PLX TECHNOLOGY®
THE I/O INTERCONNECT SOLUTION

PCI 6150 (HB4)
PCI-to-PCI Bridge
Data Book

PCI 6150 (HB4)
PCI-to-PCI Bridge
Data Book

Version 2.0

May 2003

Website: http://www.plxtech.com
Technical Support: http://www.plxtech.com/support
Phone: 408 774-9060
800 759-3735
Fax: 408 774-2169

PLX Part Number: PCI 6150-BB66PC; Former HiNT Part Number: HB4

Order Number: 6150-SIL-DB-P1-2.0

Printed in the USA, May 2003

PCI 6150 PCI-to-PCI Bridge

High Performance Asynchronous 66MHz 32-bit PCI-to-PCI Bridge for Servers, Storage, Telecommunication, Networking and Embedded Applications

PLX's latest PCI 6150 32-bit PCI-to-PCI bridge is designed for high performance, high availability applications in hot swap, bus expansions, programmable data transfer rate control, frequency conversions from slower PCI to faster PCI or from faster PCI to slower PCI buses. PCI 6150 has sophisticated buffer management and buffer configuration options designed to provide customizable performance optimization.

PCI 6150 has the biggest data FIFO among all 32-bit PCI-to-PCI bridges in the market.

- PCI Local Bus Specification Rev 2.3 support
- High speed PCI buffer supports 3.3V signaling with 5V input signal tolerance
- Asynchronous design supports standard 66Mhz to 33MHz and faster secondary port speed such as 33Mhz to 40/50/60/66MHz conversion
- Programmable Address Translation to Secondary Bus
- Flow-Through 0 wait state burst up to 4K bytes for optimal large volume data transfer
- Supports up to 4 simultaneous posted write transactions and 4 simultaneous Delayed transactions in each direction
- Provides 1K Bytes of buffering
  - 256 byte upstream posted write buffer
  - 256 byte downstream posted write buffer
  - 256 byte upstream read data buffer
  - 256 byte downstream read data buffer
- Programmable prefectch amount of up to 256 bytes for maximum read performance optimization
- Supports out of order delayed transactions
- CPCI Hot Swap Specification PICMG 2.1 R2.0 with PI = 1 support
- Device Hiding support eliminates mid-transaction extraction problems
- Serial EEPROM loadable and programmable PCI READ ONLY Register configurations.
- External arbiter or programmable arbitration for 9 bus masters on secondary interface support
- 10 Secondary clock outputs with pin controlled enable and individual maskable control
- Power Management D3 Cold Wakeup capable
- 4 GPIO pins with output control
- Enhanced address decoding
  - Support 32-bit I/O address range
  - 32-bit memory-mapped I/O address range
  - ISA aware mode for legacy support in the first 64KB of I/O address range
  - VGA addressing and palette snooping support
- Provides an IEEE standard 1149.1 JTAG interface for boundary scan test
- PCI 6150 uses Industry standard 208 pin PQFP package

HISTORY

Rev	Date	Description	Eng Chk	Mkt Chk
Rev 1.2	5/6/02	PCI 6150 Rev BA1 data book new release. • Major rewrite of the PCI 6150 data book from previous revisions. • Updated Hot Swap Capabilities.		
Rev 1.21	6/3/02	Added JTAG external pull up/low resistor requirement		
Rev 1.22	8/23/02	Update DC characteristic table		
Rev 2.0	5/28/03	This release reflects PLX part numbering. • Added Silicon Revision BB to Section 2 • Section 5, changed pin type in tables for P_SERR#, S_SERR#, S_REQ#[0], S_GNT#[8:1], S_M66EN, S_CLK[9:0], and S_RSTOUT# • Removed Section 6.2, Extended Register Map • Removed Section 6.3.3, Extended Registers • Updated Configuration Map in Section 6.1 to reflect deletion of Extended Registers • Updated Register DEh, bits 15-11 • Added three notes to table in Section 14.5, Frequency Division Options • Updated Section 20.3.1, 24-3Fh		

CONTENTS

1 Register Index

2 Ordering Information

Part Number	Rev.	Description
PCI 6150	BA	32-bit PCI-to-PCI bridge
PCI 6150	BB	32-bit PCI-to-PCI bridge

3 Using the PCI 6150

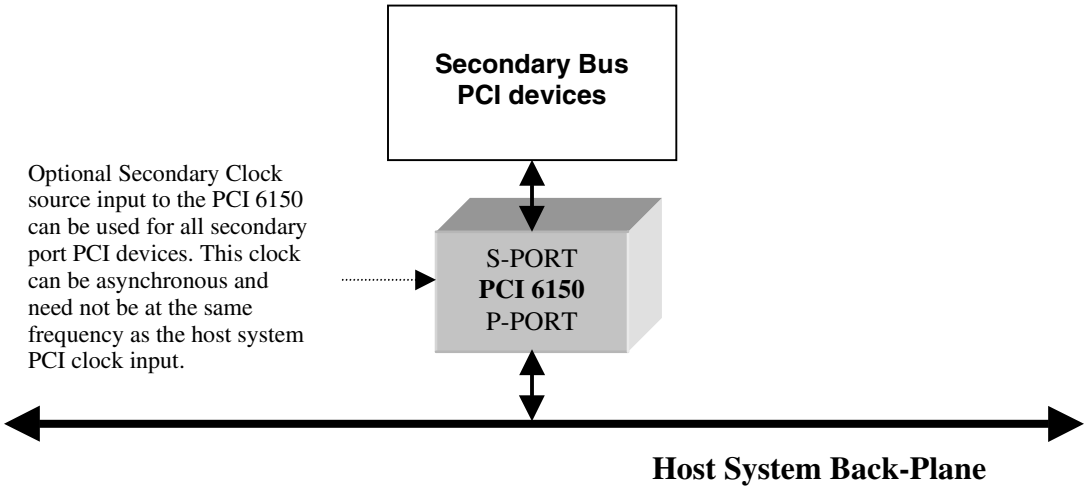
3.1 Transparent Mode Application

Since the PCI 6150 Primary and Secondary ports are asynchronous to each other, the two independent systems can run at different frequencies. It is possible to run the Secondary bus faster than the Primary bus.

PCI 6150 has powerful programmable buffer control which can be used to regulate data throughput for multiple PCI masters on the secondary port. The data prefetch size can be programmed to up to 256 bytes.

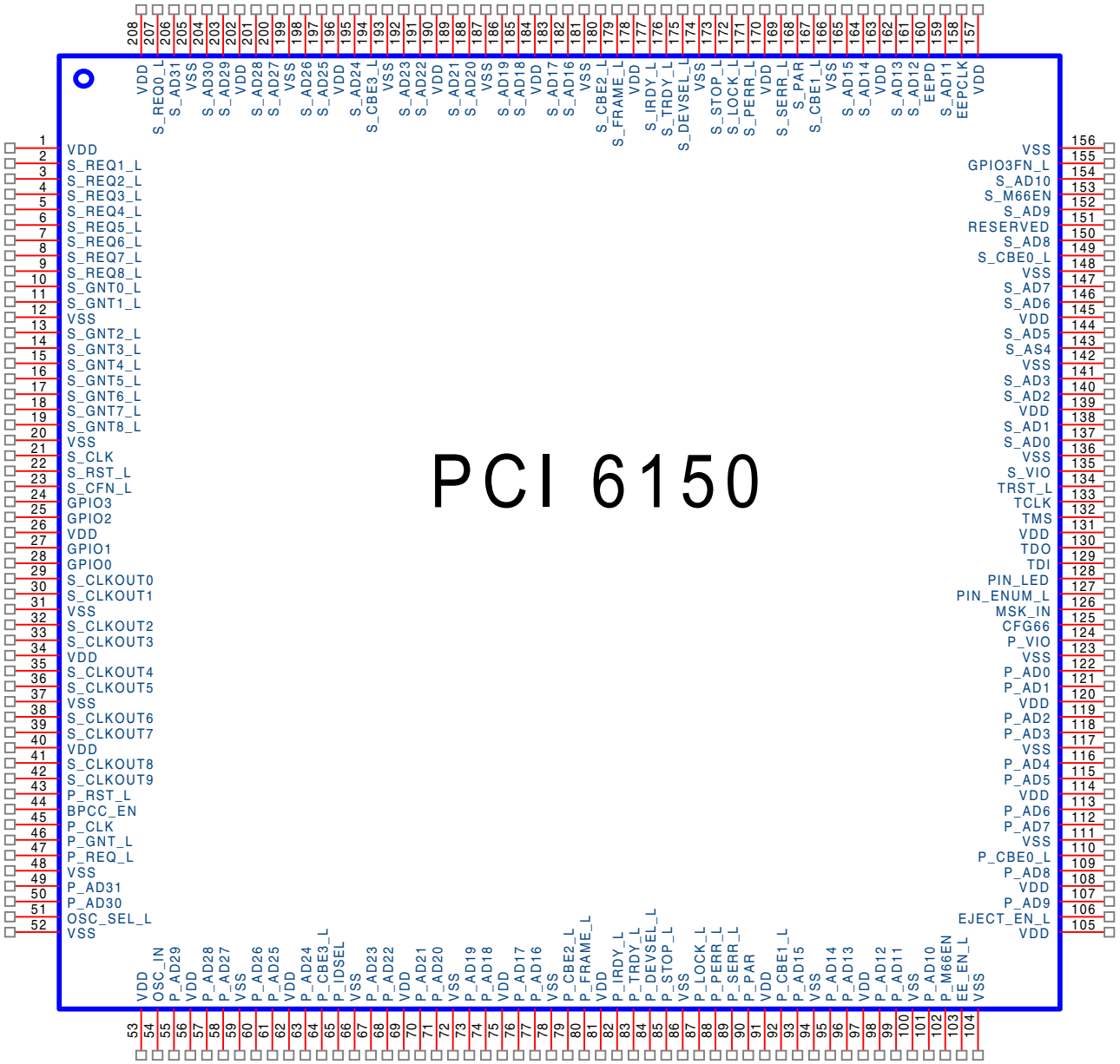
In Transparent Mode, the host system PCI bus is connected to the PCI 6150 Primary port. The Secondary PCI port can use either a custom designed external arbiter or the PCI 6150 internal arbiter. Designers can either use custom designed clock generations, or the PCI 6150 S_CLK[9:0] outputs derived out of the Primary port PCI clock input or an external oscillator, to provide clocks to secondary PCI devices and the PCI 6150 S_CLKIN input.

The basic design idea is optimized for the following:



4 Pin Diagram

PCI 6150 TOP VIEW



5 Signal Definition

Signal Types

PI	PCI Input (5V signal input tolerant, I/O VDD=3.3V)
PTS	PCI Three-state bi-directional (5V signal input tolerant, I/O VDD=3.3V)
PO	PCI Output
PSTS	PCI Sustained Three-state Output. (5V signal input tolerant, I/O VDD=3.3V)
I	CMOS Input
O	CMOS Output
IO	CMOS Bi-direct
PU	Signal is pulled-up internally
PD	Signal is pulled-down internally

5.1 Primary Bus Interface Signals

Name	Type	Description
P_AD[31:0]	PTS	**Primary address/data**: Multiplexed address and data bus. Address is indicated by P_FRAME# assertion. Write data is stable and valid when P_IRDY# is asserted and read data is stable and valid when P_TRDY# is asserted. Data is transferred on rising clock edges when both P_IRDY# and P_TRDY# are asserted. During bus idle, PCI 6150 drives P_AD to a valid logic level when P_GNT# is asserted.
P_CBE[3:0]	PTS	**Primary command/byte enables**: Multiplexed command field and byte enable field. During address phase, the initiator drives the transaction type on these pins. After that the initiator drives the byte enables during data phases. During bus idle, PCI 6150 drives P_CBE[3:0] to a valid logic level when P_GNT# is asserted.
P_PAR	PTS	**Primary Parity**: Parity is even across P_AD[31:0], P_CBE[3:0], and P_PAR (i.e. an even number of '1's). P_PAR is an input and is valid and stable one cycle after the address phase (indicated by assertion of P_FRAME#) for address parity. For write data phases, P_PAR is an input and is valid one clock after P_IRDY# is asserted. For read data phase, P_PAR is an output and is valid one clock after P_TRDY# is asserted. Signal P_PAR is three-stated one cycle after the PAD lines are three-stated. During bus idle, PCI 6150 drives PPAR to a valid logic level when P_GNT# is asserted.
P_FRAME#	PSTS	**Primary FRAME**: Driven by the initiator of a transaction to indicate the beginning and duration of an access. The deassertion of P_FRAME# indicates the final data phase requested by the initiator. Before being three-stated, it is driven to a deasserted state for one cycle.
P_IRDY#	PSTS	**Primary IRDY**: Driven by the initiator of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle.
P_TRDY#	PSTS	**Primary TRDY**: Driven by the target of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle.

P_DEVSEL#	PSTS	**Primary Device Select**: Asserted by the target indicating that the device is accepting the transaction. As a master, PCI 6150 waits for the assertion of this signal within 5 cycles of P_FRAME# assertion; otherwise, terminate with master abort. Before being three-stated, it is driven to a deasserted state for one cycle.
P_STOP#	PSTS	**Primary STOP**: Asserted by the target indicating that the target is requesting the initiator to stop the current transaction. Before being three-stated, it is driven to a deasserted state for one cycle.
P_LOCK#	PI	**Primary LOCK**: Asserted by master for multiple transactions to complete. If LOCK function is not needed, as in the case that no secondary PCI devices support LOCK, this input should be pulled “HIGH” and should not be connected to the PCI bus. This function can also be disabled by setting register 46h bit 13 to 0.
P_IDSEL	PI	**Primary ID Select**. Used as chip select line for Type-0 configuration access to PCI 6150 configuration space.
P_PERR#	PSTS	**Primary Parity Error**: Asserted when a data parity error is detected for data received on the primary interface. Before being three-stated, it is driven to a deasserted state for one cycle.
P_SERR#	OD	**Primary System Error**: Can be driven LOW by any device to indicate a system error condition. This signal should be pulled up through an external resistor.
P_REQ#	PTS	**Primary Request**: This is asserted by PCI 6150 to indicate that it wants to start a transaction on the primary bus. PCI 6150 deasserts this pin for at least 2 PCI clock cycles before asserting it again.
P_GNT#	PI	**Primary Grant**: When asserted, PCI 6150 can access the primary bus. During idle and P_GNT# asserted, PCI 6150 will drive P_AD, P_CBE and P_PAR to valid logic level.
P_M66EN	PI	**Primary 66 MHz Enable:** Set high for 66MHz primary bus. This signal, along with the S_M66EN signal, controls the frequency output to the SCLKOUT pins. See Chapter 15 for more details.

5.2 Secondary Bus Interface Signals

Name	Type	Description
S_AD[31:0]	PTS	**Secondary Address/Data**: Multiplexed address and data bus. Address is indicated by S_FRAME# assertion. Write data is stable and valid when S_IRDY# is asserted and read data is stable and valid when S_TRDY# is asserted. Data is transferred on rising clock edges when both S_IRDY# and S_TRDY# are asserted. During bus idle, PCI 6150 drives S_AD to a valid logic level when the S_GNT# is asserted.
S_CBE[3:0]	PTS	**Secondary Command/Byte Enables**: Multiplexed command field and byte enable field. During the address phase, the initiator drives the transaction type on these pins. After that the initiator drives the byte enables during data phases. During bus idle, PCI 6150 drives S_CBE[3:0] to a valid logic level when the internal grant is asserted.
S_PAR	PTS	**Secondary Parity**: Parity is even across S_AD[31:0], S_CBE[3:0], and S_PAR (i.e. an even number of '1's). S_PAR is an input and is valid and stable one cycle after the address phase (indicated by assertion of S_FRAME#) for address parity. For write data phases, S_PAR is an input and is valid one clock after S_IRDY# is asserted. For read data phase, S_PAR is an output and is valid one clock after S_TRDY# is asserted. Signal S_PAR is three-stated one cycle after the S_AD lines are three-stated. During bus idle, PCI 6150 drives S_PAR to a valid logic level when the internal grant is asserted.
S_FRAME#	PSTS	**Secondary FRAME**: Driven by the initiator of a transaction to indicate the beginning and duration of an access. The deassertion of S_FRAME# indicates the final data phase requested by the initiator. Before being three-stated, it is driven to a deasserted state for one cycle
S_IRDY#	PSTS	**Secondary IRDY**: Driven by the initiator of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle.
S_TRDY#	PSTS	**Secondary TRDY**: Driven by the target of a transaction to indicate its ability to complete the current data phase on the primary side. Once asserted in a data phase, it is not deasserted until end of the data phase. Before being three-stated, it is driven to a deasserted state for one cycle.
S_DEVSEL#	PSTS	**Secondary Device Select**: Asserted by the target indicating that the device is accepting the transaction. As a master, PCI 6150 waits for the assertion of this signal within 5 cycles of S_FRAME# assertion; otherwise, terminate with master abort. Before being three-stated, it is driven to a deasserted state for one cycle.
S_STOP#	PSTS	**Secondary STOP**: Asserted by the target indicating that the target is requesting the initiator to stop the current transaction. Before being three-stated, it is driven to a deasserted state for one cycle.
S_LOCK#	PSTS	Secondary LOCK: Asserted by master to complete multiple transactions.
S_PERR#	PSTS	**Secondary Parity Error**: Asserted when a data parity error is detected for data received on the primary interface. Before being three-stated, it is driven to a deasserted state for one cycle.
S_SERR#	PI	**Secondary System Error**: Can be driven LOW by any device to indicate a system error condition. This signal should be pulled up through an external resistor.

S_REQ#[0]	PI	**Secondary Request 0**: When External arbitration is not activated, this is asserted by external device to indicate that it wants to start a transaction on the Secondary bus. It must be externally pulled up through resistors to VDD. When External arbitration is active, this becomes the External Grant input to PCI 6150.
S_REQ#[8:1]	PI	**Secondary Requests**: This is asserted by external device to indicate that it wants to start a transaction on the Secondary bus. They must be externally pulled up through resistors to VDD.
S_GNT#[0]	PTS	**Secondary Grant 0**: This pin behaves like S_GNT#[8:1] under Transparent Mode and External arbitration is not activated. When External arbitration is active, this becomes the External Bus Request output from PCI 6150.
S_GNT#[8:1]	PTS	**Secondary Grants**: PCI 6150 asserts this pin to access the secondary bus. PCI 6150 deasserts this pin for at least 2 PCI clock cycles before asserting it again. During idle and S_GNT# asserted, PCI 6150 will drive S_AD, S_CBE and S_PAR to valid logic levels.
S_M66EN	PI/OD	**Secondary 66 MHz Enable:** Drive low if P_M66EN is low, otherwise driven from outside to select 66MHz or 33MHz. This signal, along with the P_M66EN signal, controls the frequency output to the S_CLKOUTn pins. This pin should be pulled High or Low externally. See Chapter 15 for more details

5.3 Clock Related Signals

Name	Type	Description
P_CLKIN	I	**Primary CLK input:** Provides timing for all transactions on primary interface.
S_CLKIN	I	**Secondary CLK input:** Provides timing for all transactions on secondary interface.
S_CLK[9:0]	PO	**Secondary CLK output:** Provides S_CLKIN or OSCIN (if enabled) phased shifted synchronous output clocks.
MSK_IN	I	**Secondary Clock Disable Serial Input.** This signal is used by the hardware mechanism to disable secondary clock outputs. The serial stream is received by MSK_IN, starting when P_RSTIN# is detected deasserted and S_RSTOUT# is detected asserted. This serial data is used for selectively disabling secondary clock outputs and is shifted into the secondary clock control configuration register. This input can be tied low to enable all secondary clock outputs. If tied high, the clocks will be active until high after reset. After the "1"s have been shifted in, the clocks will be driven high.
OSCSEL#	I	**External Oscillator Enable.** Enables connection of external clock for the secondary interface. If low, secondary bus clock outputs will use the clock signal from OSCIN input instead of P_CLKIN to generate S_CLK[9:0]. If HIGH, P_CLKIN is used. This pin must NOT be left unconnected.
OSCIN	I	**External Oscillator Input.** External clock input used to generate secondary output clocks when enabled through OSCSEL# pin.

5.4 Reset Signals

Name	Type	Description
P_RSTIN#	PI	**Primary Reset:** When P_RSTIN# is active, outputs are asynchronously three-stated and P_SERR# and P_GNT# floated. All primary port PCI standard configuration registers 0h-3Fh revert to their default state. **When asserted, all primary PCI signals are three-stated and no bus parking is asserted.**
S_RSTOUT#	PO	**Secondary Reset Output:** Asserted when any of the following conditions is met: 1. Signal P_RSTIN# is asserted. 2. The Secondary reset bit in the bridge control register (Register 3Eh) in configuration space is set.

5.5 Miscellaneous Signals

Name	Type	Description
S_CFN#	I	**Internal Arbiter Enable:** 0 = Use internal arbiter. 1 = Use external arbiter. S_REQ0# becomes External Arbiter GNT# input and S_GNT0# becomes REQ# output to External Arbiter.
CFG66	I	**Primary Config 66MHz:** The state of this pin is reflected in the Secondary status register at offset 1Eh. Other than this, the pin has no effect on PCI 6150 operation.
BPCC_EN	I	**Bus/Power Clock Control Management pin.** When signal is tied high and the PCI 6150 is placed in the D3hot power state, the PCI 6150 places the secondary bus in the B2 power state. The PCI 6150 disables the secondary clocks and drives them to 0. When tied low, placing the PCI 6150 in the D3hot power state has no effect on the secondary bus clocks.
GPIO[3:0]	IO-PU	**General Purpose Input Output pins.** These 4 general purpose signals are programmable as either input-only or bi-directional signals by writing the GPIO output enable control register. During PRST# asserted, GPIO[3:0] are used to shift in the clock disable serial data.
EEPCLK	O	**EEPROM Clock.** This pin is the clock signal to the EEPROM interface used during Autoload and for VPD functions.
EEPDATA	IO	**EEPROM Serial Data.** This pin is serial data interface to the EEPROM.
EE_EN#	I	**EEPROM Enable.** This input should be “0” to enable EEPROM use
PVIO	I	**Primary Interface I/O Voltage** This signal must be tied to either 3.3V or 5V, depending on the signaling voltage of the primary interface.
SVIO	I	**Secondary Interface I/O Voltage** This signal must be tied to either 3.3V or 5V, depending on the signaling voltage of the secondary interface.
Pin 151 Reserved		PCI 6150 does not use this pin.

5.6 Hot Swap Signals

In order to use Hot Swap function, both EJECT_EN# and GPIO3FN# must be connected to "0".

Name	Type	Description
ENUM#	O	**Hot Swap Interrupt:** An open drain bussed signal to signal a change in status for the chip. It is asserted through the Hot Swap registers.
LED	IO-PD	**Hot Swap LED:** Indicates the status of software connection process. If Hot Swap is NOT enabled (When either EJECT_EN# or GPIOFN# is "1"), this pin must be left unconnected or pulled to logic "0" via a 1K pull down resistor.
GPIO3FN#	I	GPIO[3] Function Select: **When GPIO3FN# is tied HIGH, GPIO[3] acts only as a GPIO pin regardless of the state of EJECT_EN#. GPIO[3] acts as Ejector input only when both GPIO3FN# and EJECT_EN# are tied LOW.**
EJECT_EN#	I-PU	**Eject Enable:** Pin used to enable GPIO[3] pin as EJECT input. If this pin is "1", GPIO[3] will function as a GPIO pin. GPIO[3] only acts as EJECT input when both GPIO3FN# and EJECT_EN# are tied LOW.

5.7 JTAG/Boundary Scan Interface Signals

Name	Type	Description
TCK	I-PU	**Test Clock:** Used to clock state information and test data into and out of PCI 6254 during operation of the TAP. This pin should be pulled high or pulled low to a known state using an external resistor.
TMS	I-PU	**Test Mode Select:** Used to control the state of the TAP controller in PCI 6254. This pin should be pulled high or pulled low to a known state using an external resistor.
TDO	O	**Test Data Output:** Used to serially shift test data and test instructions out of PCI 6254 during TAP operation.
TDI	I-PU	**Test Data Input:** Used to serially shift test data and test instructions into PCI 6254 during TAP operation. This pin should be pulled high or pulled low to a known state using an external resistor.
TRST#	I	**Test Reset:** It provides an asynchronous initialization of the TAP controller. This pin MUST be pulled high or pulled low to a known state using an external resistor. We recommend pulling low using a 330ohm resistor.

5.8 Power Signals

Name	Type	Description
VDD		+3.3V
GND		Ground

5.9 Pin Assignment Sorted By Location

PIN NO.	Signal Name	Type
1	VDD	P
2	S_REQ1#	I
3	S_REQ2#	I
4	S_REQ3#	I
5	S_REQ4#	I
6	S_REQ5#	I
7	S_REQ6#	I
8	S_REQ7#	I
9	S_REQ8#	I
10	S_GNT0#	TS
11	S_GNT1#	TS
12	VSS	P
13	S_GNT2#	TS
14	S_GNT3#	TS
15	S_GNT4#	TS
16	S_GNT5#	TS
17	S_GNT6#	TS
18	S_GNT7#	TS
19	S_GNT8#	TS
20	VSS	P
21	S_CLK	I
22	S_RST#	O
23	S_CFN#	I

PIN NO.	Signal Name	Type
36	S_CLKOUT5	O
37	VSS	P
38	S_CLKOUT6	O
39	S_CLKOUT7	O
40	VDD	P
41	S_CLKOUT8	O
42	S_CLKOUT9	O
43	P_RST#	I
44	BPCC_EN	I
45	P_CLK	I
46	P_GNT#	I
47	P_REQ#	TS
48	VSS	P
49	P_AD31	TS
50	P_AD30	TS
51	OSC_SEL#	I
52	VSS	P
53	VDD	P
54	OSC_IN	I
55	P_AD29	TS
56	VDD	P
57	P_AD28	TS
58	P_AD27	TS

PIN NO.	Signal Name	Type
71	P_AD20	TS
72	VSS	P
73	P_AD19	TS
74	P_AD18	TS
75	VDD	P
76	P_AD17	TS
77	P_AD16	TS
78	VSS	P
79	P_CBE2#	TS
80	P_FRAME#	STS
81	VDD	P
82	P_IRDY	STS
83	P_TRDY	STS
84	P_DEVSEL#	STS
85	P_STOP#	STS
86	VSS	P
87	P_LOCK#	I
88	P_PERR#	STS
89	P_SERR#	OD
90	P_PAR	TS
91	VDD	P
92	P_CBE1#	TS
93	P_AD15	TS

24	GPIO3	TS
25	GPIO2	TS
26	VDD	P
27	GPIO1	TS
28	GPIO0	TS
29	S_CLKOUT0	O
30	S_CLKOUT1	O
31	VSS	P
32	S_CLKOUT2	O
33	S_CLKOUT3	O
34	VDD	P
35	S_CLKOUT4	O

59	VSS	P
60	P_AD26	TS
61	P_AD25	TS
62	VDD	P
63	P_AD24	TS
64	P_CBE3#	TS
65	P_IDSEL	I
66	VSS	P
67	P_AD23	TS
68	P_AD22	TS
69	VDD	P
70	P_AD21	TS

94	VSS	P
95	P_AD14	TS
96	P_AD13	TS
97	VDD	P
98	P_AD12	TS
99	P_AD11	TS
100	VSS	P
101	P_AD10	TS
102	P_M66EN	I
103	EE_EN#	I
104	VSS	P
105	VDD	P

Signal Names Sorted By Pin Number (Continue)

PIN NO.	Signal Name	Type
106	EJECT_EN#	I
107	P_AD9	TS
108	VDD	P
109	P_AD8	TS
110	P_CBE0#	TS
111	VSS	P
112	P_AD7	TS
113	P_AD6	TS
114	VDD	P
115	P_AD5	TS
116	P_AD4	TS
117	VSS	P
118	P_AD3	TS
119	P_AD2	TS
120	VDD	P
121	P_AD1	TS
122	P_AD0	TS
123	VSS	P
124	P_VIO	I
125	CFG66	I
126	MSK_IN	I
127	PIN_ENUM#	O
128	PIN_LED	I/O
129	TDI	I

PIN NO.	Signal Name	Type
141	S_AD3	TS
142	VSS	P
143	S_AD4	TS
144	S_AD5	TS
145	VDD	P
146	S_AD6	TS
147	S_AD7	TS
148	VSS	P
149	S_CBE0#	TS
150	S_AD8	TS
151	Reserved	
152	S_AD9	TS
153	S_M66EN	OD
154	S_AD10	TS
155	GPIO3FN#	I
156	VSS	P
157	VDD	P
158	EEPCLK	O
159	S_AD11	TS
160	EEPD	I/O
161	S_AD12	TS
162	S_AD13	TS
163	VDD	P
164	S_AD14	TS

PIN NO.	Signal Name	Type
176	S_TRDY#	STS
177	S_IRDY#	STS
178	VDD	P
179	S_FRAME#	STS
180	S_CBE2#	TS
181	VSS	P
182	S_AD16	TS
183	S_AD17	TS
184	VDD	P
185	S_AD18	TS
186	S_AD19	TS
187	VSS	P
188	S_AD20	TS
189	S_AD21	TS
190	VDD	P
191	S_AD22	TS
192	S_AD23	TS
193	VSS	P
194	S_CBE3#	TS
195	S_AD24	TS
196	VDD	P
197	S_AD25	TS
198	S_AD26	TS
199	VSS	P

130	TDO	O
131	VDD	P
132	TMS	I
133	TCLK	I
134	TRST#	I
135	S_VIO	I
136	VSS	P
137	S_AD0	TS
138	S_AD1	TS
139	VDD	P
140	S_AD2	TS

165	S_AD15	TS
166	VSS	P
167	S_CBE1#	TS
168	S_PAR	TS
169	S_SERR#	I
170	VDD	P
171	S_PERR#	STS
172	S_LOCK#	STS
173	S_STOP#	STS
174	VSS	P
175	S_DEVSEL#	STS

200	S_AD27	TS
201	S_AD28	TS
202	VDD	P
203	S_AD29	TS
204	S_AD30	TS
205	VSS	P
206	S_AD31	TS
207	S_REQ0#	I
208	VDD	P

5.10 Pin Assignment Sorted By Pin Name

Signal Name	PIN NO.	Type
BPCC_EN	44	I
CFG66	125	I
EE_EN#	103	I
EEPCLK	158	O
EEPD	160	I/O
EJECT_EN#	106	I
GPIO0	28	TS
GPIO1	27	TS
GPIO2	25	TS
GPIO3	24	TS
GPIO3FN#	155	I
MSK	126	I
OSC_IN	54	I
OSC_SEL#	51	I
P_AD0	122	TS
P_AD1	121	TS
P_AD2	119	TS
P_AD3	118	TS
P_AD4	116	TS
P_AD5	115	TS
P_AD6	113	TS
P_AD7	112	TS
P_AD8	109	TS

Signal Name	PIN NO.	Type
P_AD21	70	TS
P_AD22	68	TS
P_AD23	67	TS
P_AD24	63	TS
P_AD25	61	TS
P_AD26	60	TS
P_AD27	58	TS
P_AD28	57	TS
P_AD29	55	TS
P_AD30	50	TS
P_AD31	49	TS
P_CBE0#	110	TS
P_CBE1#	92	TS
P_CBE2#	79	TS
P_CBE3#	64	TS
P_CLK	45	I
P_DEVSEL#	84	STS
P_FRAME#	80	STS
P_GNT#	46	I
P_IDSEL	65	I
P_IRDY#	82	STS
P_LOCK#	87	I
P_M66EN	102	I

Signal Name	PIN NO.	Type
S_AD1	138	TS
S_AD2	140	TS
S_AD3	141	TS
S_AD4	143	TS
S_AD5	144	TS
S_AD6	146	TS
S_AD7	147	TS
S_AD8	150	TS
S_AD9	152	TS
S_AD10	154	TS
S_AD11	159	TS
S_AD12	161	TS
S_AD13	162	TS
S_AD14	164	TS
S_AD15	165	TS
S_AD16	182	TS
S_AD17	183	TS
S_AD18	185	TS
S_AD19	186	TS
S_AD20	188	TS
S_AD21	189	TS
S_AD22	191	TS
S_AD23	192	TS

P_AD9	**107**	TS
P_AD10	**101**	TS
P_AD11	**99**	TS
P_AD12	**98**	TS
P_AD13	**96**	TS
P_AD14	**95**	TS
P_AD15	**93**	TS
P_AD16	**77**	TS
P_AD17	**76**	TS
P_AD18	**74**	TS
P_AD19	**73**	TS
P_AD20	**71**	TS

P_PAR	**90**	TS
P_PERR#	**88**	STS
P_REQ#	**47**	TS
P_RST#	**43**	I
P_SERR#	**89**	OD
P_STOP#	**85**	STS
P_TRDY#	**83**	STS
P_VIO	**124**	I
PIN_ENUM#	**127**	O
PIN_LED	**128**	I/O
Reserved	**151**	
S_AD0	**137**	TS

S_AD24	**195**	TS
S_AD25	**197**	TS
S_AD26	**198**	TS
S_AD27	**200**	TS
S_AD28	**201**	TS
S_AD29	**203**	TS
S_AD30	**204**	TS
S_AD31	**206**	TS
S_CBE0#	**149**	TS
S_CBE1#	**167**	TS
S_CBE2#	**180**	TS
S_CBE3#	**194**	TS

Signal Names Sorted Alphabetically (Continue)

Signal Name	PIN NO.	Type
S_CFN#	23	I
S_CLK	21	I
S_CLKOUT0	29	O
S_CLKOUT1	30	O
S_CLKOUT2	32	O
S_CLKOUT3	33	O
S_CLKOUT4	35	O
S_CLKOUT5	36	O
S_CLKOUT6	38	O
S_CLKOUT7	39	O
S_CLKOUT8	41	O
S_CLKOUT9	42	O
S_DEVSEL#	175	STS
S_FRAME#	179	STS
S_GNT0#	10	TS
S_GNT1#	11	TS
S_GNT2#	13	TS
S_GNT3#	14	TS
S_GNT4#	15	TS
S_GNT5#	16	TS
S_GNT6#	17	TS
S_GNT7#	18	TS
S_GNT8#	19	TS
S_IRDY#	177	STS

Signal Name	PIN NO.	Type
S_REQ7#	8	I
S_REQ8#	9	I
S_RST#	22	O
S_SERR#	169	I
S_STOP#	173	STS
S_TRDY#	176	STS
S_VIO	135	I
TCLK	133	I
TDI	129	I
TDO	130	O
TMS	132	I
TRST#	134	I
VDD	1	P
VDD	26	P
VDD	34	P
VDD	40	P
VDD	53	P
VDD	56	P
VDD	62	P
VDD	69	P
VDD	75	P
VDD	81	P
VDD	91	P
VDD	97	P

Signal Name	PIN NO.	Type
VDD	182	P
VDD	190	P
VDD	196	P
VDD	202	P
VDD	208	P
VSS	12	P
VSS	20	P
VSS	31	P
VSS	37	P
VSS	48	P
VSS	52	P
VSS	59	P
VSS	66	P
VSS	72	P
VSS	78	P
VSS	86	P
VSS	94	P
VSS	104	P
VSS	100	P
VSS	111	P
VSS	117	P
VSS	123	P
VSS	136	P
VSS	142	P

S_LOCK#	**172**	STS
S_M66EN	**153**	OD
S_PAR	**168**	TS
S_PERR#	**171**	STS
S_REQ0#	**207**	I
S_REQ1#	**2**	I
S_REQ2#	**3**	I
S_REQ3#	**4**	I
S_REQ4#	**5**	I
S_REQ5#	**6**	I
S_REQ6#	**7**	I

VDD	**105**	P
VDD	**108**	P
VDD	**114**	P
VDD	**120**	P
VDD	**131**	P
VDD	**139**	P
VDD	**145**	P
VDD	**157**	P
VDD	**163**	P
VDD	**170**	P
VDD	**178**	P

VSS	**148**	P
VSS	**156**	P
VSS	**166**	P
VSS	**174**	P
VSS	**181**	P
VSS	**187**	P
VSS	**193**	P
VSS	**199**	P
VSS	**205**	P

6 Configuration Registers

As a PCI bridge, PCI 6150 includes the standard Type-01h Configuration Space header defined in Bridge 1.1.

The following PCI registers are supported by PCI 6150.

6.1 Configuration Space Map

Superscript legend:
1 = Writable when Read Only Register Write Enable bit is set
2 = EEPROM loadable

Bits 31-24	Bits 23-16	Bits 15-8	Bits 7-0	Address
[1,2] Device ID		[1,2] Vendor ID		00h
Primary Status		Primary Command		04h
[1,2] Class Code			Revision ID	08h
[1,2] BIST	[1,2] Header Type	Primary Latency Timer	Cache Line Size	0Ch
Reserved				10h – 17h
Secondary Latency Timer	Subordinate Bus Number	Secondary Bus Number	Primary Bus Number	18h
Secondary Status		I/O Limit	I/O Base	1Ch
Memory Limit		Memory Base		20h
Prefetchable Memory Limit		Prefetchable Memory Base		24h
Prefetchable Memory Base Upper 32 Bits				28h
Prefetchable Memory Limit Upper 32 Bits				2Ch
I/O Limit Upper 16 Bits		I/O Base Upper 16 Bits		30h
Reserved			ECP Pointer	34h
Reserved				38h
Bridge Control		Interrupt Pin	Reserved	3Ch
Arbiter Control		Diagnostic Control	Chip Control	40h
[2] Misc Options		[2] Timeout Control	[2] Primary Flow Through Control	44h
[2] Secondary Incremental Prefetch Count	[2] Primary Incremental Prefetch Count	[2] Secondary Prefetch Line Count	[2] Primary Prefetch Line Count	48h
Reserved	[2] Secondary Flow Through Control	[2] Secondary Maximum Prefetch Count	[2] Primary Maximum Prefetch Count	4Ch
Reserved	Test register	Internal Arbiter Control		50h
EEPROM Data		EEPROM Address	EEPROM control	54h

Reserved				58h-60h
GPIO[3-0] Input Data	**GPIO[3-0] Output Enable Control**	**GPIO[3-0] Output Data**	**P_SERR# event disable**	64h
Reserved	**P_SERR# status**	**Clock Control**		68h
Reserved				6Ch-98h
Reserved	**Reserved**	**Reserved**	**ROR control**	9Ch
Reserved				A0-CCh
Reserved	Reserved	Reserved	Reserved	D0h
Reserved				D4h
Reserved				D8h
[1,2] Power Management Capabilities		**Next Item Ptr = E4**	**Capability ID = 01**	DCh
[1,2] Power Management Data	**PMCSR Bridge Support**	**[1, 2] Power Management CSR**		E0h
Reserved	**HSCSR = 00**	**Next Item Ptr = E8**	**Capability ID = 06**	E4h
VPD Register = 0000		**Next Item Ptr = 00**	**Capability ID = 03**	E8h
VPD Data Register = 0000_0000				ECh
Reserved				F0h-FCh

6.2 Transparent Mode Configuration Register Description

6.2.1 PCI Standard Configuration Registers

Vendor ID Register (Read Only) - Offset 0h
Defaults to 3388(h).

Device ID Register (Read Only) - Offset 2h
Defaults to 0022(h).

(Note: R/W - Read/Write, R/O - Read Only, R/WC - Read/ Write 1 to clear)

Primary Command Register (Read/Write) - Offset 4h

Bit	Function	Type	Description
0	I/O Space Enable	R/W	Controls the bridge's response to I/O accesses on the primary interface. 0=ignore I/O transaction 1=enable response to I/O transaction Reset to 0.
1	Memory Space Enable	R/W	Controls the bridge's response to memory accesses on the primary interface. 0=ignore all memory transaction 1=enable response to memory transaction Reset to 0.
2	Bus Master Enable	R/W	Controls the bridge's ability to operate as a master on the primary interface. 0=do not initiate transaction on the primary interface and disable response to memory or I/O transactions on secondary interface 1=enable the bridge to operate as a master on the primary interface Reset to 0.
3	Special Cycle Enable	R/O	No special cycle implementation (set to '0').
4	Memory Write and Invalidate Enable	R/O	Memory write and invalidate not supported (set to '0').
5	VGA Palette Snoop Enable	R/W	Controls the bridge's response to VGA compatible palette accesses. 0=ignore VGA palette accesses on the primary interface 1=enable response to VGA palette writes on the primary interface (I/O address AD[9:0]=3C6h, 3C8h and 3C9h) Reset to 0.
6	Parity Error Enable	R/W	Controls the bridge's response to parity errors. 0=ignore any parity errors 1=normal parity checking performed Reset to 0.
7	Wait Cycle Control	R/W	PCI 6150 performs address / data stepping (reset to '1').
8	P_SERR# Enable	R/W	Controls the enable for the P_SERR# pin. 0=disable the P_SERR# driver 1=enable the P_SERR# driver Reset to 0.
9	Fast Back to Back Enable	R/W	0=no fast back to back transaction 1=reserved. PCI 6150 does not generate Fast Back to Back cycle. Reset to 0.
10-15	Reserved	R/O	Reserved. Reset to 0.

Primary Status Register(Read/Write) – Offset 6h

Bit	Function	Type	Description
0-3	Reserved	R/O	Reserved (set to '0's).
4	ECP	R/O	Enhanced Capabilities port. Reads as 1 to indicate PCI 6150 supports an enhanced capabilities list.
5	66MHz	R/O	Reflects the state of CFG66 input pin. 1 = PCI 6150 is 66Mhz Capable.
6	UDF	R/O	No User-Definable Features (set to '0').
7	Fast Back to Back Capable	R/O	Fast back-to-back write capable on primary side (set to '1').
8	Data Parity Error Detected	R/WC	It is set when the following conditions are met: 1. P_PERR# is asserted 2. Bit 6 of Command Register is set Reset to 0.
9-10	DEVSEL timing	R/O	DEVSEL# timing. Reads as 01b to indicate PCI 6150 responds no slower than with medium timing
11	Signaled Target Abort	R/WC	Should be set (by a target device) whenever a Target Abort cycle occurs. Reset to 0.
12	Received Target Abort	R/WC	Set to '1' (by a master device) when transactions are terminated with Target Abort. Reset to 0.
13	Received Master Abort	R/WC	Set to '1' (by a master) when transactions are terminated with Master Abort. Reset to 0.
14	Signaled System Error	R/WC	Should be set whenever P_SERR# is asserted. Reset to 0.
15	Detected Parity Error	R/WC	Should be set whenever a parity error is detected regardless of the state of the bit 6 of command register. Reset to 0.

Revision ID Register (Read Only) – Offset 8h
Defaults to 04h.

Class Code Register (Read Only) – Offset 9h
Defaults to 060400h.

Cache Line Size Register (Read/Write) – Offset 0Ch
This register is used when terminating memory write and invalidate transactions. Memory read prefetching is controlled by the prefetch count registers.

Only cache line sizes (in units of 32-bits words) which are power of two are valid. Resets to 0.

Primary Latency Timer Register (Read/Write) – Offset 0Dh
This register sets the value for Master Latency Timer which starts counting when the master asserts FRAME#. Reset to 0.

Header Type Register (Read Only) – Offset 0Eh
Defaults to 1.

BIST Register (Read Only) – Offset 0Fh
This register can be written to by enabling the ROR Write Enable bit at register 9Ch bit 7. Reset to 0.

Primary Bus Number Register (Read/Write) – Offset 18h
Programmed with the number of the PCI bus to which the primary bridge interface is connected. This value is set with configuration software. Reset to 0.

Secondary Bus Number Register (Read/Write) – Offset 19h
Programmed with the number of the PCI bus to which the secondary bridge interface is connected. This value is set with configuration software. Reset to 0.

Subordinate Bus Number Register (Read/Write) –Offset 1Ah

Programmed with the number of the PCI bus with the highest number that is subordinate to the bridge. This value is set with configuration software. Reset to 0.

Secondary Latency Timer (Read/Write) – Offset 1Bh

This register is programmed in units of PCI bus clocks. Reset to 0. The latency timer checks for master accesses on the secondary side that remain unclaimed by any target.

I/O Base Register (Read/Write) – Offset 1Ch

This register defines the bottom address of the I/O address range for the bridge. The upper four bits define the bottom address range used by the chip to determine when to forward I/O transactions from one interface to the other. These 4 bits correspond to address bits <15:12> and are writeable. The upper 16 bits corresponding to address bits <31:16> are defined in the I/O base address upper 16 bits register. The address bits <11:0> are assumed to be 000h. The lower four bits (3:0) of this register set to '0001' (read-only) to indicate 32-bit I/O addressing. Reset to 0.

I/O Limit Register (Read/Write) – Offset 1Dh

This register defines the top address of the I/O address range for the bridge. The upper four bits define the top address range used by the chip to determine when to forward I/O transactions from one interface to the other. These 4 bits correspond to address bits <15:12> and are writeable. The upper 16 bits corresponding to address bits <31:16> are defined in the I/O limit address upper 16 bits register. The address bits <11:0> are assumed to be FFFh. The lower four bits (3:0) of this register set to '0001' (read-only) to indicate 32-bit I/O addressing. Reset to 0.

Secondary Status Register (Read/Write) – Offset 1Eh

Bit	Function	Type	Description
0-4	reserved	R/O	Reserved (set to '0's).
5	66MHz	R/O	Defaults to 1. PCI 6150 is 66Mhz Capable.
6	UDF	R/O	No User-Definable Features (set to '0').
7	Fast Back to Back Capable	R/O	Fast back-to-back write capable on secondary port (set to '1').
8	Data Parity Error Detected	R/WC	It is set when the following conditions are met: 1. SPERR# is asserted 2. Bit 6 of Command Register is set Reset to 0.
9-10	DEVSEL timing	R/O	Medium DEVSEL# timing (set to '01')
11	Signaled Target Abort	R/WC	Should be set (by a target device) whenever a Target Abort cycle occurs. Should be '0' after reset. Reset to 0.
12	Received Target Abort	R/WC	Set to '1' (by a master device) when transactions are terminated with Target Abort. Reset to 0.
13	Received Master Abort	R/WC	Set to '1' (by a master) when transactions are terminated with Master Abort. Reset to 0.
14	Received System Error	R/WC	Should be set whenever SSERR# is detected. Should be a '0 after reset. Reset to 0.
15	Detected Parity Error	R/WC	Should be set whenever a parity error is detected regardless of the state of the bit 6 of command register. Reset to 0.

Memory Base Register (Read/Write) – Offset 20h
This register defines the base address of the memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits corresponding to address bits <31:20> are writeable. The lower 20 address bits (19:0) are assumed to be 00000h. The 12 bits are reset to 0. The lower 4 bits are read only and set to 0.

Memory Limit Register (Read/Write) – Offset 22h
This register defines the upper limit address of the memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits corresponding to address bits <31:20> are writeable. The 12 bits are reset to 0. The lower 4 bits are read only and are set to 0. The lower 20 address bits (19:0) are assumed to be FFFFFh. Reset to 0.

Prefetchable Memory Base Register (Read/Write) - Offset 24h
This register defines the base address of the prefetchable memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits corresponding to address bits <31:20> are writeable. The 12 bits are reset to 0. The lower 4 bits are read only and are set to 0. The lower 20 address bits (19:0) are assumed to be 00000h. Reset to 0.

Prefetchable Memory Limit Register (Read/Write) – Offset 26h
This register defines the upper limit address of the memory-mapped address range for forwarding the cycle through the bridge. The upper twelve bits correspond to address bits <31:20> are writeable. The 12 bits are reset to 0. The lower 4 bits are read only and are set to 0. The lower 20 address bits (19:0) are assumed to be FFFFFh. Reset to 0.

Prefetchable Memory Base Register Upper 32 Bits (Read/Write) – Offset 28h
This register defines the upper 32 bit <63:32> memory base address of the prefetchable memory-mapped address for forwarding the cycle through the bridge. Reset to 0.

Prefetchable Memory Limit Register Upper 32 Bits (Read/Write) – Offset 2Ch
This register defines the upper 32 bit <63:32> memory limit address of the prefetchable memory-mapped address for forwarding the cycle through the bridge. Reset to 0.

I/O Base Address Upper 16 Bits Register (Read/Write) – Offset 30h
This register defines the upper 16 bits of a 32-bit base I/O address range used for forwarding the cycle through the bridge. Reset to 0.

I/O Limit Address Upper 16 Bits Register (Read/Write) – Offset 32h
This register defines the upper 16 bits of a 32-bit limit I/O address range used for forwarding the cycle through the bridge. Reset to 0.

ECP Pointer (Read/Only) – Offset 34h

Bit	Function	Type	Description
7-0	ECP Pointer	R/O	Enhanced capabilities port offset pointer. This register reads as DCh to indicate the offset of the power management registers.

Interrupt Pin Register (Read Only) – Offset 3Dh
Reads as 0 to indicate that PCI 6150 does not use any interrupt pin.

Bridge Control Register (Read/Write) – Offset 3Eh

Bit	Function	Type	Description
0	Parity Error Response Enable	R/W	Controls the bridge's response to parity errors on the secondary interface. 0=ignore address and data parity errors on the secondary interface 1=enable parity error reporting and detection on the secondary interface Reset to 0.
1	S_SERR# Enable	R/W	Controls the forwarding of S_SERR# to the primary interface. 0=disable the forwarding S_SERR# to primary 1=enable the forwarding of S_SERR# to primary Reset to 0.
2	ISA Enable	R/W	Controls the bridge's response to ISA I/O addresses, which is limited to the first 64K. 0=forward all I/O addresses in the range defined by the I/O Base and I/O Limit registers 1=block forwarding of ISA I/O addresses in the range defined by the I/O Base and I/O Limit registers that are in the first 64K of I/O space that address the last 768 bytes in each 1Kbytes block. Secondary I/O transactions are forwarded upstream if the address falls within the last 768 bytes in each 1Kbytes block There is an **ISA Enable Control bit Write Protect** mechanism control by EEPROM. When the **ISA Enable Control bit Write Protect** bit is set in the EEPROM and EEPROM initialization is enabled, PCI 6150 will change this bit to read only and ISA Enable feature will not be available. Reset to 0.
3	VGA Enable	R/W	Controls the bridge's response to VGA compatible addresses. 0=do not forward VGA compatible memory and I/O addresses from primary to secondary 1=forward VGA compatible memory and I/O address from primary to secondary regardless of other settings Reset to 0.
4	reserved	R/O	Reserved (set to 0).
5	Master Abort Mode	R/W	Controls the bridge behavior in responding to master aborts on secondary interface 0=do not report master aborts (return ffff_ffffh on reads and discards data on writes) 1=report master aborts by signaling target abort Reset to 0. Note: During lock cycles, PCI 6150 ignores this bit, and always completes the cycle as a target abort.

6	Secondary Reset	R/W	Forces the assertion of S_RSTOUT# signal pin on the secondary interface. 0=do not force the assertion of S_RSTOUT# pin 1=force the assertion of S_RSTOUT# pin Reset to 0.
7	Fast Back to Back Enable	R/W	0 = no fast back to back transaction 1= reserved. PCI 6150 does not generate Fast Back to Back cycle. Reset to 0.
8	Primary Master Timeout	R/W	Sets the maximum number of PCI clock for an initiator on the primary bus to repeat the delayed transaction request. 0=Timeout after 2^{15} PCI clocks 1=Timeout after 2^{10} PCI clocks Reset to 0.
9	Secondary Master Timeout	R/W	Sets the maximum number of PCI clock for an initiator on the secondary bus to repeat the delayed transaction request. 0=Timeout after 2^{15} PCI clocks 1=Timeout after 2^{10} PCI clocks Reset to 0.
10	Master Timeout Status	R/WC	Set to '1' when either primary master timeout or secondary master timeout. Reset to 0.
11	Master Timeout P_SERR# enable	R/W	Enable P_SERR# assertion during master timeout. 0=P_SERR# not asserted on master timeout 1=P_SERR# asserted on either primary or secondary master timeout. Reset to 0.
15-12	reserved	R/O	Reserved

Chip Control Register (Read/Write) – Offset 40h

Bit	Function	Type	Description
0	Reserved	R/O	
1	Memory write disconnect control	R/W	Controls when the chip as a target disconnects memory transactions. When 0, disconnects on queue full or on a 4KB boundary. When 1, disconnects on a cache line boundary, as well as when the queue fills or on a 4 KB boundary. Reset value is 0.
2-3	Reserved	R/W	Default to 0 and must not be changed
4	Secondary bus prefetch disable	R/W	Controls PCI 6150's ability to prefetch during upstream memory read transactions. When 0 the chip prefetches and does not forward byte enable bits during memory read transactions. When 1, PCI 6150 requests only one Dword from the target during memory read transactions and forwards read enable bits. PCI 6150 returns a target disconnect to the requesting master on the first data transfer. Memory read line and memory read multiple transactions are still prefetchable. Reset to 0.
7:5	Reserved	R/O	Reserved **(Set to 0)**

Diagnostic Control Register (Read/Write) – Offset 41h

Bit	Function	Type	Description
0	Chip reset	R/W	Chip and Secondary bus reset. Setting this bit will do a chip reset without asserting S_RSTOUT# and forces secondary reset bit in bridge control register to be set. After resetting all bits except for the secondary reset bit in bridge control register, this bit will be cleared. Write 0 has no effect.
2:1	Test mode	R/W	Reserved
3	Secondary Reset output mask	R/W	1 = Primary reset input P_RSTIN# active will not cause secondary reset output S_RSTOUT# to become active.
7:4	Reserved	R/O	Reserved (Set to 0).

Arbiter Control Register (Read/Write) – Offset 42h

Bit	Function	Type	Description
8-0	Arbiter Control	R/W	Each bit controls whether a secondary bus master is assigned to the high priority group or the low priority group. Bits <8:0> correspond to request inputs S_REQ#[8:0], respectively. Reset value is 0.
9	PCI 6150 priority	R/W	Defines whether the secondary port of PCI 6150 is in high priority group or the low priority group 0=low priority group **1=high priority group**. Reset to 1.
15:10	reserved	R/O	Reserved **(set to '0's)**

Primary Flow Through Control Register - Offset 44h

Bit	Function	Type	Description
2-0	Primary posted write completion wait count	R/W	Maximum number of clocks that PCI 6150 will wait for posted write data from initiator if delivering write data in flow through mode and internal post write queues are almost empty. If the count is exceeded without any additional data from the initiator, the cycle to target will be terminated to be completed later. 000 : PCI 6150 will terminate cycle if there is only 1 data entry left in the internal write queue. 001 : PCI 6150 will deassert IRDY#, and wait 1 clock for data before terminating cycle. … 111 : PCI 6150 will wait 7 clocks for source data.
3	Reserved	R/O	Reserved. Returns 0 when read
6-4	Primary delayed read completion wait count		Maximum number of clocks that PCI 6150 will wait for delayed read data from target if returning read data in flow through mode and internal delayed read queue is almost full. If the count is exceeded without any additional space in the queue, the cycle to target will be terminated, and completed when initiator retries the rest of the cycle. 000 : PCI 6150 will terminate cycle if only 1 data entry is left in the read queue. 001 : PCI 6150 will deassert TRDY#, and wait 1 clock for data before terminating cycle. … 111 : PCI 6150 will wait 7 clocks for source data.
7	Reserved	R/O	Reserved. Returns 0 when read.

Timeout Control Register - Offset 45h

Bit	Function	Type	Description
2-0	Maximum retry counter control	R/W	Controls maximum number of times that PCI 6150 will retry a cycle before signaling a timeout. This timeout applies to Read/Write retries and can be enabled to trigger SERR# on the primary or secondary port depending the SERR# events that are enabled. Maximum number of retries to timeout = 0000 : 2^{24} 0001 : 2^{18} 0010 : 2^{12} 0011 : 2^{6} 0111 : 2^{0} Reset to 0.
3	Reserved	R/O	
5:4	Primary master timeout divider	R/W	Provides an additional option for the primary master timeout. Timeout counter can optionally be divided by 256, in addition to its original setting in the Bridge Control register. Original setting is 32K by default and programmable to 1K. 11 : timeout counter = Primary master timeout / 256 10 : timeout counter = Primary master timeout / 16 01 : timeout counter = Primary master timeout / 8 00: counter = Primary master timeout / 1 defaults to 0
7:6	Secondary master timeout divider	R/W	Provides an additional option for the secondary master timeout. Timeout counter can optionally be divided by 256, in addition to its original setting in the Bridge Control register. Original setting is 32K by default and programmable to 1K. 11 : timeout counter = Primary master timeout / 256 10 : timeout counter = Primary master timeout / 16 01 : timeout counter = Primary master timeout / 8 00: counter = Primary master timeout / 1 defaults to 0

Miscellaneous Options - Offset 46h

Bit	Function	Type	Description
0	Write completion wait for PERR#	R/W	If 1, PCI 6150 will always wait for PERR# status of the target before completing a delayed write transaction to the initiator. Defaults to 0
1	Read completion wait for PAR	R/W	If 1, PCI 6150 will always wait for PAR status of the target before completing a delayed read transaction to the initiator. Defaults to 0
2	DTR out of order enable	R/W	If 1, PCI 6150 may return delayed read transactions in a different order than requested. Otherwise, delayed read transactions are returned in the same order as requested Defaults to 0
3	Generate parity enable	R/W	If 1, PCI 6150 as a master will generate the PAR to cycles going across the bridge, otherwise, PCI 6150 passes along the PAR of the cycle as stored in the internal buffers. Defaults to 0
6-4	Address step control	R/W	During configuration type 0 cycles, PCI 6150 will drive the address for the number of clocks specified in this register before asserting FRAME#. 000 : PCI 6150 will assert FRAME# at the same time as the address. 001 : PCI 6150 will assert FRAME# 1 clock after it drives the address. … 111 : PCI 6150 will assert FRAME# 7 clocks after it drives the address.
8-7	Reserved	R/W	Defaults to 0
9	Prefetch early termination	R/W	If 1, PCI 6150 will terminate prefetching at the current calculated count if flow through is not yet achieved, and another prefetchable read cycle is accepted by the PCI 6150. If 0, PCI 6150 will always finish prefetching as programmed at the prefetch count registers, regardless of any other outstanding prefetchable reads in the transaction queue.
10	Read minimum enable	R/W	If 1, PCI 6150 will only initiate read cycles if there is available space in the FIFO as specified by the prefetch count registers.
11	Reserved	R/O	Reserved at 0

12	Memory write and invalidate control	R/W	If 1, PCI 6150 will pass memory write and invalidate commands if there is at least 1 cache line of FIFO space available, otherwise it will complete as a memory write cycle. If 0, PCI 6150 will retry memory write and invalidate commands if there is no space for 1 cache line of data in the internal queues. Defaults to 0
13	Primary Lock Enable	R/W	If 1, PCI 6150 will follow the LOCK protocol on the primary interface. Otherwise, LOCK is ignored. Defaults to 1
14	Secondary Lock Enable	R/W	If 1, PCI 6150 will follow the LOCK protocol on the secondary interface. Otherwise, LOCK is ignored. Defaults to 0
15	Reserved	R/O	Reserved at 0

6.2.2 Prefetch Control Registers

Registers 44h, 48h – 4Dh are the prefetch control registers, and are used to fine-tune memory read prefetch behavior of the PCI 6150. Detailed descriptions of these registers can be found in Chapter 18 Flow Through Optimization.

Primary Initial Prefetch Count - Offset 48h

Bit	Function	Type	Description
5-0	Primary initial prefetch count	R/W	Controls initial prefetch count on the Primary bus during reads to prefetchable memory space. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0. Defaults to 10h
7-6	Reserved	R/O	Reserved. Returns 0 when read

Secondary Initial Prefetch Count - Offset 49h

Bit	Function	Type	Description
5-0	Secondary initial prefetch count	R/W	Controls initial prefetch count on the Secondary bus during reads to prefetchable memory space. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0. Defaults to 10h
7-6	Reserved	R/O	Reserved. Returns 0 when read

Primary Incremental Prefetch Count - Offset 4Ah

Bit	Function	Type	Description
5-0	Primary incremental prefetch count	R/W	This controls incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge will prefetch an additional "Primary incremental prefetch count" Dwords. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0. This register value must not exceed half the value programmed in the Primary Maximum Prefetch Count register. Otherwise, no incremental prefetch will be performed. Defaults to 10h
7-6	Reserved	R/O	Reserved. Returns 0 when read

Secondary Incremental Prefetch Count - Offset 4Bh

Bit	Function	Type	Description
5-0	Secondary incremental prefetch count	R/W	This controls incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge will prefetch an additional "Secondary incremental prefetch count" Dwords. This register value should be a power of 2 (only one bit should be set to 1 at any time). Value is number of double words. Bit 0 is read only and is always 0. This register value must not exceed half the value programmed in the Secondary Maximum Prefetch Count register. Otherwise, no incremental prefetch will be performed. Defaults to 10h
7-6	Reserved	R/O	Reserved. Returns 0 when read

Primary Maximum Prefetch Count - Offset 4Ch

Bit	Function	Type	Description
5-0	Primary maximum prefetch count	R/W	This value limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the primary when flow through for that entry was not achieved. This register value should be an even number. Bit 0 is read only and is always 0. Exception: 0h = 256 bytes = maximum programmable count Defaults to 20h
7-6	Reserved	R/O	Reserved. Returns 0 when read

Secondary Maximum Prefetch Count - Offset 4Dh

Bit	Function	Type	Description
5-0	Secondary maximum prefetch count	R/W	Register limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the secondary when flow through for that entry was not achieved. This register value should be an even number. Bit 0 is read only and is always 0. Exception: 0h = 256 bytes = maximum programmable count Defaults to 20h
7-6	Reserved	R/O	Reserved. Returns 0 when read

Secondary Flow Through Control Register - Offset 4Eh

Bit	Function	Type	Description
2-0	Secondary posted write completion wait count	R/W	Maximum number of clocks that PCI 6150 will wait for posted write data from initiator if delivering write data in flow through mode and internal post write queues are almost empty. If the count is exceeded without any additional data from the initiator, the cycle to target will be terminated, to be completed later. 000 : PCI 6150 will terminate cycle if there is only 1 data entry left in the internal write queue. 001 : PCI 6150 will deassert IRDY#, and wait 1 clock for data before terminating cycle. … 111 : PCI 6150 will wait 7 clocks for source data.
3	Reserved	R/O	Reserved. Returns 0 when read
6-4	Secondary delayed read completion wait count		Maximum number of clocks that PCI 6150 will wait for delayed read data from target if returning read data in flow through mode and internal delayed read queue is almost full. If the count is exceeded without any additional space in the queue, the cycle to target will be terminated, and completed when initiator retries the rest of the cycle. 000 : PCI 6150 will terminate cycle if only 1 data entry is left in the read queue. 001 : PCI 6150 will deassert TRDY#, and wait 1 clock for data before terminating cycle. … 111 : PCI 6150 will wait 7 clocks for source data.
7	Reserved	R/O	Reserved. Returns 0 when read.

Internal Arbiter Control Register - Offset 50h

Bit	Function	Type	Description
0	Low priority group fixed arbitration	R/W	If 1, the low priority group uses the fixed priority arbitration scheme, otherwise a rotating priority arbitration scheme is used Defaults to 0
1	Low priority group arbitration order	R/W	This bit is only valid when the low priority arbitration group is set to a fixed arbitration scheme. If 1, priority decreases in ascending numbers of the master, for example master #4 will have higher priority than master #3. If 0, the reverse is true. This order is relative to the master with the highest priority for this group, as specified in bits 7-4 of this register. Defaults to 0
2	High priority group fixed arbitration	R/W	If 1, the high priority group uses the fixed priority arbitration scheme, otherwise a rotating priority arbitration scheme is used Defaults to 0
3	High priority group arbitration order	R/W	This bit is only valid when the high priority arbitration group is set to a fixed arbitration scheme. If 1, priority decreases in ascending numbers of the master, for example master #4 will have higher priority than master #3. If 0, the reverse is true. This order is relative to the master with the highest priority for this group, as specified in bits 11-8 of this register. Defaults to 0
7-4	Highest priority master in low priority group	R/W	Controls which master in the low priority group has the highest priority. It is valid only if the group uses the fixed arbitration scheme. 0000 : master#0 has highest priority 0001 : … 1001 : PCI 6150 has highest priority 1010-1111 : Reserved Defaults to 0
11-8	Highest priority master in high priority group	R/W	Controls which master in the high priority group has the highest priority. It is valid only if the group uses the fixed arbitration scheme. 0000 : master#0 has highest priority 0001 : … 1001 : PCI 6150 has highest priority 1010-1111 : Reserved Defaults to 0

12-15	Bus Parking Control	R/W	Controls bus grant behavior during idle. 0000 : Last master granted is parked 0001 : Master #0 is parked … 1001 : Master #8 is parked 1010 : PCI 6150 is parked other : grant is deasserted Defaults to 0

PCI 6150 Test Register – Offset 52h

Bit	Function	Type	Description
0	EEPROM Autoload control	R/W	If 1, disables EEPROM autoload
1	Fast EEPROM Autoload	R/W	If 1, speeds up EEPROM autoload by 32 times.
2	EEPROM autoload status	R/O	Status of EEPROM autoload
3-7	Reserved	R/O	Reserved

EEPROM Control - Offset 54h

Bit	Function	Type	Description
0	Start	R/W	Starts the EEPROM read or write cycle.
1	EEPROM command	R/W	Controls the command sent to the EEPROM 1 : write 0 : read
2	EEPROM Error	R/O	This bit is set to 1 if EEPROM ACK was not received during EEPROM cycle.
3	EEPROM autoload successful	R/O	This bit is set to 1 if EEPROM autoload occurred successfully after reset, and some configuration registers were loaded with values programmed in the EEPROM. If zero, EEPROM autoload was unsuccessful or was disabled.
5-4	Reserved	R/O	Reserved. Returns '0' when read.
7-6	EEPROM clock rate	R/W	Controls frequency of EEPROM clock. EEPROM clock is derived from the primary PCI clock. 00 = PCLK/1024 (Used for 66Mhz PCI) 01 = PCLK/512 10 = PCLK/256 11 = PCLK/32 (for test mode use) defaults to 00

EEPROM Address - Offset 55h

Bit	Function	Type	Description
0	Reserved	R/O	Starts the EEPROM read or write cycle.
7-1	EEPROM address	R/W	Word address for EEPROM cycle.

EEPROM Control - Offset 56h

Bit	Function	Type	Description
15-0	EEPROM Data	R/W	Contains data to be written to the EEPROM. During reads, this register contains data received from the EEPROM after a read cycle has completed.

P_SERR# Event Disable Register - Offset 64h

Bit	Function	Type	Description
0	Reserved	R/O	Reserved. Returns 0 when read
1	Posted write parity error	R/W	Controls ability of PCI 6150 to assert P_SERR# when a data parity error is detected on the target bus during a posted write transaction. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0.
2	Posted Memory write nondelivery	R/W	Controls ability of PCI 6150 to assert P_SERR# when it is unable to deliver posted write data after 2^{24} (or programmed Maximum Retry count at Timeout Control Register) attempts. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0.
3	Target abort during posted write	R/W	Controls ability of PCI 6150 to assert P_SERR# when it receives a target abort when attempting to deliver posted write data. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0.
4	Master abort on posted write	R/W	Controls ability of PCI 6150 to assert P_SERR# when it receives a master abort when attempting to deliver posted write data. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0.
5	Delayed Configuration or IO write nondelivery	R/W	Controls ability of PCI 6150 to assert P_SERR# when it is unable to deliver delayed write data after 2^{24} (or programmed Maximum Retry count at Timeout Control Register) attempts. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0.
6	Delayed read-no data from target	R/W	Controls ability of PCI 6150 to assert P_SERR# when it is unable to transfer any read data from the target after 2^{24} (or programmed Maximum Retry count at Timeout Control Register) attempts. P_SERR# is asserted if this event occurs when this bit is 0 and SERR# enable bit in the command register is set. Reset value is 0.
7	Reserved	R/O	Reserved. Returns 0 when read.

GPIO[3:0] Output Data Register - Offset 65h

Bit	Function	Type	Description
3:0	GPIO[3:0] output write 1 to clear	R/W1 TC	Writing 1 to any of these bits drives the corresponding bit low on the GPIO[3:0] bus if it is programmed as output. Writing 0 has no effect. Read returns the last written value. Resets to 0.
7:4	GPIO[3:0] output write 1 to set	R/W1 TC	Writing 1 to any of these bits drives the corresponding bit high on the GPIO[3:0] bus if it is programmed as output. Writing 0 has no effect. Read returns the last written value. Resets to 0.

GPIO[3:0] Output Enable Register - Offset 66h

Bit	Function	Type	Description
3:0	GPIO output enable write 1 to clear	R/W1 TC	Writing 1 to any of these bits drives the corresponding bit on the GPIO[3:0] bus as input only. Writing 0 has no effect. Read returns the last value written. Resets to 0.
7:4	GPIO output enable write 1 to set	R/W1 TC	Writing 1 to any of these bits drives the corresponding bit on the GPIO[3:0] bus as output. GPIO[3:0] then drives the value set in the output data register (reg 65h). Writing 0 has no effect. Read returns the last written value. Resets to 0.

GPIO[3:0] Input Data Register - Offset 67h

Bit	Function	Type	Description
3:0	Reserved	R/O	Reserved
7:4	GPIO[3:0] input data	R/O	This read-only register reads the state of the GPIO[3:0] pins. The state is updated on the PCI clock cycle following a change in the GPIO[3:0] state.

Clock Control Register (Read/Write) – Offset 68h

Bit	Function	Type	Description
1:0	Clock 0 Disable	R/W	If either bit is 0, S_CLKOUT[0] is enabled. When both bits are 1, S_CLKOUT[0] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 0.
3:2	Clock 1 Disable	R/W	If either bit is 0, S_CLKO[1] is enabled. When both bits are 1, S_CLKO[1] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 1.
5:4	Clock 2 Disable	R/W	If either bit is 0, S_CLKO[2] is enabled. When both bits are 1, S_CLKO[2] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 2.
7:6	Clock 3 Disable	R/W	If either bit is 0, S_CLKO[3] is enabled. When both bits are 1, S_CLKO[3] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream. These bits are assigned to correspond to the PRSNT# pins for slot 3.
8	Clock 4 Disable	R/W	If 0, S_CLKO[4] is enabled. When 1, S_CLKO[4] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream.
9	Clock 5 Disable	R/W	If 0, S_CLKO[5] is enabled. When 1, S_CLKO[5] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream.
10	Clock 6 Disable	R/W	If 0, S_CLKO[6] is enabled. When 1, S_CLKO[6] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream.
11	Clock 7 Disable	R/W	If 0, S_CLKO[7] is enabled. When 1, S_CLKO[7] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream.

12	Clock 8 Disable	R/W	If 0, S_CLKO[8] is enabled. When 1, S_CLKO[8] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream.
13	Clock 9 Disable	R/W	If 0, S_CLKO[9] is enabled. When 1, S_CLKO[9] is disabled. Upon secondary bus reset, this bit is initialized by shifting in a serial data stream.
15-14	Reserved	R/O	Reserved

P_SERR# Status Register (Read/Write) – Offset 6Ah

Bit	Function	Type	Description
0	Address Parity error	R/WC	Signal P_SERR# was asserted due to address parity error on either side of the bridge. Reset to 0.
1	Posted Write Data Parity error	R/WC	Signal P_SERR# was asserted due to a posted write data parity error on the target bus. Reset to 0.
2	Post Write nondelivery	R/WC	Signal P_SERR# was asserted because PCI 6150 was unable to deliver posted write data to the target before timeout counter expires. Reset to 0.
3	Target abort during posted write	R/WC	Signal P_SERR# was asserted because PCI 6150 received a target abort when delivering posted write data. Reset to 0.
4	Master abort during posted write	R/WC	Signal P_SERR# was asserted because PCI 6150 received a master abort when delivering posted write data. Reset to 0.
5	Delayed write nondelivery	R/WC	Signal P_SERR# was asserted because PCI 6150 was unable to deliver delayed write data before time-out counter expires. Reset to 0.
6	Delayed read failed	R/WC	Signal P_SERR# was asserted because PCI 6150 was unable to read any data from the target before time-out counter expires. Reset to 0.
7	Delayed transaction master timeout	R/WC	Signal P_SERR# was asserted because a master did not repeat a read or write transaction before the master timeout counter expired on the initiator's bus. Reset to 0.

ROR Control (R/W) – Offset 9Ch

Bit	Function	Type	Description
6-0	**Reserved**	R/O	**Reserved**
7	**ROR Write Enable**	R/W	**Read Only Registers Write Enable:** Subsystem Vender ID at Register 2Ch and Subsystem ID Register at 2Eh are normally Read Only. Setting this bit to 1 will enable write to such Read Only ID Registers. Power Management Registers DEh, E0h, and E3h are normally Read Only. Setting this bit to 1 will enable write to all Read Only Power Management Registers. This bit must be cleared after the desired values have been modified in the Read Only Registers.

6.2.3 Hot Swap and Power Management Registers

Power Management Registers DEh, E0h, and E3h are normally Read Only. However their default value can be changed by firmware or software by setting the Read Only Registers Write Enable bit at Register. After any modifications to such registers, this Write Enable bit must be cleared to preserve their Read Only nature.

Capability Identifier (R/O) – Offset DCh

This register is set to 01h to indicate power management interface registers.

Next Item Pointer (R/O) – Offset DDh

Set to E4h. This field provides an offset into the function's PCI Configuration Space pointing to the location of next item in the function's capability list. In PCI 6150, this points to the hot swap registers.

Power Management Capabilities(R/O) – Offset DEh

This register is EEPROM or ROR Write controlled loadable, but is READ ONLY during normal operation.

Bit	Function	Type	Description
0-2	Version	R/O	This register is set to 001b, indicating that this function complies with Rev 1.0 of the PCI Power Management Interface Specification
3	PME Clock	R/O	This bit is a '0', indicating that PCI 6150 does not support PME# signaling.
4	Auxiliary Power Source	R/O	This bit is set to ‘0’ since PCI 6150 does not support PME# signaling
5	DSI	R/O	Device Specific Initialization . Returns ‘0’ indicating that PCI 6150 does not need special initialization
6-8	Reserved	R/O	Reserved
9	D1 Support	R/O	Returns ‘1’ indicating that PCI 6150 supports the D1 device power state
10	D2 Support	R/O	Returns ‘1’ indicating that PCI 6150 supports the D2 device power state
11-15	PME Support	R/O	Set to “0601” in Revision BA. Set to “7E01” in Revision BB.

Power Management Control/ Status(R/W) – Offset E0h

This register is EEPROM or ROR Write controlled loadable, but is READ ONLY during normal operation.

Bit	Function	Type	Description
0-1	Power State	R/W	This 2-bit field is used both to determine the current power state of a function and to set the function into a new power state. The definition of the field values is given below. 00b - D0 01b - D1 10b - D2 11b – D3hot
2-7	Reserved	R/O	Reserved
8	PME Enable	R/W	This bit is set to '0' since PCI 6150 does not support PME# signaling
9-12	Data Select	R/O	This field returns '0000b' indicating PCI 6150 does not return any dynamic data
13-14	Data Scale	R/O	Returns '00b' when read. PCI 6150 does not return any dynamic data.
15	PME Status	R/W	This bit is set to '0' since PCI 6150 does not support PME# signaling

PMCSR Bridge Support(R/W) – Offset E2h

Bit	Function	Type	Description
0-5	Reserved	R/O	Reserved
6	B2/B3 Support for D3hot	R/O	This bit reflects the state of the BPCC input pin. A '1' indicates that when PCI 6150 is programmed to D3hot state the secondary bus's clock is stopped.
7	Bus Power Control Enable	R/O	This bit reflects the state of the BPCC input pin. A '1' indicates that the power management state of the secondary bus follows that of PCI 6150 with one exception , D3hot state.

Power Management Data Register (RO) – Offset E3h

This register is EEPROM or ROR Write controlled loadable, but is READ ONLY during normal operation.

Capability Identifier (R/O) – Offset E4h

This register is **set to 06h** to indicate Hot Swap interface registers.

Next Item Pointer (R/O) - Offset E5h

Set to E8h. This field provides an offset into the function's PCI Configuration Space pointing to the location of next item in the function's capability list. In PCI 6150, this points to the Vital Product Data (VPD) registers.

Hot Swap Register(R/W) – Offset E6h

Bit	Function	Type	Description
0	DHA	R/W	Device Hiding Arm. Reset to 0. 1 = Arm Device Hiding 0 = Disarm Device Hiding DHA is set to 1 by hardware during hot swap port PCI RSTIN# going inactive and handle switch is still unlocked. The locking of the handle will clear this bit.
1	EIM ENUM# Mask Status	R/W	Enables or disables ENUM# assertion. Reset to 0. 0 = enable ENUM# signal 1 = mask off ENUM# signal
2	PIE	R/O	Pending INSert or EXTract: This bit is set when either INS or EXT is "1" or INS is armed (Write 1 to EXT bit). 1 = either an insertion or an extraction is in progress. 0 = Neither is pending
3	LOO LED status	R/W	Indicates if LED is on or off. Reset to 0. 0 = LED is off 1 = LED is on
5-4	PI	R/W	Programming Interface: Hardcode at 01: INS, EST, LOO, EIM and PIE, Device Hiding are supported.
6	EXT Extraction State	R/W1C	This bit is set by hardware when the ejector handle is unlocked and INS = 0.
7	INS Insertion State	R/W1C	This bit is set by hardware when hot swap port RSTIN# is deasserted, EEPROM autoload is completed and the ejector handle is locked. Writing 1 to EXT bit also arms INS.
15:8	Reserved	R/O	Reserved and a read returns all 0. Write has no effect.

6.2.4 VPD Registers

Capability Identifier (R/O) - Offset E8h

This register is set to 03h to indicate VPD registers.

Next Item Pointer (R/O) - Offset E9h

Set to 00h.

VPD Register (R/W) – Offset EAh

Bit	Function	Type	Description
1-0	Reserved	R/O	Reserved
7-2	VPD Address	R/W	**VPD operation**: Writing a '0' to this bit generates a read cycle from the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '0' value until EEPROM cycle is finished, then it be set to '1'. Data for reads is available at register ECh Writing a '1' to this bit generates a write cycle to the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '1' value until EEPROM cycle is finished, then it be cleared to '0'.
14-8	Reserved	R/O	Reserved
15	VPD Operation	R/W	**VPD operation**: Writing a '0' to this bit generates a read cycle from the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '0' value until EEPROM cycle is finished, then it be set to '1'. Data for reads is available at register ECh Writing a '1' to this bit generates a write cycle to the EEPROM at the VPD address specified in bits 7-2 of this register. This bit will remain at a logic '1' value until EEPROM cycle is finished, then it be cleared to '0'.

VPD Data Register (R/W) – Offset ECh

Bit	Function	Type	Description
31-0	VPD Data	R/W	**VPD Data** (EEPROM data[addr + 0x40]) - The least significant byte of this register corresponds to the byte of VPD at the address specified by the VPD Address register. The data read from or written to this register uses the normal PCI byte transfer capabilities.

7 PCI Bus Operation

This chapter presents detailed information about PCI transactions PCI 6150 responds to and PCI transactions initiated by PCI 6150.

7.1 PCI Transactions

Table 7–1 lists the command code and name of each PCI transaction that PCI 6150 initiates and responds to. The Master and Target columns indicate support for each transaction when PCI 6150 initiates transactions as a master, on the primary bus and on the secondary bus, and when PCI 6150 responds to transactions as a target, on the primary bus and on the secondary bus.

Table 7–1, PCI Transactions

Type of transaction		Initiates as Master		Responds as Target	
		Primary	Secondary	Primary	Secondary
0000	Interrupt acknowledge	N	N	N	N
0001	Special cycle	Y	Y	N	N
0010	I/O read	Y	Y	Y	Y
0011	I/O write	Y	Y	Y	Y
0100	Reserved	N	N	N	N
0101	Reserved	N	N	N	N
0110	Memory read	Y	Y	Y	Y
0111	Memory write	Y	Y	Y	Y
1000	Reserved	N	N	N	N
1001	Reserved	N	N	N	N
1010	Configuration read	N	Y	Y	N
1011	Configuration write	Type-1	Y	Y	Type-1
1100	Memory read multiple	Y	Y	Y	Y
1101	Dual address cycle	Y	Y	Y	Y
1110	Memory read line	Y	Y	Y	Y
1111	Memory write and invalidate	Y	Y	Y	Y

As indicated in Table 7–1, the following PCI commands are not supported by PCI 6150:

- PCI 6150 ignores reserved command codes and does not generate any reserved commands.
- PCI 6150 never initiates an interrupt acknowledge transaction and, as a target, ignores interrupt acknowledge transactions. Interrupt acknowledge transactions are expected to reside entirely on the primary PCI bus closest to the host bridge.
- PCI 6150 does not respond to special cycle transactions. To generate special cycle transactions on other PCI buses, either upstream or downstream, a Type-1 configuration command must be used.
- PCI 6150 does not generate Type-0 configuration transactions on the primary interface.

7.2 Single Address Phase

A 32-bit address uses a single address phase. This address is driven on AD[31:0], and the bus command is driven on P_CBE[3:0]

PCI 6150 supports the linear increment address mode only, which is indicated when the low 2 address bits are equal to 0. If either of the low 2 address bits is nonzero, PCI 6150 automatically disconnects the transaction after the first data transfer.

7.3 Dual Address Phase

PCI 6150 supports the Dual Address Cycle (DAC) bus command to transfer 64-bit addresses. In DAC transactions, the first address phase is during the initial assertion of frame, and the second address phase is one clock later. During the first address phase, the DAC command is presented on CBE[3:0], the lower 32 bits of the address on AD[31:0]. The second address phase has the cycle command on CBE[3:0], and the upper 32 bits of the address on AD[31:0].

DACs are used to access locations that are not in the first 4GB of PCI memory space. Addresses in the first 4GB of memory space always use a Single Address Cycle (SAC).
PCI 6150 supports DAC in the upstream and downstream direction.
PCI 6150 responds to DAC for the following commands only:

- Memory Write
- Memory Write and Invalidate
- Memory Read
- Memory Read Line
- Memory Read Multiple

7.4 Device Select (DEVSEL#) Generation

PCI 6150 always performs positive address decoding when accepting transactions on either the primary or secondary buses. PCI 6150 never subtractively decodes. Medium DEVSEL# timing is used for 33MHz operation and Slow DEVSEL# timing is used for 66MHz operation.

7.5 Data Phase

Depending on the command type, PCI 6150 can support multiple data phase PCI transactions. Write transactions are treated as either posted write or delayed write transactions.
Table 7–2 shows the method of forwarding used for each type of write operation.

Table 7–2, Write Transaction Forwarding

Type of Transaction	Type of Forwarding
Memory write	Posted
Memory write and invalidate	Posted
I/O write	Delayed
Type-1 configuration write	Delayed

7.5.1 Posted Write Transactions

When PCI 6150 determines that a memory write transaction is to be forwarded across the bridge, PCI 6150 asserts DEVSEL# with slow timing and TRDY# in the same cycle, provided that enough buffer space is available in the posted write data queue, and there are less than 4 outstanding posted transactions in the queue. PCI 6150 can accept 1 QUAD/DWORD of write data every PCI clock cycle; that is, no target wait states are inserted. Up to 256 bytes of posted write data is stored in internal posted write buffers and is eventually delivered to the target.

PCI 6150 continues to accept write data until one of the following events occurs:

- The initiator terminates the transaction normally.
- A cache line boundary or an aligned 4KB boundary is reached, depending on the transaction type.
- The posted write data buffer fills

When one of the last two events occurs, PCI 6150 returns a target disconnect to the requesting initiator on this data phase to terminate the transaction.

Once the posted write transaction is selected for completion, PCI 6150 requests ownership of the target bus. This can occur while PCI 6150 is still receiving data on the initiator bus. Once PCI 6150 has ownership of the target bus, and the target bus is detected in the idle condition, PCI 6150 generates the write cycle and continues to transfer write data until all write data corresponding to that transaction is delivered, or until a target termination is received. As long as write data exists in the queue, PCI 6150 can drive 1 QUAD/DWORD of write data each PCI clock cycle. If write data is flowing through PCI 6150 and the initiator stalls, PCI 6150 will insert wait states on the target bus if the queue empties.

PCI 6150 ends the transaction on the target bus when one of the following conditions is met:

- All posted write data has been delivered to the target.
- The target returns a target disconnect or target retry (PCI 6150 starts another transaction to deliver the rest of the write data).
- The target returns a target abort (PCI 6150 discards remaining write data).

The master latency timer expires, and PCI 6150 no longer has the target bus grant (PCI 6150 starts another transaction to deliver the remaining write data).

7.5.2 Memory Write and Invalidate Transactions

Memory Write and Invalidate transactions guarantee transfer of entire cache lines. By default, PCI 6150 will retry a Memory Write and Invalidate cycle until there is space for at least 1 cache line of data in the internal buffers. It will then complete the transaction on the secondary bus as a Memory Write and Invalidate cycle. PCI 6150 can also be programmed to accept Memory Write and Invalidate cycles under the same conditions as normal memory writes. In this case, if the write buffer fills before an entire cache line is transferred, PCI 6150 will disconnect and complete the write cycle on the secondary bus as a normal Memory Write cycle. (Register 46, bit12). PCI 6150 disconnects Memory Write and Invalidate commands at aligned cache line boundaries. The cache line size value in the cache line size register gives the number of DWORDs in a cache line. For PCI 6150, to generate Memory Write and Invalidate transactions, this cache line size value must be written to a value that is 8h, 10h, or 20h. If an invalid cache line size is programmed, wherein the value is 0, or is not a power of 2, or is greater than 20h DWORDs, PCI 6150 sets the cache line size to the minimum value of 8h. PCI 6150 always disconnects on the cache line boundary.

When the Memory Write and Invalidate transaction is disconnected before a cache line boundary is reached, typically because the posted write data buffer fills, the transaction is converted to a Memory Write transaction.

7.5.3 Delayed Write Transactions

A Delayed Write transaction is used to forward I/O Write and Type-1 configuration cycles through PCI 6150, and is limited to a single QUAD/DWORD data transfer.

When a write transaction is first detected on the initiator bus, PCI 6150 claims the access and returns a target retry to the initiator. During the cycle, PCI 6150 samples the bus command, address, and address parity bits. PCI 6150 also samples the first data QUAD/DWORD, byte enable bits, and data parity. Cycle information is placed into the delayed transaction queue if there are no other existing delayed transactions with the same cycle information, and if the delayed transaction queue is not full. When PCI 6150 schedules delayed write transaction to be the next cycle to be completed based on its ordering constraints, PCI 6150 initiates the transaction on the target bus. PCI 6150 transfers the write data to the target.

If PCI 6150 receives a target retry in response to the write transaction on the target bus, it continues to repeat the write transaction until the data transfer is completed, or until an error condition is encountered. If PCI 6150 is unable to deliver write data after 2^{24} attempts (programmable through register 45, bits 3-0), PCI 6150 ceases further write attempts and returns a target abort to the initiator. The delayed transaction is removed from the delayed transaction queue. PCI 6150 also asserts P_SERR# if the primary SERR# enable bit is set in the command register. When the initiator repeats the same write transaction (same command, address, byte enable bits, and data), after PCI 6150 has completed data delivery, and has all the complete cycle information in the queue, PCI 6150 claims the access returns TRDY# to the initiator, to indicate that the write data was transferred. If the initiator requests multiple QUAD/DWORD, PCI 6150 asserts STOP# in conjunction with TRDY# to signal a

target disconnect. Note that only those bytes of write data with valid byte enable bits are compared. If any of the byte enable bits are turned off (driven HIGH), the corresponding byte of write data is not compared.

If the initiator repeats the write transaction before the data has been transferred to the target, PCI 6150 returns a target retry to the initiator. PCI 6150 continues to return a target retry to the initiator until write data is delivered to the target or an error condition is encountered. When the write transaction is repeated, PCI 6150 does not make a new entry into the delayed transaction queue.

PCI 6150 implements a discard timer that starts counting when the delayed write completion is at the head of the delayed transaction queue. The initial value of this timer can be set to one of four values, selectable through both the primary and the secondary master timeout bits in the bridge control register as well as the master timeout divider bits in register 45h. If the discard timer expires before the write cycle is retried, PCI 6150 discards the delayed write transaction from the delayed transaction queue. PCI 6150 also conditionally asserts P_SERR#.

7.5.4 Write Transaction Address Boundaries

PCI 6150 imposes internal address boundaries when accepting write data. The aligned address boundaries are used to prevent PCI 6150 from continuing a transaction over a device address boundary and to provide an upper limit on maximum latency. PCI 6150 returns a target disconnects to the initiator when it reaches the aligned address boundaries under the conditions shown in Table 7–3.

Table 7–3, Write Transaction Disconnect Address Boundaries

Type of Transaction	Condition	Aligned Address Boundary
Delayed write	All	Disconnects after one data transfer
Posted memory write	Memory write disconnect control bit = 0[1]	4KB aligned address boundary
Posted memory write	Memory write disconnect control bit = 1[1]	Disconnects at cache line boundary
Posted memory write and invalidate	**Cache line size = 8,**	8h-DWORD aligned address boundary
Posted memory write and invalidate	Cache line size = 10h	10h-DWORD aligned address boundary
Posted memory write and invalidate	**Cache line size = 20h**	**20h-DWORD aligned address boundary**

[1]- Memory Write disconnect control bit is located in the chip control register at offset 40h in configuration space.

7.5.5 Buffering Multiple Write Transactions

PCI 6150 continues to accept posted memory write transactions as long as space for at least 1 DWORD of data in the posted write data buffer remains and there are less than 4 outstanding posted memory write cycles. If the posted write data buffer fills before the initiator terminates the write transaction, PCI 6150 returns a target disconnect to the initiator.

Delayed write transactions are posted as long as at least one open entry in the delayed transaction queue exists. PCI 6150 can queue up to four posted write transactions and four delayed transactions in both upstream and downstream directions.

7.5.6 Read Transactions

Delayed read forwarding is used for all read transactions crossing PCI 6150.
Delayed read transactions are treated as either prefetchable or nonprefetchable.
Table 7–4 shows the read behavior, prefetchable or nonprefetchable, for each type of read operation.

Table 7–4: Read Transaction Prefetching

Type of transaction	Read behavior
I/O read	Prefetching never done
Configuration read	Prefetching never done
Memory read	Downstream: prefetching used if address in prefetchable space Upstream: prefetching used if prefetch disable is off (default)
Memory read line	Prefetching always used if request is for more than one data transfer.
Memory read multiple	Prefetching always used if request is for more than one data transfer.

7.5.7 Prefetchable Read Transactions

A prefetchable read transaction is a read transaction where PCI 6150 performs speculative DWORD reads, transferring data from the target before it is requested from the initiator. This behavior allows a prefetchable read transaction to consist of multiple data transfers. Only the first byte enable bits can be forwarded. PCI 6150 forces all byte enable bits of subsequent transfers to be enabled.

Prefetchable behavior is used for memory read line and memory read multiple transactions, as well as for memory read transactions that fall into prefetchable memory space.

The amount of the prefetched data depends on the type of transaction. The amount of prefetching may also be affected by the amount of free buffer space available in PCI 6150, and by any read address boundaries encountered. In addition, there are several PCI 6150-specific registers that can be used to optimize read prefetch behavior.

Prefetching should not be used for those read transactions that have side effects in the target device, that is, control and status registers, FIFOs, and so on. The target device's base address register or registers indicate if a memory address region is prefetchable.

7.5.8 Nonprefetchable Read Transactions

A nonprefetchable read transaction is read transaction by the initiator into a nonprefetchable region, and is used for I/O and configuration read transactions, as well as for memory reads from nonprefetchable memory space. In this case, PCI 6150 requests 1 and only 1 DWORD from the target and disconnects the initiator after delivery of the first DWORD of read data.

Nonprefetchable read transactions should not be used for regions where extra read transactions could have side effects, such as FIFO memory, or control registers. Accordingly, if it is important to retain the value of the byte enable bits during the data phase, use nonprefetchable read transactions. If these locations are mapped in memory space, use the memory read command and map the target into nonprefetchable (memory-mapped I/O) memory space to utilize nonprefetching behavior.

7.5.9 Read Prefetch Address Boundaries

PCI 6150 imposes internal read address boundaries on read prefetching. The address boundary is used by PCI 6150 to calculate the initial amount of data that it will prefetch. During read transactions to prefetchable regions, PCI 6150 will prefetch data until it reaches one of these aligned address boundaries, unless the target signals a target disconnect before the read prefetch boundary is reached. Once the aligned address boundary is reached, PCI 6150 may optionally continue prefetching data, depending on certain conditions (see section on flow-through optimization). When PCI 6150 finishes transferring this read data to the initiator, it returns a target disconnect with the last data transfer, unless the initiator completes the transaction before all prefetched read data is delivered. Any leftover prefetched data is discarded.

Prefetchable read transactions in flow-through mode prefetch to the nearest aligned 4KB address boundary, or until the initiator deasserts FRAME#.

Table 7–5 shows the read prefetch address boundaries for read transactions during non-flow-through mode.

Table 7–5: Read Prefetch Address Boundaries

Type of transaction	Address space	Prefetch aligned address boundary
Configuration read	-	1 DWORD (no prefetch)
I/O read	-	1 DWORD (no prefetch)
Memory read	Nonprefetchable	1 DWORD (no prefetch)
Memory read	Prefetchable	Configured through prefetch count registers
Memory read line	Prefetchable	Configured through prefetch count registers
Memory read multiple	Prefetchable	Configured through prefetch count registers

7.5.10 Delayed Read Requests

PCI 6150 treats all read transactions as delayed read transactions, which means that the read request from the initiator is posted into a delayed transaction queue. Read data from the target is placed in the read data queue directed toward the initiator bus interface and is transferred to the initiator when the initiator repeats the read transaction.

When PCI 6150 accepts a delayed read request, it first samples the read address, read bus command, and address parity. When IRDY# is asserted, PCI 6150 then samples the byte enable bits for the first data phase. This information is entered into the delayed transaction queue. PCI 6150 terminates the transaction by signaling a target retry to the initiator. Upon reception of the target retry, the initiator is required to continue to repeat the same read transaction until at least one data transfer is completed, or until a target response other than a target retry (target abort, or master abort) is received.

7.5.11 Delayed Read Completion with Target

When a delayed read request is scheduled by PCI 6150 to be executed, PCI 6150 arbitrates for the target bus and initiates the read transaction, using the exact read address and read command captured from the initiator during the initial delayed read request. If the read transaction is a nonprefetchable read, PCI 6150 drives the captured byte enable bits during the next cycle. If the transaction is a prefetchable read transaction, it drives the captured first byte enable bits followed by 0 for the subsequent data phases. If PCI 6150 receives a target retry in response to the read transaction on the target bus, it continues to repeat the read transaction until at least one data transfer is completed, or until an error condition is encountered. If the transaction is terminated via normal master termination or target disconnect after at least one data transfer has been completed, PCI 6150 does not initiate any further attempts to read more data.

If PCI 6150 is unable to obtain read data from the target after 2^{24} attempts (default), PCI 6150 ceases further read attempts and returns a target abort to the initiator. The delayed transaction is removed from the delayed transaction queue. PCI 6150 also asserts P_SERR# if the primary SERR# enable bit is set in the command register.

Once PCI 6150 receives DEVSEL# and TRDY# from the target, it transfers the data stored in the internal read FIFO, before terminating the transaction. PCI 6150 can accept 1 DWORD/QWORD of read data each PCI clock cycle; no master wait states are inserted. The number of DWORD/QWORD transferred during a delayed read transaction depends on the conditions given in Table 7–5 (assuming no disconnect is received from the target).

7.5.12 Delayed Read Completion on Initiator Bus

When the transaction has been completed on the target bus, and the delayed read data is at the head of the read data queue, and all ordering constraints with posted write transactions have been satisfied, PCI 6150 transfers the data to the initiator when the initiator repeats the transaction. For memory read transactions, PCI 6150 aliases the memory read, memory read line, and memory read multiple bus commands when matching the bus command of the transaction to the bus command in the delayed transaction queue. PCI 6150 returns a target disconnect along with the transfer of the last DWORD of read data to the initiator. If PCI 6150 initiator terminates the transaction before all read data has been transferred, the remaining read data left in data buffers is discarded.

When the master repeats the transaction and starts transferring prefetchable read data from data buffers while the read transaction on the target bus is still in progress and before a read boundary is reached on the target bus, the read transaction starts operating in flow-through mode. Because data is flowing through the data buffers from the target to the initiator, long read bursts can then be sustained. In this case, the read transaction is allowed to continue until the initiator terminates the transaction, or until an aligned 4KB address boundary is reached, or until the buffer fills, whichever comes first. When the buffer empties, PCI 6150 reflects the stalled condition to the initiator by deasserting TRDY# for a maximum of 8 clock periods until more read data is available; otherwise, PCI 6150 will disconnect the cycle. When the initiator terminates the transaction, PCI 6150 deassertion of FRAME# on the initiator bus is forwarded to the target bus. Any remaining read data is discarded.

PCI 6150 implements a discard timer that starts counting when the delayed read completion is at the head of the delayed transaction queue, and the read data is at the head of the read data queue. The initial value of this timer is programmable through configuration register. If the initiator does not repeat the read transaction before Discard Timer expires, PCI 6150 discards the read transaction (and the read data from its queues). PCI 6150 also conditionally asserts P_SERR#.

PCI 6150 has the capability to post multiple delayed read requests, up to a maximum of four in each direction. If an initiator starts a read transaction that matches the address and read command of a read transaction that is already queued, the current read command is not stored as it is already contained in the delayed transaction queue.

7.5.13 Configuration Transactions

Configuration transactions are used to initialize a PCI system. Every PCI device has a configuration space that is accessed by configuration commands. All registers are accessible in configuration space only.
In addition to accepting configuration transactions for initialization of its own configuration space, PCI 6150 forwards configuration transactions for device initialization in hierarchical PCI systems, as well as for special cycle generation.

To support hierarchical PCI bus systems, Type-0 and Type-1 configuration transactions are specified.
Type-0 configuration transactions are issued when the intended target resides on the same PCI bus as the initiator. A Type-0 configuration transaction is identified by the configuration command and the Lowest 2 bits of the address set to 00b.

Type-1 configuration transactions are issued when the intended target resides on another PCI bus, or when a special cycle is to be generated on another PCI bus. A Type-1 configuration command is identified by the configuration command and the Lowest 2 address bits set to 01b.

The register number is found in both Type-0 and Type-1 formats and gives the DWORD address of the configuration register to be accessed. The function number is also included in both Type-0 and Type-1 formats and indicates which function of a multifunction device is to be accessed. For single-function devices, this value is not decoded. Type-1 configuration transaction addresses also include a 5-bit field designating the device number that identifies the device on the target PCI bus that is to be accessed. In addition, the bus number in Type-1 transactions specifies the PCI bus to which the transaction is targeted.

7.5.14 Type-0 Access to PCI 6150

The configuration space is accessed by a Type-0 configuration transaction on the primary interface. The configuration space cannot be accessed from the secondary bus. PCI 6150 responds to a Type-0 configuration transaction by asserting P_DEVSEL# when the following conditions are met during the address phase:

- The bus command is a configuration read transaction or configuration write transaction.
- Low 2 address bits P_AD[1:0] must be 00b.
- Signal P_IDSEL must be asserted.

PCI 6150 limits all configuration accesses to a single DWORD data transfer and returns a target disconnect with the first data transfer if additional data phases are requested. Because read transactions to configuration space do not have side effects, all bytes in the requested DWORD are returned, regardless of the value of the byte enable bits.

Type-0 configuration write and read transactions do not use data buffers; that is, these transactions are completed immediately, regardless of the state of the data buffers.
PCI 6150 ignores all Type-0 transactions initiated on the secondary interface.

7.5.15 Type-1 to Type-0 Translation

Type-1 configuration transactions are used specifically for device configuration in a hierarchical PCI bus system. A PCI-to-PCI bridge is the only type of device that should respond to a Type-1 configuration command. Type-1 configuration commands are used when the configuration access is intended for a PCI device that resides on a PCI bus other than the one where the Type-1 transaction is generated.

PCI 6150 performs a Type-1 to Type-0 translation when the Type-1 transaction is generated on the primary bus and is intended for a device attached directly to the secondary bus. PCI 6150 must convert the configuration command to a Type-0 format so that the secondary bus device can respond to it. Type-1 to Type-0 translations are performed only in the downstream direction; that is, PCI 6150 generates a Type-0 transaction only on the secondary bus, and never on the primary bus.

PCI 6150 responds to a Type-1 configuration transaction and translates it into a Type-0 transaction on the secondary bus when the following conditions are met during the address phase:

- The low 2 address bits on P_AD[1:0] are 01b.
- The bus number in address field P_AD[23:16] is equal to the value in the secondary bus number register in configuration space.
- The bus command on P_CBE[3:0] is a configuration read or configuration write transaction.

When PCI 6150 translates the Type-1 transaction to a Type-0 transaction on the secondary interface, it performs the following translations to the address:

- Sets the low 2 address bits on S_AD[1:0] to 00b.
- Decodes the device number and drives the bit pattern specified in Table 7–6 on S_AD[31:16] for the purpose of asserting the device's IDSEL signal.
- Sets S_AD[15:11] to 0.
- Leaves unchanged the function number and register number fields.

PCI 6150 asserts a unique address line based on the device number. These address lines may be used as secondary bus IDSEL signals. The mapping of the address lines depends on the device number in the Type-1 address bits P_AD[15:11]. Table 8–6 presents the mapping that PCI 6150 uses.

Table 7–6: Device Number to IDSEL S_AD Pin Mapping

Device Number	P_AD[15:11]	Secondary IDSEL S_AD[31:16]	S_AD Bit
0	00000	0000 0000 0000 0001	16
1	00001	0000 0000 0000 0010	17
2	00010	0000 0000 0000 0100	18
3	00011	0000 0000 0000 1000	19
4	00100	0000 0000 0001 0000	20
5	00101	0000 0000 0010 0000	21
6	00110	0000 0000 0100 0000	22
7	00111	0000 0000 1000 0000	23
8	01000	0000 0001 0000 0000	24
9	01001	0000 0010 0000 0000	25
10	01010	0000 0100 0000 0000	26
11	01011	0000 1000 0000 0000	27
12	01100	0001 0000 0000 0000	28
13	01101	0010 0000 0000 0000	29
14	01110	0100 0000 0000 0000	30
15	01111	1000 0000 0000 0000	31
Special Cycle	1XXXX	0000 0000 0000 0000	None

PCI 6150 can assert up to 16 unique address lines to be used as IDSEL signals for up to 16 devices on the secondary bus, for device numbers ranging from 0 through 15. Because of electrical loading constraints of the PCI bus, more than 16 IDSEL signals should not be necessary. However, if device numbers greater than 15 are desired, some external method of generating IDSEL lines must be used, and no upper address bits are then asserted. The configuration transaction is still translated and passed from the primary bus to the secondary bus. If no IDSEL pin is asserted to a secondary device, the transaction ends in a master abort.

PCI 6150 forwards Type-1 to Type-0 configuration read or write transactions as delayed transactions. Type-1 to Type-0 configuration read or write transactions are limited to a single 32-bit data transfer. When Type-1 to Type-0 configurations cycles are forwarded, address stepping is used, valid address is driven on the bus before the FRAME# is asserted. Type-0 configuration address stepping is programmable through register 46h, bits 6-4.

7.5.16 Type-1 to Type-1 Forwarding

Type-1 to Type-1 transaction forwarding provides a hierarchical configuration mechanism when two or more levels of PCI-to-PCI bridges are used.

When PCI 6150 detects a Type-1 configuration transaction intended for a PCI bus downstream from the secondary bus, PCI 6150 forwards the transaction unchanged to the secondary bus. Ultimately, this transaction is translated to a Type-0 configuration command or to a special cycle transaction by a downstream PCI-to-PCI bridge. Downstream Type-1 to Type-1 forwarding occurs when the following conditions are met during the address phase:

- The low 2 address bits are equal to 01b.
- The bus number falls in the range defined by the lower limit (exclusive) in the secondary bus number register and the upper limit (inclusive) in the subordinate bus number register.
- The bus command is a Configuration Read or Write transaction.

PCI 6150 also supports Type-1 to Type-1 forwarding of configuration write transactions upstream to support upstream special cycle generation. A Type-1 configuration command is forwarded upstream when the following conditions are met:

- The low 2 address bits are equal to 01b.
- The bus number falls outside the range defined by the lower limit (inclusive) in the secondary bus number register and the upper limit (inclusive) in the subordinate bus number register.
- The device number in address bits AD[15:11] is equal to 11111b.
- The function number in address bits AD[10:8] is equal to 111b.
- The bus command is a Configuration Write transaction.

PCI 6150 forwards Type-1 to Type-1 configuration write transactions as delayed transactions. Type-1 to Type-1 configuration write transactions are limited to a single data transfer.

7.5.17 Special Cycles

The Type-1 configuration mechanism is used to generate special cycle transactions in hierarchical PCI systems. Special cycle transactions are ignored by acting as a target and are not forwarded across the bridge. Special cycle transactions can be generated from Type-1 configuration write transactions in either the upstream or the downstream direction.

PCI 6150 initiates a special cycle on the target bus when a Type-1 Configuration Write transaction is detected on the initiating bus and the following conditions are met during the address phase:

- The low 2 address bits on AD[1:0] are equal to 01b.
- The device number in address bits AD[15:11] is equal to 11111b.
- The function number in address bits AD[10:8] is equal to 111b.
- The register number in address bits AD[7:2] is equal to 000000b.
- The bus number is equal to the value in the secondary bus number register in configuration space for downstream forwarding or equal to the value in the primary bus number register in configuration space for upstream forwarding.
- The bus command on CBE is a Configuration Write command.

When PCI 6150 initiates the transaction on the target interface, the bus command is changed from configuration write to special cycle. The address and data are forwarded unchanged. Devices that use special cycles ignore the address and decode only the bus command. The data phase contains the special cycle message. The transaction is forwarded as a delayed transaction, but in this case the target response is not forwarded back (because special cycles result in a master abort). Once the transaction is completed on the target bus, through detection of the master abort condition, PCI 6150 responds with TRDY# to the next attempt of the configuration transaction from the initiator. If more than one data transfer is requested, PCI 6150 responds with a target disconnect operation during the first data phase.

7.6 Transaction Termination

This section describes how PCI 6150 returns transaction termination conditions back to the initiator.
The initiator can terminate transactions with one of the following types of termination:

- Normal termination: It occurs when the initiator deasserts FRAME# at the beginning of the last data phase, and deasserts IRDY# at the end of the last data phase in conjunction with either TRDY# or STOP# assertion from the target.
- Master abort: It occurs when no target response is detected. When the initiator does not detect a DEVSEL# from the target within five clock cycles after asserting FRAME#, the initiator terminates the transaction with a master abort. If FRAME# is still asserted, the initiator deasserts FRAME# on the next cycle, and then deasserts IRDY# on the following cycle. IRDY# must be asserted in the same cycle in which FRAME# deasserts. If FRAME# is already deasserted, IRDY# can be deasserted on the next clock cycle following detection of the master abort condition.

The target can terminate transactions with one of the following types of termination:

- Normal termination: TRDY# and DEVSEL# asserted in conjunction with FRAME# deasserted and IRDY# asserted.
- Target retry: STOP# and DEVSEL# asserted without TRDY# during the first data phase. No data transfers occur during the transaction. This transaction must be repeated.
- Target disconnect (with data transfer): STOP# and DEVSEL# asserted with TRDY#. Signals that this is the last data transfer of the transaction.
- Target disconnect (without data transfer): STOP# and DEVSEL# asserted without TRDY# after previous data transfers have been made. Indicates that no more data transfers will be made during this transaction.
- Target abort: STOP# asserted without DEVSEL# and without TRDY#. Indicates that the target will never be able to complete this transaction. DEVSEL# must be asserted for at least one cycle during the transaction before the target abort is signaled.

7.6.1 Master Termination Initiated by PCI 6150

PCI 6150, as an initiator, uses normal termination if DEVSEL# is returned by the target within five clock cycles of PCI 6150's assertion of FRAME# on the target bus. As an initiator, PCI 6150 terminates a transaction when the following conditions are met:

- During a delayed write transaction, a single DWORD/QWORD is delivered.
- During a nonprefetchable read transaction, a single DWORD/QWORD is transferred from the target.
- During a prefetchable read transaction, a prefetch boundary is reached.
- For a posted write transaction, all write data for the transaction is transferred from data buffers to the target.
- For a burst transfer, with the exception of memory write and invalidate transactions, the master latency timer expires and PCI 6150's bus grant is deasserted.
- The target terminates the transaction with a retry, disconnect, or target abort.

If PCI 6150 is delivering posted write data when it terminates the transaction because the master latency timer expires, it initiates another transaction to deliver the remaining write data. The address of the transaction is updated to reflect the address of the current DWORD to be delivered.
If PCI 6150 is prefetching read data when it terminates the transaction because the master latency timer expires, it does not repeat the transaction to obtain more data.

7.6.2 Master Abort Received by PCI 6150

If the initiator initiates a transaction on the target bus and does not detect DEVSEL# returned by the target within five clock cycles of PCI 6150's assertion of FRAME#, PCI 6150 terminates the transaction as specified through the master abort mode bit of the bridge control register.

For delayed read and write transactions, PCI 6150 can either assert TRDY# and return FFFF_FFFFh for reads, or return target abort. SERR# is also optionally asserted.

When a master abort is received in response to a posted write transaction, PCI 6150 discards the posted write data and makes no more attempts to deliver the data. PCI 6150 sets the received master abort bit in the status register when the master abort is received on the primary bus, or it sets the received master abort bit in the secondary status register when the master abort is received on the secondary interface. When a master abort is detected in response to a posted write transaction, and the master abort mode bit is set, PCI 6150 also asserts P_SERR# if enabled by the SERR# enable bit in the command register and if not disabled by the device-specific P_SERR# disable bit for master abort during posted write transactions (that is, master abort mode = 1; SERR# enable bit = 1; and P_SERR# disable bit for master aborts = 0.

7.6.3 Target Termination Received by PCI 6150

When PCI 6150 initiates a transaction on the target bus and the target responds with DEVSEL#, the target can end the transaction with one of the following types of termination:

- Normal termination (upon deassertion of FRAME#)
- Target retry
- Target disconnect
- Target abort

PCI 6150 handles these terminations in different ways, depending on the type of transaction being performed.

7.6.3.1 Delayed Write Target Termination Response

When PCI 6150 initiates a delayed write transaction, the type of target termination received from the target can be passed back to the initiator. Table 7–7 shows the response to each type of target termination that occurs during a delayed write transaction.

PCI 6150 repeats a delayed write transaction until one of the following conditions is met:

- PCI 6150 completes at least one data transfer.
- PCI 6150 receives a master abort.
- PCI 6150 receives a target abort.

PCI 6150 makes 2^{24} (default) write attempts resulting in a response of target retry.

Table 7–7: Response to Delayed Write Target Termination

Target Termination	Response
Normal	Return disconnect to initiator with first data transfer only if multiple data phases requested.
Target retry	Return target retry to initiator. Continue write attempts to target.
Target disconnect	Return disconnect to initiator with first data transfer only if multiple data phases requested.
Target abort	Return target abort to initiator. Set received target abort bit in target interface status register. Set signaled target abort bit in initiator interface status register.

After PCI 6150 makes 2^{24} attempts of the same delayed write transaction on the target bus, PCI 6150 asserts P_SERR# if the primary SERR# enable bit is set in the command register and the implementation-specific P_SERR# disable bit for this condition is not set in the P_SERR# event disable register. PCI 6150 stops initiating transactions in response to that delayed write transaction. The delayed write request is discarded. Upon a subsequent write transaction attempt by the initiator, PCI 6150 returns a target abort.

7.6.3.2 Posted Write Target Termination Response

When PCI 6150 initiates a posted write transaction, the target termination cannot be passed back to the initiator. Table 7–8 shows the response to each type of target termination that occurs during a posted write transaction.

Table 7–8: Response to Posted Write Target Termination

Target termination	Response
Normal	No additional action.
Target retry	Repeat write transaction to target.
Target disconnect	Initiate write transaction to deliver remaining posted write data.
Target abort	Set received target abort bit in the target interface status register. Assert P_SERR# if enabled, and set the signaled system error bit in primary status register.

Note that when a target retry or target disconnect is returned and posted write data associated with that transaction remains in the write buffers, PCI 6150 initiates another write transaction to attempt to deliver the rest of the write data. In the case of a target retry, the exact same address will be driven as for the initial write transaction attempt. If a target disconnect is received, the address that is driven on a subsequent write transaction attempt is updated to reflect the address of the current DWORD. If the initial write transaction is a memory write and invalidate transaction, and a partial delivery of write data to the target is performed before a target disconnect is received, PCI 6150 uses the memory write command to deliver the rest of the write data because less than a cache line will be transferred in the subsequent write transaction attempt.

After PCI 6150 makes 2^{24} write transaction attempts and fails to deliver all the posted write data associated with that transaction, PCI 6150 asserts P_SERR# if the primary SERR# enable bit is set in the command register and the device-specific P_SERR# disable bit for this condition is not set in the P_SERR# event disable register. The write data is discarded.

7.6.3.3 Delayed Read Target Termination Response

When PCI 6150 initiates a delayed read transaction, the abnormal target responses can be passed back to the initiator. Other target responses depend on how much data the initiator requests. Table 7–9 shows the response to each type of target termination that occurs during a delayed read transaction.

Table 7–9: Response to Delayed Read Target Termination

Target termination	Response
Normal	If prefetchable, target disconnect only if initiator requests more data than read from target. If nonprefetchable, target disconnect on first data phase.
Target retry	Reinitiate read transaction to target
Target disconnect	If initiator requests more data than read from target, return target disconnect to initiator
Target abort	Return target abort to initiator. Set received target abort bit in the target interface status register. Set signaled target abort bit in the initiator interface status register.

PCI 6150 repeats a delayed read transaction until one of the following conditions is met:

- PCI 6150 completes at least one data transfer.
- PCI 6150 receives a master abort.
- PCI 6150 receives a target abort.
- PCI 6150 makes 2^{24} read attempts resulting in a response of target retry.

After PCI 6150 makes 2^{24} attempts of the same delayed read transaction on the target bus, PCI 6150 asserts P_SERR# if the primary SERR# enable bit is set in the command register and the implementation-specific P_SERR# disable bit for this condition is not set in the P_SERR# event disable register. PCI 6150 stops initiating transactions in response to that delayed read transaction. The delayed read request is discarded. Upon a subsequent read transaction attempt by the initiator, PCI 6150 returns a target abort.

7.6.4 Target Termination Initiated by PCI 6150

PCI 6150 can return a Target Retry, Target Disconnect, or Target-Abort to an initiator for reasons other than detection of that condition at the target interface.

7.6.4.1 Target Retry

PCI 6150 returns a Target Retry to the initiator when it cannot accept write data or return read data as a result of internal conditions. PCI 6150 returns a target retry to an initiator when any of the following conditions is met:

For delayed write transactions:

- The transaction is being entered into the delayed transaction queue.
- The transaction has already been entered into the delayed transaction queue, but target response has not yet been received.
- Target response has been received but the posted memory write ordering rule prevents the cycle from being completed.
- The delayed transaction queue is full, and the transaction cannot be queued.
- A transaction with the same address and command has been queued.
- A locked sequence is being propagated across PCI 6150, and the write transaction is not a locked transaction.
- The target bus is locked and the write transaction is a locked transaction.

For delayed read transactions:

- The transaction is being entered into the delayed transaction queue.
- The read request has already been queued, but read data is not yet available.
- Data has been read from the target, but it is not yet at the head of the read data queue, or a posted write transaction precedes it.
- The delayed transaction queue is full, and the transaction cannot be queued.
- A delayed read request with the same address and bus command has already been queued.
- A locked sequence is being propagated across PCI 6150, and the read transaction is not a locked transaction.
- The target bus is locked and the write transaction is a locked transaction.

For posted write transactions:

- The posted write data buffer does not have enough space for the address and at least two QWORDs of write data.
- A locked sequence is being propagated across PCI 6150, and the write transaction is not a locked transaction.

When a target retry is returned to the initiator of a delayed transaction, the initiator must repeat the transaction with the same address and bus command as well as the data if this is a write transaction, within the time frame specified by the master timeout value; otherwise, the transaction is discarded from the buffers.

7.6.4.2 Target Disconnected

PCI 6150 returns a Target Disconnect to an initiator when one of the following conditions is met:

- PCI 6150 hits an internal address boundary
- PCI 6150 cannot accept any more write data
- PCI 6150 has no more read data to deliver

7.6.4.3 Target-Abort

PCI 6150 returns a Target-Abort to an initiator when one of the following conditions is met:

- PCI 6150 is returning a target abort from the intended target.
- PCI 6150 detects a master abort on the target, and the master abort mode bit is set.
- PCI 6150 is unable to obtain delayed read data from the target or to deliver delayed write data to the target after 2^{24} attempts.

When PCI 6150 returns a target abort to the initiator, it sets the signaled target abort bit in the status register corresponding to the initiator interface.

8 Address Decoding

PCI 6150 uses three address ranges that control I/O and memory transaction forwarding. These address ranges are defined by base and limit address registers in the configuration space. This chapter describes these address ranges, as well as ISA-mode and VGA-addressing support.

8.1 Address Ranges

PCI 6150 uses the following address ranges that determine which I/O and memory transactions are forwarded from the primary PCI bus to the secondary PCI bus, and from the secondary bus to the primary bus:

- One 32-bit I/O address range
- One 32-bit memory-mapped I/O (nonprefetchable memory) range
- One 32-bit prefetchable memory address range

Transactions falling within these ranges are forwarded downstream from the primary PCI bus to the secondary PCI bus. Transactions falling outside these ranges are forwarded upstream from the secondary PCI bus to the primary PCI bus.

8.2 I/O Address Decoding

PCI 6150 uses the following mechanisms that are defined in the configuration space to specify the I/O address space for downstream and upstream forwarding:

- I/O base and limit address registers
- The ISA enable bit
- The VGA mode bit
- The VGA snoop bit

This section provides information on the I/O address registers and ISA mode.

To enable downstream forwarding of I/O transactions, the I/O enable bit must be set in the command register in configuration space. If the I/O enable bit is not set, all I/O transactions initiated on the primary bus are ignored. To enable upstream forwarding of I/O transactions, the master enable bit must be set in the command register. If the master enable bit is not set, PCI 6150 ignores all I/O and memory transactions initiated on the secondary bus. Setting the master enable bit also allows upstream forwarding of memory transactions.

Caution: If any configuration state affecting I/O transaction forwarding is changed by a configuration write operation on the primary bus at the same time that I/O transactions are ongoing on the secondary bus, the PCI 6150 response to the secondary bus I/O transactions is not predictable. Configure the I/O base and limit address registers, ISA enable bit, VGA mode bit, and VGA snoop bit before setting the I/O enable and master enable bits, and change them subsequently only when the primary and secondary PCI buses are idle.

8.2.1 I/O Base and Limit Address Registers

PCI 6150 implements one set of I/O base and limit address registers in configuration space that define an I/O address range downstream forwarding. PCI 6150 supports 32-bit I/O addressing, which Allows I/O addresses downstream of PCI 6150 to be mapped anywhere in a 4GB I/O address space.

I/O transactions with addresses that fall inside the range defined by the I/O base and limit registers are forwarded downstream from the primary PCI bus to the secondary PCI bus. I/O transactions with addresses that fall outside this range are forwarded upstream from the secondary PCI bus to the primary PCI bus.
The I/O range can be turned off by setting the I/O base address to a value greater than that of the I/O limit address. When the I/O range is turned off, all I/O transactions are forwarded upstream, and no I/O transactions are forwarded downstream.

The I/O range has a minimum granularity of 4KB and is aligned on a 4KB boundary. The maximum I/O range is 4GB in size.

The I/O base register consists of an 8-bit field at configuration address 1Ch, and a 16-bit field at address 30h. The top 4 bits of the 8-bit field define bits [15:12] of the I/O base address. The bottom 4 bits read only as 1h to indicate that PCI 6150 supports 32-bit I/O addressing. Bits [11:0] of the base address are assumed to be 0, which naturally aligns the base address to a 4KB boundary. The 16 bits contained in the I/O base upper 16 bits register at configuration offset 30h define AD[31:16] of the I/O base address. All 16 bits are read/write. After primary bus reset or chip reset, the value of the I/O base address is initialized to 0000_0000h.

The I/O limit register consists of an 8-bit field at configuration offset 1Dh and a 16-bit field at offset 32h. The top 4 bits of the 8-bit field define bits [15:12] of the I/O limit address. The bottom 4 bits read only as 1h to indicate that 32-bit I/O addressing is supported. Bits [11:0] of the limit address are assumed to be FFFh, which naturally aligns the limit address to the top of a 4KB I/O address block. The 16 bits contained in the I/O limit upper 16 bits register at configuration offset 32h define AD[31:16] of the I/O limit address. All 16 bits are read/write. After primary bus reset or chip reset, the value of the I/O limit address is reset to 0000 0FFFh.

Note
Write these registers with their appropriate values before setting either the I/O enable bit or the master enable bit in the command register in configuration space.

8.3 ISA Mode

PCI 6150 supports ISA mode by providing an ISA enable bit in the bridge control register in configuration space. ISA mode modifies the response of PCI 6150 inside the I/O address range in order to support mapping of I/O space in the presence of an ISA bus in the system. This bit only affects the response of PCI 6150 when the transaction falls inside the address range defined by the I/O base and limit address registers, and only when this address also falls inside the first 64KB of I/O space (address bits [31:16] are 0000h).

When the ISA enable bit is set, PCI 6150 does not forward downstream any I/O transactions addressing the top 768 bytes of each aligned 1KB block. Only those transactions addressing the bottom 256 bytes of an aligned 1KB block inside the base and limit I/O address range are forwarded downstream. Transactions above the 64KB I/O address boundary are forwarded as defined by the address range defined by the I/O base and limit registers.

Accordingly, if the ISA enable bit is set, PCI 6150 forwards upstream those I/O transactions addressing the top 768 bytes of each aligned 1KB block within the first 64KB of I/O space. The master enable bit in the command configuration register must also be set to enable upstream forwarding. All other I/O transactions initiated on the secondary bus are forwarded upstream only if they fall outside the I/O address range.

When the ISA enable bit is set, devices downstream of PCI 6150 can have I/O space mapped into the first 256 bytes of each 1KB chunk below the 64KB boundary, or anywhere in I/O space above the 64KB boundary.

8.4 Memory Address Decoding

PCI 6150 has three mechanisms for defining memory address ranges for forwarding of memory transactions:

- Memory-mapped I/O base and limit address registers
- Prefetchable memory base and limit address registers
- VGA mode

This section describes the first two mechanisms.
To enable downstream forwarding of memory transactions, the memory enable bit must be set in the command register in configuration space. To enable upstream forwarding of memory transactions, the master enable bit must be set in the command register. Setting the master enable bit also allows upstream forwarding of I/O transactions.

Caution
If any configuration state affecting memory transaction forwarding is changed by a configuration write operation on the primary bus at the same time that memory transactions are ongoing on the secondary bus, response to the secondary bus memory transactions is not predictable. Configure the memory-mapped I/O base and limit address registers, prefetchable memory base and limit address registers, and VGA mode bit before setting the memory enable and master enable bits, and change them subsequently only when the primary and secondary PCI buses are idle.

8.4.1 Memory-Mapped I/O Base and Limit Address Registers

Memory-mapped I/O is also referred to as nonprefetchable memory. The memory-mapped I/O base address and memory-mapped I/O limit address registers define an address range that PCI 6150 uses to determine when to forward memory commands. PCI 6150 forwards a memory transaction from the primary to the secondary interface if the transaction address falls within the memory-mapped I/O address range. PCI 6150 ignores memory transactions initiated on the secondary interface that fall into this address range. Any transactions that fall outside this address range are ignored on the primary interface and are forwarded upstream from the secondary interface (provided that they do not fall into the prefetchable memory range or are not forwarded downstream by the VGA mechanism).

The memory-mapped I/O range supports 32-bit addressing only. The PCI-to-PCI Bridge Architecture Specification does not provide for 64-bit addressing in the memory-mapped I/O space. The memory-mapped I/O address range has a granularity and alignment of 1MB. The maximum memory-mapped I/O address range is 4GB.
The memory-mapped I/O address range is defined by a 16-bit memory-mapped I/O base address register at configuration offset 20h and by a 16-bit memory-mapped I/O limit address register at offset 22h. The top 12 bits of each of these registers correspond to bits [31:20] of the memory address. The low 4 bits are hardwired to 0. The low 20 bits of the memory-mapped I/O base address are assumed to be 0 0000h, which results in a natural alignment to a 1MB boundary. The low 20 bits of the memory-mapped I/O limit address are assumed to be F FFFFh, which results in an alignment to the top of a 1MB block.

Note
Write these registers with their appropriate values before setting either the memory enable bit or the master enable bit in the command register in configuration space.
To turn off the memory-mapped I/O address range, write the memory-mapped I/O base address register with a value greater than that of the memory-mapped I/O limit address register.

8.4.2 Prefetchable Memory Base and Limit Address Registers

Locations accessed in the prefetchable memory address range must have true memory-like behavior and must not exhibit side effects when read. This means that extra reads to a prefetchable memory location must have no side effects. PCI 6150 prefetches for all types of memory read commands in this address space.

PCI 6150 prefetchable memory base address and prefetchable memory limit address registers define an address range that PCI 6150 uses to determine when to forward memory commands. PCI 6150 forwards a memory transaction from the primary to the secondary interface if the transaction address falls within the prefetchable memory address range. PCI 6150 ignores memory transactions initiated on the secondary interface that fall into this address range. PCI 6150 does not respond to any transactions that fall outside this address range on the primary interface and forwards those transactions upstream from the secondary interface (provided that they do not fall into the memory-mapped I/O range or are not forwarded by the VGA mechanism).

PCI 6150 prefetchable memory range supports 64-bit addressing and provides additional registers to define the upper 32 bits of the memory address range, PCI 6150 prefetchable memory base address upper 32 bits register, and the prefetchable memory limit address upper 32 bits register. For address comparison, a single address cycle (32-bit address) prefetchable memory transaction is treated like a 64-bit address transaction where the upper 32 bits of the address are equal to 0. This upper 32-bit value of 0 is compared to the prefetchable memory base address upper 32 bits register and the prefetchable memory limit address upper 32 bits register. The prefetchable memory base address upper 32 bits register must be 0 in order to pass any single address cycle transactions downstream.
The prefetchable memory address range has a granularity and alignment of 1MB. The maximum memory address range is 4GB when 32-bit addressing is used, and 2^{64} bytes when 64-bit addressing is used.

The prefetchable memory address range is defined by a 16-bit prefetchable memory base address register at configuration offset 24h and by a 16-bit prefetchable memory limit address register at offset 26h. The top 12 bits of each of these registers correspond to bits [31:20] of the memory address. The low 4 bits are hardwired to 1h, indicating 64-bit address support. The low 20 bits of the prefetchable memory base address are assumed to be 0 0000h, which results in a natural alignment to a 1MB boundary. The low 20 bits of the prefetchable memory limit address are assumed to be F_FFFFh, which results in an alignment to the top of a 1MB block.

Note

Write these registers with their appropriate values before setting either the memory enable bit or the master enable bit in the command register in configuration space.

To turn off the prefetchable memory address range, write the prefetchable memory base address register with a value greater than that of the prefetchable memory limit address register. The entire base value must be greater than the entire limit value, meaning that the upper 32 bits must be considered. Therefore, to disable the address range, the upper 32 bits registers can both be set to the same value, while the lower base register is set greater than the lower limit register; otherwise, the upper 32-bit base must be greater than the upper 32-bit limit.

8.5 VGA Support

PCI 6150 provides two modes for VGA support:

- VGA mode, supporting VGA-compatible addressing
- VGA snoop mode, supporting VGA palette forwarding

8.5.1 VGA Mode

When a VGA-compatible device exists downstream from PCI 6150, set the VGA mode bit in the bridge control register in configuration space to enable VGA mode. When PCI 6150 is operating in VGA mode, it forwards downstream those transactions addressing the VGA frame buffer memory and VGA I/O registers, regardless of the values of the base and limit address registers. PCI 6150 ignores transactions initiated on the secondary interface addressing these locations.

The VGA frame buffer consists of the following memory address range: 000A_0000h – 000B_FFFFh
Read transactions to frame buffer memory are treated as nonprefetchable. PCI 6150 requests only a single data transfer from the target, and read byte enable bits are forwarded to the target bus.
The VGA I/O addresses consist of the following I/O addresses:

- 3B0h–3BBh
- 3C0h–3DFh

These I/O addresses are aliased every 1KB throughout the first 64KB of I/O space. This means that address bits [15:10] are not decoded and can be any value, while address bits [31:16] must be all 0s.
VGA BIOS addresses starting at C0000h are not decoded in VGA mode.

8.5.2 VGA Snoop Mode

PCI 6150 provides VGA snoop mode, Allowing for VGA palette write transactions to be forwarded downstream. This mode is used when a graphics device downstream from PCI 6150 needs to snoop or respond to VGA palette write transactions. To enable the mode, set the VGA snoop bit in the command register in configuration space.
Note that PCI 6150 claims VGA palette write transactions by asserting DEVSEL# in VGA snoop mode.
When the VGA snoop bit is set, PCI 6150 forwards downstream transactions with the following I/O addresses:

- 3C6h
- 3C8h
- 3C9h

Note that these addresses are also forwarded as part of the VGA compatibility mode previously described. Again, address bits [15:10] are not decoded, while address bits [31:16] must be equal to 0, which means that these addresses are aliased every 1KB throughout the first 64KB of I/O space.

Note
If both the VGA mode bit and the VGA snoop bit are set, PCI 6150 behaves in the same way as if only the VGA mode bit were set.

9 Transaction Ordering

To maintain data coherency and consistency, PCI 6150 complies with the ordering rules set forth in the PCI Local Bus Specification, Revision 2.2.

9.1 Transaction Ordering

This section describes the ordering rules that control transaction forwarding across PCI 6150. For a more detailed discussion of transaction ordering, see Appendix E of the PCI Local Bus Specification, Revision 2.2.

9.1.1 Transactions Governed by Ordering Rules

Ordering relationships are established for the following classes of transactions crossing PCI 6150:

- Posted write transactions, comprised of memory write and memory write and invalidate transactions
  Posted write transactions complete at the source before they complete at the destination; that is, data is written into intermediate data buffers before it reaches the target.
- Delayed write request transactions, comprised of I/O write and configuration write transactions
  Delayed write requests are terminated by target retry on the initiator bus and are queued in the delayed transaction queue. A delayed write transaction must complete on the target bus before it completes on the initiator bus.
- Delayed write completion transactions, also comprised of I/O write and configuration write transactions.
  Delayed write completion transactions have been completed on the target bus, and the target response is queued in the buffers. A delayed write completion transaction proceeds in the direction opposite that of the original delayed write request; that is, a delayed write completion transaction proceeds from the target bus to the initiator bus.
- Delayed read request transactions, comprised of all memory read, I/O read, and configuration read transactions.
  Delayed read requests are terminated by target retry on the initiator bus and are queued in the delayed transaction queue.
- Delayed read completion transactions, comprised of all memory read, I/O read, and configuration read transactions.
  Delayed read completion transactions have been completed on the target bus, and the read data has been queued in the read data buffers. A delayed read completion transaction proceeds in the direction opposite that of the original delayed read request; that is, a delayed read completion transaction proceeds from the target's bus to the initiator's bus.

PCI 6150 does not combine or merge write transactions:

- PCI 6150 does not combine separate write transactions into a single write transaction—this optimization is best implemented in the originating master.
- PCI 6150 does not merge bytes on separate masked write transactions to the same DWORD address—this optimization is also best implemented in the originating master.
- PCI 6150 does not collapse sequential write transactions to the same address into a single write transaction—the PCI Local Bus Specification does not permit this combining of transactions.

9.1.2 General Ordering Guidelines

Independent transactions on the primary and secondary buses have a relationship only when those transactions cross PCI 6150.

The following general ordering guidelines govern transactions crossing PCI 6150:

- The ordering relationship of a transaction with respect to other transactions is determined when the transaction completes, that is, when a transaction ends with a termination other than target retry.
- Requests terminated with target retry can be accepted and completed in any order with respect to other transactions that have been terminated with target retry. If the order of completion of delayed requests is important, the initiator should not start a second delayed transaction until the first one has been completed. If more than one delayed transaction is initiated, the initiator should repeat all the delayed transaction requests,

using some fairness algorithm. Repeating a delayed transaction cannot be contingent on completion of another delayed transaction; otherwise, a deadlock can occur.

- Write transactions flowing in one direction have no ordering requirements with respect to write transactions flowing in the other direction. PCI 6150 can accept posted write transactions on both interfaces at the same time, as well as initiate posted write transactions on both interfaces at the same time.
- The acceptance of a posted memory write transaction as a target can never be contingent on the completion of a non-locked, nonposted transaction as a master. This is true of PCI 6150 and must also be true of other bus agents; otherwise, a deadlock can occur.
- PCI 6150 accepts posted write transactions, regardless of the state of completion of any delayed transactions being forwarded across PCI 6150.

9.1.3 Ordering Rules

Table 9–1 shows the ordering relationships of all the transactions and refers by number to the ordering rules that follow.

Table 9–1: Summary of Transaction Ordering

Pass	Posted Write	Delayed read Request	Delayed Write Request	Delayed Read Completion	Delayed Write Completion
Posted write	N[1]	Y[5]	Y[5]	Y[5]	Y[5]
Delayed read request	N[2]	Y	Y	Y	Y
Delayed write request	N[4]	Y	Y	Y	Y
Delayed read completion	N[3]	Y	Y	Y	Y
Delayed write completion	Y	Y	Y	Y	Y

Note
The superscript accompanying some of the table entries refers to any applicable ordering rule listed in this section. Many entries are not governed by these ordering rules; therefore, the implementation can choose whether or not the transactions pass each other.

The entries without superscripts reflect PCI 6150's implementation choices.
The following ordering rules describe the transaction relationships. Each ordering rule is followed by an explanation, and the ordering rules are referred to by number in Table 9–1. These ordering rules apply to posted write transactions, delayed write and read requests, and delayed write and read completion transactions crossing PCI 6150 in the same direction. Note that delayed completion transactions cross PCI 6150 in the direction opposite that of the corresponding delayed requests.

1. Posted write transactions must complete on the target bus in the order in which they were received on the initiator bus.

   The subsequent posted write transaction can be setting a flag that covers the data in the first posted write transaction; if the second transaction were to complete before the first transaction, a device checking the flag could subsequently consume stale data.

2. A delayed read request traveling in the same direction as a previously queued posted write transaction must push the posted write data ahead of it. The posted write transaction must complete on the target bus before the delayed read request can be attempted on the target bus.

   The read transaction can be to the same location as the write data, so if the read transaction were to pass the write transaction, it would return stale data.

3. A delayed read completion must ‘‘pull’’ ahead of previously queued posted write data traveling in the same direction. In this case, the read data is traveling in the same direction as the write data, and the initiator of the read transaction is on the same side of the as the target of the write transaction. The posted write transaction must complete to the target before the read data is returned to the initiator.

   The read transaction can be to a status register of the initiator of the posted write data and therefore should not complete until the write transaction is complete.

4. Delayed write requests cannot pass previously queued posted write data. As in the case of posted memory write transactions, the delayed write transaction can be setting a flag that covers the data in the posted write transaction; if the delayed write request were to complete before the earlier posted write transaction, a device checking the flag could subsequently consume stale data.

5. Posted write transactions must be given opportunities to pass delayed read and write requests and completions. Otherwise, deadlocks may occur when bridges that support delayed transactions are used in the same system with bridges that do not support delayed transactions. A fairness algorithm is used to arbitrate between the posted write queue and the delayed transaction queue.

PCI 6150 can generate cycles across the bridge in the same order than requested if bit 2 of register 46h is set. By default, requested cycles can execute out of order across the bridge, if all other ordering rules are satisfied. So, if PCI 6150 starts a delayed transaction that is retried by the target, it can execute another transaction in the delayed transaction request queue. Also, if there is both delayed read and delayed write requests in the queue, and the read data FIFOs are full, PCI 6150 may execute the delayed write request before the delayed read request.

On cycle completion, PCI 6150 may complete cycles in a different order than requested by the initiator.

9.1.4 Data Synchronization

Data synchronization refers to the relationship between interrupt signaling and data delivery. The PCI Local Bus Specification, Revision 2.2, provides the following alternative methods for synchronizing data and interrupts:

- The device signaling the interrupt performs a read of the data just written (software).
- The device driver performs a read operation to any register in the interrupting device before accessing data written by the device (software).
- System hardware guarantees that write buffers are flushed before interrupts are forwarded.

PCI 6150 does not have a hardware mechanism to guarantee data synchronization for posted write transactions. Therefore, all posted write transactions must be followed by a read operation, either from the device to the location just written (or some other location along the same path), or from the device driver to one of the device registers.

10 Error Handling

PCI 6150 checks, forwards, and generates parity on both the primary and secondary interfaces. To maintain transparency, PCI 6150 always tries to forward the existing parity condition on one bus to the other bus, along with address and data. PCI 6150 always attempts to be transparent when reporting errors, but this is not always possible, given the presence of posted data and delayed transactions.

To support error reporting on the PCI bus, PCI 6150 implements the following:

- PERR# and SERR# signals on both the primary and secondary interfaces
- Primary status and secondary status registers
- The device-specific P_SERR# event disable register
- The device-specific P_SERR# status register

This chapter provides detailed information about how PCI 6150 handles errors. It also describes error status reporting and error operation disabling.

10.1 Address Parity Errors

PCI 6150 checks address parity for all transactions on both buses, for all address and all bus commands.
When PCI 6150 detects an address parity error on the primary interface, the following events occur:

- If the parity error response bit is set in the command register, PCI 6150 does not claim the transaction with P_DEVSEL#; this may allow the transaction to terminate in a Master-Abort. If the parity error response bit is not set, PCI 6150 proceeds normally and accepts the transaction if it is directed to or across PCI 6150.
- PCI 6150 sets the detected parity error bit in the status register.
- PCI 6150 asserts P_SERR# and sets the signaled system error bit in the status register, if both of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The parity error response bit is set in the command register.

When PCI 6150 detects an address parity error on the secondary interface, the following events occur:

- If the parity error response bit is set in the bridge control register, PCI 6150 does not claim the transaction with S_DEVSEL#; this may allow the transaction to terminate in a master abort. If the parity error response bit is not set, PCI 6150 proceeds normally and accepts the transaction if it is directed to or across PCI 6150.
- PCI 6150 sets the detected parity error bit in the secondary status register.
- PCI 6150 asserts P_SERR# and sets the signaled system error bit in the status register, if both of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The parity error response bit is set in the bridge control register.

10.2 Data Parity Errors

When forwarding transactions, PCI 6150 attempts to pass the data parity condition from one interface to the other unchanged, whenever possible, to Allow the master and target devices to handle the error condition.
The following sections describe, for each type of transaction, the sequence of events that occurs when a parity error is detected and the way in which the parity condition is forwarded across the bridge.

10.2.1 Configuration Write Transactions to Configuration Space

When PCI 6150 detects a data parity error during a Type-0 configuration write transaction to configuration space, the following events occur:

- If the parity error response bit is set in the command register, PCI 6150 asserts P_TRDY# and writes the data to the configuration register. PCI 6150 also asserts P_PERR#.
  If the parity error response bit is not set, PCI 6150 does not assert P_PERR#.
- PCI 6150 sets the detected parity error bit in the status register, regardless of the state of the parity error response bit.

10.2.2 Read Transactions

When PCI 6150 detects a parity error during a read transaction, the target drives data and data parity, and the initiator checks parity and conditionally asserts PERR#.

For downstream transactions, when PCI 6150 detects a read data parity error on the secondary bus, the following events occur:

- PCI 6150 asserts S_PERR# two cycles following the data transfer, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6150 sets the detected parity error bit in the secondary status register.
- PCI 6150 sets the data parity detected bit in the secondary status register, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6150 forwards the bad parity with the data back to the initiator on the primary bus.
  If the data with the bad parity is prefetched and is not read by the initiator on the primary bus, the data is discarded and the data with bad parity is not returned to the initiator.
- PCI 6150 completes the transaction normally. For upstream transactions, when PCI 6150 detects a read data parity error on the primary bus, the following events occur:
- PCI 6150 asserts P_PERR# two cycles following the data transfer, if the primary interface parity error response bit is set in the command register.
- PCI 6150 sets the detected parity error bit in the primary status register.
- PCI 6150 sets the data parity detected bit in the primary status register, if the primary interface parity error response bit is set in the command register.
- PCI 6150 forwards the bad parity with the data back to the initiator on the secondary bus.
  If the data with the bad parity is prefetched and is not read by the initiator on the secondary bus, the data is discarded and the data with bad parity is not returned to the initiator.
- PCI 6150 completes the transaction normally.

PCI 6150 returns to the initiator the data and parity that was received from the target. When the initiator detects a parity error on this read data and is enabled to report it, the initiator asserts PERR# two cycles after the data transfer occurs. It is assumed that the initiator takes responsibility for handling a parity error condition; therefore, when PCI 6150 detects PERR# asserted while returning read data to the initiator, PCI 6150 does not take any further action and completes the transaction normally.

10.2.3 Delayed Write Transactions

When PCI 6150 detects a data parity error during a delayed write transaction, the initiator drives data and data parity, and the target checks parity and conditionally asserts PERR#.

For delayed write transactions, a parity error can occur at the following times:

- During the original delayed write request transaction
- When the initiator repeats the delayed write request transaction
- When PCI 6150 completes the delayed write transaction to the target

When a delayed write transaction is normally queued, the address, command, address parity, data, byte enable bits, and data parity are all captured and a target retry is returned to the initiator. When PCI 6150 detects a parity error on the write data for the initial delayed write request transaction, the following events occur:

- If the parity error response bit corresponding to the initiator bus is set, PCI 6150 asserts TRDY# to the initiator and the transaction is not queued. If multiple data phases are requested, STOP# is also asserted to cause a target disconnect. Two cycles after the data transfer, PCI 6150 also asserts PERR#.
  If the parity error response bit is not set, PCI 6150 returns a target retry and queues the transaction as usual. Signal PERR# is not asserted. In this case, the initiator repeats the transaction.
- PCI 6150 sets the detected parity error bit in the status register corresponding to the initiator bus, regardless of the state of the parity error response bit.

Note

If parity checking is turned off and data parity errors have occurred for queued or subsequent delayed write transactions on the initiator bus, it is possible that the initiator's reattempts of the write transaction may not match the original queued delayed write information contained in the delayed transaction queue. In this case, a master timeout condition may occur, possibly resulting in a system error (P_SERR# assertion).

For downstream transactions, when PCI 6150 is delivering data to the target on the secondary bus and S_PERR# is asserted by the target, the following events occur:

- PCI 6150 sets the secondary interface data parity detected bit in the secondary status register, if the secondary parity error response bit is set in the bridge control register.
- PCI 6150 captures the parity error condition to forward it back to the initiator on the primary bus.

Similarly, for upstream transactions, when the is delivering data to the target on the primary bus and P_PERR# is asserted by the target, the following events occur:

- PCI 6150 sets the primary interface data parity detected bit in the status register, if the primary parity error response bit is set in the command register.
- PCI 6150 captures the parity error condition to forward it back to the initiator on the secondary bus.

A delayed write transaction is completed on the initiator bus when the initiator repeats the write transaction with the same address, command, data, and byte enable bits as the delayed write command that is at the head of the posted data queue. Note that the parity bit is not compared when determining whether the transaction matches those in the delayed transaction queues.

Two cases must be considered:

- When parity error is detected on the initiator bus on a subsequent reattempt of the transaction and was not detected on the target bus
- When parity error is forwarded back from the target bus
- For downstream delayed write transactions, when the parity error is detected on the initiator bus and PCI 6150 has write status to return, the following events occur:
- PCI 6150 first asserts P_TRDY# and then asserts P_PERR# two cycles later, if the primary interface parity error response bit is set in the command register.
- PCI 6150 sets the primary interface parity error detected bit in the status register.
- Because there was not an exact data and parity match, the write status is not returned and the transaction remains in the queue.

Similarly, for upstream delayed write transactions, when the parity error is detected on the initiator bus and PCI 6150 has write status to return, the following events occur:

- PCI 6150 first asserts S_TRDY# and then asserts S_PERR# two cycles later, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6150 sets the secondary interface parity error detected bit in the secondary status register.
- Because there was not an exact data and parity match, the write status is not returned and the transaction remains in the queue.

For downstream transactions, in the case where the parity error is being passed back from the target bus and the parity error condition was not originally detected on the initiator bus, the following events occur:

- PCI 6150 asserts P_PERR# two cycles after the data transfer, if both of the following are true:
  - The primary interface parity error response bit is set in the command register.
  - The secondary interface parity error response bit is set in the bridge control register.
- PCI 6150 completes the transaction normally.

For upstream transactions, in the case where the parity error is being passed back from the target bus and the parity error condition was not originally detected on the initiator bus, the following events occur:

- PCI 6150 asserts S_PERR# two cycles after the data transfer, if both of the following are true:
  - The primary interface parity error response bit is set in the command register.
  - The secondary interface parity error response bit is set in the bridge control register.
- PCI 6150 completes the transaction normally.

10.2.4 Posted Write Transactions

During downstream posted write transactions, when the , responding as a target, detects a data parity error on the initiator (primary) bus, the following events occur:

- PCI 6150 asserts P_PERR# two cycles after the data transfer, if the primary interface parity error response bit is set in the command register.
- PCI 6150 sets the primary interface parity error detected bit in the status register.
- PCI 6150 captures and forwards the bad parity condition to the secondary bus.
- PCI 6150 completes the transaction normally.

Similarly, during upstream posted write transactions, when PCI 6150, responding as a target, detects a data parity error on the initiator (secondary) bus, the following events occur:

- PCI 6150 asserts S_PERR# two cycles after the data transfer, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6150 sets the secondary interface parity error detected bit in the secondary status register.
- PCI 6150 captures and forwards the bad parity condition to the primary bus.
- PCI 6150 completes the transaction normally.

During downstream write transactions, when a data parity error is reported on the target (secondary) bus by the target's assertion of S_PERR#, the following events occur:

- PCI 6150 sets the data parity detected bit in the secondary status register, if the secondary interface parity error response bit is set in the bridge control register.
- PCI 6150 asserts P_SERR# and sets the signaled system error bit in the status register, if all of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The device-specific P_SERR# disable bit for posted write parity errors is not set.
  - The secondary interface parity error response bit is set in the bridge control register.
  - The primary interface parity error response bit is set in the command register.
  - PCI 6150 did not detect the parity error on the primary (initiator) bus; that is, the parity error was not forwarded from the primary bus.

During upstream write transactions, when a data parity error is reported on the target (primary) bus by the target's assertion of P_PERR#, the following events occur:

- PCI 6150 sets the data parity detected bit in the status register, if the primary interface parity error response bit is set in the command register.
- PCI 6150 asserts P_SERR# and sets the signaled system error bit in the status register, if all of the following conditions are met:
  - The SERR# enable bit is set in the command register.
  - The secondary interface parity error response bit is set in the bridge control register.
  - The primary interface parity error response bit is set in the command register.
  - PCI 6150 did not detect the parity error on the secondary (initiator) bus; that is, the parity error was not forwarded from the secondary bus.

The assertion of P_SERR# is used to signal the parity error condition in the case where the initiator does not know that the error occurred. Because the data has already been delivered with no errors, there is no other way to signal this information back to the initiator.
If the parity error was forwarded from the initiating bus to the target bus, P_SERR# is not asserted.

10.3 Data Parity Error Reporting Summary

In the previous sections, PCI 6150's responses to data parity errors are presented according to the type of transaction in progress. This section organizes PCI 6150's responses to data parity errors according to the status bits that PCI 6150 sets and the signals that it asserts.

Table 10–1 shows setting the detected parity error bit in the status register, corresponding to the primary interface. This bit is set when PCI 6150 detects a parity error on the primary interface.

Table 10–1: Setting the Primary Interface Detected Parity Error Bit

Primary detected parity error bit	Transaction type	Direction	Bus where error was detected	Prim./Sec. parity error response bits
0	Read	Downstream	Primary	x/x[1]
0	Read	Downstream	Secondary	x/x
1	Read	Upstream	Primary	x/x
0	Read	Upstream	Secondary	x/x
1	Posted write	Downstream	Primary	x/x
0	Posted write	Downstream	Secondary	x/x
0	Posted write	Upstream	Primary	x/x
0	Posted write	Upstream	Secondary	x/x
1	Delayed write	Downstream	Primary	x/x
0	Delayed write	Downstream	Secondary	x/x
0	Delayed write	Upstream	Primary	x/x
0	Delayed write	Upstream	Secondary	x/x

[1]x = don't care

Table 10–2 shows setting the detected parity error bit in the secondary status register, corresponding to the secondary interface. This bit is set when PCI 6150 detects a parity error on the secondary interface.

Table 10–2: Setting the Secondary Interface Detected Parity Error Bit

Sec. Detected Parity Error Bit	Transaction Type	Direction	Bus Where Error Was Detected	Prim./Sec. parity error response bits
0	Read	Downstream	Primary	x/x[1]
1	Read	Downstream	Secondary	x/x
0	Read	Upstream	Primary	x/x
0	Read	Upstream	Secondary	x/x
0	Posted write	Downstream	Primary	x/x
0	Posted write	Downstream	Secondary	x/x
0	Posted write	Upstream	Primary	x/x
1	Posted write	Upstream	Secondary	x/x
0	Delayed write	Downstream	Primary	x/x
0	Delayed write	Downstream	Secondary	x/x
0	Delayed write	Upstream	Primary	x/x
1	Delayed write	Upstream	Secondary	x/x

[1]x = don't care

Table 10–3 shows setting the data parity detected bit in the status register, corresponding to the primary interface. This bit is set under the following conditions:

- PCI 6150 must be a master on the primary bus.
- The parity error response bit in the command register, corresponding to the primary interface, must be set.
- The P_PERR# signal is detected asserted or a parity error is detected on the primary bus.

Table 10–3: Setting the Primary Interface Data Parity Detected Bit

Primary data parity detected bit	Transaction type	Direction	Bus where error was detected	Prim./Sec. parity error response bits
0	Read	Downstream	Primary	x/x[1]
0	Read	Downstream	Secondary	x/x
1	Read	Upstream	Primary	1/x
0	Read	Upstream	Secondary	x/x
0	Posted write	Downstream	Primary	x/x
0	Posted write	Downstream	Secondary	x/x
1	Posted write	Upstream	Primary	1/x
0	Posted write	Upstream	Secondary	x/x
0	Delayed write	Downstream	Primary	x/x
0	Delayed write	Downstream	Secondary	x/x
1	Delayed write	Upstream	Primary	1/x
0	Delayed write	Upstream	Secondary	x/x

[1] x = don't care

Table 10–4 shows setting the data parity detected bit in the secondary status register, corresponding to the secondary interface. This bit is set under the following conditions:

- PCI 6150 must be a master on the secondary bus.
- The parity error response bit in the bridge control register, corresponding to the secondary interface, must be set.
- The S_PERR# signal is detected asserted or a parity error is detected on the secondary bus.

Table 10–4: Setting the Secondary Interface Data Parity Detected Bit

Secondary data parity detected bit	Transaction type	Direction	Bus where error was detected	Prim./Sec. parity error response bits
0	Read	Downstream	Primary	x/x[1]
1	Read	Downstream	Secondary	x/1
0	Read	Upstream	Primary	x/x
0	Read	Upstream	Secondary	x/x
0	Posted write	Downstream	Primary	x/x
1	Posted write	Downstream	Secondary	x/1
0	Posted write	Upstream	Primary	x/x
0	Posted write	Upstream	Secondary	x/x
0	Delayed write	Downstream	Primary	x/x
1	Delayed write	Downstream	Secondary	x/1
0	Delayed write	Upstream	Primary	x/x
0	Delayed write	Upstream	Secondary	x/x

[1] x =don't care

Table 10–5 shows assertion of P_PERR#. This signal is set under the following conditions:

- PCI 6150 is either the target of a write transaction or the initiator of a read transaction on the primary bus.
- The parity error response bit in the command register, corresponding to the primary interface, must be set.
- PCI 6150 detects a data parity error on the primary bus or detects S_PERR# asserted during the completion phase of a downstream delayed write transaction on the target (secondary) bus.

Table 10–5: Assertion of P_PERR#

P_PERR#	Transaction type	Direction	Bus where error was detected	Prim./Sec. parity error response bits
1 (deasserted)	Read	Downstream	Primary	x/x[1]
1	Read	Downstream	Secondary	x/x
0 (asserted)	Read	Upstream	Primary	1/x
1	Read	Upstream	Secondary	x/x
0	Posted write	Downstream	Primary	1/x
1	Posted write	Downstream	Secondary	x/x
1	Posted write	Upstream	Primary	x/x
1	Posted write	Upstream	Secondary	x/x
0	Delayed write	Downstream	Primary	1/x
0[2]	Delayed write	Downstream	Secondary	1/1
1	Delayed write	Upstream	Primary	x/x
1	Delayed write	Upstream	Secondary	x/x

[1]x = don't care
[2]The parity error was detected on the target (secondary) bus but not on the initiator (primary) bus.

Table 10–6 shows assertion of S_PERR#. This signal is set under the following conditions:

- PCI 6150 is either the target of a write transaction or the initiator of a read transaction on the secondary bus.
- The parity error response bit in the bridge control register, corresponding to the secondary interface, must be set.
- PCI 6150 detects a data parity error on the secondary bus or detects P_PERR# asserted during the completion phase of an upstream delayed write transaction on the target (primary) bus.

Table 10-6: Assertion of S_PERR#

S_PERR#	Transaction type	Direction	Bus where error was detected	Prim./Sec. parity error response bits
1 (deasserted)	Read	Downstream	Primary	x/x[1]
0 (asserted)	Read	Downstream	Secondary	x/1
1	Read	Upstream	Primary	x/x
1	Read	Upstream	Secondary	x/x
1	Posted write	Downstream	Primary	x/x
1	Posted write	Downstream	Secondary	x/x
1	Posted write	Upstream	Primary	x/x
0	Posted write	Upstream	Secondary	x/1
1	Delayed write	Downstream	Primary	x/x
1	Delayed write	Downstream	Secondary	x/x
0[2]	Delayed write	Upstream	Primary	1/1
0	Delayed write	Upstream	Secondary	x/1

[1]x =don't care
[2]The parity error was detected on the target (secondary) bus but not on the initiator (primary) bus.

Table 10–7 shows assertion of P_SERR#. This signal is set under the following conditions:

- PCI 6150 has detected P_PERR# asserted on an upstream posted write transaction or S_PERR# asserted on a downstream posted write transaction.
- PCI 6150 did not detect the parity error as a target of the posted write transaction.
- The parity error response bit on the command register and the parity error response bit on the bridge control register must both be set.
- The SERR# enable bit must be set in the command register.

Table 10–7: Assertion of P_SERR# for Data Parity Errors

P_PERR#	Transaction Type	Direction	Bus where error was detected	Prim./Sec. parity error response bits
1 (deasserted)	Read	Downstream	Primary	x/x[1]
1	Read	Downstream	Secondary	x/x
1	Read	Upstream	Primary	x/x
1	Read	Upstream	Secondary	x/x
1	Posted write	Downstream	Primary	x/x
0[2] (asserted)	Posted write	Downstream	Secondary	1/1
0[3]	Posted write	Upstream	Primary	1/1
1	Posted write	Upstream	Secondary	x/x
1	Delayed write	Downstream	Primary	x/x
1	Delayed write	Downstream	Secondary	x/x
1	Delayed write	Upstream	Primary	x/x
1	Delayed write	Upstream	Secondary	x/x

[1] x = don't care
[2] The parity error was detected on the target (secondary) bus but not on the initiator (primary) bus.
[3] The parity error was detected on the target (primary) bus but not on the initiator (secondary) bus.

10.4 System Error (SERR#) Reporting

Whenever the assertion of P_SERR# is discussed in this document, it is assumed that the following conditions apply:

- For PCI 6150 to assert P_SERR# for any reason, the SERR# enable bit must be set in the command register.
- Whenever PCI 6150 asserts P_SERR#, PCI 6150 must also set the signaled system error bit in the status register.

In compliance with the PCI-to-PCI Bridge Architecture Specification, PCI 6150 asserts P_SERR# when it detects the secondary SERR# input, S_SERR#, asserted and the SERR# forward enable bit is set in the bridge control register. In addition, PCI 6150 also sets the received system error bit in the secondary status register.
PCI 6150 also conditionally asserts P_SERR# for any of the following reasons:

- Target abort detected during posted write transaction
- Master abort detected during posted write transaction
- Posted write data discarded after 2^{24} attempts to deliver (2^{24} target retries received)
- Parity error reported on target bus during posted write transaction (see previous section)
- Delayed write data discarded after 2^{24} attempts to deliver (2^{24} target retries received)
- Delayed read data cannot be transferred from target after 2^{24} attempts (2^{24} target retries received)
- Master timeout on delayed transaction

The device-specific P_SERR# status register reports the reason for the assertion of P_SERR#.
Most of these events have additional device-specific disable bits in the P_SERR# event disable register that make it possible to mask out P_SERR# assertion for specific events. The master timeout condition has a SERR# enable bit for that event in the bridge control register and therefore does not have a device-specific disable bit.

11 Exclusive Access

This chapter describes the use of the LOCK# signal to implement exclusive access to a target for transactions that cross PCI 6150. **Current revision of the PCI 6150 does not support LOCKed BURST READ cycles**.

11.1 Concurrent Locks

The primary and secondary bus lock mechanisms operate concurrently except when a locked transaction crosses PCI 6150. A primary master can lock a primary target without affecting the status of the lock on the secondary bus, and vice versa. This means that a primary master can lock a primary target at the same time that a secondary master locks a secondary target.

11.2 Acquiring Exclusive Access Across PCI 6150

For any PCI bus, before acquiring access to the LOCK# signal and starting a series of locked transactions, the initiator must first check that both of the following conditions are met:

- The PCI bus must be idle.
- The LOCK# signal must be deasserted.

The initiator leaves the LOCK# signal deasserted during the address phase and asserts LOCK# one clock cycle later. Once a data transfer is completed from the target, the target lock has been achieved.
Locked transactions can cross PCI 6150 in the downstream and upstream directions, from the primary bus to the secondary bus and vice versa.

When the target resides on another PCI bus, the master must acquire not only the lock on its own PCI bus but also the lock on every bus between its bus and the target's bus. When PCI 6150 detects, on the primary bus, an initial locked transaction intended for a target on the secondary bus, PCI 6150 samples the address, transaction type, byte enable bits, and parity. It also samples the lock signal. Because a target retry is signaled to the initiator, the initiator must relinquish the lock on the primary bus, and therefore the lock is not yet established.

The first locked transaction must be a read transaction. Subsequent locked transactions can be read or write transactions. Posted Memory Write transactions that are a part of the locked transaction sequence are still posted. Memory read transactions that are a part of the locked transaction sequence are not prefetched.

When the locked delayed read request is queued, PCI 6150 does not queue any more transactions until the locked sequence is finished. PCI 6150 signals a target retry to all transactions initiated subsequent to the locked read transaction that are intended for targets on the other side of PCI 6150. PCI 6150 allows any transactions queued before the locked transaction to complete before initiating the locked transaction.

When the locked delayed read request transaction moves to the head of the delayed transaction queue, PCI 6150 initiates the transaction as a locked read transaction by deasserting LOCK# on the target bus during the first address phase, and by asserting LOCK# one cycle later. If LOCK# is already asserted (used by another initiator), PCI 6150 waits to request access to the secondary bus until LOCK# is sampled deasserted when the target bus is idle. Note that the existing lock on the target bus could not have crossed PCI 6150; otherwise, the pending queued locked transaction would not have been queued. When PCI 6150 is able to complete a data transfer with the locked read transaction, the lock is established on the secondary bus.

When the initiator repeats the locked read transaction on the primary bus with the same address, transaction type, and byte enable bits, PCI 6150 transfers the read data back to the initiator, and the lock is then also established on the primary bus.

For PCI 6150 to recognize and respond to the initiator, the initiator's subsequent attempts of the read transaction must use the locked transaction sequence (deassert LOCK# during address phase, and assert LOCK# one cycle later). If the LOCK# sequence is not used in subsequent attempts, a master timeout condition may result. When a master timeout condition occurs, SERR# is conditionally asserted, the read data and queued read transaction are discarded, and the LOCK# signal is deasserted on the target bus.

Once the intended target has been locked, any subsequent locked transactions initiated on the initiator bus that are forwarded by PCI 6150 are driven as locked transactions on the target bus.

When PCI 6150 receives a target abort or a master abort in response to the delayed locked read transaction, this status is passed back to the initiator, and no locks are established on either the target or the initiator bus. PCI 6150 resumes forwarding unlocked transactions in both directions.

11.3 Ending Exclusive Access

After the lock has been acquired on both the initiator and target buses, PCI 6150 must maintain the lock on the target bus for any subsequent locked transactions until the initiator relinquishes the lock.

The only time a target retry causes the lock to be relinquished is on the first transaction of a locked sequence. On subsequent transactions in the sequence, the target retry has no effect on the status of the lock signal.

An established target lock is maintained until the initiator relinquishes the lock. PCI 6150 does not know whether the current transaction is the last one in a sequence of locked transactions until the initiator deasserts the LOCK# signal at the end of the transaction.

When the last locked transaction is a delayed transaction, PCI 6150 has already completed the transaction on the secondary bus. In this case, as soon as PCI 6150 detects that the initiator has relinquished the LOCK# signal by sampling it in the deasserted state while FRAME# is deasserted, PCI 6150 deasserts the LOCK# signal on the target bus as soon as possible. Because of this behavior, LOCK# may not be deasserted until several cycles after the last locked transaction has been completed on the target bus. As soon as PCI 6150 has deasserted LOCK# to indicate the end of a sequence of locked transactions, it resumes forwarding unlocked transactions.

When the last locked transaction is a posted write transaction, PCI 6150 deasserts LOCK# on the target bus at the end of the transaction because the lock was relinquished at the end of the write transaction on the initiator bus.

When PCI 6150 receives a target abort or a master abort in response to a locked delayed transaction, PCI 6150 returns a target abort or a master abort when the initiator repeats the locked transaction. The initiator must then deassert LOCK# at the end of the transaction. PCI 6150 sets the appropriate status bits, flagging the abnormal target termination condition. Normal forwarding of unlocked posted and delayed transactions is resumed.

When PCI 6150 receives a target abort or a master abort in response to a locked posted write transaction, PCI 6150 cannot pass back that status to the initiator. PCI 6150 asserts SERR# on the initiator bus when a target abort or a master abort is received during a locked posted write transaction, if the SERR# enable bit is set in the command register. Signal SERR# is asserted for the master abort condition if the master abort mode bit is set in the bridge control register.

Note:
PCI 6150 has an option to ignore the lock protocol, through register 46h, bits 13 and 14.

12 PCI Bus Arbitration

PCI 6150 must arbitrate for use of the primary bus when forwarding upstream transactions, and for use of the secondary bus when forwarding downstream transactions. The arbiter for the primary bus resides external to PCI 6150, typically on the motherboard. For the secondary PCI bus, PCI 6150 implements an internal arbiter. This arbiter can be disabled, and an external arbiter can be used instead.

This chapter describes primary and secondary bus arbitration.

12.1 Primary PCI Bus Arbitration

PCI 6150 implements a request output pin, P_REQ#, and a grant input pin, P_GNT#, for primary PCI bus arbitration. PCI 6150 asserts P_REQ# when forwarding transactions upstream; that is, it acts as initiator on the primary PCI bus. As long as at least one pending transaction resides in the queues in the upstream direction, either posted write data or delayed transaction requests, PCI 6150 keeps P_REQ# asserted. However, if a target retry, target disconnect, or a target abort is received in response to a transaction initiated by PCI 6150 on the primary PCI bus, PCI 6150 deasserts P_REQ# for two PCI clock cycles. For all cycles through the bridge, P_REQ# is not asserted until the transaction request has been completely queued.

If P_GNT# is asserted (by the primary bus arbiter after PCI 6150 has asserted P_REQ#), PCI 6150 initiates a transaction on the primary bus during the next PCI clock cycle. When P_GNT# is asserted to PCI 6150 when P_REQ# is not asserted, PCI 6150 parks P_AD, P_CBE, and P_PAR by driving them to valid logic levels. When the primary bus is parked at PCI 6150 and PCI 6150 then has a transaction to initiate on the primary bus, PCI 6150 starts the transaction if P_GNT# was asserted during the previous cycle.

12.2 Secondary PCI Bus Arbitration

PCI 6150 implements an internal secondary PCI bus arbiter. This arbiter supports nine external masters in addition to PCI 6150. The internal arbiter can be disabled, and an external arbiter can be used instead for secondary bus arbitration.

12.2.1 Secondary Bus Arbitration Using the Internal Arbiter

To use the internal arbiter, the secondary bus arbiter enable pin, S_CFN#, must be tied LOW. PCI 6150 has nine secondary bus request input pins, S_REQ#[8:0], and nine secondary bus output grant pins, S_GNT#[8:0], to support external secondary bus masters. The secondary bus request and grant signals are connected internally to the arbiter and are not brought out to external pins when S_CFN# is HIGH.

PCI 6150 has a 2-level arbitration scheme in which arbitration is divided into two groups, a high priority group and a low priority group. The low priority group as a whole represents one entry in the high priority group; that is, if the high priority group consists of n masters, then in at least every n+1 transactions the highest priority is assigned to the low priority group. Priority changes evenly among the low priority group. Therefore, members of the high priority group can be serviced n transactions out of n+1, while one member of the low priority group is serviced once every n+1 transactions. Each master can be assigned to either the low priority group or the high priority group, through configuration register 42h.

Each group can be programmed to use a rotating priority or a fixed priority scheme, through configuration register 50h.

12.2.1.1 Rotating Priority Scheme

The secondary arbiter supports a programmable 2-level rotating algorithm that takes care of 10 requests/grants, including the PCI 6150. Figure 12–1 shows an example of an internal arbiter where four masters, including PCI 6150, are in the high priority group, and six masters are in the low priority group. Using this example, if all requests are always asserted, the highest priority rotates among the masters in the following fashion (high priority members are given in italics, low priority members, in boldface type):

B, m0, m1, m2, B, m0, m1, m3, B, m0, m1, m4, B, m0, m1, m5, and so on.

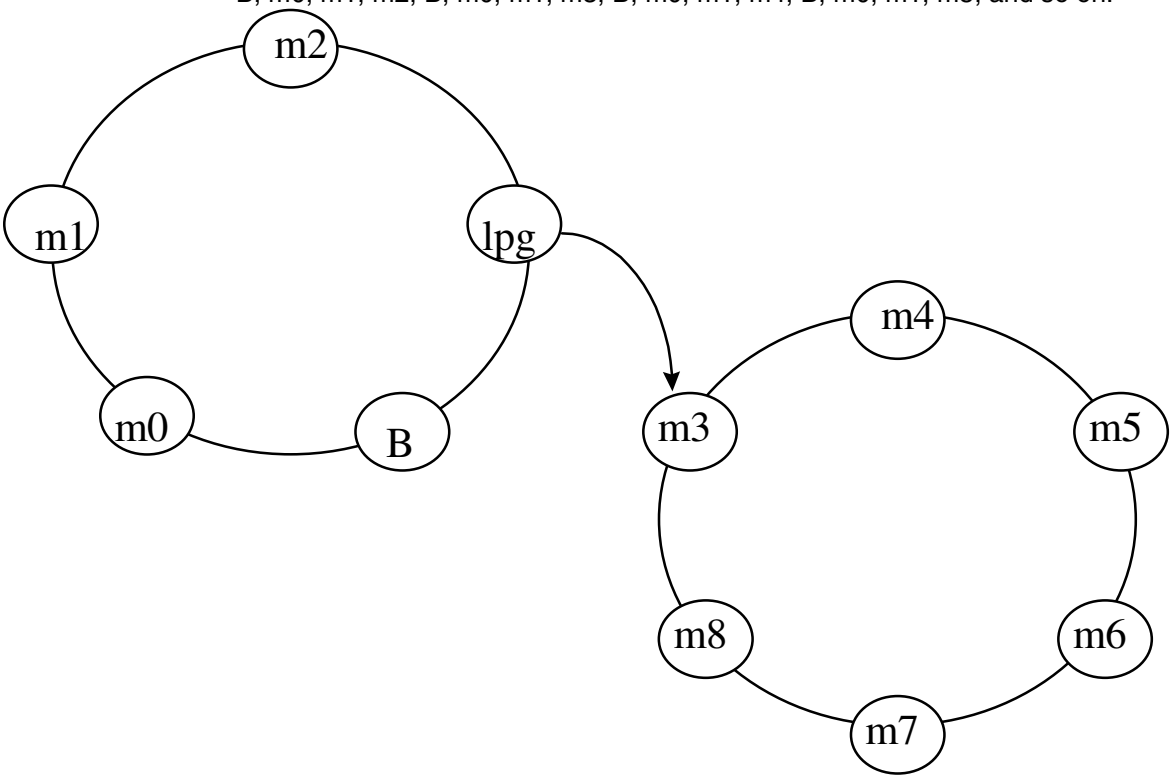


Figure 12-1 Secondary Arbiter Example

If all the masters are assigned to one group, the algorithm defaults to a rotating priority among all the masters. After reset, all external masters are assigned to the low priority group, and PCI 6150 is assigned to the high priority group. PCI 6150 receives highest priority on the target bus every other transaction, and priority rotates evenly among the other masters.

Priorities are reevaluated every time S_FRAME# is asserted, that is, at the start of each new transaction on the secondary PCI bus. From this point until the time that the next transaction starts, the arbiter asserts the grant signal corresponding to the highest priority request that is asserted. If a grant for a particular request is asserted, and a higher priority request subsequently asserts, the arbiter deasserts the asserted grant signal and asserts the grant corresponding to the new higher priority request on the next PCI clock cycle. When priorities are reevaluated, the highest priority is assigned to the next highest priority master relative to the master that initiated the previous transaction. The master that initiated the last transaction now has the lowest priority in its group. Priority is also re-evaluated if the requesting agent deasserts its request without generating any cycles while it was granted.

If PCI 6150 detects that an initiator has failed to assert S_FRAME# after 16 cycles of both grant assertion and a secondary idle bus condition, the arbiter will re-evaluate grant assignment.

12.2.1.2 Fixed Priority Scheme

PCI 6150 also has support for a fixed priority scheme within the two priority groups. In this case, configuration register 50h, bit 0 and 2 controls whether the low priority group or the high priority group will use fixed or rotating priority scheme. If a fixed priority scheme is used, then a master within the group is assigned to have the highest priority within its group, and then an option is set to control the priority of other masters relative to the highest priority master. This is controlled through bits 4-11 and bits 1 and 3 of the Internal Arbiter Control register (reg. 50h). For example, using the previous example with the groups at fixed priority, if master 7 has the highest priority of the low priority group, and PCI 6150 has the highest priority of the high priority group, and priority decreases in ascending order of masters for both groups, (bits 1,3 of register 50 are set to 1). The order of priority with the highest first, will be as follows:

B, m0, m1, m2, m7, m8, m3, m4, m5, m6

If bits 1, 3 or register 50 are set to 0, priority will decrease in the other direction , so the order becomes:

B, m2, m1, m0, m7, m6, m5, m4, m3, m8

To prevent bus contention, if the secondary PCI bus is idle, the arbiter never asserts one grant signal in the same PCI cycle in which it deasserts another. It deasserts one grant, and then asserts the next grant, no earlier than one PCI clock cycle later. If the secondary PCI bus is busy, that is, either S_FRAME# or S_IRDY# is asserted, the arbiter can deassert one grant and assert another grant during the same PCI clock cycle.

12.2.2 Secondary Bus Arbitration using an External Arbiter

The internal arbiter is disabled when the secondary bus central function control pin, S_CFN#, is tied HIGH. An external arbiter must then be used.

When S_CFN# is tied HIGH, PCI 6150 reconfigures two pins to be external request and grant pins. The S_GNT#[0] pin is reconfigured to be the PCI 6150's external request pin because it is output. The S_REQ#[0] pin is reconfigured to be the external grant pin because it is input. When an external arbiter is used, PCI 6150 uses the S_GNT#[0] pin to request the secondary bus. When the reconfigured S_REQ#[0] pin is asserted LOW after PCI 6150 has asserted S_GNT#[0], PCI 6150 initiates a transaction on the secondary bus one cycle later. If grant is asserted and PCI 6150 has not asserted the request, PCI 6150 parks the AD, CBE and PAR pins by driving them to valid logic levels.

The unused secondary bus grant outputs, S_GNT#<8:1> are driven HIGH. Unused secondary bus request inputs, S_REQ#<8:1> should be pulled HIGH.

12.2.3 Internal Arbitration Parking

If the internal secondary bus arbiter is enabled, the secondary arbiter can be optionally parked at the last active slot, or on any of the designated slots, and it can also be disabled.

PCI 6150 has the following options related to arbitration parking:

- No parking: all grants are deasserted if there are no requests asserted.
- Fixed parking: grant can be assigned to a specified master.
- Last master granted: grant is assign to the last granted master.

These options are selected through Internal Arbiter Control register at 50h, bits 12-15.

13 General Purpose I/O Interface

The PCI 6150 implements a 4 general-purpose I/O (GPIO) interface pins. During normal operation, the configuration registers control GPIO interface. In addition, the GPIO pins can be used for the following functions:

- During Secondary Reset, the GPIO interface can be used to shift in a 16-bit serial stream that serves as a secondary bus clock disable mask.
- A live insertion bit can be used, along with the GPIO[3] pin, to bring the PCI 6150 gracefully to a halt through hardware, permitting live insertion of option cards behind the PCI 6150.

Table 13-1: GPIO Pin Alternate Functions

GPIO Pin	Alternate Function
GPIO0 – pull-up	Functions as shift register clock output when PRST# asserted.
GPIO1 – pull-up	
GPIO2 – pull-up	Functions as shift/load control output to shift register when PRST# asserted
GPIO3 – pull-up	Can be used for live insertion function, if set as input and live insertion mode bit is set. If high, transaction forwarding is disabled.

13.1 GPIO Control Registers

During normal operation, the GPIO interface is controlled by the following configuration registers:

- The GPIO output data register
- The GPIO output enable control register
- The GPIO input data register

GPIO3-0 consist of five 8-bit fields:

- Write-1-to-set output data field
- Write-1-to-clear output data field
- Write-1-to-set signal output enable control field
- Write-1-to-clear signal output enable control field
- Input data field

The output enable fields control whether each GPIO signal is input or output. Each signal is controlled independently by a bit in each output enable control field. If a 1 is written to the write-1-to-set field, the corresponding pin is activated as an output. If a 1 is written to the write-1-to-clear field, the output driver is three-stated, and the pin is then input only. Writing zeros to these registers has no effect. The reset state for these signals is input only. The input data field is read only and reflects the current value of the GPIO pins. A Type-0 configuration read operation to this address is used to obtain the values of these pins. All pins can be read at any time, whether configured as input only or as bi-directional. The output data fields also use the write-1-to-set and write-1-to-clear method. If a 1 is written to the write-1-to-set field and the pin is enabled as an output, the corresponding GPIO output is driven high. If a 1 is written to the write-1-to-clear field and the pin is enabled as an output, the corresponding GPIO output is driven low. Writing zeros to these registers has no effect. The value written to the output register will be driven only when the GPIO signal is configured as output. Type-0 configuration write operation is used to program these fields. The reset value for the output is 0.

14 Clocks

This chapter provides information about the clocks.

14.1 Primary and Secondary Clock Inputs

PCI 6150 implements a separate clock input for each PCI interface. The primary interface is synchronized to the primary clock input, P_CLKIN, and the secondary interface is synchronized to the secondary clock input, S_CLKIN.

PCI 6150 operates at a maximum frequency of 66 MHz. Output clocks S_CLK[9:0] can be derived from P_CLKIN either at the same frequency, at P_CLKIN / 2, or at an asynchronous frequency.

PCI 6150 primary and secondary clock inputs can be asynchronous. The skew between P_CLKIN and S_CLKIN is irrelevant. PCI 6150 can tolerate a 1:2.5 or 2.5:1 frequency ratio between primary and secondary clock inputs.

14.2 Secondary Clock Outputs

PCI 6150 has 10 secondary clock outputs that can be used as clock inputs for up to nine external secondary bus devices with one feedback to S_CLKIN.
The rules for using secondary clocks are:

- Each secondary clock output is limited to no more than 1 PCI load.
- All clock trace length, including feedback clock to the PCI 6150, must have equal length and impedance.
- Terminating or disabling unused secondary clock outputs is recommended to reduce power dissipation and noise in the system.

14.3 Disabling Unused Secondary Clock Outputs

When secondary clock outputs are not used, both GPIO[3:0] and MSK_IN can be used to clock in a serial mask that selectively three-states secondary clock outputs. See GPIO section for details in this application.

After the serial mask has been shifted into the PCI 6150, the value of the mask is readable and can be changed in the secondary clock disable mask register. When the mask is modified by a configuration write operation to this register, the new clock mask disables the appropriate secondary clock outputs within a few cycles. This feature allows software to disable or enable secondary clock outputs based on the presence of option cards, and so on.
PCI 6150 delays deasserting the secondary reset signal, S_RSTOUT#, until the serial clock mask has been completely shifted in and the secondary clocks have been disabled or enabled, according to the mask. The delay between P_RSTIN# assertion and S_RSOUT# deassertion is 16-32 clocks.

14.3.1 Secondary Clock Control

The PCI 6150 uses the GPIO pins and the MSKIN signal to input a 16-bit serial data stream. This data stream is shifted into the secondary clock control register and is used for selectively disabling secondary clock outputs. The serial data stream is shifted in as soon as P_RSTIN# is detected deasserted and the secondary reset signal, S_RSTOUT#, is detected asserted. The deassertion of S_RSTOUT# is delayed until the PCI 6150 completes shifting in the clock mask data, which takes 16 clock cycles (32 cycles if operating at 66 MHz). After that, the GPIO pins can be used as general purpose I/O pins.

An external shift register should be used to load and shift the data. The GPIO pins are used for shift register control and serial data input. The data is input through the dedicated input signal, MSKIN. The shift register circuitry is not necessary for correct operation of the PCI 6150. The shift registers can be eliminated, and MSKIN can be tied low to enable all secondary clock outputs or tied high to force all secondary clock outputs high.

GPIO Shift Register Operation

GPIO Pin	Operation
GPIO[0]	Shift register clock output at 33 MHz maximum frequency.
GPIO[2]	0 1 Load Shift Shift Register control:

GPIO Serial Data Format

Bit Description	SCLK_O OUTPUT
[1:0] Slot 0 PRSNT#<1:0> or device 0	0
[3:2] Slot 1 PRSNT#<1:0> or device 1	1
[5:4] Slot 2 PRSNT#<1:0> or device 2	2
[7:6] Slot 3 PRSNT#<1:0> or device 3	3
[8] Device 4	4
[9] Device 5	5
[10] Device 6	6
[11] Device 7	7
[12] Device 8	8
[13] Can be used for PCI 6150 S_CLKIN input	9
[14] Reserved	Not applicable
[15] Reserved	Not applicable

The first eight bits contain the PRSNT#[1:0] signal values for four slots, and these bits control the SCLKO[3:0] outputs. If one or both of the PRSNT#[1:0] signals are 0, that indicates that a card is present in the slot and therefore the secondary clock for that slot is not masked. If these clocks are connected to devices and not to slots, one or both of the bits should be tied low to enable the clock. The next five bits are the clock mask for devices; each bit enables or disables the clock for one device. These bits control the SCLKO[8:4] outputs: 0 enables the clock, and 1 disables the clock. Bit 13 is the clock enable bit for SCLKO[9], which is connected to the PCI 6150's SCLKIN input. If desired, the assignment of SCLKO clock outputs to slots, devices, and the PCI 6150's SCLKIN input can be rearranged from the assignment shown here. However, it is important that the serial data stream format match the assignment of SCLKO outputs. The GPIO pin serial protocol is designed to work with two 74F166 8-bit shift registers.

14.4 Using an External Clock Source

PCI 6150 has 2 signals, OSC_SEL# and OSC_IN which can be used to connect an external clock source to the PCI 6150. During normal operation, PCI 6150 generates S_CLK[9:0] outputs based on the PCI CLK source. If OSCSEL# is low, PCI 6150 will derive S_CLK[9:0] outputs from the OSCIN signal instead. Clock division will still be performed on the OSCIN clock depending on the states of the P_M66EN and S_M66EN signals.

PCI 6150 also has S_CLK_STB input allowing designer to indicate the Secondary external clock source is stable. If this input is 0 indicating that the S_CLKIN is not yet stable, the PCI 6150 will not let S_RSTOUT# deassert.

14.5 Frequency Division options

The PCI 6150 has built-in frequency division options to automatically adjust the output clocks S_CLK[9:0] for 66Mhz or 33Mhz operations. The following table depicts division conditions.

P_M66EN	S_M66EN	DIVISION
1	1	1/1
1	0	1/2
0	1	1/1
0	0	1/1

Notes
1. S_M66EN pin cannot be floating.
2. In Revision BA, the PCI 6154 does not drive S_M66EN to low when P_M66EN is low.
3. In Revision BB, the PCI 6154 drives S_M66EN to low when P_M66EN is low.

14.6 Running Secondary Port Faster than Primary Port

PCI 6150 allows running the Secondary port at a faster rate than the Primary port. PCI 6150 asynchronous design supports standard 66Mhz to 33MHz and faster secondary port speed such as 33Mhz to 66MHz conversion. Designers must provide the faster clock source either through an oscillator or a clock generator. If OSCIN is used to make use of the SCLKn outputs, the division control can be disabled by pulling the S_M66EN pin HIGH and not connecting this pin to the PCI slots. Otherwise external clock can be fed directly into the S_CLKIN.

15 Frequency Operation

PCI 6150 supports up to 66 MHz operations. CFG66 and PM66EN pin inputs should be HIGH for 66MHz operation.

15.1 66-MHZ operation

Signal CFG66 must be tied high on the board to enable 66MHz operation and to set the 66MHz Capable bit in the Status register and the Secondary Status register in configuration space.

Signals P_M66EN and S_M66EN indicate whether the primary and secondary interfaces, respectively are operating at 66MHz. This information is needed to control the frequency of the secondary bus. Note that the PCI Local Bus Specification, Revision 2.2 restricts clock frequency changes above 33MHz to during PCI reset only.

The following primary and secondary bus frequency combinations are supported when using the Primary P_CLKIN to generate the secondary clock outputs:

- 66MHz primary bus, 66MHz secondary bus
- 66MHz primary bus, 33MHz secondary bus
- 33MHz primary bus, 33MHz secondary bus

If P_M66EN is low (66MHz capable, primary bus at 33MHz), then the PCI 6150 drives LOW S_M66EN to indicate that the secondary bus is operating at 33MHz. If Designers want to run the secondary bus at faster than primary bus, S_M66EN need not be connected to secondary PCI devices.

PCI 6150 can also generate S_CLK outputs from the OSCIN input if enabled. When PCI 6150 is running with external clock input that is not generated from S_CLK[9:0] outputs, the P_M66EN and S_M66EN controlled clock division will not apply.

When OSCIN or other external clock inputs are used for the secondary port, PCI 6150 can run with a maximum ratio of 1:2.5 or 2.5:1 between primary and secondary bus clocks.

16 Reset

This chapter describes the primary interface, secondary interface, and chip reset mechanisms. There are also Power Management initiated internal reset and software controlled internal reset.

Note:
After any of the resets are deasserted, PCI 6150 requires 32 clocks to initialize the bridge functions.

16.1 Primary Reset Input

PCI 6150 requires at least 2 clocks before P_RSTIN# rising edge to reset properly.

When P_RSTIN# is asserted, the following events occur:

- PCI 6150 three-states all primary PCI interface signals, except S_RSTOUT#, S_GNT#[8:0], SCLKO[9:0] within 1 clock.
- Active P_RSTIN# will cause Secondary port reset. 43 clocks after P_RSTIN# going HIGH, S_RSTOUT# will go HIGH.
- Clock disable bits begin to be shifted in at the rising edge of P_RSTIN#.

The P_RSTIN# asserting and deasserting edges can be asynchronous to P_CLK and S_CLK. PCI 6150 requires 32 primary port PCI clocks after P_RSTIN# rising edge to reset its internal logic.

When P_RSTIN# is asserted, all posted write and delayed transaction data buffers are reset; therefore, any transactions residing in buffers at the active time of P_RSTIN# are discarded.

16.2 Secondary Reset Output

PCI 6150 asserts S_RSTOUT# when any of the following conditions is met:

- Signal P_RSTIN# is asserted.
  Signal S_RSTOUT# remains asserted as long as P_RSTIN# is asserted and does not deassert until P_RSTIN# is deasserted, or while the secondary clock serial disable mask is being shifted in (16 clock cycles after P_RSTIN# deassertion).
- The Secondary Reset bit in the bridge control register is set. Signal S_RSTOUT# remains asserted until the rest control bit is cleared.

16.3 Software Chip Reset

The chip reset bit (register 41h, bit 0) in the diagnostic control register can be used to reset the entire PCI 6150 except that it will NOT cause S_RSTOUT# to go active and it will NOT three-states the signals.

When the chip reset bit is set, all registers and chip state are reset. Within 4 PCI clock cycles after completion of the configuration write operation that sets the chip reset bit, the chip reset bit automatically clears and the chip is ready for configuration.

During chip reset, PCI 6150 is inaccessible.

16.4 Power Management internal Reset

When there is a D3hot to D0 transition with Power Management Control registers bit 1:0 programmed to D0, an internal reset equivalent to P_RSTIN# input will be generated and all relevant registers will be reset. However S_RSTOUT# will NOT be active.

16.5 Reset inputs Table

The following table depicts the effect of different PCI 6150 RESET INPUT sources.

Operating Mode **Reset Inputs**	**Transparent Mode**
P_RSTIN#	**- Resets Primary and Secondary Ports** **- Causes S_RSTOUT# active** **- Causes EEPROM load**
S_RSTOUT#	**- Not used as input**
Register 41h bit 0 Chip Reset	**- Resets internal state machines**
Register 3Eh bit 6 Secondary Reset	**- Resets only Secondary Port** **- Causes S_RSTOUT# active**

17 Bridge Behavior

A PCI cycle is initiated by asserting the FRAME# signal. In a bridge, there are a number of possibilities. These are summarized in the table below. Table 17-1 shows PCI 6150's actions for different cycle types.

Table 17-1: Bridge Actions for Various Cycle Types

Initiator	Target	Response
Master on primary	Target on Primary	PCI 6150 does not respond. It detects this situation by decoding the address as well as monitoring the P_DEVSEL# for other fast and medium devices on the primary port.
Master on primary	Target on secondary	PCI 6150 asserts P_DEVSEL#, terminate the cycle normally if able to posted, otherwise return with a retry. Then passes the cycle to the appropriate port. When cycle is complete on the target port, it will wait for the initiator to repeat the same cycle and end with normal termination.
Master on primary	Target not on primary nor secondary port	PCI 6150 does not respond and the cycle will terminate as master abort.
Master on secondary	Target on the same secondary port	PCI 6150 does not respond.
Master on secondary	Target on primary or the other secondary port	PCI 6150 asserts S_DEVSEL#, terminate the cycle normally if able to posted, otherwise return with a retry. Then passes the cycle to the appropriate port. When cycle is complete on the target port, it will wait for the initiator to repeat the same cycle and end with normal termination.
Master on secondary	Target not on primary nor the other secondary	PCI 6150 does not respond.

A target then has up to three cycles to respond before subtractive decoding is initiated. If the target detects an address hit, it should assert its DEVSEL# signal in the cycle corresponding to the values of bits 9 and 10 in the Configuration Status Register.

Termination of a PCI cycle can occur in a number of ways. Normal termination begins by the initiator (master) deasserting FRAME# with IRDY# being asserted (or remaining asserted) on the same cycle. The cycle completes when TRDY# and IRDY# are both asserted simultaneously. The target should deassert TRDY# for one cycle following final assertion (sustained three-state signal).

17.1 Abnormal Termination (Initiated by Bridge Master)

17.1.1 Master Abort

Master abort indicates that PCI 6150 acting as a master receives no response (i.e., no target asserts P_DEVSEL# or S_DEVSEL#) from a target. the bridge deasserts FRAME# and then deasserts IRDY#.

17.2 Parity and Error Reporting

Parity must be checked for all addresses and write data. Parity is defined on the P_PAR and S_PAR signals. Parity should be even (i.e. an even number of '1's) across AD, CBE, and PAR. Parity information on PAR is valid the cycle after AD and CBE are valid.

17.2.1 Reporting Parity Errors

For all address phases, if a parity error is detected, the error is reported on the P_SERR# signal by asserting P_SERR# for one cycle and then three-stating two cycles after the bad address. P_SERR# can only be asserted if bit 6 and 8 in the Command Register are both set to 1. For write data phases, a parity error is reported by asserting the P_PERR# signal two cycles after the data phase and should remain asserted for one cycle when bit 8 in the Command register is set to a 1. The target reports any type of data parity errors during write cycles, while the master reports data parity errors during read cycles.

Detection of an address parity error will cause the PCI bridge target to not claim the bus (P_DEVSEL# remains inactive) and the cycle will then terminate with a Master Abort. When the bridge is acting as master, a data parity error during a read cycle results in the bridge master initiating a Master Abort.

17.3 Secondary IDSEL Mapping

When PCI 6150 detects a Type-1 configuration transaction for a device connected to the secondary, it translates the Type-1 transaction to Type-0 transaction on the downstream interface. Type-1 configuration format uses a 5-bit field at P_AD[15:11] as a device number. This is translated to S_AD[31:16] by PCI 6150. Table 17-2 explains how PCI 6150 generates the secondary IDSELs. No device is permitted to connect IDSEL to AD[16] (the source bridge is device number 0). PCI 6150 is the source bridge for its secondary bus.

Table 17-2: Generation of S_IDSEL.

P_AD[15:11]	S_AD[31:16]	Device Number	S_AD bit
0 0000b	0000 0000 0000 0001b	0 (Source Bridge)	16
0 0001b	0000 0000 0000 0010b	1	17
0 0010b	0000 0000 0000 0100b	2	18
…	…	…	…
0 1011b	0000 1000 0000 0000b	11	27
0 1100b	0001 0000 0000 0000b	12	28
…	…	…	…
0 1111b	1000 0000 0000 0000b	15	31
1 xxxxb	0000 0000 0000 0000b	Special Cycle	N/A

18 Flow Through Optimization

PCI 6150 has several options that can be used to optimize PCI transfers once flow though mode through the bridge is achieved.

During flow through posted write cycles, if there is only 1 data transfer pending in the internal post memory write queue, PCI 6150 can be programmed to wait for a specified number of clocks before disconnecting. PCI 6150 will deassert IRDY# on the target side and wait for up to 7 clocks for more data from the initiator. If during this period new write data is received from the initiator, PCI 6150 will reassert IRDY# and continue with the write cycle. If no new write data is received during this period, the PCI 6150 will terminate the cycle to the target with the last data from the queue and finish the cycle at a later time.

For flow through delayed read cycles, if the internal read queue is almost full, PCI 6150 can be programmed to wait for a specified number of clocks for read data from the target before disconnecting. During this time, the PCI 6150 will deassert IRDY# from the read source and wait for up to 7 clocks. If additional space becomes available in the read queue before the end of IRDY# inactive period, PCI 6150 will reassert IRDY# and proceed with the next read data phase. If no additional space becomes available in the read queue, the current data phase will become the last one (IRDY# is asserted, FRAME# is deasserted) and the cycle will terminate at the end of the data phase.

18.1 Read Cycle Optimization

The main function is to increase the probability of flow-through occurring during read access to prefetchable memory regions. In the case that flow-through does not occur, it would be inefficient for the PCI 6150 to prefetch too little or too much data. If PCI 6150 prefetches too little and flow-through does not occur, then the read cycles become divided into multiple cycles. If PCI 6150 prefetches too much data and the internal FIFOs fill, it has to wait for the initiator to retry the previous read cycle, and then flush unclaimed data before it can enqueue subsequent cycles. The prefetch count registers can be used to tune the PCI 6150 for various PCI masters.

The initial count is equivalent to the cache line size, and assumes that a master will want at least 1 cache line of data. The incremental count is only used when PCI 6150 does not detect flow-through for the current cycle being prefetched during the initial fetch count. PCI 6150 will continue prefetching in increments until the maximum count is reached, then disconnect the cycle.

For read prefetching, the PCI 6150 implements several registers that control the amount of data prefetched on the secondary and primary PCI bus. The following registers can be used to optimize PCI 6150 performance during read cycles:

- Primary Initial Prefetch Maximum Count
- Primary Incremental Prefetch Count
- Primary Maximum Prefetch Count
- Secondary Initial Prefetch Maximum Count
- Secondary Incremental Prefetch Count
- Secondary Maximum Prefetch Count

The PCI 6150 will prefetch either until flow-through or until prefetch must stop based on the following conditions:

(IPMC + IPC + IPC + … + IPC) < MPC where IPC < ½ MPC

IPMC = Initial Prefetch Maximum Count
IPC = Incremental Prefetch Count
MPC = Maximum Prefetch Count

If the prefetch count has not reached MPC and flow through has been achieved, the PCI 6150 will keep on prefetching until the requesting master terminates the prefetch request. Otherwise, when MPC has been reached, the PCI 6150 will not prefetch any more.

The incremental prefetch can be disabled by setting IPC >= MPC.

18.1.1 Primary/Secondary Initial Prefetch Count

This count controls the amount of data initially prefetched by the PCI 6150 on the primary/secondary bus during reads to the prefetchable memory region. This assumes there is enough space in the internal FIFO. If flow through is achieved during this initial prefetch, PCI 6150 will continue prefetching beyond this count.

18.1.2 Primary/Secondary Incremental Prefetch Count

This register controls the amount of prefetching done after the initial prefetch. If flow-through was not achieved during the initial prefetch, PCI 6150 will try to prefetch more data, until the FIFO fills, or until the maximum prefetch count is reached.

18.1.3 Primary/Secondary Maximum Prefetch Count

This register limits the amount of prefetched data for a single entry available in the internal FIFO at any time. During subsequent read prefetch cycles, the PCI 6150 will disconnect the cycle when the count of data in the FIFO for the current cycle reaches this value, and flow-through has not been achieved.

18.2 Read Prefetch Boundaries

For **memory read** and **memory read line** commands, PCI 6150 prefetches from the starting address up until the address with offset equals to the Initial Prefetch count. For example, if Initial Prefetch count equals 20H and the starting address is 10H, PCI 6150 will only prefetch 10H (20H-10H) count and afterward will start incremental prefetch until the Maximum Prefetch count is reached, or flow through is achieved. The exception is that if the Initial Prefetch Count (IPC) and starting address offset difference (IPC – Starting address offset) is less than 3 transfers apart, the PCI 6150 will not activate incremental prefetch.

For **memory read multiple** commands, if the starting address is not 0, PCI 6150 will first prefetch from the starting address up until the address with offset equals to the Initial Prefetch count. Afterwards the PCI 6150 will additionally prefetch one “Initial Prefetch count” count. For example, if Initial Prefetch count equals 20H and the starting address is 10H, PCI 6150 will first prefetch 10H (20H-10H) count and then continues to prefetch another 20H count. Afterwards, incremental prefetch is invoked until the Maximum Prefetch count is reached, or flow through is achieved.

19 IEEE 1149.1 Compatible JTAG Controller

An IEEE 1149.1 compatible Test Access Port (TAP) controller and the associated TAP pins are provided for board level continuity test and diagnostics. The TAP pins assigned are TCK, TDI, TDO, TMS and TRST#. All digital input, output, input/output pins are tested except the TAP pins and clock pin.

The IEEE 1149.1 Test Logic consists of a TAP controller, an instruction register, and a group of test data registers including Bypass, Device Identification and Boundary Scan registers. The TAP controller is a synchronous 16 state machine driven by the Test Clock (TCK) and the Test Mode Select (TMS) pins. An independent power on reset circuit is provided to ensure the machine is in RESET state at power-up.

PCI 6150 implements 3 basic instructions: BYPASS, SAMPLE/PRELOAD, and EXTEST.

20 EEPROM

Important: Wrong EEPROM data can cause the PCI 6150 to lock the system. Designers should provide an optional switch to disable the EEPROM in their board design.

PCI 6150 has an interface to EEPROM device. The interface can control an ISSI IS24C02 or compatible part, which is organized as 256X8 bits. The EEPROM is used to initialize the registers. After P_RSTIN# is deasserted, PCI 6150 will automatically load data from the EEPROM. The data structure is defined in the following section. The EEPROM interface is organized on 16-bit base in little-endian format, and PCI 6150 supplies a 7-bit EEPROM word address.

The following pins are used for the EEPROM interface:

- EEPCLK: EEPROM clock output
- EEPDATA: EEPROM bi-directional serial data pin
- EE_EN#: LOW input enables EEPROM access.

Note: The PCI 6150 does not control the EEPROM address inputs. It can only access EEPROM with address inputs set to 0.

20.1 Auto Mode EEPROM Access

Using auto mode, PCI 6150 can access the EEPROM on a WORD basis via hardware sequencer. Users need only to access a WORD data via PCI 6150 configuration registers for EEPROM START control, address, read/write command. Before each access, software should check the Auto Mode Cycle in Progress status before issuing the next START.

20.2 EEPROM Mode at Reset

Upon P_RSTIN# going high, PCI 6150 auto-loads input for EEPROM automatic load condition.

The first offset in the EEPROM contains a signature. If the signature is recognized, register auto-load will commence right after RESET. During the auto-load, PCI 6150 will read sequential words from the EEPROM and write to the appropriate registers.

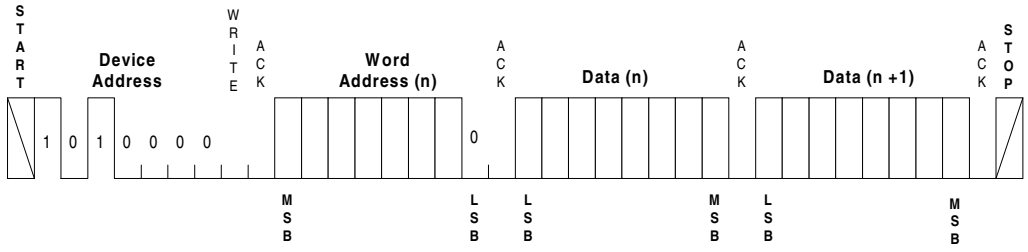
Before the PCI 6150 registers can be accessed through host, user should check the auto-load condition by reading the EEPAUTO bit. Host access is allowed only after EEPAUTO status becomes '0' which means that the autoload initialization sequence is complete.

20.3 EEPROM Data Structure

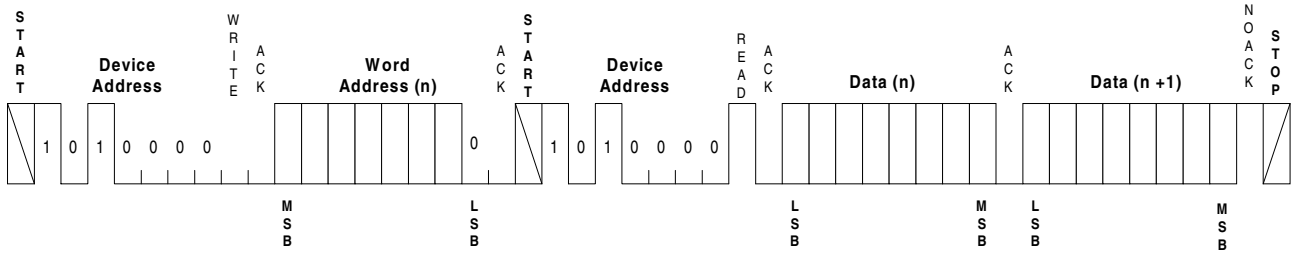
Following the reset, if the condition above is met, PCI 6150 will auto-load the registers with data from EEPROM. The following table describes the data structure used in EEPROM.

The PCI 6150 accesses the EEPROM one word at a time. It is important to note that in the data phase, bit orders are reverse of that of the address phase. PCI 6152 only supports EEPROM Device Address 0.

Write:



Read:



20.3.1 EEPROM Address and Corresponding PCI 6150 Register

EEPROM Byte Address	PCI Configuration Offset	Description
00-01h		**EEPROM signature:** Autoload will only proceed if it reads a value of 1516h on the first word loaded. 0x1516=valid signature, otherwise disable autoloading
02h		**Region Enable:** Enables or disables certain regions of the PCI configuration space from being loaded from the EEPROM. Valid combinations are: Bit 0: Reserved Bits 4-1: 0000 = stop autoload at offset 03h: Group 1 0001 = stop autoload at offset 13h: Group 2 0011 = stop autoload at offset 23h: Group 3 0111 = stop autoload at offset 27h: Group 4 1111 = autoload all EEPROM loadable registers: Group 5 Other combinations are undefined. Bits7-5: reserved
03h		**Enable Miscellaneous functions:** Bit0: **ISA Enable Control bit Write Protect**: When this bit is set, PCI 6150 will change the standard PCI-to-PCI Bridge Control Register 3Eh bit 2 into read only and ISA Enable feature will not be available. Bit1-7: Reserved
End of Group 1		
04-05h	00-01h	**Vendor ID**
06-07h	02-03h	**Device ID**
08h		**Reserved**
09h	09h	**Class Code**: Contains low byte of Class Code Register
0Ah-0Bh	0Ah-0Bh	**Class Code higher bytes**: Contains, upper bytes of Class Code Register
0Ch	0Eh	**Header Type**
0Dh	09h	Reserved
0Eh-0Fh	0Ah-0Bh	**Reserved**
10h	0Eh	**Reserved**
11h	0Fh	**BIST**
12h-13h	50h	**Internal Arbiter Control**
End of Group 2		
14h	44h	**Primary flow through control**
15h	45h	**Timeout Control**
16h-17h	46-47h	**Miscellaneous Options**
18h	48h	**Primary Initial Prefetch count**
19h	49h	**Secondary Initial Prefetch count**
1Ah	4Ah	**Primary Incremental Prefetch Count**
1Bh	4Bh	**Secondary Incremental Prefetch Count**
1Ch	4Ch	**Primary Maximum Prefetch Count**
1Dh	4Dh	**Secondary Maximum Prefetch Count**
1Eh	4Eh	**Secondary flow through control**
1Fh	E3h	**Power Management Data**
20h-21h	E0h	**Power Management CSR**
22h- 23h	DEh	**Power management Capabilities**
End of Group 3		
24-3Fh		Reserved (Must be set to 0)

21 Vital Product Data

PCI 6150 contains the Vital Product Data (VPD) registers as specified in the PCI Local Bus Specification Revision 2.2.

- The VPD information is stored in the EEPROM device along with the Autoload information.

PCI 6150 provides for storage of 192 bytes of VPD data in the EEPROM device.

- VPD related registers are located starting at offset ECh of the PCI configuration space.
- VPD also uses the enhanced capabilities port address mechanism.

22 PCI Power Management

PCI 6150 incorporates functionality that meets the requirements of the PCI Power Management Specification, Revision 1.0. These features include:

- PCI power management registers using the enhanced capabilities port (ECP) address mechanism
- Support for D0, $D3_{hot}$ and $D3_{cold}$ power management states
- Support for D0, D1, D2, $D3_{hot}$ and $D3_{cold}$ power management states for devices behind the bridge
- Support of the B2 secondary bus power state when in the $D3_{hot}$ power management state

Table 22-1 below shows the states and related actions that the PCI 6150 performs during power management transitions. (no other transactions are permitted.)

Table 22-1: States and Related Actions During Power Management Transitions

Current State	Next State	Action
D0	$D3_{cold}$	Power has been removed from the PCI 6150. A power-up reset must be performed to bring the PCI 6150 to D0.
D0	$D3_{hot}$	If enabled to do so by the BPCCE pin, the PCI 6150 will disable the secondary clocks and drive them low.
D0	D2	Unimplemented power state. The PCI 6150 will ignore the write to the power state bits (power state remains at D0).
D0	D1	Unimplemented power state. The PCI 6150 will ignore the write to the power state bits (power state remains at D0).
$D3_{hot}$	D0	The PCI 6150 enables secondary clock outputs and performs an internal chip reset. Signal S_RSTOUT# will not be asserted. All registers will be returned to the reset values and buffers will be cleared.
$D3_{hot}$	$D3_{cold}$	Power has been removed from the PCI 6150. A power-up reset must be performed to bring the PCI 6150 to D0.
$D3_{cold}$	D0	Power-up reset. The PCI 6150 performs the standard power-up reset functions.

23 Hot Swap

In order to use Hot Swap function, both EJECT_EN# and GPIO3FN# must be connected to "0".

PCI 6150 incorporates functionality that meets the requirements of the CompactPCI Hot Swap specification PICMG 2.1 R2.0 with High Availability Programming Interface level 1 (PI=1). The CompactPCI Hot Swap register block is located at PCI configuration offset E4h. Designers should refer to the CompactPCI Hot Swap specification for detailed implementation guideline.

Important Note:
If Hot Swap feature is not enabled, L_STAT inputs must be connected to logic "0". ENUM# can be left unconnected.

PCI 6150 Hot Swap register is located at E6h.

23.1 Hot Swap Signals

PCI 6150 has the following hot swap related pins:

- ENUM#: This is the output signal ENUM# to notify the system host that either a board has been freshly inserted or is about to be extracted. ENUM# is an open collector signal. ENUM# is asserted if INS or EXT bit is set and EIM is 0.
- LED: This is the status BLUE LED. LED is illuminated if RSTIN# is asserted. LED is also illuminated when the LOO bit is asserted and RSTIN# is deasserted. LED is an active HIGH signal that allows other circuit to drive the BLUE LED.
- EJECT (GPIO3): This is the Handle Switching input. This signal should be debounced by external hardware and must be connected to logic "0" if hot swap function is not used. This signal can cause the assertion of ENUM#.

23.2 Device Hiding

PCI 6150 implements Device Hiding to eliminate mid-transaction extractions. PCI 6150 invokes Device Hiding by hardware upon hot swap after P_RSTIN# becomes inactive and ejector handle is still unlocked.

PCI 6150 will be quiesced by software before Device Hiding is invoked. Current transaction will be completed as early as possible. PCI 6150 will not initiate a transaction as a master and will not respond as a target to IO transactions and will not signal interrupts.

When Device Hiding is invoked, PCI 6150 shall terminate current configuration transaction by signaling Disconnect. Following the completion of the current transaction (which is disconnected), PCI 6150 shall not respond as a target to any subsequent transactions until Device Hiding is canceled.

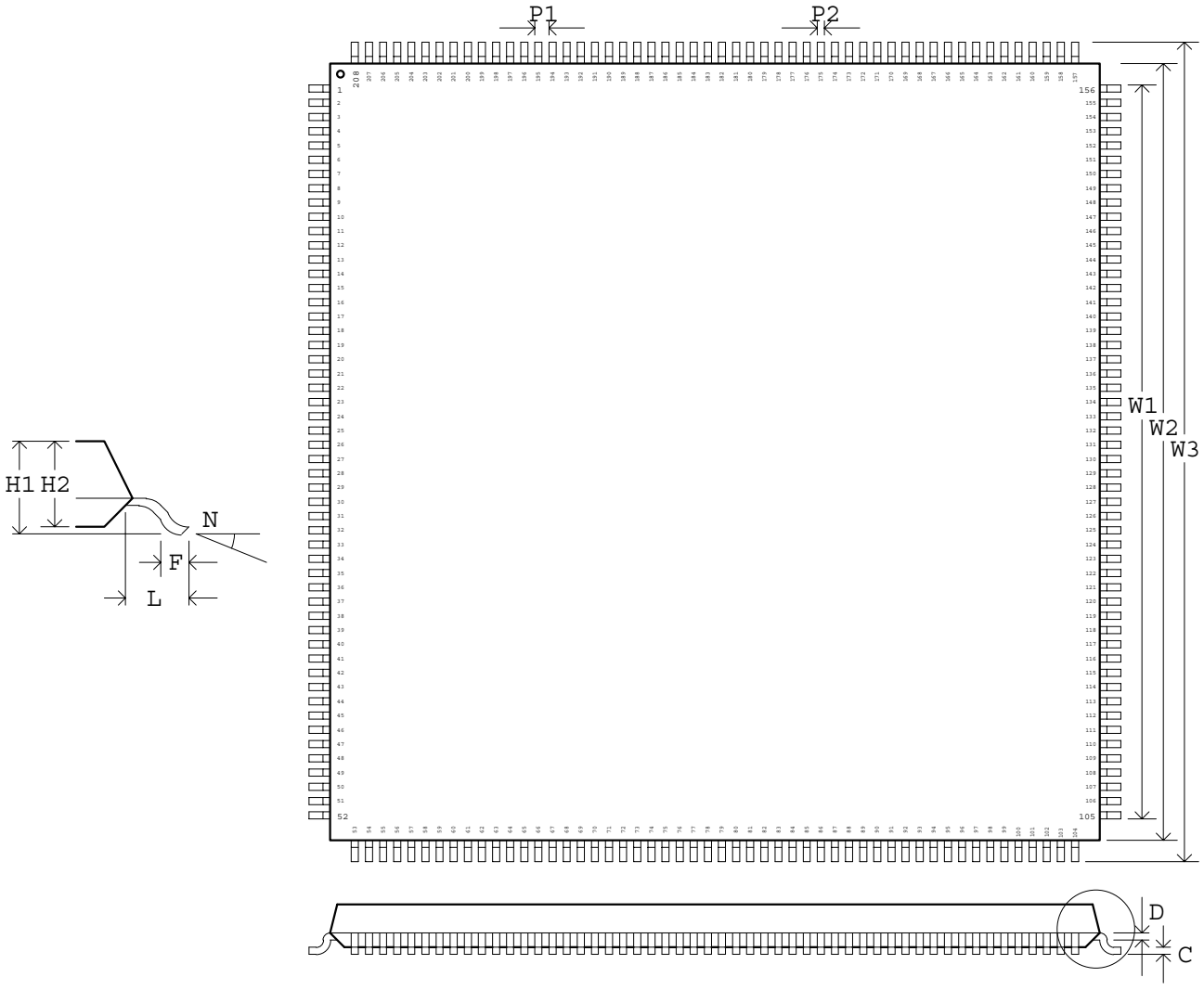
If PCI 6150 is not participating in a transaction when Device Hiding is invoked, PCI 6150 shall not respond as a target to any subsequent transactions until Device Hiding is canceled.

Device Hiding is cancelled when the handle switch is relocked.

24 Package Specifications

PCI 6150 comes in an industry standard square 208 pin Quad Flat Pack (QFP) package.

This specification outlines the mechanical dimensions for PCI 6150 chip.



The following table lists the package dimensions in millimeters.

Symbol	Dimension Name	Minimum	Nominal	Maximum
W1				
W2	Package width (length)	27.95	28.00	28.05
W3	Package overall width (length)		30.60	
P1	Lead pitch		0.50	
P2	Lead width	0.17		0.27
C	Lead thickness	0.09		0.20
D			0.13	
H1	Package overall height		4.20	
H2	Package thickness	3.17		3.95
L	Lead length		1.30	
F	Foot length	0.45	0.60	0.75
N	Foot angle	0		7

25 Electrical Specifications

25.1 Maximum Ratings

(Above which the useful life may be impaired. For user guidelines, not tested.)

Parameter	Minimum	Maximum
Storage Temperature Range	-55 °C	125 °C
Junction Temperature		125 °C
Supply Voltage, V_{DD}		3.9V
Maximum Voltage to signal pins		5.5V
Maximum Power		1.8W

Note: Stresses greater than those listed under MAXIMUM RATINGS may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

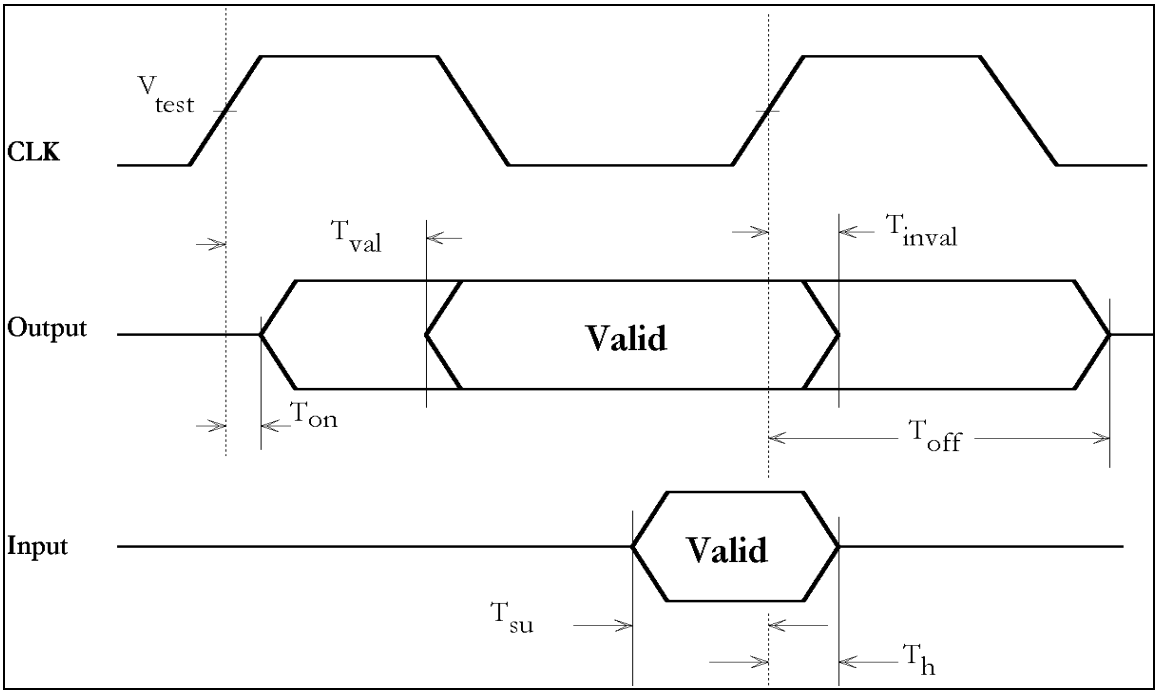
25.2 Functional Operating Range

Parameter	Minimum	Maximum
Supply Voltage	3.0 V	3.6 V
Operating ambient temperature	0 °C	70 °C

25.3 DC Electrical Characteristics

Symbol	Parameter	Condition	Min	Max	Unit	Notes
V_{DD}	Supply Voltage		3.0	3.6	V	
V_{IO}	**PVIO, SVIO** pin Interface I/O Voltage		3.0	5.5	V	
V_{ih}	Input HIGH Voltage		0.5 V_{DD}	V_{IO}	V	
V_{il}	Input LOW Voltage		-0.5	0.3 V_{DD}	V	
V_{ol}	Output LOW Voltage	I_{iout} = 1500 µA		0.1 V_{DD}	V	
V_{oh}	Output HIGH Voltage	I_{iout} = -500 µA	0.9 V_{DD}		V	
I_{il}	Input Leakage Current	$0 < V_{in} < V_{DD}$		±2	µA	
C_{in}	Input Pin Capacitance			7.0	pF	

25.4 PCI Signal Timing Specification



25.4.1 PCI Signal Timing

Symbol	Parameter	Minimum	Maximum	Unit
T_{val}	CLK to signal valid delay - bused signals	2	8	ns
$T_{val(ptp)}$	CLK to signal valid delay – point to point	2	8	ns
T_{on}	Float to active delay	2	-	ns
T_{off}	Active to float delay	-	14	ns
T_{su}	Input setup time to CLK – bused signals	3	-	ns
$T_{su(ptp)}$	Input setup time to CLK – point to point	5	-	
T_{h}	Input signal hold time from CLK	0.5	-	ns