion management interrupt (generated by sensing a change in state on CD[2:1]). When a card is removed from a socket, the OZ6812 will automatically disable the Vcc and Vpp supplies to the socket and notify the system via management interrupts. The OZ6812 provides control to the required pins over an external parallel or serial socket power-control device to switch socket voltages on or off.

Parallel Socket Power Control Mode:

The OZ6812 supports a conventional parallel socket power interface. The OZ6812 generates four pins **VPPD0 (VPP_PGM), VPPD1 (VPP_VCC), VCCD0# (VCC_3#) and VCCD1# (VCC_5#)** for each socket to control the socket power through an external power FET chip such as the Micrel MIC2563A. See Figure 1.

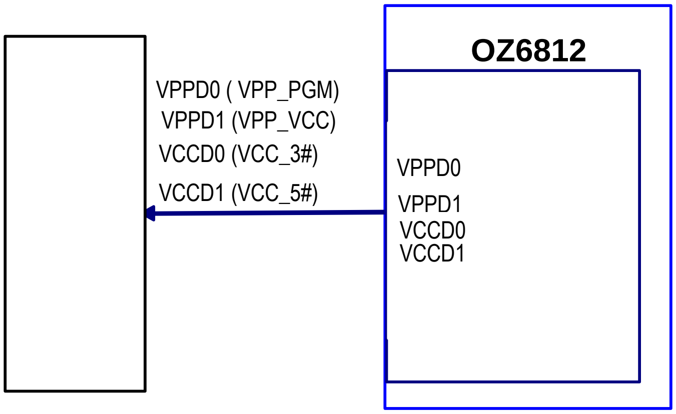


Figure 1. Parallel Signaling Mode

Serial Signaling Mode:

In this mode, power to the socket is controlled by software via the PC Card Socket Register/Control Base Register (PCI Configuration Register 10h). When a legal Vcc or Vpp is requested by PCMCIA software, the OZ6812 uses a synchronous serial protocol to communicate with a PC Card power interface switch. The OZ6812 uses a three-wire bus interface: clock (SCLK pin 74), data (SDATA pin 73), and latch line (SLATCH pin 71) to communicate with the power switch. Power information encoded on the serial SDATA line will simultaneously specify Vpp and Vcc settings for the socket. The SCLK, derived from a 32Khz source, serves as a reference clock for the OZ6812 and as a clock to the interface switch. Serial power information is transferred over the SDATA line at SCLK clock rates and synchronously latched in the interface switch by the SLATCH signal. See Figure 2.

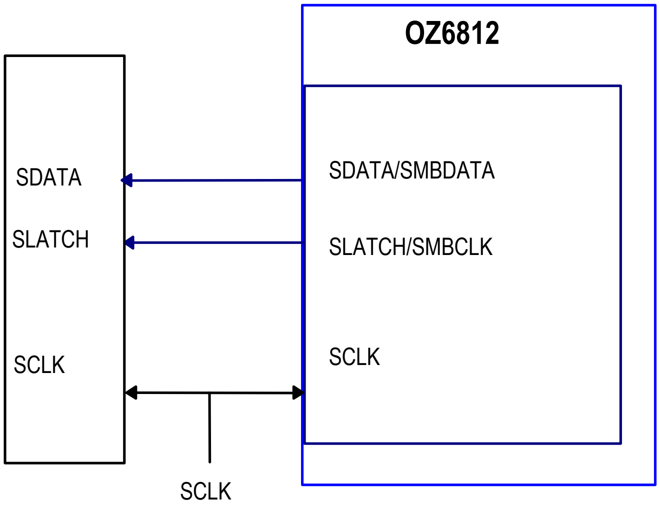


Figure 2. Serial Signaling Mode

System Management Bus (SMBus) Signaling Mode:

In this mode, the OZ6812 supports the System Management Bus (SMBus) protocol which employs a two wire serial interface; SMBDATA (pin 73) and SMBCLK (pin 74) as a reference for the OZ6812. The Maxim MAX 1601 dual-channel PC Card power switching network supports this SMBus protocol and accepts serial data from the SMBDATA pin and a serial clock from the SMBCLK pin. See Figure 3.

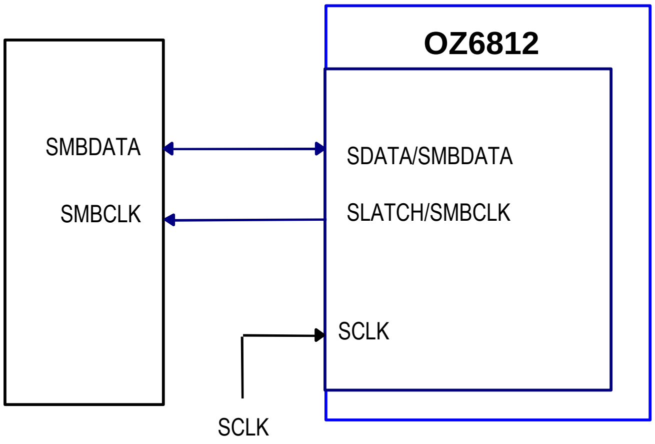


Figure 3. SMBus Signaling Mode

3. Interrupt Support

The OZ6812 detects interrupts and/or events at the PC Card interface and notifies the host interrupt controller via one of several interrupt signaling protocols. The OZ6812 supports two classes of interrupts; socket/card functional interrupts initiated by PC Cards activating their RDY/IREQ# signal and CSC management interrupts. CSC management interrupts are events at the PC Card interface which cause notification of host software for service. CSC management interrupts are triggered by PC Card/CardBus status changes such as; Card insertion or removal, Battery dead indicator (BVD1) or I/O-type card status change (STSCHG#); Battery warning indicator (BVD2) change on a memory-type card, or Ready (RDY) status change on a memory-type card.

The 16-bit I/O and CardBus PC Cards both have similar methods for signaling interrupts and also use two signals: one to indicate a change in card status, the other 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. Card insertion and removal events are independent of card type since the same card detect signal are used in both cases and the OZ6812 cannot distinguish between card types.

The PC Card standard describes the power-up sequence that must be followed by the OZ6812 when an insertion event occurs. Upon power-up sequence completion, the OZ6812 interrupt scheme can notify the host system that a PC Card has been activated. When an interrupt is signaled by the OZ6812, the interrupt service routine must determine which of the events caused the interrupt. Since there are many events which can cause interrupts, internal registers in the OZ6812 provide flags to identify to the host-interrupt-service routine which interrupt source caused the interrupt. By first reading these status bits, the interrupt-service routine can determine which action to take.

There are various methods in clearing the OZ6812 interrupt status flag bit. ExCA provides two methods to clear 16-bit PC Card-related interrupt fags. 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. The CardBus specification requires an explicit write of 1 to clear CardBus-related interrupt flags.

The PC Card standard does not specify how the interrupt routing should be accomplished. Interrupt routing is managed by the CardBus bridge and system board logic. It is up to the system designer to determine how the interrupts will be routed from the card's CINT# pin to a given IRQ line.

The PC Card standard does define an interrupt sharing mechanism that allows multiple CardBus functions to share a single CINT# pin. Each function within a multiple function PC Card has its own enabler that includes an interrupt service routine (ISR) designed specifically for that function. The interrupt sharing mechanism requires that the enabler ISR for each function be registered and managed by card services (CS).

The OZ6812 provides several interrupt signaling schemes to accommodate a variety of platforms. The different mechanisms for dealing with interrupts are based on various specifications and industry standards. The ExCA register set provides interrupt control for 16bit PC Card functions and requires an ISA architecture defined totem-pole IRQs. The CardBus specification provides interrupt control based on a PCI architecture defined open-drain IRQs. In compliance to the latest PC 98 guidelines, O2Micro’s OZ6812 supports simultaneous configuration of both ISA and PCI Interrupt modes. OZ6812 is dynamically configured to utilize ISA Interrupt mode for PC Card 16 cards or concurrently use PCI Interrupt mode for CardBus cards. Only PC card 16-bit (R2 type) function interrupt needs the ISA interrupt support, the CSC (Card Status Change) interrupt for R2 & R3 cards and PC card 32 bit (R3 type) function interrupt can use the PCI interrupt INTA# .

The OZ6812 supports multiple interrupt modes; PCI + ISA Interrupt scheme PCI Interrupt scheme, Intel's PC-PCI Serial Interrupt Protocol, PCI/Way and VESA Serialized IRQ

A PC Card 16 (R2 type) interrupt routing can be configured for either PC/PCI, PCI/Way or ISA IRQ while a CardBus (R3 type) is always configured for PCI Interrupt mode. Interrupt modes are selectable through bits [1:0] of the ExCA O2Micro Mode Control D Register (Index + 3Bh).

3.1. PCI + Parallel ISA-Architecture-Compatible Mode

The ExCA specification recommends that the interrupt from the PC Card be steered to any one of 10 IRQs (IRQ3, IRQ4, IRQ5, IRQ7, IRQ9, IRQ10, IRQ11, IRQ12, IRQ14, and IRQ15) on the ISA bus. The host software must first configure the OZ6812 to use ISA IRQ signaling by programming all multifunction Routing Registers. These IRQs represent the common interrupts expected by PC Card applications and several free IRQs for CSC routing. See Figure 4.

OZ6812's 10 IRQs can all be configured as edge-mode interrupts to support the standard interrupt from I/O type PC card and card status change interrupts, or as level mode interrupts to support I/O type PC card with pulse-mode interrupt requests. The OZ6812 can also be configured to run IRQ14 level mode, while all other IRQs are in edge-trigger mode. Therefore, PC cards with pulse-mode and edge-mode interrupts, and card status change interrupts are supported simultaneously.

The ten IRQ terminals will remain in high-impedance state until the ExCA CSC and functional interrupt routing registers are set to a valid state. All unused interrupt pins should be pulled high by a 43kΩ resistor. The following steps assume that the system has powered up and RSTIN# is high (deasserted):

1) If a PC Card is installed in the socket and requires functional interrupts, write to the lower nibble of ExCA register 03h/803h for desired functional interrupt routing for the socket.
2) Write the appropriate mask register bits to enable interrupt generation for desired events.
3) Upon card removal, host software masks any functional interrupts that were set for that socket.
4) Upon card insertion, host software reconfigures the mask and routing registers to support the new card requirements.

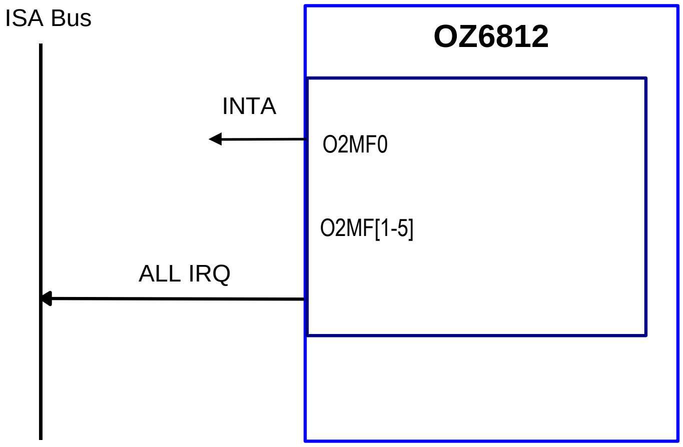


Figure 4. PCI + ISA Legacy Parallel Interrupt Mode

3.2. PCI Architecture Interrupt Mode

The OZ6812 also supports interrupt signaling compliant with the PCI local bus specification. The CardBus interface uses a single interrupt pin, CINT# shared with the socket. The single CardBus interrupt pin is meant to be shared by all functions implemented within a CardBus card. It is the responsibility of the O/S make the appropriate calls to socket services (SS) to setup the OZ6812 such that CardBus's CINT# line is steered to the specified interrupt output (e.g. PCI interrupt pin, serialized interrupt, or IRQ line). When the OZ6812 is configured for PCI interrupt signaling as INTA#, this pin behaves as open-drain PCI interrupts.

The host software needs to configure the OZ6812 for PCI signaling. The Interrupt Line Register (PCI configuration register offset 3Ch) must be written in order to route CSC and functional interrupts from the socket. The following steps assume that the system has powered up and RSTIN# is high (deasserted).

1) Set bit [3:0] of PCI configuration register 3Ch to route CSC interrupt & functional interrupt from the PC Card to INTA# .
2) Write to the appropriate mask register bits to enable interrupt generation for desired events.
3) Upon card-removal, host software masks any functional interrupts that were set for that socket.
4) Upon card insertion, host software reconfigures the mask and routing registers to support the new card requirements.

Each function within a CardBus card must have its own set of configuration registers occupying its function-specific configuration address space (64 dwords in size). The first 16 dwords of a function's configuration space is referred to as its configuration header. Each configuration header contains a read-only interrupt pin register determining whether the function uses interrupts. 00h doesn't use interrupts and a non-zero value indicates interrupt usage.

If the interrupt pin register indicates the function uses an interrupt, the interrupt descriptor table in the function's Card Information Structure (CIS) can specifiy the IRQ line(s) where the CINT# signal would be routed. The CardBus card's client driver parses the CIS to determine the interrupt routing. It then requests that one of the desired IRQs be allocated to its function by making a service call to Card Services (CS). It is CS's the responsibility to determine if the request IRQ line is available and, if so, to allocate the IRQ line to the client driver and its card function. Once an IRQ line has been allocated to a card function and all other system resources needed by the function have been allocated, the client driver will set to the appropriate OZ6812 configuration register.

In this mode the OZ6812 has three different ways to generate interrupts to the interrupt controller. The OZ6812 can support ISA IRQ and PCI Interrupt at the same time. The CardBus uses PCI interrupts and PC Card 16-bit cards (R2) uses ISA 10 IRQs in the parallel or serial mode. The interrupt modes selectable are:

- PCI Interrupt Mode
- PC/PCI Interrupt Mode
- PCI/Way Interrupt Mode

3.3. PCI Interrupt Mode

This is the PCI default mode for CardBus cards which use the status change interrupt and CINT# event interrupts. Socket A would individually use INTA# as the PCI-type INT#' open-drain interrupts. Refer to Figure 5.

3.4. PC/PCI Interrupt Mode

This mode supports the mobile PC/PCI Extended Programming Mode. In this mode the SIN# and SOUT# interface with a serial interrupt controller (SIC). SIN# on the OZ6812 is the serial interrupt input line from other devices in the interrupt loop, and SOUT# is the serial interrupt output line containing the logical 'AND' of the interrupt's level as occurred in OZ6812, along with SIN# interrupts. The SIC is clocked by PCI CLK, and CCLKRUN# is used by the OZ6812 to restart PCI CLK if it has stopped. The number of interrupts supported depends on the SIC configuration. See Figure 6.

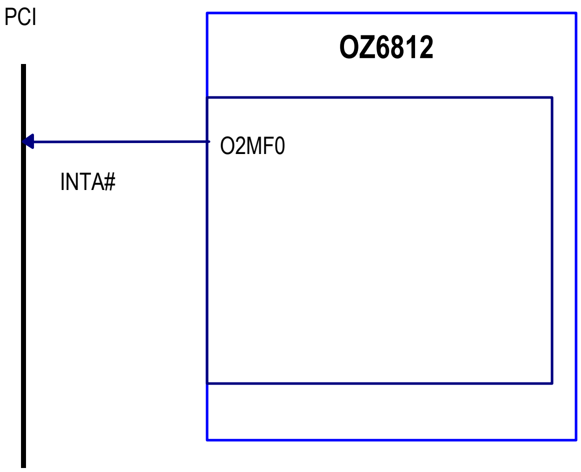


Figure 5. PCI Interrupt Mode

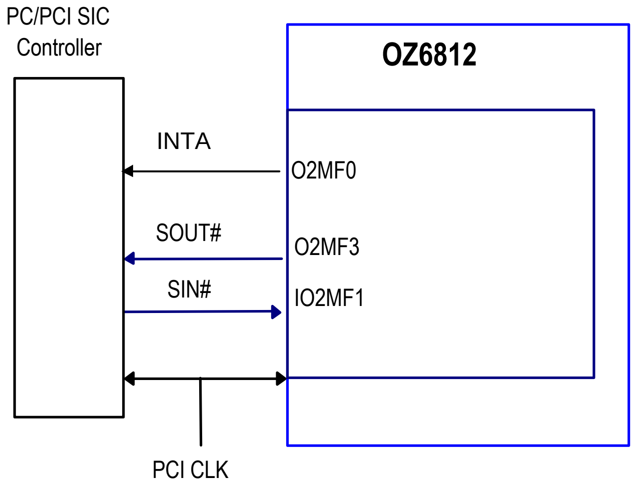


Figure 6. PC /PCI Interrupt Mode

3.5. Serialized IRQ (PCI/Way) Interrupt Mode

This operation mode uses the newly released PCI/Way single pin interrupt system. In this mode the OZ6812 provides an IRQ serializer pin to the PCI/Way compliant motherboard chip set interrupt controller. The IRQSER is a wired-or bi-directional serial interrupt line which replicates the state of each internal IRQ. The PCI INTA#, and IRQSER pins provides support for all ISA IRQs and PCI interrupts, which helps meet the Microsoft PC97 requirement.

In a system using the serialized IRQ protocol, the host software must program the O2Micro Mode Control Register C (offset 3Bh) to use serialized IRQs. The serialized interrupt protocol implemented in the OZ6812 uses a single pin 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. See Figure 7.

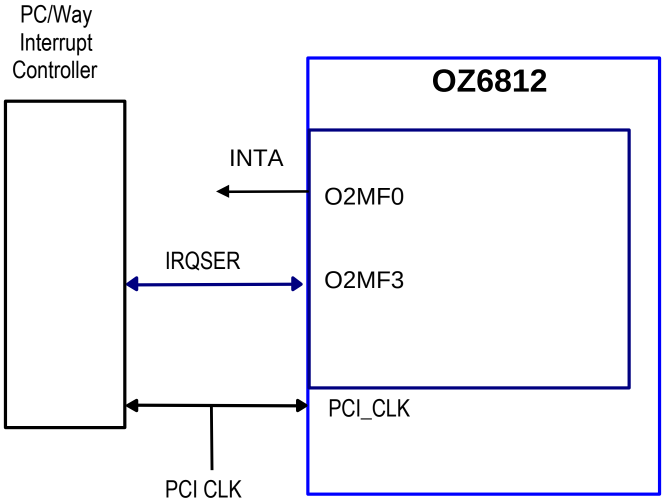


Figure 7. PCI/Way Interrupt Mode

Figures 8 and 9 illustrate how the serialized IRQ protocol works. Figure 8 shows the start cycle and the first several IRQ sampling periods. Figure 9 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: IRQO, IRQ1, SMI, IRQ3, IRQ4 … IRQ15, and IOCHCK. 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 IOCHCK period, but can be extended to allow more sampling periods for platform-specific functions.

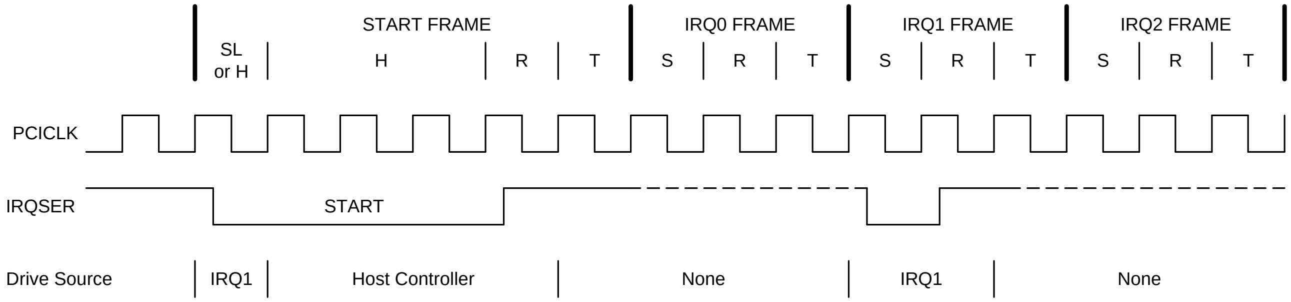


Figure 8. IRQ Protocol Start Cycle

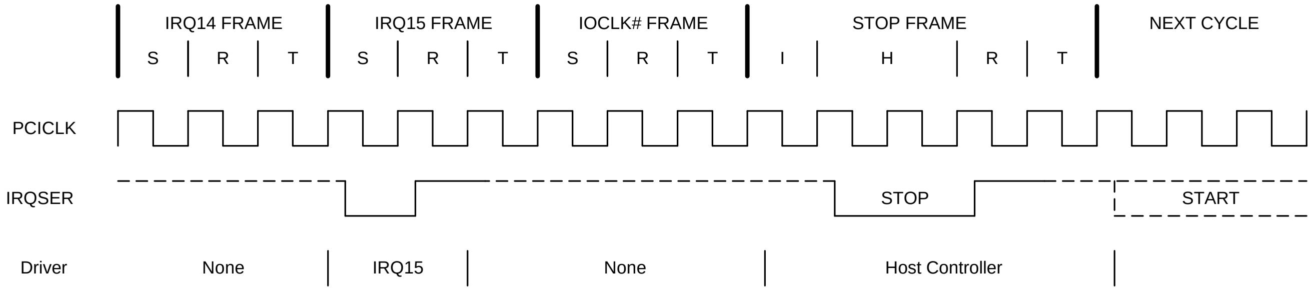


Figure 9. IRQ Protocol Stop Cycle

Legend:
H = Host Control T = Turn-around
SL = Slave Control S = Sample
R = Recovery I = Idle

3.6. VESA Serialized IRQ

This mode is very similar to the PCI/WAY Interrupt Mode. It also uses the IRQSER, PCI INTA#, INTB#, INTC# and INTD# pins to cover all ISA interrupts, IRQ0-IRQ15. This mode is enabled by setting the O2Micro Control1 Register offset D0h; bit [12:10] determines the IRQ routing method and bit[16] select a serial IRQ.

3.7. Win98 IRQ Support

In compliance to the latest PC 98 guidelines, O2Micro's OZ6812 supports simultaneous configuration of both ISA and PCI Interrupt modes for maximum design flexibility. OZ6812 CardBus Bridge is configurable to utilize ISA Interrupt mode for PC Card 16 cards and PCI Interrupt mode for PC Card 32 cards concurrently.

R2 Card (PC Card 16 card) is in either PC/PCI, PCI/Way or ISA IRQ Interrupt modes selectable through the System Interrupt Mode bits (bit [1:0] of O2Micro Mode Control D Register), while R3 Card (PC Card 32 card) are always in PCI Interrupt mode. For R2 Card with PC/PCI Interrupt mode, INTC# and INTD# are dedicated as SOUT# and SIN#, respectively.

If R2 Card is configured for PCI/Way Interrupt mode, IRQ5/SOUT#/ISLD/IRQSER is used as IRQSER interrupt while IRQ7/SIN#/ISDAT is not used. For R3 Card in PCI Interrupt mode, INTA# is used as interrupts for the socket.

4. CCLKRUN# Signal

The OZ6812 supports the PCI clock run protocol as defined in the PCI mobile design guide revision 1.0., and includes CardBus clock generation logic and a tri-statable input/output CCLKRUN# pin. This signal is multiplexed with the O2MF6 signal and is enabled by setting bit 2 of the ExCA O2Micro Control D Register .

The clock generation logic keeps CCLKRUN# asserted (low) during normal clock operation and continuously monitors it for requests from master or target devices for a change in the CardBus clock state. When a device signals the OZ6812 that the PCI clock is about to be stopped by driving CCLKRUN# high, the OZ6812 either signals that it is acceptable to stop the PCI clock by not driving CCLKRUN# or signals to keep the clock running by pulling CCLKRUN# low. An active CardBus Card may not permit the CardBus clock to be stopped, therefore requiring the OZ6812 to signal that the PCI clock must continue running. This gives the CardBus Card time to complete it's current task before the CardBus clock stops. If the CardBus clock is already stopped, the OZ6812 can allow the PCI clock to be stopped.

When in master mode, the OZ6812 clock generation logic drives high the CCLKRUN# pin for one clock to inform CardBus devices that the clock frequency is about to be stopped or slowed down. After driving CCLKRUN# high for one clock, the clock generation logic tri-states its CCLKRUN# output driver and begins to monitor the status. A pull-up resistor on the CCLKRUN# pin is required for maintaining the deasserted (high) state.

The CardBus Card clock (CCLK) can only be stopped or slowed down during PCI bus and CardBus socket idle periods and if the CardBus socket is powered. The CardBus Card must be powered, reset deasserted, and have no activity on the socket for eight CardBus clock cycles before requesting to slow or stop the CardBus Card clock. Activity on the socket is determined by monitoring the CFRAME, CIRDY, CREQ and CBLOCK signals from the CardBus Card. Any transaction requests from the PCI bus before completion of eight inactive clock cycles keeps the CardBus clock from slowing down or stopping.

The CardBus Card clock is automatically restarted to the PCI clock frequency when any PC Card is installed or removed from a socket or there is any activity on the CardBus Card interface or any accesses to the CardBus Card from the PCI bus. For 16-bit PC Cards, the PCI clock will be restarted when either IREQ, STSCHG/RI or DREQ signals are asserted. For CardBus cards, the PCI clock will be restarted when either CINT, CREQ, CCLKRUN or CSTSCHG are asserted.

5. Win95 Support

The OZ6812 is enumerated and configured in Win95, as any other PCI bus bridge. To obtain Win95 compatibility, the system BIOS is required to initialize the OZ6812 to Intel 82365 compatible mode and report it as device "PNP0E03, Intel 82365-compatible CardBus Controller" with a compatible ID of pnp0e00 and an I/O resource of two ports such as 3e0-3e1h. The following are the configuration space initialization steps required to put the OZ6812 into legacy mode:

1) Command register (offset 04h) is set to 07h to enable I/O, memory and bus mastering.
2) Register base address (offset 10h) is set to 0h.
3) All memory and I/O windows (offset 1c-38h) are set to 0h.
4) Interrupt line register (offset 3Ch) is set to FFh, no IRQ is assigned.
5) Legacy base address (offset 44h) is set to lecacy mode I/O base address.

Once configured for legacy mode, the built in Win95 socket services driver (Socketv.vxd) will access it as an Intel PCIC compatible controller at the base I/O address specified in step 5 above. The OZ6812 registers will respond identical to the industry standard ExCA base register set. Note that the Win95 PCI enumerator (Pci.vxd) does not know about the CardBus PCI header (Type 2), so it will not report the OZ6812 device at all.

5.1. Microsoft Windows Logo Compliance

The "Designed for Microsoft Windows" logo provides end users with the assurance that the PC hardware will work well with Windows and that it takes advantage of features built in to the operating system. Products that receive this logo go through a rigorous series of tests administered by the Windows Hardware Quality Labs (WHQL) to determine that the hardware meets all "Designed for Microsoft Windows" criteria.

Part of the logo testing requirements asserts that a product must run successfully on both Windows 95 and Windows NT unless prevented by architectural differences between the two operating systems. Windows 95 and Windows NT 4.0 share such technologies as the Win32 API, OLE, networking, and user interface, so it is possible to run applications based on a common API set and a common object technology. Both products contain the essential infrastructure for desktop management by using a system registry that is accessible remotely using Win32 APIs.

The OZ6812 is in full compliance with new combined Win95/NT Logo certification program. See Appendix 2 for a sample requirements checklist for logo certification.

6. ACPI – PCI BUS Power Management Interface

6.1. OnNow and ACPI

The objective for OnNow is to immediately make the PC ready when the user presses the power button. When not in use, the PC would be considered off but still capable of responding to wakeup events. This means that while the PC may appear to be off to the user, data and programs are reliably saved and excess energy is not consumed. Wakeup events would be triggered by device inputs such as a phone ringing or a timed alarm set by software.

The goal is to have the operating system and applications work together to intelligently operate the PC resulting in effective power management. All devices connected to the PC or added by the user are expected to participate in the device power management scheme. Any new device can have its power state changed as system usage dictates.

6.2. ACPI for PC 97

A key component of the OnNow design initiative is a new system board design based on the Advanced Configuration and Power Interface (ACPI) specification. ACPI defines a flexible and abstract hardware interface that enables a wide variety of PC systems to implement power and thermal management functions while still meeting the cost and feature requirements of the target market.

ACPI places power management under control of the Windows operating system instead of BIOS. The specification allows for a collection of power consumption information from the entire system and gives complete device activity control to Windows, enabling it to power devices on an as needed basis. In contrast, a BIOS based power management system depends on power consumption demand where devices can only be turned off after periods of inactivity. ACPI also provides device configuration and generic system event mechanisms for Plug and Play. ACPI therefore replaces the Plug and Play BIOS and unifies the power management interface with the Plug and Play interface. It also provides support for configurations and resource types that could not be supported before.

For OnNow, ACPI enables the operating system to direct power management throughout the PC system, improving integration, increasing effectiveness, and simplifying implementation. System designers have the freedom to implement a wide range of solutions, from the very simple to the very aggressive, while still maintaining full operating system support under both Windows 95 and Windows NT.

This pervasive implementation of power management enables applications to routinely implement support for these capabilities, introduce new OnNow features, and improve the effectiveness of power management in the PC.

6.3. OZ6812 ACPI Compliance

The OZ6812 implements power management based on activity on the primary PCI bus and/or the secondary bus on the CardBus or 16-bit Card. The OZ6812 will automatically enter into a lower power consumption state when memory and I/O windows are disabled or a socket becomes empty. Socket power management is achieved by programming the Power Management Register (A0h), Power Management Control/Status Register (A4h) and Power Control (ExCA index+02h) registers.

The CardBus specification provides power management capabilities for Card socket but not for the PCI bus interface. In addition. the PCI Local Bus Specification has no provision for supporting power management functionality. This has recently been addressed by PCI bus proposal 194 which specifies a standard set of PCI peripheral power management hardware interfaces and behavioral policies. This PCI bus power management specification combined with the CardBus power management specification fits well under the ACPI umbrella, enabling the operating system to intelligently manage the power of both PCI functions and PC Card socket.

The OZ6812 is compliant with the ACPI revision 1.0, the PCI Bus Power Management Interface Specification for PCI to CardBus Bridges and the Device Class Power Management Reference Specification – PC Card Controller Device Class. The OZ6812 implements the necessary additional Power Management Register Block in its configuration space (A0h and A4h) required for executing any one of the four power management states; ***D0***, ***D1, D2,*** ***D3***$_{hot}$, and ***D3***$_{cold}$. While in either ***D0***, ***D1***, ***D2***, or ***D3***$_{hot}$, the OZ6812 remains compliant with the PCI Local Bus Specification, Revision 2.1 and the PC Card Standard – Electrical Specification. See ACPI Power State Table in Appendix 3 for details.

D0 Active State:

This state is assumed to be the highest level of power consumption. The OZ6812 is completely active and responsive, and is expected to remember all relevant context continuously.

D0 Uninitialized State:

The OZ6812 must initially be put into the ***D0*** state before it can be initialized. Upon entering ***D0*** from power on reset, or transition from ***D3**$_{hot}$*, the OZ6812 will be in an uninitialized state unless **PME_En** is true, in which case PME context is retained. The operating system will program the PMCSR (Power Management Control Status Register). The D0 state can be entered from a D1 state as a result of inserting or removing a card or other activity which will cause the generation of an CINT# interrupt

D1 State:

This state can be considered as a “light sleep” state. Some parts of the OZ6812 may be processing background tasks such as monitoring the PCI bus and PC Card socket for activity. In general, D1 is expected to save less power and preserve more device context than D2.

D2 State:

When the OZ6812 and its associated cards are not being used, it may be put into ***D2***. In this state the OZ6812 will retain the ability to fully recover to its previous condition while providing significant power savings. In this state the only PCI bus operation that the OZ6812 will initiate is a power management event (PME) and will only respond to PCI configuration accesses (memory and I/O registers are disabled). The D2 state saves more power and preserves less device context than D1 or D0. ACPI system software must restore the OZ6812 to a ***D0*** active before any memory or I/O space can be accessed.

D3 State:

The OZ6812 transitions into a ***D3*** state either by software (***D3**$_{hot}$*,) or by physically removing power (***D3**$_{cold}$*). When in **D3**$_{hot}$, the OZ6812 transitions to an uninitialized ***D0*** state via software by writing to it’s *PMCSR* register or by having it’s Bus Segment Reset (PCI **RST#**, CardBus card **CRST#**) asserted. When in the ***D3**$_{cold}$* state it can only be transitioned to an uninitialized ***D0*** state by reapplying Vcc and asserting Bus Segment Reset (**RST#**). When the OZ6812 is brought back to ***D0*** (the only legal state transition from ***D3***), software will need to perform a full reinitialization, including its PCI configuration space.

Software Accessible D3 (D3$_{hot}$):

When in ***D3**$_{hot}$* the OZ6812 will respond to configuration space accesses as long as power and clock are supplied. Any wake event will request Windows to bring back the OZ6812 to the full on state, ***D0***, through the PME#. When programmed to ***D0*** the OZ6812 will perform the equivalent of an internal warm (soft) reset, eliminating the requirement of a hardware reset (PCI **RST#** and **CRST#**/**RESET** need not be asserted). During this transition the OZ6812’s PCI and CardBus bus signal drivers remain disabled to avoid software hang conditions.

Power Off (D3$_{cold}$)

By default the OZ6812 transitions immediately to ***D3**$_{cold}$* upon removal of all Vcc power. A full PCI 2.1 compliant power-on reset sequence (PCI **RST#** asserted) is required to restore the OZ6812 to ***D0*** (***D0*** Uninitialized state). The power-on defaults must be restored to the OZ6812 by hardware whenever the transition from ***D3*** to ***D0*** is initiated through assertion of PCI **RST#**,. It must then be fully initialized and reconfigured by software after making the transition to the ***D0*** uninitialized state. The OZ6812 can not preserve the PME context when going through the ***D3**$_{cold}$* to ***D0*** transition because it does not support 3.3Vaux wake-up mechanism in ***D3**$_{cold}$* .

6.4. Power State Transitions

All OZ6812 power management state changes are explicitly controlled by software except for hardware reset, which brings all functions to the ***D0*** Uninitialized state. Table 2 shows the values to be programmed in the PMCSR.

State	PMCSR bit [1:0]	Description
D0`	00	
D1	01	
D2	10	
D3-hot	11	
D3-cold	11	Remove all Vcc

Table 2. Available Power States

7. Power Management Events (PME#)

A power management event is the process by which the OZ6812 requests a change of its power consumption state. The OZ6812 will assert a ***Power Management Event*** (**PME#)** whenever it generates or detects an event that will require the system to change its power state. For example, when the OZ6812 is in a powered down state, a detected telephone ring will require a power state change**.** It will continue to drive **PME#** low (even if the source of the power management event is no longer valid) until power management software either clears (logic "1") the **PME_En** bit or clears the **PME_Status** bit in the *PMCSR*. When PME is enabled and **PME_En** set, interrupts normally associated with controller status, change interrupts, are routed to the **PME#** pin instead of the controller status change interrupt. CardBus card functional and status change interrupts are not functional except in the ***D0*** state.

The **PME#** signal is an open drain, active low signal driven by the OZ6812. This signal can be routed to any one of the OZ6812's 13 IRQs or power control pins.

It is the responsibility of the system designer to route this PME# signal to the appropriate system logic to wake the system. For example, in an ACPI compliant system, it may be routed to the SCI# interrupt. The asynchronous **PME#** signal is bussed within the system in the same way that PCI Bus functional interrupt requests are routed. All sources of **PME#** are connected together (wire OR'ed) to present a single point of connection into the system. When this signal is used, the system designer must provide a pull-up resistor; otherwise it is left as a no connect. The value of this resistor is derived by taking into account the output capacitance of the open drain driver to ensure that **PME#** charges up to a logic high voltage level in less than 100 ns.

See Section 4.3.3 of the PCI Local Bus Specification Revision 2.1 for information on pull-up resistors.

8. Interface I/O Register Addressing

All OZ6812 Socket Status & Control Registers for CardBus Cards can be accessed through PC Card Socket Registers/Control Base Register (PCI Configuration Register 10h) offset 00h to 7FFh. The ExCA registers implemented in the OZ6812 can be accessed also via PC Card Socket Status/Control Base Register offset 800h to FFFh or via the 16-bit PC Card Legacy Mode Base Address Register (PCI Configuration Register 44h) for 16-bit PC Cards.

For the PC Card 16-Bit Interface Legacy Mode Base Address Register, (PCI Configuration Register 44h) the first I/O address is the OZ6812's index register. The second I/O address (PC Card 16-Bit Interface Legacy Mode Base Address Register + 01) is the OZ6812's data register.

The index register and the data register are read/write registers. The OZ6812 will not respond to a data register read or write operation or to an index register read operation unless the index register has first been written to with a valid index.

9. Write Buffer and CardBus Master Mode

The OZ6812 is a PCI-TO-PCI bridge type CardBus controller. Since the CardBus and PCI are running at the same clock speed and are the same bus width, there is basically no performance difference between the two.

The OZ6812 supports full depth 32-bit write buffers for PCI-To-CardBus and CardBus-To-PCI master modes. The OZ6812 can perform the zero wait state burst write to Cardbus cards and Cardbus master cards burst write to PCI Bus.

10. PCI 2.1 Subsystem Vendor ID to meet PC 98 requirements

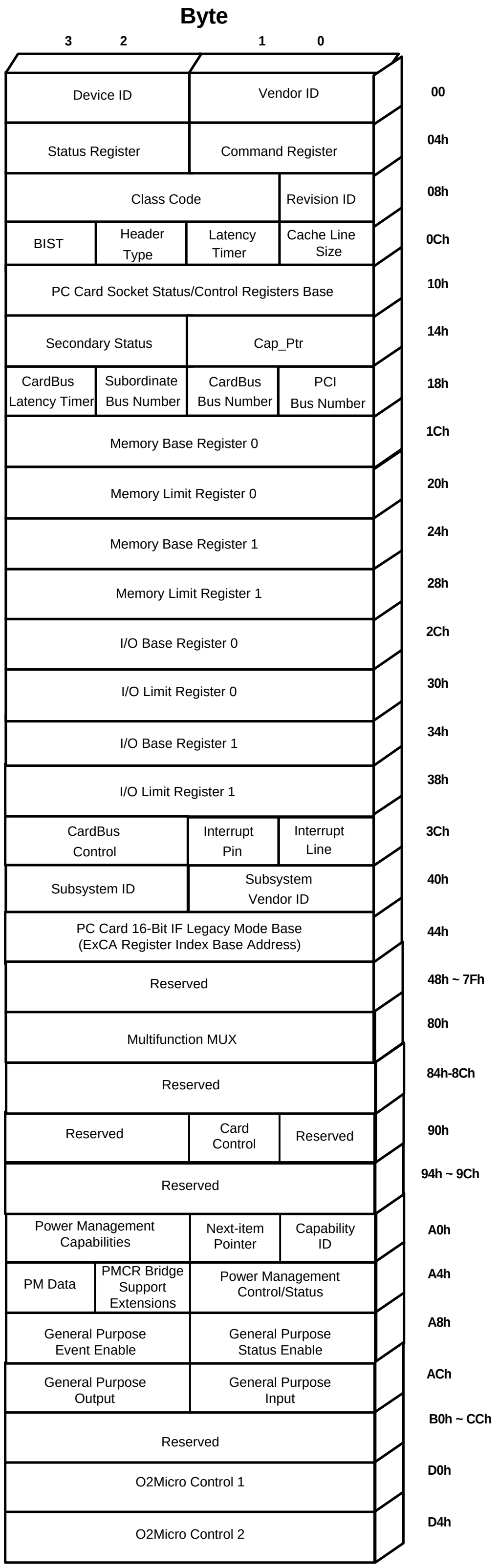
The OZ6812 meets the PC 98 requirements for Subsystem Vendor ID support. OZ6812 implements a write-enable bit in the PCI user-defined space. The BIOS can turn this bit on, change the Subsystem Vendor ID, then turn it off.

The OZ6812 implements the Write once- Read only PCI Subsystem Vendor ID register . If BIOS writes the value to this register, it locks the value then becomes the read only register till you turn off the auto-lock bit.

PCI CONFIGURATION REGISTERS

The OZ6812 is a single-function device. It is defined as closely as possible to a PCI-to-PCI Bridge device. PCMCIA ExCA register are accessed through either Memory Base Address Register or the Legacy Mode Base Address Register.

CAUTION: It bits indicated as read only (R:) are to be written to, they should be written to zero.



Byte

3	2	1	0	
Device ID		Vendor ID		00
Status Register		Command Register		04h
Class Code			Revision ID	08h
BIST	Header Type	Latency Timer	Cache Line Size	0Ch
PC Card Socket Status/Control Registers Base				10h
Secondary Status		Cap_Ptr		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
CardBus Control		Interrupt Pin	Interrupt Line	3Ch
Subsystem ID		Subsystem Vendor ID		40h
PC Card 16-Bit IF Legacy Mode Base (ExCA Register Index Base Address)				44h
Reserved				48h ~ 7Fh
Multifunction MUX				80h
Reserved				84h-8Ch
Reserved		Card Control	Reserved	90h
Reserved				94h ~ 9Ch
Power Management Capabilities		Next-item Pointer	Capability ID	A0h
PM Data	PMCR Bridge Support Extensions	Power Management Control/Status		A4h
General Purpose Event Enable		General Purpose Status Enable		A8h
General Purpose Output		General Purpose Input		ACh
Reserved				B0h ~ CCh
O2Micro Control 1				D0h
O2Micro Control 2				D4h

Vendor ID Register (Offset : 00h)

BIT POSITION	NAME	DESCRIPTION
15:0.	Vendor ID	This read-only field is the Vendor identification assigned to O2Micro by the PCI Special Interest Group. This field will always read back 1217h.

Device ID Register (Offset : 02h)

BIT POSITION	NAME	DESCRIPTION
15:0	Device ID	This read-only field is the device identification assigned to this device by 02Micro. This field will always read back 6872 for the OZ6812. (Revision number identification for the OZ6812 part itself is indicated by the Revision ID field in the **Revision ID** and **Class Code** register at offset 08h).

Command Register (Offset : 04h)

BIT POSITION	NAME	DESCRIPTION
15:10	Reserved	These bits are read only and return 0's when read. Writes have no effect.
9	Fast Back-to-Back Enable	The OZ6812 will not generate fast back-to-back transactions; therefore, this bit is READ only. This bit returns 0's when read. 0 = Disable Fast back-to-back function. 1 = Not Allow.
8	System Error(SERR#) Enable	This bit controls the enable for the SERR# driver on the PCI interface. SERR# can be asserted after detecting an address parity error on the PCI bus. Both this bit and bit 6 must be set for the OZ6812 to report address parity errors. 0 = Disable the SERR# response **(default).** 1 = Enable the SERR# response.
7	Wait Cycle Enable	This bit is used to control whether a device performs address/data stepping. OZ6812 has this bit read/write and is initialized to 1 after RST#. 0 = Disables OZ6812's ability to perform address/data stepping **(default).** 1 = Enables OZ6812's ability to perform address/data sleeping.
6	Parity Error Response/ Report (PERR#) Enable	This bit controls the OZ6812's response to parity errors. PERR# can be asserted after detecting an address parity error on the PCI bus. Both this bit and bit 8 must be set for the OZ6812 to report address padty errors. 0 = Disables OZ6812's Parity Error Response/Report function **(default).** 1 = Enables OZ6812's Parity Error Response/Report function.
5	VGA Palette Snoop	This bit controls how PCI devices handle accesses to VGA palette registers. The OZ6812 does not support VGA palette snooping; therefore, this bit is hard wired to 0. This bit is read only and returns 0 when read. Writes to this bit have no effect. 0 = Disables OZ6812's VGA Palette Snoop function. 1 = Not Allow.
4	Memory Write and Invalidate Enable	This is an enable bit for using the Memory Write and Invalidate command. When this bit is 1, OZ6812 generates the command. When it is 0, Memory Write will be used instead. State after RST# is 0. 0 = Disables Memory Write and Invalidate command **(default).** 1 = Enables Memory Write and Invalidate command.
3	Special Cycles	This bit controls whether or not a PCI device ignores PCI Special cycles. The OZ6812 does not respond to Special Cycle operations; therefore, this bit is hard wired to 0. This bit is read only and returns 0 when read. Writes to this bit have no effect. 0 = Disables response to Special Cycle operations. 1 = Not Allow.
2	Bus Master Control	This bit controls whether or not the OZ6812 can act as a PCI bus initiator (master). The OZ6812 can take control of the PCI bus only when this bit is set. 0 = Disables the OZ6812's ability to generate PCI bus accesses **(default).** 1 = Enables the OZ6812's ability to generate PCI bus accesses.
1	PCI Memory Space Enable	This bit controls whether or not the OZ6812 may claim cycles in PCI memory space. 0 = Disables the OZ6812's response to memory space accesses **(default).** 1 = Enables the OZ6812's response to memory space accesses.
0	I/O Space Enable	This bit controls whether or not the OZ6812 may claim cycles in PCI I/O space. 0 = Disables the OZ6812's response to I/O space accesses **(default).** 1 = Enables the OZ6812's response to I/O space accesses.

Status Register (Offset : 06h)

BIT POSITION	NAME	DESCRIPTION
15	Address/Data Parity Error Detected	This bit indicates a parity error, either address or data, has been detected. 0 = Address or data parity error not detected **(default).** 1 = Address or data parity error detected.
14	System Error (SERR#) Generated	This bit is set when SERR# is enabled and the OZ6812 signaled a system error to the host. 0 = System error not signaled to the host **(default).** 1 = System error signaled to the host.
13	Received Master Abort	This bit is set when a cycle initiated by the OZ6812 on the PCI bus has been terminated by a master abort. 0 = No Master Abort received **(default)**. 1 = Received Master Abort.
12	Received Target Abort	This bit is set when a cycle initiated by the OZ6812 on the PCI bus was terminated by a target abort. 0 = No Target Abort received **(default)**. 1 = Received Target Abort.
11	Signaled Target Abort	This bit is set by the OZ6812 when it terminates a transaction on the PCI bus with a target abort. 0 = No Target Abort signaled **(default)**. 1 = Target Abort signaled.
10:9	DEVSEL# Timing	These read only bits encode the timing of DEVSEL and are hardwired to 10b.
8	Master Data Parity Error Reported	This bit indicates OZ6812's response to CardBus data parity errors. 0 = No CardBus data parity errors response made **(default)**. 1 = Response to CardBus data parity error when any of the following conditions are met: a. PERR# was asserted on the CardBus interface. b. The OZ6812 was the bus master during the data parity error c. The parity error response bit is set in the Command Control register (04h).
7:5	Reserved.	These bits are read only and return 0's when read. Writes have no effect.
4	Capabilities List	This bit indicates that capabilities in addition to standard PCI capabilities are implemented. This bit is read only and returns 1 when read. The linked list of PCI Power Management Capabilities is implemented in this function. 0 = Not Allow. 1 = PCI Power Management Capabilities are implemented.
3:0	Reserved	These bits are read only and return 0's when read. Writes have no effect.

Revision ID Register (Offset : 08h)

BIT POSITION	NAME	DESCRIPTION
7:0	Revision ID	This read only register indicates the silicon revision of the OZ6812. ex. "02"- Revision A.

Class Code Register (Offset : 09h)

BIT POSITION	NAME	DESCRIPTION
23:0	Class Code	This read-only field identifies the OZ6812 as a PCI-to PCMCIA bridge device. It will read back 060700h.

Cache Line Size Register (Offset : 0Ch)

BIT POSITION	NAME	DESCRIPTION
7:0	Cache Line Size	This register is programmed by host software to indicate the system cache line size.

Latency Timer Register (Offset : 0Dh)

BIT POSITION	NAME	DESCRIPTION
7:0	Latency Timer	This register specifies the latency timer for the 0Z6812, in units of PCI clock cycles. When the OZ6812 is a PCI bus initiator and asserts FRAME# the latency timer will begin counting from zero. If the latency timer expires before the OZ6812 transaction has terminated, then the OZ6812 will terminate the transaction when its GNT#is deasserted.

Header Type Register (Offset : 0Eh)

BIT POSITION	NAME	DESCRIPTION
7:0	Header Type	This read only register returns 8h when read, indicating that the OZ6812 configuration spaces adhere to the CardBus bridge PCI header. The CardBus bridge PCI header ranges from PCI register 0 to 7Fh. 80h to FFh are user definable registers.

BIST Register (Offset : 0Fh)

BIT POSITION	NAME	DESCRIPTION
7:0	BIST Type	Since the OZ6812 does not support a built-in self test, this register is read only and returns the value of 00h when read.

PC Card Socket Registers/Control Base Register (Offset : 10h)

BIT POSITION	NAME	DESCRIPTION
31:0	Control Base	This register is programmed with a base address referencing the CardBus socket registers and the memory mapped ExCA register set. Bits 31:12 are read/write, and allow the base address to be located anywhere in the 32-bit PCI memory address space on a 4 Kilobyte boundary. Bits 11:0 are read only, returning 0's when read. When software writes all 1 's to this register, the value read back will be FFFF F000h, indicating that at least 4 Kilobytes of memory address space are required. The CardBus registers start at offset 000h, and the memory mapped ExCA registers begin at offset 800h.

Capabilities Pointer, Cap_Ptr (Offset : 14h)

BIT POSITION	NAME	DESCRIPTION
15:8	Reserved	These bits are read only and return 0's when read. Writes have no effect.
7:0	Cap_Ptr	This register provides a pointer into the PCI configuration header where the PCI power management registerblock resides. PCI header double-words at A0h and A4h provide the power management (PM) registers. This register is read only and returns A0h when read.

CardBus Status Register (Offset: 16h)

BIT POSITION	NAME	DESCRIPTION
15	Address/Data Parity Error Detected	This bit is set when a CardBus parity error is detected; either address or data parity errors.
14	System Error (CSERR#) Generated	This bit is set when CSERR# is signaled by a CardBus card, The OZ6812 does not assert the CSERR# signal.
13	Received Master Abort	This bit is set when a cycle initiated by the OZ6812 on the CardBus bus has been terminated by a master Abort.
12	Received Target Abort	This bit is set when a cycle initiated by the OZ6812 on the CardBus bus with a target abort.
11	Signaled Target Abort	This bit is set by the OZ6812 when it terminates a transaction on the CardBus bus with a target abort.
10:9	CDEVSEL# Timing	These read only bits encode the timing of CDEVSEL# and are hardwired 01b indicating that the OZ6812 asserts this signal at a medium speed.
8	Master Data Parity Error Reported	This bit controls OZ6812's response to CardBus parity errors. 0 = No CardBus data parity errors response made **(default)**. 1 = Response to CardBus data parity error when any of the following conditions are met: a. CPERR# was asserted on the CardBus interface. b. The OZ6812 was the bus master during the data parity error. c. The parity error response bit is set in the Command Control register (3Eh).
7:0	Reserved	These bits are read only and return 0's when read. Writes have no effect.

PCI Bus Number Register (Offset : 18h)

BIT POSITION	NAME	DESCRIPTION
7:0	PCI Bus Number	This read/write register is programmed by the host system to indicate the bus number of the PCI bus to which the OZ6812 is connected. The OZ6812 uses this register, in conjunction with the CardBus Bus Number and Subordinate Bus Number registers, to determine when to forward PCI configuration cycles to its secondary bus.

CardBus Bus Number Register (Offset : 19h)

BIT POSITION	NAME	DESCRIPTION
7:0	CardBus Bus Number	This read/write register is programmed by the host system to indicate the bus number of the CardBus bus. The OZ6812 uses this register, in conjunction with the PCI Bus Number and Subordinate Bus Number registers, to determine when to forward PCI configuration cycles to its secondary bus.

Subordinate Bus Number Register (Offset : 1Ah)

BIT POSITION	NAME	DESCRIPTION
7:0	Subordinate Bus Number	This read/write register is programmed by the host system to indicate the highest numbered bus below the CardBus bus. The OZ6812 uses this register, in conjunction with the PCI Bus Number and CardBus bus Number registers, to determine when to forward PCI configuration cycles to its secondary bus.

CardBus Latency Timer Register (Offset : 1Bh)

BIT POSITION	NAME	DESCRIPTION
7:0	CardBus Latency Timer	This read/write register is programmed by the host system to specify the latency timer for the OZ6812 CardBus interface, in units of CCLK# cycles. When the OZ6812 is a CardBus initiator and asserts CFRAME#, the CardBus latency timer will begin counting. If the latency timer expires before the OZ6812 transaction has terminated, then the OZ6812 will terminate the transaction at the end of the next data phase. A recommended minimum value for this register is 20h, and will allow most transactions to be completed.

Memory Base Registers 0,1 (Offset : 1Ch, 24h)

BIT POSITION	NAME	DESCRIPTION
31:0	Memory Base 0,1	These registers indicate the lower address of a PCI memory address range and are used by the OZ6812 to determine when to forward a memory transaction to the CardBus bus, and likewise, when to forward a CardBus cycle to PCI. Bits 31-12 of these registers are read/write and allow the memory base to be located anywhere in the 32-bit PCI memory space on 4 Kilobyte boundaries. Bits 11-0 are read only and always returns O's. The memory base register or the memory limit register must be non-zero in order for the OZ6812 to claim any memory transactions through CardBus memory windows (i.e. these windows are not enabled by default to pass the first 4 Kilobytes of memory to CardBus)

Memory Limit Registers 0,1 (Offset : 20h, 28h)

BIT POSITION	NAME	DESCRIPTION
31:0	Memory Limit 0,1	These registers indicate the upper address of a PCI memory address range and are used by the OZ6812 to determine when to forward a memory transaction to the CardBus bus, and likewise, when to forward a CardBus cycle to PCI. Bits 31-12 of these registers are read/write and allow the memory base to be located anywhere in the 32-bit PCI memory space on 4 Kilobyte boundaries. Bits 11-0 are read only and always returns O's. Writes to these bits have no effect. Bits 8 and 9 of the Bridge Control Register specify whether Memory Windows 0 and 1 are prefetchable or non-prefetchable. The memory base register or the memory limit register must be non-zero in order for the 0Z6812 to claim any memory transactions through CardBus memory windows (i.e. these windows are not enabled by default to pass the first 4 Kilobytes of memory to CardBus)

I/O Base Registers 0,1 (Offset : 2Ch, 34h)

BIT POSITION	NAME	DESCRIPTION
31:0	I/O Base 0,1	These registers indicate the lower address of a PCI I/O address range and are used by the OZ6812 to determine when to forward an I/O transaction to the CardBus bus, and likewise, when to forward a CardBus cycle to the PCI bus. The lower 16 bits of this register locate the bottom of the I/O window within a 64 Kilobyte page, and the upper 16 bits are a page register which locates this 64 Kilobyte page in 32-bit PCI I/O address space. Bits 15-2 are read/write and allow the I/O limit address to be located anywhere in the 64 Kilobyte page (indicated by bits 31-16 of the appropriate I/O base register) on doubleword boundaries. Bits 31-16 are read only and always return 0's when read. Bits 1-0 are read only and always return "01". **Note:** The I/O base and the I/O timit register must be non-zero to enable any I/O transactions.

I/O Limit Register 0,1 (Offset : 30h, 38h)

BIT POSITION	NAME	DESCRIPTION
31:0	I/O Limit 0,1	These registers indicate the upper address of a PCI I/O address range and are used by the OZ6812 to determine when to forward an I/0 transaction to the CardBus bus, and likewise, when to forward a CardBus cycle to PCI. The lower 16 bits of this register locate the top of the I/O window within a 64 Kilobyte page, and the upper 16 bits are a page register which locates this 64 Kilobyte page, and the upper 16 bits are a page register which locates this 64 Kilobyte page in 32-bit PCI I/O address space. Bits 15-2 are read/write and allow the I/O limit address to be located anywhere in the 64 Kilobyte page (indicated by bits 31-16 of the appropriate I/O base register) on doubleword boundaries. Bits 31-16 are read only and always return 0's when read. Bits 1-0 are read only and always return "01". **Note:** The I/O base and the I/O limit register must be non-zero to enable an I/O transaction

Interrupt Line Register (Offset : 3Ch)

BIT POSITION	NAME	DESCRIPTION
7:0	Interrupt Line	This register is read/write programmed by BIOS/OS. It indicates INTA# routing information.

Interrupt Pin Register (Offset : 3Dh)

BIT POSITION	NAME	DESCRIPTION
7:0	Interrupt Pin	This is a read only register and returns 01 h. It indicates INTA# is being used by an interrupt signaling mode.

Bridge Control Register (Offset : 3Eh)

BIT POSITION	NAME	DESCRIPTION
15:11	Reserved	These bits are read only and return 0's when read. Writes have no effect.
10	Write Posting Enable	Enables write posting to and from the CardBus socket. Write posting enables posting of write data on burst cycles. Operating with write posting disabled will inhibit performance on burst cycles. Note that bursted write data can be posted, but various write transactions may not. 0 = Disable OZ6812's Write Posting function. **(default)** 1 = Enables OZG812's Write Posting function.
9	Memory 1 Prefetch Enable	This bit specifies whether or not memory window 1 is prefetchable. This bit is encoded as: 0 = Memory window 1 is nonprefetchable. 1 = Memory window 1 is prefetchabie **(default)**.
8	Memory 0 Prefetch Enable	This bit specifies whether or not memory window 0 is prefetchable. This bit is encoded as: 0 = Memory window 0 is nonprefetchable. 1 = Memory window0 is prefetchanble **(default)**.
7	IREQ-INT Enable	This bit is used to select whether PC Card functional interrupts are routed to PCI interrupts or to the IRQ specified in the ExCA registers. 0 = Functional interrupts routed to PCI interrupts **(default)**. 1 = Functional interrupts routed by ExCA registers.
6	CardBus Reset	When this bit is set the CRST# signal is asserted on the CardBus interface. The CRST# signal may also be asserted by passing a PCIRST# assertion to Cardbus. 0 = CRST# deasserted. 1 = CRST# asserted **(default)**.

BIT POSITION	NAME	DESCRIPTION
5	Master Abort Mode	This bit controls how the OZ6812 responds to a master abort when the OZ6812 is an initiator on the CardBus interface. 0 = Master aborts not reported **(default)**. 1 = Signal target abort on PCI and signal SERR#, if enabled.
4	Reserved	These bits are read only and return 0's when read. Writes have no effect.
3	VGA Enable	This bit affects how the OZ6812 responds to VGA addresses. When this bit is set, accesses to VGA addresses will be forwarded, meaning 7A000hBFFFFh, I/O 3B0h to 3BBh, and 3C0h to 3DFh.
2	ISA Enable	This bit affects how the OZ6812 passes I/O cycles within the 64 Kbyte ISA range. When set, the OZ6812 will not forward the last 768 bytes of each 1K I/O range to CardBus. 0 = ISA not enabled **(default)**. 1 = ISA Enabled.
1	CSERR#	This bit controls the response of the OZ6812 to CSERR# signals on the CardBus bus. 0 = CSERR# is not forwarded to PCI SERR# **(default)**. 1 = CSERR# is forwarded to PCI SERR#.
0	Parity Error Response Enable	This bit controls the response of the OZ6812 to CardBus parity errors. 0 = CardBus parity errors are ignored **(default)**. 1 = CardBus parity errors are reported using CPERR#.

Subsystem Vendor ID Register (Offset : 40h)

BIT POSITION	NAME	DESCRIPTION
15:0	Subsystem Vendor	This register is used for system and option card identification purposes, and may be required for certain operating systems. This register is write once only. After the first write, it becomes read only. To write a different value, PCI reset must be asserted.

Subsystem ID Register (Offset : 42h)

BIT POSITION	NAME	DESCRIPTION
15:0	Subsystem ID	This register is used for system and option card identification purposes, and may be required for certain operating systems. This register is write once only. After the first write, it becomes read only. To write a different value, PCI reset must be asserted.

PC Card 16 Bit I/F Legacy Mode Base Address Register (Offset: 44h) (ExCA Register Index Base Address)

BIT POSITION	NAME	DESCRIPTION
31:0	PCCard 16 Bit I/F Legacy Mode Base Address Register	The OZ6812 supports the Index/Data scheme of accessing the ExCA registers, which is mapped by this register. An address written to this register is the address for the Index register and address+l is the Data address. Using this access method, applications requiring Index/Data ExCA access can be supported. The base address can be mapped anywhere in 32-bit I/O space on a doubleword boundary; hence, bit 1-0 are read only, returning 01 b when read. Refer to the ExCA register set description for register offsets.

Multifunction MUX (Offset :8Ch)

BIT POSITION	NAME	DESCRIPTION
31:28	Reserved	These bits are read only and return 0's when read. Writes have no effect.
27:24	MFCFG6	Multifunction terminal 6 (MF6) configuration. These bits indicate the internal routing to the MF6 pin. 0000 = Reserved 0001 = CLKRUN# PCI CLKRUN# signal 0010 = Reserved 0011 = IRQ3 Parallel ISA IRQ3 0100 = IRQ4 Parallel ISA IRQ4 0101 = IRQ5 Parallel ISA IRQ5 0110 = Reserved 0111 = IRQ7 Parallel ISA IRQ7 1000 = Reserved 1001 = IRQ9 Parallel ISA IRQ9 1010 = IRQ10 Parallel ISA IRQ10 1011 = IRQ11 Parallel ISA IRQ11 1100 = IRQ12 Parallel ISA IRQ12 1101 = Reserved 1110 = IRQ14 Parallel ISA IRQ14 1111 = IRQ15 Parallel ISA iRQ15
23:20	MFCFG5	Multifunction terminal 5 (MF5) configuration. These bits indicate the internal routing to the MF5 pin. 0000 = GPI4 General purpose input **(default)** 0001 = GPO4 General purpose output 0010 = DMA GNT# PC/PCI DMA grant signal 0011 = IRQ3 Parallel ISA IRQ3 0100 = IRQ4 Parallel ISA IRQ4 0101 = IRQ5 Parallel ISA IRQ5 0110 = ZVSTAT Zoom video status output 0111 = ZVSEL0 Zoom video select output 1000 = Reserved 1001 = IRQ9 Parallel ISA IRQ9 1010 = IRQ10 Parallel ISA IRQI0 1011 = IRQ11 Parallel ISA IRQ11 1100 = LED_SKT SKTACTV/LEDOUT output 1101 = ZV ACTIVITY Zoom video activity 1110 = GF'E# General purpose event signal 1111 = IRQ15 Parallel ISA IRQ15
19:16	MFCFG4	Multifunction terminal 4 (MF4) configuration. These bits indicate the internal routing to the MF4 pin. 0000 = GPI3 General purpose input **(default)** 0001 = GPO3 General purpose output 0010 = PCI LOCK# PCI lock signal 0011 = IRQ3 Parallel ISA IRQ3 0100 = IRQ4 Parallel ISA IRQ4 0101 = IRQ5 Parallel ISA IRQ5 0110 = ZVSTAT Zoom video status output 0111 = ZVSEL0 Zoom video Select output 1000 = Reserved 1001 = IRQ9 Parallel ISA IRQ9 1010 = IRQ10 Parallel ISA IRQ10 1011 = IRQ11 Parallel ISA IRQ11 1100 = RI_OUT# Ring indicate output 1101 = LED SKT SKTACTV/LEDOUT output 1110 = GPE# General purpose event signal 1111 = IRQ15 Parallel ISA IRQ15

BIT POSITION	NAME	DESCRIPTION
15:12	MFCFG3	Multifunction terminal 3 (MF3) configuration. These bits indicate the internal routing to the MF3 pin. 0000 = Reserved 0001 = IRQSER Serial IRQ 0010 = SOUT# PC/PCI interrupt output signal 0011 = IRQ3 Parallel ISA IRQ3 0100 = IRQ4 Parallel ISA IRQ4 0101 = IRQ5 Parallel ISA IRQ5 0110 = Reserved 0111 = IRQ7 Parallel ISA]RQ7 1000 = Reserved 1001 = IRQ9 Parallel ISA IRQ9 1010 = IRQ10 Parallel ISA IRQ10 1011 = IRQ11 Parallel ISA IRQ11 1100 = IRQ12 Parallel ISA iRQ12 1101 = ZV ACTIVITY Zoom video activity 1110 = IRQ14 Parallel ISA IRQ14 1111 = IRQ15 Parallel ISA IRQ15
11:8	MFCFG2	Multifunction terminal 2 (MF2) configuration. These bits indicate the internal routing to the MF2 pin. 0000 = GPI2 General purpose input **(default)** 0001 = GPO2 General purpose output 0010 = DMAREQ# PC/PCI DMA request signal 0011 = IRQ3 Parallel ISA IRQ3 0100 = IRQ4 Parallel ISA IRQ4 0101 = IRQ5 Parallel ISA IRQ5 0110 = ZVSTAT Zoom video status output 0111 = ZVSEL0 Zoom video select output 1000 = Reserved 1001 = IRQ9 Parallel ISA IRQ9 1010 = IRQ10 Parallel ISA IRQ10 1011 = IRQ11 Parallel ISA IRQ11 1100 = RI OUT# Ring indicate output 1101 = ZV ACTIVITY Zoom video activity 1110 = GPE# General purpose event signal 1111 = IRQ7 Parallel ISA IRQ7
7:4	MFCFG1	Multifunction terminal 1 (MF1) configuration. These bits indicate the internal routing to the MF1 pin. 0000 = GPI1 General purpose input **(default)** 0001 = GPO1 General purpose output 0010 = SIN# PC/PCI serial interrupt input signal 0011 = IRQ3 Parallel ISA IRQ3 0100 = IRQ4 Parallel ISA IRQ4 0101 = IRQ5 Parallel ISA IRQ5 0110 = ZVSTAT Zoom video status output 0111 = ZVSEL0 Zoom video select output 1000 = Reserved 1001 = IRQ9 Parallel ISA IRQ9 1010 = IRQ10 Parallel ISA IRQ10 1011 = IRQ11 Parallel ISA IRQ11 1100 = LED SKT SKTACTV/LEDOUT output 1101 = ZV ACTIVITY Zoom video activity 1110 = GPE# General purpose event signal 1111 = IRQ15 Parallel ISA IRQ15

BIT POSITION	NAME	DESCRIPTION
3:0	MFCFG0	Multifunction terminal 0 (MF0) configuration. These bits indicate the internal muting to the MF0 pin. 0000 = GPI0 General purpose input **(default)** 0001 = GPO0 General purpose output 0010 = INTA PCI interrupt signal, INTA 0011 = IRQ3 Parallel ISA iRQ3 0100 = IRQ4 Parallel ISA IRQ4 0101 = IRQ5 Parallel ISA iRQ5 0110 = ZVSTAT Zoom video status output 0111 = ZVSEL0 Zoom video select output 1000 = Reserved 1001 = IRQ9 Parallel ISA IRQ9 1010 = IRQ10 Parallel ISA IRQ10 1011 = IRQ11 Parallel ISA IRQ11 1100 = LED_SKT SKTACTV/LEDOUT output 1101 = ZV ACTIVITY Zoom video activity 1110 = GPE# General purpose event signal 1111 = IRQ15 Parallel ISA IRQ15

Card Control Register (Offset : 91h)

BIT POSITION	NAME	DESCRIPTION
7	RIOUT_SEL	This bit controls the RI_OUT function. 0 = Disable RI_OUT. **(default)** 1 = Enable RI_OUT.
6:0	Reserved	These bits are read only and return 0's when read. Writes have no effect.

Power Management Register (Offset : A0h) Read Only

BIT POSITION	NAME	DESCRIPTION
31:27	PME_Support	This five-bit field indicates the power states in which the function may assert PME#. A value of 0b for any bit indicates that the function is not capable of asserting the PME# signal while in that power state. bit(27) XXXX0b - PME# cannot be asserted from DO bit(28) XXX1Xb - PME# can be asserted from D1 bit(29) XX1XXb - PME# can be asserted from D2 bit(30) X1XXXb - PME# can be asserted from D3hot bit(31) 0XXXXb - PME# cannot be asserted from D3cold
26	D2_Support	This device supports the D2 Power Management State. 0 = Disable D2 Power Management State. **(default)** 1 = Enable D2 Power Management State.
25	D1_Support	This device supports the D1 Power Management State. 0 = Disable D1 Power Management State. **(default)** 1 = Enable D1 Power Management State.
24	Dyn_Data _Support	Dynamic Data is not supported.
23:22	Reserved	These bits are read only and return 0's when read. Writes have no effect.
21	DSI	No Device Specific Initialization is required. 0 = Disable DSI. 1 = Not Allowed.
20	APS	Auxiliary Power Source not supported.
19	PMECLK	No PCI dock is required for this function to generate PME#.
18:16	Version	A value of 001 b indicates this device complies with the Revision 1.0 of the ACPI 1.0 specification.
15:8	Next Item Pointer	No additional items in the Capabilities List.
7:0	ID	This field, when '01 h" identifies the linked list item as the PCI Power Management Capabilities Registers.

Power Management Control/Status Register (Offset: A4h)

BIT POSITION	NAME	DESCRIPTION
31:24	Data	This register is used to report the state dependent data requested by the Data_Select field. The value of this register is scaled by the value reported by the Data_Scale field. Read Only. Units are in Watts.
23:16	Bridge Support Extensions	Not support. Read Only
15	PME_Status	This bit is set to indicate PME# request regardless of PME_En bit. PME# signal independent of the state of the PME En bit. Writing a "1" to this bit will clear it. Writing a "0" has no effect.
14:13	Data_Scale	This two bit read-only field indicates the scaling factor to be used when interpreting the value of the Data register. The value & meaning of this field will vary depending on which data value has been selected by the Data_Select field. Read Only. **Data_Scale Interpretation** 0= Unknown 1= 0.1x 2= 0.01x 3= 0.001x
12:9	Data_Select	This four-bit field selects which data is to be reported through the Data register and Data_Scale field. **Value in Data_Select** **Data Reported** 0 DO Power consumed 1 D1 Power consumed 2 D2 Power consumed 3 D3 Power consumed 4 DO Power dissipated 5 D1 Power dissipated 6 D2 Power dissipated 7 D3 Power dissipated 8 - 15 Reserved
8	PMD_En	PME# assertion. 0 = Disable PME# assertion. **(default)** 1 = Enables PME# assertion.
7:5	Reserved	These bits are read only anti return 0's when read. Writes have no effect.
4	Ddata_PME# En	Dynamic Data PME# Enable is not support.
3:2	Reserved	These bits are read only and return 0's when read. Writes have no effect.
1:0	PowerState	This two-bit field sets the power state of the device. The definition of the field values is given below. 00b - DO 01b - D1 10b - D2 11 b - D3hot

General Purpose Event Status Register (Offset :A8h)

BIT POSITION	NAME	DESCRIPTION
15:5	Reserved	These bits are read only and return 0's when read. Writes have no effect.
4	GP4 STS	This bit is set to indicate a change in status of the MF5 input. This is R/WC bit. Writing a 1 clears this bit.
3	GP3 STS	This bit is set to indicate a change in status of the MF4 input. This is R/WC bit. Writing a 1 clears this bit.
2	GP2 STS	This bit is set to indicate a change in status of the MF2 input. This is R/WC bit. Writing a 1 clears this bit.
1	GP1 STS	This bit is set to indicate a change in status of the MF1 input. This is R/WC bit. Writing a 1 clears this bit.
0	GP0 STS	This bit is set to indicate a change in status of the MF0 input. This is R/WC bit. Writing a 1 clears this bit.

General Purpose Event Enable Register (Offset :AAh)

BIT POSITION	NAME	DESCRIPTION
15:5	Reserved	These bits are read only and return 0's when read. Writes have no effect.
4	GP4 ENB	If this bit is set, a GPE# is signaled when GP4 STS is set and MF5 is configured as GPI4.
3	GP3 ENB	If this bit is set, a GPE# is signaled when GP3 STS is set and MF4 is configured as GPI3.
2	GP2 ENB	If this bit is set, a GPE# is signaled when GP2 STS is set and MF2 is configured as GPI2.
1	GP1 ENB	If this bit is set, a GPE# is signaled when GP1 STS is set and MF1 is configured as GPI1.
0	GP0 ENB	If this bit is set, a GPE# is signaled when GP0 STS is set and MF0 is configured as GPI0.

General Purpose Input Register (Offset :ACh)

BIT POSITION	NAME	DESCRIPTION
15:5	Reserved	These bits are read only and return 0's when read. Writes have no effect.
4	GPI4	This bit reflects the logical value at the MF5 input. Writes have no effect.
3	GPI3	This bit reflects the logical value at the MF4 input. Writes have no effect.
2	GPI2	This bit reflects the logical value at the MF2 input. Writes have no effect.
1	GPI1	This bit reflects the logical value at the MF1 input. Writes have no effect.
0	GPI0	This bit reflects the logical value at the MF0 input. Writes have no effect.

General Purpose Output Register (Offset :AEh)

BIT POSITION	NAME	DESCRIPTION
15:5	Reserved	These bits are read only and return 0's when read. Writes have no effect.
4	GPI4	This bit sets MF5 of configured as GPO5.
3	GPI3	This bit sets MF4 of configured as GPO4.
2	GPI2	This bit sets MF2 of configured as GPO2.
1	GPI1	This bit sets MF1 of configured as GPO1.
0	GPI0	This bit sets MF0 of configured as GPO0.

O2Micro Control1 Register (Offset :D0h)

BIT POSITION	NAME	DESCRIPTION
31:24	Reserved	These bits are read only and return 0's when read. Writes have no effect.
23	PCI_VCC Select	0 = PCI _VCC is connected to +3.3V. **(default)** 1 = PCI_VCC is connected to 5V.
22	Aux_VCC Select	0 = Aux_VCC is connected to +5V. **(default)** 1 = Aux_VCC is connected to 3.3V.
21:20	Reserved	These bits are read only and return 0's when read. Writes have no effect.
19:18	Command Width	00 = PCMCIA command active width 165ns. **(default)** 01 = PCMCIA command active width 80ns. 10 = PCMCIA command active width 40ns. 11 = Reserved.
17	Reserved	These bits are read only arid return 0,'s when read. Writes have no effect.
16	Include_PCI_INT	0 = PCI interrupt is not included in serial IRQ. **(default)** 1 = PCI interrupt is included in the serial IRQ stream. See Bits 10-12.
15:13	Reserved	These bits are read only and return 0's when read. Writes have no effect.
12:10	Serial IRQ (PCI Interupt Select)	When Serial IRQ is enabled and PCI interrupt is indicated to use serial IRQ outputs, the PCI interrupts can be routed to PCI INTA#, INTB#, INTC# or INTD#. These bits determine the routing method: 000 = INTA# selected. 001 = INTB# selected. 010 = INTC# selected. 011 = INTD# selected. 100 = Reserved. 101 = Reserved. 110 = Reserved 111 = Reseved.
9	ZV_ACTV_Enable	Zoom Video Activity Enable. 0 = Disable Zoom Video activity. **(default)** 1 = Enable Zoom Video activity.
8	SKT_ACTV_Enable	Socket Activity Enable. 0 = Disable socket activity LED Output. **(default)** 1 = Enable socket activity LED Output.
7:0	Reserved	These bits are read only and return 0's when read. Writes have no effect.

O2Micro Control2 Register (Offset :D4h)

BIT POSITION	NAME	DESCRIPTION
31:14	Reserved	These bits are read only and return 0's when read. Writes have no effect.
13	SCLK_ENABLE	0 = The SCLK EN is input to the chip. **(default)** 1 = The SCLK EN is outptjt from the chip.
12:0	Reserved	Used for diagnostic test mode.

CardBus Socket Status & Control Register

These registers are be mapped into memory space to allow faster access than would be possible if they were in configuration space. All of these registers shall be initialized by PCIRST#.

Status Event Register (Offset: 00h)

The Status Event Register indicates a change in socket status has occurred. These bits do not indicate what the change is, only that one has occurred. Software must read the Socket Present State Register for current status. All of the bits in this register can be cleared by writing I to that bit. These bits can be set by software through writing I to the corresponding bit in the Force Register. All bits in this register are cleared by PCIRST#.

These bits will be set after CRST# if CSTSCHG is asserted or Card Detects is active.

BIT POSITION	NAME	DESCRIPTION
31:4	Reserved	These bits are read only and return 0's when read. Writes have no effect
3	PWR_EVENT	Power cycle complete event. Indicates socket power up completed. The Present State register should be read to determine that the voltage requested was actually applied. 0 = No power cycle complete event indicated **(default)**. 1 = Power cycle complete event indicated.
2	CDEVENT2	Card Detect Event 2. indicates a change has occurred in the corresponding Card Detect on the CD2#/CCD2# signal. 0 = CD2#/CCD2# change not detected **(default)**. 1 = CD2#/CCD2# change detected.
0	CSTSEVENT	Card Status Change VVakeup bit. Indicates that the CSTSCHG WAKEUP signal has been asserted. It only indicates the assertion event. It is not a reflection of the CSTSCHG bit from the card. It will be latched by the controller and must be explicitly cleared by the appropriate software. The status change interrupt is based on this bit. When the socket is powered clown, this bit is a WAKEUP. 0 = Disabled **(default)**. 1 = Enabled.

Status Mask Register (Offset: 04h)

This register gives software the ability to control what status change interrupts are generated. This register is cleared by a PCIRST#.

BIT POSITION	NAME	DESCRIPTION
31:4	Reserved	These bits are read only and return 0's when read. Writes have no effect.
3	PWR_ENA	Power Event Interrupt Enabled. When set, enables interrupt on PWR_EVENT bit. 0 = Disabled **(default)**. 1 = Enables interrupt.
2	CD2ENA	Card Detect Event 2 Interrupt Enable. Enables interrupt on CDEVENT2. 0 = Disabled **(default)**. 1 = Enables interrupt.
1	CD1ENA	Card Detect Event 1 Interrupt Enable. Enables interrupt on CDEVENT1. 0 = Disabled **(default)**. 1 = Enables interrupt.
0	CSTSENA	Card status change interrupt enable. When set, enables an interrupt based on bit 1, 00h. 0 = Disabled **(default)**. 1 = Enable Card status change interrupt.

Present State Register (Offset: 08h)

The Socket Present State Register reflects the present value of the sockets status. Some of the bits in this register are merely reflections of interface signals while others are flags to indicate current status. This register is initialized after PCI Reset.

BIT POSITION	NAME	DESCRIPTION
31:28	Vcc Voltage Available	Indicates the Vcc voltages available for the socket in this machine. Bit31 YV (hardwired to 0) Bit30 XV (hardwired to 0) Bit29 3V (hardwired to 0011) Bit28 5V (hardwired to 0011)
27:14	Reserved	These bits are read only and return 0's when read. Writes have no effect.
13	YVCard	Future voltage support. Reserved.
12	XVCard	Future voltage support. Reserved.
11	3VCard	Indicates installed card installed supports 3.3V operation. 0 = Disabled **(default)**. 1 = Installed card supports 3.3V operation.
10	5VCard	Indicates installed card supports 5.0V operation. 0 = Disabled **(default)**. 1 = Installed card supports 5.0V operation.
9	BadVccReq	Indicates software attempted to apply an unsupported or incorrect voltage to a socket. 0 = Disabled {default). 1 = incorrect voltage application attempted.
8	DataLost	Indicates that a card was removed while the interface was active. Data may have been lost. 0 = Disabled **(default)**. 1 = Card removed while interface was active.
7	NotACard	Indicates an undetermined card is installed in the socket. 0 = Disabled **(default)**. 1 = Installed card undetermined.
6	Interrupt	Interrupt bit. This bit reflects the state of CINT#. 0 = CINT# NOT asserted **(default)**. 1 = CINT# asserted.
5	CBCard	CardBus Card. indicates a CardBus is inserted. 0 = Card inserted is NOT a CardBus. **(default)**. 1 = Card inserted is a CardBus.

BIT POSITION	NAME	DESCRIPTION
4	PCCARD	PCCard. Indicates a PCCard is inserted. 0 = Card inserted is NOT a PCCard. **(default)**. 1 = Card inserted is a PCCard.
3	PWR_STA	Power Status. Indicates the socket is powered up when set. When cleared, the socket is powered down. 0 = Socket is powered down **(default)**. 1 = Socket is powered up.
2	CD2STA	Card Detect Status 2. Indicates PCCard presence. It is a reflection of CD2#/CCD2#. 0 = PCCard not present **(default)**. 1 = PCCard present.
1	CD1STA	Card Detect Status 1. Indicates PCCard presence. It is a reflection of CDI#/CCDI#. 0 = PCCard not resent **(default)**. 1 = PCCard present.
0	CSTSSTA	Card Status Change Status. This bit indicates the current status of the CSTSSTA pin on the CardBus interface. 0 = Disabled **(default)**. 1 = Enabled.

Force Register (Offset 0Ch) Write Only

The Force Register is a write only register. It provides software the ability to set various status and event bits. Writing a one to a bit in this register sets the corresponding bit in tlne Status Event Register and/or the Present State Register.

BIT POSITION	NAME	DESCRIPTION
31:15	Reserved	These bits are read only and return 0's when read. Writes have no effect.
14	CV TestFRC	Card Voltage Test Force. Causes the OZ6812 to test and determine card type and voltages supported. This test runs automatically when a new card is inserted. 0 = Disabled **(default)**. 1 = Enabled.
13:12	Reserved	These bits are read only and return 0's when read. Writes have no effect.
11	3VcardFRC	This bit sets the 3VCard bit in the Present State Register. 0 = Disabled **(default)**. 1 = Enabled.
10	5VCardFRC	This bit sets the 5VCard bit in the Present State Register, 0 = Disabled **(default)**. 1 = Enabled.
9	BadVccReq FRC	This bit sets the BadVccReq bit in the Present State Register, 0 = Disabled **(default)**. 1 = Enabled.
8	Data Lost FRC	This bit sets the Data Lost bit to be set in the Present State Register. 0 = Disabled **(default)**. 1 = Enabled.
7	NotACard FRC	Sets the NotACard bit in the Present State Register. If a card is installed in the socket, writes to this bit are ignored. 0 = Disabled **(default)**. 1 = Enabled.
6	Reserved	This bit is read only and return 0's when read. Writes have no effect,
5	CBCardFRC	CardBus Card Force. Sets the CBCard bit in the Present State Register. If a card is installed in the socket, writes to this bit are ignored. 0 = Disabled **(default)**. 1 = Enabled.
4	PCCardFRC	PCCard Force. Sets the PCCard bit in the Present State Register. If a card is installed in the socket, writes to this bit are ignored. 0 = Disabled **(default)**. 1 = Enabled.

BIT POSITION	NAME	DESCRIPTION
3	PWR FRC	PowerCycle Complete Force Sets the PowerCycle Complete in the Event Register, The Present State Register remains unchanged. 0 = Disabled **(default)**. 1 = Enabled.
2	CD2FRC	Card Detect 2 Force. Sets the Card Detect 2 bit in Event Register. The Present State Register remains unchanged. 0 = Disabled **(default)**. 1 = Enabled.
1	CD1FRC	Card Detect 1 Force. Sets the Card Detect 1 bit in Event Register. The Present State Register remains unchanged. 0 = Disabled **(default)**. 1 = Enabled.
0	CSTSFRC	Card Status Force. Sets the Card Status hit in the Event Register. The Present State Register remains unchanged. 0 = Disabled **(default)**. 1= Enabled.

Control Register(Offset: 10h)

The Socket Control Register controls the socket's Vcc and Vpp. This register is cleared by PCIRST#.

BIT POSITION	NAME	DESCRIPTION
31:8	Reserved	These bits are read only and return 0's when read. Writes have no effect.
7	StopClock	Rewrite causes the bridge to stop the CardBus clock (CCLK) using the CCLKRUN# protocol. This allows software control of the CCLKRUN# protocol in those systems that do not support CCLKRUN# on the host side of the controller. 0 = Disable CCLKRUN# protocol **(default)**. 1 = Enable
6:4	Vcc Control	Used to apply Vcc to the PCCard via external control logic. The bridge determines the voltages that can be applied by decoding the CD and VS signals per the CardBus specification and software must determine the needed voltage from the card's CIS. **Bit6 Bit5 Bit4 Vcc Requested** 0 0 0 0V 0 0 1 Reserved 0 1 0 5.0V 0 1 1 3.3V 1 0 0 Reserved 1 0 1 Reserved 1 1 0 Reserved 1 1 1 Reserved
3	Reserved	This bit is read only and returns 0's when read. Writes have no effect.
2:0	Vpp Control	Used to apply Vpp to the PCCard via external control logic. The bridge determines the voltages that can be applied by decoding the CD and VS signals per the CardBus specification and software must determine the needed voltage from the card's CIS. **Bit2 Bit1 Bit0 VPP Requested** 0 0 0 0V 0 0 1 12.0V 0 1 0 5.0V 0 1 1 3.3V 1 0 0 Reserved 1 0 1 Reserved 1 1 0 Reserved 1 1 1 Reserved

Socket Zoomed Video Control Register (Offset: 20h)

The Zoomed Video Control Register provide control of each socket zoomed video mode enable . All bits in this register should be set to zero by PCIRST#.

BIT POSITION	NAME	DESCRIPTION
31:1	Reserved	These bits are read only and return 0's when read. Writes have no effect.
0	Zoomed Video	Zoomed Video enable. This bit reflects ExCA register 3Ah, bit 3. 0 = Zoomed Video disabled **(default)**. 1 = Zoomed Video enabled.

Socket Interrupt and MHPG DMA Control Register (Offset: 24h)

The Socket Interrupt and MHPG DMA Control Register provide control of each socket status change and CINT# interrupt enable.

BIT POSITION	NAME	DESCRIPTION
31:2	Reserved	These bits are read only and return 0's when read. Writes have no effect.
2:0	MHPG DMA Channel#	Define the DMA socket's channel number from 0h-7h

O$_2$MICRO MISC CONTROL Register (Offset: 28h)

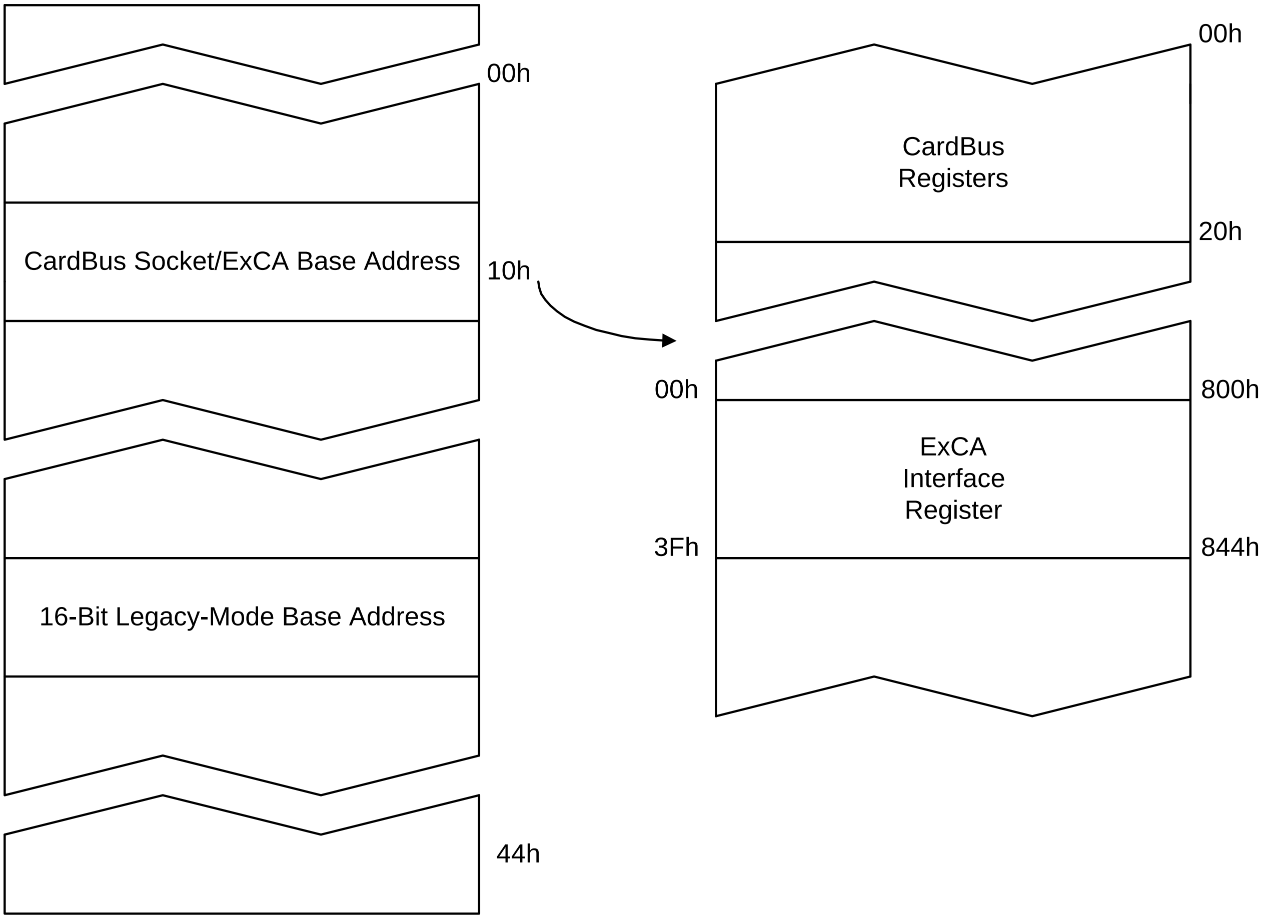
This register bit 1:0 will directly reflect at bit 6:5 of Index Register 38.
This register bit 9:8 will directly reflect at bit 1:0 of index Register 3B.
This register bit 11 will directly reflect at bit 7 of Index Register 3B.
This register bit 16 will directly reflect at bit 4 of Index Register 3D.
This register bit 23:22 will directly reflect at bit 7:6 of Index Register 3D.
This register bit 25:24 will directly reflect at bit 1:0 of index Register 3E.
This register bit 28:27 will directly reflect at bit 4:3 of Index Register 3E.

BIT POSITION	NAME	DESCRIPTION
31:29	Reserved	These bits are read only and return 0's when read. Writes have no effect.
28	SKT_ACTV	This bit selects LED_OUT/SKT_ACTV pin between its two modes 0 = LED_OUT/SKT_ACTV pin is LED_OUT **(default)** 1 = LED_OUT/SKT_ACTV pin is SKT_ACTV
27	LED_OUT Enable	This bit enables the output for pin LED_OUT. 0 = Disables the output for pin LED_OUT**(default)** 1 = Enables the output for pin LED OUT
26	Reserved	This bit is read only and return 0's when read. Writes have no effect.
25	SPKR_OUT Enable	This bit enables the output for pin SPKR OUT. 0 = Disables the output for pin SPKR_OUT**(default)** 1 = Enables the output for pin SPKR_OUT
24	MHPG DMA MODE	This bit enables MHPG DMA Mode. 0 = MHPG DMA Mode not enabled **(default)** 1 = MHPG DMA Mode enabled
23	CB_FIFO	This bit enables CardBus Data FIFO full buffer. 0 = CardBus Data FIFO full buffer disabled 1 = CardBus Data FIFO full buffer enabled **(default)**
22	MEM_POST WR	This bit enables CardBus bridge memory post write function. 0 = Cardbus bridge memory post write function disabled 1 = Cardbus bridge memory post write function enabled **(default)**
21:17	Reserved	These bits are read only and return 0's when read. Writes have no effect.
16	PCI_FIFO	This bit enables PCI Data FIFO. 0 = PCI Data FIFO disabled 1 = PCI Data FIFO enabled **(default)**
15:12	Reserved	These bits are read only and return 0's when read. Writes have no effect.
11	Parallel ISA	This bit enables ISA Interrupt mode. 0 = PCI-compatible interrupt mode **(default)** 1 = ISA-legacy interrupt mode
10	Reserved	This bit is read only and return 0's when read. Writes have no effect.
9:8	System Interrupt Mode	These bits indicate the type of interrupt mode selected. 00 = PC/PCI interrupt Mode 01 = Reserved 10 = PCI/Way interrupt Mode 11 = PCI Interrupt Mode **(default)**
7:2	Reserved	These bits are read only and return 0's when read. Writes have no effect.
1:0	Power Chip	These bits indicate the socket power control device being used. 00 = Reserved 01 = Parallel socket power **(default)** 10 = Serial socket power 11 = SMBus socket power

ExCA REGISTERS

The ExCA registers implemented in the OZ6812 are registered compatible with Intel 82365SL-DF PCMCIA Controller. The OZ6812 provides two ways for accessing the ExCA-compatible registers:

1) Directly mapped into PCI memory space at offset 800h (PCI Conhfiuration Register 10h).
2) Using Index/Data I/O method at the address program in the 16 bit PCCard Legacy Mode Base Address Register (PCI Configuration Register 44h)



Index Register Table

Socket Memory Offset	Socket I/O Offset	Register Name
800h	00h	Identification and Revision
801h	01h	Interface Status
802h	02h	Power Control
803h	03h	Interrupt and General Control
804h	04h	Card Status Change
805h	05h	Card Status Change Interrupt Configuration
806h	06h	Address Window Enable
807h	07h	I/O Window Control
808h	08h	I/O Window 0 Address Start LOW Byte
809h	09h	I/O Window 0 Address Start HIGH Byte
80Ah	0Ah	I/O Window 0 Address Stop LOW Byte
80Bh	0Bh	I/O Window 0 Address Stop HIGH Byte
80Ch	0Ch	I/O Window 1 Address Start LOW Byte
80Dh	0Dh	I/O Window 1 Address Start HIGH Byte
80Eh	0Eh	I/O Window 1 Address Stop LOW Byte
80Fh	0Fh	I/O Window 1 Address Stop HIGH Byte
810h	10h	Memory Window 0 Address Mapping Start LOW Byte
811h	11h	Memory Window 0 Address Mapping Start HIGH Byte
812h	12h	Memory Window 0 Address Mapping Stop LOW Byte
813h	13h	Memory Window 0 Address Mapping Stop HIGH Byte
814h	14h	Memory Window 0 Offset LOW Byte
815h	15h	Memory Window 0 Offset HIGH Byte
816h	16h	Card Detect and General Control Register
840h	17h	Memory Window 0 Address Mapping Start UPPER Byte
818h	18h	Memory Window 1 Address Mapping Start LOW Byte
819h	19h	Memory Window 1 Address Mapping Start HIGH Byte
81Ah	1Ah	Memory Window 1 Address Mapping Stop LOW Byte
81Bh	1Bh	Memory Window 1 Address Mapping Stop HIGH Byte
81Ch	1Ch	Memory Window 1 Offset LOW Byte
81Dh	1Dh	Memory Window 1 Offset HIGH Byte
81Eh	1Eh	Global Control Register
841h	1Fh	Memory Window 1 Address Mapping Start UPPER Byte
820h	20h	Memory Window 2 Address Mapping Start LOW Byte
821h	21h	Memory Window 2 Address Mapping Start HIGH Byte
822h	22h	Memory Window 2 Address Mapping Stop LOW Byte
823h	23h	Memory Window 2 Address Mapping Stop HIGH Byte
824h	24h	Memory Window 2 Offset LOW Byte
825h	25h	Memory Window 2 Offset HIGH Byte
826h	26h	O_2 Micro Mode Control A
842h	27h	Memory Window 2 Address Mapping Start UPPER Byte
828h	28h	Memory Window 3 Address Mapping Start LOW Byte
829h	29h	Memory Window 3 Address Mapping Start HIGH Byte
82Ah	2Ah	Memory Window 3 Address Mapping Stop LOW Byte
82Bh	2Bh	Memory Window 3 Address Mapping Stop HIGH Byte
82Ch	2Ch	Memory Window 3 Offset LOW Byte
82Dh	2Dh	Memory Window 3 Offset HIGH Byte
82Eh	2Eh	O_2 Micro Mode Control B
843h	2Fh	Memory Window 3 Address Mapping Start UPPER Byte
830h	30h	Memory Window 4 Address Mapping Start LOW Byte
831h	31h	Memory Window 4 Address Mapping Start HIGH Byte
832h	32h	Memory Window 4 Address Mapping Stop LOW Byte
833h	33h	Memory Window 4 Address Mapping Stop HIGH Byte
834h	34h	Memory Window 4 Offset LOW Byte
835h	35h	Memory Window 4 Offset HIGH Byte
836h	36h	I/O Window 0 Address Offset LOW Byte
837h	37h	I/O Window 0 Address Offset HIGH Byte
838h	38h	I/O Window 1 Address Offset LOW Byte
839h	39h	I/O Window 1 Address Offset HIGH Byte
83Ah	3Ah	O_2 Micro Mode Control C
83Bh	3Bh	O_2 Micro Mode Control D
83Ch	3Ch	Centralized DMA Register
83Dh	3Dh	FIFO Enable Register
83Eh	3Eh	O_2Micro Mode Control E
844h	3Fh	Memory Window 4 Mapping Start UPPER Byte

Identification and Revision Register (Read Only)
Index + 00h (CardBus 800h)

The Identification and Revision Register is used by the system software to determine the type of PCCards supported, and to identify what version of the OZ6812 is present. This register default can be altered by O2Micro Control B Register.

BIT POSITION	NAME	DESCRIPTION
7:6	OZ6812 ID	These bits indicate the type of PC Cards supported by the OZ6812. These bits are hard wired a [1:0], indicating that Memory and I/O are supported.
5:4	Reserved	These bits will be read back as "0"s
3:0	OZ6812 Stepping	These four bits indicate the current revision level of the OZ6812. The stepping code will be 0100.

Interface Status Register (Read Only)
Index + 01h (CardBus 801h)

The Interface Status Register provides the current status of the PCCard interface.

BIT POSITION	NAME	DESCRIPTION
7	Reserved	Read as "0".
6	PCCard Power Active	Indicates the current power status of the socket. 0 = Power not applied to PCCard. **(default)** 1 = Power is applied to PCCard.
5	Ready/Busy#	Indicates the ready condition of the PC Memory Card. 0 = CardBus is busy. **(default)** 1 = CardBus is ready.
4	Memory Write Protect	Indicates the WP state of the memory PCCard. However, memory write will not be blocked unless the write protect bit in the associated Card Memory Offset Address Register HIGH byte register is set to "1". 0 = Write enabled. **(default)** 1 = Write protected.
3:2	Card Detect 2 and 1	Together they indicate whether card is present at the socket and fully seated. Bits are set if the CDI#, CD2# signal on the PCCard interface are active. Bits are set to "0" if the CDI#, CD2# signals on the PCCard interface are inactive. 00 = Card not present /not fully seated. **(default)** 11 = Card present and fully seated.
1:0	Battery Voltage Detect 2 and 1	For I/O PC Cards, bit 0 indicates the current state of the status change signal, and bit 1 indicates the current state of the SPKR#. Refer to interrupt General Control Register (index + 03h) bit 7&5. **BVD1 BVD2 Memory Card Battery Status** 0 0 Battery Dead 0 1 Battery Dead 1 0 Warning 1 1 Battery Good

Power Control Register (Read/Write) Index + 02h (CardBus 802h)

BIT POSITION	NAME	DESCRIPTION
7	Output Enable	This bit controls the PC card interface outputs. 0 = Output tristated. **(default)** 1 = Output enabled.
6:5	Reserved	This bit will be read back as "0"s,
4	PCCard Power Enable	This bit controls PCCard power. 0 = PCCard Power disabled. **(default)** 1 = PCCard Power enabled.
3	Vcc Select	This bit selects Vcc power to the socket. 0 = 5.0V is applied to Vcc. **(default)** 1 = 3.3V is applied to Vcc.
2	Reserved	This bit will be read back as "0"s.
1:0	Vpp Select	These bits select the voltage applied to Vpp. **[1] [0]** 0 0 0V **(default)** 0 1 Vcc selected 1 0 +12V 1 1 0V

Interrupt and General Control Register (Read/Write) Index + 03h (CardBus 803h)

This register controls the interrupt steering for the PC I/O card and PCCard Type.

BIT POSITION	NAME	DESCRIPTION
7	Ring Indicate Enable	This bit controls status change/Ring Indicate signal function form the I/O PCCard. If bit is set to "1" and the PC card Type bit (Bit 5), is set to "1" (I/O PC Card), the (VDI/STSCHG#) signal from the I/O PC Card is used as a ring indicator signal. 0 = Status Change enabled. **(default)** 1 = Ring Indicate enabled.
6	PCCard Reset	This bit controls PCCard reset. 0 = Reset is asserted. **(default)** 1 = Reset is not asserted.
5	PCCard Type (Memory or I/O)	This bit configures the interface for I/O memory type. 0 = Memory type enabled. **(default)** 1 = I/O type enabled.
4	Teserved	Read as "0"
3:0	Functional IRQ Select (I/O Cards only)	PCCard IREQ# Interrupt Steering Table. These bits select the IRQ for PCCard interrupt. **Note:** CardBus Control Register Bit7 must be set to enable IRQ selection (3Eh) see P.32 **Bit 3 Bit 2 Bit 1 Bit 0** 0 0 0 0 No IRQ selected **(default)** 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 Register (Read/Write)
Index + 04h (CardBus 804h)

This register indicates that a changed of status has occurred. It contains the sources for the card status change interrupts. These sources can be enabled by setting the corresponding bit in the Card Status Change Interrupt Configuration Register (Index + 05h). The bits in this register will be read back as 0 if disabled in the Card Status Change Interrupt Configuration Register.

These bits must be cleared by writing 1 to the appropriate bit. If the Explicit write Back Card Status Change Acknowledge bit is not set in the Global Control Register (Index + 1Eh), this register will be cleared automatically each time it is read. The interrupt signal caused by card status change will be active until all of the bits in this register are cleared.

If the Explicit write Back Card Status Change Acknowledge bit is not set, when enabled on a system IRQ line, it will remain active until the Card Status Change register is read. The read operation to the Card Status Change Register will reset all the bits in that register.

In the case that there are any change interrupts pending, and another status change interrupt condition occurs the OZ6812 will not generate a second interrupt pulse. This means the interrupt handles the routine.

The Interrupt Service Routine must ensure interrupt requests are serviced before emitting the service routines.

BIT POSITION	NAME	DESCRIPTION
7:4	Reserved	These reserved bits always read "0".
3	Card Detect Change	This bit indicates a card change has been detected. 0 = No change detected. **(default)** 1 = Change has been detected.
2	Ready Change	This bit indicates that the memory PCCard is ready to accept a new data transfer. 0 = Memory PCCard busy. **(default)** 1 = A LOW to HIGH transition has been detected on the READY/BUSY# signal indicating that the memory PCCard is ready to accept a new data transfer.
1	Battery Warning	This bit indicates that a battery warning condition has been detected. 0 = No battery warning condition detected. **(default)** 1 = Battery warning condition detected. **Note:** Always read 0 for I/O PC Cards.
0	Battery Dead (BVD1/STSCHG#)	This bit indicates when a battery dead condition has been detected. For Memory PCCards: 0 = No Battery dead condition detected. **(default)** 1 = Battery dead condition detected. For I/O PCCards, this bit is set to 1 when a HIGH to LOW transition has been detected on the BVD/STSCHG# signal. 0 = No STSCHG# change detected. **(default)** 1 = HIGH to LOW transition detected on STSCHG# **Note:** For I/O PCCards, this bit will always be 0 if Ring Indicate Enabled is set in the Interrupt and General Control Register (Index +03).

Card Status Change Interrupt Configuration Register (Read/Write)
Index + 05h (CardBus 805h)

This register selects card status change interrupt and the card status change interrupt enables.

BIT POSITION	NAME	DESCRIPTION
7:4	Status Change IRQ Select	Status Change IRQ select for the Card Status Change Interrupt **Bit 7** **Bit 6** **Bit 5** **Bit 4** 0 0 0 0 No IRQ Selected {default} 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	Card Detect Change Enable	This bit enables a Card Status Change interrupt. 0 = Card Status Change Interrupt generation disabled. **(default)** 1 = Card Status Change Interrupt enabled.
2	Ready Change Enable	This bit enables a Ready Change Interrupt. 0 = Ready Change interrupt generation disabled. **(default)** 1 = Ready Change Interrupt generation enabled.
1	Battery Warning Enable	This bit enables battery warning condition interrupts. 0 = Battery warning interrupt generation disabled. **(default)** 1 = Setting bit to "0" disables the generation of a card status change interrupt when a battery warning condition has been detected.
0	Battery Dead Enable (BVD1/STSCHG#)	This bit enables Battery Dead or Status Change interrupt. For many PCCards, this bit enables Battery Dead Condition Interrupts. 0 = Disables battery dead condition interrupts. **(default)** 1 = Enables battery dead condition interrupts. For I/O PCCards, this bit enables Status Change Interrupts. 0 = Disables status change interrupts. **(default)** 1 = Enables status change interrupts. Note: For I/O PCCards, this bit will always be 0 if Ring Indicate Enabled is set in the Interrupt and General C.~ntrol Register (Index +03).

Address Window Enable Register (Read/Write)
Index + 06h (CardBus 806h)

This register enables the memory and I/O mapping windows of the PCCard. The start/stop offset must be set before enabling any window.

BIT POSITION	NAME	DESCRIPTION
7	I/O Window 1 Enable	This bit enables I/O address 1 window, 0 = Disables I/O window 1. **(default)** 1 = Enables I/O window 1.
6	I/O Window 0 Enable	This bit enables I/O address 0 window. 0 = Disables I/O window 0. **(default)** 1 = Enables I/O window 0.
5	Decode A23-A12	This bit enables 4K memory window decoding. 0 = 128K memory window decoding enabled. **(default)** 1 = 4K memory window decoding enabled.
4	Memory Window 4 Enable	This bit enables memory address 4 window. 0 = Disables memory window 4. **(default)** 1 = Enables memory window 4.
3	Memory Window 3 Enable	This bit enables memory address 3 window. 0 = Disables memory window 3. **(default)** 1 = Enables memory window 3.
2	Memory Window 2 Enable	This bit enables memory address 2 window. 0 = Disables memory window 2. **(default)** 1 = Enables memory window 2.
1	Memory Window 1 Enable	This bit enables memory address 1 window. 0 = Disables memory window 1. **(default)** 1 = Enables memory window 1.
0	Memory Window 0 Enable	This bit enables memory address 0 window. 0 = Disables memory window 0. **(default)** 1 = Enables memory window 0.

I/O Window Control Register (Read/Write)
Index + 07h (CardBus 807h)

BIT POSITION	NAME	DESCRIPTION
7	I/O Window 1 Wait State	This bit configures the I/O window 1 wait state. 0 = 16 bit cycle standard length (3 ISA cycles) enabled. **(default)** 1 = 16 bit cycle extended by 1 ISA cycle (4 ISA cycles).
6	I/O Window 1 ZeroWait State	This bit configures the I/O window 1 ZeroWait state. 0 = 8 bit cycle standard length (6 ISA cycles). **(default)** 1 = 8 bit cycle extended length (3 ISA cycles).
5	I/O Window 1 IOCS16# Source	This bit configures I/O window 1 IOSC16# generation. 0 = IOCS16# generated based on the value of I/O Window 1 (bit4). **(default)** 1 = IOCS16# generated based on the IOIS16# signal from the PCCard and I/O Window 1 (bit4) is ignored.
4	I/O Window 1 Data Size	This bit selects the I/O window 1 data size. 0 = 8 bit I/O data path selected to the PCCard. **(default)** 1 = 16 bit I/O data path selected to the PCCard.
3	I/O Window 0 Wait State	This bit configures the I/O window 0 wait state. 0 = 16 bit cycle standard length (3 ISA cycles) enabled. **(default)** 1 = 16 bit cycle extended by 1 ISA cycle (4 ISA cycles).
2	I/O Window 0 ZeroWait State	This bit configures the I/O window 0 ZeroWait state. 0 = 8 bit cycle standard length (6 ISA cycles). **(default)** 1 = 8 bit cycle standard length (3 ISA cycles).
1	I/O Window 0 IOCS16# Source	This bit configures I/O window 0 IOSC16# generation. 0 = IOCS163 generated based on the value of I/O Window 0 (bit4). **(default)** 1 = IOCS16# generated based on tile IOIS16# signal from the PCCard and I/O Window 0 (bit4) is ignored.
0	I.O Window 0 Data Size	This bit selects the I/O window 0 data size. 0 = 8 bit I/O data path selected to the PCCard. **(default)** 1 = 16 bit I/O data path selected to the PCCard.

I/O Window 0 Address Start Low Byte Register (Read/Write)
Index + 08h (CardBus 808h)

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

BIT POSITION	NAME	DESCRIPTION
7:0	Address 7:0	I/O Window 0 Address Start A7:A0

I/O Window 0 Address Start High Byte Register (Read/Write)
Index + 09h (CardBus 809h)

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

BIT POSITION	NAME	DESCRIPTION
7:0	Address 15:8	I/O Window 0 Address Start A15:A8

I/O Window 0 Address Stop Low Byte Register (Read/Write)
Index + 0Ah (CardBus 80Ah)

This register contains the high order address bits used to determine the stop address of I/O window 0. This provides a minimum I byte window for I/O window 0 address.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 7:0	I/O Window 0 Address Stop A7:A0

I/O Window 0 Address Stop High Byte Register (Read/Write)
Index + 0Bh (CardBus 80Bh)

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

BIT POSITION	NAME	DESCRIPTION
7:0	Address 15:8	I/O Window 0 Address Start A15:A8

I/O Window 1 Address Start Low Byte Register (Read/Write)
Index + 0Ch (CardBus 80Ch)

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

BIT POSITION	NAME	DESCRIPTION
7:0	Address 7:0	I/O Window 1 Address Start A7:A0

I/O Window 1 Address Start High Byte Register (Read/Write)
Index + 0Dh (CardBus 80Dh)

This register contains the high order address bits used to determine the start address of I/O window 1.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 15:8	I/O Window 1 Address Start A15:A8

I/O Window 1 Address Stop Low Byte Register (Read/Write) Index + 0Eh (CardBus 80Eh)

This register contains the high order address bits used to determine the stop address of I/O window 1. This provides a minimum 1 byte window for I/O window 1 address.

BIT POSITION	NAME	DESCRIPTION
7:0	Address7:0	I/O Window 1 Address Stop A7:A0

I/O Window 1 Address Stop High Byte Register (Read/Write) Index + 0Fh (CardBus 80Fh)

This register contains the high order address bits used to determine the stop address of I/O window 1.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 15:8	I/O Window 1 Address Start A15:A8

I/O Window 0 Address Offset Low Byte Register (Read/Write) Index + 36h (CardBus 836h)

This register contains the low order address bits used to determine the offset address of I/O window 0.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 7:0	I/O Window 0 Address offset A7:A0

I/O Window 0 Address Offset High Byte Register (Read/Write) Index + 37h (CardBus 837h)

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

BIT POSITION	NAME	DESCRIPTION
7:0	Address 15:8	I/O Window 0 Address offset A15:A8

I/O Window 1 Address Offset Low Byte Register (Read/Write) Index + 38h (CardBus 838h)

This register contains the low order address bits used to determine the offset address of I/O window 1.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 7:0	I/O Window 1 Address offset A7:A0

I/O Window 1 Address Offset High Byte Register (Read/Write) Index + 39h (CardBus 839h)

This register contains the low order address bits used to determine the offset address of I/O window 1.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 15:8	I/O Window I Address offset A15:A8

Memory Window 0 Address Start Low Byte Register (Read/Write) Index + 10h (CardBus 810h)

This register contains the low order address bits used to determine the start address of the corresponding ExCA memory address mapping window. This provides a minimum memory mapping window of 4 Kbytes.

NOTE: A memory window can not be set up below the first 64K of address space.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 19:12	ExCA Memory Window Start Address A19:A12.

Memory Window 0 Address Start High Byte Register (Read/Write) Index + 11h (CardBus 811h)

This register contains the high order address bits used to determine the start address of the corresponding ExCA memory window 0. Each ExCA memory window 0 has an associated data path size controlled by bit 7. Accesses to each ExCA memory window 0 have the potential to occur with zero additional states, controlled bit 6.

BIT POSITION	NAME	DESCRIPTION
7	Data Size	This bit selects the memory data path size for the PCCard. 0 = 8 bit memory data path selected. **(default)** 1 = 16 bit memory data path selected.
6	ZeroWait State	This bit selects the zero additional states in ExCA memory window 0 access. 0 = 8 bit or 16 bit ExCA memory access cycles completed in 6 ISA cycles or 3 ISA cycles respectively. **(default)** 1 = 8 bit and 16 bit ExCA memory access cycles completed in 3 ISA cycles and 2 ISA cycles respectively.
5:4	Scratch Bits	These bits can be used for general purpose register storage and retrieval.
3:0	Address 23:20	ExCA Memory Window 0 Start Address A23:A20.

Memory Window 0 Address Stop Low Byte Register (Read/Write)
Index + 12h (CardBus 812h)

This register contains the low order address bits used to determine the stop address of the corresponding ExCA memory address mapping window. This provides a minimum memory mapping window of 4 Kbytes.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 19:12	Memory Window 0 Stop Address A19:A12.

Memory Window 0 Address Stop High Byte Register (Read/Write)
Index + 13h (CardBus 813h)

This register contains the high order address bits used to determine the stop address of the corresponding ExCA memory address mapping window.

<table>
<tr><th>BIT POSITION</th><th>NAME</th><th>DESCRIPTION</th></tr>
<tr><td>7:6</td><td>Wait State
Select Bits 1:0</td><td>These bits determine the number of additional wait states for a 16 bit access to the ExCA memory window.
<table>
<tr><th>Wait State
Bit 1</th><th>Wait State
Bit 0</th><th>Number of Additional
Wait States</th><th>Number of ISA
Cycles per Access</th></tr>
<tr><td>0</td><td>0</td><td>Standard 16 bit Cycle</td><td>3</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>4</td></tr>
<tr><td>1</td><td>0</td><td>2</td><td>5</td></tr>
<tr><td>1</td><td>1</td><td>3</td><td>6</td></tr>
</table>
<b>Note:</b> For 8 bit access to the ExCA memory window 0, the wait state is controlled by the zero wait state bit (bit 6) of the Memory Window 0 Address Start High Byte Register (Index +1 lh). See P. 55</td></tr>
<tr><td>5:4</td><td>Reserved</td><td>These reserved bits always read "0".</td></tr>
<tr><td>3:0</td><td>Address 23:20</td><td>ExCA Memory Window 0 Start Address A23:A20.</td></tr>
</table>

Memory Window 0 Offset Low Byte Register (Read/Write)
Index + 14h (CardBus 814h)

This register contains the low order address bits added to the ExCA address bits A19:A12 to generate the memory address for the PCCard.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 19:12	Memory Window 0 Offset Address A19:A12

Memory Window 0 Offset High Byte (Read/Write)
Index + 15h (CardBus 815h)

This register contains the high order address bits which are added to the ExCA address bits A23:A20 to generate the memory address for the PCCard. The software write protect of the PCCard memory for the corresponding ExCA memory window is controlled by this register. This register also controls whether the corresponding system memory window is mapped to Attribute or Common Memory on the PCCard.

BIT POSITION	NAME	DESCRIPTION
7	Write Protect	This bit controls the write protect function of the PCCard. WP Switch on the memory card alone will not block the memory write cycle, but only set the Memory Write Protect bit (bit 4) in the Interface Status register (index + 01h). 0 = Write operations to the PCCard allowed for ExCA memory window. **(default)** 1 = Write operations to the PCCard inhibited for ExCA memory window.
6	Reg Active	This bit controls whether Common Memory or Attribute Memory is accessed on the PCCard. 0 = Common Memory on the PCCard accessed by driving REG# to HIGH. **(default)** 1 = Attribute memory on the PCCard accessed by asserting REG# to LOW.
5:0	Address 25:20	Memory Window 0 Offset Address A25:A20.

Memory Address 0 Start Upper Byte Register (Read/Write)
Index + 17h (CardBus 817h)

This register compares PCI address bits 31:24 for ExCA memory window 0. It decides where the window 0 resides in the 16 Mbyte page memory region at the 4Gbyte PCI address space.

BIT POSITION	NAME	DESCRIPTION
7:0	Address 31:24	Memory Window 0 address start upper address byte A 31:24 are compared to PCI address.

ExCA MEMORY WINDOW 1 - 4 ADDRESS MAPPING REGISTERS (READ/WRITE)

ExCA Memory Addresses 1-4 register functions duplicate Address 0. Below are the register addresses of each of the registers.

Socket I/O Offset	Socket Memory Offset	Register Name
Index + 18h	818h	Memory Window 1 Address Mapping Start LOW Byte
Index + 19h	819h	Memory Window 1 Address Mapping Start HIGH Byte
Index + 1Ah	81Ah	Memory Window 1 Address Mapping Stop LOW Byte
Index + 1Bh	81Bh	Memory Window 1 Address Mapping Stop HIGH Byte
Index + 1Ch	81Ch	Memory Window 1 Offset LOW Byte
Index + 1Dh	81Dh	Memory Window 1 Offset HIGH Byte
Index + 1Fh	841h	Memory Window 1 Address Mapping Start UPPER Byte
Index + 20h	820h	Memory Window 2 Address Mapping Start LOW Byte
Index + 21h	821h	Memory Window 2 Address Mapping Start HIGH Byte
Index + 22h	822h	Memory Window 2 Address Mapping Stop LOW Byte
Index + 23h	823h	Memory Window 2 Address Mapping Stop HIGH Byte
Index + 24h	824h	Memory Window 2 Offset LOW Byte
Index + 25h	825h	Memory Window 2 Offset HIGH Byte
Index + 27h	842h	Memory Window 2 Address Mapping Start UPPER Byte
Index + 28h	828h	Memory Window 3 Address Mapping Start LOW Byte
Index + 29h	829h	Memory Window 3 Address Mapping Start HIGH Byte
Index + 2Ah	82Ah	Memory Window 3 Address Mapping Stop LOW Byte
Index + 2Bh	82Bh	Memory Window 3 Address Mapping Stop HIGH Byte
Index + 2Ch	82Ch	Memory Window 3 Offset LOW Byte
Index + 2Dh	82Dh	Memory Window 3 Offset HIGH Byte
Index + 2Fh	843h	Memory Window 3 Address Mapping Start UPPER Byte
Index + 30h	830h	Memory Window 4 Address Mapping Start LOW Byte
Index + 31h	831h	Memory Window 4 Address Mapping Start HIGH Byte
Index + 32h	832h	Memory Window 4 Address Mapping Stop LOW Byte
Index + 33h	833h	Memory Window 4 Address Mapping Stop HIGH Byte
Index + 34h	834h	Memory Window 4 Offset LOW Byte
Index + 35h	835h	Memory Window 4 Offset HIGH Byte
Index + 3Fh	844h	Memory Window 4 Address Mapping Start UPPER Byte

Card Detect and General Control Register (Read/Write)
Index + 16h (CardBus 816h)

BIT POSITION	NAME	DESCRIPTION
7:6	Reserved	These reserved bits always read "0".
5	Software Card Detect Interrupt	This bit is used by software to generate a Card Change. Writing a 1 to this bit will cause a card status interrupt. For an interrupt to be generated, the Card Detect Change Enable bit in the Card Status Change Interrupt Configuration Register (bit 3, index + 05h) must also be set. The S/W Card Detect Interrupt bit will always read back as 0. 0 = Not interrupt generated. **(default)** 1 = Card change interrupt generated.
4	Card Detect Resume Enable	This bit enables Card Detect Resume. RI_OUT# must be enabled and configured before setting this bit. 0 = Card Detect Resume Disabled. **(default)** 1 = Card Detect Resume Enabled. If the card status change is routed to the IRQ signals, the setting of Card Detect Resume Enable bit to "1" will prevent IRQ signal from going active as a result of card status change. Once the resume software has detected a card detect change interrupt from **RI_OUT#** (by reading the Card status Change register), the software should initiate a software card detect change so the card detect change condition will generate active interrupt on the IRQ signals (depending on the active configuration).
3:1	Reserved	These reserved bits always read "0".
0	16-Bit Memory Delay Inhibit	This bit is used to delay the falling edge of WE# and OE# by 60ns. This bit is only configured for 16 bit memory windows. 0 = Delay enabled. **(default)** 1 = Delay inhibited.

Global Control Register (Read/Write)
Index + 1Eh (CardBus 81Eh)

BIT POSITION	NAME	DESCRIPTION
7:4	Reserved	These reserved bits always read "0".
3	Functional IRQ Level Mode Enable	The bit selects level or edge mode functional interrupt. 0 = Edge Mode. **(default)** 1 = Level Mode.
2	Explicit Write Back Card Status Change Acknowledge	This bit controls the method which Card Status Change Register bits (Index + 04h) are cleared. 0 = Card Status Change Register cleared when read. **(default)** 1 = Card Status Change Register cleared when written.
1	Status Change Interrupt Level Mode Enable	This bit selects level or edge mode status change interrupt. 0 = Edge Mode selected. **(default)** 1 = Level Mode selected.
0	Power Down	This bit enables ExCA Power Down mode. In Power Down mode, I/O Memory Windows are disabled, PCCard accesses are inhibited, and all internal ExCA registers except this register are inaccessible. PCCard inputs are still active. 0 = Power Down mode disabled. **(default)** 1 = Power Down mode enabled.

The O$_2$Micro Mode

O$_2$Micro Mode Control A Register (Read/Write)
Index + 26h (CardBus 826h)

BIT POSITION	NAME	DESCRIPTION
7	Reserved	These reserved bits always read "0".
6:5	Power Mode	This bit selects the power control method X0 = Serial Power **(default)** 01 = Parallel Power Socket mode 11 = SMBus Power mode
4:0	Reserved	These reserved bits always read "0".

O2Micro Mode Control B Register (Read/Write)
Index + 2Eh (CardBus 82Eh)

BIT POSITION	NAME	DESCRIPTION
7	RI_OUT Enable	This bit enables RI_OUT. 0 = Disable RI_OUT. **(default)** 1 = Enable RI_OUT.
6:4	Reserved	These reserved bits always read "0".
3	VS2# Status	This bit reflects the current status of VS2#. Cards that only operate at 3.3V will drive this pin to a "0". 0 = VS2# Low. **(default)** 1 = VS2# High.
2	VSI# Status	This bit reflects the current status of VS1 #. Cards that only operate at 3.3V will drive this pin to a "0". 0 = VSI# Low. **(default)** I = VSI# High.
1:0	Chip ID Select	These bits set the read only ChipID register (index + 00h). **Bit1** **Bit 0** **Chip ID** 1 1 82 – Reserved 1 0 87 – O2Micro Mode 0 1 84 – Inte1365SL C Step **(default)** 0 0 83 – Inte1365SL B Step

O2Micro Mode Control C Register (Read/Write)
Index + 3Ah (CardBus 83Ah)

BIT POSITION	NAME	DESCRIPTION
7:4	Reserved	These reserved bits always read "0".
3	Zoom Video Enable	This register controls the tristating of Zoom Video signals. 0 = Normal Function. **(default)** 1 = Zoom Enabled.
1:0	DREQ Select/Enable	This bit selects which pin DREQ is on. 00 = Disable **(default)** 01 = INPACK# 10 = WP/IOIS16 11 = BVD2/SPKR#

O2Micro Mode Control D Register (Read/Write)
Index + 3Bh (CardBus 83Bh)

BIT POSITION	NAME	DESCRIPTION
7:4	Reserved	Read as 0100
3	CardBus CLKRUN# Enable	This bit enables CardBus CLKRUN#. 0 = CardBus CLKRUN# disabled. **(default)** 1 = CardBus CLKRUN# enabled.
2	PCI CLKRUN# Enable	This bit enables PCI CLKRUN#. 0 = PCI CLKRUN# disabled. **(default)** 1 = PCI CLKRUN# enabled.
1:0	System Interrupt Mode	These bits indicate the type of interrupt mode selected in PCI-Compatible interrupt mode 00 = PC/PCI interrupt mode 01 = Reserved. 10 = Serial IRQ (PCI/Way) interrupt mode 11 = PCI interrupt mode **(default)**

Centralized DMA Register (Read/Write)
Index + 3Ch (CardBus 83Ch)

BIT POSITION	NAME	DESCRIPTION
7:5	Reserved	These reserved bits always read "0".
4	Status Change Interrupt Enable	This bit indicates whether Status Change interrupts will be generated. 0 = Socket Status Change interrupts will not be generated. 1 = Socket Status Change interrupt will be generated. **(default)**
3	Functional Interrupt Enable	This bit indicates whether Functional interrupts will be generated. 0 = CardBus Functional Interrupt will not be generated. 1 = CardBus Functional Interrupt will be generated. **(default)**
2:0	Centralized DMA Channel #	These bits select from DMA Channels 0h - 7h.

FIFO Enable Register (Read/Write)
Index + 3Dh (CardBus 83Dh)

BIT POSITION	NAME	DESCRIPTION
7	PCI Write FIFO Enable	This bit enables PCI to CardBus Memory Write FIFO. 0 = PCI to CardBus Memory Write FIFO disabled. **(default)** 1 = PCI to CardBus Memory Write FIFO enabled.
6	PCI Post Write Enable	This bit enables PCI to CardBus Memory Post Write. 0 = PCI to Cardbus memory post write disabled. **(default)** 1 = PCI to Cardbus memory post write enabled.
5	Reserved	These reserved bits always read "0".
4	PCCard Post Write Enable	This bit enables PCI to Pccard Memory Write Posting. 0 = PCI to PCCard Memory Write Posting disabled. **(default)** 1 = PCI to PCCard Memory Write Posting enabled.
3:0	Reserved	These reserved bits always read "0".

O_2 Mode Control E Register (Read/Write)
Index + 3Eh (CardBus 83Eh)

BIT POSITION	NAME	DESCRIPTION
7:5	Reserved	These reserved bits always read "0".
4:3	SKT_ACTV	This bit selects between SKT_ACTV and LED_OUT function. X1 = LED_OUT enabled. 10 = SKT ACTV enabled.
2	Reserved	These reserved bits always read "0".
1	SPKR_OUT Enable	The bit enables SPKR OUT function. 0 = SPKR_OUT disabled. 1 = SPKR OUT enabled. **(default)**
0	Centralized DMA Enable	This bit enables Centralized DMA function. 0 = Centralized DMA disabled. **(default)** 1 = Centralized DMA enabled

DC CHARACTERISTICS

DC Table for Vcc = 4.5V to 5.5V

Symbol	Parameter	Min	Max	Units
V_{CC}	Power Supply Voltage	4.5	5.5	
V_{IH}	Input HIGH Voltage	2.0	3.5; 2.2 typ.	V
V_{Il}	Input LOW Voltage	0.8	1.5; 0.8 typ.	V
V_{OH}	Output HIGH Voltage	2.4		V
V_{OL}	Output LOW Voltage		0.4	V
I_{IL}	Maximum Input Leakage Current		+/- 10	µA
I_{OL}	Maximum Output Leakage		+/- 10	µA
I_{CC}	Supply Current		50	mA
I_{CC1}	Supply Current / Power Down Mode [Outputs Tri-stated, Internal Clock stop]	35	**-**	µA

DC Table for Vcc = 3.0V to 3.6V

Symbol	Parameter	Min	Max	Units
V_{CC}	Power Supply Voltage	3.0	3.6	
V_{IH}	Input HIGH Voltage	0.5	2.15; 2.0 typ.	V
V_{Il}	Input LOW Voltage	0.3	0.95; 0.8 typ.	V
V_{OH}	Output HIGH Voltage	2.4		V
V_{OL}	Output LOW Voltage		0.4	V
I_{IL}	Maximum Input Leakage Current		+/- 10	µA
I_{OL}	Maximum Output Leakage		+/- 10	µA
I_{CC}	Supply Current		30	mA
I_{CC1}	Supply Current / Power Down Mode [Outputs Tri-stated, Internal Clock stop]	20	**-**	µA

Capacitance

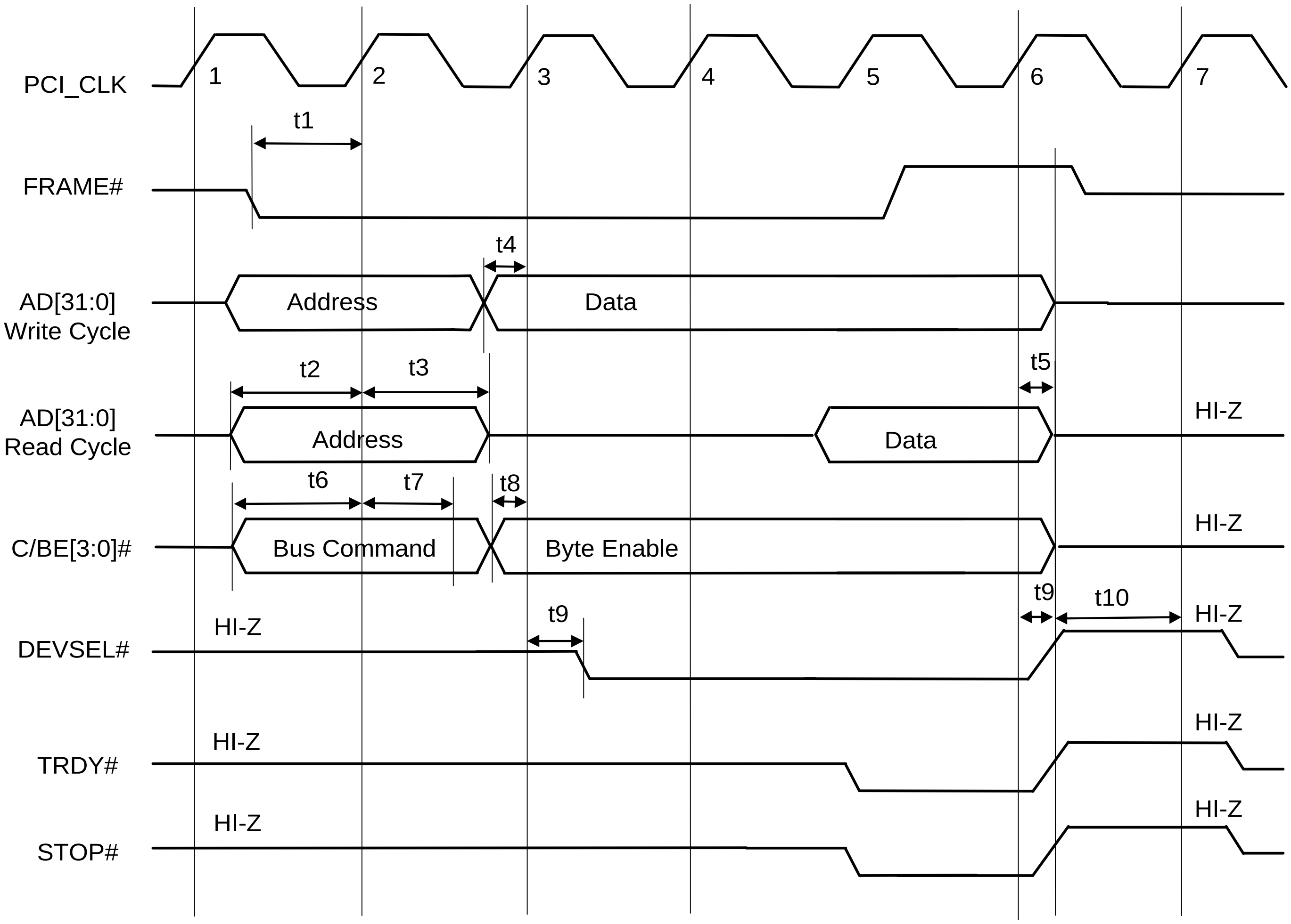
Symbol	Parameter	0 Degree C to 70 Degree C	Units
C_{IN}	Maximum Input Capacitance	10	pF
C_{OUT}	Maximum Output Capacitance	10	pF
C_{IO}	Maximum I/O Capacitance	10	pF

Absolute Maximum Ratings

Symbol	Parameter	Value	Units
V_{CC}	DC Power Supply Voltage	-0.5 to + 7.0	V
$V_{IN,}$ V_{OUT}	DC Input, Output Voltage	-0.5 to V_{DD} + 0.5	V
I	DC Current Drain V_{DD} and V_{SS} Pins	100	mA
T_{STG}	Storage Temperature	-55 to +150	Degree C
T_{L}	Lead Temperature	250	Degree C
$T_{OPER|}$	Operation Temperature	0 to +70	Degree C

AC CHARACTERISTICS

PCI Bus Timing

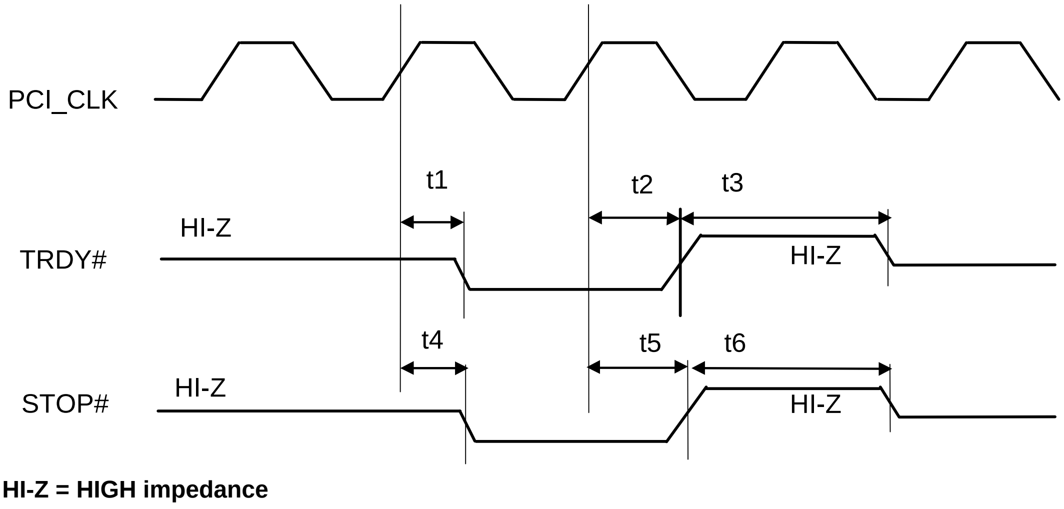


HI-Z = HIGH impedance

FRAME#, AS[31:0], C/BE[3:0]#, DEVSEL#, TRDY# AND STOP#
(PCI Bus)

TRDY#,STOP# Timing

Symbol	Parameter	MIN	MAX	MIN	MAX	Units
t1	TRDY# active delay from PCI_CLK	-	11	-	11	ns
t2	TRDY# inactive delay from PCI_CLK	-	11	-	11	ns
t3	TRDY# HIGH before HI-Z	1	-	1	-	PCI_CLK
t4	STOP# active delay from PCI_CLK	-	11	-	11	ns
t5	STOP# inactive delay from PCI_CLK	-	11	-	11	ns
t6	STOP# HIGH before HI-Z	1	-	1	-	PCI_CLK

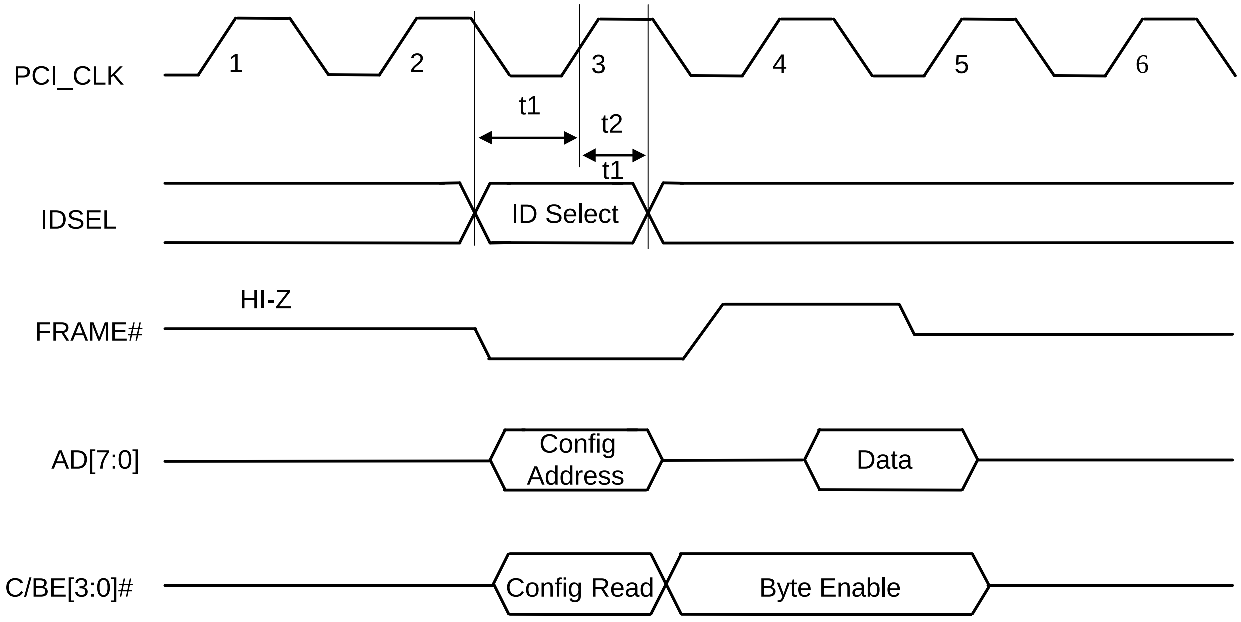


HI-Z = HIGH impedance

TRDY# AND STOP# Delay (PCI Bus)

IDSEL Timing in a Configuration Cycle

Symbol	Parameter	MIN	MAX	Units
t1	IDSEL setup to PCI_CLK	7	-	ns
t2	IDSEL hold from PCI_CLK	0	-	ns

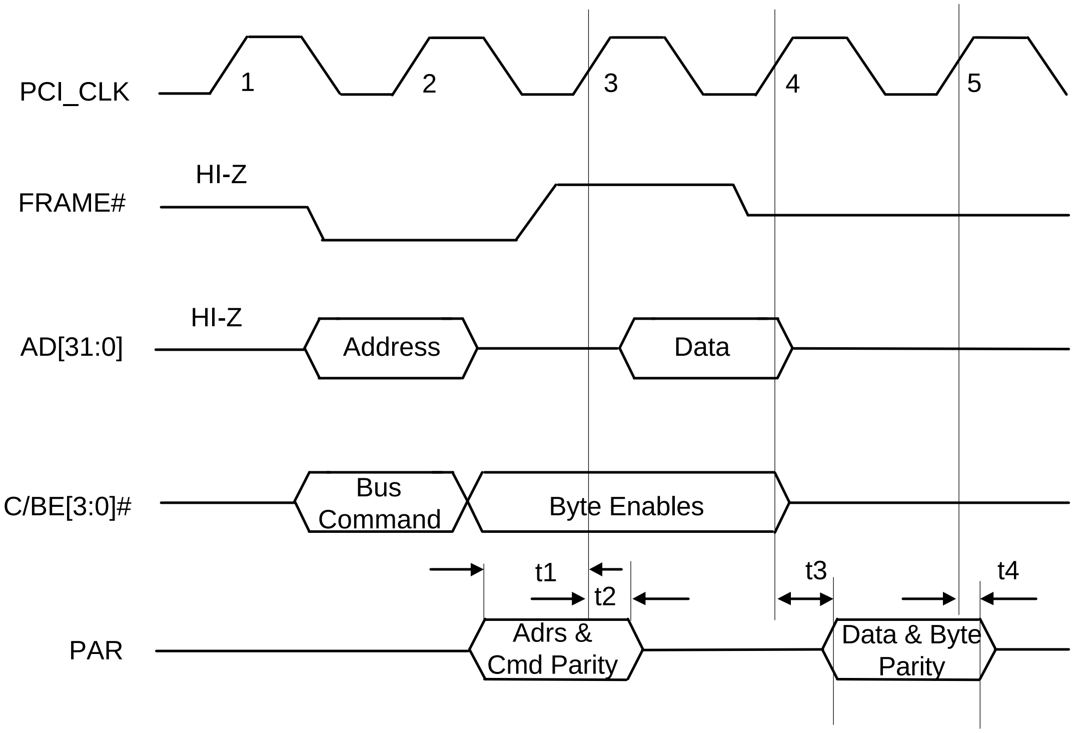


HI-Z = HIGH impedance

IDSEL Timing in a Configuration Cycle (PCI Bus)

PAR Timing

Symbol	Parameter	MIN	MAX	Units
t1	PAR setup to PCI_CLK (input to CL-PD6730)	7	-	ns
t2	PAR hold from PCI_CLK (input to OZ6812)	0	-	ns
t3	PAR valid delay from PCI_CLK (output from OZ6812)	-	11	ns
t4	PAR hold from PCI_CLK (output from OZ6812)	0	-	ns



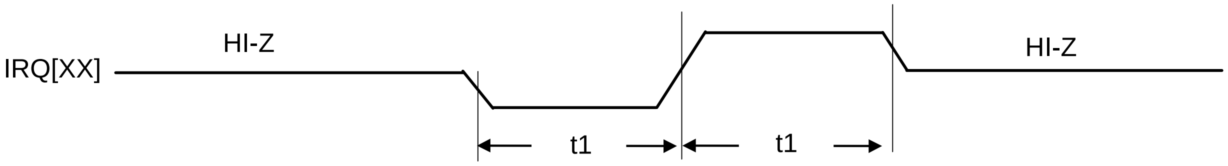
PAR goes HIGH or LOW depending on AD[31:0] and C/BE[3:0]# values.

PAR Timing (PCI Bus)

System Interrupt Timing

Pulse Mode Interrupt Timing

Symbol	Parameter	MIN	MAX	Units
t1	IRQ[XX] LOW or HIGH	16	16	PCI_CLK

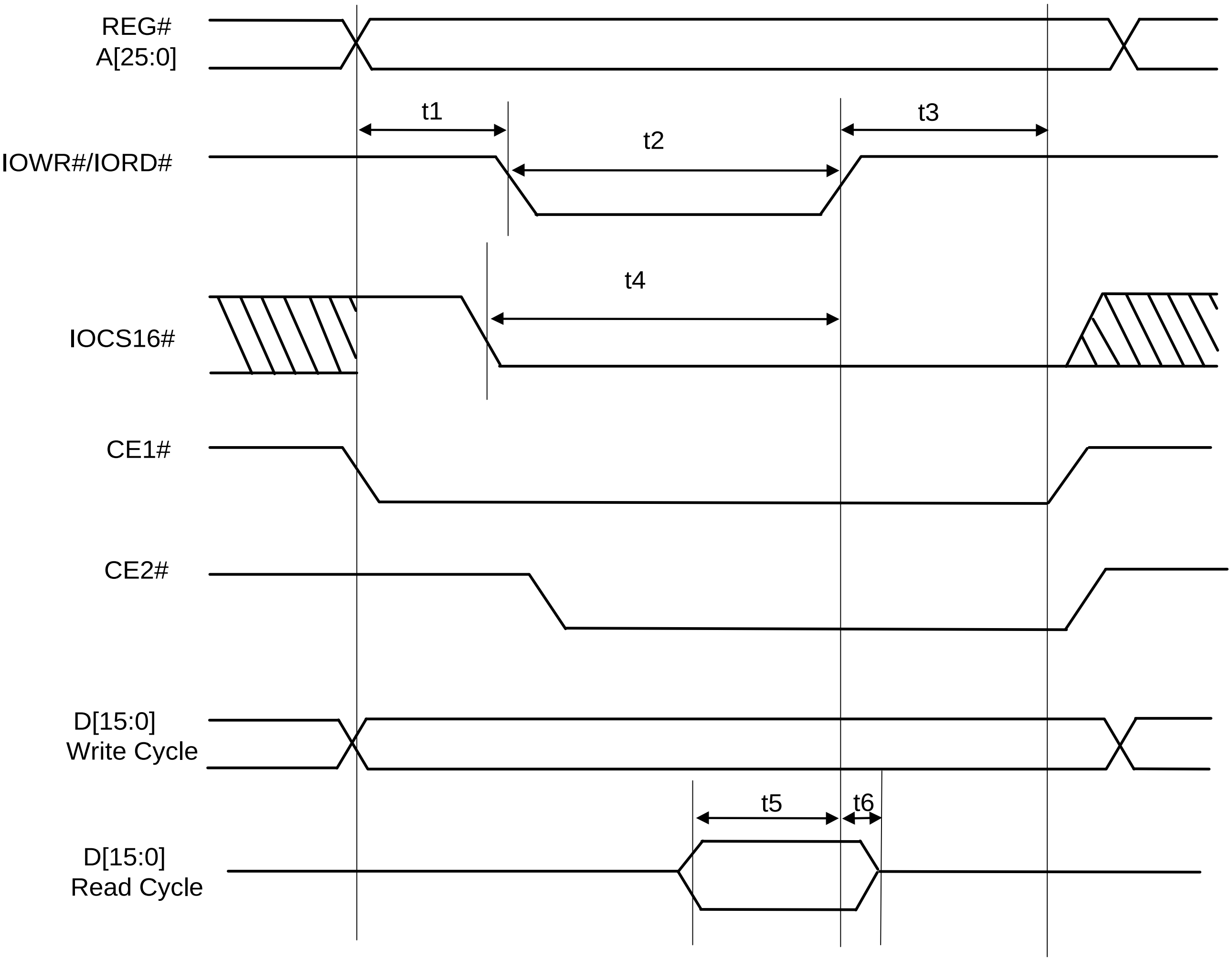


HI-Z = HIGH impedance

Pulse Mode Interrupt Timing

I/O Read/Write Timing

Symbol	Parameter	MIN	MAX	Units
t1	REG# or Address setup to Command active	70		ns
t2	Command pulse width	165		ns
t3	Address hold and Write Data valid from Command inactive	20		ns
t4	Card IOCS16# delay from valid Address(PC Card specification)		35	ns
t5	Data setup before IORD# inactive	60		ns
t6	Data hold after IORD# inactive	0		ns

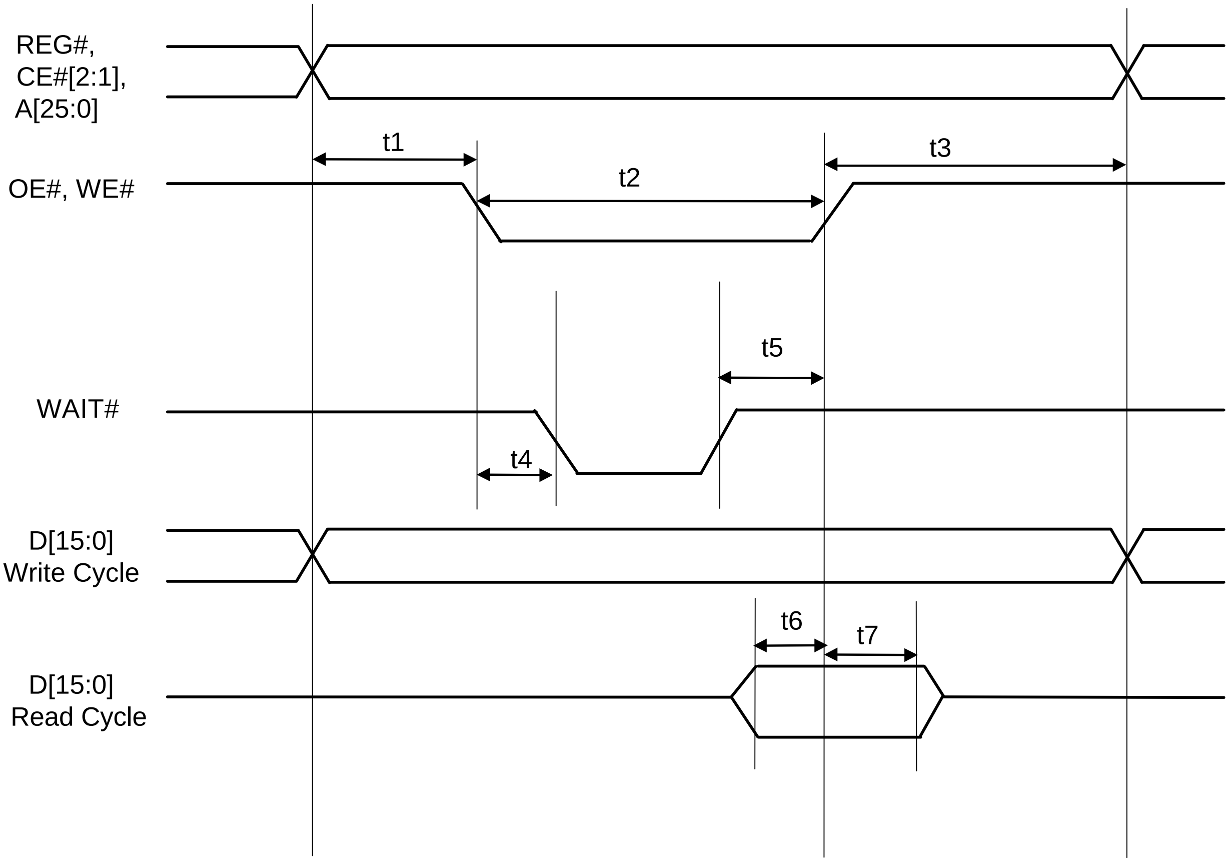


Memory I/O Read/Write Timing

PC Card Bus Timing

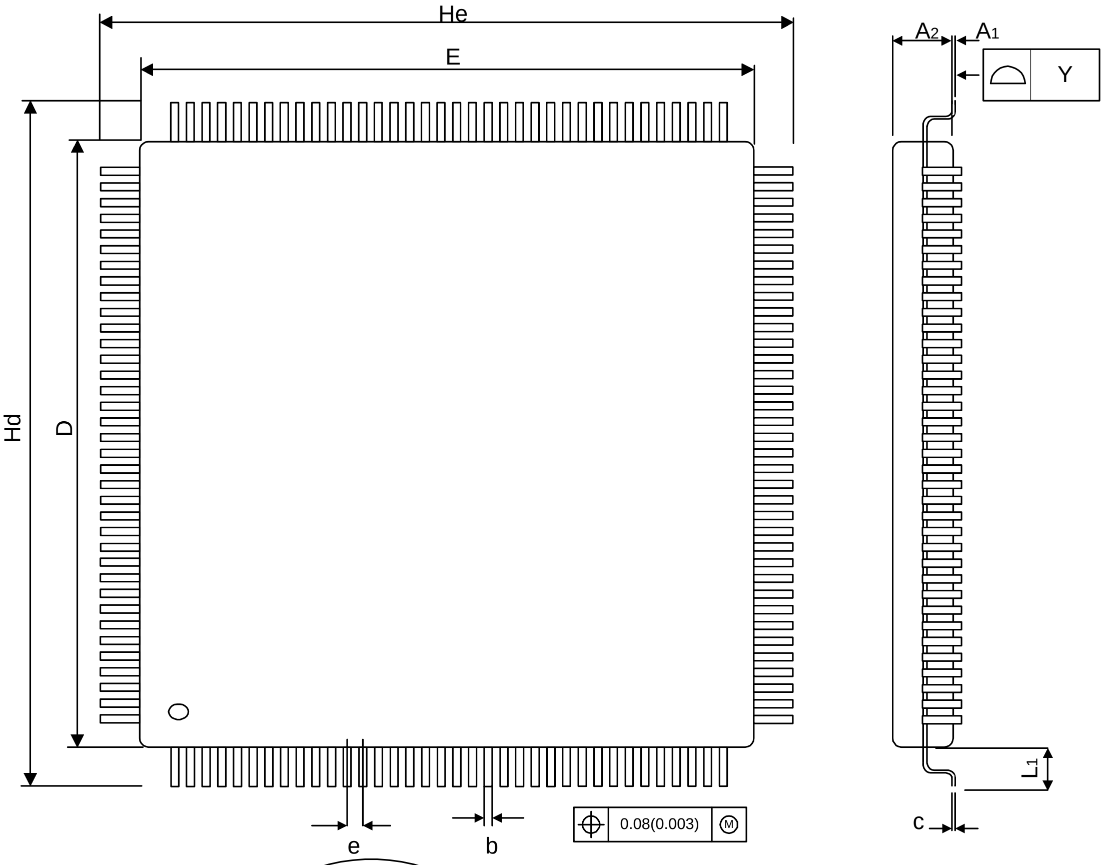
Memory Read/Write Timing

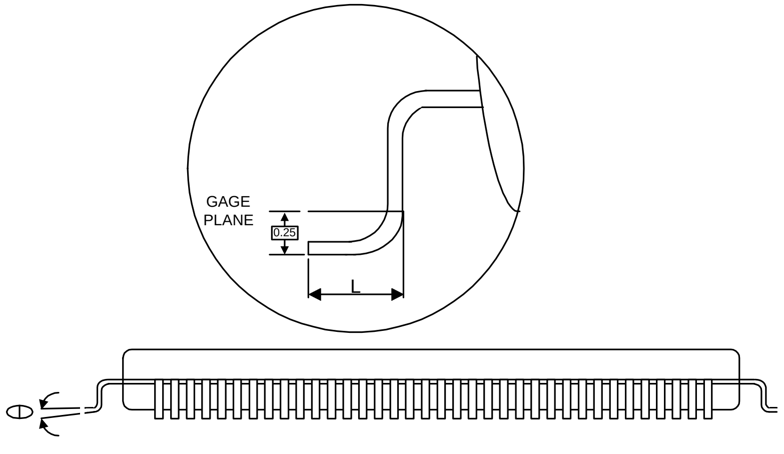
Symbol	Parameter	MIN	MAX	Units
t1	REG#, CE[2:1]#; Address, and Write Data setup to Command active	30		ns
t2	Command pulse width	150		ns
t3	Address hold and Write Data valid from Command inactive	20		ns
t4	WAIT# active from Command active		35	ns
t5	Command hold from WAIT# inactive	0		ns
t6	Data setup before OE# inactive	80		ns
t7	Data hold after OE# inactive	30		ns



Memory Read/Write Timing

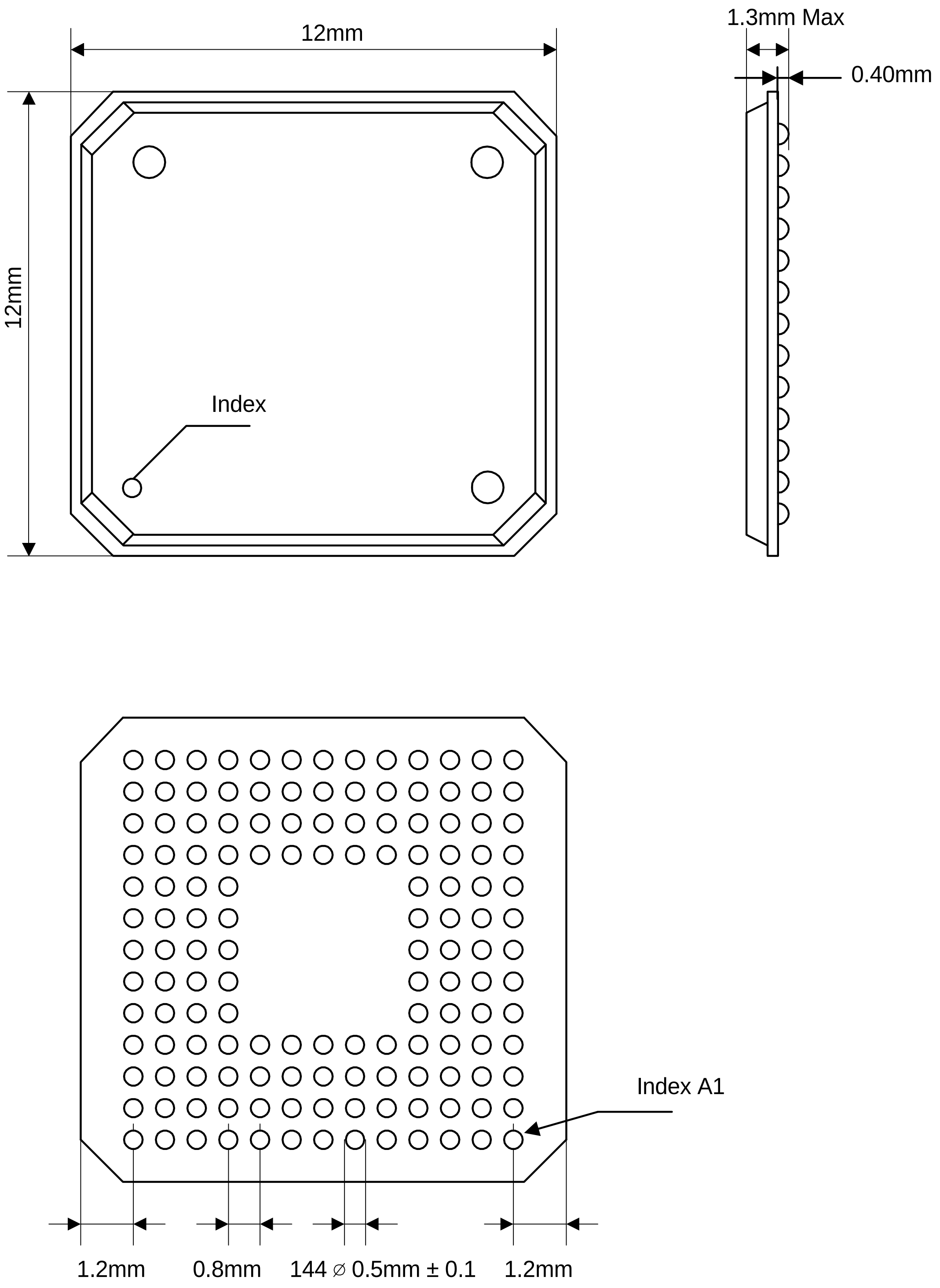
Package Information - 144 Pin LQFP





SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.17	0.22	0.27	0.007	0.009	0.011
c	0.090		0.200	0.004		0.008
D		20.00			0.787	
E		20.00			0.787	
e		0.50			0.020	
Hd		22.00			0.866	
He		22.00			0.866	
L	0.45	0.60	0.75	0.018	0.024	0.030
L1		1.00			0.039	
Y			0.08			0.003
θ	0		7	0		7

144 Pin Mini - BGA



APPENDIX 1 – PCI / CARDBUS COMPARISON

PCI & CardBus Interface Differences

Pin Definition Differences

PCI	CardBus
PCI bus uses IDSEL as a chip select for configuration read and write cycles. The OZ6860 uses IDSEL to enable the PCI system configuration cycles.	CardBus interface does not have an IDSEL signal since CardBus PC Cards are always the intended target of configuration cycles on the socket interface.
SBO# and SDONE are optional pins to signal address snooping results to the target of a transaction.	CardBus does not have the SBO# or SDONE signals.
Support for PCI 64 bit extensions.	CardBus does not have the 64-bit bus extension pins due to pin constraints in the 68-pin connector.
Support for IEEE 1149.1 "Standard Test Access Port and Boundary Scan Architecture".	CardBus does not have the JTAG pins required to support IEEE 1149.1 across the interface.
No support for CSTSCHG pin.	CardBus uses CSTSCHG to signal battery low/dead, write protect, ready, and remote wakeup conditions.
No support for CAUDIO signal.	Usage of this signal requires either a local speaker, a connection to the system speaker via cable, or a sideband signal when mounted on the motherboard.
No support for CCLKRUN# signal	Cardbus uses this hardware mechanism to start the clock, or continue it for a period of time.
PCI allows up to four interrupts (INTx#).	CardBus uses a single interrupt signal (CINT#).
Support for CPERR# and CSERR# is optional.	CardBus must implement CPERR# and CSERR# along with the associated parity generation and checking logic.

Functional Differences

PCI	CardBus
Interrupt Acknowledge command supported.	This command is not supported due to the dynamic insertion/removal nature of PC Cards.
Dual address cycle (DAC) command support	64-bit addressing commands are not supported.
PCI requires any bus master to assume ownership and drive CAD[31:0], CC/BE[3:0]#, and CPAR when CGNT# is asserted but REQ# isn't..	CardBus cards cannot be designated the default owners of an idle interface.

Electrical Differences

PCI	CardBus
Transmission line environment, reflected waves.	Require special slew rate controlled buffers to control signal rise/fall times. Limited number of ground pins on a CardBus connector cannot tolerate the switching characteristics of PCI buffers.
5.0V or 3.3V signaling environment.	Only 3.3V signaling environment.
Tval is measured at the DC 0.4 (Vcc) value.	Tval is measured at the DC Vih and Vil values.
Minimum time from CCLK stable to the deassertion of CRST# is 100 microseconds	Minimum time from CCLK stable to the deassertion of CRST# as 100 clocks.
Standard PCI connector pin assignments.	CardBus connector pins are in a different sequence than PCI to better align CardBus signals with PCMCIA 2.0 and 1.0/JEIDA 4.1 and 4.0 protocols since the adapter must configure the interface for any of the three.
No specification for maximum current immediately after power on or reset.	Specify a maximum current (Icc) immediately following power on or reset

Configuration Space Differences

PCI	CardBus
Predefined header region in configuration space.	Due to dynamic insertion and removal concerns, not all PCI defined configuration space fields are supported. Configuration information is in the CIS.
Optional command register fields.	Memory Space and Parity Error Response must always be implemented.
Optional status register fields.	Signaled System Error must always be implemented since address and data parity errors checking and reporting is required.
No CIS pointer.	Four bytes in the configuration space header are for a CIS pointer.
Separate Memory and I/O.	Memory-mapped base address register for whenever I/O space is used by the card.

Summary of 16-bit PC Card and CardBus differences

Summary of 16-bit PC Card and CardBus differences

16-bit PC card	CardBus
8-, 16-bit interface	8-, 16-, 32-bit interfaces
Non-multiplexed address and data	Multiplexed address and data
Slave only bus	Slave and master capabilities supported
Asynchronous bus	Synchronous bus 33-MHz clock with dynamic clock management capability
Memory, attribute memory, and I/O spaces	Memory, I/O, and configuration spaces
64 MB of address space	4 GB of address space
Fastest cycle time of 100 ns, 20 MB/ sec transfer rate	133 MB/sec peak transfer rate using burst mode
Type I, II, and III form factor	Type I, II, and III form factor
Function configuration registers to configure and control the PC Card interface	Configuration header and function event registers to configure and control the PC Card
Pulse or level-triggered interrupts	Level-triggered interrupts only
Switched Vcc and Vpp lines recommended	Switched Vcc and Vpp lines required
No error checking	Parity based error checking
Separate memory-only and I/O interface	Single interface
Event management support for STSCHG# and WP, BVD1, BVD2, Ready, and Wakeup events	Event management support for CSTSCHG for various card and socket events
Audio binary tone support	Audio, binary, and PWM support
Non-cacheable and nonexclusive transfers	Support for cacheable, and exclusive transfers
3.3V and 5V Vcc	3.3V Vcc only
Voltage sensing to detect threshold	Uses CVS1-2 and CCD1-2# to determine 16-bit PC Card or CardBus PC Card

APPENDIX 2 - PC CARD LOGO CERTIFICATION CHECKLIST

PC Card Basic Requirements:

1) All devices compliant with the PC Card Standard released February 1995

PC Card Socket Controllers:

2) Support Industry-standard ExCA base-register set.
3) Maintain mapping of IRQ Routing Register bits to system interrupt vectors.
4) Support industry-standard definition for CardBus bridges.
5) Support both ISA and PCI interrupts on CardBus controllers.
6) BIOS initializes CardBus controller in 82365-compatible mode and reports it as PNP0E03 for backward compatibility.
7) Writible PCI Configuration Space bits not shared by CardBus controllers.
8) Each R2 memory window in CardBus controller has it own page register.

Plug and Play Design for PC Card 16:

9) Required I/O card tuples supported.
10) Configuration entry tuples listed in priority order.
11) Maximum configuration options specified.

Plug and Play Design for CardBus:

12) Configuration space meets the Common Silicon Guidelines.
13) RESERVED fields compliant with PCI v. 2.1.
14) CardBus required and recommended tuples implemented.
15) Writable PCI Configuration Space bits not shared by CardBus controllers.

Power Management for PC Card:

16) Compliance with "Device Class Power Management Reference Specification" for PC Card controller.
17) PC Card 16 cards implement power-related events using the ReqAttn bit and the #STSCHG mechanism.
18) CardBus controllers and cards implement PCI power management specifications.
19) ZV-compatible PC Cards compliant with the PC Card standard definitions for Zoomed Video.

Device Drivers and Installation for PC Card:

20) No user intervention required for correctly installing devices.
21) Device functional immediately without restarting the system .
22) ZV-compatible PC Card driver uses DirectDraw Live Video Extensions .

APPENDIX 3 - ACPI POWER STATES

	D_0	D_1	D_2	D_3

VCC Configuration	PCI_VCC = ON
	CORE_VCC = ON
	AUX_VCC = ON
	SKTA_VCC = ON/OFF
		PCI_VCC = ON
		CORE_VCC = ON
		AUX_VCC = ON
		SKTA_VCC = ON/OFF
			PCI_VCC = ON/OFF
			CORE_VCC = ON
			AUX_VCC = ON
			SKTA_VCC = ON/OFF
				PCI_VCC = ON/OFF
				CORE_VCC = ON
				AUX_VCC = ON
				SKTA_VCC = ON/OFF

PCI_CLK **CB_CLK**	ON (CLKRUN#) ON	ON (CLKRUN#) ON	ON/OFF (CLKRUN#) OFF	OFF OFF
PME Context	Accessible (Preserved)	Accessible (Preserved)	Accessible (Preserved)	Preserved
PCI Configuration	Accessible	Accessible (Preserved)	Accessible (Preserved)	Preserved
Slot Interface Memory, I/O	Functional	Nonfunctional	Nonfunctional	Nonfunctional
Power Consumption (CardBus Cards Inserted)	40mA	15mA	10mA (CLK running)	10µA
Power State Transaction	D1 D2 D3	D0 D2 D3	D0 D3	D0
Power to Cards	Available	Available	Available	Available
Card Status Change Interrupt	Functional	Mask (Int. Status Pre'd, PME# Inserted)	Mask (Int. Status Pre'd, PME# Inserted)	Mask (Int. Status Pre'd, PME# Inserted)
Functional Interrupt	Functional	Mask	Mask	Mask
PME# Wake up Events	Functional CD Change, CSTSCHG, STSCHG# (Battery warm/dead/ status change, Ring_In)	Functional CD Change, CSTSCHG, STSCHG# (Battery warm/dead/ status change, Ring_In)	Functional CD Change, CSTSCHG, STSCHG# (Battery warm/dead/ status change, Ring_In)	Functional CD Change, CSTSCHG, STSCHG# (Battery warm/dead/ status change, Ring_In)

APPENDIX 4 - MICROSOFT WEBSITE DOCUMENTS

http://www.microsoft.com/hwdev/busbios/CARDBUS1.HTM

CardBus Host Controllers and Windows Compatibility

CardBus Host Controllers Are PCI Bus Bridges

CardBus host controllers are simply a new type of peripheral component interconnect (PCI) bridge. These controllers perform the bus bridging function between a PCI bus and CardBus. To smoothly integrate with the PCI standard, manufacturers of CardBus controllers have developed an industry standard for this type of PCI bridge. This standard defines a PCI header type field (82h), plus a set of required register definitions that represent the standard interface between the host software the controller. The Socket Services drivers for Microsoft® Windows® 95 are written to support this standard definition, even though it might not be formally approved by the PCI Special Interest Group (SIG).

Because CardBus host controllers are PCI bus bridges, they will be supported (enumerated and configured) by the PCI software in Windows 95, just as other PCI bus bridges. The PCI bus bridge support has been enhanced for Windows 95 OSR 2 to support full, dynamic configuration of devices on arbitrary PCI topologies. This support is based on new requirements for PCI interrupt routing and bridge-window configuration currently in process at PCI SIG. Because of this, full compliance with the latest PCI requirements is required for CardBus support.

Special Considerations for CardBus Host Controllers

Because of backward compatibility with 16-bit Exchangable Card Architecture (ExCA) standard, the CardBus (Yenta) specification introduces ambiguity. There are two interfaces through which to program the controller: ExCA registers and CardBus Socket registers. In order to remove this ambiguity and to support both legacy 16-bit software and new CardBus software, these two interfaces must be linked to maintain consistency of the controller and the card's state, regardless of which interface is used.

Therefore, for Windows compatibility, the controller must allow programming from either interface (where it makes sense), and the programming from one side should be reflected automatically to the other side (where applicable). This includes CSC interrupt control, power control, controller status, and card status.

Another special consideration for CardBus controllers is the power protection feature. The Yenta specification requires the controller to do power-protection checking before applying voltages to an inserted card. This feature must be implemented flexibly in order to maintain backward compatibility. Specifically, power protection should be applied only to protect the card from a higher voltage than its VS1/VS2 pins specify; applying a lower voltage should be allowed.

In addition, for R2 cards only, there should be a way to override power protection. Some R2 cards specify higher voltages in their configuration tuples than is reported on their VS1/VS2 pins.

CardBus Host Controller initialization also has special considerations for compatibility with standard Windows drivers. Specifically, CardBus drivers for Windows expect the following:

Interrupt Control

- CSC Interrupt in PCI mode.
- CardBus Card interrupt in PCI mode.
- R2 Card interrupt in ISA mode. Can be dynamically switched to PCI mode (sharing interrupt with CSC), if necessary.
- CardBus Socket Event and Socket Mask registers used to handle CSC interrupt regardless of card type (for both CardBus cards and R2 cards).
- Many CardBus controller implementations allow several ISA interrupt request (IRQ) modes, such as serial ISA IRQ mode. This must be done transparent to software. The controller driver will not perform any initialization to set up and enable these modes.

Power Control

- CardBus Socket Control register used to control power for both CardBus cards and R2 cards. (This is because there is no ExCA register standard on 3.3-volt support).
- 3.3-volt or lower support for CardBus card; 5-volt or 3.3-volt support for R2 cards.

Other Special Considerations

- *R2 Memory Window Control*

The ExCA standard states that an ExCA memory window can only be within 0M to 16M, which is an ISA bus limitation. Because a CardBus controller is by definition a PCI device, this limitation no longer applies. The Yenta specification specifies optional extended R2 memory window base/limit registers at offset 0x840-0x844. For Windows compatibility, these registers are required. Windows depends on these registers in order to map memory beyond 16M for R2 cards if the CardBus controller is behind a positive decode PCI bridge (that is, in a docking station).

- *Configuration Space Registers*

The Yenta specification is incorrect in stating that some bits in the Command registers can be shared between the two functions (that is, the two sockets), in particular for the IOSpaceEnable and MemSpaceEnable bits. This is in error; Windows relies on these bits to disable a CardBus socket. If they were shared, the sockets could not be independently disabled. The same issue applies to the Bridge Control register. For compatibility with Windows, the bits in these registers must not be shared between the two sockets. See the Note on PCI Configuration Space for CardBus later in this article for more information.

BIOS Support for Backward Compatibility

Note: BIOS enumeration applies only to legacy (non-ACPI) systems. ACPI defines a new method for switching a CardBus controller from PCIC to CB mode. PCIC mode is NOT supported in ACPI-aware operating systems. For more information, see the OnNow home page.

For backward compatibility with Windows 95, there are steps the BIOS can take. Specifically, the BIOS must initialize the CardBus controller in Intel® 82365-compatible mode and report it as device PNP0E03, Intel 82365-compatible CardBus controller. Specifically, the requirements are as follows for BIOS POST time (CardBus controller Configuration Space initialization):

- Command register (that is, offset 0x04) set to 0x07 (that is, IOSpaceEnable, MemSpaceEnable, BusMasterEnable)
- All memory and I/O windows (that is, offset 0x1c-0x38) closed (set to base > limit) LegacyBaseAddress (that is, offset 0x44) set to legacy mode I/O base address (such as 0x3e0)
- The RegisterBaseAddress (that is, offset 0x10) set to 0. If support for other environments (Windows 3.1, MS-DOS®) is needed, some other value may be set.
- Interrupt Line register (that is, offset 0x3c) set to 0xff (no IRQ is assigned). If support for other environments (Windows 3.1, MS-DOS) is needed, an assigned IRQ line may be set here. Notice, however, that this register must be set to 0xff at the time that the device is disabled by the operating system and set into CardBus mode (see the BIOS Report for Plug and Play Enumeration section later in this article).

This is for Windows 95 backward compatibility and puts the CardBus controller into legacy mode where Socketsv.vxd (the Windows 95 Socket Services driver) can access it as an Intel PCIC compatible controller at an I/O address (for example, 0x3e0).

Notice that the BIOS must be at least PCI 2.1 compliant and must support the $PIR Interrupt Routing Table. (If the $PIR Table is not supported, the operating system can be configured manually to utilize BIOS function GetIRQRouting [AX=0xb10e], but this may not work on some machines). The $PIR Table must return the necessary PCI IRQ routing information, including the routing information for the CardBus controller.

In general, if the CardBus controller is built in on the system board, there must be a slot routing entry for it in the table. If the CardBus controller is a PCI add-in card, there must be routing information entries for each PCI slot in the system.

BIOS Report for Plug and Play Enumeration

Note: BIOS enumeration applies only to legacy (non-ACPI) systems. ACPI defines a new method for switching a CardBus controller from PCIC to CB mode. PCIC mode is NOT supported in ACPI-aware operating systems. For more information, see the OnNow home page.

During Plug and Play BIOS enumeration, the BIOS should report the CardBus controller as *pnp0e03, with a compatible ID of *pnp0e00 and the I/O resource of two ports (for example, 0x3e0-0x3e1). Windows 95 doesn't know about *pnp0e03, but it does know about *pnp0e00, so it will load Socketsv.vxd (for generic Intel PCIC-compatible controllers) and everything will work with the above BIOS initialization.

The Windows 95 PCI enumerator (Pci.vxd) does not know about the CardBus controller's PCI header (type 2), so it will not report the CardBus controller device at all.

In Windows 95 OSR 2 and beyond, when the BIOS enumerator sees *pnp0e03, it will hide the device and call the BIOS to disable it. When the BIOS receives the disable call for *pnp0e03, it should do the following (but only if the CardBus controller is still in legacy PCIC mode as described in the Note on PCI Configuration Space for CardBus section later in this article; don't do this if the device is already in CardBus mode):

- Command register (that is, offset 0x04) set to 0
- RegisterBaseAddress (that is, offset 0x10) set to 0
- Interrupt Line register (that is, offset 0x3c) set to 0xff
- LegacyBaseAddress (that is, offset 0x44) set to 0
- Other controller-specific initialization required to put the controller in CardBus mode

Notice that Windows 95 OSR 2 can call the BIOS to disable *pnp0e03 multiple times (such as whenever the operating system re-enumerates the hardware). The BIOS should perform the above steps only if the controller is still in legacy PCIC mode. If the controller is already in CardBus mode (that is if LegacyBaseAddress register, offset 0x44, is cleared), then the call should be ignored by returning SUCCESS.

Note on PCI Configuration Space for CardBus

CardBus designers must not share writable PCI Configuration Space bits in a multifunction PCI device. This is a requirement for the "Designed for Microsoft Windows" logo program under the PC 97 guidelines.

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

For more information about PC 97 design requirements for CardBus, see the PC Card requirements defined in PC 97 Hardware Design Guide.

Disclaimer for Working Documents

http://www.microsoft.com/hwdev/busbios/PCCARDWP.HTM

PC Card Standard and Windows 95: A Developer's Update

This paper is an update to the original PCMCIA Developers white paper published in July 1994 as "PCMCIA Card Support in Windows Chicago."

Introduction

Microsoft® Windows® 95 introduced true Plug and Play support for PC Cards. The system-wide Plug and Play infrastructure offered consistent, robust configuration for PC Cards while abstracting bus-specific issues.

The new implementation appears in Windows 95 OSR 2. The purpose of this paper is to provide IHVs and OEMs with all the information needed to support new devices in the Windows environment.

PCI-to-PC Card Bridges

After Windows 95 was released, a new type of controller appeared on the market that interfaced to the PCI bus rather than the ISA bus. This was referred to as a PCI-to-PC Card bridge. Interfacing this device to the system presents some difficulties, particularly with respect to interrupts. Two methods are available to connect interrupt lines to this device: one that is compatible with ISA interrupt requests (IRQs), and the other that takes advantage of the PCI interrupt mechanism. Windows requires the use of the ISA-compatible mechanism for support of PC Card 16 cards. Windows also requires the PCI interrupt mechanism for support of CardBus cards.

For PC Card 16 card support, the system design must maintain the mapping of the PCMCIA controller's IRQ Routing Register bits to system interrupt vectors. This means that when an interrupt is programmed in the controller to occur on the IRQx pin, the system's IRQ routing causes the interrupt controller to generate the interrupt vector for IRQx and no other IRQ.

For CardBus card support, the system design (including the PCI-to-CardBus bridge) must fully support PCI 2.1 plus the additional PCI Interrupt Routing Table and Bridge Window specifications. For details, see the PCI information in the PC 97 Hardware Design Guide.

Note: Systems that implement PCI-to-CardBus bridges must implement both interrupt mechanisms to support both types of cards. The PC Card software in Windows will dynamically configure the bridge to use ISA interrupts for PC Card 16 cards and to use PCI interrupts for CardBus cards.

Multiple Voltage (5V/3.3V) PC Card Support

In future versions of Windows, 3.3V-only and 3.3V/5V cards will be supported. The voltage policy or multiple voltage cards will be to prioritize 3.3V configurations (if supported by the system) over 5V configurations, regardless of the order of the Configuration_Table_Entry tuples. Aside from this specific exception, all other prioritization of configurations will be based on the order of the Configuration_Table_Entry Tuples, as was the case in Windows 95.

Windows 95 does not explicitly support multiple voltage PC Cards, but a 5V/3.3V card can be successfully used in a 5V-only system by taking advantage of the Windows 95 configuration prioritization rules. To do this, the Configuration_Table_Entry tuples for configurations that use 5V must appear before the Configuration_Table_Entry tuples for configurations that use 3.3V. Given this, Windows 95 will give priority to the 5V configurations and configure the card correctly.

Multiple Function PC Cards

Windows will support multiple function PC Cards (MFC cards) in a manner significantly different from how combination PC Cards are supported under the original release of Windows 95. MFC cards are not backward compatible with Windows 95. Combination cards are multifunction cards designed to the PCMCIA 2.1 specification.

For MFC cards, each function on the card is treated independently. Each function can be configured and enabled independently, and each function's hardware must make no assumptions about whether any other function on the card is enabled, configured, or even present.

Similarly, device drivers for functions on MFC cards must not make any assumptions about the existence or state of any other function that might be on the card, and device drivers must not attempt to access or configure any other function.

The sole requirement for drivers to be MFC-compatible is to share interrupts by correctly using the system-provided services for shared interrupts. For information about writing device drives, see the Windows 95 Device Driver Kit (DDK) available through MSDN Professional membership.

MFC cards have different installation requirements from those for combination cards. Most important, there is no need for the card to have a multifunction class device INF file. Instead, only class-specific INF files are required for each function. Each function on the MFC card will receive its own device ID according to the new ID structure defined later in this paper. For MFC functions, the string "DEV#" is added just before the manufacturer ID values, where "#" is the function number starting from 0.

Also, Overriding Logical Configurations are still supported, but are only required if the function's Configuration_Table_Entry tuples are incorrect.

The form of the device ID for MFC cards is as follows:

PCMCIA\Manufacturer string-Product String-DEV#-XXXX-YYYY

where:

- *Manufacturer string* - String one of the Level-1 (L1) version tuple (CISTPL_VERS_1).
- *Product string* - String two of the L1 version tuple (CISTPL_VERS_1).
- *DEV#* - The multifunction device number. For example, for function 0 this value would be DEV0.
- *XXXX* - The manufacturer ID (first word) of the manufacturer ID tuple (CISTPL_MANFID).
- *YYYY* - The product ID (second word) of the manufacturer ID tuple (CISTPL_MANFID).

CardBus Support and Windows 95

CardBus was designed as a combination of PC Card 16 and PCI. In integrating with the system, there are a number of compatibility issues that affect how CardBus cards are supported. To minimize these compatibility issues and to offer the most robust and feature-rich support for CardBus, Windows uses a combination of PC Card software and PCI software to support CardBus cards.

The PC Card software is responsible for enumerating the card, configuring the voltage for the card, and handling dynamic events such as removals or other STSCHG notifications. The PCI software is responsible for configuring the device as part of the overall PCI topology, including bridging and interrupt routing issues. This combination approach offers the best compatibility with existing drivers and BIOSs, while delivering true dynamic Plug and Play support.

CardBus Cards

CardBus cards must fully support the PCI Configuration Space standard. Unfortunately, the CardBus standard allows card vendors to not implement certain critical fields in the Configuration Space (described as "allocated" in the PC Card standard). However, the Common Silicon Guidelines (for silicon that is common to both PCI and CardBus products) do recommend that these fields be implemented. For compatibility with Windows, the Common Silicon Guidelines for Configuration Space must be implemented

The required allocated fields are listed in the following table.

Required Allocated Fields

Field	Description
Vendor ID	This read-only field contains a unique ID (in PCI space) for the manufacturer of the card. It is allocated by the PCI Special Interest Group (SIG).

Device ID Revision ID	These read-only fields are vendor-assigned values that uniquely identify the device (among all vendors PCI or CardBus products).
Class Code	These read-only fields are defined in PCI 2.1. They describe what type of device this card is.
Max_Lat Min_Gnt	These read-only fields specify the desired settings for Latency Timer values, according to PCI 2.1. Values of zero (0) indicate that the device has no major requirements for the settings of latency timers.
Interrupt Line	This register must be read-write and must not be connected to anything, just as on PCI cards. It is used to store the current IRQ routing for the device.

In addition, the CardBus specification lists two fields as RESERVED (offset 2C in the Configuration Space), which have since been defined in PCI 2.1. These are also required on CardBus cards for Windows compatibility.

Required RESERVED Fields for CardBus

RESERVED Field	Description
Subsystem ID	If different than Device ID
Subsystem Vendor ID	If different from Vendor ID

Windows supports the same set of tuples as required by the PC Card standard. This information is used as supplemental information for devices that are not fully described using the PCI Configuration Space. The required tuples are summarized in the following table.

Required and Recommended Tuples for CardBus

Tuple code	Tuple ID	Comments
Required tuples:		
CISTPL_CONFIG_CB	04h	
CISTPL_CFTABLE_ENTRY_CB	05h	
CISTPL_BAR	07h	
CISTPL_LINKTARGET	13h	Required as first tuple by PC Card standard.
CISTPL_VERS_1	15h	
CISTPL_MANFID	20h	
CISTPL_END	FFh	Required as end-of-chain tuple by PC Card standard.
Recommended tuples:		
CISTPL_FUNCID	21h	

CardBus Host Controllers

For information, see CardBus Host Controllers and Windows Compatibility.

New Device IDs for I/O PC Cards

In addition to support for new features, minor enhancements relating to general PC Card support have also been made. For example, the structure of the device ID for I/O PC Cards is being modified to address a problem. The structure used in Windows 95 created an overly unique ID that reflected any and all changes to the CIS of the card and unnecessarily affected the mapping of cards to device drivers. Any tuple change required an associated change to the Device INF file so that Windows 95 could automatically install the device. In many cases, such as tuple bug fixes or other configuration-related changes not relevant to the device driver, this was not the desired effect.

The new method limits ID changes to those that require a new INF file. This method puts the IHV in control of determining when the ID must change, based on any related INF file changes that might be required.

The existing device IDs for PC Cards will continue to be used as an equivalent ID. This ensures that existing cards and INF files will continue to work unmodified. In addition, a new device ID will also be created.

The form of the new device ID is the following:

PCMCIA\Manufacturer string-Product String-XXXX-YYYY

where:

- *Manufacturer string* - String one of the L1 version tuple (CISTPL_VERS_1).
- *Product String* - String two of the L1 version tuple (CISTPL_VERS_1).
- *XXXX* - The manufacturer ID (first word) of the manufacturer ID tuple (CISTPL_MANFID).
- *YYYY* - The product ID (second word) of the manufacturer ID tuple (CISTPL_MANFID).

Notice that for a card to work with an existing INF file, its CIS must contain the same manufacturer string, product string, manufacturer ID, and product ID of the card for which the INF file was created. Conversely, if a new INF file is required for the card, one of these CIS components must be different from that of the original card.

The list of required tuples for I/O PC Cards is not changed in this version of PC Card software.

Update on Memory Card Support

Windows 95 support for memory card is not designed to be Plug and Play as is the support for I/O cards. To maintain compatibility with the Flash File System drivers, memory cards must be supported as legacy devices. However, new memory card technologies can be supported under Windows with a protected-mode Memory Technology Driver (MTD). MTD can be automatically installed and loaded in the same way as Plug and Play device drivers. This is done based on the Plug and Play ID created for the memory card. The device ID for memory cards is based on the PCMCIA JEDEC ID (CISTPL_JEDEC-C, 18h).

The structure of the device ID is the following:

PCMCIA\MTD-<JEDEC_ID>

where *<JEDEC_ID>* consists of the PCMCIA JEDEC ID.

For an example of device IDs for memory cards, see the file MTD.INF in the \Windows\Inf directory of Windows 95.

Conclusion

Windows is supporting the PC Card standard in a compatible, Plug and Play fashion as new capabilities are introduced. Building on the bus-independent Plug and Play infrastructure in Windows 95, these advancements can be made with no impact on development of drivers or on compatibility with existing drivers.

The Dtpl.exe utility, which assists developers in creating a CIS that is compatible with Windows 95, has been updated to reflect the changes described in this article.

Disclaimer for Working Documents

http://www.microsoft.com/hwdev/busbios/DTPL.HTM

PC Card: Display Tuple Utility

DTPL.EXE is a tool designed for interpreting the tuples on a PC Card.

This tool is intended to help you design or verify card information structure (CIS) for use with future Microsoft implementations of PCMCIA support in the Microsoft® Windows®95 operating system. For more information on these implementations, see the Windows 95 DDK.

DTPL runs in real mode on MS-DOS® using real-mode Card Services and Socket Services implementations. It is intended to be run on PC Cards that have not been configured by a Card Services client or enabler. Therefore, Card Services clients and PC Card enablers should not be loaded or active when using this tool.

This tool displays information derived from the tuples on the PC Card. This includes the Plug and Play device ID, the Plug and Play logical configurations, and a display of each tuple on the card.

Usage:

DTPL [**-d**[**c**] [# Bytes]] [**-f** filename] [**-a**] [**-i**] [**-l**] [**-r**] [**-t**] [*Socket#*]

Display tuple data for PC Cards.

-a Displays the alternate Plug and Play device IDs for the PC Card.
-d Displays attribute memory bytes.
-dc Displays common memory bytes.
# *Bytes* Displays an optional number of memory bytes; otherwise, 256 bytes will be displayed.
-f Defines a filename to take the tuple input from instead of calling Card Services.
-i Displays the Plug and Play device IDs for the PC Card.
-l Displays the Plug and Play logical configurations for the PC Card.
-r Displays raw tuple data bytes, formatted for input file.
-t Displays all the tuples on the PC Card.
Socket# Displays an optional socket; otherwise, all sockets will be displayed.

The **-f** option allows the tuple input to be taken from a file. This allows tuples to be interpreted without the need for a real-mode Card Services driver and also allows tuples to be tested before they are placed on a PC Card.

The tuples in this file must be defined in an ASCII HEX format. Each hex value must have two digits. The parser for this is very simple.

Comments can be placed in the tuple file using a semicolon (;) character.

For example, the following tuple is a configuration entry tuple for a COM port.

```
; This is a COM1 port
1B 11 ; Config entry and link values.
E0 01 1D 48 D5 02 1D FC
14 A0 60 F8 03 07 30 3C 00
```

The optional Socket# is ignored when the -f option is used, and the socket for the tuples will be set to 0xFF.

The **-i** option displays the Plug and Play device ID for the PC Card. The Plug and Play device ID is the ID that will be generated by the PCMCIA bus enumerator for the device and used by the Plug and Play system. This ID should be used when generating the INF file for a device.

The device ID is created from the manufacturer name string, the product name string, and a 16-bit CRC of a set of tuples. The ID is created by concatinating the PCMCIA prefix, the manufacturer name string, the product name string, and a 16-bit CRC. The device ID has the following format:

PCMCIA\*Manuf_name-Prod_name-Crc*

The Manuf_name and Prod_name are obtained from the CISTPL_VERS_1 tuple. If the CISTPL_VERS_1 tuple is not available or if the manufacturer name is NULL, then the string UNKNOWN_MANUFACTURER will be included in its place.

The CRC will be created from the following tuple data.

CISTPL_DEVICE
CISTPL_VERS_1
CISTPL_CONFIG
CISTPL_CFTABLE_ENTRY
CISTPL_MANFID

Only the first two strings in the CISTPL_VERS_1 tuple -- the manufacturer name string and the product name string -- are included in the CRC. This is because serial numbers are occasionally placed in the additional strings of this tuple. These serial numbers would cause the CRC to be different for each card even though they are the same.

The tuples are not parsed during this CRC computation to determine the length, so the link value is used as the length of each tuple's data to include in the CRC. The length is reduced if the configuration registers are located in the address range skipped by the link. The offset of the configuration registers is determined from the CISTPL_CONFIG tuple. The offset of each tuple and its existence in attribute memory is determined from the Card Services flags and CIS offset values in the GetTupleData packet. Since these fields are implementation-specific in Card Services, this check is not done in this DTPL tool. As a result, the CRC value created by DTPL may differ from that created by Windows. This rarely occurs; however, the ID created by Windows should be used if they differ.

The total length of the device ID string is limited to 128 characters, including the null terminator. The manufacturer and product name will be truncated to maintain this length restriction in the ID string.

The characters in the manufacturer and product name strings that are greater than or equal to a space (0x20) or less than (0x7F) will be copied into the name string. Any other characters outside this range will be skipped. This makes it easier to include these characters in the INF files for the device.

Spaces and commas in the device ID will be converted to underscores (_).

The **-a** option displays the alternate Plug and Play device ID for the PC Card. An alternate device ID is created that contains the manufacturer ID values in place of the CRC. This alternate device ID has the following format:

PCMCIA\*Manuf_name-Prod_name-Manf-Card*

The Manuf_name and Prod_name are the same as defined in the original device ID. The Manf and Card values are obtained from the CISTPL_MANFID tuple and are displayed as two 16-bit hex values.

The purpose of the alternate ID is to allow card manufacturers to modify the configuration tuples on the card without changing the device ID for the card. This will allow the same INF file to be used for cards with minor updates. Any time the card is modified in such a way that the INF file would require changes, the card portion of the manufacturer ID must be changed.

For more information on the use of this alternate device ID, see PC Card Standard and Windows 95: A Developer's Update.

The **-l** option displays the Plug and Play logical configurations for the PC Card. The Plug and Play logical configurations are generated by the PCMCIA bus enumerator from the configuration and configuration entry tuples. The display of these configurations is intended to provide some feedback on the interpretation of the configuration tuples on the PC Card. The logical configuration will be used by the Plug and Play configuration manager to determine the configuration for the PC Card. The information displayed includes I/O, IRQ, and memory resource data, along with the contents of a Card Services RequestConfiguration argument structure.

The **-r** option displays each tuple as its raw data bytes. This is in a format that can be easily edited and used with the DTPL tool as an input file using the -f option. Notice that any strings that are not data bytes must be commented out with a semicolon (;) or removed from the input file.

The **-t** option displays each tuple in a more readable format. Many of the tuples are interpreted and displayed in an expanded form. For example, this is done for the configuration entry tuples, which are difficult to understand without such expansion. Following the expanded information for each tuple is a dump of the tuple data bytes in an ASCII HEX format, followed by an ASCII dump of each byte in the tuple. Non-printable bytes are diplayed as a period (.) in the ASCII dump. Tuples that are not expanded are displayed in an ASCII HEX format only.

Note: This tool has been tested using the SystemSoft and Phoenix Card Services drivers. If it fails to work on other Card Services implementations, try using the -f option, which reads the tuple data from a file.

Changes in current version (Version 6.02.26):

- Added alternate device ID and multifunction child device ID.
- Added multifunction logical configurations and tuple dumps.
- Added display of custom interface ID from the CISTPL_CONF tuple.
- Added support for access to indirect tuple memory.

Disclaimer for Working Documents

http://www.microsoft.com/hwdev/busbios/pciirq.htm

PCI IRQ Routing Table Specification

Microsoft Corporation, Version 1.0, February 27, 1996

Important: This information applies only to legacy PCs running the Microsoft® Windows® 95 OSR 2 operating system. ACPI-based systems must follow the implementation defined in the ACPI specification. For information, see the OnNow home page.

Purpose

One drawback of Microsoft® Windows® 95 is its inability to dynamically route PCI interrupts to interrupt requests (IRQs). The BIOS must assign IRQs to all PCI devices during power-on self-test (POST), and once chosen, the interrupts cannot be moved.

In order to support assignment and reassignment of PCI IRQs, future versions of Windows will need to know how the system board has wired each PCI slot's interrupt pins to the PCI Interrupt Router's interrupt pins. This information cannot be detected without special-purpose hardware, so Windows must obtain it from another source. This document describes how future versions of Windows will use the BIOS to determine this information.

Assumptions

Each PCI system board consists of one or more slots and a PCI Interrupt Router. Each slot has four interrupt pins, known as INTA#, INTB#, INTC#, and INTD#. The PCI Interrupt Router has several interrupt pins, known as PIRQ1#, PIRQ2#, PIRQ3#, ... PIRQn#. There is no PIRQ0#. The INTn# pins for each slot may be wire OR'd with other INTn# pins from the same or other slots, and these groups of pins may also be connected to a PIRQn# pin on the Interrupt Router.

The actual PIRQ value assigned to each interrupt pin on each Interrupt Router is assigned by the chip-set vendor. Microsoft will work closely with chip-set vendors to assign appropriate PIRQ values for existing and future PCI chip sets.

Overview

The PCI IRQ routing information will be stored in a table in BIOS ROM. The table will be stored on a 16-byte boundary and will contain a signature and checksum for detection and validation purposes. The table will:

- Identify the location of the PCI Interrupt Router.
- Identify a compatible PCI Interrupt Router.
- Identify the IRQs devoted exclusively to PCI usage.
- Show how each slot's interrupt pins are wire OR'd together into links.
- Indicate which link is connected to each of the Interrupt Router's interrupt pins

This data is similar to the data available via the PCI BIOS 2.1 function GetIRQRoutingTable, but it contains the following additional features:

- The link numbers are standardized on a per chip set basis
- The PCI Interrupt Router is identified by Bus and DevFunc
- There is a field for a compatible Interrupt Router
- The table is in ROM and does not rely on the BIOS for tricky 32-bit memory copying code

Detection

The PCI IRQ Routing Table can be detected by searching the system memory from F0000h to FFFFFh at every 16-byte boundary for the PCI IRQ routing signature ("$PIR"). Once the signature is found, the following items need to be validated:

- Version. Must be 1.0.
- Table size. Must be larger than 32 and must be a multiple of 16.
- Checksum. The entire structure's checksum must be 0.

Description

The PCI IRQ Routing Table has the following structure:

Byte Offset	Size in Bytes	Name
0	4	Signature
4	2	Version
6	2	Table Size
8	1	PCI Interrupt Router's Bus
9	1	PCI Interrupt Router's DevFunc
10	2	PCI Exclusive IRQs
12	4	Compatible PCI Interrupt Router
16	4	Miniport Data
20	11	Reserved (Zero)
31	1	Checksum
32	16	First Slot Entry
48	16	Second Slot Entry
(N + 1) * 16	16	Nth Slot Entry

Signature:
The signature for this table is the ASCII string "$PIR". Byte 0 is a 24h, byte 1 a 50h, byte 2 is a 49h, and byte 3 is 52h.

Version:
The version consists of a Minor version byte followed by a Major version byte. Since this specification describes the Version 1.0 table format, byte 4 of the table is a 00h and byte 5 is a 01h.

Table Size:
This is a 16-bit value that holds the size of the PCI IRQ Routing Table in bytes. If there were five slot entries in the table, this value would be 32 + (5 * 16) = 112.

PCI Interrupt Router's Bus
This contains the bus number of the PCI Interrupt Router device.

PCI Interrupt Router's DevFunc
This contains the Device and Function number of the PCI Interrupt Router device. The Device is in the upper five bits, the Function in the lower three.

PCI Exclusive IRQs:
This is an IRQ bitmap that indicates which IRQs are devoted exclusively for PCI usage. For example, if IRQ11 is devoted exclusively to PCI and cannot be assigned to an ISA device, then bit 11 of this 16-bit field should be set to 1. If there are no IRQs devoted exclusively to PCI, then this value should be 0.

Compatible PCI Interrupt Router:
This field contains the Vendor ID (bytes 10 and 11) and Device ID (byes 12 and 13) of a compatible PCI Interrupt Router, or zero (0) if there is none. A compatible PCI Interrupt Router is one that uses the same method for mapping PIRQn# links to IRQs, and uses the same method for controlling the edge/level triggering of IRQs. This field allows an operating system to load an existing IRQ driver on a new PCI chip set without updating any drivers and without any user interaction.

Miniport Data:
This DWORD is passed directly to the IRQ Miniport's Initialize() function. If an IRQ Miniport does not need any additional information, this field should be set to zero (0).

Reserved:
These bytes are reserved for future use and must be set to zero (0).

Checksum:
This byte should be set such that the sum of all of the bytes in the PCI IRQ Routing Table, including the checksum, and all of the slot entries, modulo 256, is zero.

Slot Entry:
Each slot entry is 16-bytes long and describes how a slot's PCI interrupt pins are wire OR'd to other slot interrupt pins and to the chip set's IRQ pins. Each entry has the following format:

Byte Offset	Size in Bytes	Name
0	Byte	PCI Bus Number
1	Byte	PCI Device Number (in upper five bits)
2	Byte	Link Value for INTA#
3	Word	IRQ Bitmap for INTA#
5	Byte	Link Value for INTB#
6	Word	IRQ Bitmap for INTB#
8	Byte	Link Value for INTC#
9	Word	IRQ Bitmap for INTC#
11	Byte	Link Value for INTD#
12	Word	IRQ Bitmap for INTD#
14	Byte	Slot Number
15	Byte	Reserved

PCI Bus Number:
The bus number of the slot.

PCI Device Number:
The device number of the slot.

Link Value for INTn#:
A value of zero means this interrupt pin is not connected to any other interrupt pins and is not connected to any of the Interrupt Router's interrupt pins.

The non-zero link values are specific to a chip set and decided by the chip-set vendor. Here is a suggested implementation:

A value of 1 through the number of interrupt pins on the Interrupt Router means the pin is connected to that PIRQn# pin of the Interrupt Router.

- A value larger than the number of interrupt pins on the Interrupt Router means the pin is wire OR'd together with other slot interrupt pins, but the group is not connected to any PIRQn# pin on the Interrupt Router.

- Other interpretations of the link values are possible. For instance, the link value may indicate which byte of Configuration Space to access for this link, or which I/O Port to access for the link. The specific interpretation of the link value is decided by the manufacturer of the Interrupt Router and is supported by the driver for that router.

IRQ Bitmap for INTn#:
This value shows which of the standard AT IRQs this PCI's interrupts can be routed to. This provides the routing options for one particular PCI interrupt pin. In this bitmap, bit 0 corresponds to IRQ0, bit 1 to IRQ1, and so on. A 1 bit in this bitmap indicates that routing is possible; a 0 bit indicates that no routing is possible.

This bitmap must be the same for all pins that have the same link number.

Slot Number:
This value is used to communicate whether the table entry is for a system-board device or an add-in slot. For system-board devices, the slot number should be set to zero. For add-in slots, the slot number should be set to a value that corresponds with the physical placement of the slot on the system board. This provides a way to correlate physical slots with PCI device numbers.

Values (with the exception of zero) are OEM-specific. For end-user ease-of-use, slots in the system should be clearly labeled (such as solder mask, back panel, and so on).

It should be noted that the slot entries of the PCI IRQ Routing Table are compatible with the PCI IRQ Routing Options Table of the PCI BIOS Specification, Revision 2.1. This makes it possible to support both the PCI IRQ Routing Table and the PCI BIOS specification with only one table in ROM.

Notes

Moving the Interrupt Routing Table from a PCI BIOS call into a ROM table has a couple of side effects. Since the table is in ROM, it is no longer dynamic. There are a couple of scenarios related to bridges and docking where a dynamic table is a disadvantage.

Every time a PCI device, slot, or Interrupt Router is described by location in the IRQ Routing Table, it includes the device's bus. Unfortunately, the PCI bus number of a device behind a PCI-to-PCI bridge can change without even calling the PCI BIOS. Since the IRQ Routing Table is in ROM, the table cannot be updated on the fly to reflect new bus numbers.

PCI dockable portables have the ability to add new PCI devices and slots while the system is running. Since the IRQ Routing Table is in ROM, the table cannot be updated on the fly to describe new devices.

How each of these problems gets addressed depends to a large extent on the design of the PCI bus architecture of the computer. Here are some examples:

Single PCI Bus Desktop System with PCI-to-PCI Bridge Add-in Card

The routing table for a single bus desktop system with a PCI-to-PCI bridge add-in card does not need to include the IRQ routing for devices behind the add-in card. The PCI-to-PCI bridge specification already describes how to route the INTn# lines from the bridge's children to the bridge's INTn# lines. The IRQ Routing Table in the BIOS only needs to describe the routing of the bridge's INTn# lines to the system's PCI Interrupt Router. There is no problem supporting such systems with an IRQ Routing Table in ROM.

Dockable Portable Docking Through a Transparent PCI-to-PCI Bridge

Transparent PCI-to-PCI bridges do not support primary, secondary, or subordinate bus numbers. The devices behind the bridge have the same PCI bus number as the transparent bridge itself. This architecture is commonly seen on dockable PCI portables, where the docking station devices appear behind the transparent bridge.

Since the bridge is invisible to the PCI bus hierarchy, the operating system does not have any idea how the IRQs should be routed through the bridge. In this case, the IRQ Routing Table must report the IRQ routing for all devices behind the PCI-to-PCI bridge, even when the dock is not currently present.

However, this could prove to be a problem if the portable can dock to multiple docking stations, each with different routings. Manufacturers who make such docking hardware will have to ensure that the IRQ Routing Table in ROM is the union of all known and planned docks for the portable, and that two docks do not use different routing for the same Dev.

For example, if one dock routes Dev 9 Pin A to Link 2, and another dock routes Dev 9 Pin A to Link 3, then that system cannot be supported by this specification.

Dockable Portable Docking Through a PCI-to-PCI Bridge to Separate PCI Bus

Other portables support docking via a PCI-to-PCI bridge (either positive or subtractive decode) that supports the primary/secondary/subordinate bus numbers. Devices behind the PCI-to-PCI bridge have their own PCI bus number. The simplest approach to supporting these systems is to follow the PCI-to-PCI bridge specification for routing the bridge's secondary bus INTn# pins to the bridge's primary bus INTn# pins.

If a vendor implements IRQ routing through a non-standard routing mechanism, the best solution is for the PCI BIOS to configure the PCI-to-PCI bridge to the same bus number on each boot, and then to put the bus's devices and slots into the IRQ Routing Table. An application or the operating system could change the bus numbers at a later time, but at least the ROM table was accurate on bootup.

This problem is much more difficult to solve when the portable boots up undocked and then later warm or hot docks. The BIOS should not assign any bus numbers when warm or hot docking (that is the responsibility of the operating system). Yet, the BIOS has no idea what bus numbers the operating system will use when it docks, so it has no idea what bus numbers to use in the IRQ Routing Table.

IRQ routing support for such PCI systems will likely be poor or non-existent. It is strongly recommended that vendors follow the standard PCI-to-PCI bridge IRQ routing on internal or proprietary PCI-to-PCI bridges.

Dual PCI ("Peer") Bus System

Some systems have two PCI buses hanging off the CPU. Since one bus is not sitting behind a PCI-to-PCI bridge, there is no standard routing mechanism for devices behind the non-zero PCI bus. Devices on both buses must appear in the IRQ Routing Table. Since the top level PCI bus numbers rarely change (and are often built directly into the hardware), it should be easy to support such systems with an IRQ Routing Table in ROM.

See also PIIXA and IRQ Routing for OSR 2, which describes how to get PCI IRQ routing to work with the PIIXA chip set.

Disclaimer for Working Documents

http://www.microsoft.com/hwdev/busbios/PCIIxa.HTM

PIIXA and IRQ Routing for OSR 2

PCI IRQ routing has been supported under the Microsoft®Windows® 95 operating system since the release of OEM Service Release 2 (OSR 2). It is also supported under the Windows 98 operating system. With this support, the BIOS should not configure any non-boot PCI or ISA device. All PCI devices do not need to receive individual interrupt requests (IRQs) if the operating system can provide IRQ routing on the particular PC, which is determined for each chip set supported by Windows 95. This is based on whether a valid IRQ table is available either from Advanced Configuration and Power Interface (ACPI) table or the $PIR table, or from the real-mode or protected-mode BIOS.

Complete information about how to implement this support for Windows 95 OSR 2 is available in the PCI IRQ Routing Table Specification. This method should be used until systems include an ACPI-compliant BIOS.

To get IRQ routing to work with the PIIX4 chip set, a miniport driver must be implemented. (PIIX4 is compatible with PIIX 2 and uses the same miniport.) Therefore, the OEM must implement one of the following solutions:

1.Implement a $PIR table with the PIIX2. To do this, change the BIOS, adding the PIIX2 (8086-122E) as the compatible PCI Interrupt Router in the $PIR table. This will load the PIIX2 IRQ miniport on a PIIX4 system. Notice that this will not work with real-mode and protected-mode BIOS routing tables.

2.Implement the INF file line in Machine.inf that tells Windows 95 which miniport to load if the machine supports only BIOS calls to get the IRQ routing table. To do this, use the same routing table for PIIX4 as for PIIX2, and add the following entry in the Machine.inf for PIIX4 so that the operating system loads the PIIX2 miniport for PIIX4 systems. Search Machine.inf for the following:

```
;IRQ Miniport Data
;
HKLM,System\CurrentControlSet\Services\VxD\PCI\IRQMiniports\122E8086,Name,,"Intel 82371FB"
HKLM,System\CurrentControlSet\Services\VxD\PCI\IRQMiniports\122E8086,Path,,"PCIMP.PCI"
HKLM,System\CurrentControlSet\Services\VxD\PCI\IRQMiniports\122E8086,Instance,1,01,00,00,00
```

After this entry, add:

```
HKLM,System\CurrentControlSet\Services\VxD\PCI\IRQMiniports\71108086,Name,,"Intel 82371FB"
HKLM,System\CurrentControlSet\Services\VxD\PCI\IRQMiniports\71108086,Path,,"PCIMP.PCI"
HKLM,System\CurrentControlSet\Services\VxD\PCI\IRQMiniports\71108086,Instance,1,01,00,00,00
```

Reference for PCI IRQ routing:

If you need technical assistance with Windows 95 or Windows 98 miniports, please contact ihv@microsoft.com with "Miniports" in the subject line. Please include your name, title, company name, and phone and fax numbers.